INSTITUTO UNIVERSITARIO AERONAUTICO

DISEÑO, SIMULACIÓN E IMPLEMENTACIÓN EN FPGA DE UN FILTRO ADAPTADO CON APLICACIÓN A CONTRA CONTRA MEDIDAS ELECTRÓNICAS.

MALDONADO, Hugo
MOLINA, Luciano

Presentado ante la facultad de Ingeniería del Instituto Universitario Aeronáutico, para la obtención del título de grado “Ingeniero en Telecomunicaciones”.

Tutores:
NAGUIL, Jorge
GLOZA, Gonzalo

Tribunal de Evaluación:
SAUCHELLI, Víctor Hugo
FERREYRA, Pablo Alejandro

Diseño, Simulación e Implementación en FPGA de un Filtro Adaptado con aplicaciones a contra contra medidas electrónicas.

Instituto Universitario Aeronáutico
Córdoba, Argentina. Diciembre de 2011

RESUMEN

Frente la necesidad del grupo de trabajo encargado del Programa de Investigación y Desarrollo para la Defensa (PIDDEF), surge la propuesta de desarrollo e implementación de un algoritmo para el tratamiento de señales en tiempo real con aplicación a la Defensa Electrónica.

En los comienzos del trabajo se analizaron teóricamente los algoritmos disponibles para el tratado de señales que cumplan con los requisitos de matched filtering. Para obtener resultados prácticos se realizaron un serie de pruebas y simulaciones en MatLab® que permitieron determinar si el algoritmo en cuestión se adapta a los objetivos del proyecto. Particularmente se realizó el estudio de pre-factibilidad del algoritmo LMS y se analizó las posibilidades reales de ser aplicado a un sistema RADAR. Por otro lado se desarrolló el modelo basado en la teoría del filtro apareado con el cual se obtuvieron mejoras acordes a las metas establecidas en los objetivos específicos del trabajo.

Como segunda etapa de desarrollo se reunió información de los métodos de implementación sobre plataforma digital. Se estudiaron las placas que actualmente se encuentran a nuestra disposición determinando, según sus correspondientes hojas de especificaciones, aquella que contenga el hardware necesario para lograr los tiempos de procesamiento adecuados al sistema que se implementará en ella. Se concluyó que la plataforma Xilinx® Virtex-II es de utilidad para el trabajo.

Por último se utilizaron herramientas de software para la práctica de hardware in the loop. Particularmente System Generador, como aplicación de MatLab®, provee la lógica necesaria para la implementación física del modelo desarrollado. Se contrastaron los resultados simulados y los obtenidos de la implementación para el análisis de resultados, los cuales fueron expuestos al final de este informe.

ÍNDICE GENERAL

INTRODUCCIÓN GENERAL

El procesamiento de señales ha ocupado un lugar de privilegio dentro de cualquier tipo de sistema electrónico, aprovechando la miniaturización y el incremento de la capacidad de los elementos para el desarrollo de tecnologías funcionales en distintos campos de aplicación.

La invención del radar proporcionó una herramienta muy útil principalmente con fines militares, proveyendo a las fuerzas de un arma con gran poder táctico y de protección. Surge como actividad derivada de la incorporación de este nuevo instrumento, aquellos estudios de las señales que utiliza para prever como podría el enemigo alterar, dañar, o eludir las señales emitidas por el radar, y de esta manera acomplejar el equipamiento propio dificultándole la tarea al adversario.

Saber cuándo el enemigo envía flotas, aviones, o detectar proyectiles a gran distancia puede resultar una actividad vital en el transcurso de un pleito entre naciones. Por lo que mientras más seguro, confiable, preciso y menos vulnerable sea el instrumento que detecte estas amenazas, mayor ventaja se tendrá sobre el enemigo.

La no vulnerabilidad del equipo se logra luego del estudio de los métodos que se utilizan para interferir el buen funcionamiento de él. Generalmente estos intentos se realizan con fuentes interferentes que atentan contra la señal de interés añadiéndole una gran cantidad de ruido.

En este sentido la finalidad del presente Trabajo Final de Grado es el estudio, desarrollo e implementación en FPGA de un filtro apareado, aplicado a contra contra medidas electrónicas. Él mismo fue diseñado para evitar que la interferencia destruya completamente la información, optimizando el funcionamiento del receptor mediante la maximización de la relación señal a ruido a la salida de éste.

Al ser el alcance de este proyecto diseñar un filtro aplicado a radar y debido a la falta de instrumentación de laboratorio especializada para sistemas de radio, como generadores e interfaces de señal de radar aptos para este proyecto, analizadores lógicos, y componentes como Conversores Analógicos-Digital (ADC por sus siglas en ingles de Analog to Digital Converter) y Digital-Analogico (DAC por sus siglas en ingles de Digital to Analog Converter), solo se realizó la comprobación de su correcto funcionamiento mediante simulaciones tanto a nivel sistema como a nivel funcional en lenguaje de descripción de hardware. Además se introduce un tipo de simulación adicional de mayor precisión denominada co-simulación por hardware, donde el diseño es implementado en la FPGA, posibilitando la utilización de un software de computadoras para trabajar como si fueran generadores de señales y analizadores lógicos y de espectro. Estos procesos permiten determinar si el diseño es factible de implementar en la plataforma seleccionada.

El informe comienza con la descripción de la Guerra Informática con el objeto de acotar el campo de aplicación del radar orientando el trabajo al uso militar del mismo.

En el capítulo 1 se desarrollan los métodos utilizados en la Guerra Electrónica para la práctica de las ECM (contra medidas electrónicas) y las ECCM (contra contra medidas electrónicas).

El capítulo 2 presenta al sistema radar. Se desarrolla la teoría básica de funcionamiento, las fuentes de ruido con las que interacciona y los modelos utilizados como trasmisor y receptor.

El capítulo 4 describe el procesamiento digital, donde se incluyen los tipos de filtros posibles de utilizar, haciendo hincapié y profundizando el estudio en el filtro FIR que fue el seleccionado para este trabajo. Se incluyen las plataformas digitales presentes en el mercado en particular la Virtex II usada para la implementación del Matched Filter. Al final del capítulo 3 se analizan las herramientas de software utilizadas para la simulación, co-simulacion e implementación del filtro y los errores de cuantificación generados al digitalizar una señal.
El capítulo 4 desarrolla el Matched Filter, se estudia su teoría y se muestran las simulaciones realizadas tanto en MatLab como en Simulink.
La implementación física del modelo se realiza en el capítulo 5, respetando los pasos necesarios para optimizar el proceso completo de implementación. Se muestra paso a paso las distintas etapas por las que se debe pasar antes de la programación final de la placa.
En el capítulo 6 se analizan los resultados obtenidos, llegando a conclusiones respecto al rendimiento del filtro. También se realizan propuestas para continuar con el desarrollo del filtro y modelos de receptor con un mayor grado de calidad en función a requerimientos de las ECCM.

OBJETIVO DEL PROYECTO.

Una vez implementado el filtro apareado se corroboró su verdadera necesidad dentro de un sistema de radar. Con los resultados obtenidos y mediciones tomadas se realizaron objeciones que determinen si la incorporación del mismo en el procesamiento de señales de radar es de utilidad, primordial, o por el contrario puede prescindirse de él.
Para ello se consideraron los siguientes objetivos específicos:

- Análisis de las señales RADAR, y su interacción con el medio.
- Diseño del Matched Filter.
- Simulación del filtro diseñado.
- Selección de la arquitectura a implementar.
- Dimensionamiento de la plataforma de implementación.
- Implementación del sistema sobre plataforma digital.
- Verificación y validación.
- Conclusiones y Observaciones.

El beneficiario directo de este trabajo es el grupo encargado del proyecto PIDDEF, el cual tiene a su cargo el estudio de métodos y sistemas en el campo de la Guerra Electrónica.

CAPÍTULO 1: LA GUERRA DE INFORMACIÓN

1.1 INTRODUCCIÓN

En todo sistema de comunicación es importante mantener seguro los datos que se trasmitan. Este problema se acentúa cuando esos datos se trasmiten por el aire y llevan información que puede ser utilizada por el enemigo para determinar, por ejemplo, nuestra posición. La acción de proteger los datos y la de intentar obtener información del enemigo se conoce como guerra informática. La protección de la información no es tarea de los últimos tiempos, el continuo desarrollo y el incremento en la necesidad de protección influyen en el estudio de nuevos métodos aplicables a la defensa y al ataque.
Este tipo de guerra no solamente está aplicado al uso militar, sino también en todo aquel sistema que requiera de un alto grado de seguridad para su información. Las grandes compañías manejan información totalmente confidencial, cuya divulgación o intercepción por parte de la competencia podría influir considerablemente en el futuro, e incluso culminar con la existencia de ella.

1.2 PILARES DE LA GUERRA DE INFORMACIÓN

Los procedimientos a realizarse luego de la obtención de información del medio de propagación compartido por los agentes en conflicto, pueden estar dentro de los siguientes puntos generales [1]:

- Destrucción de los sistemas de información.
- Operaciones psicológicas.
- El engaño.
- Operativos de seguridad.
- Guerra Electrónica.

La destrucción de los sistemas de información sería el caso extremo, cuando la hostilidad ya se encuentra en marcha. Las operaciones psicológicas son usadas hace varios años. Hacer creer sobre una situación que no está sucediendo, por ejemplo, trasmisiones de radio en una región para aparentar un fuerte armamento, pueden ser tácticas muy efectivas.
Procedimientos para cuidar información útil para el enemigo, como utilizar contenedores, o mantener precauciones durante una llamada telefónica, puede ser determinante en una IW (Guerra Informática), éstos son ejemplos de Operativos de Seguridad.
La guerra electrónica es la actividad que tiene como objetivo explotar, reducir o impedir el uso de parte o la totalidad del espectro electromagnético, para obtener algún tipo de beneficio propio. Es éste el ámbito donde se desarrollan las actividades de ECM (Contra Medidas Electrónicas), y ECCM (Contra Contra Medidas

Electrónicas). Particularmente el filtro adaptado se encuentra dentro de los procedimientos de ECCM.

1.3 LA GUERRA ELECTRÓNICA

Todo equipo militar tiene a su disposición distintos sistemas que intentan lograr una posición estratégicamente superior a la del adversario, mediante el aprovisionamiento de datos. Estos datos pueden interpretarse para obtener información sobre posicionamiento, tipos de suelo, detección de blancos, velocidad de blancos, entre otros. Pero a su vez, el "atacante", puede interferir en la obtención de estos datos, al utilizar el mismo espacio radioeléctrico, y de esta forma atenuar, evitar o modificar los datos que intentamos descifrar.
Estas prácticas comenzaron en la segunda guerra mundial, junto con la invención del radar, y no han parado hasta el momento, más aún, se ha convertido en uno de los factores determinantes en la valoración de supervivencia y vulnerabilidad.
Básicamente la guerra electrónica consta de tres actividades [1].

- Soporte electrónico: Adquirir información del adversario interceptando su energía irradiada. Dentro del espacio de frecuencias en que trabaja un equipo se encuentran numerosas fuentes de interferencia, unas inevitables y otras no, por lo que hay que detectar cuál de esas fuentes forman parte de un ataque electrónico. Puede reunirse información útil con solo medir parámetros externos, como frecuencia, modulación, tasa de transmisión, etc. Luego "inteligencia de señales" se encarga de obtener más información.

- Ataque electrónico: Transmisión de señales para evitar el acceso enemigo a la información. Recordamos que la señal que quiere acceder a nuestra información puede provenir de diferentes sistemas, por lo que se deben conocer parámetros básicos para poder interferirla. Ej.: jamming (ver sección 2.9).

- Protección Electrónica: Consiste en el uso de estrategias para evitar las dos primeras actividades. El conocimiento del espacio en el que se va a tratar de transmitir la señal es cada vez más importante en la era de la guerra de información, para lograr que se trasmita de manera precisa, y oportuna. De igual forma se debe cuidar la información que se trasmite y que no sea manipulable por el enemigo. Se puede realizar la protección electrónica de muchas maneras. EMCON (Control de emisiones) es una de las más simples, en la que solo se permite trasmitir en determinados períodos de tiempo, haciendo dificultoso para el enemigo interceptar e identificar la frecuencia de trabajo. Otra manera de proveer protección es con el uso de baja probabilidad de detección de la comunicación (spread spectrum, ver el capítulo 3). Screen Jamming es otra herramienta para proteger información, colocando un jammer entre la red de comunicación y el sistema enemigo, para evitar que este último acceda a la información. También se puede realizar la encriptación de la información.

CAPÍTULO 2: EL RADAR

2.1 INTRODUCCIÓN

La palabra RADAR es la abreviatura de Radio Detection And Ranging, básicamente su funcionamiento se basa en la transmisión de energía electromagnética en un determinado espacio, por medio de ondas moduladas, con el propósito de detectar diferentes clases de objetivos. La detección se logra cuando las ondas trasmitidas impactan sobre la superficie de los objetivos que se encuentran dentro del espacio cubierto por el radar y producen reflexiones que regresan hacia él. El receptor se vale de estos ecos para determinar el rango, velocidad, posición angular y demás información sobre el objetivo que produjo su respectivo eco.

Puede clasificarse los radares según su campo de aplicación. Por ejemplo, radares militares, aeronáuticos, marítimo, meteorológico, circulación y seguridad, etc.

Otra clasificación es según la forma de onda que transmita, por ejemplo, onda continua, continua con modulación o del tipo pulsado. Este último es aquel que utiliza como señal trasmitida un tren de pulsos que modula una señal sinusoidal, y pueden identificarse tres tipos, los de baja frecuencia, media frecuencia, y alta frecuencia que utilizan la ionosfera para detectar objetos más allá del horizonte visual. De aquí en adelante trataremos con radares pulsados.

Según el modo de funcionamiento se distinguen el radar primario y el radar secundario. El primario detecta la presencia de un blanco basándose en la energía reflejada. El secundario es un radar activo que envía una serie de pulsos a la aeronave. El transpondedor responde con su identificación, velocidad, rumbo y demás datos de importancia.

El filtro que se desarrolla en este trabajo se aplica a radar primario cumpliendo con los requerimientos del Proyecto PIDDEF.

2.2 EVOLUCIÓN NACIONAL DEL RADAR

En 1947 se dejó en manos de la recién creada Aeronáutica Militar el poder de ejercer la autoridad aeronáutica para la gestión del tránsito aéreo civil. Frente a un crecimiento, tanto de la aviación comercial como de la aviación general, a fines de la década del 70 y comienzos del 80 la fuerza aérea encaro un proyecto de control del espacio aéreo, con doble función, el apoyo a la aviación civil y el apoyo a la defensa aeroespacial, denominado Sistema Integrado de Control del Espacio Aéreo (SICEA).

Este proyecto comenzó con la instalación de un sistema de control en las cercanías del aeropuerto de Ezeiza, pero debido a su costo insostenible tuvo que ser dejado de lado apenas finalizado la cuarta parte de él.

Mientras tanto, en 1987 la Fuerza Aérea realizó la primera actualización y puesta a cero de un Radar Thomson instalado en 1973 para el área terminal de Buenos Aires junto con el centro de control de Ezeiza.

La próxima incorporación se realizó en el año 1986 en la ciudad de Córdoba, instalando un radar y centro de control, adquiridos en la empresa italiana “Alenia”:

En 1992, la fuerza aérea encara un nuevo proyecto de control aéreo civil con la idea de ser financiado por los propios fondos de la fuerza, que luego de su gestión y posterior aprobación concluyo como el Plan Nacional de Radarización. Teniendo en cuenta que el costo del proyecto rondó los 400.000 millones de dólares, la misma ley que habilitó el plan, permitió al Ministerio de Defensa realizar una licitación internacional para una primera etapa del plan nacional de Radarización.
Mientras se implementaba dicho plan, en 1994 la fuerza aérea inaugura la infraestructura de un sitio Radar y centro de control en Mendoza, en este caso donado por el gobierno italiano.
En 1995 se actualizó el radar secundario del sitio de Ezeiza, convirtiéndolo en uno de última generación con la característica de ser Monopulso.
En 1996 se instala un Radar secundario monopulso con el fin de proveer información al radar de Córdoba y Ezeiza respecto al movimiento en ruta aérea de las aeronaves.
En 1997 se instala un Radar y Centro de control en el área Terminal de mar del Plata. En este mismo año la OACI habilitó los Radares monopulso para brindar servicio de aproximación de aeródromos, lo que llego a una tercera actualización del radar de Ezeiza, permitiéndole girar a 7,5 RPM. También se encara la puesta a cero de los Radares de Córdoba y Mendoza.
A fines del año 1997 se produce la apertura de las ofertas Técnicas del Plan Nacional de Radarización. En el año 1999, ya adjudicada la concesión, debió ser anulada debido a denuncias realizadas por la empresa perdedora de dicha licitación. Demoras que llevan a un nuevo plan de Radarización en el año 2001.
Se tuvo que tener en cuenta el error informático llamado “Y2K”, deficiencias de los software´s al pasar a un nuevo milenio. Esto llevo a una pequeña actualización del Radar de Ezeiza.
Mientras se decidía sobre el nuevo plan nacional, las necesidades técnicas no podían dejarse de lado, por lo que con el apoyo extra-presupuestario del Ministerio de Defensa, la Fuerza Aérea encara la total actualización del radar de Ezeiza. Actualización que no solo brinda información Radar, sino que realizaba el tratamiento automático de planes de vuelo y apoyo a la toma de decisiones en el control de flujo aéreo. Dado que este Radar se convirtió en el Centro de Gestión de Tráfico Aéreo, no solo procesaba la información proporcionada por el mismo, sino también la información proporcionada por los Radares de Mendoza, Mar del Plata, Paraná y Córdoba, y de acuerdo con la Administración de Aviación Civil de Uruguay, el Radar Carrasco.
A partir del año 2002 la fuerza aérea investiga junto con INVAP la posibilidad del desarrollo de Radares Secundarios Monopulso de última generación. Estudio que luego fue aprobado por el Ministerio de Defensa y autoridades Nacionales, posibilitando la fabricación no solo del prototipo sino también de diez radares más disponibles para la gestión de tránsito aéreo civil.
En 2003 la empresa nacional INVAP ya tenía a su cargo la tarea de mejoramiento de los servicios de control de tráfico, proporcionando un control unificado de todas las áreas terminales y aerovías del país.
En el año 2004 quedó establecido el Sistema Nacional de Vigilancia y Control Aeroespacial (SINVICA) cuyo objetivo es el control efectivo del espacio aéreo nacional para la aviación civil y la defensa. Como objetivo secundario (y muy

importante) fue dar la mayor participación de la industria argentina para su implementación.
Así nació, a cargo de INVAP, el Plan Argentino de Desarrollo de Sensores Radar. Plan que permitió la instalación del primer radar argentino 3D operativo en el corriente año, y hasta el momento 8 radares secundarios funcionando en los aeropuertos de Quilmas, Santa Rosa, Neuquén, Bariloche, Bahía Blanca, Córdoba, San Luis y Tucumán.

2.3 PRINCIPIO DE FUNCIONAMIENTO

Uno de los principales propósitos del radar primario es detectar un objeto y determinar la distancia a la cual se encuentra. Esta distancia se denomina rango y se obtiene a partir del tiempo en que tarda un pulso en ir hasta el objeto y regresar de él. Matemáticamente, el rango está dado por:

$$R = \frac{c.\Delta t}{2} \quad ; \qquad \text{R[m], } \Delta\text{t[s], c[m/s]} \qquad \text{Ec.2.3.1}$$

Dónde:
R = Rango o distancia a la que se encuentra el blanco.
c = Velocidad de la luz ($3x10^8$ m/s).
Δt = Tiempo de retardo del eco procedente del blanco detectado.

En la figura 2.3.1 se muestran los pulsos trasmitidos y los ecos que llegan al receptor luego de ser reflejados por el objetivo. En este caso se suponen pulsos ideales sin interferencias ni atenuaciones, alteraciones que en la práctica son inevitables.

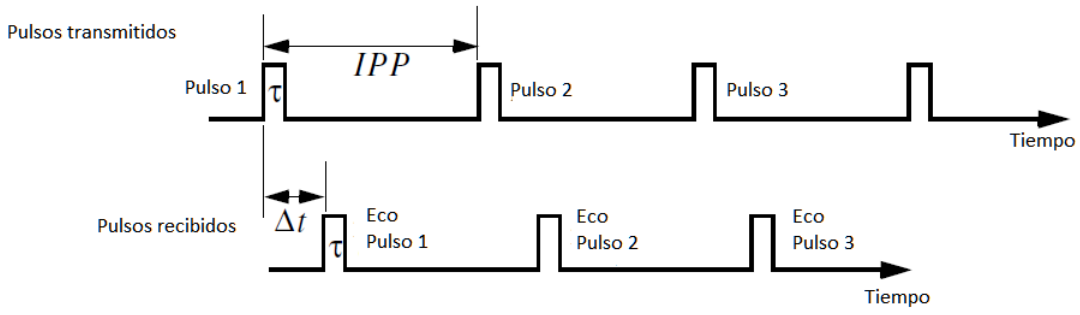

Fig.2.3.1 – Pulsos transmitidos y ecos recibidos.

A partir de la figura anterior podemos establecer los siguientes parámetros.

Período de repetición del Pulso:

$$IPP = T = \frac{1}{fr} \qquad \text{Ec.2.3.2}$$

Donde fr es la frecuencia de repetición de pulsos.

El ancho de banda del pulso está dado por:

$$B = \frac{1}{\tau} \qquad \text{Ec.2.3.3}$$

Donde τ[s] es la duración del pulso.

En cada período sólo se trasmite energía durante τ segundos, el resto del tiempo se espera el eco. Se define como Ciclo de Trabajo a la relación entre el período de repetición de pulsos y el ancho del pulso (Ecuación 2.3.4).

$$d_t = \frac{T}{\tau} \qquad \text{Ec.2.3.4}$$

La potencia promedio trasmitida es,

$$P_{av} = P_t.d_t \qquad \text{Ec.2.3.5}$$

Donde P_t es la Potencia pico transmitida.

La energía del pulso es,

$$E = P_t.\tau = P_{av}.T = \frac{P_{av}}{f_r} \qquad \text{Ec.2.3.6}$$

Al trasmitir dos pulsos consecutivos, el segundo eco recibido puede ser interpretado de dos maneras distintas. La primera alternativa es considerarlo como un segundo eco producido por el mismo objetivo que produjo el primer eco. La segunda alternativa es considerarlo como un eco del primer pulso, producido por un objetivo muy alejado. Estos casos se representan como sigue,

$$R_2 = \frac{c.\Delta t_2}{2} \qquad \text{ó} \qquad R_2 = \frac{c.(T + \Delta t_1)}{2} \qquad \text{Ec.2.3.7}$$

Dónde:

R_2 = Rango o distancia a la que se encuentra el segundo blanco.

Δt_2 = Tiempo de retardo del eco procedente del posible segundo blanco detectado.

Δt_1 = Tiempo de retardo del eco procedente del primer blanco detectado.

Este análisis puede observarse en la figura 2.3.2 a continuación.

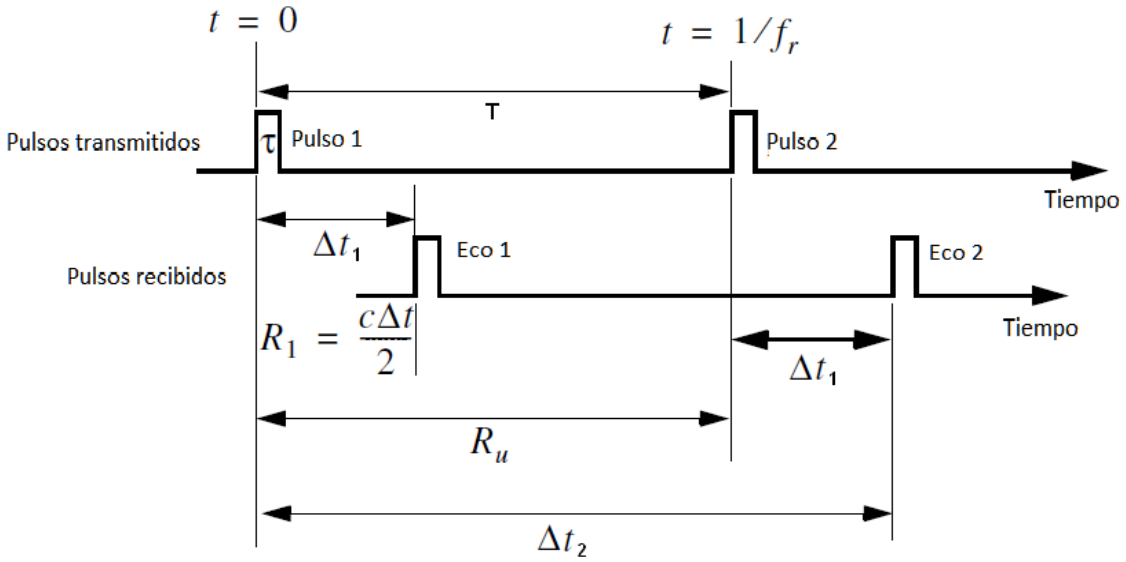

Fig.2.3.2 – Análisis e interpretación de los ecos recibidos.

El rango inequívoco es asociado el rango 2 debido a la incertidumbre que ocasiona. Entonces, una vez que el radar trasmite el primer pulso debe esperar lo suficiente como para que el eco producido por el objetivo más alejado regrese antes que se trasmita el segundo pulso. El valor de esta espera se conoce como Rango Inequívoco y se calcula como sigue:

$$R_u = \frac{c.T}{2} = \frac{c}{2.f_r} \qquad \text{Ec.2.3.8}$$

2.4 ECUACIÓN BÁSICA DEL RADAR

Si la potencia Pt es trasmitida por una antena isotrópica, la densidad de potencia a una distancia R del radar es,

$$P_D = \frac{P_t}{4\pi.R^2} \qquad \text{Ec.2.4.1}$$

Dónde P_D es la Densidad de Potencia

Al utilizar una antena que presenta ganancia en una determinada dirección (Directividad), respecto a la antena isotrópica, la ecuación 2.4.1 se modifica como sigue:

$$A_{ef} = \frac{G.\lambda^2}{4\pi} \qquad \text{Ec.2.4.2}$$

Dónde:

A_{ef} = Área efectiva de la antena.

G = Ganancia de la antena.

λ = Longitud de onda de la frecuencia de señal transmitida.

$$P_D = \frac{P_t.G_{tx}}{4\pi.R^2}$$ Ec.2.4.3

Cuando la energía radiada por la antena trasmisora incide en un objeto este refleja ondas en todas direcciones. La cantidad de energía reflejada es proporcional al tamaño del objeto, orientación, forma, tipo de material, etc. Todo estos factores se incluyen en un parámetro denominado RCS (Radar Cross Section de sus siglas en inglés) y se identifica con σ.

$$\sigma = \frac{P_r}{P_D}$$ Ec.2.4.4

Dónde P_r es la potencia reflejada por el blanco.

Por lo tanto la densidad de potencia recibida por el radar es:

$$P_{Dr} = \frac{P_t.G_{tx}.\sigma.A_{ef}}{\left((4\pi R)^2\right)^2}$$ Ec.2.4.5

A partir de la ecuación 2.4.2 se puede definir entonces que:

$$P_{Dr} = \frac{P_t.G^2{}_{tx}.\sigma.\lambda^2}{(4\pi)^3.R^4}$$ Ec.2.4.6

Para obtener el rango máximo posible, se utiliza la sensibilidad del receptor $S_{\min}$. Luego:

$$R_{\max} = \left(\frac{P_t.G^2{}_{tx}.\sigma.\lambda^2}{(4\pi)^3.S_{\min}}\right)^{1/4}$$ Ec.2.4.7

2.5 FUENTES DE RUIDO INEVITABLES

La señal trasmitida incide sobre la superficie reflectora, y el receptor capta esta pequeña porción de señal. El nivel de la señal recibida debe superar un valor umbral y el nivel de relación señal a ruido preestablecidos por el fabricante del receptor. Se interpreta como ruido a toda energía electromagnética que atenta contra la habilidad del receptor para detectar la energía deseada.
En la práctica el nivel de ruido es el factor decisivo al momento de establecer si la señal es legible o no. Si bien la intensidad de señal es un parámetro importante, en realidad el parámetro de nuestro interés y determinante en la detección de un blanco es la relación señal a ruido.
Existen diversas fuentes de ruido, a continuación se mencionan alguna de ellas:

Ruido térmico:
Se debe al movimiento de los electrones por el solo hecho de estar por encima del cero absoluto. Puede ser trasladada o irradiada como cualquier otro tipo de energía electromagnética. En el dominio del tiempo el ruido térmico presenta una distribución gaussiana. Constante para todo el rango de frecuencias.

Fuentes de ruido internas:
Es causado por el choque de los electrones dentro de los conductores. Esos choques se deben al movimiento aleatorio que presentan los electrones. La corriente eléctrica puede considerarse como un control en la dirección de ese movimiento aleatorio. Todo componente en la práctica presenta una resistencia, lo que causa la existencia de este tipo de ruido.
Cada circuito del receptor genera ruido térmico, y gran parte de éstos se suman a la señal. Todas estas fuentes de ruido están referidas a la entrada del receptor. Puede referirse todas estas fuentes de ruido a la entrada del receptor, incluyéndolas en un parámetro denominado figura de ruido (F), y luego considerar a ese receptor libre de ruido. Este ruido puede incluirse en el ruido total equivalente

$$N=K.T.B.F \qquad \text{Ec.2.5.1}$$

Dónde:

N = Ruido Térmico.
K = Constante de Boltzmann, 1.381×10^{-23} J K^{-1}.
B = Ancho de Banda de Trabajo.
F = Figura de Ruido.

La interferencia y compatibilidad electromagnética puede suceder debido a los osciladores y a los clock`s del sistema.

Fuentes de Ruido Externas:
Existen diversas fuentes de ruido externas al sistema, que varían según el rango de frecuencias que se considere. Muchas maquinas creadas por el hombre producen interferencias, por ejemplo, máquinas de soldadura, el arranque de un auto o el horno microondas. La cantidad de este tipo de ruido presente, depende del número de fuentes interferentes, por lo tanto, el ruido es mayor en regiones densamente pobladas que en zonas rurales.
La atmósfera que rodea la tierra contiene energía térmica presente en todo momento, energía que es captada por las antenas del sistema y se manifiesta en ruido térmico presente a la entrada de los equipos. (Comparada con las demás fuentes, es de menor intensidad).
El ruido galáctico es debido a la energía procedente de las estrellas. Este ruido se tiende a ser de banda ancha, conteniendo componentes en una porción significativa del espectro.

Otra fuente de ruido atmosférico son las tormentas eléctricas. Los rayos influyen mayormente en la región de HF (Altas frecuencias), por lo que puede recorrer grandes distancias.
La cantidad total de ruido adicional debido a estos tipos de fuentes se puede obtener con el uso de gráficos y/o tablas.
Matemáticamente esto puede expresarse como se describe a continuación:

$$N_{Tot} = KTBF.N_{Externo} \qquad \text{Ec.2.5.2}$$

Dónde:
N_{Tot} = Ruido total.
$N_{Externo}$ = Ruido externo debido a factores climáticos

Luego, despejando los parámetros conocidos, obtenemos que:

$$\frac{N_{Tot}}{KTB} = F.N_{Externo} \qquad \text{Ec.2.5.3}$$

$$\frac{N_{Tot}}{KTB} dB = F_{dB} + 10\log(N_{Externo}) \qquad \text{Ec.2.5.4}$$

Interferencia Cocanal – Múltiples Trayectorias:
Estas son dos interpretaciones del mismo problema, y se da cuando un sistema capta dos señales que se trasmiten a la misma frecuencia de portadora. La diferencia entre estas dos formas de interferencia se da si existe o no correlación con la señal transmitida. Es decir, si la señal interferente proviene de otra fuente, no presentará correlación, pero si la señal es la reflexión de la trasmitida, será una versión desplazada de la original, por lo que tendrá un determinado grado de correlación.
Los efectos que producen la interferencia cocanal y múltiple trayectoria, son distintos. Los múltiples caminos causan esencialmente una disminución de la señal producida por la diferencia de fase entre la señal directa y la reflejada. La interferencia cocanal produce perturbaciones aleatorias sobre la señal. Un ejemplo se da en las señales AM (Amplitud Modulada), cuando al receptor le llegan dos señales de frecuencia similar, pueden escucharse ambas al mismo tiempo. Pero generalmente si una es 6 dB mayor que la otra, no habría problemas.

2.6 RUIDO INTENCIONAL – CONTRAMEDIDAS ELECTRÓNICAS

Las contra medidas electrónicas son aquellos dispositivos que tienen como tarea engañar o burlar los sistemas de detección como el radar y sonar. Por lo tanto son fuentes de ruido de carácter intencional generado por el adversario, y poseen la dificultad de que por más que las señales interferentes sean trasmitidas por un aparato eléctrico, detrás de él se encuentra un conjunto de personas que aplican todos sus conocimientos para cumplir su objetivo.
Una de las fuentes intencionales más utilizadas y conocidas es el Jamming [3].

Mediante esta práctica su puede atacar al enemigo de dos maneras diferentes. Se puede negar el acceso a nuestra información, o se puede lograr que el adversario no pueda interpretar su información. Hay muchas maneras de hacerlo, como el camuflaje, el engaño, o la guerra. Esta última es la que ataca al adversario para negarle el transporte de información.
La técnica utilizada por el Jammer puede variar según la banda de frecuencias que se pretende afectar, el tiempo en que el jamming interfiere, o la información que pretende dañar [3]. Se explican algunas de estas técnicas a continuación.

Jamming de banda ancha:
Consiste en introducir ruido en todo el rango de frecuencias en que esté operando el sistema. Es aplicable a cualquier tipo de señal.

Jamming de banda angosta:
En este caso se trasmite energía a una determinada frecuencia para introducir ruido solo en un canal específico. La eficiencia de este método dependerá del nivel de conocimiento que se tenga de la señal a interferir.

Jamming por tonos:
Se envía uno o varios tonos dentro del ancho de banda de la señal que se trasmita. Su eficiencia depende de cómo se introduzca el tono, lo que implica un gran estudio de la señal a interferir. Por ejemplo, si se trasmite un tono que se posicione sobre la frecuencia del uno, la fase del tono trasmitido debe coincidir con el de la señal, de lo contrario no producirá grandes efectos. Es decir, se logra interferir cuando se posiciona el tono en una frecuencia que represente un símbolo, y a su vez, respete su fase.

