



UNIVERSIDAD
DE PIURA

REPOSITORIO INSTITUCIONAL
PIRHUA

APLICACIÓN DE LA TRANSFORMADA WAVELET A SEÑAL DE BAJA POTENCIA EN ENTORNO DE MATLAB

Edward Yamunaqué-Chunga

Piura, junio de 2016

FACULTAD DE INGENIERÍA

Máster en Ingeniería Mecánico-Eléctrica con Mención en Automática y
Optimización

Yamunaqué, E. (2016). *Aplicación de la Transformada Wavelet a señal de baja Potencia en Entorno de Matlab* (Tesis de Máster en Ingeniería Mecánico-Eléctrica con mención en Automática y Optimización). Universidad de Piura. Facultad de Ingeniería. Piura, Perú.



Esta obra está bajo una licencia

[Creative Commons Atribución-NoComercial-SinDerivar 4.0 Internacional](https://creativecommons.org/licenses/by-nc-nd/4.0/)

[Repositorio institucional PIRHUA – Universidad de Piura](https://repositorio.institucional.pirhua.edu.pe/)

U N I V E R S I D A D D E P I U R A

FACULTAD DE INGENIERÍA



“Aplicación de la Transformada *Wavelet* a señal de baja Potencia en Entorno de Matlab”

Tesis para optar el Grado de Máster en
Ingeniería Mecánico – Eléctrica con mención en Automática y Optimización

Edward José Yamunaqué Chunga

Asesor: Mgtr. Italo Chinchay Ulloa

Piura, Junio 2016

A Dios.

A mi madre Rosa Chunga, a mi familia por su comprensión, y al Consejo Nacional de Ciencia, Tecnología e Innovación Tecnológica (CONCYTEC) por la oportunidad de participar en el programa de maestría IME.

Prólogo

La transformada *Wavelet* es una muy buena herramienta tanto en una como en dos dimensiones, sin embargo esta transformada contrariamente de su importancia y efectividad en diversos procesos de estudio de señales, no se encuentra muy divulgada en el ámbito académico fuera de los estudios de maestría y doctorado o de matemáticas puras a nivel nacional.

La transformada *Wavelet* es una función matemática que permite obtener datos de una señal seccionándolos en pequeños componentes en escala frecuencia-tiempo y poder analizarlos en forma separada. En comparación con la transformada de Fourier, la *Wavelet* permite trabajar sobre ondas no estacionarias, con discontinuidades o picos, y respecto de otras herramientas que permiten el análisis de este tipo de ondas se distingue por su capacidad de Multirresolución. De esta forma se muestran las propiedades de la transformación *Wavelet* con una señal de baja potencia adquirida por un sistema de adquisición de datos también realizado en este trabajo, con el fin de deducir la conveniencia de adoptarla para algunas aplicaciones posteriores descritas brevemente.

El tema del estudio nace como iniciativa del estudio de la transformada *Wavelet*. Para lograr este tipo de estudio, se realiza también un estudio de software y hardware para lograr un sistema de adquisición de señal de muy buenas características comparadas a las tradicionales en el mercado basadas en tecnología FPGA, respetando que parte del software está dentro del movimiento *open-source* en que se construye las tramas de comunicación. Como parte del proyecto se utilizaron los software: Python, Vhdl, Verilog y Matlab. Con los programas mencionados se ha podido mejorar la señal de captura para utilizarla en otros estudios, por ejemplo: el estudio de reconocimiento de patrones en las emanaciones de radiación de un microprocesador.

Como corresponde también, mi agradecimiento a mis profesores y compañeros de investigación por la motivación brindada durante el desarrollo de la presente tesis. De manera especial agradezco al Mg. Ing. Italo Chinchay Ulloa, por sus valiosas enseñanzas, certero asesoramiento académico y confianza en la materia de investigación. Asimismo a al Ing. William Ipanaqué Alama por su atención que me ofrecieron ante cualquier duda o problema que se presentaba durante el desarrollo del proyecto. Finalmente a CONCYTEC y la Universidad de Piura, por la oportunidad brindada para la formación académica y personal.

Finalmente a mis amigos José Carlos Oviden y Pablo Mogollon por su agradable amistad y colaboración académica.

Resumen

En los proyectos de estudio de señales unidimensionales muchas veces se realiza un estudio global en el tiempo, pero sin embargo esta misma señal se puede separar y observar el comportamiento local de la señal, permitiéndonos discriminar ciertas características no deseadas como por ejemplo el ruido, también mediante el algoritmo Multirresolución se puede comprimir los datos de la señal.

Este trabajo constituye principalmente una introducción a la teoría de *Wavelet*. Para poder abordar este tema se estudia de manera breve la transformada de Fourier y Gabor para luego dar una explicación general de los conceptos sobre Transformada de *Wavelet* en su forma continua y discreta. Posteriormente se introduce *Wavelets* como una herramienta alternativa para filtrado y compresión de señales unidimensionales.

Para utilizarla eficientemente Uno de los métodos de análisis multi-resolución es la transformada *Wavelet* discreta (DWT), la cual proporciona una discriminación simultánea tiempo-frecuencia de una señal y además permite identificar los coeficientes más significativos de su representación. Para lograr esto, se deben integrar varios equipos, el más importante de ellos es el sistema de adquisición de datos. Este sistema embebido contiene un protocolo de comunicación independiente y es justamente el motivo de investigación de la presente tesis.

Índice

Prólogo	vii
Índice	xi
Capítulo 1	15
1 Introducción.....	15
1.1 Objetivos.....	15
1.1.1 Objetivo General	15
1.1.2 Objetivo Especifico	15
1.2 Estructura de la tesis	16
1.3 Estado del arte.....	17
Capítulo 2	19
2 Información preliminar y estudios previos.....	19
2.1 Señales No estacionarias.....	19
2.2 El consumo de potencia en dispositivos CMOS.....	19
2.3 Ataques por canal lateral.....	20
2.4 Radiación electromagnética en dispositivos CMOS.....	21
Capítulo 3	25
3 Estudio de las Herramientas de Hardware comerciales utilizadas para la medición...	25
3.1 Sonda de Campo Magnético	26
3.2 Amplificador de bajo ruido.....	26
3.3 <i>OpenADC</i>	27
3.4 FPGA	28
3.4.1 <i>Spartan – 6-LX9</i>	28
3.4.1.1 Especificaciones	29
3.4.2 El arreglo de compuertas programable.....	30
3.4.3 Lenguaje de Diseño de Hardware(VHDL/Verilog)	30
3.4.4 Descripción de la herramienta utilizada	31
3.4.5 Ventajas de una arquitectura FPGA	31
Capítulo 4	33

4	FPGA, Python, comunicación serial y almacenamiento de datos	33
4.1	Proceso para el diseño de FPGA	34
4.1.1	Creación de archivos de diseño en VHDL	35
4.1.2	Simulación del diseño.....	35
4.1.3	Síntesis del diseño	36
	4.1.4 Mapeo	37
4.1.5	Diseño, mapeo y Ruteo.....	37
4.1.6	Asignación de pines.....	37
4.1.7	Programación.....	39
4.2	Almacenamiento de la programación en memoria	40
4.3	Python 2.7.3	40
4.3.1	Características.	41
Capítulo 5	43
5	Representaciones Tiempo-Frecuencia.....	43
5.1	Transformada de Fourier	43
5.1.1	Definición	43
5.1.2	Observaciones	47
5.1.3	Desventajas.....	47
5.2	Transformada Discreta de Fourier (DFT).....	48
5.2.1	Definición	48
5.3	Transformada de Fourier de tiempo corto (Gabor-STFT).....	50
5.3.1	Definición	51
Capítulo 6	53
6	Transformada <i>Wavelet</i> basado en señales unidimensionales	53
6.1	Transformada <i>Wavelet</i> Continua (CWT).....	54
6.1.1	Introducción.....	54
6.1.2	Familias <i>Wavelet</i> o <i>Wavelet</i> madre	55
6.2	Transformada <i>Wavelet</i> Discreta (DWT).....	57
6.2.1	Introducción.....	57
6.3	Análisis Multirresolución (MRA).....	59
6.4	Eliminación de ruido usando <i>Wavelet</i>	61
6.4.1	Procedimiento general de filtrado mediante <i>Wavelet</i>	62
6.4.2	Técnicas de cancelación de ruido.....	63
6.4.2.1	Selección de madre <i>Wavelet</i>	63
6.4.2.2	Determinación del nivel de descomposición.....	63
6.4.3	Segmentación o Umbral	64

	xi
6.4.4 Selección del umbral	65
6.4.5 Criterios de evaluación:	66
6.4.5.1 Proporción Señal-ruido:	66
6.4.5.2 Error cuadrático medio(MSE)	66
6.5 Compresión de señales unidimensional	66
6.5.1 Compresión mediante transformada <i>Wavelet</i>	67
Capítulo 7	69
7 Experimento	69
7.1 Filtrado y Compresión	71
Conclusiones	73
Referencias Bibliográficas.....	75
Anexos.....	79

Capítulo 1

Introducción

Mediante el estudio y análisis de la teoría de *Wavelet* enfocado al tratamiento de señales unidimensionales se busca hacer una comparación con la teoría de Fourier y de Gabor y luego establecer así diferencias entre una y otra. Con el fin de comprobar sus ventajas se analiza una señal real, adquirida por nuestro propio desarrollo de adquisición, ésta aplicación tiene el fin de ver las ventajas de la transformada *Wavelet* que permita ser utilizado con mayor frecuencia en el Laboratorio de Sistemas Automáticos de Control de la Universidad De Piura dentro de las telecomunicaciones como en los sistemas de control o en otras áreas similares donde se analice una señal en el tiempo y en diferentes escalas. Además se plantea tener un módulo formado por: sonda, *OpenADC*, FPGA y PC para obtener el resultado experimental, y el programa MATLAB para analizar la señal adquirida.

En los proyectos de estudio de señales unidimensionales muchas veces se realiza un estudio global en el tiempo, pero sin embargo esta misma señal se puede separar y observar el comportamiento local de la señal, permitiéndonos observar con más detalles la señal como por ejemplo los cambios bruscos de amplitud de dicha señal, además se pueden realizar otras aplicaciones como de filtrado y compresión de señal.

La orientación que va a tener la presente tesis es el desarrollo del estudio de la transformada *Wavelet* y en especial el análisis Multirresolución. Aquí nace el tema de tesis que tiene por objetivo integrar los conocimientos matemáticos, físicos, programación y de herramientas electrónicas, haciendo posible la realización del estudio de la transformada *Wavelet* de una señal real.

1.1 Objetivos

1.1.1 Objetivo General

- Aplicación de la Transformada *Wavelet* a señal de baja Potencia en Entorno de Matlab.

1.1.2 Objetivo Especifico

- Realizar un sistema de adquisición de datos para lograr la toma de datos de señales de baja potencia y de rango de frecuencias amplias desde bajas hasta altas frecuencias.
- Estudiar los conceptos matemáticos de la transformada *Wavelet*.

- Estudiar y analizar en método de análisis Multirresolución.
- Analizar la metodología de desarrollo de la Transformada *Wavelet* Discreta.
- Utilizar la transformada *Wavelet* para eliminar el ruido blanco o Gaussiano que se genera en la señal durante el proceso de captura.
- Utilizar la transformada *Wavelet* para la reducción de tamaño de la señal unidimensional.
- Aprender cómo nos puede ayudar la transformada *Wavelet* para el estudio de señales no estacionarias.
- Conocer los conceptos fundamentales de la transformada *Wavelet* Continua y Discreta para poder aplicarla.

1.2 Estructura de la tesis

La tesis que aquí se propone, se ha descrito dividiendo el documento en siete capítulos y tres anexos

La siguiente lista detalla la información contenida en cada capítulo y anexo que componen este documento:

Capítulo 2. Hace una pequeña introducción al criptoanálisis y los ataques por canal lateral, para centrarse en la variante por análisis electromagnético que es la base de este proyecto. Se revelan sus fundamentos y las diferentes técnicas aparecidas en la literatura.

Capítulo 3. Se explica la metodología elegida para la toma de datos del dispositivo embebido bajo análisis mediante los ataques por canal electromagnético, argumentando cada una de las decisiones realizadas y se describe cada uno de los dispositivos usados.

Capítulo 4. Se hace una descripción detallada de la configuración detallada del hardware y software desarrollado para llevar a un ataque por canal lateral electromagnético.

Capítulo 5. Presenta un estudio de las representaciones tiempo frecuencia, ventajas y desventajas. Para ello se hace uso de señales de prueba.

Capítulo 6. Se analiza la señal con un análisis Multirresolución tanto para el proceso de filtrado y reducción de la señal.

Capítulo 7. Describe otros resultados experimentales efectuados para el desarrollo de este trabajo.

1.3 Estado del arte

Se indica a continuación algunas de las más recientes publicaciones donde se emplea a un ATmega como chip donde está implantado un algoritmo de encriptación:

[1] *Semi – Supervised Template Attack*. Los autores de la universidad Pública de *Bruxelles* – Bélgica, en el año 2013 realizaron una propuesta y estudio de viabilidad de un sistema para el estudio del consumo de un ATmega328P mediante una resistencia de 47 Ohm .Esta señal es capturada por un osciloscopio *Agilent Infiniium 80000B* en la cual cuenta con una frecuencia de muestreo de 2Ghz - 40Ghz , con esta señal capturada realizan una serie de estudios off-line como “*Pesos Hamming, Template attack, Análisis estadístico*” demostrando que se puede aprovechar la fuga de información en un dispositivo embebido criptográfico, controlando el dispositivo cambiando de clave o de información al sistema embebido. En la figura 1, se muestra el esquema realizado.

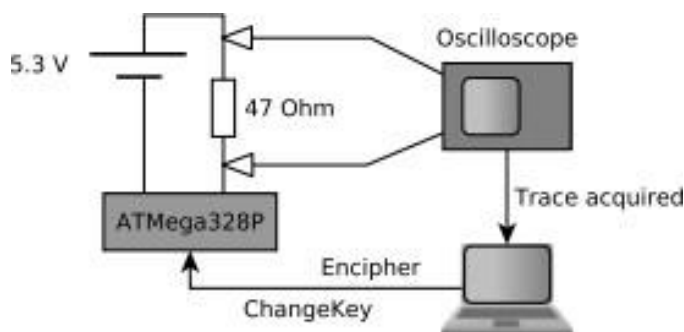


Figura 1 - Esquema de adquisición de Datos.

Fuente: [1]

[2] *An Experimental CPA attack for Arduino Cryptographic Module and Analysis in Software-based CPA Countermeasures*. Los autores de la “ Universidad de *Dongseo*”- Corea, en el año 2014 se plantea que puede extraer información valiosa midiendo el consumo del microcontrolador, se implantó un algoritmo de encriptación y se llega a la conclusión que este dispositivo es vulnerable a los ataques de SCA. Se usó un algoritmo de análisis de correlación de potencia. Mediante uso del osciloscopio se usó para obtener el resultado experimental, y el programa MATLAB para el proceso de verificación y para el analizar los datos.

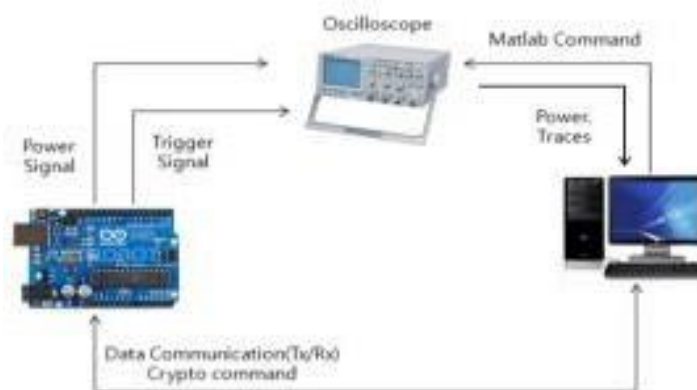


Figura 2 - Configuración de la adquisición

Fuente: [2]

[3] *Power-Based Side-Channel Attack for AES Key Extraction on the ATmega328 Microcontroller*. Los autores de la universidad de Cambridge – USA, en el 5 de diciembre del 2015 realizaron una propuesta y estudio de viabilidad de un sistema para el estudio del consumo de un ATmega328P mediante una resistencia de 100 Ohm, en el microcontrolador se implementó un algoritmo de encriptación AES (*Advanced Encryption Standard*)-128 bits. Esta señal es capturada por un osciloscopio Tektronix 5054B en la cual cuenta con una frecuencia de muestreo de 4Ghz, al inicio se consideró la toma de datos a esa frecuencia pero para acelerar el proceso se redujo a 2500Mhz llegando a la conclusión que reduciendo la tasa de muestreo se logró resultados similares.

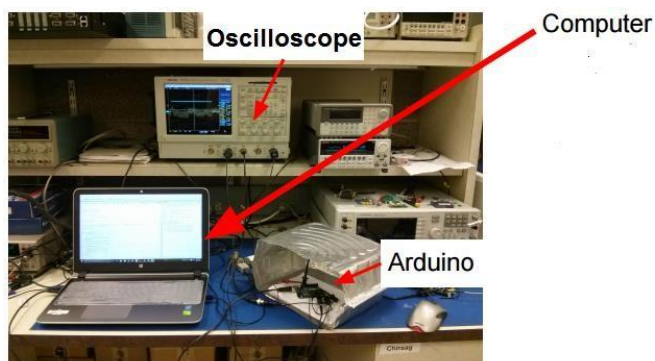


Figura 3 - Plataforma para la toma de datos

Fuente: [3]

[4] *Novel Applications of Wavelet Transforms based Side-Channel Analysis*, Los autores del centro de investigación COMELEC – Francia, el 2011 realizaron una propuesta y estudio de viabilidad de un sistema para el estudio del campo magnético generado por un FPGA en la que esta implementado un algoritmo AES (*Advanced Encryption Standard*)-128 bits. Esta señal es capturada por un sonda tipo antena tipo Rohde Schwarz conectado a un osciloscopio Agilent 54855 Infiniium en la cual cuenta con una frecuencia de muestreo de 40Ghz, el análisis se realiza en MATLAB con la herramienta “wavemenu”, filtra y comprime la señal usando Wavelet además utiliza la técnica PCA para el reconocimiento de patrones.

[5] *A Case Study of Side – Channel Analysis Using Decoupling Capacitor Power Measurement with the OpenADC*. Los autores de la universidad de Dalhousie – California desarrollaron un sistema de adquisición de datos de bajo costo, formado por (OpenADC+FPGA+AVR), demuestra que el campo electromagnético proporciona mejores resultados que el estudio de una derivación de la corriente en el capacitor.