Jamming por pulsos:
Señal interferente de banda ancha pero se trasmite solo en determinado instantes de tiempo. La eficiencia presenta características similares al Jamming de banda ancha. Permite ahorro de energía.

Jamming por barrido:
Una vez introducido el ruido en una porción del espectro se sigue con un barrido en frecuencia sobre rango en que trabaje el sistema. Se utiliza en sistemas con FHSS (Espectro ensanchado por salto de frecuencia) para localizar la nueva frecuencia luego del salto. La velocidad del barrido debe ser lo suficientemente alta como para detectar el salto, pero no demasiado como para interferir una pequeña parte de la señal.

2.7 RUIDOS PROPIOS DEL RECEPTOR

El ruido es el factor principal que limita la sensibilidad del receptor, y es por ello que se hace un estudio detallado para describirlo cuantitativamente.

Por más que se considere un medio exento de todo tipo de ruido, y un trasmisor ideal que tampoco genere ruido internamente, va a existir en él una fuente generadora de ruido térmico debido al propio movimiento de los electrones.
Para receptores del tipo superheterodino, el ancho de banda del radar es prácticamente el ancho de banda de la frecuencia intermedia [2].

$$B_n = \frac{\int_{-\infty}^{\infty} |H(f)|^2 df}{|H(f_0)|^2} \qquad \text{Ec.2.7.1}$$

Este ancho de banda no es el que comúnmente figura en la hoja de datos (a 3 dB de máxima potencia). H(f) es la respuesta en frecuencia de la etapa FI (filtro), y H(fo) es el valor máximo de la respuesta en frecuencia que generalmente es a la mitad de la banda. Cuando el valor normalizado de H(f) es igual a 1, Bn es llamado ancho de banda del ruido y es el ancho de banda de un filtro rectangular equivalente cuya potencia de ruido a la salida es igual a la del filtro con respuesta H(f).
El ancho de banda a 3 dB es muy usado y fácil de medir, pero el ancho de banda de ruido implica un conocimiento total de la respuesta H(f). Sin embargo, generalmente la respuesta en frecuencia de la mayoría de los filtros es tal que el ancho de banda a 3 dB y el ancho de banda de ruido no difieren apreciablemente.
En la práctica se ve que la potencia de ruido medida difiere al valor del ruido térmico. Esto indica que existen otras fuentes de ruido y hay que tenerlas en cuenta.
El ruido total presente a la salida de un receptor puede ser considerado como la potencia de ruido térmico generada por un receptor ideal multiplicada por la figura de ruido del mismo.

$$F_n = \frac{N_0}{K.T_0.B_n.G_a} \qquad \text{Ec.2.7.2}$$

N_0 es el ruido medido (en la parte lineal) a la salida del receptor, To es la temperatura estándar (290ºk), G_a es la ganancia (señal de salida(S_0)/señal de entrada(S_i)), B_n es el ancho de banda de la FI.
Si $K.T_0.B_n$ es el ruido a la entrada, la ecuación 2.7.2 se escribe como,

$$F_n = \frac{S_i / N_i}{S_0 / N_0} \qquad \text{Ec.2.7.3}$$

La figura de ruido se interpreta como una medida de la degradación de la relación señal a ruido cuando la señal pasa por el receptor.
Reordenando e igualando la ecuación 2.7.3 con la ecuación 2.7.2 se obtiene:

$$S_i = \frac{K.T_0.B_n.F_n.S_0}{N_0} \qquad \text{Ec.2.7.4}$$

$$S_{i-\min} = K.T_0.B_n.F_n.\left(\frac{S_0}{N_0}\right)_{\min} \qquad \text{Ec.2.7.5}$$

La menor señal de entrada se corresponde con la mínima relación señal a ruido a la salida de la etapa FI del receptor.
Esta ecuación se utiliza en la ecuación del radar (Ecuación 2.4.7) para obtener el rango máximo del radar.

2.8 PROBABILIDAD DE DETECCION, PROBABILIDAD DE FALSA ALARMA Y RELACIÓN SEÑAL A RUIDO

En esta sección se detalla la teoría estadística del ruido para obtener la relación señal a ruido presente a la salida de filtro (Amplificador de FI, Frecuencia Intermedia), necesario para el cálculo de la probabilidad de detección sin exceder una probabilidad de falsa alarma preestablecida [3].
El ruido entrante en el filtro es asumido del tipo Gaussiano (Distribución Normal), con una función de densidad de probabilidad p(v) dada por:

$$p(v) = \frac{1}{\sqrt{2.\pi.\varphi_0}}.\exp\left(\frac{-v^2}{2\varphi_0}\right) \qquad \text{Ec.2.8.1}$$

Donde ϕo es la varianza, y el valor medio de v se toma como cero.
Ahora si se considera que el ruido pasa por un filtro de banda estrecha (Comparado con la FI), la densidad de probabilidad de la envolvente del ruido a la salida es:

Ecuación de Rice: $p(v) = \frac{R}{\varphi_0}.\exp\left(-\frac{R^2}{2\varphi_0}\right)$ Ec.2.8.2

Donde R es la amplitud de la envolvente del ruido a la salida del filtro.
Esta última ecuación es una forma de la función de densidad de probabilidad de **Rayleigh.** Esta distribución modela el desvanecimiento rápido de la envolvente de la señal, y es aplicable en casos de múltiples trayectorias.
La probabilidad de que la envolvente de ruido se encuentre entre los valores v1 y v2 es,

$$P(v1 \lhd R \lhd v2) = \int_{v1}^{v2} \frac{R}{\varphi_0} \exp\left(-\frac{R^2}{2\varphi_0}\right) dR \qquad \text{Ec.2.8.3}$$

La probabilidad de que la envolvente de ruido exceda un valor V_T (valor umbral), es

$$P(v_T \lhd R) = \int_{V_T}^{\infty} \frac{R}{\varphi_0} \exp\left(-\frac{R^2}{2\varphi_0}\right) dR = \exp\left(-\frac{{V_T}^2}{2\varphi_0}\right) = P_{fa} \qquad \text{Ec.2.8.4}$$

Cada vez que el voltaje de entrada supera V_T se considera que fue detectado el objetivo, sin embargo la ecuación 2.8.4 establece la probabilidad de que el ruido supere también V_T, en otras palabras, es la probabilidad de falsa alarma. Esta probabilidad puede considerarse como la relación de tiempo en que la envolvente del ruido se encuentra sobre V_T y por debajo de V_T.

$$P_{fa} = \frac{1}{B.T_{fa}} \qquad \text{Ec.2.8.5}$$

Reemplazando en ecuación 2.8.4 se obtiene que:

$$T_{fa} = \frac{1}{B}.\exp\left(\frac{{V_T}^2}{2\varphi_0}\right) \qquad \text{Ec.2.8.6}$$

Por ejemplo, considerar un filtro con ancho de banda en FI igual a 1 MHz y un tiempo medio de falsa alarma permitido de 15 min, la probabilidad de falsa alarma es de 1.11 e-9. Usando la ecuación 2.8.4, el valor del umbral necesario para cumplir con ese tiempo es 6.45 veces el valor eficaz (rms) del ruido.
La ecuación 2.8.6 muestra una relación exponencial entre el tiempo de falsa alarma y el valor umbral. Una consecuencia de esto es que, suponiendo que la transmisión sea interrumpida y asumiendo que el tiempo medio de falsa alarma se mantiene igual, la probabilidad de falsa alarma incrementa considerablemente. Sin embargo no tiene mucha implicancia en la tensión umbral debido a esa relación entre ellas.
Hasta ahora se ha supuesto que a la entrada del filtro sólo hay ruido. Considerar ahora una señal sinusoidal de frecuencia igual a la mitad de la banda de FI, sometida a este mismo ruido. A la salida del detector de envolvente se obtendrá una función de densidad de probabilidad dada por:

$$p_s(R) = \frac{R}{\varphi_0} \exp\left(-\frac{R^2 + A^2}{2\varphi_0}\right) . I_0\left(\frac{RA}{\varphi_0}\right) \qquad \text{Ec.2.8.7}$$

Donde I_0 es la **función de Bessel** de orden cero con argumento RA/ϕo.
La ecuación 2.8.7 es llamada función de densidad de probabilidad de Rice, que es una generalización de la distribución de Rayleigh.
La distribución de Rice es utilizada para el modelado del desvanecimiento rápido de la señal en casos que exista una componente fuerte de ella, es decir un haz directo.
Ahora la probabilidad de detección es la probabilidad de que la envolvente R exceda el valor umbral.

$$P_d = \int_{VT}^{\infty} p_s(R) dR \qquad \text{Ec.2.8.8}$$

$$P_d = \int_{VT}^{\infty} \frac{R}{\varphi_0} \exp\left(-\frac{R^2 + A^2}{2\varphi_0}\right) . I_0\left(\frac{RA}{\varphi_0}\right) . dR \qquad \text{Ec.2.8.9}$$

Esta integral no puede ser valuada por métodos simples, sino que es necesario realizar expansión en series para aproximar su valor, siempre y cuando se cumple con RA/ϕo>>1, A>>|R-A |.
Otra solución es utilizar métodos gráficos como el que se muestra a continuación.
La densidad de probabilidad de solo ruido (Ecuación 2.8.2) se muestra en la figura 2.8.1 junto con la de señal más ruido (Ecuación 2.8.9) con $A/(\phi o)^{1/2}$.
Se establece un valor umbral $VT/(\phi o)^{1/2} = 3,5$.

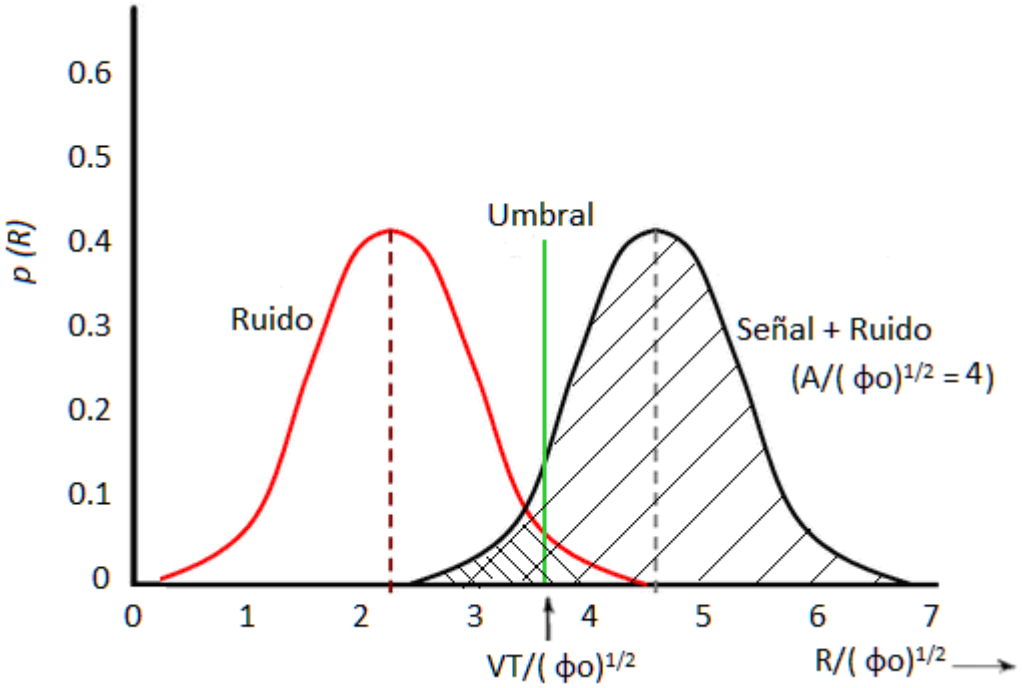

Fig.2.8.1 – Probabilidad de falsa alarma y probabilidad de detección.

El área sombreada a la derecha de $VT/(\phi o)^{1/2}$ debajo de la curva señal más ruido representa la probabilidad de detección. El área doblemente sombreada debajo de la curva de solo ruido representa la probabilidad de falsa alarma. Como dijimos previamente, en el gráfico se ve que al aumentar el nivel umbral $VT/(\phi o)^{1/2}$ la probabilidad de falsa alarma disminuye, pero también lo hace la probabilidad de detección.

La extensión en serie de la ecuación 2.8.2 se utiliza para graficar una familia de curvas que relacionan la probabilidad de detección con la amplitud de la señal.
Aunque los diseñadores del receptor trabajan con niveles de tensión, a los ingenieros del sistema radar les son más útiles las relaciones de potencia. La extensión en serie puede ser convertida a potencia reemplazando relación señal a ruido (RMS) por lo siguiente:

$$\frac{A}{\sqrt{\varphi_0}} = \frac{Señal}{Ruido_{rms}} = \frac{\sqrt{2}Señal_{rms}}{Ruido_{rms}} = \left(2\frac{Señal_{pot}}{Ruido_{pot}}\right)^{1/2} = \left(\frac{2N}{S}\right)^{1/2} \qquad \text{Ec.2.8.10}$$

También se reemplazará VT/2 ϕo por ln(1/P_{fa}) **[3]**. Usando estas relaciones se grafica en la figura 2.8.2 la probabilidad de detección en función de la relación señal a ruido, y la probabilidad de falsa alarma como parámetro.

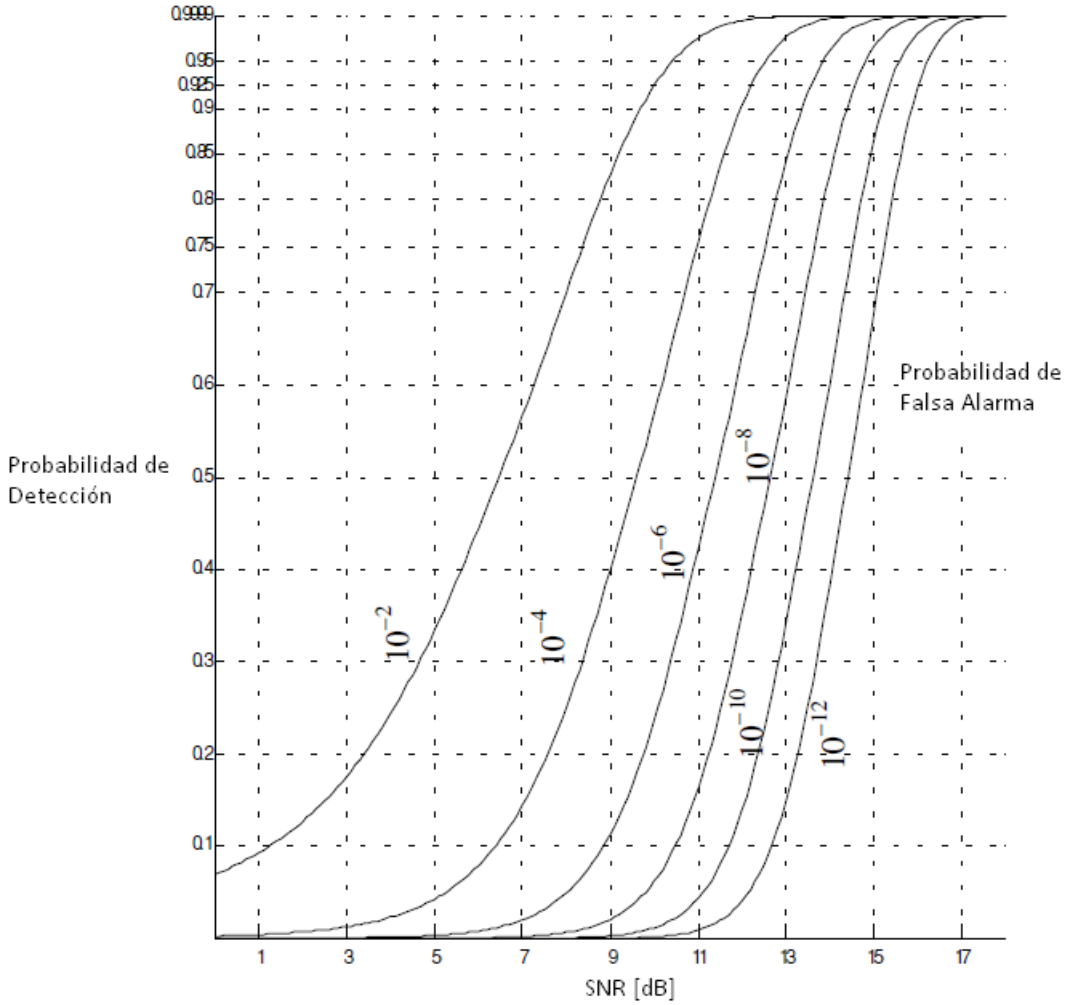

Fig.2.8.2 – Probabilidad de detección en función del SNR y la probabilidad de falsa alarma.

Tanto el tiempo de falsa alarma como la probabilidad de detección, especifican un determinado sistema.

Para utilizar el grafico primero se calcula la probabilidad de falsa alarma y luego con la ayuda del gráfico se obtiene la relación señal a ruido. Esta es la relación señal a ruido que se utiliza en la ecuación de la mínima señal detectable presentada anteriormente en la sección anterior.

Por ejemplo, si deseamos tener un tiempo de falsa alarma de 15 min y el ancho de banda de la FI es 1MHz, la probabilidad de falsa alarma es 1.1 e-9. Ingresando al gráfico se necesita una relación señal a ruido de 13,1 para una probabilidad de detección de 0,5. Una relación señal a ruido de 16,5 para una probabilidad de detección de 0,999.

Se puede concluir de la figura 2.8.2 que mientras más grande sea la señal comparada con el ruido, la detección es más probable que ocurra. Pero esto no puede decirse cuando se tiene en cuenta la probabilidad de falsa alarma. Del ejemplo desarrollado, se ve que un cambio en la relación señal a ruido de 3,4 dB, el sistema pasa de una probabilidad de detección de 0,5 a 0,999.
El cambio en el RCS es mucho más influyente en la relación señal a ruido que un cambio en la probabilidad de detección.
La relación señal a ruido requerida para la detección no es sensible al tiempo de falsa alarma.

2.9 TIPO DE SEÑALES DE RADAR.

El diseño de un trasmisor radar debe tener en cuenta una serie de puntos importantes a ser evaluados previo a la implementación del receptor.
El trasmisor debe proveer la suficiente energía como para que los ecos que lleguen al receptor permitan detectar el blanco. La forma en que ésta energía se trasmite depende del tipo de radar que se desea diseñar. Parámetros como potencia, frecuencia, ancho de banda, frecuencia de repetición de pulsos, ancho de pulsos, determinan el tipo de receptor que se deberá utilizar. Estos parámetros se deben fijar en función de la aplicación que se del sistema. En caso de una implementación real del sistema completo, se deben tener en cuenta aspectos como la dimensión de las antenas, peso de los mecanismos de funcionamiento y demás.
Existen dos grandes familias de radar que se distinguen por el tipo de señales que utilizan para su funcionamiento, el CW (Continuos Wave Radar – Radar de Onda Continua) y el PR (Pulse Radar – Radar Pulsado o de Pulsos). A partir de estas dos clases surgen distintos tipos de radares dependiendo, tal como se mencionó, de la aplicación que se le dé al mismo.
Claramente la principal diferencia entre las dos familias de radar es el modo de transmisión de la señal, uno lo hace de manera continua y el otro trasmite pulsos con determinada frecuencia de repetición. Dentro de los PR podemos encontrar radares de alta resolución, radares de seguimiento, radares de vigilancia, radares indicadores de objetivos en movimiento, radares doppler, radares de apertura sintética, entre otros. Todos ellos utilizan formas de onda pulsada a la cual se puede aplicarle algún tipo de codificación, modulación o compresión de pulsos según corresponda. Las técnicas de compresión de pulsos se describen en el capítulo 2.10.
Para el cumplimiento de los objetivos en este trabajo, se utilizaron señales de pulsos compuestos por señales senoidales y modulación lineal de frecuencia. Esta forma de onda se denomina “Chirp”.
Utilizar un chirp como señal transmitida implica realizar spread spectrum (Espectro expandido o ensanchado del espectro). El spread spectrum son técnicas que se utilizan en la transmisión de datos en radio frecuencias utilizando un espectro mayor que el que requiere el sistema para funcionar. El principal objetivo de estos métodos es coexistir de la mejor manera con la interferencia presente en el medio. También hace el sistema menos vulnerable respecto a usuarios no deseados.

Se utiliza un chirp con frecuencias comprendidas entre los 100 KHz y 9.375 MHz. No se realiza otro tipo de modulación ni se aplican métodos de codificación de la señal ya que no es aplicable a las pruebas determinantes del rendimiento realizadas en el modelo implementado.

2.10 COMPRESIÓN DE PULSOS.

Al comprimir el pulso que se va a trasmitir disminuye la potencia trasmitida lo que induce a una disminución de la relación señal ruido en el receptor. El rango de un radar se incrementa disminuyendo el ancho del pulso. Es decir, existe una relación de compromiso entre SNR y rango, ya que ambos parámetros varían según el ancho del pulso transmitido.
El objetivo de las técnicas de compresión de pulsos es lograr un ancho del pulso suficientemente grande para lograr una considerable potencia de trasmisión, pero no excederse para cumplir con requerimientos de rango.

Producto tiempo - ancho de banda

En un sistema radar implementado con un filtro apareado, la potencia de ruido blanco dentro del ancho de banda del filtro viene dada por:

$$N_i = 2\frac{N_0}{2}.B = N_0.B \qquad \text{Ec.2.10.1}$$

$N_0/2$ es la densidad espectral de potencia del ruido blanco.
El factor 2 es para tener en cuenta las componentes + y – de frecuencia.
La potencia media para un pulso de duración τ es,

$$S_i = \frac{E}{\tau} \qquad \text{Ec.2.10.2}$$

La relación señal ruido a la entrada del filtro apareado es,

$$(SNR)_i = \frac{S_i}{N_i} = \frac{E}{N_0.B.\tau} \qquad \text{Ec.2.10.3}$$

La relación entre la SNR a la salida y la SNR presente a la entrada es,

$$\frac{(SNR)_{to}}{(SNR)_i} = 2B.\tau \qquad \text{Ec.2.10.4}$$

Este resultado se refiere al producto tiempo ancho de banda para una determinada forma de onda, o su correspondiente Matched Filter.

El valor de B.τ a partir del cual la SNR de la salida es mayor que la de la entrada, se denomina ganancia de compresión. Este producto puede incrementarse más allá de la unidad utilizando modulación en frecuencia.
Se presenta nuevamente la ecuación del radar, teniendo en cuenta el ruido blanco presente a su entrada.

$$(SNR)_0 = \left(\frac{P_t.G^2{}_{tx}.\sigma.\lambda^2}{(4\pi)^3.L..k.T.B.R^4} \right) \qquad \text{Ec.2.10.5}$$

Donde L son las pérdidas totales.
Las técnicas de compresión trasmiten pulsos relativamente anchos y procesa los ecos en pulsos angostos.
Se puede lograr que el pulso este constituido por muchos sub-pulsos cuyos anchos sean el deseado para la compresión.
Partiendo de la ecuación del radar,

$$(SNR)_{\pi c} = \frac{P_t.\tau.G^2{}_{tx.}\sigma.\lambda^2}{(4\pi)^3.R^4.k.T_e.F.L} \qquad \text{Ec.2.10.6}$$

Donde $\tau = \frac{1}{B}$. Duración de

La ecuación para el pulso sin compresión puede escribirse como,

$$(SNR)_{\pi c} = \frac{P_t.(\tau = n.\tau').G^2{}_{tx.}\sigma.\lambda^2}{(4\pi)^3.R^4.k.T_e.F.L} \qquad \text{Ec.2.10.7}$$

Donde n indica el número de sub-pulsos de duración τ' que componen el pulso trasmitido.
Para un conjunto de parámetros dados de Radar, la SNR permanece invariante siempre que no cambie el ancho del pulso, independientemente del ancho de banda.
El rango se mantiene invariante mientras que la resolución del rango mejora significativamente ($\Delta R = {}^{c}\!/_{2B}$), manteniendo el ancho del pulso y aumentando el ancho de banda.

2.11 COMPRESION DE PULSO LFM**.**

Esto se logra añadiendo modulación en frecuencia a un pulso largo que se trasmitirá, y usando un receptor con filtro apareado que permite la compresión del pulso. Como resultado, la salida del filtro se ve comprimida un factor $\xi = B.\tau'$. Entonces, utilizando pulsos largos y modulación LFM de banda ancha, grandes relaciones de compresión pueden ser logradas. En la figura 2.11.1 se muestra el proceso ideal de compresión.

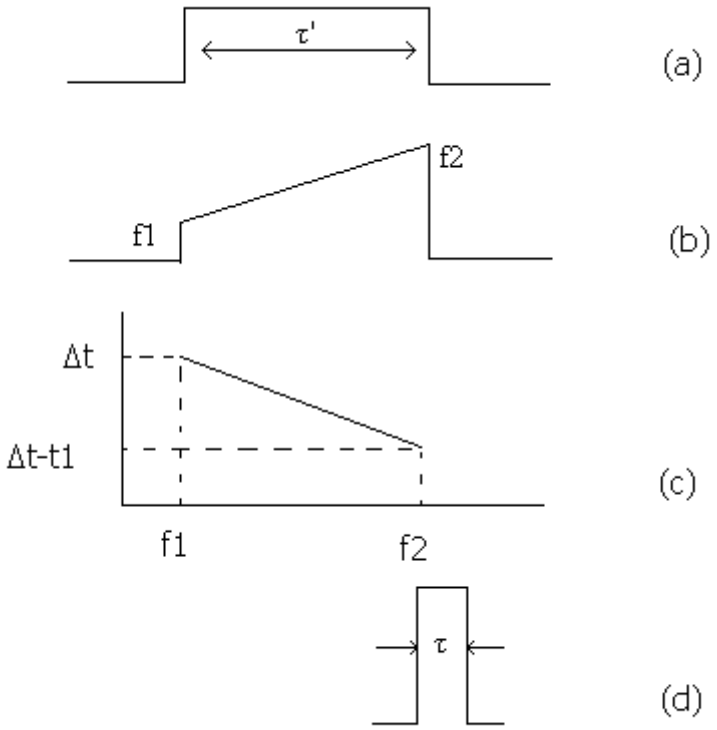

Fig.2.11.1 – Proceso de compresión de pulso

La parte (a) muestra el pulso trasmitido, en (b) se ve la modulación en frecuencia, (c) muestra la característica que presenta el filtro respecto al retardo en el tiempo, y (d) es el pulso comprimido.
Para generar un pulso "chirp" con compresión lineal, la frecuencia justamente presenta relación lineal con el tiempo. La fase es función cuadrática.
La ecuación general que representa una con LFM es,

$$LFM = k.\cos\left(\pi.\frac{B}{T}.t^2 + \varphi\right) \qquad \text{Ec.2.11.1}$$

Un sistema diseñado con filtro apareado y que utilice señales chirp reconocerá los elementos transmitidos, retrasará los resultados y mostrará un pulso comprimido con amplitud proporcional al eco recibido.
Para cumplir con objetivos de resolución es necesario aplicar métodos de ventaneo para reducir la influencia de lóbulos laterales.

2.12 MODELO DE TRANSMISOR

Según se describió en capítulos anteriores, sobre compresión de pulsos y LFM, se diseñó un modelo de trasmisor de radar pulsado con señal chirp, de 3.2 µs de duración y 9.375 MHZ de ancho de banda, cuyo diagrama en bloques se muestra en la figura 2.12.1.

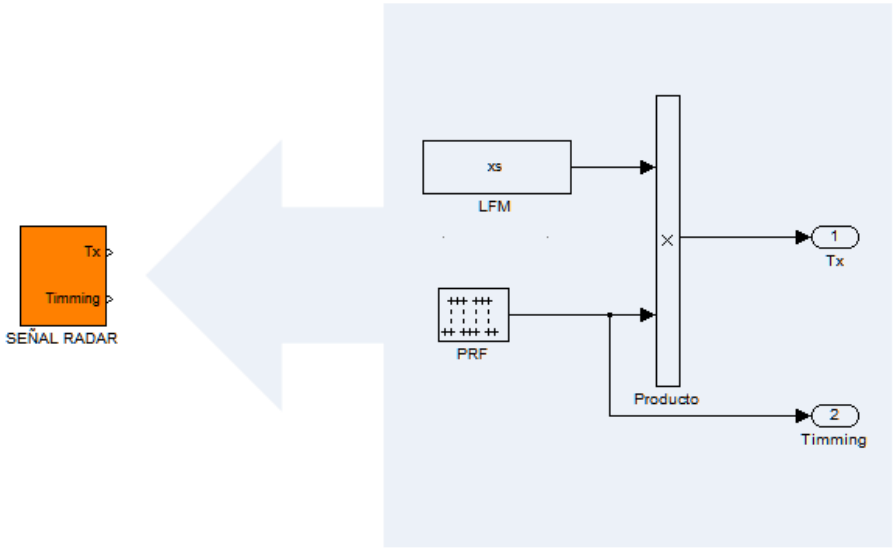

Fig.2.12.1 – Modelo de transmisor radar en Matlab/Simulink

La siguiente figura ilustra la señal chirp generada por el transmisor de radar observado en la figura anterior.

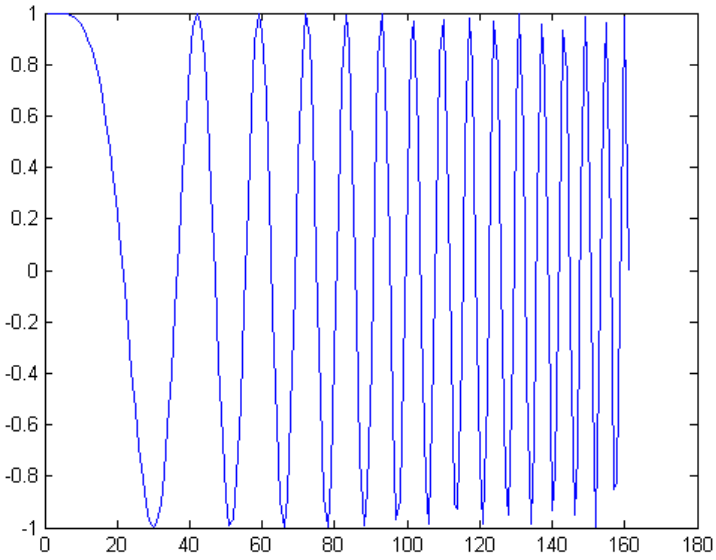

Fig.2.12.2 – Pulso Chirp utilizado en el modelo de transmisor.

Considerando este modelo básico de trasmisor, fácilmente pueden lograrse las formas de onda necesarias para realizar pruebas determinantes del rendimiento del filtro.
Como se mencionó, no se agregan bloques de codificación o modulación extra.

2.13 TECNICAS DE VENTANEO.

Las técnicas de ventaneo o Windowing, limitan las señales a un número finito de muestras, lo que producen alteraciones en las componentes espectrales de las mismas. Una señal que muestra un impulso en el espectro, significa que es una señal continua en el tiempo y de única frecuencia. Al aplicarle algún tipo de ventana la amplitud del pulso disminuye considerablemente y surgen componentes espurias distribuidas en dicho espectro [4]. Esto produce dificultades para la detección de señales cuyas frecuencias sean próximas pero, en un principio, no es algo que influya significativamente en el tratamiento temporal de señales.
La amplitud de un determinado eco depende de las características, dimensión y distancia respecto al receptor, del objetivo correspondiente. Por lo tanto pueden darse casos en que el eco de un objetivo de gran porte o a menor distancia oculte el eco de otro objetivo.
Consideramos al ventaneo como una multiplicación,

$$x(n) = h(n).v(n) \qquad \text{Ec.2.13.1}$$

Donde h(n) son los coeficientes del filtro, v(n) representa la ventana cuyas características dependen del tipo elegido, y x es la señal obtenida luego de aplicada la técnica de ventaneo.
Los tipos de ventanas clásicos utilizados para este fin son la de Hamming, Hanning, Triangular, Gaussiana, Kaiser, Bessel, Blackman, Harris, entre otras. Cada una de ellas con características aplicables a determinados sistemas [5].
Se utilizó aquellas ventanas con las que se obtuvieron mejores resultados. Se presenta la ventana de Hamming en la siguiente ecuación:

$$v(n) = 0{,}53836 - 0{,}46164.\cos\left(\frac{2.\pi.n}{N-1}\right) \qquad \text{Ec.2.13.2}$$

Donde N es el tamaño de la ventana. La figura 2.13.1 ilustra la ventana de Hamming para un N = 161.

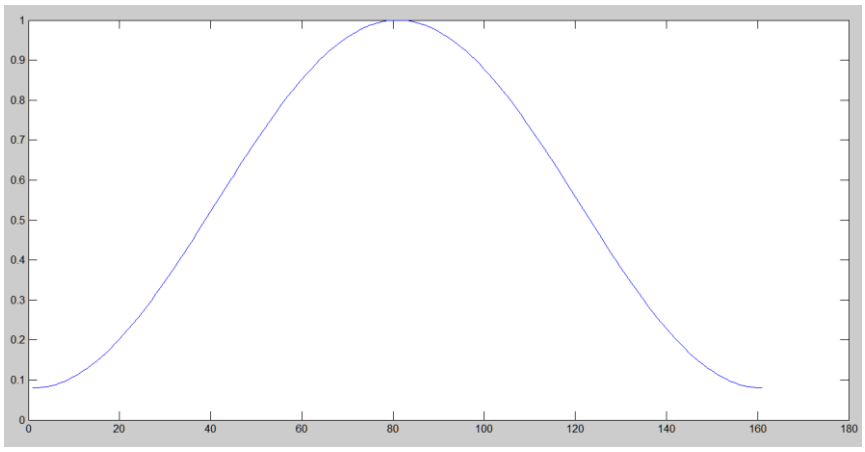

Fig.2.13.1 – Ventana de Hamming para N=161

En el capítulo 4 se describen las pruebas del filtro apareado aplicando las técnicas de ventaneo.

2.14 EL RECEPTOR RADAR

La función del receptor es detectar la señal deseada aun cuando ésta se encuentra interferida debido a diferentes fuentes de ruido, como lo son el jamming, ruidos ambientales o por clutter`s. Es necesario separar la información útil, amplificarla hasta un determinado nivel necesario para ser procesada y por último ser entregada al usuario.