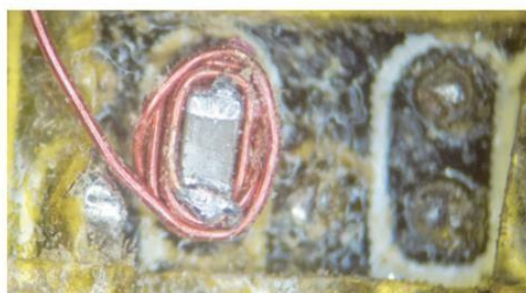


Figura 4- Enrollado de cable de cobre AWG34 alrededor de un capacitor

Fuente: [5]

Capítulo 2

Información preliminar y estudios previos

Con el fin de tener un estudio de un análisis de señales no estacionaria mediante transformada *Wavelet* se planteó mejorar la señal capturada de una señal de radiación de un dispositivo electrónico para posibles análisis de reconocimiento de características o estudio de ataques de canal lateral (SCA). Para dicho fin se hace un estudio previo de algunas definiciones para luego en capítulos posteriores realizar enfocarnos más en la realización de cada etapa desde la adquisición de datos hasta la evaluación de la señal de tal forma que se deje lista la señal para poder seguir con diferentes estudios posteriores de la señal.

2.1 Señales No estacionarias

Las señales en la naturaleza varían en gran medida y a veces puede ser difícil para caracterizarlas. Por lo general, diferenciamos entre las señales deterministas y estocásticas (aleatorias), donde una señal determinista se conoce de manera explícita y un proceso estocástico es una realización de una colección de señales que se caracteriza por propiedades como valor esperado y la varianza. También existen diferencias entre las señales estacionarias y no estacionarias donde una señal estacionaria tiene propiedades invariantes en el tiempo, es decir, propiedades que no cambian con tiempo. El tiempo es una manera fundamental de estudiar una señal, pero también puede estudiar la señal en otras representaciones, donde uno de los más importantes es la frecuencia [6].

El análisis tiempo-frecuencia de señales no estacionarias y variables en el tiempo, en general, en los últimos años se está investigando con mayor frecuencia.

2.2 El consumo de potencia en dispositivos CMOS

El consumo de energía en los microcontroladores y microprocesadores ha sido estudiado y sigue siendo tema de estudio esto se ve en un gran número de artículos de los pasados y presentes años. Los cambios de potencia en el dominio del tiempo reflejan un comportamiento interno del circuito de hardware y los datos procesados. Sin embargo estas huellas digitales explotables para las unidades lógicas específicas se mezclan con numerosas trazas de voltaje o corriente. La cuestión que se plantea es poder mejorar la señal capturada utilizando la herramienta de la transformada de *Wavelet*.

El estudio del dispositivo se enfoca al dispositivo CMOS fabricado por “*ATMEL*”. Estos dispositivos CMOS se construyen a base de celdas lógicas básicamente por transistores como se aprecia en la figura 5.

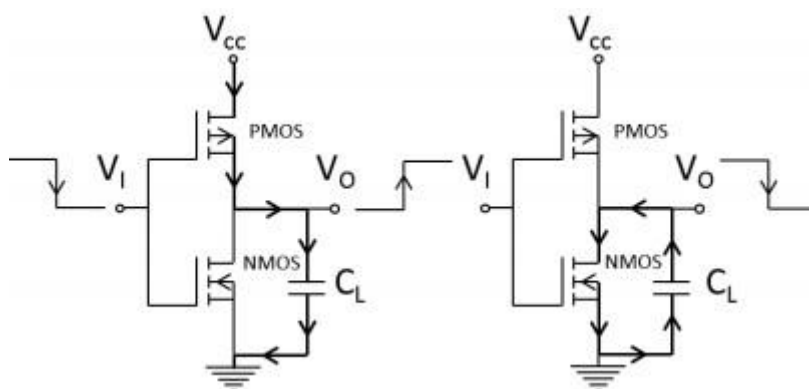


Figura 5- Compuertas básicas tipo CMOS

Fuente: internet [2]

Se considera que por la conmutación de alto a bajo o de bajo a alto (consumo dinámico) de las compuertas básicas existirá un consumo de potencia, sin embargo existen otras como [3]:

- Temperatura.
- Pendiente del ciclo del reloj.
- Tensión de la alimentación.
- Ciclo de trabajo entre otros.

Este comportamiento de consumo dinámico dependiente del valor del dato y es lo que origina las fugas de información y permite el estudio de su radiación [4], [5].

2.3 Ataques por canal lateral

El Ataque de canal lateral (SCA) tiene la finalidad de extraer ciertas características mediante la explotación de la fuga física del dispositivo en estudio. Diferentes tipos de hardware tienen fuga de información física cuando está realizando operaciones. Tal información puede ser: el consumo de energía, la radiación electromagnética y otros, estos se aprecian en la figura 6. Un atacante puede utilizar esta información de canal lateral, y realizar análisis matemáticos y podría ser capaz de extraer información secreta.



Figura 6-Fuga de información

Fuente: internet [11].

Los ataques de canal lateral intentan recuperar cierta información analizando una fuga de radiación en el dispositivo. Es un tipo de ataque no invasivo, donde el “atacante” trata de explotar las fugas físicas de información observables durante el procesamiento. En función de la señal física analizada, se pueden clasificar a su vez en [11]:

- Ataques por Análisis de Consumo: Se observa el consumo eléctrico a través de una resistencia.
- Ataques por Análisis Electromagnético: Se analiza el campo electromagnético (EM) del microprocesador. Se capta la señal de radiación mediante una sonda.

Los ataques no invasivos son eficientes, y en algunas ocasiones, la simplicidad y la información de este tipo de ataques de la misma forma el costo de instrumentos no son tan caros como los que se necesitarían al hacer un ataques invasivos [12], han contribuido a su popularidad haciendo que este tipo de ataques representen un serio problema a la seguridad para la mayoría de dispositivos criptográficos. Por ello, la comunidad criptográfica está dedicando importantes recursos para su desarrollo al mismo tiempo el aumento de instrumentos matemáticos más que puramente físicos [13].

2.4 Radiación electromagnética en dispositivos CMOS

Emanaciones de EM surgen como consecuencia de flujo de corriente a raíz de las operaciones del procesamiento de datos. Cada componente de conducción de corriente del dispositivo produce sus propias emanaciones basadas en sus características físicas y eléctricas [11].

De las numerosas emanaciones, los inducidos por las operaciones de procesamiento de datos llevan más información valiosa relacionada con el procesamiento. En los dispositivos CMOS, idealmente, la corriente fluye sólo cuando hay un cambio en el estado lógico de un dispositivo. Además, todo el procesamiento de datos se controla típicamente por un "cuadrado de onda" con forma de reloj. Cada flanco de reloj desencadena una secuencia corta de estado cambiante eventos y corrientes en las unidades de procesamiento de datos [7].

Existe una relación entre los datos procesados en el interior de los dispositivos de cifrado y EM del mismo [14]. Esto es debido al hecho de que la radiación EM se deduce por la corriente que varía de acuerdo a la ley de Faraday como se muestra en la ecuación (1).

Al realizar unas medidas de campo EM a una distancia menor que aproximadamente la longitud de onda λ de la fuente dividido por seis¹ (zona de campo cercano), las señales pueden ser consideradas cuasi-estáticas, por lo que la ley de la ley de *Biot-Savart* es aplicable:

$$dB = \frac{\mu dlx\dot{r}}{4\pi r^2} \quad (1)$$

¹ Exactamente a $\lambda/2\pi$, siendo $\lambda = \frac{c}{f\sqrt{\epsilon_r}}$, donde: $c = 3 \cdot 10^8 m/s$, f = frecuencia (HZ) y ϵ_r = Permitividad relativa (en el vacío).

Donde el campo magnético depende de la corriente y la orientación del campo depende de la dirección de la corriente, según ello se demuestra que el campo electromagnético depende del dato procesado.

Por otro lado, con la ley de Faraday, se demuestra que la variación del flujo magnético ²alrededor de una sonda de medida producirá un voltaje en la misma, el flujo magnético a través de una superficie es:

$$\Phi_B = \int_S B \cdot dS \quad (2)$$

Φ_B : Flujo magnético que atraviesa a sonda.

B : Densidad de campo magnético.

dS : Diferencial de la superficie.

S : Superficie.

Por otro lado la fuerza electromotriz ε , en la sonda depende del ritmo de cambio del flujo, la variación de flujo constituye la Ley de Faraday:

$$\varepsilon = - \frac{d\Phi_B}{dt} \quad (3)$$

Donde Φ_B es el flujo magnético que atraviesa la sonda.

La característica esencial de la variación de flujo magnético a través de cualquier superficie es que induce un campo eléctrico E en el contorno que delimita la cobertura de la antena. La fuerza electromotriz inducida está definida como la circulación de este campo a los largo del contorno.

$$\varepsilon = \oint_C E \cdot dl \quad (4)$$

En la figura 7 se esquematiza la situación de la generación de campo eléctrico E en una sonda formada por una espira conductora situada dentro de un campo magnético B variable [15].

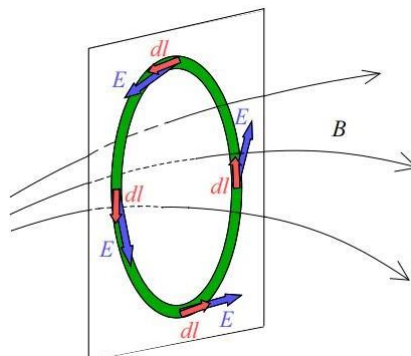


Figura 7 - Inducción de campo magnético B en antena

Fuente: internet [11].

² El flujo magnético es una medida de cantidad de magnetismo, es una proporcionalidad entre el flujo magnético a través de una superficie y el número de líneas de campo que la atraviesan.

Si igualamos las ecuaciones (1-4), entonces la superficie S a través de la que calculamos el flujo magnético Φ_B es una superficie delimitada por el contorno C donde se calcula la fuerza electromotriz \mathcal{E} .

$$\oint_C \mathbf{E} \cdot d\mathbf{l} = - \frac{d}{dt} \int_S \mathbf{B} \cdot d\mathbf{S} \quad (4)$$

Se utiliza una antena para registrar la radiación magnética. La antena debe colocarse cerca de la Dispositivo donde hay fuga de EM, con el fin de producir más fuerte radiación electromagnética en un campo cercano [12].

Capítulo 3

Estudio de las Herramientas de Hardware comerciales utilizadas para la medición

El uso de distintas herramientas de desarrollo en diferentes plataformas electrónicas ayuda a tener muchas alternativas al momento de elegir por cual se decide usar, para la elección de una u otra herramienta de hardware. Para su elección se debe tener en cuenta diferentes características como el precio, tiempo de entrega, fidelidad, historial y ejemplos realizados con anterioridad.

En este capítulo se explicará la parte de integración de hardware para hacer posible la toma de datos de señales de baja potencia. También se abordarán las características de las herramientas utilizadas. Posteriormente, se detallará la forma de comunicación entre los mismos. En la literatura ([1] - [5]) para hacer este tipo de adquisición se ha usado osciloscopios - *datalogger* sumamente costosos, pues tienen características importantes como su tasa de muestreo y la cantidad de memoria para poder registrar los datos, en este trabajo se pretende realizar con herramientas más baratas que puedan sustituir estos equipos costosos tendiendo como apoyo una laptop. La idea que se tiene por realizar se aprecia en la figura 8, aquí se tiene un bosquejo del funcionamiento. Se pretende capturar la fuga de datos del microprocesador mediante una antena de cobre con área de abertura \vec{S} por donde pasará la radiación, esta antena lleva la señal del campo magnético \vec{B} a un señal eléctrica I , para luego ser llevada a una tarjeta llamada *OpenADC*, esta formará parte de un sistema de adquisición de datos junto a un FPGA, registrando los datos en una laptop mediante el puerto USB.

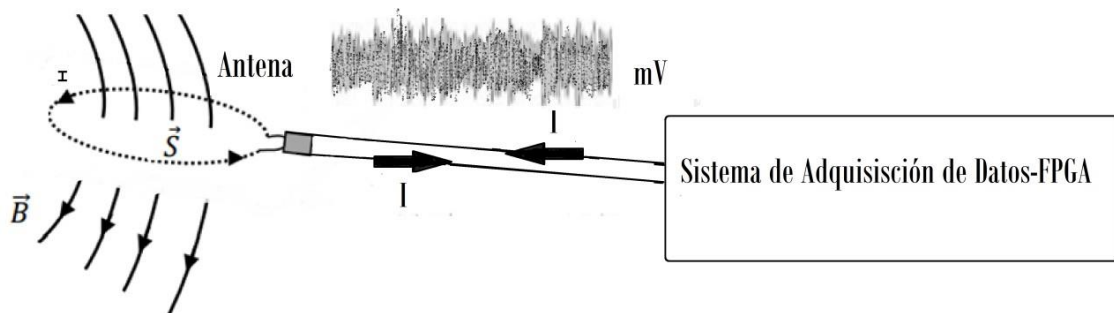


Figura 8-Sistema de Adquisición de Datos - FPGA.

Fuente: propia.

Las siguientes herramientas se usaron por su fidelidad de ejecución ya que ya han sido comprobados para otros fines, además se encuentra disponible su manual de funcionamiento [16] y una muestra de código para poder lograr una comunicación con el FPGA.

3.1 Sonda de Campo Magnético

Esta es básicamente un cable coaxial y se usa como medio de captura de la señal electromagnética, es un transductor cuyo principal inconveniente de tales sondas es su señal de salida muy baja (típicamente de 2 a 4 mV pico a pico. La debilidad de amplitud se compensa con el uso de una etapa de amplificación muy eficiente.

Una ventaja importante de este tipo de sensores inductivos es su banda ancha. Es decir, una frecuencia de resonancia que es mucho mayor que la frecuencia más alta de radiación del chip analizado. Información detallada del proceso de construcción se encuentra en [17], [18].



Figura 9-Sonda de Campo Magnético

Fuente: <http://store.newae.com/h-field-probe-old-version/>

Esta clase de sonda, tiene como ventaja de ser fácil de montar, y de conectarlo a través de su conector SMA macho.

3.2 Amplificador de bajo ruido

La señal de entrada necesita una ganancia puesto que la señal de salida de la sonda es muy débil para ser leída, en la imagen Figura 10 se muestra un amplificador de +20db en la banda de 0.1-1000 MHz y un acople de 50 ohm de entrada, el diseño del circuito del amplificador y características se encuentran en [28].



Figura 10-Amplificador de señal de baja potencia

Fuente: <http://store.newae.com/low-noise-amplifier-assembled-tested/>

Esta tarjeta presenta una facilidad de conexión y ha sido probado y realizado en [19] .

3.3 OpenADC

Una vez amplificada la señal se necesita digitalizar la señal, ello se realiza utilizando un convertidor analógico - digital AD9215 incorporado en la tarjeta, este cuenta con 10 bit enviando los datos de manera paralela, característica importante para la transmisión de datos de manera veloz, cuenta con una frecuencia de muestreo de hasta 105 Mhz.

La Tarjeta *OpenADC* se muestra en la figura 11 fue realizada por la empresa NEWAE cuyo principal tutor es el investigador y estudiante de doctorado *Collyn O'Flynn* que cuenta con varios proyectos relacionados, ha desarrollado la *ChipWhisperer* [19] ver figura 12, primera herramienta donde estudia ataques de análisis de potencia de canal lateral a tarjetas utilizando su propias herramientas, la cual parte de su investigación es el diseño *OpenADC* [16].

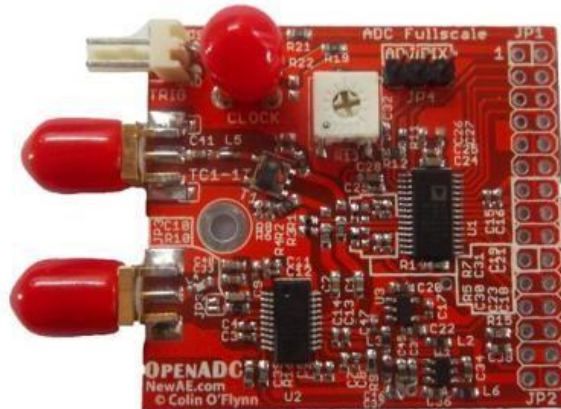


Figura 11-Tarjeta OpenADC

Fuente: <http://store.newae.com/OpenADC/>

OpenADC es un hardware abierto y solución de software abierto para necesidades de digitalización rápida. Diseñado para adaptarse a una variedad de tarjetas de desarrollo FPGA, a continuación se presenta sus principales características:

- La velocidad de muestreo es alto, puede llegar hasta 105 Mhz.
- El costo es reducido comparado con otras de características similares.
- Tiene ganancia externa de entrada entre -10dB a 55dB.
- Su ancho de banda es aproximado de 50 Mhz.
- Tiene la opción de tener un clock externo.

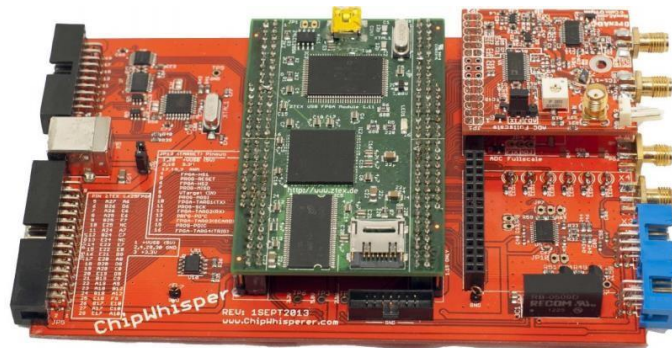


Figura 12 - Proyecto ChipWhisperer

Fuente: <http://store.newae.com/OpenADC/>

3.4 FPGA

La etapa de mayor exigencia en cuanto a configuración fue precisamente realizar un diseño con el fin de lograr una toma de datos con las mejores características posibles de memoria y velocidad de comunicación, un diseño que permita poder recibir los datos digitalizados, almacenarlos y a la vez enviando bytes por el puerto serial a la computadora donde los datos son almacenados en una hoja de Excel. El diseño permite el uso de un la tarjeta *OpenADC*, pero sin uso de entrada externa de reloj ni menos uso de *trigger*, se implementó una arquitectura que utilice comunicación serial y lectura de datos de la información respetando el protocolo de comunicación serial, las subrutinas están creadas en VHDL y Verilog.