Un receptor no sólo se diseña teniendo en cuenta el tipo de señal que va a detectar, sino también la naturaleza del ruido, interferencia y los ecos de clutter con los que la señal deseada tendrá que competir.

El ruido puede penetrar al sistema a través de las antenas, o puede ser generado por el mismo receptor internamente. Generalmente utilizando microondas el ruido inevitable que entra por las antenas es menor que el del propio receptor, por lo que la sensibilidad de él se determina teniendo en cuenta esta última fuente de ruido.

Dependiendo del campo de aplicación del radar, se deberán realizar algunas propiedades y sacrificar otras capacidades para lograr el mejor funcionamiento posible para el que será diseñado. Los parámetros a tener en cuenta son ganancia, fase, estabilidad, rango dinámico, sintonización, confianza, estabilidad, entre otros.

El receptor más utilizado es el superheterodino, debido a su buena sensibilidad, alta ganancia, selectividad y fiabilidad. Los factores determinantes de la sensibilidad del radar son la figura de ruido y la relación frente espalda (front-end en literaturas de habla inglesa).

Un buen diseño del receptor se debe centrar en maximizar la relación señal ruido a su salida y es aquí donde el algoritmo de Matched Filtering aplica sus capacidades para lograrlo. El filtro se incorpora en un módulo intermedio del receptor donde la señal ya pasó por un previo filtrado para acotar el ancho de banda de la señal de entrada, y en una instancia previa a la entrega de la información al usuario.

Para el desarrollo del modelo final del filtro aplicable al receptor radar se realizó una serie de estudios y pruebas previas los cuales sirvieron como experiencia necesaria para seleccionar el algoritmo y la estructura adecuada para el cumplimiento de los requerimientos de filtrado.
Se hace mención especial al algoritmo de filtrado adaptivo LMS [18][19], el cual se desarrolló en una primera instancia de aprendizaje durante este proyecto. La teoría de este filtro y las simulaciones realizadas se encuentran en el **Anexo A**, ya que no se consideró relevante incorporarlo en el cuerpo del informe, basados en los resultados y mediciones obtenidos.
Luego de esta etapa de aprendizaje y de numerosas pruebas, se desarrolló el filtro apareado descripto en los capítulos posteriores.

CAPÍTULO 3: PROCESAMIENTO DIGITAL

3.1 INTRODUCCIÓN

El comienzo de la era de las comunicaciones surge con la invención del transistor. Dispositivo electrónico que dio lugar a los circuitos integradas de cada vez mayor escala de integración.

Como resultado de estos avances, los sistemas limitados en capacidad, rendimiento y eficiencia crecen a pasos agigantados, provocando una revolución en la forma de comunicación.

Cada vez se requieren mayores anchos de banda para traficar un gran volumen de información y todo esto en menores tiempos de procesamiento, implicando elevar las frecuencias de funcionamiento de los dispositivos y optimizar los recursos, provocando el calentamiento de los mismos hasta el límite de su destrucción.

La capacidad de representar un suceso de la naturaleza mediante una combinación de números binarios ofrece un gran poder de manipulación y transformación de estas señales y prácticamente realizar con ellas lo que se desee. Pero esta conversión analógica/digital tiene su precio, se incorporan errores de cuantificación y redondeo tanto al comienzo del procesamiento como al momento de entregar los datos al usuario para su evaluación, obteniendo como resultado una aproximación excelente (pero no exacta) a la realidad.

El procesamiento digital se utiliza en una gran cantidad de sistemas con propósitos y objetivos deferentes. Utilizando en cada caso plataformas y hardware acorde a la aplicación que se le dé.

En este capítulo se incorporan las estructuras existentes de filtro digital analizando ventajas y desventajas de cada uno de ellos. Seguido de las plataformas disponibles y la utilizada para este proyecto. Al final se dan a conocer las herramientas tratadas para realizar las simulaciones e implementación del modelo de filtro apareado.

3.2 FILTROS DIGITALES

En el ámbito del procesamiento analógico o digital de señales, el filtrado es una parte muy importante del sistema, ya sea para reducir el ancho de banda, eliminar componentes de frecuencia, restaurar o separar una señal.

El filtro analógico se caracteriza por tener un menor coste, mayor rapidez y un gran rango dinámico, pero el filtro digital ofrece otras ventajas, como por ejemplo sufren menor interferencia electromagnética, son estables frente a cambios de temperatura y al paso del tiempo, poseen alta precisión y por sobre todo son **programables**, lo que les da una gran versatilidad al momento de tratar con señales compuestas o variantes.

Un filtro digital manipula señales en tiempo discreto en base a una función de filtrado. Esta función es implementada mediante multiplicadores, sumadores y retardadores que combinan la señal de entrada con los coeficientes del filtro para

producir la señal de salida. La función de transferencia H(z) de un filtro digital es [14]:

$$H(z) = \frac{\sum_{k=0}^{M} b_k . z^{-k}}{\sum_{k=0}^{N} a_k . z^{-k}} \qquad \text{Ec.3.2.1}$$

Los coeficientes a_k y b_k son los que determinan el tipo respuesta del filtro.
Como se mencionó, es un sistema en tiempo discreto, por lo que en caso de tener que filtrar una señal analógica, se debe proveer al sistema con conversores A/D y D/A, siempre respetando el teorema del muestreo (ver sección 3.7). En la figura 3.2.1 se muestra un esquema básico del filtrado digital. Al conjunto de sistemas utilizados para realizar el tratamiento de una señal digital se lo denomina generalmente DSP, acrónimo que significa Digital Signal Processing.

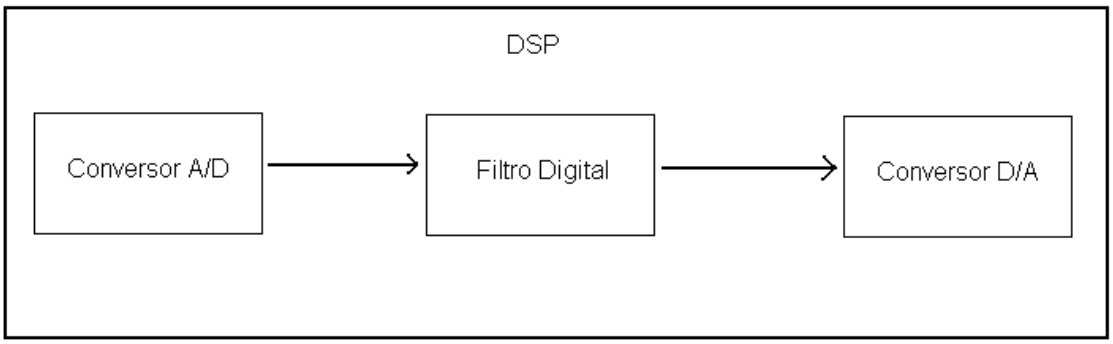

Fig.3.2.1 – Esquema básico de filtrado digital.

En el bloque “Filtro Digital” pueden incorporarse dos grandes familias de filtros, los FIR y los IIR.

Un filtro FIR (Respuesta Finita al Impulso) se caracteriza por tener respuesta finita al impulso, no tener retroalimentación y poseer todos los polos en el origen.
La primera propiedad significa que frente a una señal impulsiva de entrada el filtro responderá con una cantidad finita de términos no nulos. No posee retroalimentación ya que la salida solo se basa en el estado actual de las entradas del filtro. Y los polos en cero implican estabilidad.
A partir de la ecuación 3.2.1 y basados en sus características mencionadas, se llega a la función de transferencia del filtro FIR, dada por:

$$H(z) = \sum_{k=0}^{M} b_k . z^{-k} \qquad \text{Ec.3.2.2}$$

La estructura de implementación de este tipo de filtros se muestra en la figura 3.2.2 a continuación:

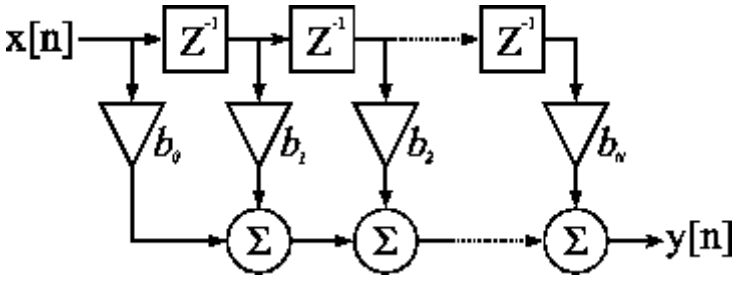

Fig.3.2.2 – Estructura de implementación de filtros FIR.

En la práctica, los filtros FIR se emplean en problemas de filtrado en los que se precisa una característica de fase lineal dentro de la banda de paso del filtro. A su vez, esta característica de fase lineal de los filtros FIR conlleva a un coste de mayor consumo de procesamiento, debido a la necesidad de mayor cantidad de coeficientes para lograr respuestas similares a un filtro IIR.

Los filtros IIR (Respuesta Infinita al Impulso) como su nombre lo indica poseen respuesta infinita frente a un impulso a la entrada, es decir, nunca vuelve al reposo. En este caso la salida del filtro depende tanto de las entradas presentes y pasadas como de las salidas en instantes anteriores. La función de transferencia H(z) de estos tipos de filtros es:

$$H(z) = \frac{\sum_{k=0}^{M} b_k . z^{-k}}{1 + \sum_{k=1}^{N} a_k . z^{-k}} \qquad \text{Ec.3.2.3}$$

Y su estructura de implementación es la siguiente:

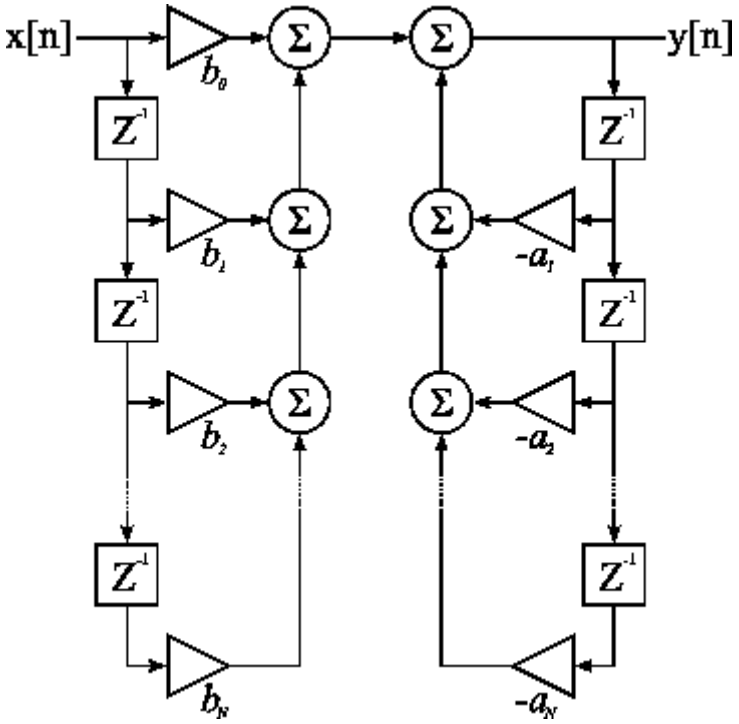

Fig.3.2.3 - Estructura de implementación de filtros IIR.

La ubicación de los polos y ceros determinarán la estabilidad y causalidad del sistema. Es normal pensar que este tipo de filtros requiera de un mayor espacio en memoria debido al mayor número de componentes para su realización, pero dado un determinado objetivo de filtrado, el filtro IIR posee un orden significativamente inferior comparado a un filtro FIR que cumpla con dichos objetivos.
Generalmente, los filtros IIR producen distorsiones en la señal procesada debido a sus características de fase no lineal en función de la frecuencia. Sin embargo, por regla general, un filtro IIR tiene lóbulos secundarios más pequeños en la banda de rechazo que un filtro FIR con el mismo número de coeficientes. Por esta razón, si es tolerable cierta distorsión, es preferible un filtro IIR, principalmente porque su implementación precisa muy pocos coeficientes, requiere menos memoria y presenta menos complejidad de cálculo.
El tipo de filtro que se emplee, ya sea FIR o IIR, depende de la naturaleza de las señales a tratar. En este proyecto en particular, se optó por la implementación de filtros FIR garantizando que todas las señales que se procesen sean tratadas por un filtro con características de fase lineal, evitando cualquier problema de distorsión que pueda ocurrir.

3.3 FILTROS FIR

Como se mencionó anteriormente, los filtros FIR tienen repuesta finita al impulso, es decir que su salida frente a este estímulo es igual a un número finito de términos no nulos. La función de transferencia de estos filtros es la expresada en la ecuación 3.2.2 y debido a que sus polos están en el origen, estos filtros son siempre estables.
Si se considera al sistema causal, entonces la expresión siguiente caracteriza a un filtro FIR de coeficientes b_K en el dominio del tiempo:

$$h[n] = \sum_{k=0}^{M-1} b_k . \delta[n-k] \qquad \text{Ec.3.3.1}$$

Donde δ[n-k] representa el impulso unitario discreto y M es el número de coeficientes del filtro.
Alternativamente podemos expresar la salida y(n) como la suma de convolución de la respuesta impulsiva del sistema h(n) con la señal de entrada x(n).

$$y[n] = \sum_{k=0}^{M-1} h[k].x[n-k] = \sum_{k=0}^{M-1} b_k . x[n-k] \qquad \text{Ec.3.3.2}$$

Las raíces de este último polinomio, como se puede observar, constituyen los ceros del filtro (donde la respuesta de esta función se hace cero).
Los filtros FIR son muy fáciles de realizar. La mayoría de los procesadores de señales digitales tienen unas arquitecturas internas que hacen factible su construcción.

ARQUITECTURA DE FILTROS FIR

Para su implementación en plataforma digital pueden utilizarse tres tipos de arquitecturas: algoritmo de Aritmética Distribuida y algoritmo de Multiplicadores Acumuladores en forma Directa y en forma Transpuesta [14].
El empleo de la aritmética distribuida (DA) para el procesado digital está justificado por su eficiencia de cálculo. La ventaja principal de la DA es su eficiencia de mecanización y la disminución de recursos internos que origina su utilización. Debido a que se trata de un proceso de naturaleza serie la desventaja que presenta es su relativa lentitud.
La aritmética distribuida (DA) es un método alternativo de realizar operaciones aritméticas que involucren la suma y el producto, en implementaciones software o hardware [7].
Dentro de las ventajas de la aritmética distribuida se encuentran su fácil y eficiente proceso de implementación, además de la reducción del número de compuertas usadas. La desventaja que se le puede adjudicar a la DA es su baja velocidad de respuesta, debido a su naturaleza serial; pero ésta se puede convertir en ventaja cuando se emplean diversas técnicas para mejorarla como: fraccionamiento de los datos de entrada y/o operación en forma totalmente paralela [7] [8].
La DA se ha usado principalmente en el procesamiento digital de señales cuando es requerida una suma de productos como la mostrada en la ecuación 3.3.3 Específicamente puede emplearse en el diseño e implementación de filtros digitales o en estructuras de diversas transformadas en el dominio de la frecuencia [7]. También es útil en sistemas de comunicaciones y en sistemas de control.
En la ecuación 3.3.3 se muestra una suma de productos, que define la respuesta de un sistema lineal e invariante en el tiempo. Donde y es la respuesta del sistema, X_K es la variable de entrada y A_K es un arreglo de coeficientes constantes.

$$y(n) = \sum_{k=1}^{k} A_k . x_k(n) \qquad \text{Ec.3.3.3}$$

Si X_K es un número binario en complemento dos escalado, entonces X_K se puede expresar así:

$$x_k(n) = -b_{k0} + \sum_{n=1}^{N-1} b_{kn} . 2^{-n} \qquad \text{Ec.3.3.4}$$

Dónde b_{Kn} son bits (0 ó 1), b_{K0} es el bit de signo y $b_{K\,N-1}$ es el bit menos significativo.
Al reemplazar la ecuación 3.3.4 en la 3.3.3 y reordenando las sumatorias se llega a:

$$y(n) = \sum_{n=1}^{N-1} \left[\sum_{k=1}^{k} A_k . b_{kn} \right] . 2^{-n} + \sum_{k=1}^{k} A_k . - b_{k0} \qquad \text{Ec.3.3.5}$$

Explícitamente, la ecuación 3.3.5 puede darse en términos de los todos los productos y sumas parciales de la siguiente forma:

$$y = -\left[x_{10}.A_1 + x_{20}.A_2 + \ldots + x_{k0}.A_k\right] + \left[x_{11}.A_1 + x_{21}.A_2 + \ldots + x_{k1}.A_k\right]2^{-1}$$
$$+ \ldots\ldots + \left[x_{1(B-2)}.A_1 + x_{2(B-2)}.A_2 + \ldots + x_{k(B-2)}.A_k\right]2^{-(B-2)} \qquad \text{Ec.3.3.6}$$
$$+ \left[x_{1(B-1)}.A_1 + x_{2(B-1)}.A_2 + \ldots + x_{k(B-1)}.A_k\right]2^{-(B-1)}$$

Si es observado el término en los corchetes de la ecuación 3.3.5, esta expresión posee 2^K posibles valores como lo demuestran los términos en los corchetes de la ecuación 3.3.6. Luego se pueden calcular estos valores y almacenarlos en una memoria o DALUT (Distributed Arithmetic Look Up Table por sus siglas en inglés); con lo cual se evita el cálculo de los productos parciales en tiempo real. Los datos de entrada son usados para direccionar la DALUT, y la salida es guardada en uno o varios registros para ser sumados usando desplazamientos denotados por los factores exponenciales en la ecuación 3.3.6 y obtener de esta manera el resultado final. Como ejemplo del contenido de una DALUT es mostrada la Figura 3.3.1.

0
A1
A2
A1 + A2
A3
A1 + A3
A2 + A3
A1 + A2 + A3

Fig 3.3.1 - Contenido de una DALUT con K=3.

Otro algoritmo de implementación del filtro FIR, se denomina MAC (Multiply Accumulate). Éste implica que se realiza el computo de cada producto entre los b_K y la señal x(n) con su correspondiente retraso, donde cada uno de esos resultados parciales se va acumulando hasta realizarse los M productos necesarios para obtener una muestra de señal de salida y(n). Para la obtención de cada muestra y(n), se necesitan M multiplicaciones y M - 1 sumas de forma recursiva.
Una visión simplificada de un FIR MAC basada en la utilización de un solo motor MAC se muestra en la figura 3.3.2.

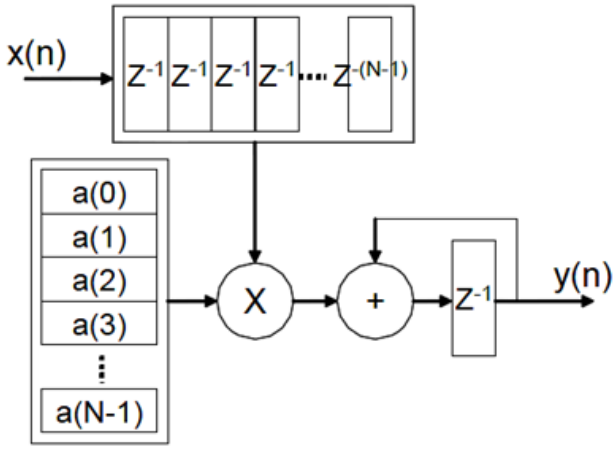

Fig.3.3.2 – Diagrama en bloques de un FIR con MAC simple.

La puesta en práctica de extensiones Multi-MAC sólo es requerida para implementaciones de filtros de altos requerimientos (un mayor número de coeficientes, tasas más altas de muestreo, más canales, etc.)
El número de multiplicadores necesarios para aplicar un filtro se determina calculando el número de multiplicaciones necesarias para realizar el cálculo (teniendo en cuenta la simetría de la estructura de coeficiente y los cambios de frecuencia de muestreo) y dividiendo por el número de clock´s disponibles para procesar cada entrada de una muestra.
El número de clock´s necesarios siempre se redondea hacia abajo y el número de multiplicadores se redondea al entero más cercano. Si al realizar el cálculo F_{CLK}/F_S*Número de Canales hay un resto distinto de cero, algunos de los motores MAC hará el cálculo con menos coeficientes que los otros, y los coeficientes de éste se rellenan con ceros para evitar respuestas erróneas del filtros.
En la implementación del filtro se genera automáticamente una aplicación que cumple con los requisitos de desempeño definidos por el usuario basado en la velocidad del clock del sistema, la frecuencia de muestreo, el número de canales, y la tasa de cambio.
En la figura 3.3.3 se muestra una estructura MAC de forma Directa.

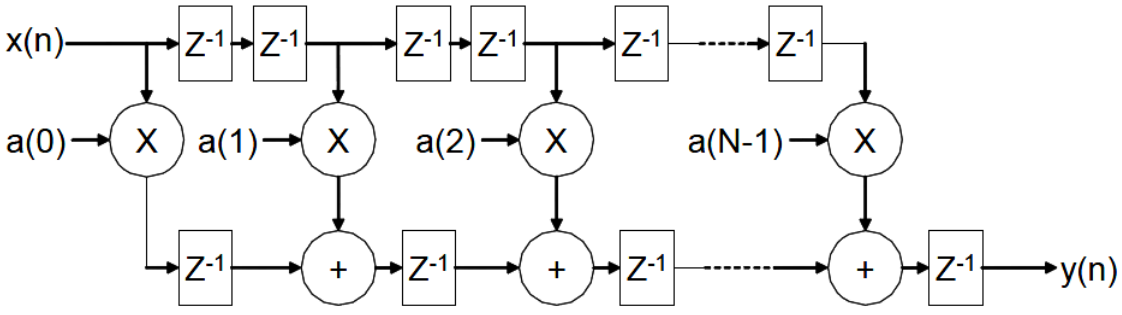

Fig.3.3.3 - Estructura Directa de un Filtro FIR.

La Figura 3.3.4 muestra una implementación Multi-MAC para esta arquitectura.

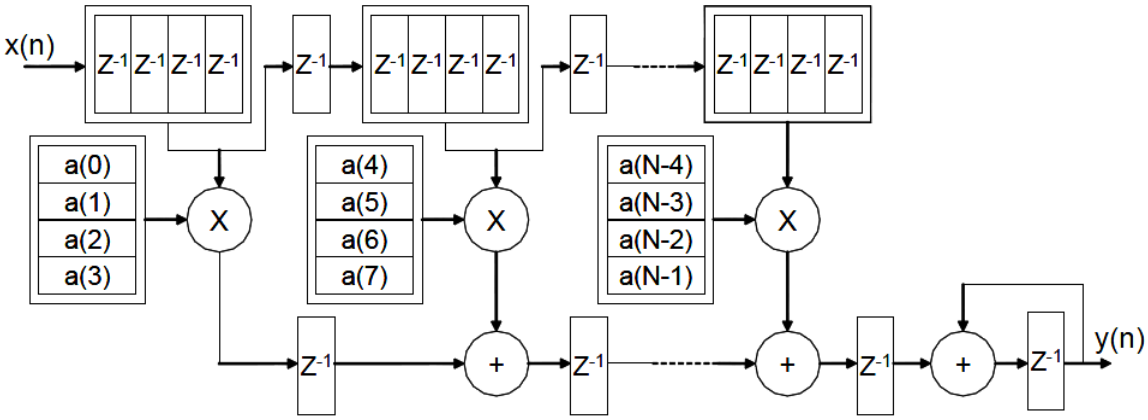

Fig.3.3.4 - Estructura Directa de un filtro FIR Multi-MAC.

La arquitectura directa permite implementaciones con uso eficiente de área (Consumo de recursos) y alto desempeño del filtro. Esta estructura se aplica también para explotar la simetría de los coeficientes ofreciendo ahorro de recursos.

Otro tipo de arquitectura común en filtros FIR es la llamada estructura Transpuesta, la cual puede derivarse de la estructura Directa sin más que modificar la posición de los elementos de retardo tal como se ilustra en la figura 3.3.5.
La ventaja de cambiar el orden de dichos elementos permite aprovechar la técnica de pipelining. El pipelining es una técnica en diseño computacional y electrónico, que permite descomponer un proceso en múltiples pasos intermedios, siendo la salida de uno la entrada del próximo paso, interponiendo registros en las entradas y salidas de dichos pasos intermedios. Esto permite trabajar a velocidades de clock mayores, lo que da como resultado una velocidad mayor de procesamiento final en el proceso completo.

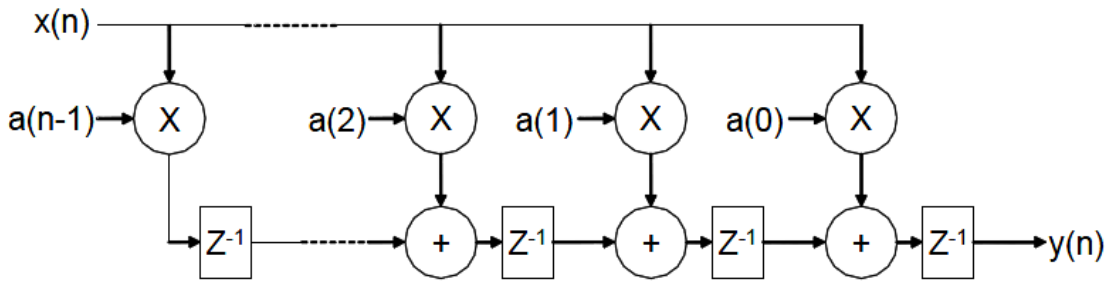

Fig.3.3.5 - Estructura Transpuesta de un filtro FIR.

La figura 3.3.6 muestra una implementación Multi-MAC para esta arquitectura.

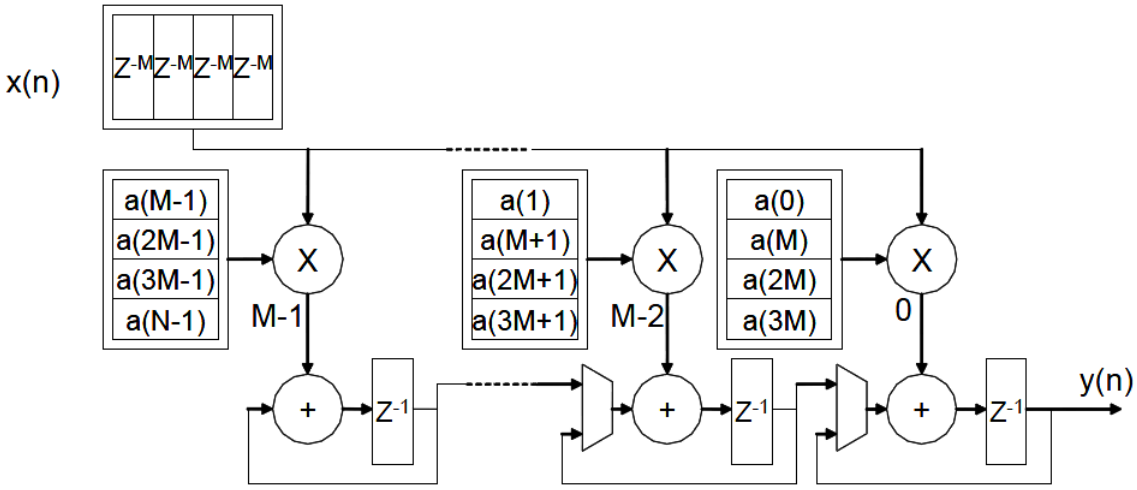

Fig.3.3.5 - Estructura Transpuesta de un filtro FIR Multi-MAC.

Esta estructura ofrece una aplicación de baja latencia, y para algunas configuraciones también puede ofrecer un ahorro de recursos adicionales sobre la estructura Directa. No requiere de un acumulador y se puede utilizar menos recursos de memoria de datos, a pesar de que no se aprovecha la simetría de los coeficientes, lo cual se describe a continuación.

COEFICIENTES

La respuesta al impulso de los filtros FIR posee simetría significativa. Esta simetría en general, puede ser explotada para minimizar los requisitos de implementación, optimizando la cantidad de recursos utilizados del hardware. La figura siguiente muestra la respuesta al impulso para un filtro FIR de coeficientes simétricos de 9 etapas.

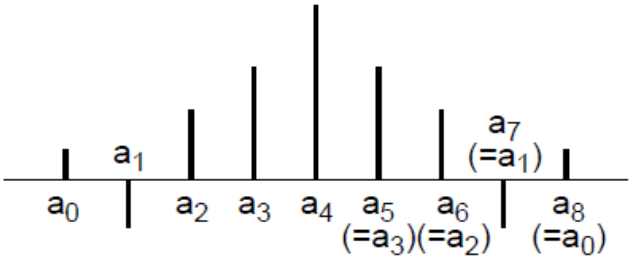

Fig.3.3.6 – Respuesta simétrica al impulso.

En lugar de aplicar este filtro, utilizando la arquitectura mostrada en la figura 3.3.3, se puede utilizar una estructura más eficiente, tal como se muestra en la figura 3.3.7. En general, el enfoque anterior requiere M multiplicaciones y (M-1) sumas. En contraste, la arquitectura en la Figura 3.3.7 sólo requiere [M/2] multiplicaciones y M sumas. Esta reducción significativa en la carga de trabajo computacional puede ser explotada para generar implementaciones más eficientes en el hardware.

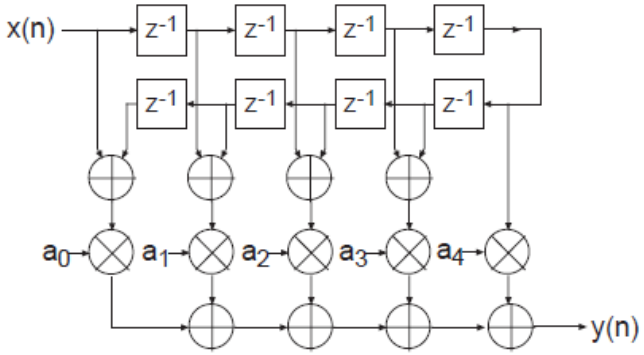

Fig.3.3.7 - Arquitectura para un número par de coeficientes.

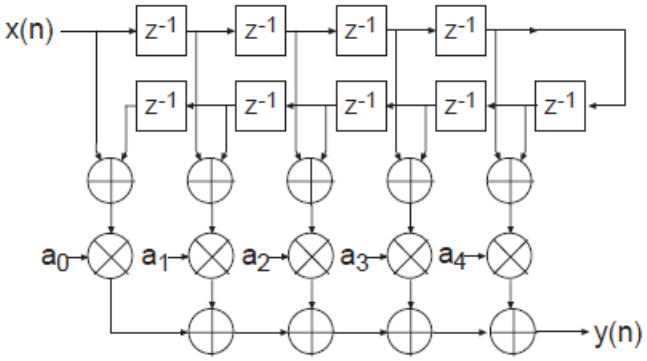

Fig.3.3.8 - Arquitectura para un número impar de coeficientes.

La figura 3.3.9 muestra la respuesta al impulso de un FIR con simetría negativa, o impar.

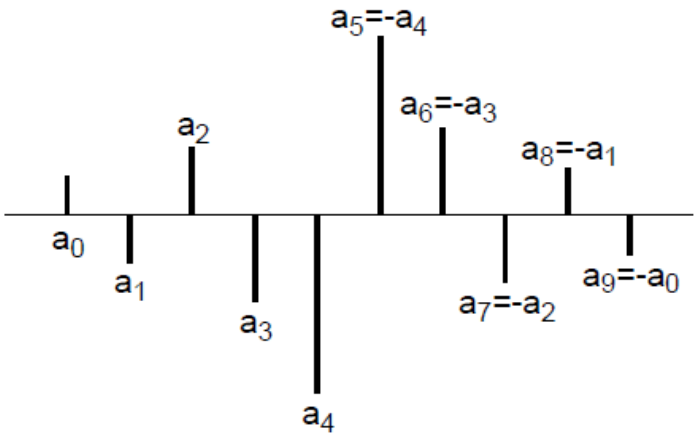

Fig.3.3.9 - Respuesta al impulso con simétrica negativa.

Esta simetría es fácil de explotar de una manera similar a la mostrada en la figura 3.3.7 y en la figura 3.3.8. En este caso, los sumadores son sustituidos por una combinación de sumadores y restadores, como se ilustra en la figura 3.3.10.

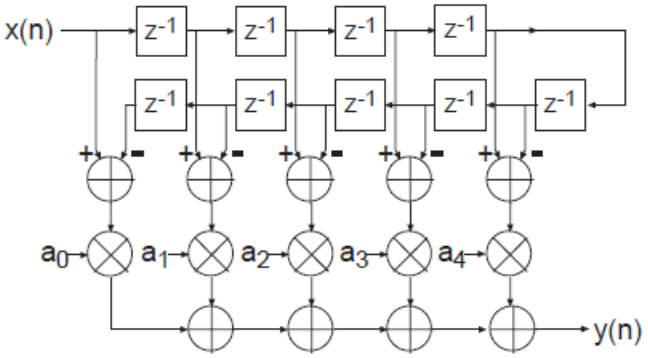

Fig.3.3.10 – Arquitectura de un FIR con simétrica negativa

En el desarrollo de este TFG se utilizó un filtro FIR con coeficientes no simétricos, implementado con multiplicadores y acumuladores en forma transpuesta debido a que permite incrementar la performance del filtro a frecuencias elevadas.

3.4 PLATAFORMAS DIGITALES

Los filtros digitales tienen la gran ventaja de poder ser diseñados, simulados e implementados desde un computador para finalmente ser implementados en una plataforma digital.

Existen tres categorías fundamentales de dispositivos de hardware que permiten implementar algoritmos de procesamiento digital de señales: DSPs, FPGAs y ASICs. Estos dispositivos constituyen componentes de hardware, siendo su elección un compromiso entre flexibilidad, capacidades de procesamiento y consumo de potencia.

Los DSP están basados en arquitecturas de microprocesadores optimizadas para el procesamiento digital de señales, que permiten el uso de lenguajes de alto nivel, como lenguaje C, con lo cual ofrecen la mayor flexibilidad entre los dispositivos antes mencionados.

Los ASICs proveen la implementación de hardware más optimizada en términos de velocidad y consumo de potencia. Sin embargo los ASICs implican diseños sofisticados de gran complejidad, con menos capacidades de flexibilidad y tiempos de diseño más elevados. Estas características llevaron a los alumnos a descartar esta opción.

Las FPGAs, por otro lado, ofrecen muchas capacidades de reconfiguración a nivel de hardware. Los diseñadores pueden configurar los arreglos de compuertas tanto como lo deseen, pero una vez configurado, el funcionamiento es estático. Proveen mayor flexibilidad que los ASICs pero menor a la ofrecida por los DSPs.

Los circuitos y los algoritmos de procesamiento digital de señal son implementados mediante lenguajes de descripción de hardware (HDL por sus siglas en ingles de Hardware Description Language) como VHDL o Verilog.

La utilización de dispositivos FPGA se incrementa en aplicaciones que requieren varios DSPs o DSPs multi-núcleo. Dichas aplicaciones complejas pueden ser implementadas por una o varias FPGAs de manera más eficiente.

Si bien la flexibilidad es mayor en los dispositivos DSP, las FPGA permiten un diseño más modular debido al alto grado de paralelismo que ofrecen y conexiones más veloces entre módulos del diseño, además de proveer grandes capacidades de reconfiguración a nivel hardware. Incluso las FPGAs más nuevas en el mercado poseen cientos de módulos específicos para realizar funciones de procesamiento digital de señal, como multiplicaciones y acumulación, sumas/restas, divisiones y raíces cuadradas, entre otras, a altas frecuencias de clock y de forma paralela [10].
Estas razones llevaron a los alumnos a optar por la utilización de FPGAs como dispositivo de hardware para el desarrollo de la plataforma reconfigurable propuesta.
Más precisamente, se contó con la disponibilidad de la placa de desarrollos XUPV2P compuesta por una FPGA Virtex II Pro de la empresa Xilinx Inc. provista por el Instituto Universitario Aeronáutico [11].

3.5 INTRODUCCIÓN A LAS FPGA. XILINX VIRTEX II PRO Y PLACA DE DESARROLLO XUPV2P.

Las FPGA son dispositivos semiconductores que contienen bloques de lógica cuya conexión y funcionalidad pueden ser programadas tantas veces como el usuario/desarrollador requiera. Su ámbito de aplicación es el mismo que el de los ASIC (Application Specific Integrated Circuit), pero sus características son diferentes, pues las FPGA son más lentas, consumen más y tienen limitaciones de espacio. Sin embargo, sus ventajas son una gran versatilidad y su bajo precio por unidad y al desarrollar sistemas, por lo que hoy en día son ampliamente utilizadas.

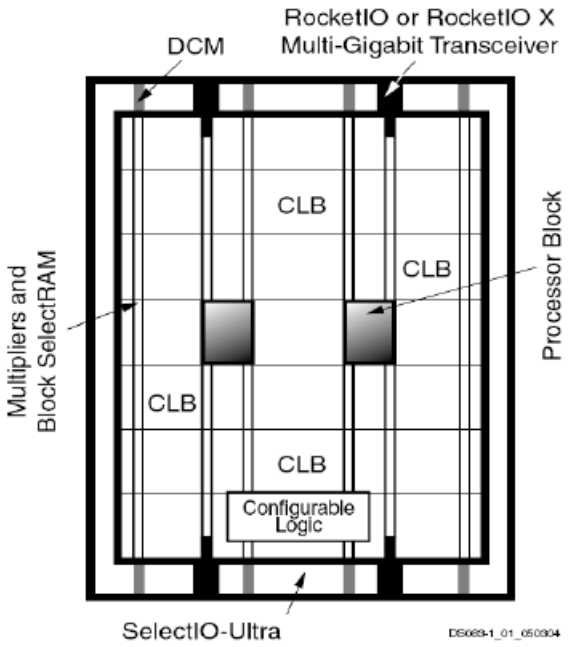

Fig 3.5.1 Arquitectura modular de la FPGA Virtex II Pro de Xilinx (Fuente: Xilinx).

En la figura 3.5.1 se puede observar un esquema de la arquitectura de la FPGA que fue seleccionada para el presente trabajo final de grado, la Virtex-II. Como se puede apreciar, se compone principalmente de [11]:

- CLB. Configurable Logic Blocks. Son los bloques que llevan a cabo la carga funcional del sistema. Constan, en este caso, de 4 slices cada uno, y son, por supuesto, programables en casi todos sus parámetros. Cada uno de los slices, está compuesto por dos módulos funcionales básicos implementables en forma de LUT (LookUp Table) de 4 entradas, en forma de registro de desplazamiento de 16 bits o de memoria SelectRAM distribuida también de 16 bits. Los dos elementos de memoria que se ven en el margen derecho son implementados según la decisión del usuario, pudiendo ser utilizados como latches activados por nivel o flip-flops por flanco, ambos de diversa índole. Además, cada slice cuenta con una gran cantidad de lógica de control y acarreo para llevar a cabo funciones más complejas o con mayor número de entradas.

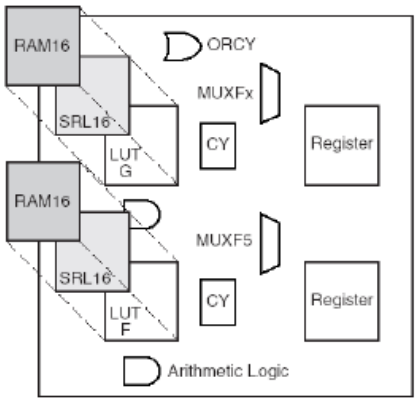

FIg 3.5.2 - Arquitectura interna de cada uno de los slices que componen un CLB en la Virtex II Pro (Fuente: Xilinx).

- IOB. In/Out. Esta FPGA cuenta con varios tipos de módulos de E/S. Aunque cambian sus especificaciones, su función es la de proporcionar una interfaz entre el exterior del dispositivo y la red de interconexión interna.

- DCM. Digital Clock Managers. Módulo que gestiona las señales de clock, de modo que lleguen a la vez a todos los puntos del dispositivo y que a su vez puede generar otras a partir de las recibidas.

- POWERPC 405E. Power PC Hard Core Processor 405E de IBM. Cuenta con dos procesadores embebidos dentro del dispositivo, por lo que resulta apto para el desarrollo de sistemas embebidos basados en co-diseños HW-SW. Son procesadores RISC (Reduced Instruction Set Architecture) de 32 bits que alcanzan una frecuencia de trabajo máxima de 400MHz. Su diseño se basa en la arquitectura Power de IBM, con una segmentación en 5 etapas y con una memoria caché de primer nivel de 16KB separada para instrucciones y datos.

- BRAM. Block Random Access Memory. Incorpora a su vez bloques de 18K de RAM distribuida por columnas entre el resto de elementos. La FPGA con código XCV2VP30, dispositivo con el que se dispone en este proyecto, implementa 136 bloques de memoria RAM dedicada de 18Kb, es decir, 2448KB distribuidos por todo el dispositivo. En caso de necesitar más memoria que la ofertada por uno de esos componentes, es fácilmente ampliable si se comunican varios módulos en cascada.

- Multiplicadores de 18 bits. Se trata de bloques dedicados capaces de multiplicar dos números de 18 bits con signo de forma rápida y con bajo consumo si se compara con un multiplicador similar sintetizado en slices. Pueden usarse independientemente o junto a bloques de SelectRAM con los que se sincronizan perfectamente.

- Red de Interconexión. Los recursos locales y globales de enrutado de la Virtex-II están optimizados para conseguir la máxima velocidad de transferencia posible sin comprometer para ello la flexibilidad. Es por ello que la arquitectura de la red de interconexión local se distribuye junto al resto de los elementos de la placa, a través de switches idénticos, como se puede apreciar en la figura 3.5.3. Globalmente, estos switches quedan unidos mediante los canales globales de enrutado desplegados en la placa. Dependiendo de las necesidades de conexión entre dos elementos dados, se elegirá el mejor camino entre la líneas existentes, ya sean horizontales o verticales, largas, hex, directas o dobles y con mejores o peores prestaciones.

La FPGA basa su funcionamiento en la cadena de configuración o bitstream, cargada por el usuario. Ésta es almacenada en la memoria SRAM que incorpora para tal fin.
Esta cadena binaria de configuración es la que denota las funciones, enrutamiento y demás información necesaria.

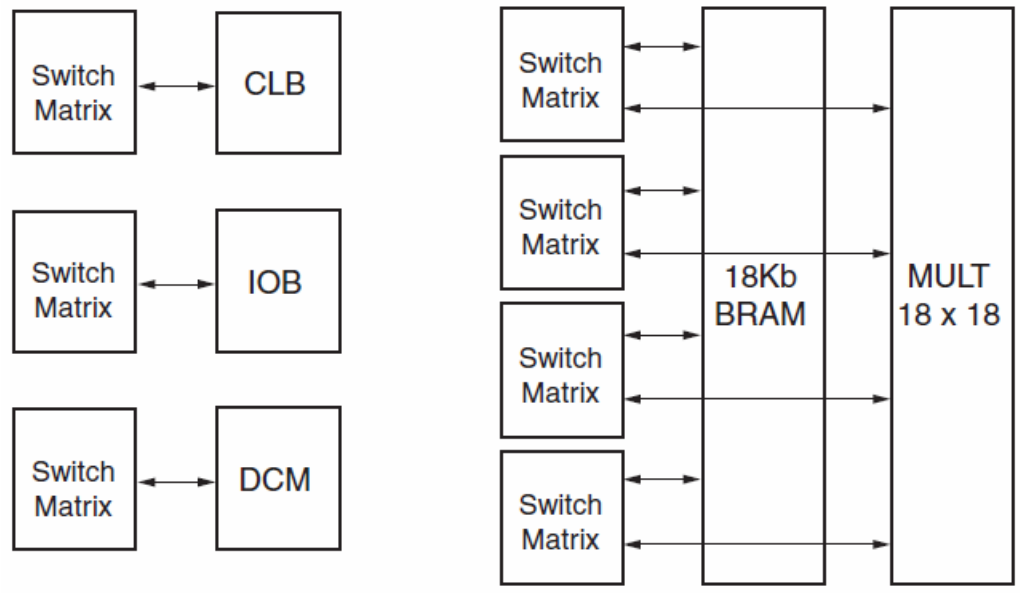

Fig 3.5.3 Tecnología activa de interconexión (Fuente: Xilinx).

Una vez presentada la FPGA con la que se trabajó, queda introducir la placa de desarrollo que sirvió de plataforma en el presente trabajo. Contamos con dispositivos basados en Virtex II Pro que se comercializan en placas de desarrollo con nomenclatura XUPV2P, acrónimo con el mismo significado. Según [11], se puede especificar que cuentan principalmente con:

- FPGA Virtex-II Pro y dos Power PC 405E empotrados.
- Hasta 2GB de memoria DDR SDRAM.
- Dispositivo de red Ethernet 10/100 integrado.
- Puerto serie RS-232.
- Dos puertos serie PS-2.
- Cuatro LED conectados a la FPGA.
- Cuatro botones conectados a la FPGA.
- Cinco pulsadores conectados a la FPGA.
- CODEC AC-97 para audio, con amplificador incorporado y salida audio.
- Entrada de sonido para, por ejemplo, un micrófono.
- Salida XSGA.
- Tres puertos Serial ATA.
- Reloj del sistema de 100MHz.

En la figura 3.5.4 podemos apreciar con más detalle cómo está dividido el dispositivo. Se puede observar que cuenta con la FPGA como elemento principal, una serie de entradas entre las que destacan las diferentes tensiones a las que es capaz de trabajar, la frecuencia de trabajo y la procedencia del bitstream de configuración, así como los módulos más arriba enumerados. El hecho de contar con esa gran cantidad de interfaces, tanto simples como complejos la hace versátil y a la vez potente, siendo de gran utilidad como plataforma de aprendizaje, pero también para desarrollos bastante más extensos y complicados. El aspecto físico de la placa puede apreciarse en la figura 3.5.5.

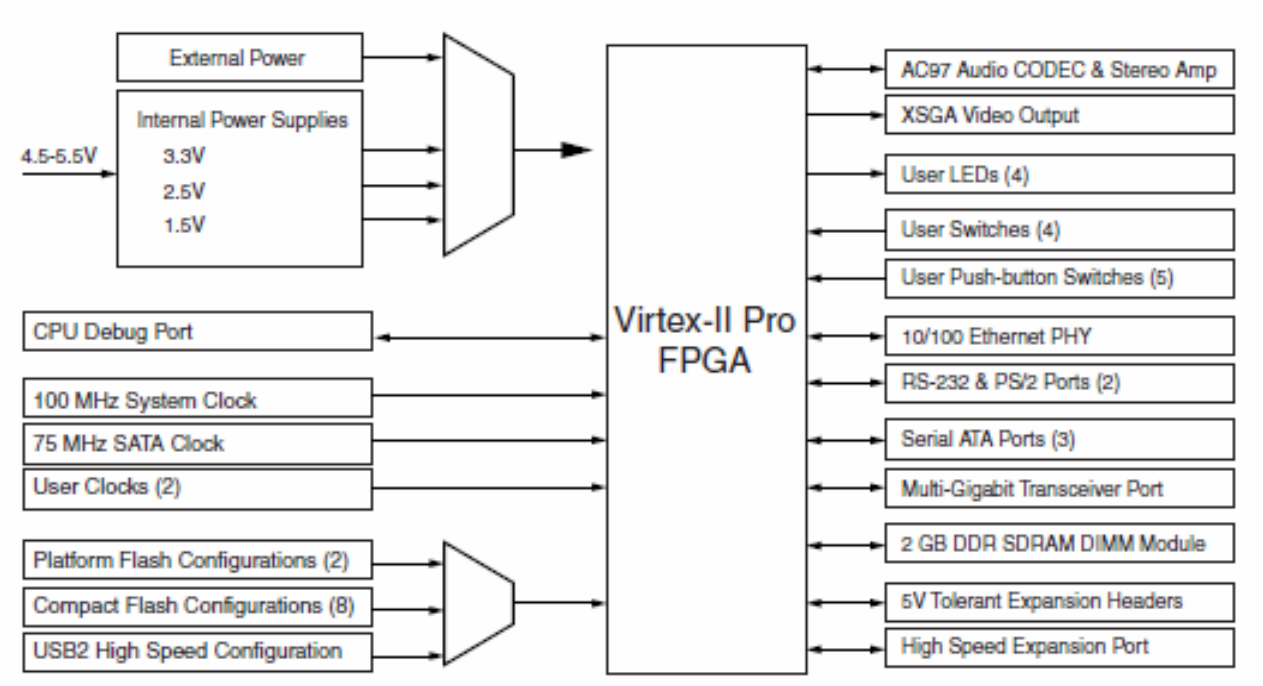

Fig 3.5.4. Diagrama modular de los principales bloques de la placa de desarrollo XUP Virtex-II Pro (Fuente: Xilinx).

La configuración del dispositivo reconfigurable se puede llevar a cabo siguiendo varios enfoques, seleccionables desde la propia placa a través de uno de los switches, que, coloreados en rojo se encuentran en la parte derecha de la placa según se observa en la Figura 3.5.5.

- JTAG (Joint Test Action Group). El proceso de configuración sigue las pautas explicadas en este estándar de validación de módulos y PCB (Printed Circuit Board). La fuente de datos de la configuración puede provenir de la tarjeta de datos Compact Flash conectable (Figura 3.5.5, parte derecha) a la propia placa o desde una fuente JTAG externa, como puede ser un cable PC4 (Paralell Conection 4) o el interfaz USB (Universal Serial Bus). Proporciona una velocidad de carga de datos baja, aunque por otra parte es el modo por defecto y también el más extendido.

- SelectMAP maestro. Es el estándar definido por Xilinx para la carga de bitstreams de configuración desde su herramienta propia iMPACT. La fuente de los datos es la PROM (Programmable Read-Only Memory) integrada en la placa. Ésta proporciona gran velocidad de transferencia y puede ser configurada por el usuario.

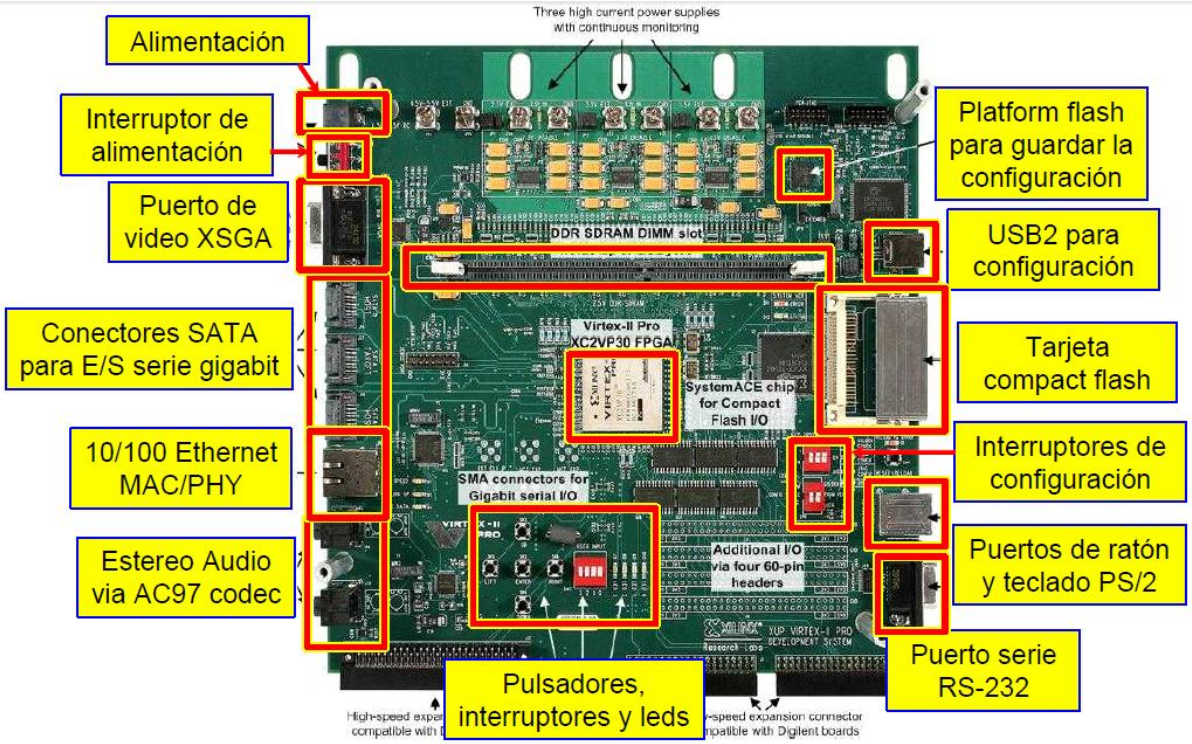

Fig.3.5.5 Placa de desarrollo XUPV2P

3.6 HERRAMIENTAS DE DISEÑO.

Para el diseño, simulación e implementación del presente TFG se utilizó la herramienta System Generator de *The MathWork MATLAB/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 *LogiCOREs de Xilinx* [13].

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 de 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. Además hay bloques de Xilinx IP que también pueden ser implementados con la debida licencia. En la figura 3.6.1 a continuación se muestran algunos bloques de System Generator.

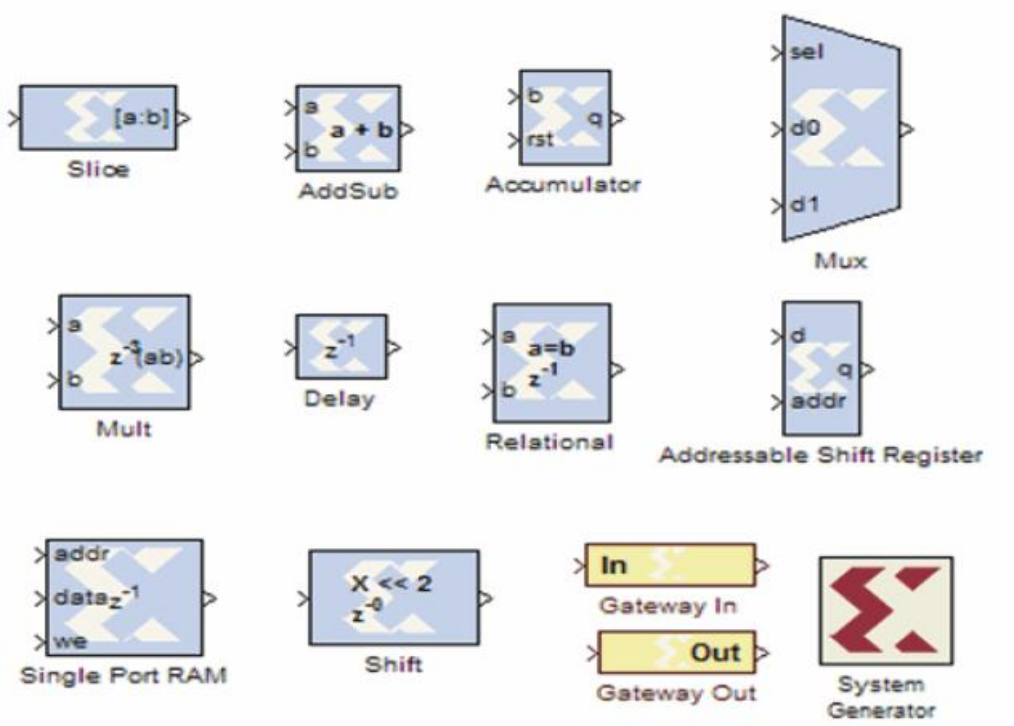

Figura 3.6.1 Bloques Xilinx Dps para simulink

BLOCKSET DE XILINX EN SIMULINK

Xilinx *Blockset* es una familia de bibliotecas que contienen los bloques básicos del System Generator. Algunos bloques son de bajo nivel, facilitando el acceso al hardware específico del dispositivo. Otros son alto nivel, de ejecución procesamiento de señales y algoritmos avanzados de comunicación. Para mayor comodidad, los bloques con una amplia aplicabilidad (por ejemplo, los bloques *Gateway*) son miembros de varias bibliotecas. Cada bloque se encuentra en la biblioteca índice. Las bibliotecas se describen en la tabla 3.6.1.

Librería	Descripción
Index	Cada Bloque en el Blockset de Xilinx.
Basic Elements	Bloques de elementos estándar para la construcción de lógica digital.
Communication	Bloques de corrección de errores y moduladores, de uso común en los sistemas de comunicaciones digitales.
Control Logic	Bloques de circuitos de control y máquinas de estado
Data Types	Los bloques que convertir tipos de datos (incluye gateways)
DSP	Bloques de Procesamiento Digital de señales (DSP)
Math	Bloques que implementan funciones matemáticas
Memory	Bloques para la implementación y acceso a memoria.
Shared Memory	Bloques para la implementación y acceso a memoria compartida de Xilinx
Tools	bloques de Herramientas, por ejemplo, la generación de código (bloque System Generator), estimación de recursos, HDL co-simulación, etc

Tabla 3.6.1 Bibliotecas de Xilinx Blockset

XILINX REFERENCE BLOCKSET

Xilinx Reference Blockset contiene compuestos de bloques de System Generator que implementan una amplia gama de funciones. Bloques en este Blockset se organizan por su función en diferentes bibliotecas. Las bibliotecas se describen en la Tabla 3.6.2 Cada bloque en este Blockset es un compuesto es decir, se implementa como un subsistema de enmascarados, con los parámetros que configuran el bloque [13].

Librería	Descripción
Communication	Bloques de corrección de errores y moduladores, de uso común en los sistemas de comunicaciones digitales.
Control Logic	Bloques de circuitos de control y máquinas de estado
Imaging	Bloques de Procesamineto de Imágenes.
DSP	Bloques de Procesamiento Digital de señales (DSP)
Math	Bloques que implementan funciones matemáticas

Tabla 3.6.2 Bibliotecas de Xilinx Reference Blockset

Se puede utilizar los bloques de las bibliotecas de referencia Blockset como están, o como puntos de partida en la construcción de diseños que tienen características similares. Cada bloque de referencia tiene una descripción de su aplicación y los requisitos de recursos de hardware. En la figura 3.6.2 se muestra la ubicación de los Blockset de Xilinx en el Toobox de simulink.

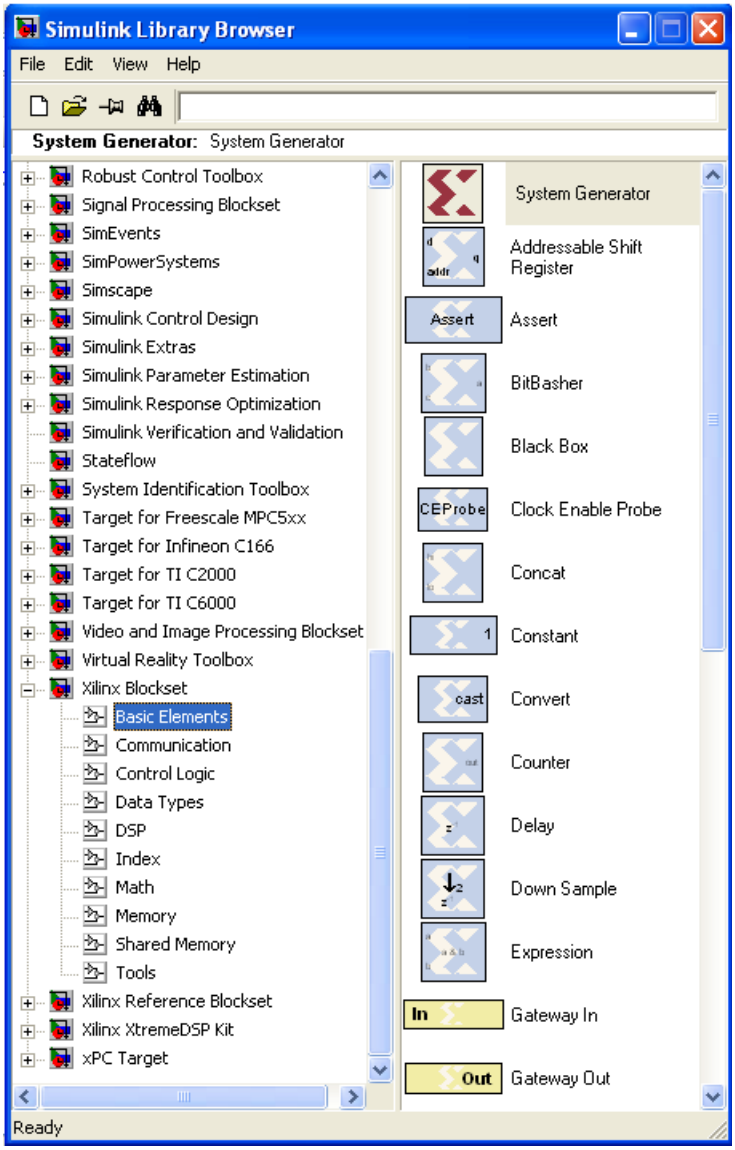

Figura 3.6.2 Blockset de Xilinx en Toobox de Simulink

TIPOS DE SEÑALES EN SYSTEM GENERATOR

Los bloques en Xilinx System Generator operan con valores booleanos o valores arbitrarios en punto fijo. Esto es para dar una mejor aproximación a la simulación hardware en simulink (Ver sección 3.7). En contraste Simulink trabaja con números de punto flotante de doble precisión. La conexión entre los bloques de Xilinx System Generator y los bloques de Simulink son los bloques "Gateway".

El Bloque "Gateway In" convierte una señal de doble precisión en una señal de Xilinx, y el bloque "Gateway Out" convierte una señal de Xilinx en una de doble precisión. Las señales continuas de Simulink deben ser muestreadas por el bloque "Gateway In".

La mayoría de los bloques de Xilinx son polimórficos, es decir, son capaces de deducir los tipos adecuados de salida basándose en sus tipos de entrada. Cuando se especifica *"full precisión"* en los parámetros en el cuadro de dialogo del bloque, System Generator elige el tipo de salida para garantizar que no se pierda precisión.

La extensión de signo y el relleno de ceros se generan automáticamente si es necesario. Esto permite establecer el tipo de salida de un bloque y para especificar como la cuantización y saturación deben ser manejados. Posibilidades de cuantificación objetiva incluyen el redondeo hacia más o menos infinito, dependiendo del signo, o el truncamiento. Las opciones disponibles de saturación son *saturation*, *truncation* y *reporting overflow as an error.*
En la parte de System Generator de un modelo simulink, cada señal debe ser muestreada. Tiempos de muestra deben ser heredados usando reglas de propagación de Simulink, o estableciendo explícitamente la configuración en el cuadro de dialogo de un bloque.

BLOQUE SYSTEM GENERATOR

Cualquier diseño que incluya un bloque de Xilinx debe incluir este bloque. Es el encargado de proporcionar el control del sistema y los parámetros de las simulaciones, e invocar el generador de código. En la figura 3.6.3 se muestra el cuadro donde especificar los parámetros para este bloque [13]. Estos son:

- *Compilation:* especifica el tipo de compilación que se producirá cuando se invoque al generador de código.

- *Part:* define la FPGA usada.

- *Target Directory:* nombre del directorio en el que guardar los resultados de la compilación.

- *Synthesis Tool:* indica la herramienta usada para sintetizar el diseño. Las posibles opciones son Synplicity's Synplify Pro, Synplify, y Xilinx's XST.

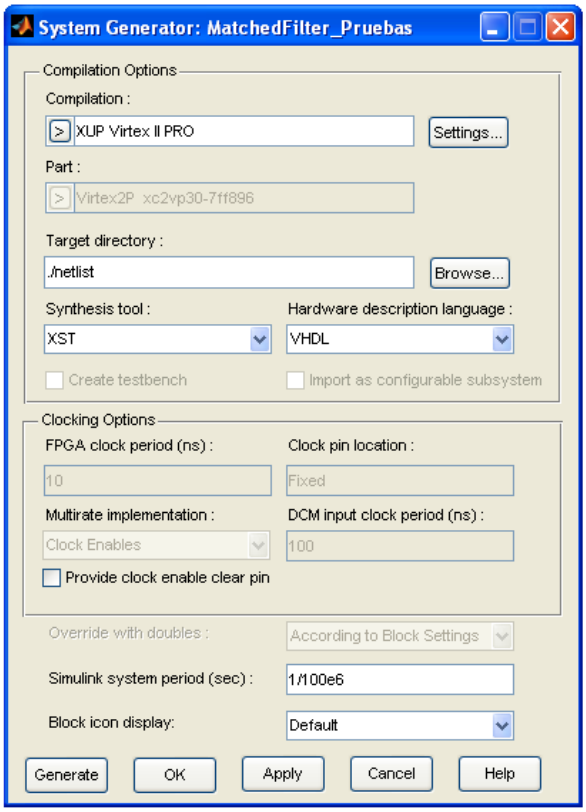

Figura 3.6.3 Parámetros del bloque System Generator

- *Hardware Description Language:* especifica el lenguaje HDL que se usará para la compilación. Puede ser VHDL o Verilog.

- *FPGA Clock Period:* define el período, en nanosegundos, del reloj hardware. Este valor debe ser un entero.

- *Clock Pin Location:* indica cuál es el pin del reloj. Esta información se le pasa a las herramientas de implementación de Xilinx a través de un archivo (*xcf* o *ngc*) en el que también se incluyen otro tipo de restricciones.

- *Create Testbench:* si está activada esta opción le indica a System Generator que cree un *testbench*.

- *Import as Configurable Subsystem:* le indica a System Generator que haga dos cosas:
  - Que construya un bloque como resultado de la compilación.
  - Que construya un subsistema configurable.

- *Provide clock enable clear pin:* le indica a System Generator que proporcione un puerto *ce_clr* en el *top level* que controla la lógica del reloj (*clock wrapper*). La señal *ce_clr* se usa para resetear la lógica de generación de habilitación del clock. *clock wrapper* es un mecanismo de encapsulación necesario para generar la lógica del reloj y que forma el *top-level* del diseño.

- *Override with doubles:* indica que todos los cálculos deberían realizarse usando precisión doble.

- *Simulink System Period:* define el período de Simulink, en segundos.

- *Block Icon Display:* especifica el tipo de información que será mostrada en los iconos de los bloques. Las opciones disponibles son:
  - *Default:* muestra la información por defecto del bloque.
  - *Sample rates:* indica la tasa normalizada para cada puerto de entrada y salida. Esta información suele ser útil para comprobar que no hay errores.
  -*Pipeline stages:* indica la latencia de los puertos de entradas. Esta información no está disponible para algunos bloques.
  - *HDL port names:* muestra el nombre de los puertos de entrada y salida.
  - *Input data types:* muestras los tipos de datos de las señales de entrada a los bloques. Esta información suele ser útil para resolver errores.
  - *Output data types:* muestras los tipos de datos de las señales de salida de los bloques. Esta información suele ser útil para resolver errores.

TIPOS DE COMPILACIÓN

- **HDL**

Ésta es la compilación por defecto. En ella se generan los archivos HDL, NGC y EDIF necesarios para implementar el modelo diseñado. Adicionalmente se crean otros archivos que simplifican este proceso. Todos los archivos creados se guardan en el directorio que se indique en el bloque *System Generator.*

- **NGC**

Si se selecciona esta opción permite a System Generator crear un archivo NGC. Este archivo contiene tanto la información lógica como los requisitos temporales y localización de los pines de los puertos. Esto significa que toda la información relativa al diseño está contenida en un solo archivo.

- **Bitstream**

Esta compilación crea un archivo *bit* que se puede cargar directamente en la FPGA. Este archivo se llama *nombre_cw.bit* y se guarda en el directorio que se indique en el bloque *System Generator*.

- **Hardware Co-simulation**

Este tipo de compilación es la que se utiliza para crear el modelo que se usa en la co-simulación hardware.

- **Timing**

En ocasiones el hardware creado por System Generator puede no cumplir con los requisitos de tiempo. System Generator proporciona una herramienta que permite realizar un análisis temporal para resolver este tipo de conflictos. Mediante este

análisis se muestra el camino más lento de la parte hardware diseñada, así como aquellos que no cumplen con los requisitos temporales. Para poder realizar este tipo de análisis System Generator se basa en la herramienta Trace, que forma parte del paquete ISE de Xilinx.

OPCIONES DE CLOCKING.

Como se muestra en la figura 3.6.3 cuando se utiliza el bloque de System Generator para compilar el diseño en Hardware, hay tres opciones de clocking para implementación multirate: (1) Clock Enables (por defecto), (2) Hybrid DCM-CE, and (3) Expose Clock Ports.