El uso de la plataforma FPGA es básicamente por una importante propiedad, ésta plataforma permite poder procesar datos a frecuencias medias – altas, de tal manera que hay un sincronismo con un ADC9215 de la tarjeta *OPENADC* que presenta un rango de operación parecidas a las de la FPGA.

Un FPGA normal contiene más de un millar de estos elementos lógicos. Este dispositivo contiene grupos de matrices, básicamente formado por dos grupos, bloques lógicos genéricos y conmutadores programables tal como se puede observar en la figura 13 [20].

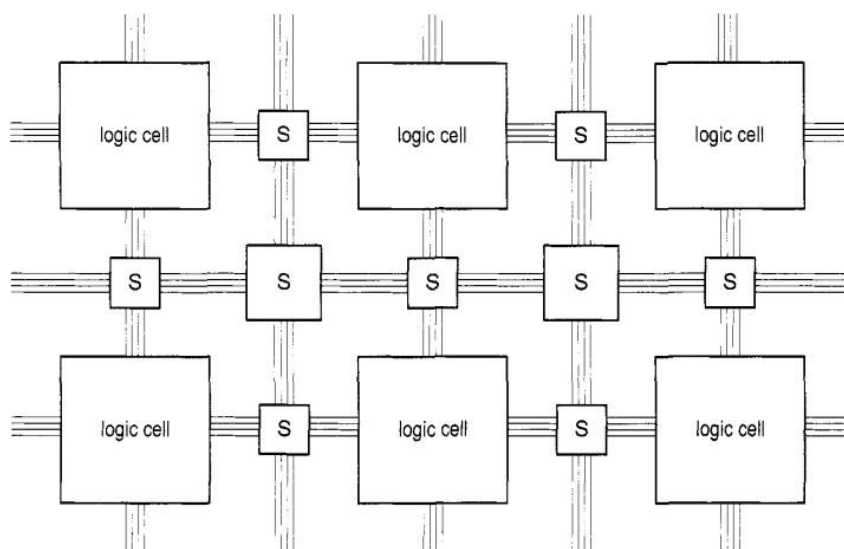


Figura 13- Arquitectura de una FPGA
Fuente: *FPGA Prototyping By VHDL Examples*

3.4.1 Spartan – 6-LX9

Son las FPGAs de mayor equilibrio entre costo y utilidad, líderes de la industria aeroespacial, un número diverso de protocolos de E / S admitidas. Estos dispositivos son ideales para una variedad de aplicaciones avanzadas que se encuentran en el puente de información y entretenimiento automotriz, consumo, la automatización industria entre otros [21].

La familia *Spartan6* de la empresa *Avnet Electronic* proporciona capacidades de conducción de la integración del sistema con el menor costo total para aplicaciones de alto volumen. Proporciona densidades que van desde 3.840 a 147.443 celdas lógicas. Construida sobre una madura tecnología de proceso de cobre de bajo consumo de 45 nm que proporciona el equilibrio óptimo de costo, potencia y rendimiento, la familia Spartan-6 ofrece un nuevo y más eficiente, de doble registro de tablas lógicas de búsqueda de 6 entradas (LUT) y una rica selección de bloques a nivel de sistema incorporadas. Estos incluyen 18 Kb (2 x 9 Kb) RAM de bloque, controladores de memoria SDRAM, bloques de gestión de reloj de modo mixto mejorados [22].

El Spartan-6 FPGA LX9 proporciona un entorno de hardware completo para los diseñadores para acelerar su tiempo de comercialización. El kit proporciona una plataforma estable para desarrollar y probar los diseños dirigida al bajo costo y bajo consumo de energía Xilinx Spartan-6 FPGA. El dispositivo Spartan-6 FPGA LX9 instalado ofrece un entorno de creación de prototipos para demostrar efectivamente los beneficios mejorados de bajo coste. En la Figura 14 se muestra la tarjeta electrónica FPGA Spartan-6 LX9, tarjeta que se usará en el desarrollo del proyecto, además se aprecia las distintas partes que la conforman, las mismas que se detallan en [22] y [23].

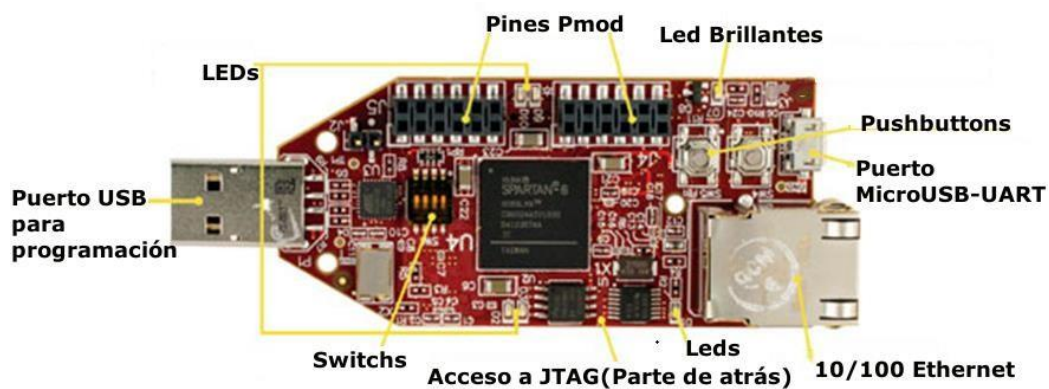


Figura 14-FPGA-Spartan - 6-LX9

Fuente: <http://news.softpedia.com/news/Avnet-Intros-the-Low-Cost-Xilinx-Spartan-6-LX9-FPGA-MicroBoard-Priced-at-89-182394.shtml>

3.4.1.1 Especificaciones

Existen una variedad de modelos de distintas empresas que fabrican FPGA, el diseñador utiliza según los requerimientos que desea. La tarjeta FPGA usado en este proyecto presenta las siguientes especificaciones [22]:

- Número de bloques de silicio : 1,430
- Número de Celdas Lógicas : 9, 152
- CLB Flip-Flops : 11,440
- Memoria RAM(Kb) 90
- Bloque RAM(18 Kb) 32
- Bloques totales de RAM(Kb) 576
- Bloques controladores de memoria 2
- Boques controladores de Clock(DCM+PLL) 2

Estas características proporcionan una alternativa a los productos programable *ASIC* de bajo costo personalizados con facilidad de uso sin precedentes. Los *FPGAs Spartan-6* ofrecen la mejor solución para los diseños de grandes volúmenes lógicos, diseños orientados al consumidor y aplicaciones embebidas de bajo costo.

3.4.2 El arreglo de compuertas programable

En este proyecto se usó un FPGA de la serie Spartan 6 de Xilinx. Esta familia presenta varias características que la hacen apropiadas para la lograr adquirir las señales, como son multiplicadores y bloques de memoria. En la figura 1515 1515151515se muestra el diagrama a bloques para los dispositivos de la familia *Spartan6*.

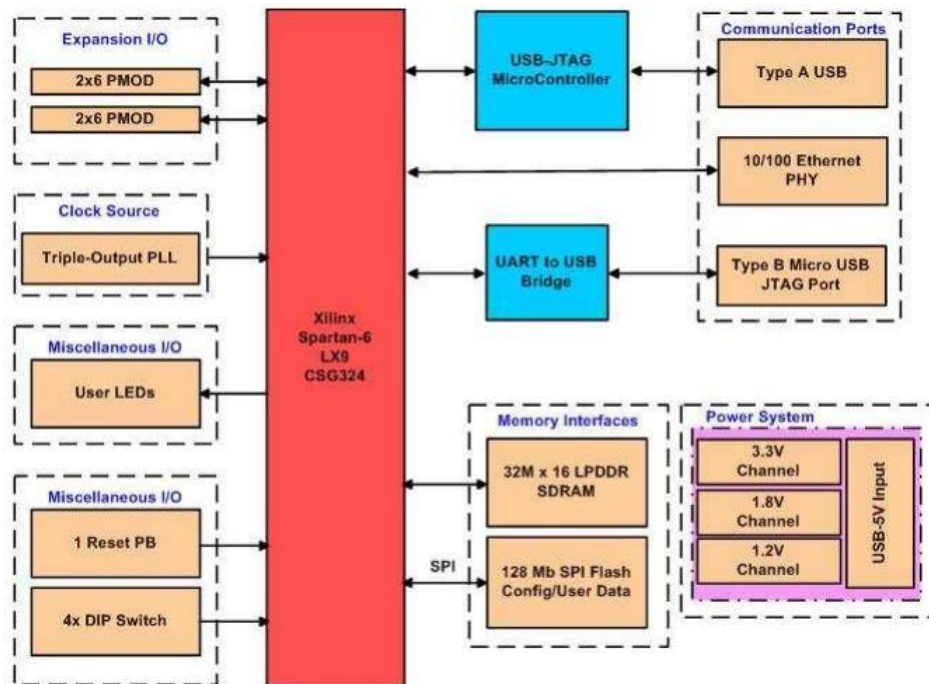


Figura 15-Spartan-6-LX9-Diagrama de Bloques

Fuente: internet [23]

Para conocer detalladamente la función de cada bloque se recomienda ir a su hoja de datos [23], para nuestro propósito se utilizó los bloques *PMOD* para hacer una conexión con la tarjeta *OpenADC*. Se usaron *leds* como indicadores, el botón de reseteo, para el almacenamiento temporal de datos, se usaron memorias *SDRAM* y memoria *SPI Flash* para guardar datos de configuración, para la comunicación con la laptop se realizó a través del puerto micro-*USB*, la alimentación es a través del puerto *USB* regulada a 3.3v.

3.4.3 Lenguaje de Diseño de Hardware(VHDL/Verilog)

Es el enfoque convencional para el diseño de sistemas digitales. Bloques programables pueden ser usados para implementar lógicas simples hasta algoritmos complejos como por ejemplo procesamiento de video, algoritmos de encriptación, etc. o quizás muy largos proyectos, tales como sistemas de control.

3.4.4 Descripción de la herramienta utilizada

La herramienta que se utilizó para el diseño es el software Xilinx ISE versión 6, está formada por 4 ventanas (Figura 16):

- **Ventana de archivos fuentes:** aquí se muestran los ficheros fuentes utilizados en el diseño y las dependencias entre ellos. También es aquí donde se elige el tipo de dispositivo donde se desea implementar el diseño.
- **Ventana de procesos:** Esta ventana muestra todos los procesos necesarios para la ejecución de cada etapa de diseño. La lista de procesos se modifica dinámicamente dependiendo del tipo de fuente seleccionado en la ventana de archivos fuente [24].
- **Ventana de edición:** aquí se escribe la edición de los ficheros fuentes o bien se puede editar también el diseño en caso de esquemas o máquinas de estado.
- **Ventana de información o mensajes de consola:** Esta ventana está situada en la parte inferior. Muestra mensajes de error o información emitidos por la ejecución de los programas de compilación, implementación, etc.

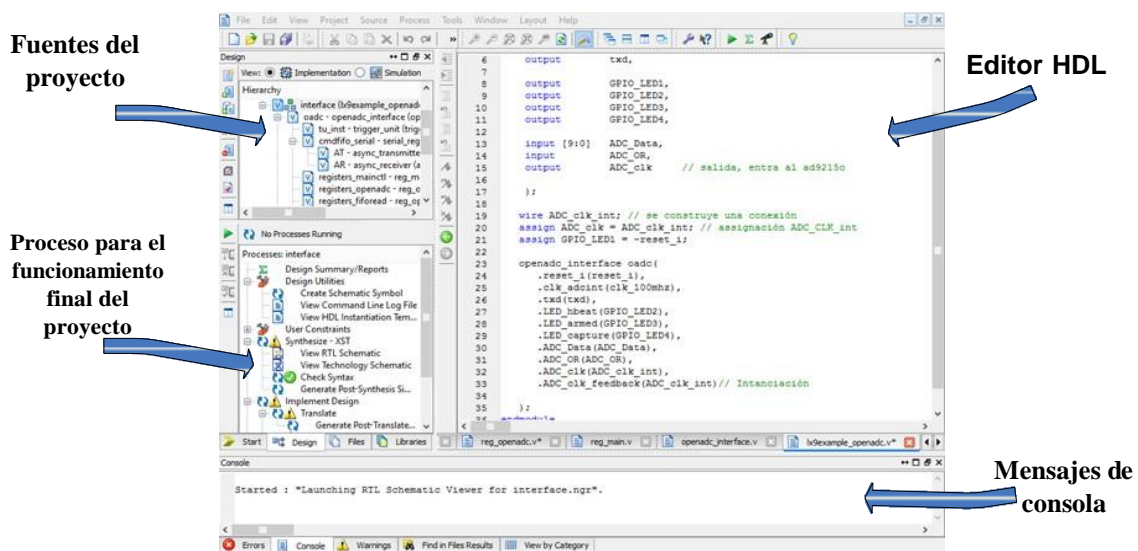


Figura 16-ISE de Xilinx-Project Navigator- Herramienta de Descripción

Fuente: propia

ISE de Xilinx puede usarse para editar tanto en VHDL como en Verilog hasta en MyHDL. Por motivos propiamente del código original se tuvo que adaptarse en Verilog y parte de código en VHDL.

3.4.5 Ventajas de una arquitectura FPGA

La mejora de la tecnológica en las últimas décadas han empujado hacia delante de manera significativa las mejoras de rendimiento de los sistemas FPGA. La industria FPGA ya ha evolucionado como dominante para la elección final por el usuario, son rápidos y baratos en contraste con otros dispositivos, principalmente debido a los siguientes méritos:

- **Bajo Costo.** - Son cómodos y fácil de conseguir, el entorno de diseño es amigable, además el requerimiento de energía es mucho menor comparado con los microcontroladores, su arquitectura de los FPGA se basan en *LUTs*.
- **Alta Velocidad.**-El tiempo de ejecución de un diseño es mucho menor a comparación a la tecnología ASIC además de microcontroladores, puesto que la tecnología FPGA se basa en *LUTs*.
- **Mantenimiento a largo plazo.**- los chips FPGA son campo actualizable y no requieren el tiempo y los gastos involucrados con nuevo diseño ASIC. Protocolos de comunicación digitales, por ejemplo, tienen especificaciones que pueden cambiar con el tiempo, y las interfaces basadas en ASIC pueden causar mantenimiento y con visión de compatibilidad desafíos. Siendo reconfigurable, chips FPGA pueden seguir el ritmo de las futuras modificaciones que pudieran ser necesarias. Como un producto o sistema madura, puede realizar mejoras funcionales sin tener que gastar tiempo rediseño de hardware o modificar el diseño de la placa [34].
- **Confiabilidad.**-Los sistemas basados en el procesador están continuamente en riesgo de tareas de tiempo crítico como los FPGAs, que no utilizan sistemas operativos, minimizar los problemas de fiabilidad con verdadera ejecución paralela y hardware determinista dedicado a cada tarea.
- **Desempeño.**- Tomando ventaja del hardware paralelo, FPGAs supera la potencia de cálculo de los procesadores de señales digitales (DSPs) rompiendo el paradigma de ejecución secuencial y logrando más por ciclo de reloj.

Capítulo 4

FPGA, Python, comunicación serial y almacenamiento de datos

En este capítulo se describe cómo el software fue entendido de su fuente original para poder crear nuestro propio sistema de adquisición de datos, usando las herramientas de software suministrado con el FPGA Spartan 6. El *ISE Project Navigator* es un entorno de programación que viene con la compra del FPGA o se puede descargar de la página web de la misma. El capítulo también describe cómo se capturó la data mediante el puerto serial de la Pc usando Python, para luego procesar los datos en Matlab.

En esta ocasión, Vhdl como Verilog serán participes para la configuración del FPGA Spartan 6, por otro lado Python se usará para el almacenamiento de los datos, una aplicación más que se une a las listas donde se puede usar este *software*. La facilidad de Python para el diseño de una interfaz es útil tomando en cuenta que en este caso se necesita rapidez en la comunicación serial y la opción de crear interfaz gráfica de manera que usa menores recursos de la pc, habilita la opción de usarlo como un programa de uso para cualquier pc.

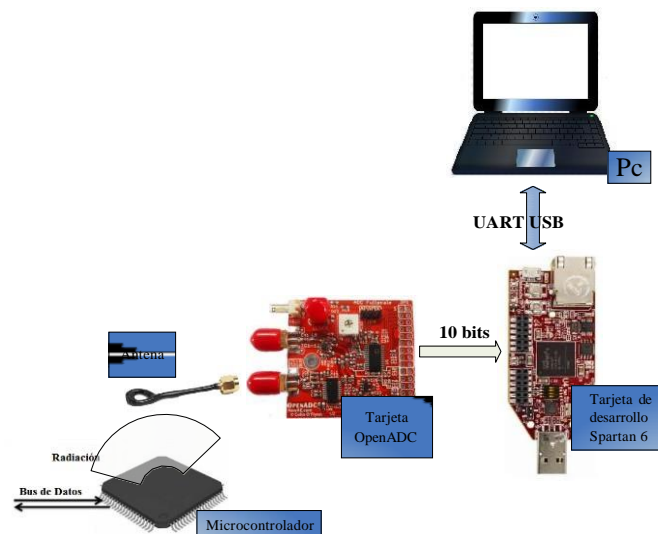


Figura 17-Esquema General del Proyecto

Fuente: propia.

En la figura 17 se muestra las componentes que se usan para lograr tomar datos. La radiación del microprocesador es capturada mediante la sonda convirtiendo la señal magnética a eléctrica. El hardware *OpenADC* consiste únicamente en un ADC de 10 bit

(ADC9215) que digitaliza la señal y envía los datos al FPGA para poder almacenar bloques de 48M y enviarlos por el puerto *micro_USB*.

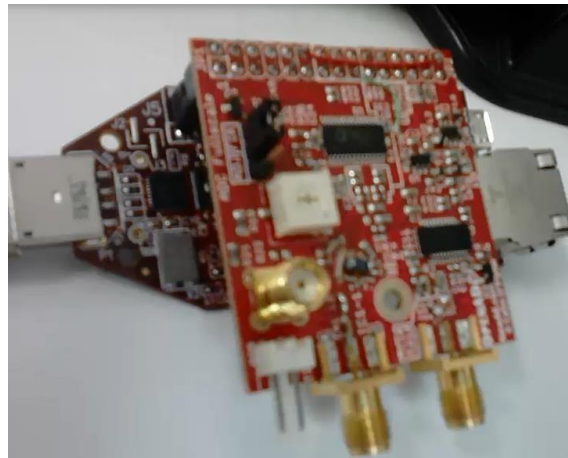


Figura 18 -OpenADC montado sobre una placa FPGA
Fuente: propia.