- **Clock Enable**

Cuando System Generator compila un modelo en el hardware con la opción *Clock Enable*, System Generator conserva la información de frecuencia de muestreo del diseño de tal manera que las partes correspondientes en hardware funcionan a velocidades adecuadas. System Generator genera frecuencias relacionadas utilizando un reloj en conjunto con el clock enable, permitiendo una sola frecuencia. El período de cada clock enable es un múltiplo entero del periodo del reloj del sistema.

Dentro de Simulink, ni relojes, ni clock enable se requieren como señales explícitas en un diseño del System Generator. Cuando el System Generator compila un diseño en el hardware, utiliza las frecuencias de muestreo en el diseño para deducir que clock enables son necesarios. Para ello, emplea dos valores especificados por el usuario del Bloque de System Generator: el período del sistema en simulink y el período del reloj del FPGA. Estos números definen el factor de escala entre el tiempo en una simulación de Simulink, y el tiempo en la aplicación de hardware real. El periodo del sistema en simulink debe ser el máximo común divisor (mcd) de los períodos de muestreo que aparece en el modelo, y el período de reloj FPGA, en nanosegundos, del reloj del sistema. Si p representa el período del sistema en Simulink, y c representa el período del reloj del sistema FPGA, entonces si un componente que tiene unidades de tiempo kp en Simulink, k momentos del reloj del sistema (de ahí nanosegundos kc) en el hardware.

Para ilustrar este punto, se puede considerar un modelo que tiene tres periodos de muestra de diferentes componentes en Simulink 2, 3 y 4. El mcd de estos períodos de muestra de los componentes es 1, este valor debe especificarse en el campo de periodo de Simulink en el bloque de System Generator. Suponiendo que el clock de la FPGA es 10 ns. Con esta información, el periodo del clock enable correspondiente puede ser determinado en el hardware.

En cuanto a hardware se refiere el clock enable correspondiente para periodos de muestra de simulink 2, 3 y 4 como CE2, CE3 y CE4, respectivamente. La relación de cada período de clock enable con el período del reloj del sistema puede ser determinado dividiendo el correspondiente período de los componentes de Simulink por el valor del período del sistema en Simulink. Por lo tanto, los periodos de CE2, CE3 y CE4 son iguales a 2, 3, 4 respectivamente ya

que el valor del periodo del sistema en simulink es1. En la figura 3.6.4 se muestra un diagrama de este análisis.

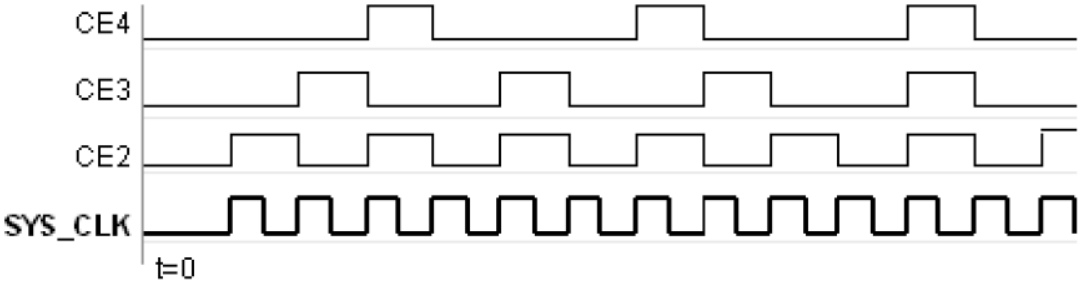

Figura 3.6.4 Clock Enables para diferentes periodos de muestra.

- **Opción Hybrid DCM-CE.**

Si el objetivo es implementar en un FPGA con un *Digital Clock Manager* (DCM), se puede escoger manejar un árbol de reloj con DCM. La opción DCM es deseable cuando *fanout* de alta en las redes de Reloj hacen que sea difícil lograr el cierre de tiempo.

System Generator instancia el DCM en un contenedor de reloj top-level HDL y configura el DCM para proporcionar hasta tres puertos de reloj a diferentes frecuencias para Virtex 4 y Virtex 5 y hasta dos puertos de reloj para la Spartan-3A DSP. Si el diseño tiene más puertos de reloj que el DCM puede soportar, los relojes restantes son compatibles con la configuración de CE (*clock Enable*).

La opción *Hybrid DCM-CE* tiene limitaciones conocidas en algunos bloques de System Generator que se muestran a continuación.

- Clock Enable Probe
- Clock Probe
- DAFIR
- Downsample
- FIR Compiler
- Parallel to Serial
- Time Division De-Multiplexer
- Time Division Multiplexer
- Upsample

- **Opción Expose Clock Ports**

Al seleccionar esta opción, el System Generator crea un contenedor top_level que expone un puerto de reloj para cada frecuencia. Entonces se puede instanciar manualmente un generador de reloj fuera de la unidad de diseño para los puertos del reloj.

3.7 ARITMETICA DE PUNTO FIJO Y CUANTIFICACIÓN.

Puesto que todo algoritmo deberá implantarse en un dispositivo de procesamiento de señales, donde el número de bits para representar las cantidades es limitado, es

importante garantizar que los errores introducidos debido a problemas de representación, no se incrementen a lo largo del proceso de digitalización de la señal y filtrado; puesto que si esto sucede pueden ocurrir respuestas erróneas del mismo.
En el mundo de la programación existen dos tipos de representación numérica:

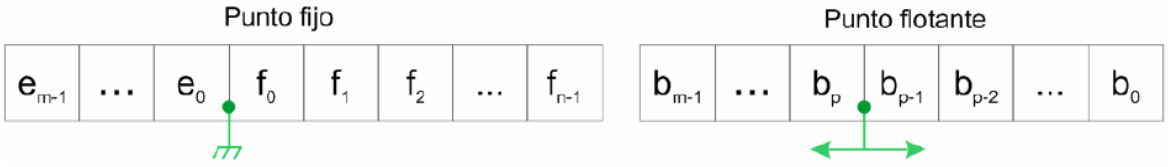

Fig.3.7.1 - Diferencias entre punto fijo y punto flotante.

En el caso de la aritmética de punto fijo, el punto binario se encuentra siempre en la misma posición, es decir, existirán m bits para la parte entera y n bits para la parte decimal. En algunos casos puede ocurrir que m = 0 (no existe parte entera) o bien, n = 0 (no existe parte decimal). En cambio, para la aritmética de punto flotante, la ubicación del punto binario puede variar (Figura 3.7.1).
En el presente TFG se utilizó aritmética de punto fijo ya que es la utilizada por la mayoría de los DSP´s disponibles en el mercado.

ARITMÉTICA DE PUNTO FIJO

Se sabe que un registro de 4 bits permite representar 16 combinaciones diferentes, pero hay que definir que representa ese conjunto combinaciones o cómo interpretarlos.
Por ejemplo se supone que un ADC de 4 bits, el cual se encuentra digitalizando una señal cualquiera, entrega la siguiente muestra (En base binaria):

$$X_0 \equiv 10010_2$$

Este número binario no representa un solo valor, ni varios, sino infinitos, ya que la interpretación de esa combinación dependerá de lo que el usuario defina como convención en su proyecto. Por ejemplo, se podría suponer la siguiente interpretación (en base decimal):

$$X_0 \equiv \left(3b_4 + 2b_3 + 0.5b_2 + 0.5b_1 + b_0\right)_{10}$$

Así, el número decimal representado por la muestra anterior será:

$$X_0 \equiv \left(3\times 1 + 2\times 0 + 0.5\times 0 + 0.5\times 1 + 0\right)_{10} \equiv 3.5_{10}$$

En microcontroladores, DSP´s y microprocesadores comerciales esta convención está especificada por el fabricante.
Para un conjunto de bits como el de la figura 3.7.2, existe una serie de interpretaciones posibles que son ampliamente aceptadas y que se resumen en la tabla 3.7.1 [16][17].

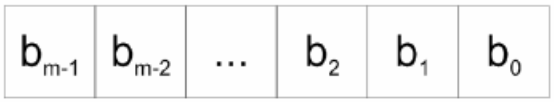

Fig.3.7.2 - Conjunto de bits.

Interpretación	Expresión	Rango
Entero sin signo	$x(n) = \sum_{i=0}^{m-1} b_i.2^i$	$0 \le x_Q \le 2^m - 1$
Entero con signo	$x(n) = -b_{m-1}.2^{m-1} + \sum_{i=0}^{m-2} b_i.2^i$	$-2^{m-1} \le x_Q \le 2^{m-1} - 1$
Fraccional sin signo en complemento a 2	$x(n) = \sum_{i=0}^{m-1} b_i.2^{-(m-i)}$	$0 \le x_Q \le 1 - 2^{-m}$
Fraccional con signo en complemento a 2	$x(n) = -b_{m-1} + \sum_{i=0}^{m-2} b_i.2^{-(m-i-1)}$	$-1 \le x_Q \le 1 - 2^{-(m-1)}$

Tabla 3.7.1 - Posibles interpretaciones de un conjunto de bits.

Generalmente, para el procesamiento de señales se emplea la aritmética fraccional con signo en complemento a 2, mientras que para indexado y punteros a memoria se utiliza la aritmética entera.
Implementar por hardware una aritmética de punto fijo conlleva una mayor simplicidad, lo cual se traduce directamente en menores costos. Además ocupa menor superficie de silicio respecto a una unidad de punto flotante, lo que permite agregar al procesador más módulos y memoria.
Entre las desventajas más significativas se encuentra el hecho de que los errores de truncamiento con esta arquitectura pueden ser significativos, degradando la calidad de los algoritmos implementados.
Para lograr en punto fijo la misma precisión que se logra con punto flotante se necesitaría una cantidad muy grande de bits. En el estándar ANSI/IEEE Std. 754-1985 se encuentra que el menor valor representable en punto flotante de 32 bits es ± 1,2 10^{-38}. Para lograr la misma precisión en punto fijo se requerirían:

$$2^{-n} \equiv 1{,}2 \times 10^{-38} \Rightarrow n \equiv -\log_2\left(1{,}2 \times 10^{-38}\right) \equiv 125{,}97 bits \cong 126 bits$$

Casi el cuádruple de bits que para la misma precisión en aritmética flotante

CUANTIFICACION

Ciertas características de las señales de radiofrecuencia al propagarse por el medio, como el desvanecimiento (causado por los múltiples trayectos que sigue la señal al propagarse) y la obstrucción de las mismas entre el transmisor y el receptor, en conjunto con el bloqueo e interferencia debido a la presencia de otras señales transmitidas, da como resultado señales de un amplio rango dinámico (Relación entre el nivel máximo y mínimo de la señal recibida).
Si bien es posible emplear escasos componentes para el tratamiento analógico de las señales, como amplificadores de bajo ruido (LNA, por sus siglas en ingles de Low Noise Amplifier) destinados a amplificar las señales débiles recibidas, y filtros anti-alias para eliminar señales indeseadas adoptar este tipo de arquitectura es posible solo si las señales analógicas pueden ser llevadas al dominio digital de manera apropiada. Es por ello que los conversores analógico-digital (ADC) juegan un papel muy importante en estos sistemas [6].
Los parámetros que influyen en la digitalización de la señal se listan a continuación [9]

- Resolución: implica la cantidad de bits utilizada para la cuantificación de la señal de entrada. Mientras mayor sea el número de bits, menor es la magnitud del ruido de cuantificación introducido, pero a su vez, la complejidad y consumo de potencia del ADC se incrementa

- Relación Señal Ruido (SNR): se refiere a la sensibilidad del conversor ante señales débiles. Esta depende del ruido de cuantificación introducido dada la cantidad de bits empleada por el ADC, como así también el ancho de banda del mismo.

- Rango dinámico libre de espurios (SFDR, siglas de Spurious-Free Dynamic Range en literaturas de habla inglesa): parámetro medido ante entradas senoidales, expresa la diferencia entre la componente fundamental de la señal de entrada y el nivel de componentes espurios generados debido a la no linealidad del conversor ADC. Mientras mayor sea el ancho de banda de la señal, mejor es el seguimiento de la misma por parte del ADC, mejorando así el parámetro SFDR.

- Ancho de banda: en estas aplicaciones es necesario que el ancho de banda del conversor ADC sea elevado debido a la necesidad de digitalizar distintos tipos de señales en distintos rangos de frecuencia. Este valor puede ser considerablemente mayor que la frecuencia de muestreo, como es el caso de conversores aptos para muestrear señales pasa banda.

- Frecuencia de muestreo: altas tasas de muestreo no solo ayudan a cumplir los requisitos teóricos de muestreo, sino que también ayuda a distribuir el ruido de cuantificación sobre el espectro, mejorando la densidad espectral del ruido y por ello, la sensibilidad del receptor. Sin embargo, esto aumenta la complejidad de los conversores, como así también el consumo de

potencia de los mismos. la principal característica del muestreo es que debe cumplir con el teorema del muestreo de Nyquist:

- Para poder muestrear una señal, requerimos que esta sea limitada en banda. Su frecuencia máxima es llamada f_M.

- Para no perder información sobre la señal durante el proceso de muestreo, tenemos que muestrear a una tasa mayor que el doble de f_M; es decir, la frecuencia de muestreo $fs > 2f_M$.

El proceso de cuantificación consiste esencialmente en escoger cuántas amplitudes de pulso serán permitidas, y convertir la señal de entrada a otra señal donde los pulsos tienen sólo las amplitudes permitidas. Conviene escoger un número de amplitudes que sea potencia de dos, por la forma en que funciona el siguiente proceso. Los pasos a seguir son entonces:

- Se escoge el número de amplitudes M = 2k.
- Se calculan las amplitudes posibles, con extremos en los valores mínimo y máximo de la señal de entrada.
- Para cada pulso de la señal, se cambia su amplitud a la amplitud posible más cercana (proceso conocido como redondeo).

El número de amplitudes permitidas por el cuantificador se conoce también como niveles de cuantificación. Cuando las amplitudes posibles se distribuyen de manera uniforme entre el máximo y el mínimo, se dice que la cuantificación es uniforme. A la diferencia entre niveles, Δ, se le conoce como paso de cuantificación.
A cada muestra cuantificada se le denomina xq(n). En la figura 3.7.5 se muestra este proceso para M = 8. Normalmente la entrada al cuantificador debe estar acotada; de no ser así, la diferencia entre una muestra y su valor cuantificado pudiera ser muy grande. Si,

$$|x(n)| \leq x_{\max} \qquad \text{Ec.3.7.1}$$

Y si tenemos un cuantificador con M niveles de cuantificación, el paso de cuantificación está dado por

$$\Delta = \frac{x_{\max}}{2^{k-1}} \qquad \text{Ec.3.7.2}$$

A continuación se visualiza el proceso de digitalización de la señal

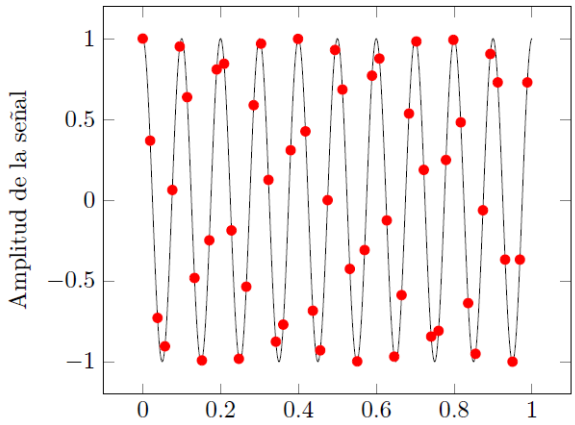

Fig.3.7.3 - Señal original x(t) = cos(2π10t), en negro; en rojo, muestras de x(t) tomadas a frecuencia fs = 53 Hz.

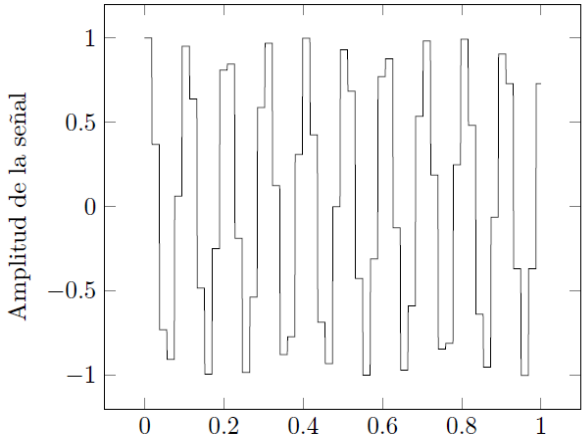

Fig.3.7.4 - Señal x(t) muestreada con sample-and-hold.

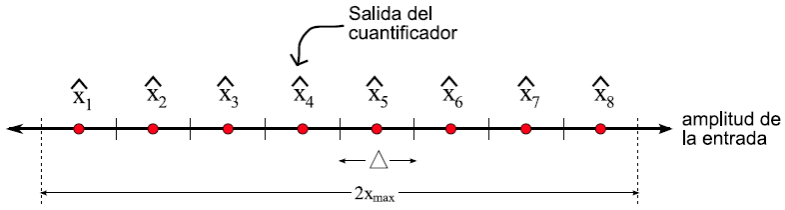

Fig.3.7.5 - Diagrama de un cuantificador para M = 8.

Al valor $RD \equiv 2x_{\max}$ se le conoce como el rango dinámico del cuantificador. Para utilizar al máximo el rango dinámico del cuantificador, se supone que la señal está centrada, es decir, que su amplitud máxima es muy similar al negativo de su amplitud mínima.
El error de cuantificación para la muestra n es la diferencia entre el valor real de la muestra y su valor cuantificado, es decir:

$$e(n) \equiv x_q(n) - x(n) \qquad \text{Ec.3.7.3}$$

El error está siempre en el rango:

$$-\frac{\Delta}{2} \le e(n) \le \frac{\Delta}{2}$$

Lo importante de notar aquí es que el proceso de cuantificación implica una pérdida de la información de la señal transmitida. La señal que se recupera de las muestras cuantificadas no es idéntica a la señal original. Se dice que el proceso de cuantificación introduce ruido a la señal; este ruido se conoce como ruido de cuantificación.
El ruido de cuantificación se analiza de la siguiente manera. Primero, se calcula la potencia de la señal sin cuantificar, x(t), con la siguiente fórmula:

$$P_x \equiv \frac{1}{N}\sum_{n=1}^{n=N} x^2(n) \qquad \text{Ec.3.7.4}$$

Y se calcula la potencia de la diferencia entre la señal sin cuantificar y la señal cuantificada xq(t) como sigue:

$$P_q \equiv \frac{1}{N}\sum_{n=1}^{n=N} \left(x_q(n) - x(n)\right)^2 \qquad \text{Ec.3.7.5}$$

En estas ecuaciones, N es el número de muestras que se tomaron de la señal x(t).
La división entre Px y Pq se conoce como la relación señal a ruido (SNR):

$$SNR_q \equiv \frac{P_x}{P_q} \qquad \text{Ec.3.7.6}$$

Que suele medirse en decibeles:

$$SNR_{q,dB} \equiv 10\log_{10}\frac{P_x}{P_q} \qquad \text{Ec.3.7.7}$$

Otra forma de calcular la relación señal a ruido es como sigue. En muchos casos es razonable suponer que el ruido está distribuido de forma uniforme entre -Δ/2 y Δ/2, por lo que su potencia es:

$$P_q \equiv \frac{\Delta^2}{12} \equiv \frac{x_{\max}^2}{12\left(2^{k-1}\right)^2} \equiv \frac{x_{\max}^2}{12\left(2^{2k-2}\right)} \equiv \frac{x_{\max}^2}{3\times 2^{2k}} \equiv \frac{x_{\max}^2}{3\times M^2} \equiv \frac{RD^2}{12\times M^2}$$

Entonces el SNR puede calcularse como:

$$SNR_{q,dB} \equiv 10\log_{10}\left(\frac{P_x \times 12 \times 2^{2k}}{RD^2}\right) \equiv 10\left(\log_{10} 12 + \log_{10} 2^{2k} - \log_{10}\frac{RD^2}{P_x}\right) \equiv 10.8 + 6.02k - 10\log_{10}\frac{RD^2}{P_x}$$

Se dice entonces que la relación señal a ruido mejora en 6 dB cada vez que se añade un bit (se aumenta k en uno) a la cuantificación. Hay que recalcar que esta fórmula sólo es válida para un error uniforme de cuantización.

En muchas ocasiones no se cuenta con la señal original y por tanto no se conocen los valores de RD y de Px. En este caso se hace una simplificación y se supone que la señal que se cuantifica es senoidal.

CAPÍTULO 4: MATCHED FILTER

4.1 INTRODUCCIÓN

Un filtro apareado es un sistema lineal invariante cuya función principal es detectar la presencia de una señal conocida, o referencia, dentro de una señal recibida. La señal a la salida del filtro será la correlación de la señal referencia con la señal desconocida. Este procedimiento es equivalente a realizar la convolución de la señal desconocida con una versión retardada de la señal, que usamos como referencia.
El filtro apareado es el sistema óptimo para maximizar la SNR en presencia de ruido blanco aditivo y gaussiano. Su uso es común en aplicaciones de radar, donde se envía una señal que luego se pretende detectar.

4.2 DESARROLLO TEÓRICO

Una red cuya función de respuesta en frecuencia maximiza la relación señal(pico) a ruido(medio) presente en la salida, se denomina filtro apareado.
La función de transferencia H(f) expresa la amplitud y fase relativa a la salida de la red, respecto a la entrada, cuando por ella se le ingresa una senoide pura. La magnitud de H(f) es la característica pasa banda del receptor. Cuando esta característica pasabanda es más extensa comparada con la que ocupa la señal de energía, un ruido extra se suma afectando la relación señal ruido. Por otro lado, si se considera una característica pasabanda mas angosta que la que ocupa la señal, se disminuye tanto la energía del ruido como la de la señal, produciendo nuevamente una disminución de la relación señal a ruido. Esto significa que hay un ancho de banda que optimiza la relación señal a ruido y es el de nuestro interés. Una regla básica es que este ancho de banda sea aproximadamente igual a la inversa del ancho del pulso. Aproximación válida para radares pulsados con receptores del tipo superheterodino. La especificación exacta del receptor óptimo consiste en la función de respuesta en frecuencia y la forma de onda recibida.
Para una señal de entrada s(t), con una determinada relación entre la energía E de la señal y energía de ruido No(potencia de ruido por hertz), se demuestra que la función de respuesta en frecuencia de un SLIT que maximiza la relación potencia pico de señal a potencia de ruido medio a la salida, para una determinada relación señal ruido (Energía) a la entrada es [3]:

$$H(f) = G_a S^*(f) \exp(-j2\pi . f . t_1) \qquad \text{Ec.4.2.1}$$

El desarrollo de esta ecuación se encuentra en la siguiente sección.
S(f) es la transformada de Fourier de la señal de entrada, * denota el complejo conjugado, t_1 es el tiempo en que se observa el máximo de la señal, Ga es la ganancia del filtro (generalmente 1).

El ruido que acompaña la señal se asumirá estacionario y con espectro uniforme, es decir, trataremos con ruido blanco. En caso de no respetar estas aproximaciones la ecuación 4.2.1 se deberá modificar.
La función de respuesta en frecuencia del filtro apareado es el conjugado del espectro de la forma de onda recibida, excepto por un corrimiento de fase dado por $\exp(-j2\pi ft_1)$, que varía uniformemente con la frecuencia. Esto causa en retraso en el tiempo, necesario para la realización física, ya que naturalmente no puede existir respuesta antes de que se aplique una señal a la entrada.
La función de transferencia pude escribirse en términos de amplitud y fase, al igual que el espectro de señal que llega, de esta forma la ecuación 4.2.1 queda,

$$|H(f)|\exp(-j\phi_m) = |S(f)|\exp[j(\phi_s(f) - 2\pi.f.t_1)] \qquad \text{Ec.4.2.2}$$

$$|H(f)| = |S(f)| \qquad \text{Ec.4.2.3}$$

$$\phi_m = -\phi_s - 2\pi.f.t_1 \qquad \text{Ec.4.2.4}$$

Es decir, las magnitudes del espectro del filtro apareado y el espectro de la señal es la misma. Pero la fase del filtro es la negativa de la señal más un desfasaje proporcional a la frecuencia.
A veces se especifica al filtro apareado con su respuesta al impulso, equivalente a la transformada inversa de Fourier de H(f).

$$h(t) = \int_{-\infty}^{\infty} H(f)\exp(j2\pi.f.t)df \qquad \text{Ec.4.2.5}$$

Sustituyendo la ecuación 4.2.1 en la ecuación 4.2.5

$$h(t) = \int_{-\infty}^{\infty} G_a.S^*(f)\exp(-j2\pi.f.t_1)\exp(j2\pi.f.t)df \qquad \text{Ec.4.2.6}$$

$$h(t) = \int_{-\infty}^{\infty} G_a.S^*(f)\exp(-j2\pi.f(.t_1 - t))df \qquad \text{Ec.4.2.7}$$

Dónde $S^*(f) = S(-f)$. Por lo tanto:

$$h(t) = G_a\int_{-\infty}^{\infty} S(f)\exp(2\pi.f(t_1 - t))df = G_a s(t_1 - t) \qquad \text{Ec.4.2.8}$$

Este resultado es interesante. Muestra que la respuesta al impulso del filtro apareado es una imagen de la señal de entrada retrasada en el tiempo.
La propiedad más importante de este tipo de filtro es que sin importar la forma que tenga la onda de entrada, la máxima relación entre la potencia pico de la señal

y la potencia media de ruido es dos veces la energía contenida en la señal dividida por la potencia de ruido por hertz de ancho de banda.
El filtro adaptivo mencionado en capítulos anteriores presenta buenas características prácticas pero no son aplicables en sistemas de radar por motivos que serán expuestos en el capítulo de Resultados y Conclusiones. En base a ello y ya que fue propuesto como objetivo del presente trabajo final, se desarrolla a partir de este capítulo un filtro digital basado en la teoría del filtro apareado.
Este tipo de filtros se basa en la relación que existe entre la señal proveniente del medio (eco) y la que fue enviada anteriormente para la detección del objetivo. Toda señal adicional que no proviene del trasmisor debería ser atenuada ya que no posee correlación con la primera. Se demostrará en una primera instancia con simulaciones que este filtro es capaz de detectar un determinado objetivo cuando la señal está sometida a diferentes fuentes de ruido.
Finalmente es este tipo de filtro el que se implementará, con el cual se obtendrán los resultados, conclusiones y observaciones finales.

4.3 OBTENCIÓN DE LA RESPUESTA EN FRECUENCIA DEL MATCHED FILTER

Hay que demostrar que la función de transferencia es

$$H(j) = G_a S^*(j).e^{(-j2n.f.t1)} \qquad \text{Ec.4.3.1}$$

Cuando a la entrada se presenta ruido blanco (espectro de densidad uniforme, no es función de la frecuencia).

La relación a maximizar es la siguiente:

$$R_f = \frac{[s_{0(t)}]^2{}_{\max}}{N} \qquad \text{Ec.4.3.2}$$

El numerador indica el máximo valor de salida, y el denominador el ruido presente a la salida del receptor.

$$|s_0(t)| = \left| \int_{-\infty}^{\infty} S(f).H(f).e^{j2\pi.f.t} df \right| \qquad \text{Ec.4.3.3}$$

La potencia media de ruido a la salida del mismo filtro es:

$$N = \frac{N_0}{2} \int_{-\infty}^{\infty} |H(f)|^2 .df \qquad \text{Ec.4.3.4}$$

Donde No es la potencia de ruido por hertz, a la entrada.
Asumiendo que el valor máximo a la salida del filtro ocurre en t_1, la ecuación 4.3.2 queda:

$$R_f = \frac{\left| \int_{-\infty}^{\infty} S(f).H(f).e^{j2\pi.f.t1} df \right|^2}{\frac{N_0}{2} \int_{-\infty}^{\infty} |H(f)|^2 df}$$ Ec.4.3.5

La desigualdad de schwarz establece lo siguiente:

$$|\langle P,Q\rangle|^2 \leq |\langle P,P\rangle| . |\langle Q,Q\rangle|$$ Ec.4.3.6

$$\left| \int P*Q.dx \right|^2 \leq \int P*P.dx \cdot \int Q*Q.dx$$ Ec.4.3.7

La igualdad se establece si se mantiene una relación lineal entre P y Q, es decir:

P = k.Q Ec.4.3.8

Físicamente significa que P y Q son paralelas o una de las dos es igual a cero. En este caso se tiene:

$$P* = S(f).e^{j2\pi.f.t1}$$ Ec.4.3.9

$$Q = H(f)$$ Ec.4.3.10

Sabiendo que,

$$\int P*P.dx = \int |P|^2 dx$$ Ec.4.3.11

Volviendo a la ecuación de Rf, y sustituyendo,

$$R_f \leq \frac{\int_{-\infty}^{\infty} P*P.df \cdot \int_{-\infty}^{\infty} Q*Q.df}{\frac{N_0}{2} \int_{-\infty}^{\infty} |Q|^2 .df}$$ Ec.4.3.12

$$R_f \leq \frac{\int_{-\infty}^{\infty} |P|^2 .df \cdot \int_{-\infty}^{\infty} |Q|^2 .df}{\frac{N_0}{2} \int_{-\infty}^{\infty} |Q|^2 .df}$$ Ec.4.3.13

$$R_f \leq \frac{\int_{-\infty}^{\infty} |P|^2 .df}{\frac{N_0}{2}}$$ Ec.4.3.14

El teorema de Parseval establece que:

$$\int |F(f)|^2 df = \int f^2(t) dt$$

Físicamente esta ecuación indica que la energía contenida en la señal f(t) es la misma que la energía contenida en su transformada de Fourier a lo largo de todas sus componentes. Luego,

$$R_f \leq \frac{\int_{-\infty}^{\infty} p^2(t).dt}{\frac{N_0}{2}} = \frac{2E}{N_0} \qquad \text{Ec.4.3.15}$$

La función que maximiza la relación R(f) se obtiene de la condición de igualdad P=kQ, es decir que:

$$S^*(f).e^{-j2\pi.f.t1} = k.H(f) \qquad \text{Ec.4.3.16}$$

$$\boxed{H(f) = G_a.S^*(f).e^{-j2\pi.f.t1}} \quad , \quad G_a = \frac{1}{k} \qquad \text{Ec.4.3.17}$$

4.4 SIMULACION DEL MATCHED FILTER EN MATLAB

Se utilizaron funciones provistas por MatLab® para generar la señal trasmitida, para el cálculo de los coeficientes determinantes del filtro y la representación de los resultados.
Se simuló el funcionamiento del filtro en una situación ideal (ausencia de ruido), y luego en presencia de ruido y señales espurias que pueden estar presentes en el medio real de funcionamiento.
Para ello se generó una señal de radar tal como la descripta en la sección 2.12. La figura 4.4.1 ilustra el pulso de radar mencionado.

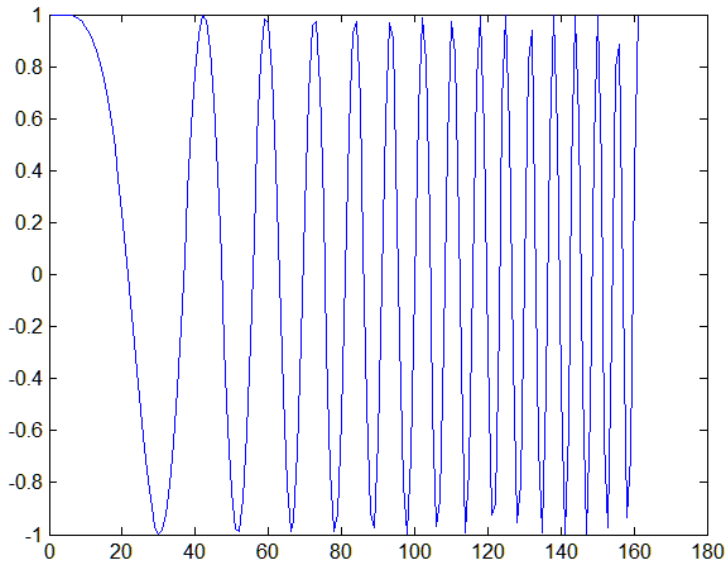

Fig.4.4.1

Como establece la teoría, la respuesta impulsiva del filtro es la misma señal invertida y con un retardo temporal. Utilizando los coeficientes calculados según lo dicho, el resultado obtenido de la convolución suponiendo una señal sin ruido es:

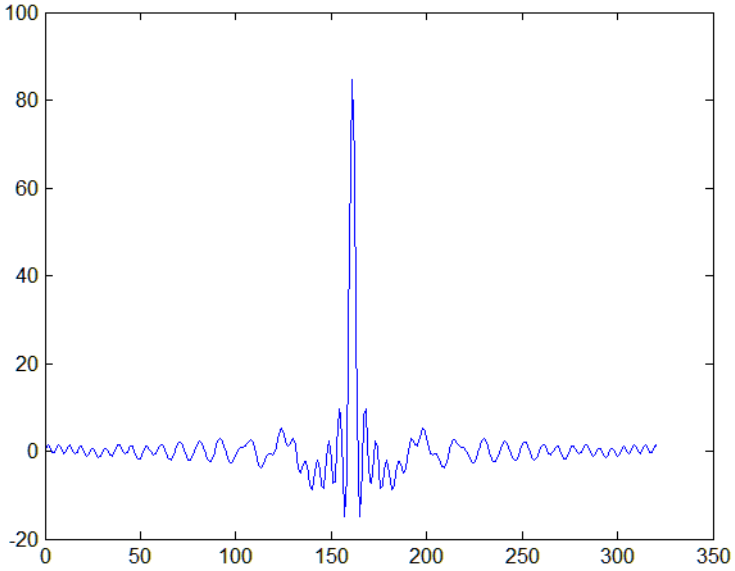

Fig.4.4.2

Puede apreciarse que la respuesta del filtro es un pulso comprimido, cuya posición en el eje temporal es 160*ts (siendo ts = 1/fs = 20ns), es decir 3.2 µs correspondiéndose con la duración del pulso radar. Esto se debe a que la señal ingresada al filtro no tiene retardo. Así mismo puede observarse la gran amplitud del pulso comprimido.
Luego, se generó una señal aleatoria (Ruido gaussiano) con valores de distribución normal con media 0 y desviación estándar 1, la cual puede observarse en la figura 4.4.3 a continuación:

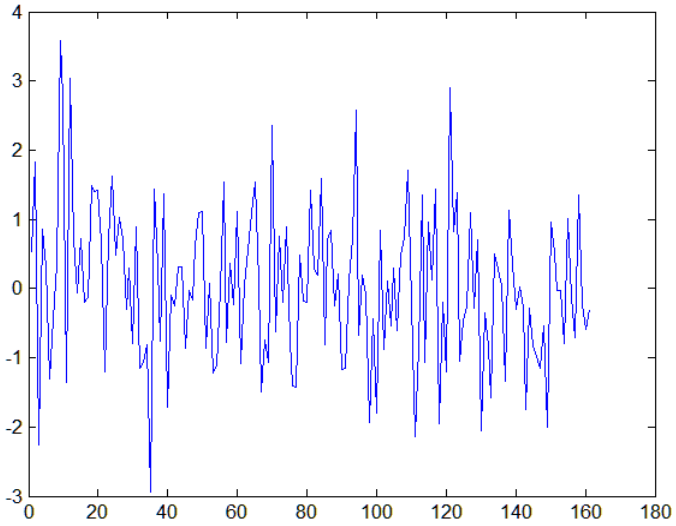

Fig.4.4.3

Sumando la señal anterior, con la señal de radar, se obtiene la señal de entrada mostrada en la figura 4.4.4 a continuación.

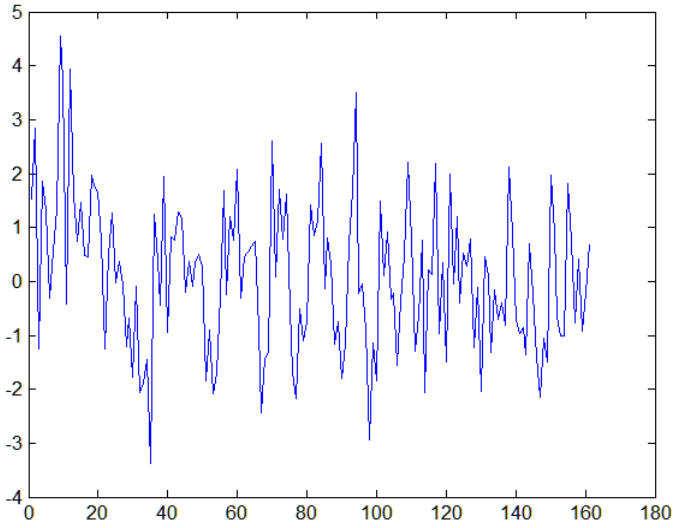

Fig.4.4.4

Ingresando la señal generada anteriormente al filtro apareado, la cual tiene una SNR = -11.07 dB, se obtiene la siguiente salida:

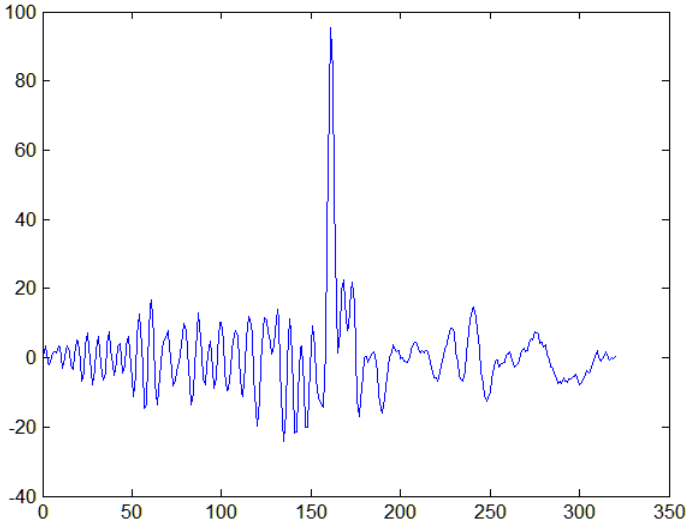

Fig.4.4.5

En la cual puede observarse que el pulso comprimido es de gran amplitud respecto al piso de ruido, obteniendo una SNR = 9.81 dB.

A continuación se detallan las mismas pruebas simulando un objetivo alejado, cuyo retardo del eco es 4 µs, tal como se muestra en la figura 4.4.6:

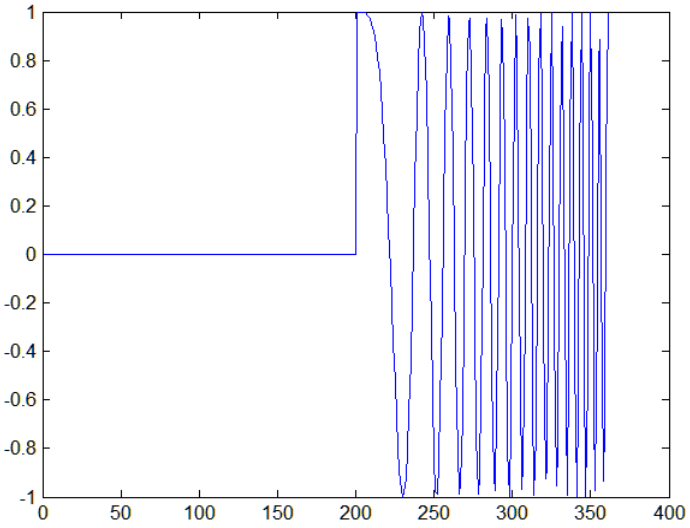

Fig.4.4.6

Esta señal eco recibida genera a la salida del filtro apareado la siguiente señal:

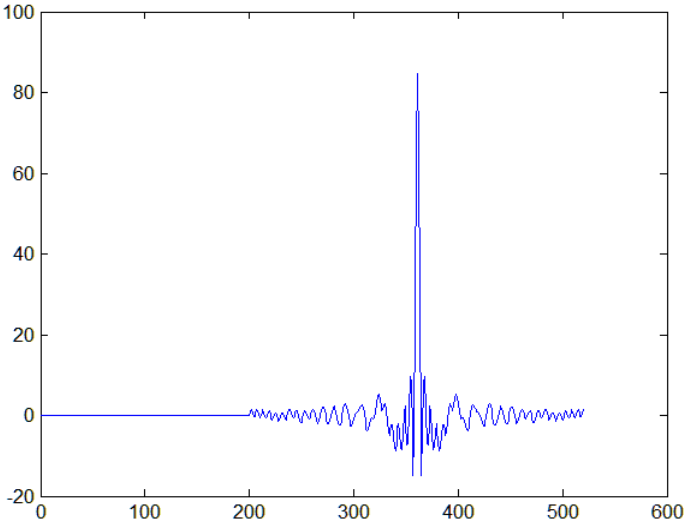

Fig.4.4.7

En la figura anterior puede observarse que el pulso comprimido se ubica en el eje temporal en el valor 360, el cual se corresponde con 360*ts = 7,2 µs, siendo 3.2 µs la duración del pulso radar, lo que implica un retardo del eco de 4 µs.

4.5 DISEÑO Y SIMULACIÓN DEL MATCHED FILTER EN MATLAB/SIMULINK

Utilizando las herramientas provistas por Matlab/Simulink, se diseñó el filtro apareado a partir de bloque FIR. El modelo diseñado se muestra a continuación en la figura 4.5.1:

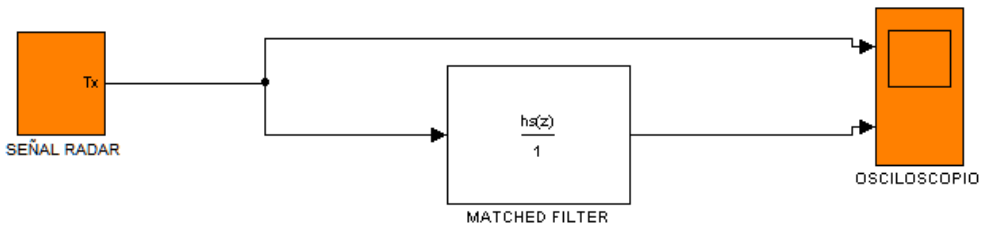

Fig.4.5.1 – Filtro apareado diseñado en Matlab/SIMULINK.

El bloque de SEÑAL RADAR, corresponde con el modelo de transmisor abordado en la sección 2.12.
En primera instancia se estimuló el filtro con la señal radar sin ruido aditivo, cuya salida resultante se muestra en la figura 4.5.2.

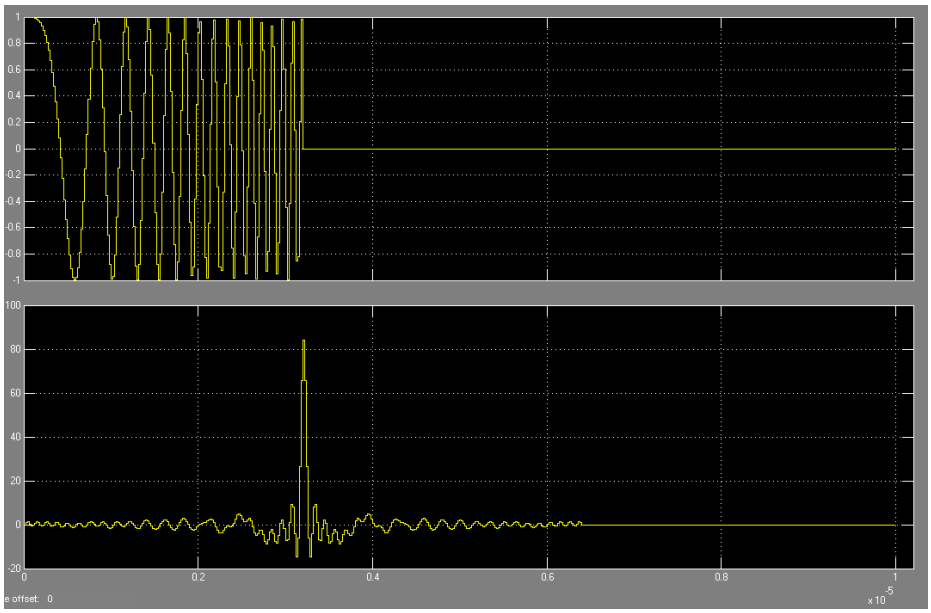

Fig.4.5.2 – Señal de radar y Respuesta del filtro apareado respectivamente.

Se puede observar que la señal de radar no tiene retardo y por ello el pulso comprimido tampoco muestra un retardo.
Luego se procedió a agregar ruido gaussiano de distribución normal con media igual cero y varianza 1, y un retardo de 5 µs en la señal ingresada al filtro apareado, obteniendo los siguientes resultados:

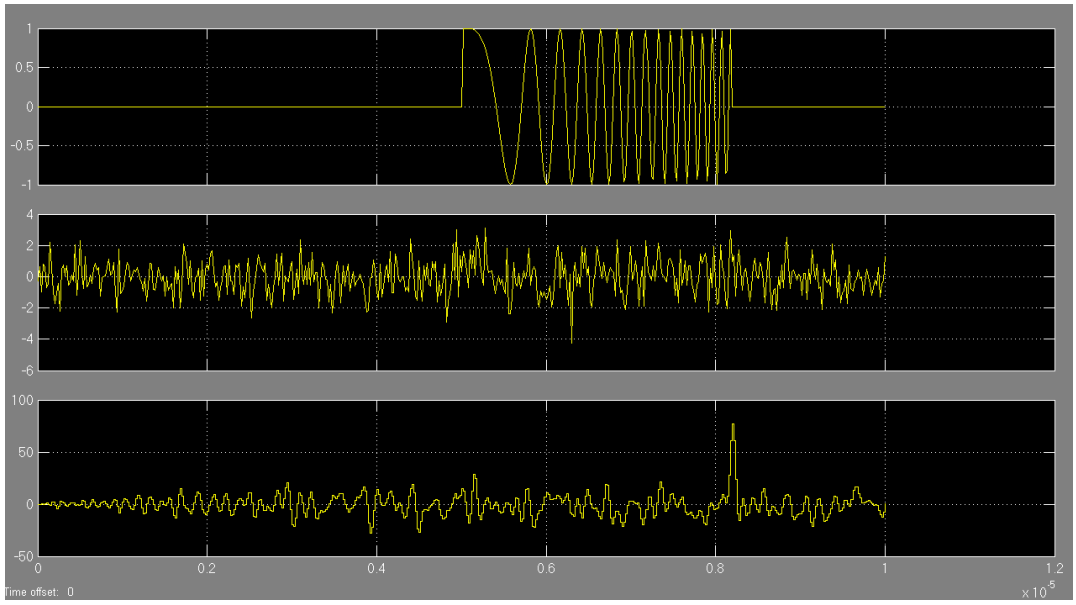

Fig.4.5.3 – Señal de radar retardada, Señal más ruido Gaussiano y Respuesta del filtro apareado.

Apreciando las escalas puede observarse que la señal obtenida a la salida del filtro tiene un gran SNR y además que el pulso comprimido muestra un retardo de 5 µs tal lo esperado.

TECNICAS DE VENTANEO

A continuación se introducen las pruebas realizadas de la técnica de ventaneo de Hamming, para cual se simuló seis ecos con retardos de 3 µs, 4 µs, 4.2 µs, 7 µs, 10 µs y 10.5 µs respectivamente.

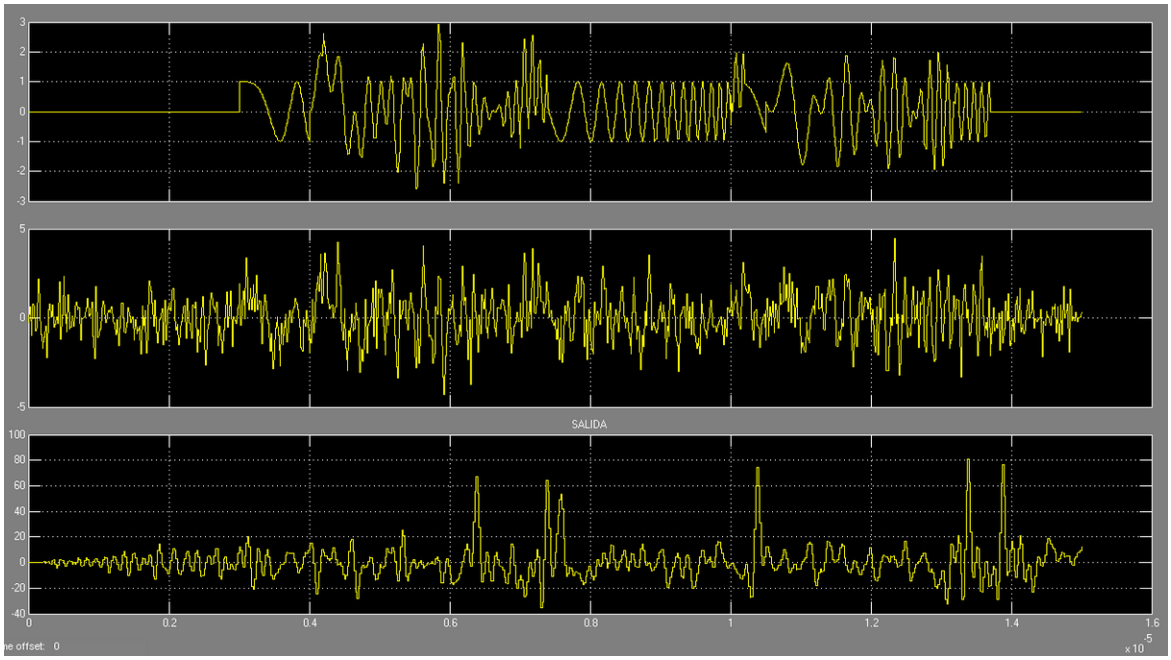

Fig.2.13.3 – a) Señal recibida sin ruido b) Señal recibida con ruido c) Respuesta del filtro apareado sin ventaneo.

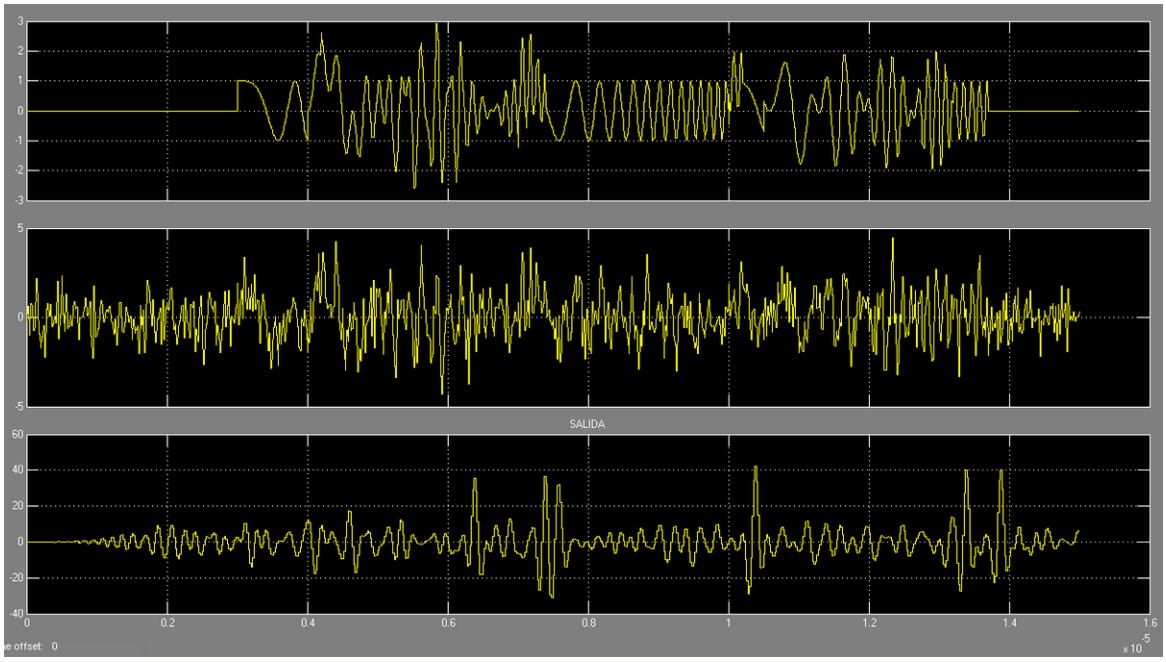

Fig.2.13.4 – a) Señal recibida sin ruido b) Señal recibida con ruido c) Respuesta del filtro apareado con ventana de Hamming.

En las figuras anteriores se muestran los mejores resultados obtenidos en el modelo implementado de filtro adaptado logrados utilizando la ventana de Hamming. Se habla de mejores resultados comparando con aquellos obtenidos con otro tipo de ventana ya que, como se observa, las amplitudes de los pulsos comprimidos durante la recepción disminuye considerablemente, por lo que no se pudo afirmar que se logra una mejor respuesta aplicando la técnica de ventaneo.
Se prefiere optar por el tratamiento del eco sin ningún tipo de ventana, basando la decisión en criterios de maximización en la relación señal a ruido de la señal filtrada.
Como ya se aclaró, el objetivo del trabajo es corroborar la performance del filtro adaptado inmerso en un medio nocivo. La optimización en la detección de blancos próximos queda fuera del alcance de dicho trabajo.

CAPÍTULO 5: IMPLEMENTACIÓN

5.1 INTRODUCCIÓN

El proceso de diseño en FPGA comienza con las especificaciones funcionales del sistema digital a implementar, y culmina en la programación del dispositivo con los archivos generados durante el proceso. Este involucra una serie de etapas, las cuales se detallan en la siguiente figura.

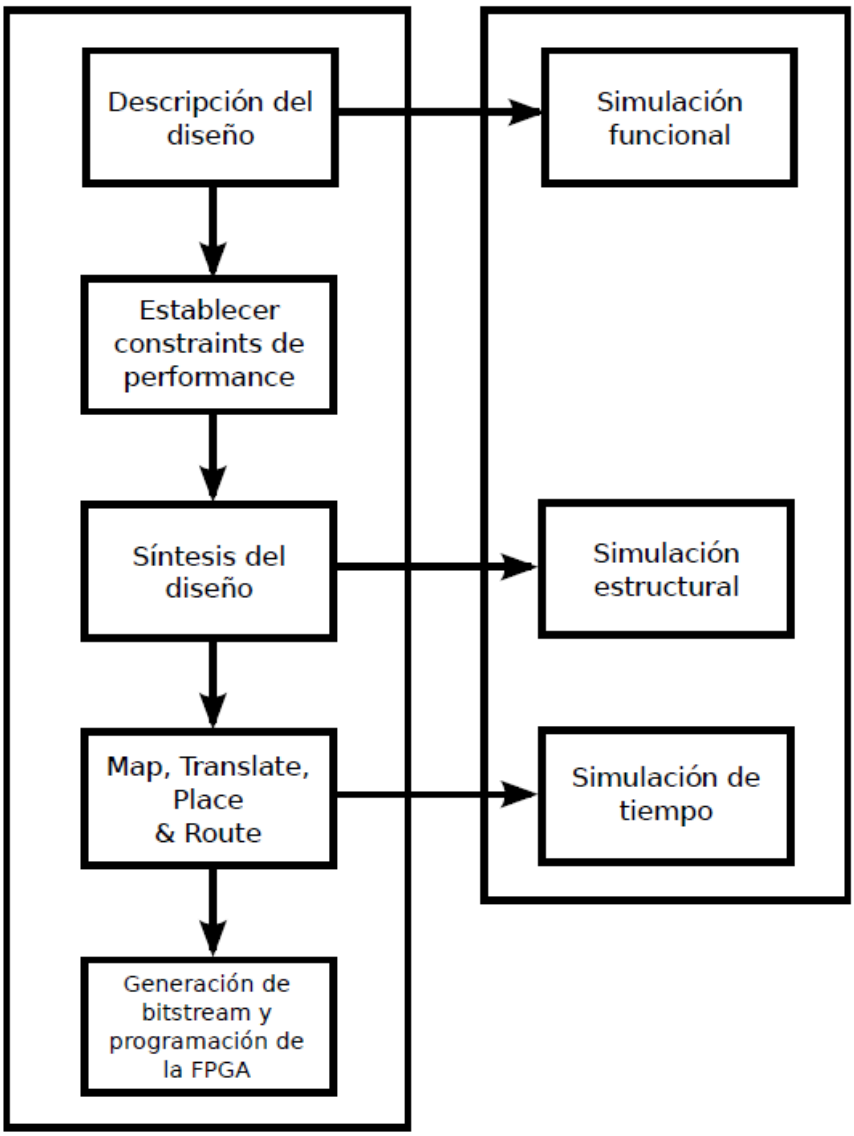

Fig 4.2.1 - Proceso de diseño en FPGA.

La primera etapa implica describir el diseño en lenguaje HDL o mediante un editor de esquemáticos. Para este trabajo en particular, el código HDL se generó automáticamente por System Generator en base al modelo creado para el desarrollo de cada componente funcional y del sistema completo. Esto permitió la

obtención de un modelo manipulable desde el punto de vista práctico, totalmente flexible frente a cambios de parámetros que comprueban el funcionamiento y ayudan a obtener un comportamiento eficaz del sistema completo.
Una vez concluida esta etapa se procede a realizar una simulación funcional del circuito donde se verifica que el diseño se comporte de acuerdo a lo esperado.
Corroborado el funcionamiento del sistema, se debe incluir una serie de constraints (restricciones de diseño) que sirven en el resto del proceso para asegurarse que el sistema cumpla con los requisitos de performance establecidos. Estos constraints pueden ser de tiempo, donde se indica la frecuencia máxima a la que el diseño debe funcionar, constraints de área, donde se especifican regiones a usar en la FPGA por alguna parte del diseño y constraints de asignación de pines, donde las entradas y salidas del sistema se asignan a pines de la FPGA. Estas restricciones son definidas dentro de System Generator.
Una vez establecidos estos requerimientos, se procede al proceso de síntesis del circuito mediante herramientas específicas como Xilinx ISE [15].
Esto implica una traducción del comportamiento del sistema mediante ecuaciones, a un nivel de compuertas lógicas que se corresponde con los bloques de lógica dentro de la FPGA, generando un archivo denominado netlist, el cual describe la conexión entre dicha lógica.
Finalizado este proceso se obtiene un estimado de la máxima frecuencia a la que el sistema puede funcionar, como así también los recursos consumidos en la FPGA. Si a este momento los resultados obtenidos difieren de los parámetros necesarios para concluir con un resultado óptimo, se deberán realizar modificaciones en el diseño para conseguir los requisitos establecidos en el proceso de diseño.
A partir de esta etapa es posible de realizar un tipo de simulación denominada estructural, donde se examina el comportamiento del sistema una vez sintetizado. A su vez a partir de este punto, las simulaciones realizadas permiten la validación de la descripción de hardware realizada en el primer paso, ya que el diseño fue traducido a componentes lógicos a emplear en la FPGA.
Si los resultados obtenidos en la etapa anterior son satisfactorios, se continúa con el proceso de implementación. Este proceso comprende las siguientes etapas:

- Unir las diferentes netlists del sistema en un único archivo. Este proceso de denomina Translate.

- Agrupar la lógica descripta en la netlist y traducirlas a componentes físicos de la FPGA. Esta etapa de denomina Map.

- Ubicar los componentes dentro de la FPGA y realizar las conexiones entre ellos. Este proceso se denomina Place and Route.

Una vez culminada la etapa anterior, es posible obtener reportes de tiempo generados por las herramientas de implementación (Xilinx ISE), en donde se detalla si los constraints de tiempo fueron cumplidos o si se necesita realizar modificaciones en algunos componentes del sistema.
Además se obtienen reportes de los recursos finales empleados por el diseño.

En este punto es posible realizar una simulación de tiempos, la cual difiere de las anteriores por el hecho de que se incluyen los retrasos de las señales entre los componentes. Los resultados obtenidos en esta simulación son los más cercanos a los que se obtendrá en la verificación del sistema implementado.
Si hay conformidad con los resultados obtenidos, se procede a hacer uso de las aplicaciones de implementación para producir un archivo denominado bitstream el cual será responsable de programar la FPGA.
Para la comprobación del diseño se realizó un proceso de verificación y validación informal, el cual comprende los siguientes pasos:

- Proceso de Auditoría: se detalla cómo se realizan las simulaciones y por qué motivo las mismas son válidas para verificar que las características mencionadas en la simulación cumplen con las especificaciones establecidas.

- Proceso de Testeo o Simulación: se procede a explicar y mostrar las simulaciones realizadas, fundamentando a través de gráficos, tablas o comparaciones donde se aprecien los resultados obtenidos durante la simulación.

Por medio de estos pasos fue posible realizar un proceso sencillo de verificación del diseño propuesto, para asegurarse que el problema se está resolviendo correctamente y que el sistema cumple con las características establecidas. La verificación permitió entonces comprobar que la implementación del sistema en lenguaje VHDL sea una implementación correcta del modelo de simulación realizado.
Por otra parte, en procesos de diseño y desarrollo se debe corroborar que el proyecto sea realmente lo que proponía en una primera instancia, que lo implementado cumpla con las características necesarias para solucionar el problema a partir del cual surge la idea o tema del trabajo.
Es por ello que la última etapa es la Validación. Esta se aplica a aquellos procesos que buscan determinar si un modelo es correcto o no respecto al sistema real, es decir en este caso, se fundamenta si el sistema desarrollado es apto para el propósito para el cual fue diseñado, en sus condiciones de ser empleado como un filtro apareado aplicado a ECCM.
Mediante estos pasos se pudo establecer la confianza de que el diseño realizado es adecuado para el propósito determinado.

5.2 DISEÑO EN SYSTEM GENERATOR

Para el desarrollo de este bloque funcional, se utilizaron librerías provistas por System Generator para la implementación de filtro FIR en arquitecturas Transpuesta [12].
En la tabla 5.2.1 se resumen algunos parámetros de diseño del filtro apareado implementado.

Parámetro	Valor
Arquitectura	Transpuesta
Número máximo de Coeficientes (M)	161
Número de Canales	1
Entrada efectiva de muestro (F_{CLK}/F_S*Número de Canales)	2
Factor de diezmado	1
Coeficientes	Fijos

Tabla 5.2.1 - Parámetros de diseño de filtro apareado.

Los constraints de tiempo fueron establecidos en System Generator para asegurarse que el diseño funcione a la frecuencia máxima establecida.
Los componentes provistos por System Generator poseen la capacidad de indicar el tipo de optimización (de área o tiempo) requerido en el proceso de síntesis e implementación. Dado que el sistema opera a frecuencias poco elevadas, el criterio de optimización elegido fue el de área, indicando a la herramienta de síntesis e implementación que deberá realizarse el mayor esfuerzo en el área de lógica utilizada en la FGPA.
En System Generator, los diseños generalmente están basados en un único clock. Es tarea de la herramienta derivar los clock enable (CE) necesarios para el control de sistemas con múltiples frecuencias de muestreo, como es en este caso en particular. Esto ayuda a que los constraints de tiempo se puedan cumplir más fácilmente.
Siguiendo la tabla 5.2.1, el primer factor de diseño fue la arquitectura a utilizar. Como se explicó en la sección 3.3, la arquitectura transpuesta optimiza el funcionamiento del filtro [12], mejorando la performance del mismo con respecto a la arquitectura directa.
La cantidad máxima de coeficientes en los filtros FIR está dada por la cantidad de multiplicadores embebidos en la FPGA a utilizar, ya que las librerías utilizadas en System Generator para la implementación de filtros FIR hacen uso de los mismos.
Debido a que la frecuencia de clock es el doble que la frecuencia de muestreo se utilizó reúso de hardware por un factor igual a 2. Esto es, que en lugar de utilizar un filtro con M multiplicadores en paralelo se utilicen M/2 multiplicadores en paralelo, realizando el procesamiento en 2 ciclos, disminuyendo de esta forma los recursos utilizados en la FPGA. Esta característica es de suma importancia en el desarrollo del proyecto, permitiendo reducir el hardware utilizado.
Ya que la FPGA utilizada para este diseño dispone de 136 multiplicadores embebidos [11], el número máximo de coeficientes fue fijado en 161, pero con un factor de reúso igual a 2, lo que implica 81 coeficientes por ciclo, cuantificados con 16 bits, de los cuales 14 de ellos son utilizados para representar la parte fraccionaria.
En la figura 5.2.1 se puede observar el modelo de simulación del filtro apareado realizado en Simulink/System Generator.

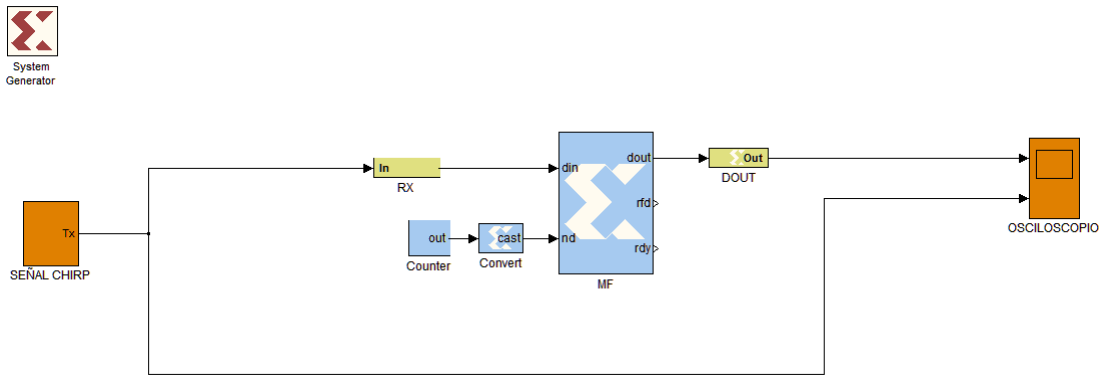

Fig.5.2.1 - Modelo de simulación de filtro apareado.

Las entradas y salidas del bloque corresponden a los puertos din y dout respectivamente.
La señal de entrada se representa por palabras de 16 bits con 14 bits de parte fraccionaria, con el objetivo de cubrir el mayor rango dinámico posible (Ver sección 3.7). Así mismo la señal de salida se representa con 16 bits con redondeo simétrico a cero, evitando de esta manera cualquier desbordamiento [12].
La señal nd activa por alto habilita el ingreso de una nueva muestra en la entrada din.
El filtro posee salidas para el control de flujo de datos, las cuales indican cuando una nueva muestra está disponible en el puerto de salida (rdy) y cuando el filtro está disponible para aceptar una nueva muestra en su entrada (rfd).
En la figura 5.2.2 se ilustra el ciclo de procesamiento del filtro apareado.

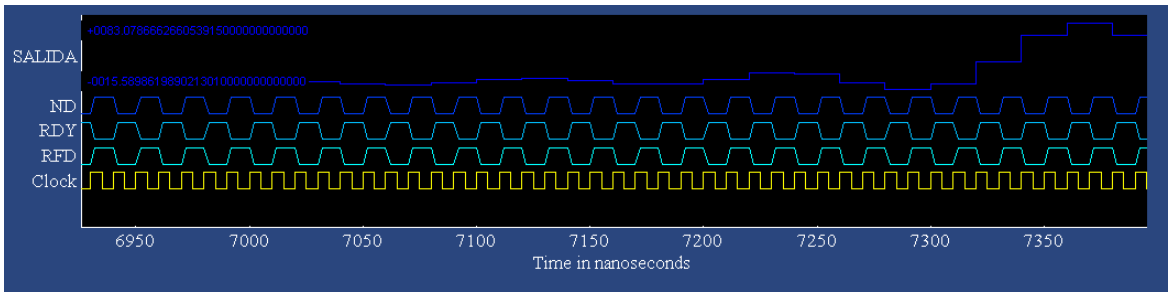

Fig.5.2.2 – Ciclo de trabajo del filtro apareado.

Para realizar las pruebas de funcionamiento, se utilizaron señales de radar descriptas en la sección 2.12, las cuales permitieron observar el proceso de filtrado de manera correcta.
En primera instancia se simulo utilizando la señal radar antes mencionada ingresada al filtro sin retardo y sin ruido aditivo. Luego se simuló con dos pulsos de radar ingresando al filtro apareado sin ruido aditivo con un retardo de 5 µs y 9 µs respectivamente.
En la figura 5.2.3 se ilustran los resultados obtenidos en la segunda simulación.