Esta placa se puede conectar a la mayoría de las tarjetas de desarrollo FPGA con los suficientes pines de entrada y salida disponible. La tarjeta *OpenADC* montado sobre una placa de desarrollo FPGA comercial se muestra en la figura 18. La tarjeta FPGA proporciona un control, interfaz USB y una memoria de muestras 48M.

4.1 Proceso para el diseño de FPGA

Para lograr implementar una circuitería funcional en un FPGA, se debe pasar por una serie de etapas, las cuales son mostradas en la figura 19 , estos procesos **¡Error! No se encuentra el origen de la referencia.** están ordenadas con el fin de obtener el diseño definitivo. En cualquier caso, si la funcionalidad del diseño no es la esperada, como resultado de realizar la simulación, es necesario revisar el código de VHDL o Verilog para conseguir la funcionalidad requerida. En la figura 19 aparecen en un recuadro tanto las etapas de diseño, mapeo como la de ruteo, esto es porque a estas etapas en conjunto se les conoce como implementación. A continuación se da una explicación de cada una de las etapas.

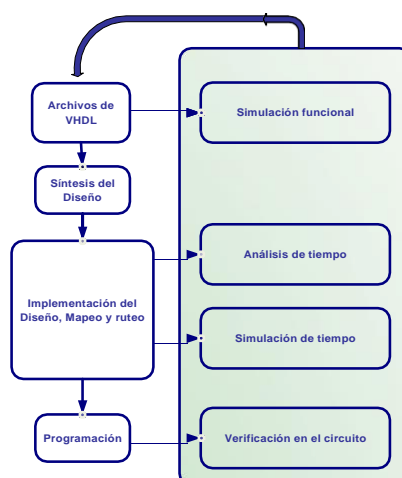


Figura 19-Etapas para el diseño en FPGA
Fuente: Internet [35]

4.1.1 Creación de archivos de diseño en VHDL

En esta parte se genera el programa en VHDL, el cual es un archivo de texto con la descripción de *hardware* deseada. En este archivo se describen los diferentes componentes que forman el circuito diseñado. El texto completo puede quedar en un sólo archivo, o puede ser dividido en varios segmentos para tener archivos más compactos. Si se hace esto último, existen las unidades de diseño que son necesarias al usar VHDL, estas unidades de diseño son: Entidad, Arquitectura, Configuración, Declaración de paquetes, Cuerpo del paquete [27]. Los archivos de diseño pueden ser realizados con cualquier procesador de texto, aunque existen herramientas para simulación o síntesis que contiene su propio editor de texto, muchos de estos facilitan al diseñador.

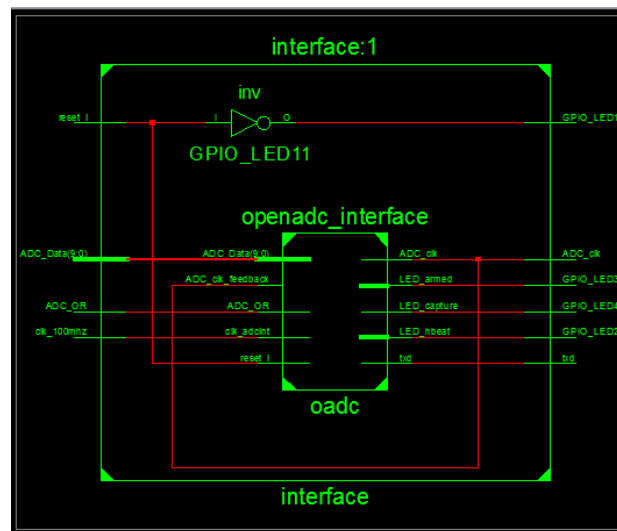


Figura 20- Interfaz OpenADC - Spartan6

Fuente: propia

El resultado de realizar las configuraciones del hardware queda expresada en un bloque como el de la figura 20, la cual está configurado para que que haga posible la comunicación de los datos de la tarjeta *OpenADC* al FPGA. Además de un boton de reset, LED de alertas y los más importante la unión de los ciclos de reloj de ambas herramientas para que estén sincronizadas.

4.1.2 Simulación del diseño

Una vez finalizada la descripción de VHDL-Verilog, es importante saber si el diseño así generado realizaría la función deseada de tal manera que se pueda ajustar. Esto se puede lograr con la simulación del diseño, la cual además es útil para verificar la sintaxis del programa o algunos errores de conexión interna. Hay que tomar en cuenta que en este punto hace falta considerar algunos factores, como los retardos de tiempo inherentes de los componentes, por lo que esta simulación es sólo una primera aproximación. Existen varias herramientas de simulación, se ha utilizado *ISE Foundation* pero existen varios otros como por ejemplo *Active-HDL*, *ModelSim*, *CADENCE HDL*, *Leonardo Spectrum*, *MAXPLUS*, entre otros [27]. Cada una de estas herramientas permite la descripción, síntesis, simulación y programación de los dispositivos lógicos programables.

4.1.3 Síntesis del diseño

En esta parte se crea una descripción a nivel de compuertas del diseño obtenido. El resultado de esta etapa es la obtención de una lista con las compuertas necesarias para implementar un diseño, y además de todas las conexiones necesarias entre las mismas. En esta etapa se puede realizar la simulación funcional, para verificar que el arreglo de compuertas siga efectuando la función deseada. Las componentes utilizadas se muestran en la figura 21.

```

=====
HDL Synthesis Report

Macro Statistics
# Adders/Subtractors                                : 15
 16-bit adder                                         : 1
 16-bit subtractor                                    : 1
 17-bit adder                                         : 2
 17-bit subtractor                                    : 1
 2-bit adder                                          : 1
 2-bit addsub                                         : 1
 25-bit adder                                         : 1
 32-bit adder                                         : 4
 4-bit adder                                          : 1
 5-bit adder                                          : 1
 9-bit adder                                          : 1
# Registers                                           : 91
 1-bit register                                       : 43
 10-bit register                                      : 4
 16-bit register                                      : 3
 17-bit register                                      : 2
 2-bit register                                       : 4
 25-bit register                                      : 3
 32-bit register                                      : 12
 4-bit register                                       : 5
 5-bit register                                       : 1
 6-bit register                                       : 1
 8-bit register                                       : 10
 9-bit register                                       : 3
# Comparators                                         : 13
 1-bit comparator equal                              : 1
 10-bit comparator greater                           : 1
 16-bit comparator equal                              : 1
 16-bit comparator greater                           : 2
 32-bit comparator equal                              : 2
 32-bit comparator greater                           : 1
 4-bit comparator greater                             : 1
 8-bit comparator greater                             : 2
 9-bit comparator greater                             : 2
# Multiplexers                                        : 239
 1-bit 2-to-1 multiplexer                             : 38
 1-bit 8-to-1 multiplexer                             : 1
 16-bit 2-to-1 multiplexer                             : 5
 2-bit 2-to-1 multiplexer                             : 6
 25-bit 2-to-1 multiplexer                             : 2
 3-bit 2-to-1 multiplexer                             : 5
 32-bit 2-to-1 multiplexer                             : 2
 4-bit 12-to-1 multiplexer                             : 1
 4-bit 13-to-1 multiplexer                             : 1
 4-bit 15-to-1 multiplexer                             : 1
 4-bit 2-to-1 multiplexer                             : 32
 5-bit 2-to-1 multiplexer                             : 5
 6-bit 2-to-1 multiplexer                             : 5
 7-bit 2-to-1 multiplexer                             : 5
 8-bit 2-to-1 multiplexer                             : 127
 9-bit 2-to-1 multiplexer                             : 3
# Logic shifters                                     : 10
 10-bit shifter logical right                         : 1
 56-bit shifter logical right                         : 8
 88-bit shifter logical right                         : 1
# FSMs                                                : 3

```

Figura 21- Reporte de los bloques utilizados.

Fuente: propia

4.1.4 Mapeo

El mapeo usa la descripción de compuertas generadas por la herramienta de síntesis, distribuyéndolas entre los diferentes recursos con los que cuenta el FPGA, como son los bloques de memoria, multiplicadores, y otros.

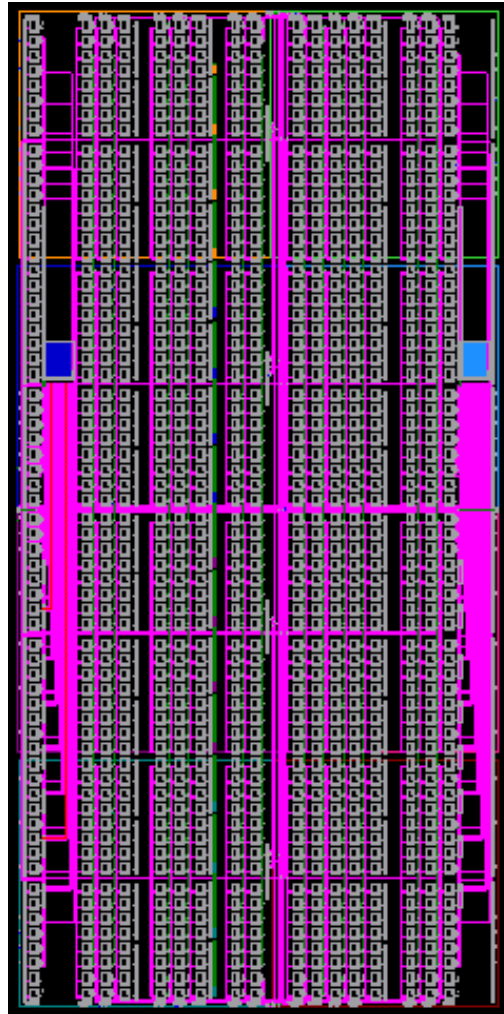


Figura 22-Mapeo

Fuente: propia

4.1.5 Diseño, mapeo y Ruteo

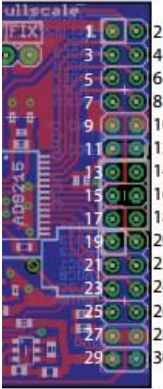
En el mapeo fueron definidos los recursos del FPGA que se deben usar para implementar el diseño. Con la colocación se seleccionan los componentes específicos a ser usados, y con el ruteo se definen las conexiones entre los diferentes componentes seleccionados. Después de esta etapa puede ser realizada otra simulación cuyo propósito es asegurarse que el diseño obtenido cumpla con los requerimientos de tiempo. Es en esta etapa donde intervienen factores como retardo entre los elementos.

4.1.6 Asignación de pines

Se utiliza la tarjeta *OpenADC*. Esta tarjeta se conecta con el FPGA a través de alguno de los puertos de expansión. Los pines *LNA*, *trigger*, no se utilizan para nuestro propósito, los

bits más significativos de la señal de salida del *OpenADC* es el pin 3 que va conectado a la señal ADC9 del FPGA y el bit menos significativo es el pin 24 que va conectado a la señal ADC0 del FPGA.

En la figura 23, presenta la relación de los pines de la tarjeta *OpenADC* [28] con su correspondencia con los pines de la tarjeta FPGA [23] según ello se crea un archivo UCF(“*User Constraints File*”) como se ve en la figura 25.



Pin Number	Pin Function	Pin Function	Pin Number
1	SMA to FPGA (Clock In)	Trigger Input to FPGA	2
3	ADC9	ADC_OR	4
5	ADC7	ADC8	6
7	ADC5	ADC6	8
9	GND	GND	10
11	DVCC (+2.25V to +3.6V)	DVCC(+2.25V to +3.3V)	12
13	No Connection	No Connection	14
15	No Connection	No Connection	16
17	No Connection	No Connection	18
19	ADC3	ADC4	20
21	ADC1	ADC2	22
23	ADC Clock	ADC0	24
25	LNA Gain Voltage (PWM)	LNA Gain Mode (Hi/Low)	26
27	GND	GND	29
29	AVCC (+3.1V* to 4.5V)	AVCC (+3.1V* to 4.5V)	30

Figura 23-pines de la tarjeta OPENADC

Fuente: internet [28]

Para efectuar la asignación de los pines, se debe seleccionar la opción *User Constraints Assign Package Pin*, de la ventana *Implementation Constraints File*; la cual se presenta en la figura 24.

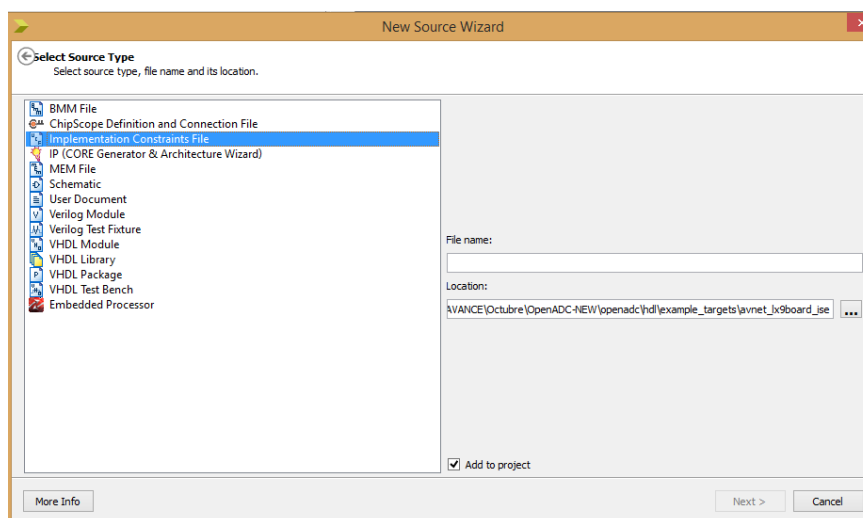


Figura 24-Configuración de los pines del FPGA-OPENADC

Fuente: propia

```

NET ADC_Data[9]          LOC=F16    | IOSTANDARD = LVCMOS33 | TNM_NET = ADCClock;
NET ADC_Data[7]          LOC=C17    | IOSTANDARD = LVCMOS33 | TNM_NET = ADCClock;
NET ADC_Data[5]          LOC=C18    | IOSTANDARD = LVCMOS33 | TNM_NET = ADCClock;
NET ADC_OR               LOC=G14    | IOSTANDARD = LVCMOS33 | TNM_NET = ADCClock;
NET ADC_Data[8]          LOC=D17    | IOSTANDARD = LVCMOS33 | TNM_NET = ADCClock;
NET ADC_Data[6]          LOC=D18    | IOSTANDARD = LVCMOS33 | TNM_NET = ADCClock;

NET ADC_Data[3]          LOC=H12    | IOSTANDARD = LVCMOS33 | TNM_NET = ADCClock;
NET ADC_Data[1]          LOC=G13    | IOSTANDARD = LVCMOS33 | TNM_NET = ADCClock;
NET ADC_clk              LOC=E16    | IOSTANDARD = LVCMOS33 | SLEW=FAST | DRIVE=12;

```

Figura 25- Archivo UCF (“User Constraints File”)

Fuente: propia

4.1.7 Programación

Después de haber realizado toda la configuración del FPGA esta se guarda en un archivo *bitstream*.

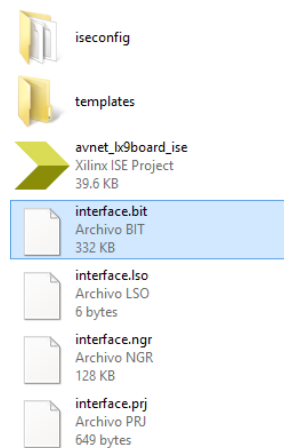


Figura 26-Archivo de programación *bitstream*

Fuente: propia

La programación consiste en enviar este archivo al FPGA para que establezca las conexiones apropiadas entre los diferentes componentes del mismo, en este momento es posible obtener el archivo necesario para programar el FPGA, conocido como *bitstream* tal como se ve en la figura 27.

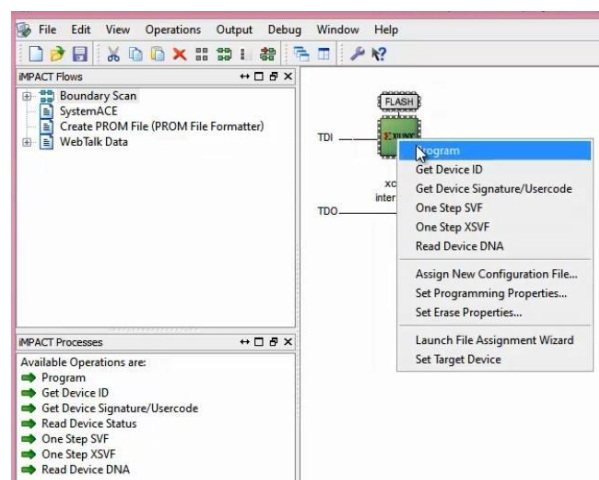


Figura 27-Programación del "Spartan6"

Fuente: propia

Este proceso se realiza mediante el computador conectado a la tarjeta *FPGA Spartan* por el puerto *USB*. Para que el *FPGA* sea reconocido por la laptop debe de instalarse un software de instalación sea *CP210xVCPInstaller_x64* o *CP210xVCPInstaller_x32* según el procesador de nuestra *PC* o *laptop* sea de 64 bits o de 32 bits.



Figura 28-Software de Instalación de FPGA

Fuente: propia

4.2 Almacenamiento de la programación en memoria

Dado que el dispositivo a usar es una *FPGA*, su naturaleza es *RAM*, es decir que cada vez que se suspenda la fuente de alimentación la configuración del dispositivo desaparece deben configurarse cada vez que se alimente a energía. El método más común para guardar la programación *FPGAs* - *Xilinx* es mediante el uso de *Xilinx PROM*. Archivos *PROM* incluyen información sobre la configuración del dispositivo. Varios formatos de archivo *PROM* están disponibles: *MCS*, *EXO*, *TEK*, *HEX*, *UFP*, *BIN*, e *ISC* [38]. El archivo *PROM* que usaremos será de formato *.mcs*. Algunas familias *FPGA* de *Xilinx* se pueden configurar mediante dispositivos *SPI* o *BPI flash* compatible. Debe programar estos dispositivos de memoria que utilizan archivos *PROM* creados a partir del archivo “*bitstream*” mencionado líneas antes.

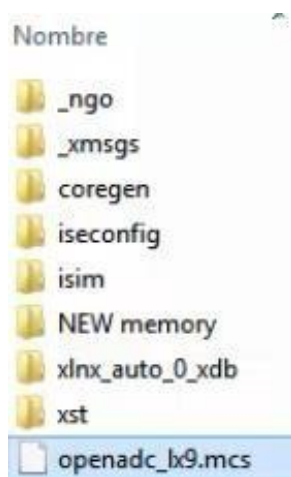


Figura 29-Archivo “.mcs”

Fuente: propia

4.3 Python 2.7.3

Python es un lenguaje de programación interpretado y orientado a objetos creado por Guido Van Rossum en 1991. Python es un lenguaje de programación poderoso, pero a su vez muy fácil de aprender y entender [30].