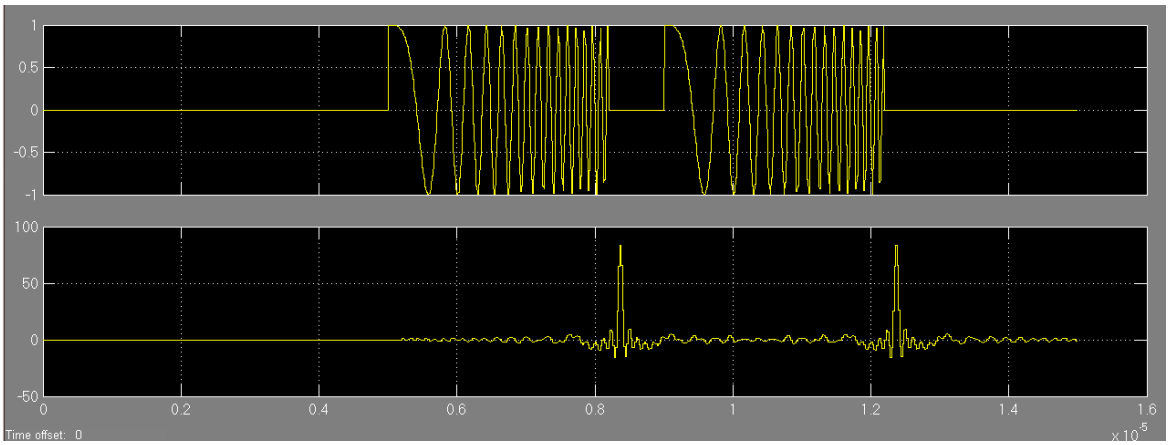

Fig.5.2.3 – Simulación del filtro apareado con dos ecos, cuyo retardo son de 5 µs y 9 µs respectivamente.

Hay que contemplar que el filtro genera 0.8 µs de retardo por procesamiento.
En la figura anterior puede observarse que la posición temporal de los pulsos comprimidos es igual al retardo del pulso de radar presente en la entrada del filtro, desplazado 4 µs que es el tiempo de procesamiento del mismo más la duración del pulso radar.
La figura 5.2.4 muestra la simulación con dos pulsos de radar ingresando al filtro apareado con un retardo de 4 µs y 7 µs respectivamente y con ruido gaussiano de distribución normal, varianza igual a 1 y media igual a 0.

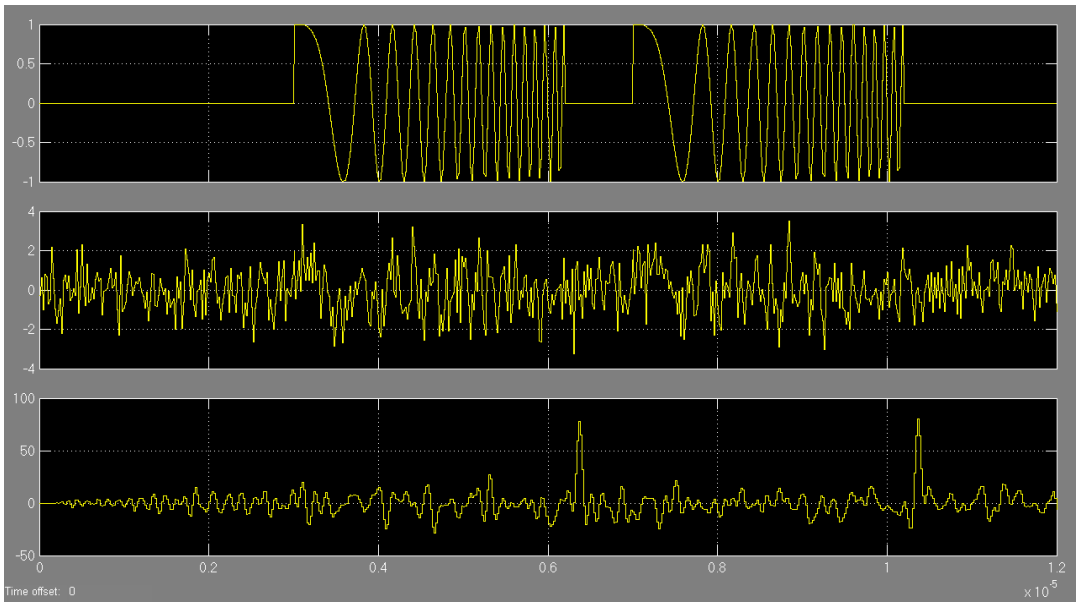

Fig.5.2.4 – Simulación del filtro apareado con dos ecos, cuyo retardo son de 4 µs y 7 µs respectivamente más ruido gaussiano.

En dicha imagen puede apreciarse que la SNR de la señal de entrada es de -10.27 dB. A la salida del filtro podemos observar que la SNR es de 10.2 dB, con lo cual se verifica que se maximiza la SNR en la respuesta del filtro apareado.

Si bien no es del alcance de este proyecto desarrollar un receptor de radar completo, se decidió agregar una etapa que permita calcular según el retardo del eco recibido, la distancia a la cual se encuentra el blanco, a los fines de visualizar una de las aplicaciones del filtro apareado. Para ello se utilizó un detector de envolvente, con un umbral de detección de 63.98 dBm (Ver sección 2.9). La figura 5.2.5 muestra el diseño final con el detector incluido.

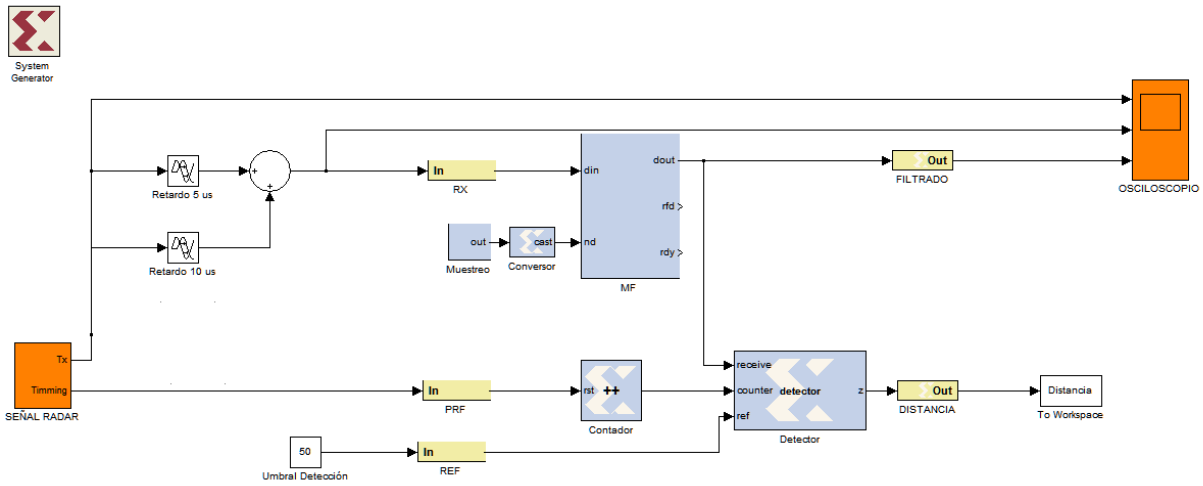

Fig.5.2.5 – Modelo completo de simulación del filtro apareado.

Luego se realizó la simulación del filtro apareado con el detector de envolvente incluido. Para dicha prueba se utilizaron dos pulsos de radar retardados 3 µs y 10 µs respectivamente. Observando la variable "Distancia" en el workspace se verificó que arrojó los valores 453 m y 1506 m respectivamente.

5.3 SIMULACION HDL

Siguiendo con los procesos de simulación detallados al comienzo del capítulo, se procedió a la traducción del modelo del filtro apareado a lenguaje VHDL para ser simulado en el tiempo mediante su descripción en hardware.
En la figura 5.3.1 se puede apreciar el diagrama esquemático del filtro apareado implementado en VHDL.

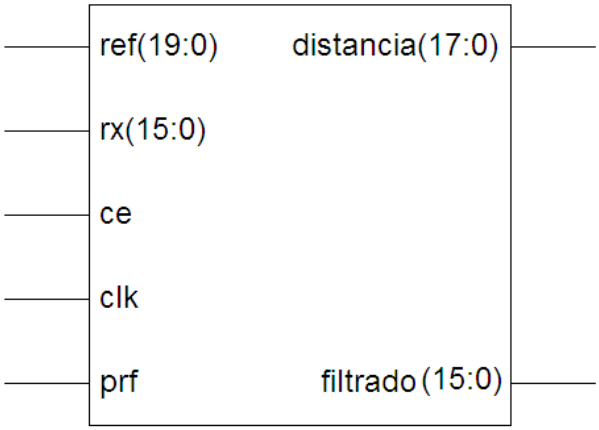

Fig.5.3.1 - Diagrama esquemático del filtro apareado implementado.

De acuerdo con los resultados obtenidos en los procesos de simulación, esta implementación del filtro apareado cumple con las características de diseño establecidas, y valida su utilización propuesto en este TFG.
En la figura 5.3.2 se ilustra los recursos consumidos por el filtro apareado luego del proceso de implementación.

Device Utilization Summary				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	4,033	27,392	14%	
Number of 4 input LUTs	6,611	27,392	24%	
Logic Distribution				
Number of occupied Slices	3,687	13,696	26%	
Number of Slices containing only related logic	3,687	3,687	100%	
Number of Slices containing unrelated logic	0	3,687	0%	
Total Number of 4 input LUTs	6,632	27,392	24%	
Number used as logic	6,594			
Number used as a route-thru	21			
Number used as Shift registers	17			
Number of bonded IOBs				
Number of bonded	96	556	17%	
Number of RAMB16s	1	136	1%	
Number of MULT18X18s	81	136	59%	
Number of BUFGMUXs	1	16	6%	

Fig.5.3.2 - Recursos utilizados en la implementación del filtro apareado.

En dicha imagen se resume el número de elementos de lógica consumidos por el diseño, como flip flops, LUTs, multiplicadores embebidos (MULT18x18s) y bloques de memoria RAM (RAMB16s). Además se detalla la función que cumple cada LUT en el proyecto, es decir, si son utilizadas para almacenar lógica booleana, registros de desplazamiento o memoria RAM.
El elemento BUFGMUX es un buffer de clock implementado por la herramienta de Xilinx, el cual permite multiplexar entre dos señales de clock disponibles en su entrada.
Los valores estimados de la potencia consumida por el diseño se obtuvieron mediante la aplicación XPower Analyzer, incluida en el paquete de herramientas Xilinx ISE. En la figura 5.3.3 se detallan con mayor precisión.

Name	Value	Used	Total Available	Utilization (%)
Clocks	0.01112 (W)	1	---	---
Logic	0.01424 (W)	6627	27392	24.2
Signals	0.05093 (W)	14936	---	---
IOs	0.02672 (W)	96	588	16.3
BRAMs	0.00010 (W)	1	136	0.7
MULTs	0.01083 (W)	81	136	59.6
Total Quiescent Power	**0.10313 (W)**			
Total Dynamic Power	**0.11395 (W)**			
Total Power	**0.21708 (W)**			
Junction Temp	**25.0 (degrees C)**			

Fig.5.3.2 - Valores estimados de consumo de energía del diseño.

El parámetro “Total Quiescent Power” indica el consumo de potencia del sistema en funcionamiento normal, mientras que “Total Dynamic Power” da una representación de los valores pico de potencia que puede llegar a consumir el circuito.

Luego de considerar las características del sistema se procedió a realizar la simulación por medio de ModelSim.

La figura 5.3.3 muestra el resultado de la simulación en ModelSim utilizando la misma señal de radar utilizada en la primera simulación en System Generator.

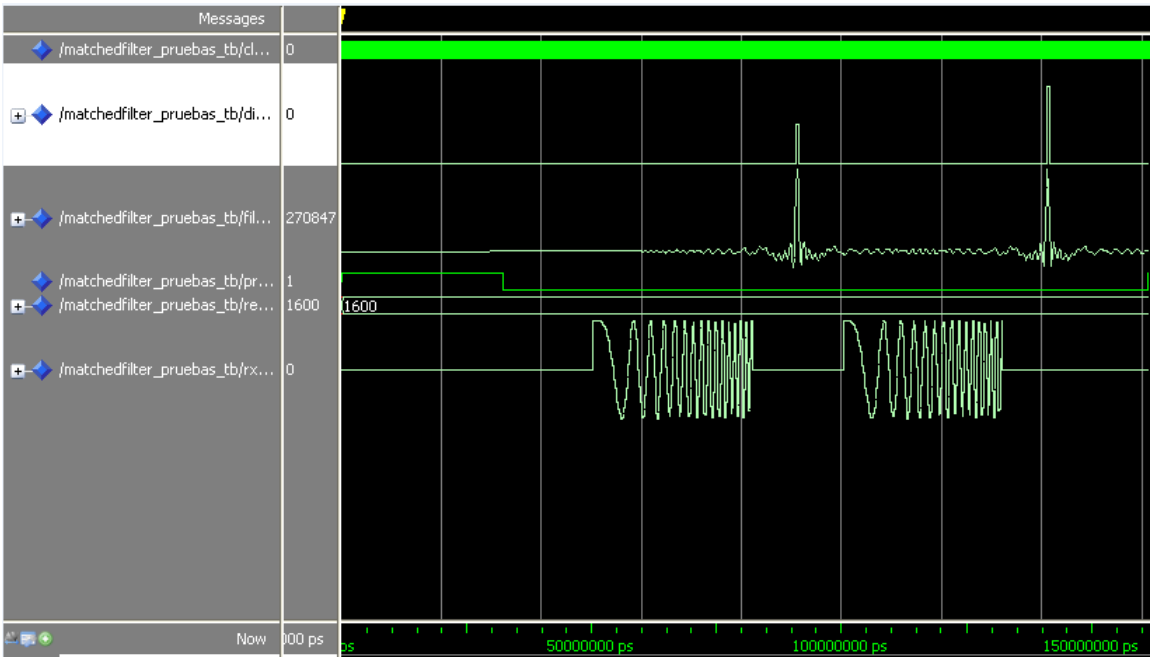

Fig.5.3.3 – Resultado de la simulación del filtro apareado en ModelSim.

El tipo de testbench utilizado realizó una comparación entre los resultados obtenidos en la simulación en Simulink y los obtenidos luego del proceso de

síntesis e implementación. Estos resultados son mostrados en la consola de Modelsim.

En este caso, de 1501 muestras procesadas, no hubo discrepancias. Para demostrar esto, la figura 5.3.4 ilustra una porción del reporte generado por Modelsim.

Por medio de esta simulación se pudo verificar el correcto funcionamiento del sistema.

Además, el reporte entregado por el simulador permitió verificar la correlación entre el modelo realizado a nivel de sistemas y su correspondiente traducción a nivel de compuertas lógicas.

```
# Beginning comparisons for instance matchedfilter_pruebas_distancia
#
#
# Beginning comparisons for instance matchedfilter_pruebas_filtrado
#
#
# ** Simulation summary for instance matchedfilter_pruebas_distancia:
#    Samples Processed: 1501
#            - Checked: 1501 (100.0%)
#            - Ignored: 0 (0.0%) Don't Cares
#    Test completed with no errors.
#
# ** Simulation summary for instance matchedfilter_pruebas_filtrado:
#    Samples Processed: 1501
#            - Checked: 1501 (100.0%)
#            - Ignored: 0 (0.0%) Don't Cares
   Test completed with no errors.
```

Fig. 5.3.4: Reporte generado por Modelsim.

5.4 CO-SIMULACIÓN POR HARDWARE

La co-simulación por hardware es un método de simulación de sistemas de hardware el cual permite, una vez puesto en un entorno de simulación y modelado el diseño, implementar el sistema en un dispositivo e interactuar con el software de simulación para contrastar, verificar y validar dicho sistema.

El proceso de co-simulación por hardware permite generar por software los estímulos del sistema y obtener las respuestas representándolas a través del mismo software, siendo el sistema a verificar implementado completamente en hardware. Esta manera de simulación permite obtener resultados reales cercanos a los obtenidos por medio de la implementación de un prototipo.

Al ser el hardware el encargado del procesamiento de la señal, los tiempos de simulación disminuyen. Para realizar estas funciones, el software de simulaciones debe comunicarse con la plataforma de hardware a través de protocolos de comunicaciones y puertos como pueden ser USB, Ethernet, JTAG, etc.

El caso particular de este TFG, para realizar la verificación y validación del sistema final, se utilizó una herramienta que permite que el proceso interno de co-

simulación por hardware sea totalmente transparente para el usuario, haciéndose cargo de todos los procesos de comunicación entre el software y la plataforma de simulaciones. La herramienta utilizada es la provista por System Generator, que para la placa de desarrollos utilizada en este trabajo permitió la aplicación de co-simulación por hardware a través del puerto USB de la PC.
Para aplicar este proceso, primero se compila el modelo a simular bajo las condiciones de compilación para co-simulación por hardware.
Durante esta compilación, System Generator aplica todos los constraints necesarios para la implementación en hardware y genera los códigos HDL correspondientes al diseño, para luego sintetizarlos y generar así un archivo bitstream, el cual se descarga a la FPGA para implementar el sistema.
Dentro del simulador se obtiene un bloque del modelo a implementar, el cual posee puertos de entrada y salida correspondientes al diseño que se ha desarrollado. A estos puertos de entrada y salida se les puede conectar generadores de señales u otros componentes, tanto de System Generator como de Simulink/Matlab, para generar los vectores de prueba y capturar las salidas producidas en el hardware implementado.
El bloque de co-simulación interactúa tanto con la FPGA como con Simulink durante la simulación, proveyendo a la FPGA los datos a procesar tomados desde el simulador y llevando los datos procesados por el hardware al simulador nuevamente para ser representados y analizados posteriormente.
Los datos son enviados a los puertos de entrada de este bloque de co-simulación y es este el encargado de llevar los datos hasta la FPGA por medio de una interfaz apropiada (JTAG, USB o Ethernet) para que se procesen las muestras según los algoritmos diseñados, y en sentido contrario, es el bloque de co-simulación el encargado de tomar las muestras procesadas y entregárselas nuevamente al simulador para su posterior análisis.
Este tipo de simulación es muy importante ya que posibilita disminuir el tiempo de la simulación por medio del procesamiento de datos en la FPGA y a su vez permite verificar que el modelo diseñado es totalmente ***implementable*** en hardware, permitiendo observar su comportamiento como prototipo para evaluar su funcionamiento y poder validar si el mismo estará apto para ser implementado y cumplir con los objetivos para el que fue diseñado.
La figura 5.4.1 muestra el bloque de co-simulación generado para el diseño realizado.

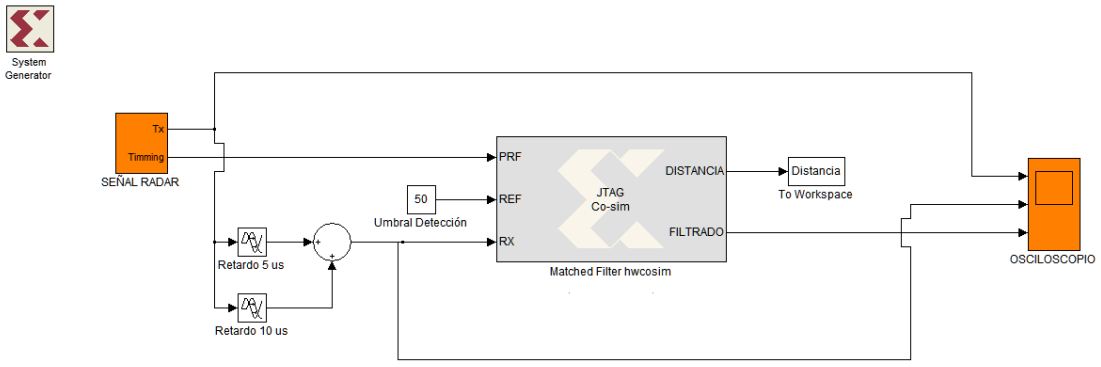

Fig.5.4.1 – Bloque de co-simulación por hardware.

Una vez compilado el diseño se procede a conectar la FPGA via USB con la PC, desde la cual se va a co-simular enviando los estímulos y capturando los resultados obtenidos de la FPGA.
La figura 5.4.2 muestra el resultado de la co-simulación por hardware de 10 µs de duración aplicando una señal de radar descripta en la sección 2.12 a la entrada sin retardo ni ruido aditivo.

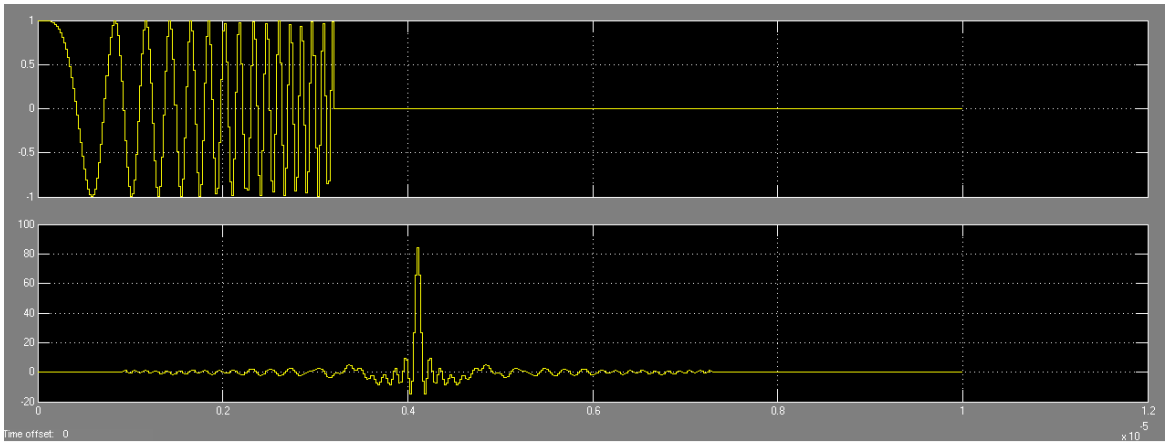

Fig.5.4.2 – Co-simulación por hardware del filtro apareado.

Puede apreciarse que el resultado de la co-simulación por hardware genera una respuesta del filtro semejante, sino idéntica a la simulada en etapas anteriores, comprobando la efectividad de la implementación en la FPGA.
La figura 5.4.3 muestra la co-simulación por hardware para una señal radar sin retardo y con ruido gaussiano de distribución normal, varianza 0.1 y media igual a 0.

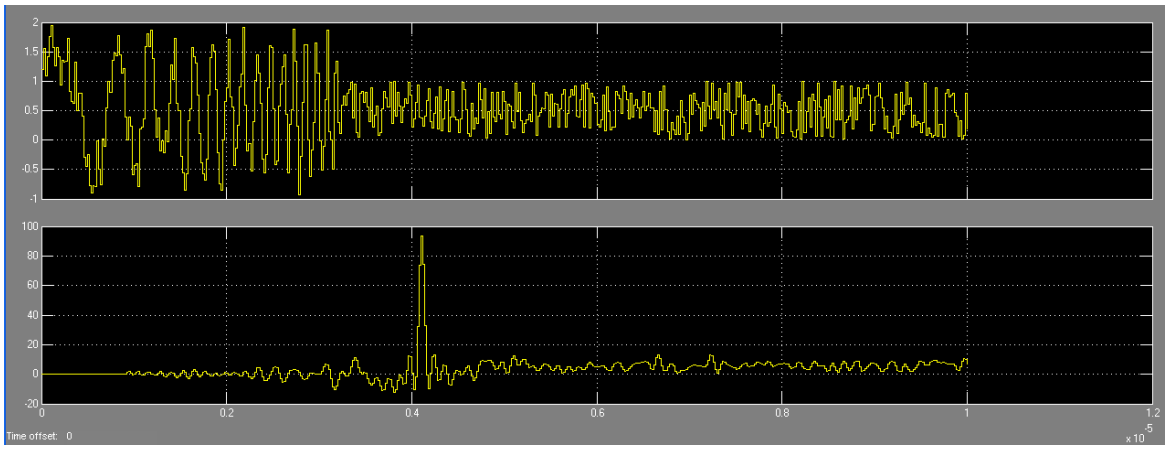

Fig.5.4.3 – Co-simulación por hardware del filtro apareado.

Puede apreciarse en esta imagen que si bien el pulso de radar puede distinguirse, y tiene una SNR = -0.274 dB, a la salida del filtro se maximiza la SNR obteniendo un valor de 19.79 dB.

La siguiente figura muestra una co-simulación aumentando la potencia de ruido con una varianza de 1 y media igual a 0.

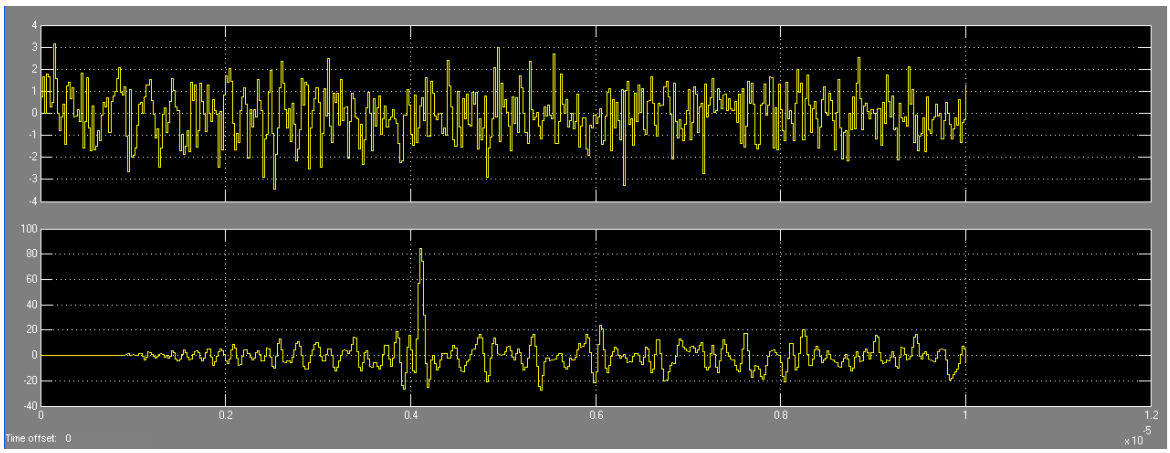

Fig.5.4.4 – Co-simulación por hardware del filtro apareado.

En esta imagen se puede observar que no se distingue el pulso de radar debido al fuerte efecto del ruido gaussiano aditivo. Dicha señal de entrada tiene una SNR de -10.27 dB y la respuesta del filtro apareado genera una señal con SNR de 10.21 dB.

SIMULACIONES CON ECM

A continuación se verifica el funcionamiento del filtro con distintas formas de onda que podría utilizar el adversario como actividad de ECM. Se utilizan frecuencias cercanas a la de funcionamiento del filtro para determinar cómo reacciona frente a estímulos con un cierto grado de correlación con la señal transmitida. Se analizan las amplitudes del pulso producto de la compresión y el filtro apareado, y las amplitudes de componentes indeseadas en la respuesta del filtro que pueden provocar errores al momento de la detección.
Los tipos de Jamming ya fueron presentados. Se aclara nuevamente que se interpreta como Jamming a todo tipo de señal, intencional o no, que interfiere destructivamente a aquella que se necesita para la detección de un objetivo.

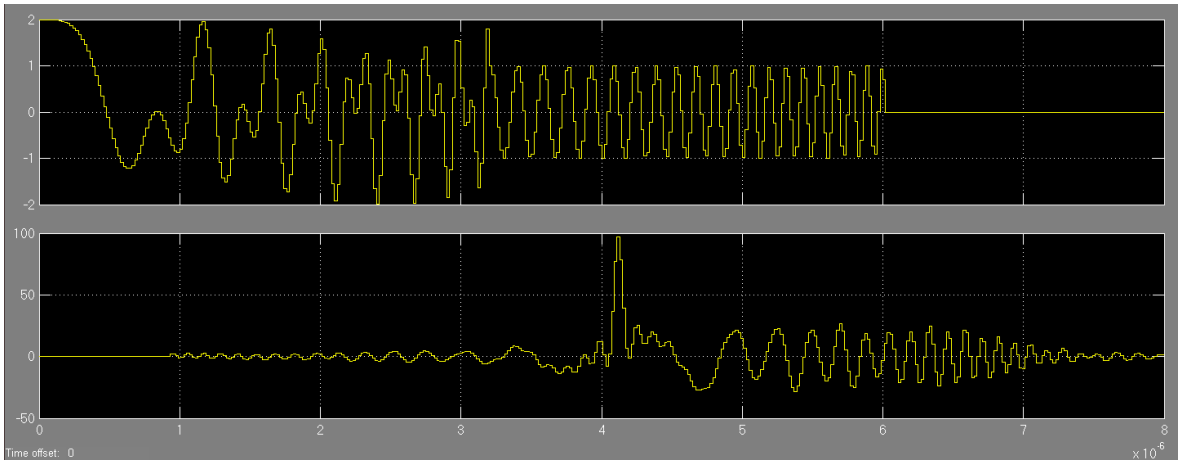

Fig.5.4.5 – Jamming de 9.375 MHZ de ancho de banda y 6 μs de duración.

La SNR de entrada es de 0 dB y a la salida se obtiene una SNR de 10.55 dB.

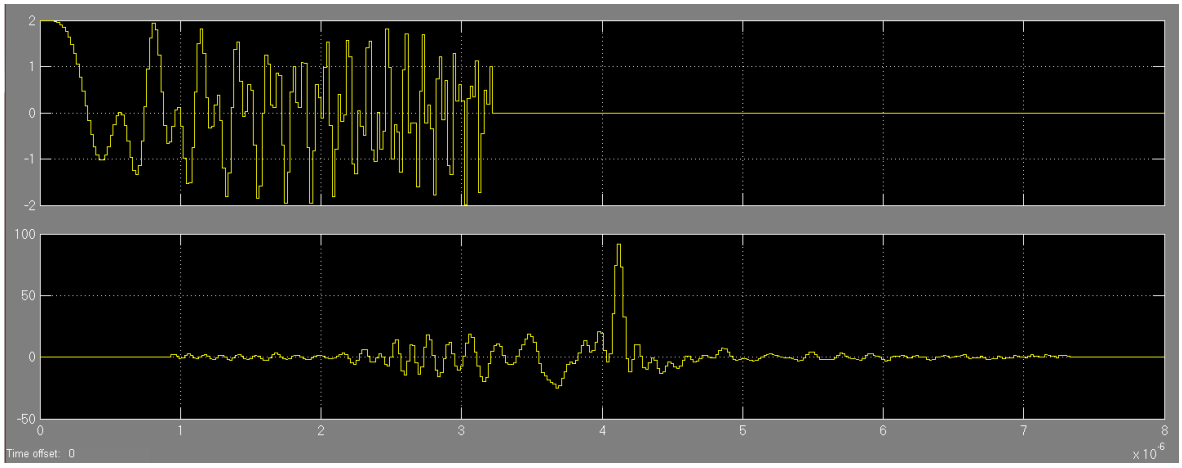

Fig.5.4.6 – Jamming de 20 MHZ de ancho de banda y 3.2 μs de duración.

En esta simulación, la SNR a la salida es de 14.03 dB.

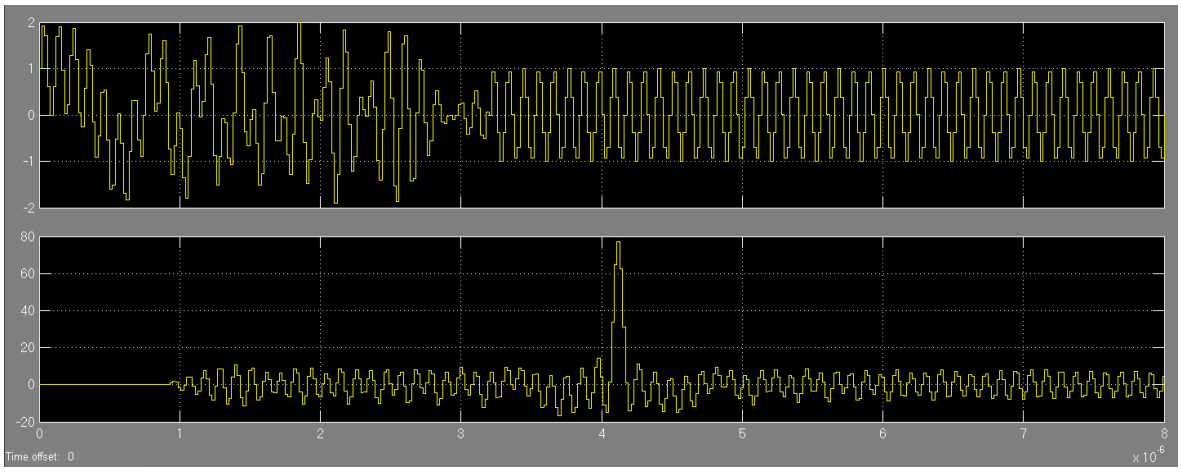

Fig.5.4.7 - Señal senoidal de 9.375 MHZ.

Se obtiene una SNR de 17.37 dB a la salida.
De todas las pruebas realizadas se observa que el pulso comprimido a la salida del filtro apareado mantiene una diferencia favorable en la amplitud respecto a las demás componentes consideradas como ruido, lo que otorga una buena fuente de información, confiable y precisa al detector para que, en base a ella, determine la presencia de un objeto o simplemente indique una falsa alarma.

CAPÍTULO 6: RESULTADOS, CONCLUSIONES Y PROPUESTAS DE FUTUROS DESARROLLOS

6.1 INTRODUCCIÓN

A lo largo de este capítulo se reúnen los resultados finales de las pruebas relevantes realizadas al modelo implementado sobre plataforma digital. Para ello se emplea el método Co-simulación por Hardware explicado en capítulos anteriores.
A partir de dichos resultados se prosigue con las conclusiones de la experiencia lograda durante el transcurso del desarrollo del modelo de filtro apareado. Por último se analizarán aquellos aspectos a desarrollar a futuro, que no fueron incluidos en el presente informe para no perder de vista los objetivos establecidos.

6.2 RESULTADOS

En el capítulo 2.14 se mencionó el filtro adaptivo LMS. Teoría y simulaciones se encuentran en el Anexo A.
Este filtro se desarrolla satisfactoriamente teniendo en cuenta que las pruebas realizadas fueron con ondas continuas y de relativa baja frecuencia. Los valores obtenidos de señal y ruido a la salida pueden llevar a la conclusión de que es un algoritmo completamente útil respecto al filtrado en tiempo real de señales de radar en general. Pero hay que tener en cuenta que en este trabajo se decidió aplicar el filtro a radares pulsados, lo cual implica el uso de señales compuestas por pulsos de corta duración lo que lleva a anchos de banda y potencias elevadas. Estos pulsos de corta duración llevarían al filtro LMS a trabajar a extremada rapidez para obtener el tiempo necesario para la convergencia de coeficientes y disminuir el error lo suficiente como para lograr la relación señal a ruido requerida para la detección correcta de un objetivo. Para implementar un sistema de estas características se requiere hardware de mucha capacidad de procesamiento y memoria debido a los clock´s de alta frecuencia y a la cantidad de etapas necesarias para el procesamiento.
Este fue el motivo principal que llevo a descartar este modelo y continuar el estudio de otros algoritmos que se adapten mejor a las necesidades del proyecto.
No se realizaron pruebas en altas frecuencias (microondas) por falta de hardware necesario, por ello se trabajó sólo a frecuencias de banda base (FI) del orden del MHZ, siendo además en esta etapa (FI) donde realmente se implementa un filtro apareado.
En los procesos de simulaciones presentados, se muestra como el filtro apareado diseñado reacciona frente a cambios de estímulos y/o parámetros en las señales de entrada obteniendo respuestas satisfactorias, con niveles de SNR elevados, respecto a las entradas, lo que permite una detección óptima.
Se realizaron pruebas con técnicas de ventaneo, y si bien arrojo resultados interesantes con la ventana de Hamming, en el desarrollo de este TFG no se implementó debido a que no es del alcance de este proyecto diseñar un receptor

completo, dejando esto como desarrollo a futuro, en el cual se contemplara la optimización en la detección de blancos muy próximos y demás características.
También se simularon señales de ECM que atentan contra la detección optima del filtro, tales como señales jamming, con un alto grado de correlación con la señal transmitida, ya que son a frecuencias similares a los pulsos de radar utilizados en el diseño, obteniendo resultados aceptables.

6.3 CONCLUSIONES

El principal objetivo de este trabajo final de grado fue diseñar un filtro apareado aplicado a radar y que contribuya a ECCM permitiendo maximizar la relación señal a ruido de la señal presente a la entrada del mismo.
En el capítulo 4 se detalló la teoría del filtro apareado, y unas primeras simulaciones en las cuales pudo apreciarse que el diseño modela las ecuaciones matemáticas del filtro apareado.
Un radar pensado para sobrevivir en la guerra electrónica tiene que poseer técnicas de compresión de pulsos, saltos en frecuencia, utilizar pulsos cortos y demás prácticas para disminuir la vulnerabilidad del sistema. En este sentido el filtro diseñado, generó respuestas óptimas y de acuerdo a lo planificado en la etapa de diseño.
En el capítulo 5, se expuso la metodología de diseño de hardware empleada en este proyecto, la cual concluyó con simulaciones funcionales de los componentes y del sistema final, lo que permitió exhibir la factibilidad de implementación de la arquitectura del sistema propuesto. También se demostró el proceso de verificación denominado co-simulación por hardware, el cual permitió obtener resultados aproximados a los que se obtendrán en la verificación del sistema implementado en la FPGA.
Pudo comprobarse que no hubo discrepancias entre el modelo simulado y el diseño final implementado a nivel de compuertas lógicas, obteniendo resultados cercanos a la realidad, lo cual garantiza la factibilidad de implementación del sistema diseñado.
De acuerdo con los resultados obtenidos en los procesos de simulación, esta implementación del filtro apareado cumple con las características de diseño establecidas, y valida su utilización propuesto en este TFG

6.4 DESARROLLOS FUTUROS

No se realizó la implementación del filtro adaptivo – LMS por los motivos expuestos en los resultados. Se puede analizar con mayor profundidad este modelo para determinar definitivamente la imposibilidad de aplicación a sistemas de radar pulsados. Se propone a futuro verificar la utilización de este algoritmo en radares de onda continua, con el cual podrían obtenerse buenos resultados.
Si bien se utilizó un tipo de señal radar, definida en la sección 2.12, puede utilizarse un banco de señales de radar, desarrollando un filtro apareado con un set de

coeficientes, correspondientes a estas señales, logrando de esta manera aumentar la no vulnerabilidad del sistema radar.

Se propone además a futuro, diseñar una etapa de recepción del sistema utilizando una modulación BPSK *y* QPSK con códigos convolucionales, disminuyendo más aún el grado de vulnerabilidad del receptor.

Aunque la plataforma Virtex II no mostró problemas en el uso de su capacidad de hardware y en los tiempos de procesamiento necesarios para el matched filtering diseñado, se recomienda implementar el diseño en una plataforma digital más moderna, optimizando los recursos consumidos de la FPGA y disminuyendo los tiempos de procesamiento, al tiempo que permita aumentar la frecuencia de trabajo y la utilización de conversores AD y DA de alta performance.

ANEXO A: FILTRO ADAPTIVO LMS

El objetivo de este algoritmo es hallar los coeficientes necesarios para lograr que a la salida del filtro se presente una señal lo más parecida posible a una señal de referencia o conocida. El proceso se basa en una señal error y en base a la información provista por ella se realizan los procedimientos necesarios para modificar los coeficientes y disminuir la diferencia entre ambas señales. En otras palabras, si tenemos una muestra de la señal que fue transmitida, y obtenemos del medio el eco proveniente del objetivo con el ruido aditivo, al aproximar la señal dañada a la deseada, indirectamente se busca optimizar el valor de la relación señal a ruido presente en ella.
El filtro actúa en base al gradiente de la señal error, moviéndose en sentido contrario al de mayor crecimiento, minimizando esta función.
La salida de un filtro FIR se caracteriza por sus coeficientes bn siendo su salida igual a,

$$y(n) = \sum_{k=0}^{P} b(k)x(n-k) \qquad \text{Ec.A1}$$

Donde b(k) son los coeficientes.
La señal error se entiende como la diferencia entre la señal deseada y la obtenida a la salida del filtro,

$$e(n) = d(n) - y(n) \qquad \text{Ec.A2}$$

Se define una función J que representa el error, En base a ella se determinarán los nuevos coeficientes del filtro y minimizar dicho error.

$$J = \frac{1}{2}\left[e(n)\right]^2 = \frac{1}{2}\left[d(n) - y(n)\right]^2 = \frac{1}{2}\left[d(n) - \sum_{k=0}^{P} b(k)x(n-k) \right]^2 \qquad \text{Ec.A3}$$

Se eleva al cuadrado para obtener valores siempre positivos y se divide por dos para limitar al valor medio los resultados. Esto significa que maximizar la relación señal a ruido a la salida del filtro implica aproximar lo más posible a cero el valor de la función J.
Se calcula el vector gradiente de J,

$$\nabla J = \frac{\partial J}{\partial b(k)} = \left[d(n) - \sum_{k=0}^{P} b(k)x(n-k) \right] - x(n-k) = -e(n)x(n-k) \qquad \text{Ec.A4}$$

Por lo tanto el próximo conjunto de coeficientes viene dado por,

$$b'(n) = b(n) - \mu\nabla J = b(n) + e(n)x(n-k) \qquad \text{Ec.A5}$$

El cambio de signo es para ir en sentido contrario al de mayor crecimiento.

μ es una constante que indica la rapidez con que se alcanza el mínimo valor. Un valor alto de μ significa un filtro con gran capacidad de adaptación.
Expresando vectorialmente los nuevos coeficientes,

$$\begin{bmatrix} b'(1) \\ b'(2) \\ \dots \\ b'(p) \end{bmatrix} = \begin{bmatrix} b(1) \\ b(2) \\ \dots \\ b(p) \end{bmatrix} + \mu.e(n) \begin{bmatrix} x(n-1) \\ x(n-2) \\ \dots \\ x(n-p) \end{bmatrix} \qquad \text{Ec.A6}$$

A continuación se presentará el desarrollo de la simulación en MatLab de este algoritmo a partir de un script, formando parte de las pruebas y resultados previos a la implementación física del modelo final del filtro.

PRUEBA DEL ALGORITMO LMS

Se creó la función coef_adap en un script de MatLab® basado en la teoría que rige al algoritmo LMS.
La estructura es la siguiente,

[h, y, e] = coef_adap(x, d, h0, mu)

Dónde:

x: es la señal eco recibida.

d: es la señal transmitida.

h0: son los coeficientes iniciales.

mu: es el tamaño del paso (aprendizaje).

h: son los nuevos coeficientes.

y: es la señal de salida.

e: es la señal de error.

El código de esta función es el siguiente:

```
function [h,y,e] = coef_adap(x,d,h0,mu)
h=h0;
P=length(h);
N=length(x);
y=zeros(1,N);
e=y;
rP=0:-1:-P+1;
for k=P:N,
xx=x(k+rP);
y(k)=xx*h';
e(k)=d(k)-y(k);
h=h+mu*e(k)*xx;
end
```

Coef_adap intenta acercar la señal x a d, partiendo de los coeficientes iniciales h0, y basándose en la señal error para el cálculo de los nuevos coeficientes. La constante mu es la velocidad con la que el proceso acerca x a d.
En el caso de que la señal de entrada no tenga ningún tipo de correlación con la señal deseada el sistema lineal utilizado (FIR) no podrá modelar la señal de salida y el algoritmo no funcionará.
Se realizó un script que utiliza a su vez las funciones “firpmord” y “firpm” provistas por MatLab®. La primera función calcula el orden, frecuencia normalizada, amplitud de la banda de paso y el peso de un determinado filtro FIR. La segunda utiliza los parámetros de salida de la función anterior para calcular los coeficientes determinantes de este filtro, estos serían la condición inicial del proceso de adaptación (h0).
A continuación se muestra un modelo sencillo que representa una onda trasmitida por el radar. Se realizó a partir de una senoide más ruido generado por la función “randn”, modulada por un tren de pulsos cuadrados.
Por último se llamó a la función recién creada “coef_adap” que nos entrega las señales resultantes mostradas en las siguientes figuras.
El script mencionado a partir del cual se obtuvieron los resultados que se muestran a continuación es,

```
Fs = 48000;                % Frecuencia de Muestreo

Fpass = 9600;              % Frecuencia de paso
Fstop = 12000;             % Frecuancia de Rechazo
Dpass = 0.057501127785;    % Ripple en banda de paso
Dstop = 0.001;             % Atenuacion en banda de Paso
dens  = 20;                % Factor de densidad

[N, Fo, Ao, W] = firpmord([Fpass, Fstop]/(Fs/2), [1 0], [Dpass,
Dstop]);

b  = firpm(N, Fo, Ao, W, {dens});

t=0:0.001:2;
p=pi/2;
A=1;
x=0.5+0.5*square(4*pi*t);
x1=(A*sin(5*2*pi*t+p)).*x;
x2=(x1+1*randn(1, 2001)).*x;
MU=0.009;
H, Y, E]=chef_adap(x2, x1, b, MU);
Y2=Y+E;
```

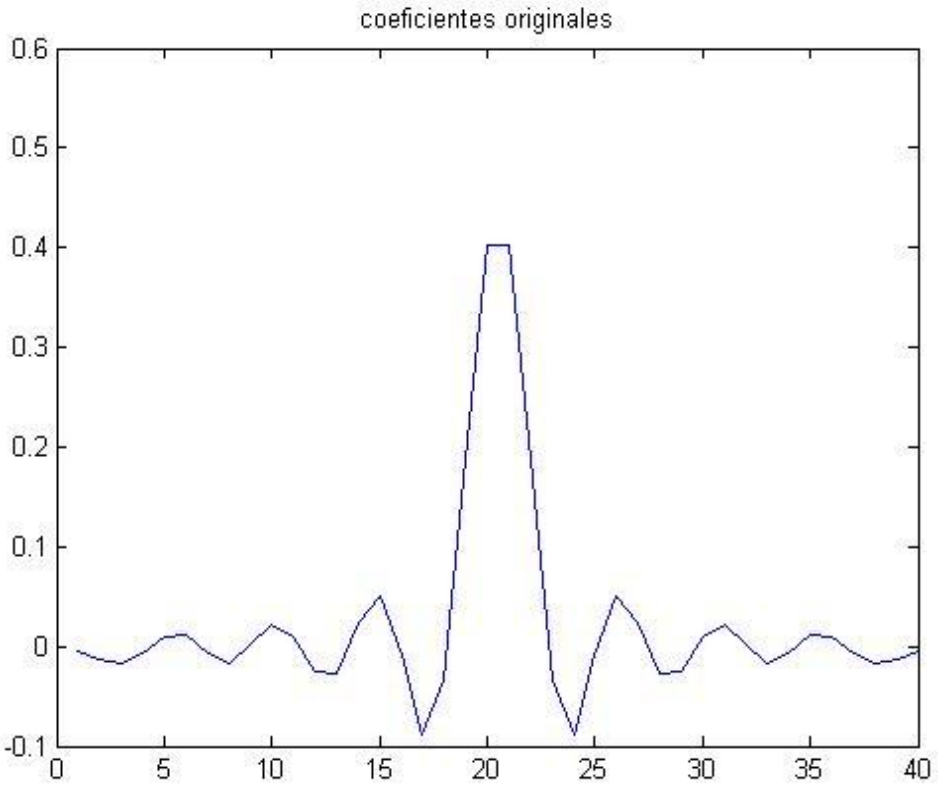

Fig.A1 – Coeficientes originales.

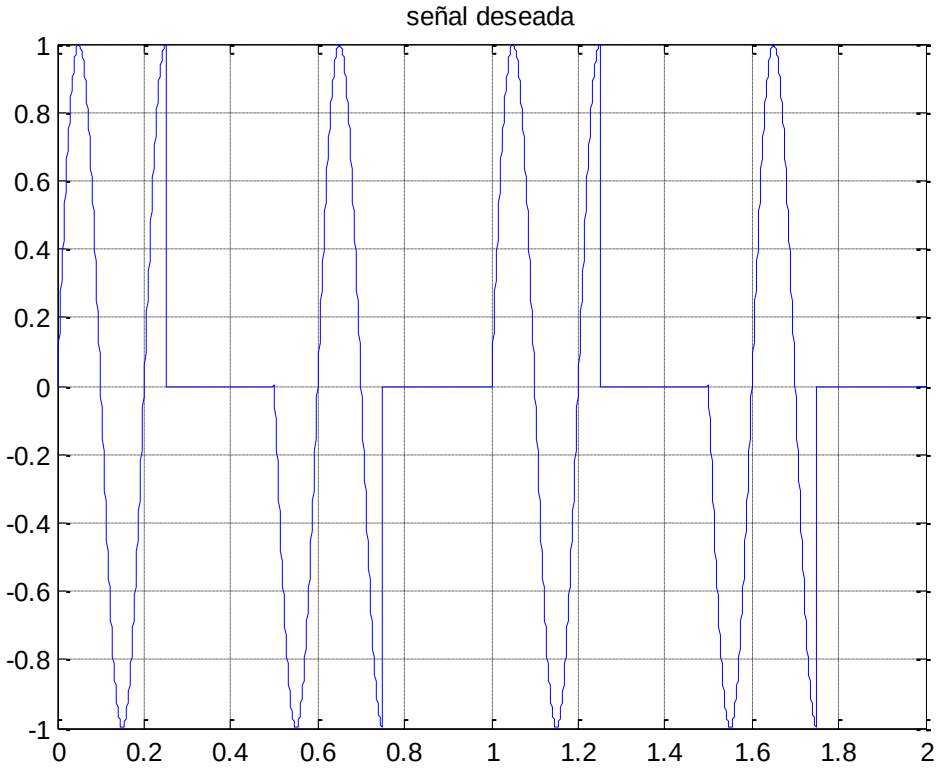

Fig.A2 – Señal deseada.

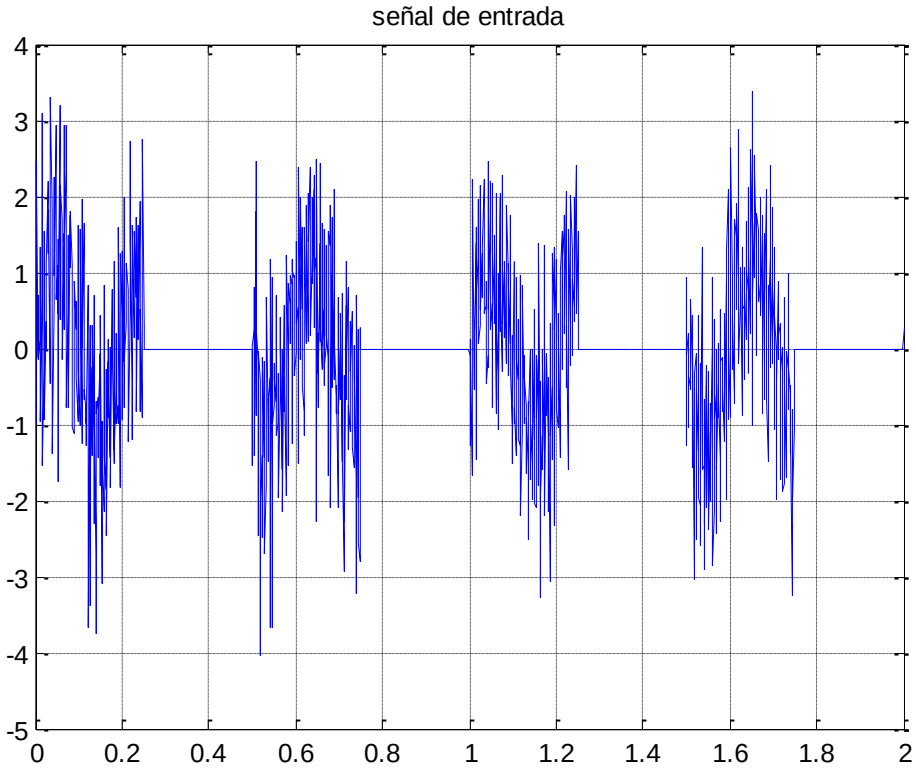

Fig.A3 – Señal deseada más ruido.

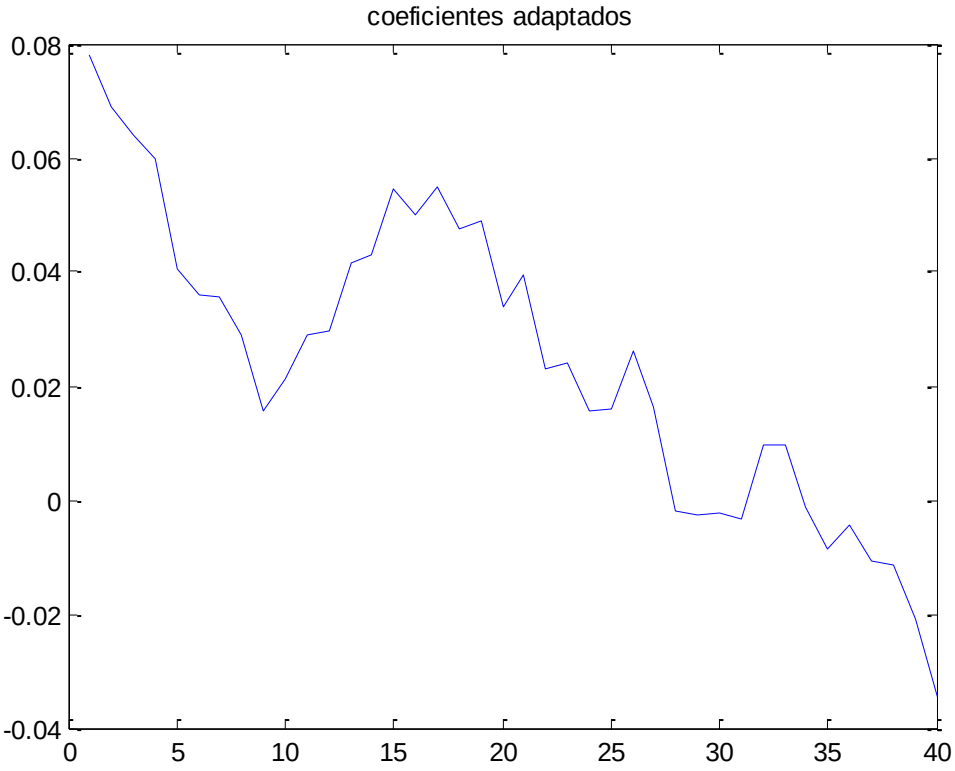

Fig.A4 – Nuevo set de coeficientes.

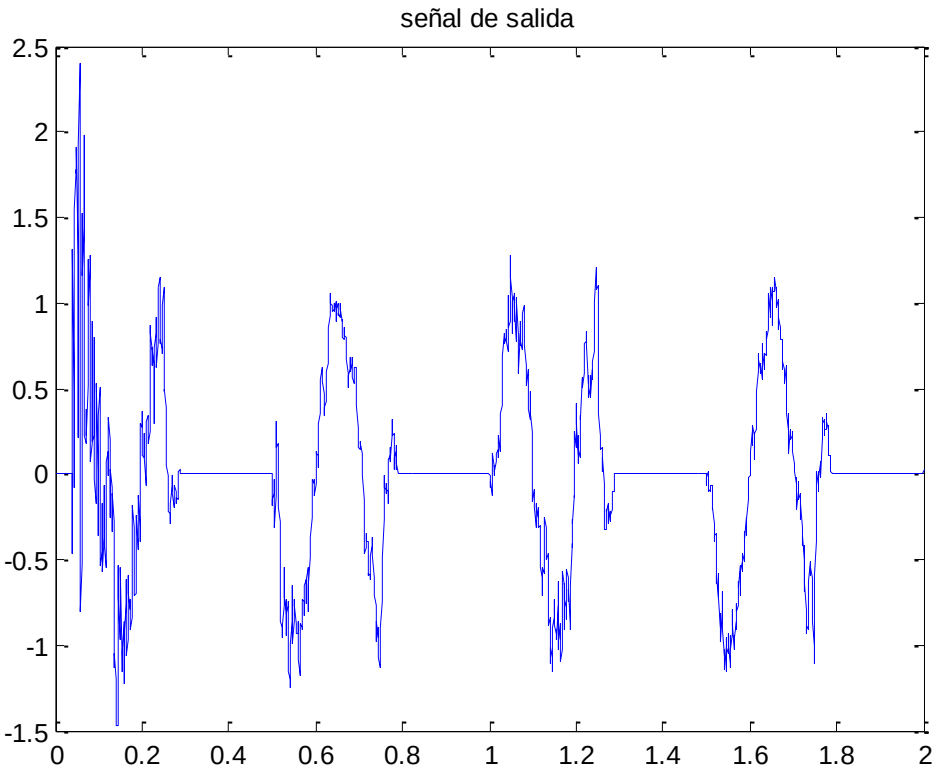

Fig.A5 – Señal obtenida del filtrado.

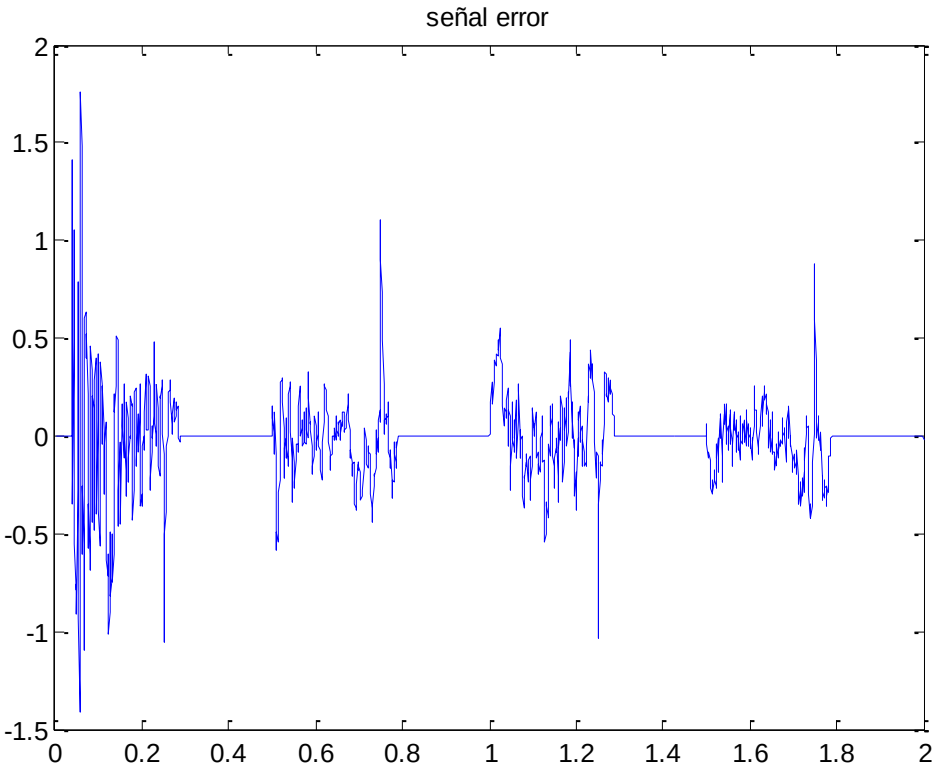

Fig.A6 – Señal Error.

FILTRO ADAPTIVO – LMS EN SIMULINK

Se utilizó las herramientas y bloques provistas por Simulink para el diseño y simulación del filtro adaptivo – LMS ya simulado matemáticamente por el workspace de MatLab.
El diagrama empleado para realizar estas simulaciones se muestra en la Fig.3.13.1 donde se ve una fuente senoidal digitalizada y una de ruido blanco, gatway`s de entrada y salida que adaptan las señales al formato necesario para el procesamiento, y los scope´s encargados de mostrarnos los resultados. En la Fig.3.13.2 se muestra la estructura interna del bloque “Filter LMS”.

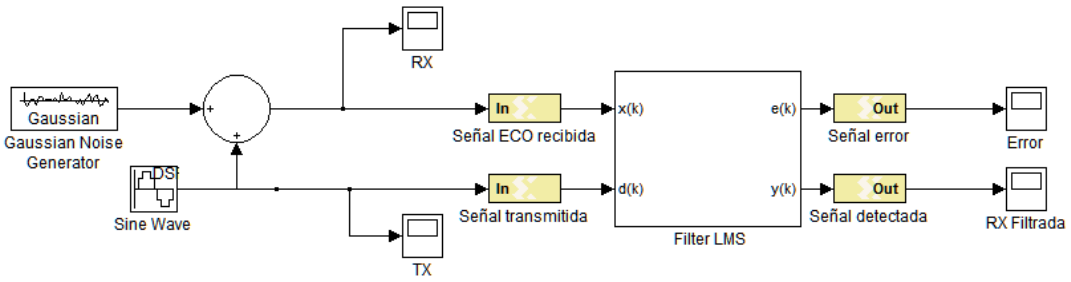

Fig.A7 – Esquema del filtro adaptativo LMS en System Generator.

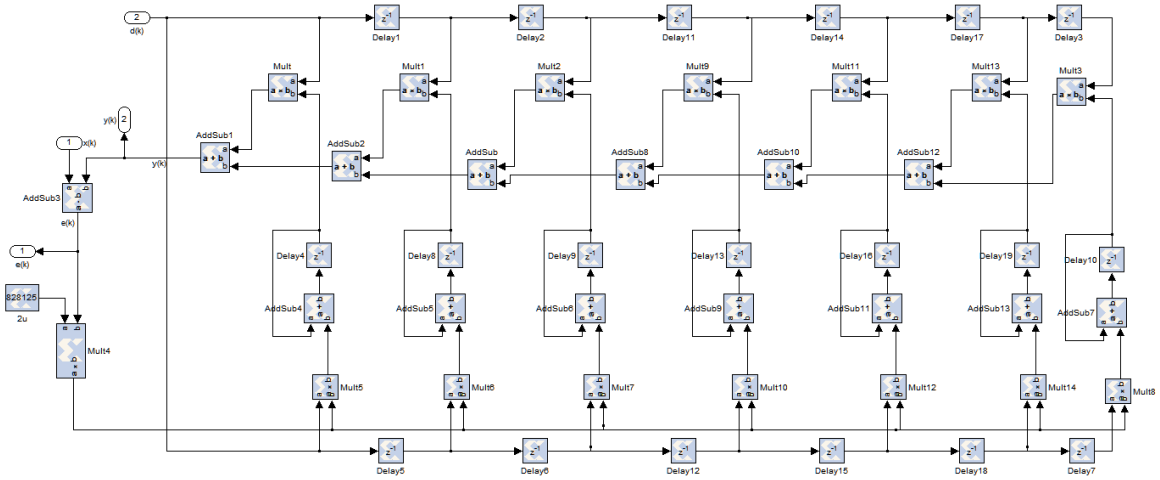

Fig.A8 – Arquitectura interna del bloque “Filter LMS”.

Se mostrará a continuación las señales de entrada y aquellas obtenidas a las salidas del filtro.

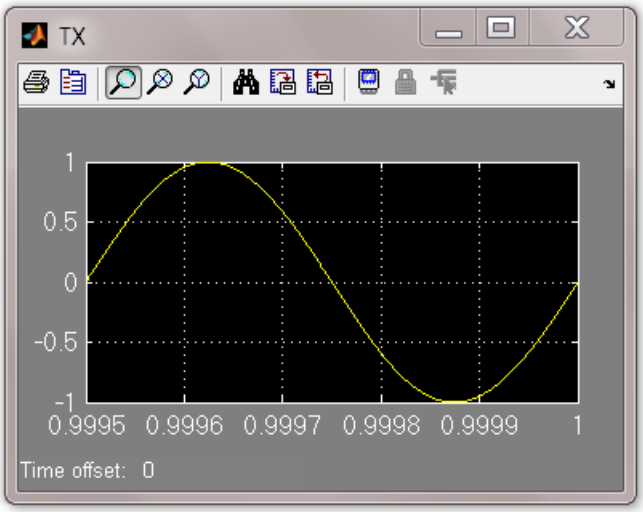

Fig.A9 – Señal de entrada

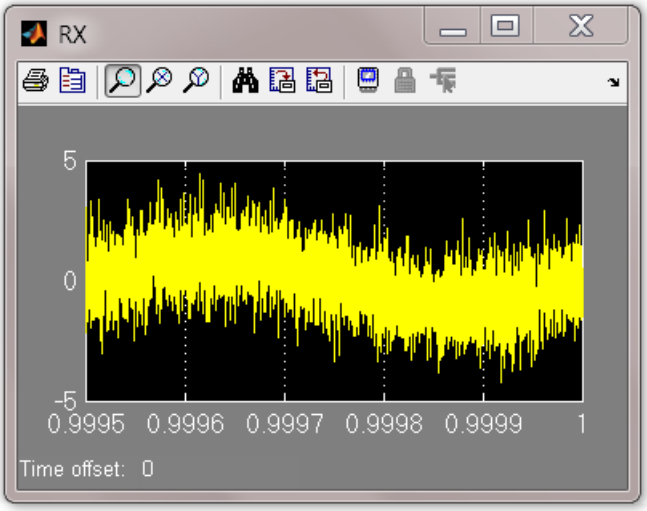

Fig.A10 – Entrada mas ruido

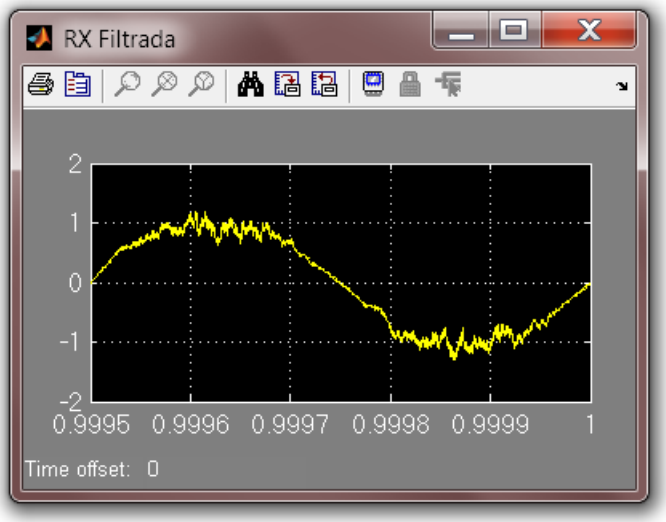

Fig.A11 – Señal filtrada

REFERENCIAS

[Ref 1] – Introduction to Communication Electronic Warfare Systems – Richard Poisel.
[Ref 2] – Introduction to RADAR Systems – Merrill I. Skolnik.
[Ref 3] – RADAR system Analysis and Design Using MatLab – Bassem R. Mahafza.
[Ref 4] – Teoría de Señales – Victor Hugo Sauchelli.
[Ref 5] – ”On the use of window for harmonic analysis with the DFT”. *IEEE Processing* - F. J. Harris
[Ref 6] - RADAR HANDBOOK - Merrill I. Skolnik - Editor in Chief - Third Edition
[Ref 7] - S. A. White “Applications of Distributed Arithmetic to Digital Signal Processing: A tutorial Review”, IEEE ASSP Magazine Vol 6 No 3, 1989.
[Ref 8] - Xilinx® DSP Application Notes, “The Role of Distributed Arithmetic in FPGA-based Signal Processing”,1996
[Ref 9] - Brad Brannon. Understanding state of the art in adcs. Article in www.rfdesign.com, May 2008.
[Ref 10] - Xilinx. Dsp: Designing for optimal results - high performance dsp using virtex-4 fpgas. Technical report, Edition 1.0, 2005.
[Ref 11] - Xilinx. Virtex II Pro and Virtex II Pro X - FPGA User Guide. Xilinx, ug012 edition, November 2007.
[Ref 12] - Xilinx. FIR Compiler V4.0 Datasheet, ds534 edition, June 2008.
[Ref 13] - Xilinx. System Generator for DSP - User Guide. Xilinx, 10.1.3 edition, September 2008.
[Ref 14] - John G. Proakis. Digital Signal Processing. Prentice Hall, 2007.
[Ref 15] - Xilinx Inc. ISE 10.1 Quick Start Tutorial. 2008.
[Ref 16] - Steven W. Smith, 2003. The Scientist and Engineer's Guide to Digital Signal Processing.
[Ref 17] - B. A. Shenoi, 2006. Introduction to digital signal processing and digital design. Editorial Wiley.
[Ref 18] - D.B. Van Veen and K.M. Buckley. “Beamforming: A Versatile Approach to Spatial Filtering”. IEEE. ASSP Magazine. April, 1998.
[Ref 19] - PROPOSICIÓN Y SIMULACIÓN DE UN ALGORITMO ADAPTATIVO PARA SISTEMAS DE ANTENAS INTELIGENTES - Perla Espinosa Díaz y Carlos Villarroel González.