Luego de haber muestreado y digitalizada la señal de entrada a través de la tarjeta *FPGA* es necesario guardar los datos para luego ser analizados en *Matlab*. Utilizando la herramienta de programación *Python* se realiza la adquisición obtenidos por medio del *UART* para llevar a cabo la rutina de almacenamiento.

Para poder programar el puerto serie por Python evidentemente tenemos que tener instalado Python, que según la distribución de Windows ó Linux que tengamos, Debian, Ubuntu, Mandriva lo instalaremos con cualquier gestor de instalación que tengamos.

El software está escrito en lenguaje Python, por lo que es portable a cualquier sistema operativo con intérprete Python (Unix, OSX, Windows), la versión presentada se probó para sistemas *windows xp* y *window 8*. Las librerías de instalación depende del sistema operativo de la pc.

4.3.1 Características.

El software es implementado en Python. Python fue elegido por una variedad de razones:

- Es nativo “*Cross - platform*”³.
- Realización de GUI sencillo a través de PySide.
- Python es un lenguaje de programación modular. El usuario puede generar módulos reutilizables por otros programas [40]. De una forma fácil se puede enlazar con otros lenguajes incluyendo C/C++ y Matlab.
- Tiene una larga colección de módulos que hacen posible el desarrollo de proyectos de forma sencilla.
- Se puede realizar gráficos en 2D y 3D.
- Es de código abierto (libre) y gratuito, o que hace que Python pueda ser libremente usado y distribuido, inclusive para propósito comercial. La Licencia Python es administrada por la Fundación de Software de Python (“*Python Software Foundation*”).
- Python puede ser ejecutado en Windows, Linux/Unix, Mac OSX, además existen versiones que pueden ser usadas en máquinas virtuales.NET y java [31].

³ <http://movilforum.com/conoce-que-es-el-cross-platform/>

Capítulo 5

Representaciones Tiempo-Frecuencia

Las representaciones tiempo-frecuencia se aplica a señales con contenido en frecuencia variante en el tiempo. Estas señales se pueden representar adecuadamente mediante una distribución tiempo-frecuencia, la cual puede mostrar la forma en la cual se distribuye la energía de la señal. Así, en el procesamiento de la señal, se pueden aprovechar las características producidas por la concentración de la energía en dos dimensiones (tiempo y frecuencia) en vez de sólo una.

Así, mientras que una función en el dominio temporal indica cómo la amplitud de la señal cambia en el tiempo, su representación en el dominio de la frecuencia permite conocer cuan a menudo esos cambios tienen lugar. La vinculación entre estas dos presentaciones la brinda la Transformada de Fourier cuya idea fundamental es la de descomponer la señal en la suma pesada de funciones sinusoidales. Si bien la Transformada de Fourier en muchas situaciones es de gran utilidad, no resulta en todos los casos apta para analizar señales de la vida real, que son normalmente de duración finita y aun a veces de corta duración. Comparar señales del tipo de las sísmicas o las biomédicas, como así también los transitorios y las señales de radar con las señales sinusoidales que se extienden en el tiempo de $-\infty$ a $+\infty$ no resulta ser lo más adecuado, repasaremos brevemente la teoría de Fourier y la teoría de Fourier por ventanas.

5.1 Transformada de Fourier

En el siglo 19th Fourier demostró que cualquier función periódica se puede expresar como una suma infinita de funciones exponenciales complejas periódicas. En otras palabras, cualquier función periódica se puede expresar a través de funciones mucho más simples. Esta idea más tarde se generalizó a las funciones y discreta no periódicas, por lo que era conveniente para el cálculo rápido.

5.1.1 Definición

La transformada de Fourier nace a partir del límite de operatividad de una serie de Fourier con señales aperiódicas. Partiremos desde la definición de la serie de Fourier.

La serie de Fourier Desde el punto de vista matemático puede expresar cualquier función periódica $f(t)$, con un periodo de T , que es: $f(t) = f(t + 2\pi) = \dots = f(t + 2n\pi)$ para $n \in \mathbb{N}$, se puede expresar como una combinación lineal de todas las funciones senos y cosenos. La serie de Fourier se muestra en la ecuación1:

$$f(t) = a_0 + \sum_{n=1}^{\infty} a_n \cos(n\omega_0 t) + \sum_{n=1}^{\infty} b_n \sin(n\omega_0 t) \quad (1)$$

$$a_0 = \frac{1}{T_0} \int_0^{T_0} f(t) dt = \frac{1}{T_0} \int_{-T_0/2}^{T_0/2} f(t) dt$$

$$a_n = \frac{2}{T_0} \int_0^{T_0} f(t) \cos(n\omega_0 t) dt = \frac{2}{T_0} \int_{-T_0/2}^{T_0/2} f(t) \cos(n\omega_0 t) dt$$

$$b_n = \frac{2}{T_0} \int_0^{T_0} f(t) \sin(n\omega_0 t) dt = \frac{2}{T_0} \int_{-T_0/2}^{T_0/2} f(t) \sin(n\omega_0 t) dt$$

Estas igualdades se dan por la propiedad de periodicidad de estas funciones, entonces:

Es indistinto integrar $T_0/2 < T < T_0/2$ o sobre el intervalo $0 < T < T_0$

ω_0 Es la frecuencia fundamental que está inversamente proporcional al periodo de la función como lo muestra la ecuación

$$\omega_0 = \frac{2\pi}{T_0} \quad (2)$$

La serie de Fourier puede ser traducido a números complejos, como primer paso antes de formalizar transformada de Fourier.

$$\begin{aligned} \cos(n\omega_0 t) &= \frac{1}{2} (e^{in\omega_0 t} + e^{-in\omega_0 t}) \\ \sin(n\omega_0 t) &= \frac{1}{2i} (e^{in\omega_0 t} - e^{-in\omega_0 t}) \end{aligned}$$

Entonces reemplazando en la ecuación 1 se tiene una nueva expresión:

$$f(t) = a_0 + \sum_1^{+\infty} \frac{a_n - jb_n}{2} e^{in\omega_0 t} + \sum_1^{+\infty} \frac{a_n + jb_n}{2} e^{-in\omega_0 t}$$

Para reducir la ecuación de la serie de Fourier a una expresión de números complejos podemos substituir expresiones, creando un variable c_n , donde se define como:

$$f(t) = \sum_{-\infty}^{+\infty} c_n e^{in\omega_0 t}$$

$$c_n = \frac{1}{T_0} \int_{-T_0/2}^{T_0/2} f(t) e^{-in\omega_0 t} dt,$$

$$c_n = \begin{cases} \frac{a_n - ib_n}{2} & ; & n > 0, \\ a_0 & ; & n = 0, \\ \frac{a_n + ib_n}{2} & ; & n < 0, \end{cases} \quad (3)$$

$$n = 0, +/ - 1, +/ - 2, \dots$$

La ecuación 3 es la expresión compleja de la serie de Fourier de señales periódicas, c_n son los coeficientes complejos de Fourier.

Para señales no periódicas se diseñó la Transformada de Fourier, Para aproximar las Transformadas de Fourier a partir de las series de Fourier se debe tener un solo periodo.

La idea es que una señal aperiódica sea igual a una señal periódica con un periodo infinito ($T_0 \rightarrow +\infty$). De la ecuación 2, se puede concluir que a medida que el periodo va incrementando ($\omega_0 \rightarrow 0$), ello significa que los armónicos están separados de manera infinitesimal⁴, podemos considerarlo como variable continua.

La transformada de Fourier puede ser considerado como el límite formal de la serie de Fourier cuando el periodo tiende al infinito [32]. Si se considera un periodo infinito ($T_0 \rightarrow +\infty$), entonces $\omega_0 \rightarrow 0$, podemos considerarlo como variable continua, por ello de la ecuación 3 se replantea:

$$\lim_{\omega_0 \rightarrow 0} f(t) \Rightarrow f(t) = \sum_{-\infty}^{+\infty} d_n e^{in\omega_0 t}$$

Cuando $\omega_0 \rightarrow 0$, la sumatoria se convierte en integral y los valores de d_n serán cada vez más pequeños.

$c_n T_0 \rightarrow$ es la envolvente, si se define $X(f)$ como la envolvente de $c_n T_0$, reemplazando la variable discreta $n\omega_0$ por una variable f continua, se tiene finalmente:

$$X(f) = c_n T_0 = \int_{-\infty}^{+\infty} f(t) e^{-i2\pi f t} dt \quad (4)$$

⁴ Un infinitesimal o infinitésimo se puede definir como una cantidad infinitamente pequeña



Figura 30-Transformada de Fourier
Fuente: Toobox4-Matlab

En la figura 30, se puede observar de manera representativa del resultado de la transformada de Fourier.

En este apartado se explica brevemente lo que hace y las desventajas en el contexto de esta tesis. La transformada de Fourier (FT) nos da una idea de las frecuencias que aparecen en la señal y con qué fuerza. Esto puede ser ilustrado por el ejemplo mostrado en la figura 31, aquí se muestra una señal estacionaria y la transformada de Fourier de la señal.

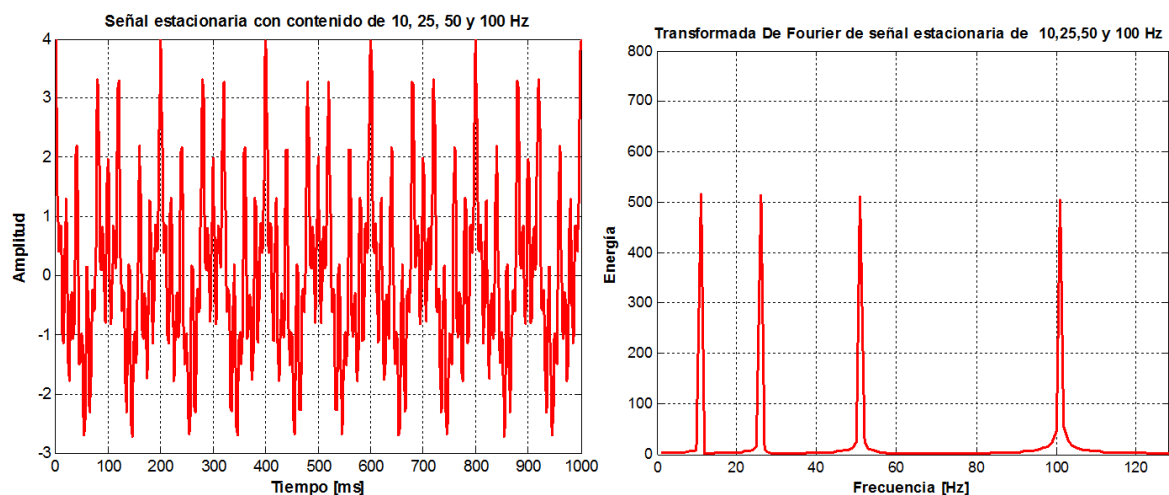


Figura 31 - Transformada de Fourier (Lado Derecho) de una señal estacionaria (Lado Izquierdo).
Fuente: Propia.

En la transformada de Fourier vemos cuatro picos, un pico a cada una de las frecuencias de la señal original consistía. Los picos deben tener la misma altura, pero debido a que el FT se calcula aquí en un conjunto de muestras en lugar de la función real de algunos picos son más altos que otros. Se aprecia a simple vista que la señal original parece un poco caótica la transformada de Fourier se puede leer fácilmente, sin mucho esfuerzo.

Ahora veamos si señal con las mismas frecuencias (10 Hz, 25 Hz, 50 Hz y 100 Hz), pero en lugar de todas las frecuencias al mismo tiempo (como lo fue en el ejemplo anterior), aquí las frecuencias son consecutivas. El gráfico resultante se muestra en la figura 32.

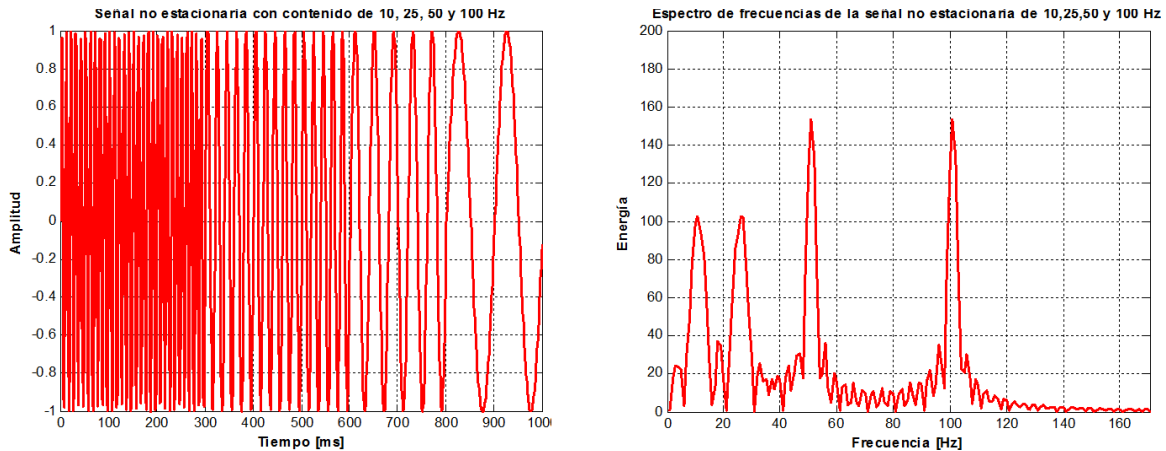


Figura 32 - Transformada de Fourier (Lado Derecho) de una señal No estacionaria (Lado Izquierdo).

Fuente: Toobox4-Matlab.

En la misma figura 32 vemos que aunque la señal original es bastante diferentes de la señal estacionaria, la Transformada de Fourier es muy similar. Se ve que el espectro de frecuencia existen cuatro picos, uno para cada frecuencia. También vemos aquí algo de ruido, esto es debido a las transiciones de una frecuencia a otra.

La mayoría de las señales que aparecen en la vida real son no estacionarias. Por ejemplo las trazas de potencia que se quiere analizar son no estacionarias. Entonces se concluye que las transformadas de Fourier no son la mejor opción para su análisis.

5.1.2 Observaciones

- La transformada de Fourier nace desde el hecho de que no todas las señales en la vida diaria son periódicas, es por ello que se dice que la TF es el límite de la serie de Fourier haciendo que el periodo sea infinito ($T_0 \rightarrow +\infty$) y desde luego ($\omega_0 \rightarrow 0$).
- La Transformada de Fourier se limita al estudio de señales de energía finita.

5.1.3 Desventajas

- La Transformada de Fourier es ampliamente utilizado en el procesamiento y análisis de señales obteniéndose muy buenos resultados, siempre y cuando las señales sean periódicas, no obstante no ocurre lo mismo con señales no estacionarias.
- La Transformada de Fourier detecta la presencia de una determinada frecuencia pero no brinda información acerca de la evolución en el tiempo de las características espectrales de la señal, como por ejemplo el comienzo, el fin de una señal y menos el instante de alguna aparición de alguna singularidad en el tiempo, por ejemplo para señales “explosivas”.

5.2 Transformada Discreta de Fourier (DFT)

Debido a que los computadores trabajan sólo con datos discretos, y la necesidad de hacer un cálculo más rápido de la transformada de Fourier usando el computador entonces surge la necesidad de la discretización.

5.2.1 Definición

$$X(w_k) = \sum_{n=0}^{N-1} x(t_n) e^{-i\omega_k t_n}, k = 0, 1, 2, \dots, N-1,$$

$x(t_n)$ = amplitud de la señal de entrada en t_n (Seg)

$t_n = nT$, instante de muestreo(seg), n es un entero ≥ 0

T = intervalo de muestreo(seg)

$X(w_k)$ = espectro de $x(t_n)$ (valor complejo) en la frecuencia w_k

$w_k = K\Omega$ = muestra de frecuencia(rad/seg)

$\Omega = \frac{2\pi}{NT}$ = intervalo de muestreo de radian – frecuencia ($\frac{\text{rad}}{\text{seg}}$)

$f_s = \frac{1}{T}$ = frecuencia de muestreo(Hertz)

N = número de muestras de tiempo(valor entero)

De la ecuación 4 de la transformada de Fourier de una señal, que por comodidad se muestra nuevamente:

$$X(f) = \int_{-\infty}^{+\infty} x(t) e^{-i2\pi f t} dt$$

Si se toma N muestras de $x(t)$, y estas muestras pueden ser definidas como:

$x[0], x[1], \dots x[n], \dots x[N-1]$.

Se puede pensar cada muestra $x[n]$ como un impulso que tiene un área de $x[n]$:

$$X(f) = \int_0^{(N-1)T} x(t) e^{-i2\pi f t} dt = x[0]e^{-i0} + x[1]e^{-i2\pi f T} + \dots + x[n]e^{-i2\pi f nT} + \dots \\ \dots + x[N-1]e^{-i2\pi f (N-1)T}$$

Así $X(f)$ se convierte en:

$$X(f) = \sum_{n=0}^{N-1} x[n] e^{-i2\pi f nT}$$

Puesto que hay sólo un número finito de datos de entrada, la DFT trata los datos como si se tratara de período, y evalúa la ecuación para la frecuencia fundamental (ω):

$$\omega = 0, \frac{2\pi}{NT}, \frac{2\pi}{NT} \times 2, \dots, \frac{2\pi}{NT} \times n, \dots, \frac{2\pi}{NT} \times (N-1)$$

Por lo tanto, la transformada de Fourier discreta de la secuencia $x[n]$ se puede definir como:

$$X[k] = \sum_{n=0}^{N-1} x(n)e^{-i2\pi kn/N}, \quad k = 0, 1, 2, \dots, N-1 \quad (5)$$

Donde si $W = e^{-i2\pi k/N}$ y $W = W^{2N} = 1$.

La ecuación se puede escribir en forma de matriz:

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ \vdots \\ X[N-1] \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & W & W^2 & W^3 & \dots & W^{N-1} \\ 1 & W^2 & W^4 & W^6 & \dots & W^{2(N-1)} \\ 1 & W^3 & W^6 & W^9 & \dots & W^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W^{N-1} & W^{2(N-1)} & W^{3(N-1)} & \dots & W^{(N-1)(N-1)} \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ \vdots \\ x[N-1] \end{bmatrix}$$

A manera de ejemplo vamos a suponer que tenemos una señal muestreada en cuatro puntos:

$$x[n] = [2 \quad 3 \quad -1 \quad 4], \quad N = 4, \quad (n = 0, 1, 2, 3)$$

De la ecuación 5 se tiene: $X[k] = \sum_{n=0}^3 x(n)e^{-\frac{i2\pi kn}{4}} = \sum_{n=0}^3 x(n)e^{-\frac{i\pi kn}{2}}$

$$X[k] = \sum_{n=0}^3 x(n)(-i)^{nk}$$

$$X[0] = 2 + 3 + (-1) + 4 = 8 \Rightarrow \|X[0]\| = 8$$

$$X[1] = 2 + (-3i) + 1 + 4i = 3 + i \Rightarrow \|X[1]\| = 3.16$$

$$X[2] = 2 + (-3) + (-1) - 4 = -6 \Rightarrow \|X[2]\| = 6$$

$$X[3] = 2 + (3i) + 1 - 4i = 3 - i \Rightarrow \|X[3]\| = 3.16$$

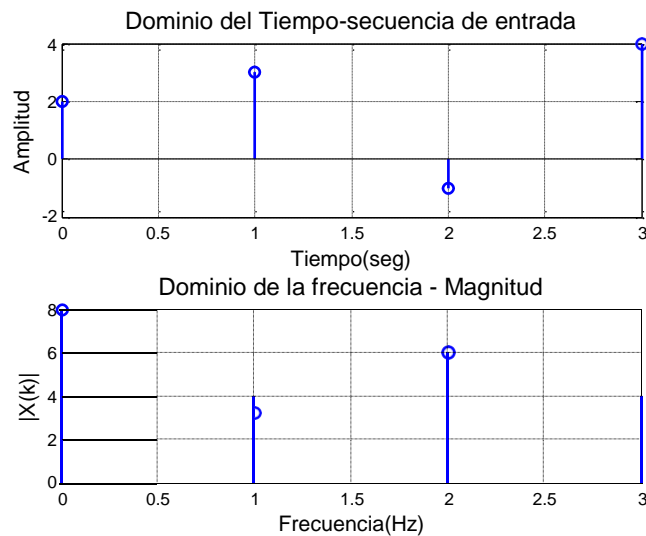


Figura 33 -Transformada Discreta de Fourier

Fuente: Toobox4-Matlab

En la figura 34 se muestra la DFT de una muestra en el tiempo de $x[n]$. Se puede concluir y observar que si se obtienen N muestras de una señal en la banda de tiempo de un sistema de adquisición de datos y se aplica la DFT, el resultado también será N muestras pero la información que contiene está en la banda de frecuencias.

5.3 Transformada de Fourier de tiempo corto (Gabor-STFT)

Este es básicamente el mismo que FT pero utiliza una ventana. En lugar de realizar la transformada de Fourier a toda la señal sólo lo hace dentro de una ventana. La ventana tiene un ancho de banda predefinido y comienza al principio de la señal. La Figura 34 muestra los resultados del STFT de los dos ejemplos utilizados anteriormente en esta sección.

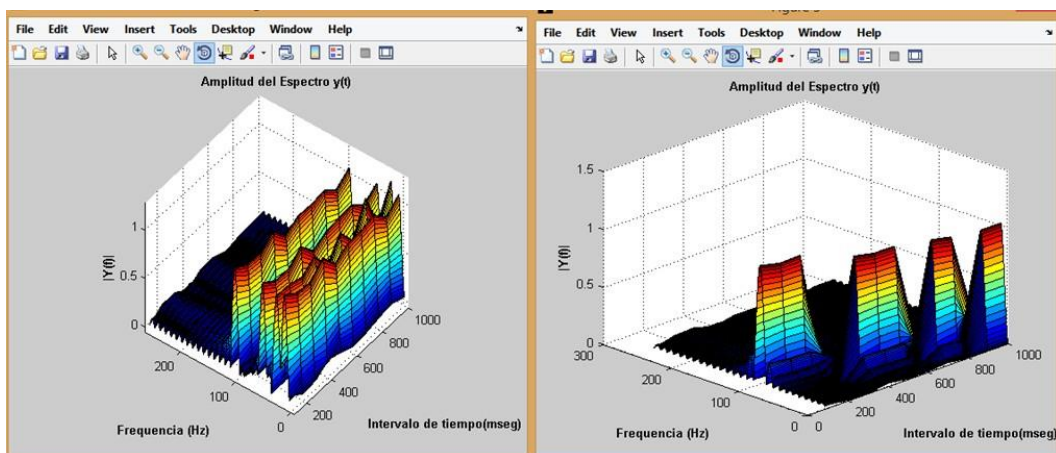


Figura 34-Resultados de STFT de una señal estacionaria (izquierda) y la señal no estacionaria (derecha)

Fuente: Toobox4-Matlab

Ahora podemos distinguir claramente las señales. Sin embargo, existe un problema significativo con STFT, el problema es con el ancho de la ventana. Una gran ventana nos da pobre tiempo de resolución (cuanto más ancha sea la ventana de la STFT más se parece a un FT normal). Por otro lado, si la ventana es angosta tiene menos resolución para frecuencias. En el caso de una pequeña ventana de los picos de la FT se ensanchan por lo que se vuelve menos claro saber la frecuencia real.

5.3.1 Definición

Con el fin de superar las limitaciones de la transformada de Fourier, una especie de tiempo localizada transformada de Fourier se puede introducir. Este concepto, que ha sido propuesto inicialmente por Gabor en [46], se conoce como *Short Time Fourier Transform* (STFT). LA STFT es capaz de recuperar tanto la información de tiempo y frecuencia de una señal. Técnicamente, STFT emplea una función de ventana deslizante $g(t)$ que se centra en τ tiempo; y puede expresarse por la siguiente ecuación:

$$X(t, f) = \int_{-\infty}^{+\infty} f(t) g^*(t - \tau) e^{-i2\pi f t} dt$$

Donde $g(t)$ es una ventana deslizante, la cual tiene un ancho fijo y cambia a lo largo del eje x por un factor τ . Esta función ventana $g(t)$ es una ventana Gaussiana esta expresada en la ecuación 6.

$$\omega(t, \tau, a) = \exp\left(-a \frac{(t - \tau)^2}{2}\right) \quad (6)$$

Donde t es el desplazamiento temporal y τ el coeficiente de expansión. Una gráfica representativa se puede observar en la figura 35, cuando $\tau = 500$ y $a = 100$.

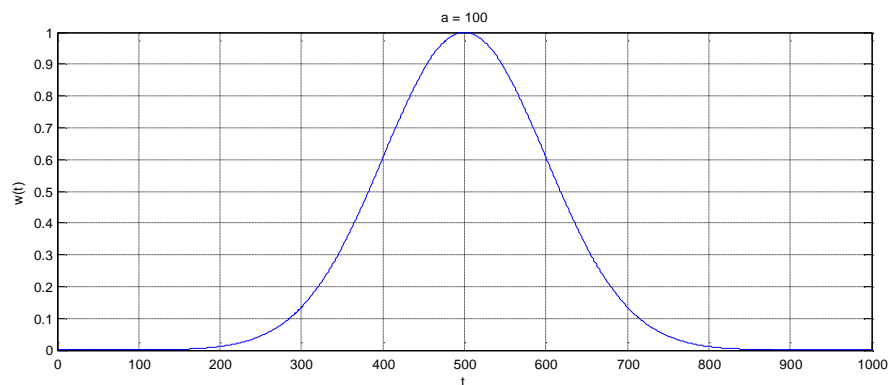


Figura 35-Ventana Gaussiana

Fuente: Toolbox de Matlab

Donde $*$ denota el conjugado complejo del operador. Cuando la ventana se mueve por τ , en un tiempo y espacio la transformada de Fourier se realiza sobre la señal original $f(t)$ dentro de la ventana. De esta manera, la STFT descompone una señal de dominio de tiempo en una representación de dos dimensiones de tiempo-frecuencia. Tal representación tiempo-frecuencia proporciona alguna información sobre si alguna frecuencia está presente continuamente sobre toda la señal o sólo en un intervalo de tiempo específico (sólo una parte de la señal). Sin embargo, usando una ventana deslizante con los resultados de

longitud fija en un nuevo problema. En realidad, el análisis STFT es críticamente dependiente de la ventana elegida $g(t)$. Teóricamente, no es posible saber exactamente que frecuencias ocurren en un momento específico, sólo algunas frecuencias (un intervalo de frecuencias) puede ser detectada. En otras palabras, no es posible conseguir tanto una buena resolución en el tiempo y una buena resolución de frecuencia con STFT. De hecho, por un lado, una ventana corta, que es conveniente para alta detección de frecuencias, proporciona una buena resolución en el tiempo, pero varias frecuencias no se revelan. Por otro lado, una ventana larga, que es conveniente para la detección de baja frecuencias, proporciona una resolución de tiempo inferior, pero una mejor resolución de frecuencia.

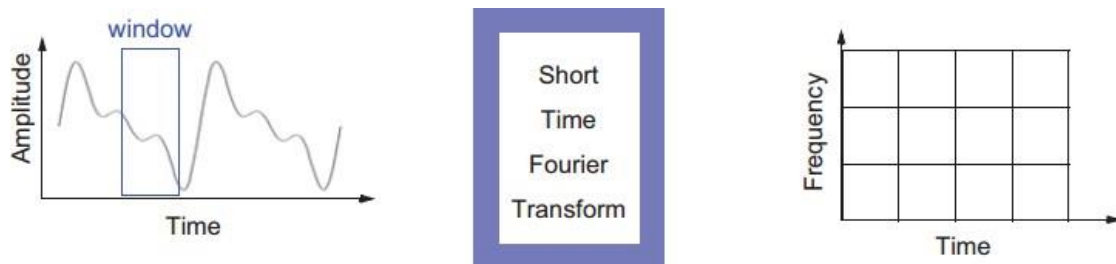


Figura 36- Transformada de Fourier por ventanas

Fuente: Toobox4-Matlab

En la figura 36, se puede observar de manera representativa el resultado de *Short Time Fourier Transform* (STFT), se aprecia que la representación es tiempo-frecuencia es a través de ventanas de la misma proporción de distribución.

Capítulo 6

Transformada *Wavelet* basado en señales unidimensionales

Se pueden definir las *Wavelet* como familias de funciones que se encuentran definidas en el espacio y se emplean como funciones de análisis, examinando la señal de interés en el plano tiempo-frecuencia para obtener sus características periódicas o no periódicas [6].

Análisis *Wavelet* utiliza un método más adaptable a señales estacionarias de altas frecuencias, y es capaz de revelar las tendencias, discontinuidades en derivadas de orden superior.

Una *Wavelet* ψ es una forma de onda de la duración efectiva limitado que tiene un valor medio de cero:

$$\int_{-\infty}^{+\infty} \psi(t) dt = 0. \quad (7)$$

Que se pueden aplicar dos transformaciones básicas:

1. **Cambio de escala:** por la que es comprimida o dilatada mediante el parámetro escala “a”.
2. **Traslación:** con la que la señal o *Wavelet* es trasladada por el parámetro traslación “b” a lo largo del eje de tiempos.

Así, cada familia *Wavelet* viene definida por $\psi_{a,b}$ mediante la siguiente expresión:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}} \psi\left(\frac{t-b}{a}\right) \quad a > 0. \quad (8)$$

Los componentes de una familia se generan a partir de la *Wavelet* madre o *Wavelet* de análisis $\psi(t)$, [34], por medio de la señal, y de la variable b, parametro de traslación, que permite mover la señal en el dominio del tiempo.

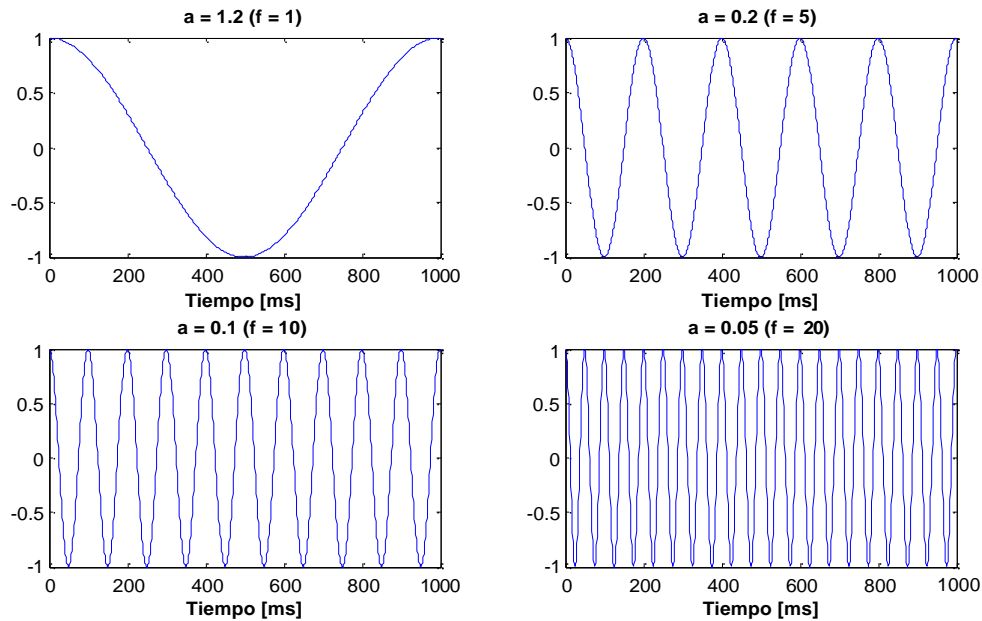
Se especifica en la las transformaciones de traslacion y cambio de escala.

Tabla 1 -Transformaciones de traslación y cambio de escala en una Wavelet

Traslación	Cambio de Escala	Traslación y Cambio de Escala
$\psi(t - b)$	$\frac{1}{\sqrt{a}} \psi\left(\frac{t}{a}\right)$	$\frac{1}{\sqrt{a}} \psi\left(\frac{t - b}{a}\right)$

Fuente: internet [1]

Con estas transformaciones se generan todas las funciones $\psi_{a,b}(t)$ a partir de las funciones madre o *Wavelet* madre $\psi(t)$.

**Figura 37-Ejemplo de una señal coseno para distintas escalas**

Fuente: propia

La *Wavelet* $\psi(t)$ con valores de escala $a > 1$ se dilata, mientras que para valores de $a < 1$ se comprime la señal del eje de tiempos, aumentando su frecuencia a medida que la escala “a” disminuye y viceversa.

Por lo anterior entonces se puede concluir que existe una relación inversa entre la escala a de la señal *Wavelet* y su frecuencia. En la figura 37 se puede ver la relación de la señal *Wavelet* y la frecuencia.

En cambio, el factor de traslación permite adelantar la señal ($b < 0$) o retrasarla con ($b > 0$), respecto al origen de tiempos.

6.1 Transformada Wavelet Continua (CWT)

6.1.1 Introducción

La *Wavelet* continua (CWT) es la suma sobre todos los tiempos de versiones desplazados y escalados de la ψ (función *Wavelet*). El CWT, cuando se aplica sobre la señal original $f(t)$, se expresa como:

$$CWT f(\tau, s) = \int_{-\infty}^{+\infty} f(t) \psi_{\tau,s}^*(t) dt. \quad (9)$$

$$\psi_{\tau,s}(t) = \frac{1}{\sqrt{|s|}} \psi\left(\frac{t - \tau}{s}\right). \quad (10)$$

El cálculo de la CWT a través de una señal nos da resultados en muchos coeficientes, que son funciones del parámetro de traslación τ y el parámetro de escala s . De hecho, τ es proporcional a la información de tiempo. Se especifica la ubicación de la *Wavelet* en el tiempo; τ por la variación de la *Wavelet* puede ser desplazado sobre la señal. La escala s es inversamente proporcional a la frecuencia de la información. La variación de s modifica no sólo la frecuencia central de la *Wavelet*, sino también la longitud de la ventana. Grandes escalas están relacionadas con las frecuencias bajas, dando la información global de la señal. Considerando que, escalas pequeñas corresponden a altas frecuencias, revelando los detalles más finos de la señal. Por otra parte, la energía de la señal se mantiene constante en cada escala, ya que se normaliza por $\frac{1}{\sqrt{|s|}}$. A pesar de la alta precisión

proporcionada por el CWT en el análisis de una señal, el cálculo de CWT es redundante y hace mucho consumo de tiempo al procesarlo. Por lo general, el cálculo de la CWT se lleva a cabo mediante la adopción de valores discretos para el parámetro de escala s y el parámetro τ de traslación.



Figura 38 - Transformada Wavelet

Fuente: Toobox4-Matlab

En la figura 38, se puede observar de manera representativa el resultado de *Wavelet Transform* (WT), se aprecia que la representación es tiempo-frecuencia es a través de ventanas de diferentes proporciones de distribución es aquí donde se diferencia de otros algoritmos.

6.1.2 Familias *Wavelet* o *Wavelet* madre

La función $\psi_{\tau,s}(t)$ se le llama *Wavelet* madre por las siguientes dos razones:

- El término *Wavelet* significa “onda pequeña”. La pequeñez se refiere al hecho de que es de longitud finita, además la función es de naturaleza oscilatoria.
- El término madre es por el hecho de que esta es un prototipo para generar otras funciones ventanas en otras regiones distintas.

La elección de la *Wavelet* es dependiendo de las características de la señal y de la naturaleza de la aplicación. Saber las propiedades del análisis *Wavelet* y síntesis se puede elegir una *Wavelet* que sea óptimo para la aplicación [49].

Las distintas familias *Wavelet* se pueden distinguir según algunas características importantes como:

- Soporte de las *Wavelet* en el tiempo y en la frecuencia y proporción de decaimiento.
- Simetría o anti-simetría de la *Wavelet*. La reconstrucción de la señal es los filtros es de fase lineal.
- Número de momentos de desvanecimiento. *Wavelet* con un incremento número de desvanecimiento resultan en diversas representaciones de la señal.
- La regularidad de la señal. ondas más suaves proporcionan mejor la resolución de frecuencia, además, los algoritmos iterativos para la construcción *Wavelet* convergen más rápidamente.

Para saber más de las propiedades de las familias *Wavelet* se puede escribir los nombres abreviados en Matlab, los nombres abreviados se muestran en la Figura 39. Con ayuda del comando ***waveinfo*** seguido del nombre *Wavelet* abreviado.

Según lo dicho anteriormente se ha hecho un estudio donde se concluye que se prefiere el uso de determinadas familias *Wavelet* para hacer uso de las mismas, para análisis discreto se encuentran las *Wavelet* ortogonales por ejemplo (*Daubechies* que se caracteriza por tener una buena fase lineal pero es asimétrica) y las *Wavelets biortogonal* y *Wavelet B-spline*. [6]. En la figura 40 se presentan algunas *Wavelets* madres más usadas en la literatura.

Para un análisis continuo *Wavelet* se encuentra mejores resultados haciendo el uso de familias *Morlet*, *Meyer*, ó *gaussiano*.

Wavelet Family Short Name	Wavelet Family Name
'haar'	Haar wavelet
'db'	Daubechies wavelets
'sym'	Symlets
'coif'	Coiflets
'bior'	Biorthogonal wavelets
'rbio'	Reverse biorthogonal wavelets
'meyr'	Meyer wavelet
'dmey'	Discrete approximation of Meyer wavelet
'gaus'	Gaussian wavelets
'mexh'	Mexican hat wavelet
'morl'	Morlet wavelet
'cgau'	Complex Gaussian wavelets
'shan'	Shannon wavelets
'fbsp'	Frequency B-Spline wavelets
'cmor'	Complex Morlet wavelets

Figura 39-Familias *Wavelet* (abreviaturas)-Matlab

Fuente: Toobox4-Matlab

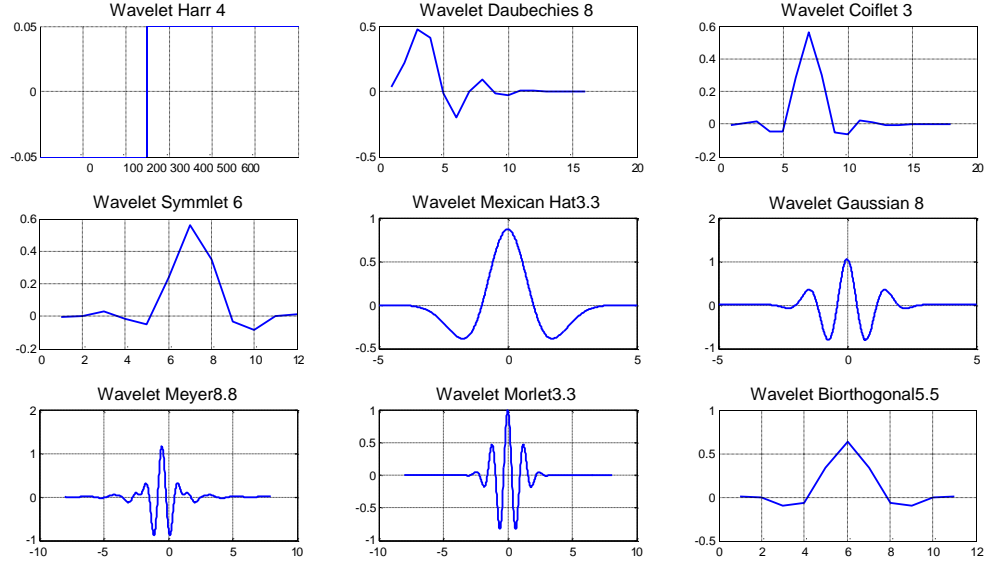


Figura 40-Wavelets madres más usadas
Fuente: propia

Todas las funciones están explicadas en [36], dependiendo cuál es su finalidad de dicho análisis, algunas funciones madres pueden servir mejor que otras.

6.2 Transformada Wavelet Discreta (DWT)

6.2.1 Introducción

Se trata de reconstruir una señal original usando una suma infinita de coeficientes *Wavelet* discretos en contraposición de la integral continua que requiere CWT. Esta suma conduce a una Transformada *Wavelet* Discreta (DWT).

De forma similar a la transformada de Fourier Discreta y a la Transformada de Fourier Discreta de tiempo reducido(Gabor), se tiene la Transformada *Wavelet* Discreta (DWT) [37]. A diferencia del tiempo y frecuencia discretos en los análisis de Fourier, se tienen valores discretos del parámetro escala a y del parámetro traslación b , de forma diferente. Se toma la escala a de la forma 2^{-j} y la traslación b con la estructura $k2^{-j}$, donde $j, k \in \mathbb{Z}$. Con estos valores la integral de la ecuación 9 se convierte en la siguiente:

$$W_{\psi}x(b, a) = W_{\psi}x(k2^{-j}, 2^{-j}) = 2^{\frac{j}{2}} \int_{-\infty}^{+\infty} x(t) \psi(2^j t - k) dt. \quad (11)$$

Ahora se discretiza la función $x(t)$, que por simplicidad se toma una frecuencia de muestreo de 1. La ecuación anterior se convierte en la siguiente:

$$W_{\psi}x(k2^{-j}, 2^{-j})K \approx 2^{\frac{j}{2}} \sum_n x(n) \psi(2^j n - k) \quad (12)$$

El algoritmo de la transformada *Wavelet* Discreta (DWT) es como sigue:

Dada una señal S de longitud N , la DWT consiste de $\log_2 N$ etapas a lo mucho. La primera etapa produce, partiendo de S , dos conjuntos de coeficientes: coeficientes de aproximación CA_1 , y coeficientes de detalle CD_1 . Estos vectores son obtenidos mediante la convolución de S con el filtro pasa bajas Lo_D para la aproximación,

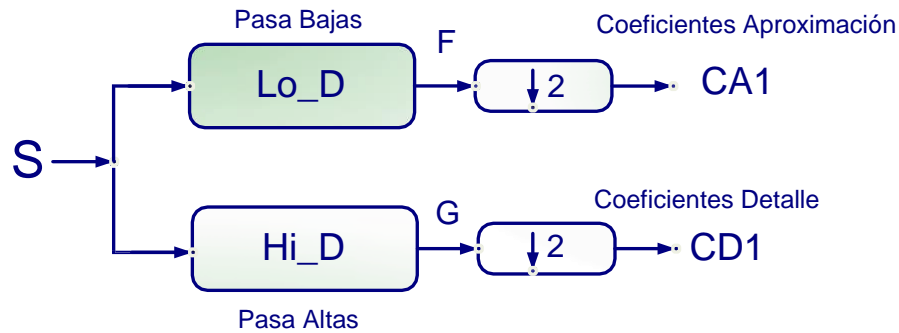


Figura 41- Primera etapa de la DWT

Fuente: propia

Y con el filtro pasa altas Hi_D para los detalles, seguido por una decimación diática (*downsampling*). La longitud de cada filtro es igual a $2N$. Si n es igual a la longitud de S , las señales F y G son de longitud $n + 2N - 1$.

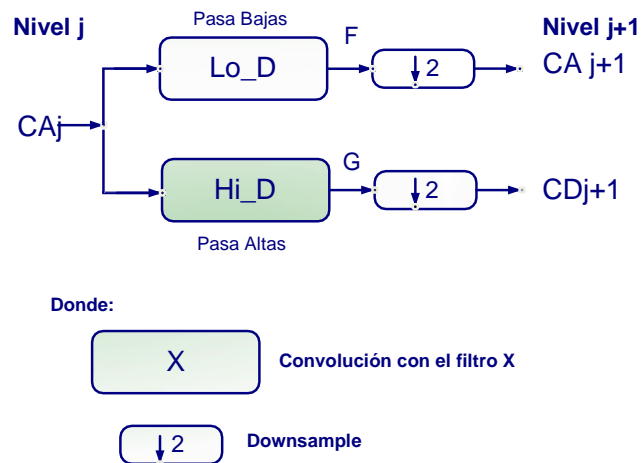


Figura 42 - DWT unidimensional. Etapa generalizada $j, j+1$

Fuente: propia

El próximo paso particiona los coeficientes de aproximación CA_1 en dos partes usando el mismo esquema, reemplazando S por CA_1 , produciendo CA_2 y CD_2 , y así sucesivamente. La descomposición *Wavelet* de la señal S en el nivel j tiene la estructura: $[CA_j, CD_j, \dots, CD_1]$.

6.3 Análisis Multirresolución (MRA)

En el análisis Multirresolución, o algoritmo piramidal, la idea es la misma que en la CWT: obtener una representación tiempo-escala de una señal discreta, pero reduciendo el tiempo de procesamiento de forma significativa. En este caso, filtros de media banda⁵ son usados para analizar la señal en diferentes escalas. La señal se pasa a través de filtros paso alto para analizar las componentes de alta frecuencia, y de filtros paso bajo para analizar las componentes de baja frecuencia. Estas operaciones cambian la resolución de la señal, la escala se cambia mediante operaciones de interpolación y submuestreo. La forma más compacta de describir este proceso así como de representar los procesos para determinar los coeficientes *Wavelet*, es la representación de los filtros en forma de operador. Para una secuencia $f[n]$ que representa la señal discreta a ser descompuesta, los operadores H y G se definen según las expresiones siguientes:

$$(Hf)_k = \sum_n h[n - 2k]f[n] \quad (13)$$

$$(Gf)_k = \sum_n g[n - 2k]f[n] \quad (14)$$

Las ecuaciones (13-14) representan el filtrado de la señal donde [36]:

- $h[n]$, $g[n]$ son filtros digitales.
- El factor $2k$ representa el sub- muestreo.
- Los operadores H y G corresponden a un paso en la descomposición *Wavelet*

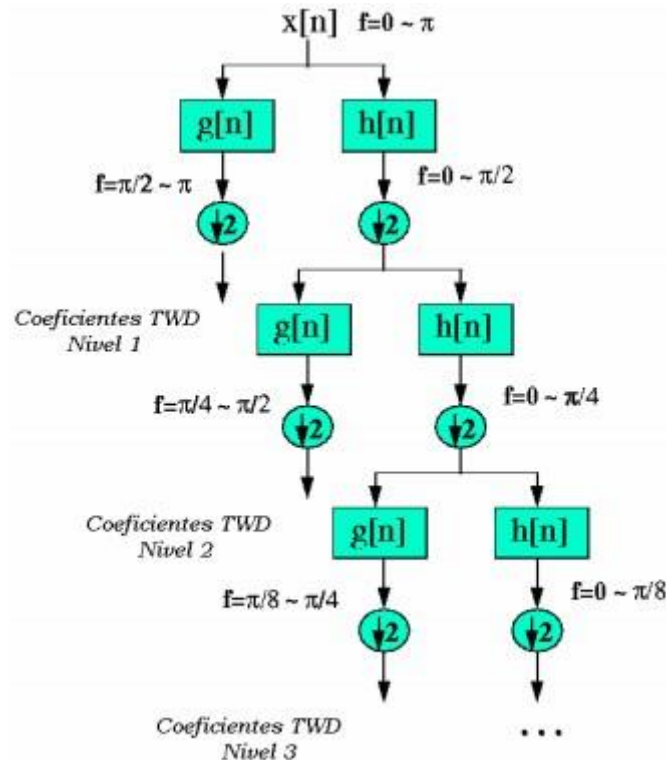


Figura 43 - Esquema de MRA que genera la descomposición *Wavelet*

Fuente: [36]

⁵ <http://www4.tecnun.es/asignaturas/tratamiento%20digital/tema9.pdf>

En términos generales, la DWT realiza una descomposición recursiva de la banda de frecuencia más baja (Aproximaciones) obtenida a partir de una descomposición anterior. Como resultado, un conjunto jerárquico de aproximaciones y detalles se pueden obtener a través de los distintos niveles de descomposición. Este procedimiento, conocido como el análisis de resolución múltiple (MRA), se introdujo por Mallat y puede llevarse a cabo usando un algoritmo eficiente computacionalmente [53]. La figura 44 muestra un ejemplo de una descomposición *Wavelet* de tres niveles de un $x_n = f = S$ señal. En primer lugar, la x_n señal se divide en una aproximación A_1 y un detalle D_1 . A continuación, la aproximación A_1 también se divide en una segunda aproximación nivel A_2 y D_2 , la aproximación A_2 también se divide en una tercera aproximación nivel A_3 y D_3 .

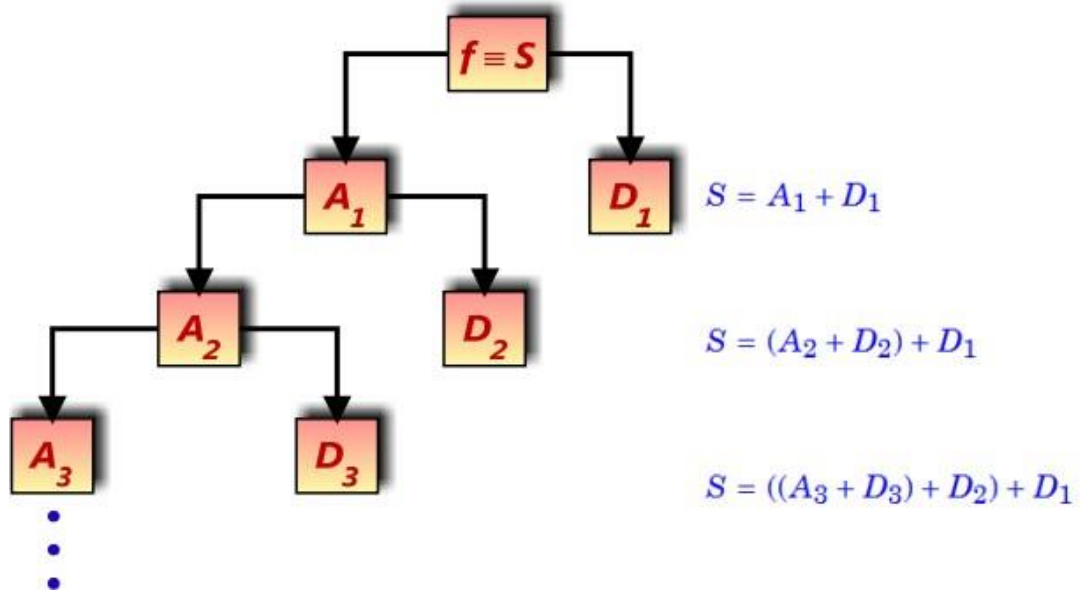


Figura 44- Descomposición de una señal en múltiples niveles (MRA)

Fuente: internet [54]

Según lo anterior si se tiene una señal $x_n(n=1,2,\dots,N)$, N es la cantidad de muestras, la transformada *Wavelet* Discreta(DWT) de un intervalo de tiempo de x_n se puede expresar en funciones base $\psi_{j,k}$, $\phi_{j,k}$:

$$\begin{aligned}
 X(t) = & \sum_K s_{J,K} \phi_{J,K}(t) + \sum_K d_{J,K} \psi_{J,K}(t) + \sum_K d_{J-1,K} \psi_{J-1,K}(t) + \dots \\
 & + \sum_K d_{1,K} \phi_{1,K}(t)
 \end{aligned} \tag{15}$$

Donde $s_{J,K}$, $d_{J,K}$, \dots , $d_{1,K}$ son los coeficientes *Wavelet*; J es un número natural dependiente principalmente en N y la función base; y K varía desde 1 al número de coeficientes en el componente especificado.

La base para la descomposición anterior se forma a partir de la función *Wavelet* madre $\psi(t)$ y la función *Wavelet* padre $\phi(t)$ mediante la traslación en el tiempo y la dilatación en escala:

$$\psi_{j,k}(t) = 2^{-j/2} \psi(2^{-j}t - k) \quad j, k \in \mathbb{Z} \quad (16)$$

$$\phi_{j,k}(t) = 2^{-j/2} \phi(2^{-j}t - K) \quad a, b \in \mathbb{Z} \quad (17)$$

Donde $k = 1, 2, \dots, N/2$, en la que N es el número de muestras de datos; $j = 1, 2, \dots, J$, en la que J es un pequeño número natural; \mathbb{Z} es el conjunto de los enteros.

6.4 Eliminación de ruido usando *Wavelet*

La introducción del estudio de la eliminación de ruido usando algoritmo *Wavelet* se empezó con la investigación de D.L. Donoho en su paper *Translation-Invariant De-Noising* [40] desde aquí hacia adelante se empezó una serie de investigaciones [41, 42, 43, 44]. En la toma de datos, uno de los principales factores que perturban la fiabilidad y exactitud de los resultados son las señales de ruido que se encuentran. Para superar esta deficiencia, esta sección presenta un enfoque de la transformada *Wavelet Discreta* (DWT) para eliminar el ruido de las señales medidas.

El ruido incrustado en señales, a veces incluso con una relación SNR muy bajo, perturba la fiabilidad y la precisión de las mediciones, y por lo tanto establece un límite fundamental en la detección de pequeños defectos. Por lo tanto, la reducción de ruido se debe tratar como una característica esencial en el procesamiento de las pruebas de dinámica estructural.

Hasta ahora, un gran número de algoritmos de eliminación de ruido de señal se han propuesto en la literatura, tales como el filtro de paso bajo lineal [45], el filtro de Kalman [46], el filtro de mediana [47] y filtrado adaptativo basado en redes neuronales [48]. Sin embargo, estos métodos convencionales tienen defectos inherentes; por ejemplo, un filtro de paso bajo lineal no es bueno en un caso cuya señal se superpone el ruido a menudo en muchas bandas de frecuencia [41].

Al combinar el dominio del tiempo y el análisis de dominio de la frecuencia clásica, la transformada *Wavelet* (WT) se convierte en una herramienta potencial para el análisis de datos. El creciente interés en este método se debe a varios factores:

- Sencillez del enfoque.
- Reducción de la complejidad computacional asociado con el algoritmo,
- La capacidad de proporcionar la representación simultáneamente espectral y el orden temporal de los componentes de descomposición de la señal.

El rendimiento de eliminación de ruido es discutido por varios parámetros de procesamiento, incluyendo el tipo de *Wavelet* madre, nivel de descomposición, el método de umbralización, y las reglas de selección de umbral. Para superar los inconvenientes de los métodos tradicionales de *hard-thresholding* y *soft-thresholding*, una técnica de umbralización mejorada llama se presenta el esquema de umbral basado en la función *sigmoide*.

6.4.1 Procedimiento general de filtrado mediante *Wavelet*

Si la señal medida es t_i y se puede expresar como la suma de dos componentes:

$$y(t_i) = x(t_i) + e_i \quad (18)$$

Donde $x(t_i)$ es la señal pura, e_i es el ruido, $y(t_i)$ es la señal dañada por el ruido, en el que, mucha información valiosa para su estudio. Eliminar el ruido por completo es imposible. El objetivo de la eliminación de ruido es la obtención de una señal $y(t_i)$ tan cerca como sea posible a $x(t_i)$, minimizando así el efecto de e_i .

En general, el procedimiento de eliminación de ruido basado en *Wavelet* se puede describir en tres pasos como sigue [49]:

Paso 1.-Descomposición: Elegimos una madre *Wavelet*, seleccionamos el nivel. Calculamos la descomposición *Wavelet* de la señal al nivel N.

Paso 2.- Umbralización de coeficientes detalle: Para cada nivel de 1 a N, seleccionar un umbral y aplicar de umbral a los coeficientes de detalle.

Paso 3.- Reconstrucción de la señal: Calcular la reconstrucción de la señal usando la transformada *Wavelet* inversa(IWT) basado en la aproximación inicial coeficientes de nivel N y los coeficientes de detalle modificadas de los niveles de 1 a N.

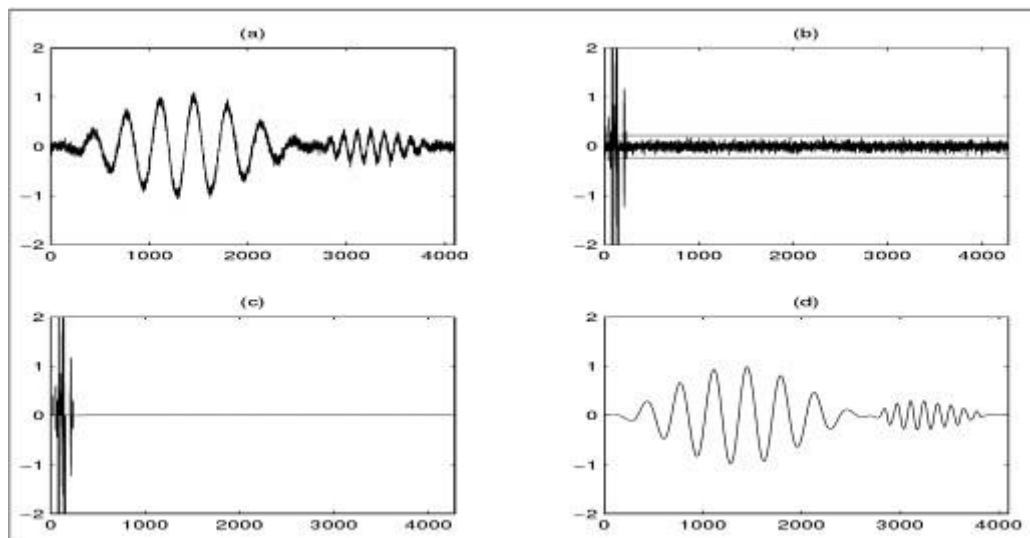


Figura 45 - Procedimiento de filtrado *Wavelet*

Fuente: [63]

En la figura 45 se tiene los resultados de los 3 pasos para lograr un filtrado de una señal(a), Aquí se usó la transformada db de N=10 niveles(b), luego en (c) se utilizó un umbral de ± 0.227 y se aplicó a los coeficientes detalles y el (d) se reconstruye la señal mediante la Transformada *Wavelet* Inversa.

6.4.2 Técnicas de cancelación de ruido

6.4.2.1 Selección de madre *Wavelet*

La elección de la madre *Wavelet* para el procedimiento de eliminación de ruido antes mencionado es teóricamente arbitraria, pero es crítica e importante en la práctica, ya que afecta el rendimiento de la técnica. Si se elige una *Wavelet* correcta puede dar lugar a un análisis ortogonal, permitiendo un algoritmo rápido, existiendo la posibilidad de reconstrucción perfecta, buena localización en el tiempo y frecuencia. De hecho, los coeficientes de la transformada *Wavelet* representan lo bien que la señal coincide con las réplicas a escala con la *Wavelet* madre.

En este trabajo nos restringiremos al uso de la *Wavelet* madre llamado *Daubechies* por prueba y error ya que para este tipo de señal no se tiene mucha información o recomendación que *Wavelet* madre utilizar. En todos los experimentos nos restringiremos al uso de *Daubechies* 4(DB4) para la preservación de los detalles finos de la señal.

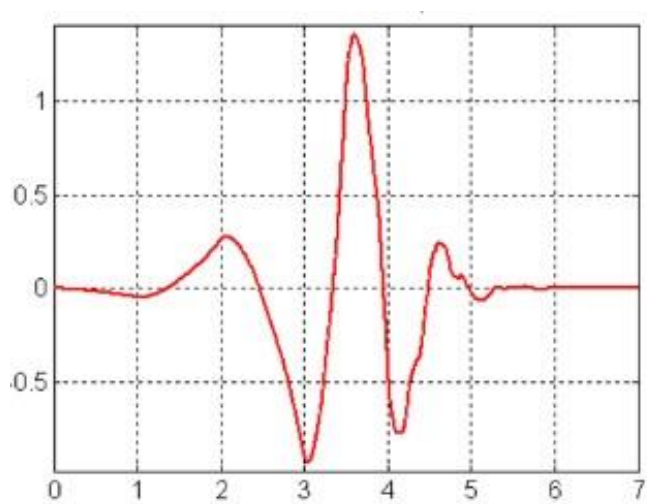


Figura 46 -Wavelet madre utilizada: *Daubechies* 4

Fuente: Propia

En la figura se muestra la *Wavelet Daubechies* de nivel 4(DB4), con 4 picos que puede proporcionar una mejor localización y frecuencia de aproximación y por ello dar lugar a un mejor rendimiento.

6.4.2.2 Determinación del nivel de descomposición

El segundo factor que influye mucho en la eliminación de ruido de la señal es el nivel de descomposición DWT. Si el nivel de descomposición no es suficiente, la mejora de la relación señal a ruido (SNR) será limitada. Por otra parte, si el nivel de descomposición es demasiado, la cantidad de cálculos aumentará en gran medida, y la reducción de ruido puede ser no satisfactoria también. Todavía no se ha propuesto ningún procedimiento ideal para elegir el mejor nivel de descomposición DWT, aquí, una especie de "prueba de ruido blanco" está probada [50].

Como se dijo antes, en el DWT una señal de medición se divide en dos sub-señales, un detalle y un aproximado, entonces el sub-señal de detalle a su vez se divide en dos otros

sub-señales, respectivamente, formando un árbol binario. Dado que la amplitud y el tiempo de llegada de cada pico de ruido es aleatorio, esto hará que una perturbación que afecta a la señal o las máscaras de la señal objetivo completamente (es decir, el ruido en la señal experimental se distribuye uniformemente sobre todos los coeficientes de detalle de ondas con amplitudes pequeñas, que generalmente obedece a la distribución de ruido blanco). De la teoría WT, el ruido blanco sigue siendo el ruido blanco a través de la WT ortogonal. Pero, a diferencia con el ruido, la señal deseada, con cierto tiempo y localización de frecuencia, se concentra en sólo unos coeficientes, que presentan las características de ruido no blancas.

El procedimiento para la elección del nivel racional de descomposición DWT sobre la base de "prueba de ruido blanco" se resume como sigue:

- Paso (1) La descomposición de la señal con ruido de entrada en $i = 1$ nivel de la aproximación A_1 y coeficientes detalle D_1 .
- Paso (2) Mantener el coeficiente de aproximación A_1 obtenida en la etapa (1) y realice la prueba de ruido blanco en el coeficiente detallada D_1 . Siga tomando la descomposición de A_1 si D_1 pasa la prueba de ruido blanco;
- Paso (3) Repita los pasos anteriores, es decir, para llevar a cabo la prueba de ruido blanco en el coeficiente correspondiente detallada después de la descomposición de cada capa hasta que el coeficiente detalle no puede pasar la prueba de ruido blanco;
- Paso (4) Abandonar el último resultado de la descomposición en la que el coeficiente detalla de no ha pasado la prueba de ruido blanco, es decir, el nivel de descomposición debe ser $n - 1$ si se descompone n veces.

6.4.3 Segmentación o Umbral

El propósito del procedimiento de umbral es eliminar o suprimir los pequeños coeficientes de valor de ondas que representan principalmente el contenido de ruido. Este proceso se puede llevar a cabo siguiendo dos reglas básicas:

Umbralización *hard* o duro ("mantener o matar") y umbralización *soft* o suave (reducir o matar) introducido por Donoho y Johnstone [51]. En umbralización duro, coeficientes con valores absolutos más bajos que el umbral se ponen a cero, mientras que la umbralización suave además reduce los coeficientes distintos de cero restantes hacia cero.

A continuación se describen brevemente, para conocer con más detalle podemos ver [52, 49].

❖ Umbral *Hard*

En este tipo de umbral, si el valor de una señal está por debajo de un valor (umbral) preestablecido previamente, esta señal se pone a cero.

$$\tilde{w}_{k,j} = \begin{cases} w_{k,j}, & |w_{k,j}| \geq T \\ 0, & \text{otra manera} \end{cases} \quad (19)$$

Donde $\tilde{w}_{k,j}$ y $w_{k,j}$ representan los coeficientes DWT reducidos y originales respectivamente y T representa el umbral.

❖ Umbral *soft*

Es una extensión de umbral *hard*

Se define como:

$$\tilde{w}_{k,j} = \begin{cases} \text{sgn}(w_{k,j})f(|w_{k,j}| - T), & |w_{k,j}| \geq T \\ 0, & \text{otra manera} \end{cases} \quad (20)$$

Se basa en el establecimiento de los elementos primero con los valores absolutos inferior al umbral a cero, y luego la reducción de los otros coeficientes.

Donde $\text{sgn}(\cdot)$ es la función signo, que devuelve 1 si el elemento es mayor que 0, 0 si es igual a cero y -1 si menos de 0. La señal recuperada se puede obtener de $\tilde{w}_{k,j}$ por el inverso WT (IWT).

Los inconvenientes de la umbralización *hard* y *soft* son que el umbral *hard* no es continua en el umbral mientras que el umbral *soft* no es diferenciable en este valor; un pre-requisito para cualquier problema de optimización.

6.4.4 Selección del umbral

Donoho y Johnstone desarrollaron una regla de umbral universal [49], que puede eliminar eficazmente el ruido aleatorio gaussiano. La regla de umbral universal es como sigue:

$$T = \sigma\sqrt{2\ln N} \quad (21)$$

Donde N es la longitud de los coeficientes, σ es la desviación estándar del ruido y se define como:

$$\sigma = \frac{|\text{median}(w_{1,j})|}{0.6745} \quad (22)$$

Donde MAD representa la desviación absoluta media de los coeficientes, y T denota el umbral.

La expresión dada en (21) considera que el valor de varianza de ruido, σ , como constante de diferente intervalo de tiempo. Pero la varianza del ruido puede variar con el tiempo dando como resultado varios valores diferentes de la varianza de la varianza del ruido en diferentes intervalos de tiempo.

El umbral universal puede ser injustificadamente grande desde su dependencia del número de muestras, aquí unos umbrales dependientes del nivel, que son más adaptables a las características de ruido y de señal, es adoptado por la fórmula:

$$T = \sigma\sqrt{2\ln N} = \frac{|\text{median}(w_{1,j})|}{0.6745} \sqrt{2\ln N} \quad (23)$$

Por lo tanto, en este documento, la selección de umbral adaptativo varianza de los coeficientes *Wavelet*, que se encarga de la variación de la varianza del ruido en cuenta, se utiliza.

6.4.5 Criterios de evaluación:

Con el fin de verificar el rendimiento de la eliminación de ruido enfoque propuesto, generado por ordenador ruidos con amplitudes variables se añaden a las señales de referencia conocidas; Por otra parte, los algoritmos clásicos se llevan a cabo para la comparación. Un número de parámetros cuantitativos se puede utilizar para evaluar el rendimiento del procedimiento de eliminación de ruido en términos de la calidad de la señal reconstruida. En este caso, se comparan los siguientes parámetros:

6.4.5.1 Proporción Señal-ruido:

$$SNR(dB) = 10 \ln \frac{\sum_{k=1}^N x^2(k)}{\sum_{k=1}^N [\bar{x}(k) - x'(k)]^2} \quad (24)$$

Donde $x'(k)$ es la señal contaminada y $x(k)$ es la señal original, N es una constante, sobre el número de muestras de la señal.

6.4.5.2 Error cuadrático medio(MSE):

Es usado para evaluar la calidad de filtrado de la señal , lo recomendable es que el valor sea lo más pequeño.

$$MSE = \sqrt{\frac{\sum_{n=1}^N |x(n) - x'(n)|^2}{N}} \quad (25)$$

6.5 Compresión de señales unidimensional

Un sistema típico de procesamiento de señales unidimensionales adquiere una gran cantidad de datos que deben almacenarse o transmitirse. Es necesario aplicar algún método para reducir el espacio de almacenamiento preservando el contenido significativo de la información para poder reconstruir posteriormente la señal. En algunas aplicaciones, este proceso de compresión tendrá que realizarse en tiempo real pero este no es el caso. Existen por lo tanto tres factores que caracterizan a los algoritmos de compresión de datos:

- **Eficiencia de compresión:** el objetivo es minimizar el número de bits de código almacenados eliminando redundancias presentes en la señal original. Se define la razón de compresión (RC) como el cociente entre el número de bits de la señal original por el número de bits almacenado en la señal comprimida. Factores tales como la anchura de banda, frecuencia de muestreo y fidelidad tienen un efecto importante sobre la razón de compresión.
- **Fidelidad de reconstrucción:** Un algoritmo de compresión de datos debe representar los datos con una fidelidad aceptable. En el campo de la Ingeniería

Biomédica, es frecuente validar la aceptabilidad clínica de la señal reconstruida mediante inspección visual. Se pueden medir también los residuos, es decir, la diferencia entre la señal reconstruida y la original mediante parámetros como el PRD (el tanto por ciento de la raíz cuadrada de las diferencias medias al cuadrado), definido como:

$$PRD = \frac{\sum_{i=1}^N [|x_{original}(i) - x_{reconstruida}(i)|]^2}{\sum_{i=1}^N [x_{original}(i)]^2} \times 100\% \quad (26)$$

Donde N es el número de muestras y las $x_{original}(i)$ y $x_{reconstruida}(i)$ representan las muestras originales y reconstruidas, respectivamente.

- **Complejidad:** Esta característica incide directamente en el tiempo de cálculo del algoritmo, y por tanto influye fundamentalmente en el caso de implementaciones en tiempo real.

La elección de un algoritmo de compresión se basa en un balance de las características comentadas, en función de la aplicación concreta.

6.5.1 Compresión mediante transformada *Wavelet*

La compresión mediante transformada *Wavelet* se basa en la posibilidad de representar la señal original mediante algunos coeficientes de aproximación y detalle de la descomposición obtenida mediante la transformada. Los términos de aproximación son las componentes de baja frecuencia y alto valor de escala, mientras que los de detalle corresponden a las frecuencias altas (escalas bajas). El proceso de descomposición de la señal puede verse, por tanto, como un filtrado sucesivo pasa-bajo y pasa-alto, de múltiples niveles, que para el caso de la transformada *Wavelet* Discreta proporciona el árbol de descomposición figura 44 .

El proceso de compresión se realiza en tres etapas. En primer lugar, se calcula la descomposición de la señal mediante la aplicación de una determinada *Wavelet* para un cierto nivel N. A continuación, se determinan los umbrales para las componentes de detalle correspondientes a todos los niveles desde 1 a N. Por último, la reconstrucción de la señal comprimida se realiza utilizando los coeficientes de aproximación originales para nivel N y los de detalle modificado tras la aplicación del umbral desde 1 hasta N.

El método de elección de umbrales en el paso 2 da lugar a dos aproximaciones diferentes de compresión. En un caso, se aplica un umbral global para todas las componentes quedándose con los coeficientes de mayor valor absoluto. En otro, se aplican umbrales dependientes del nivel, que pueden ser ajustados por separado proporcionando un mayor control sobre el proceso.

Capítulo 7

Experimento

Para realizar lo planteado por esta tesis se tiene la señal adquirida, tomando 1,048.576 trazas o muestras. Esta es una señal real adquirida a una frecuencia de muestreo de 100Mhz con el diseño realizado en la arquitectura FPGA y *OpenADC*.

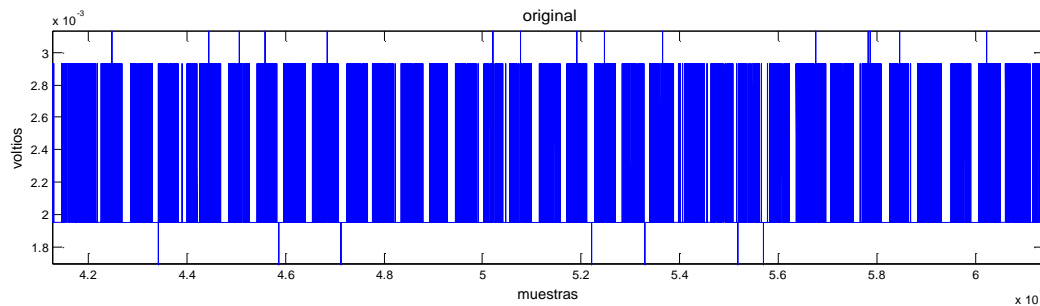


Figura 47 - señal extraída de la antena

Funte:Propia

Se le aplicó la transformada *Wavelet* continua a escalas de 1,2,4,7,10,...,46 escalas, la señal a mayor número de escala la frecuencia disminuye, en las escalas más bajas se aprecia mayor cantidad de cambios abruptos.

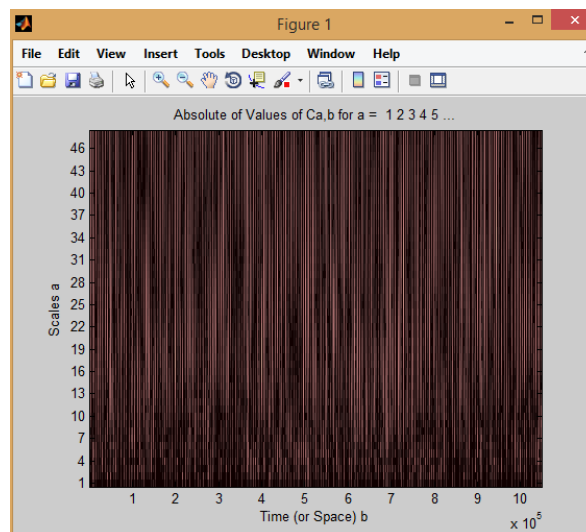


Figura 48- Transformada *wavelet* continua

Funte:Propia

Luego se aplicó el análisis Multirresolución a 3 escalas usando una *wavelet* tipo *haar*, dividiendo la señal en detalles y aproximaciones, los coeficientes tipo detalle se aprecian con mayor amplitud ya que estos representan el ruido (señal de alta frecuencia) y en las aproximaciones son representaciones de la señal de bajas frecuencias.

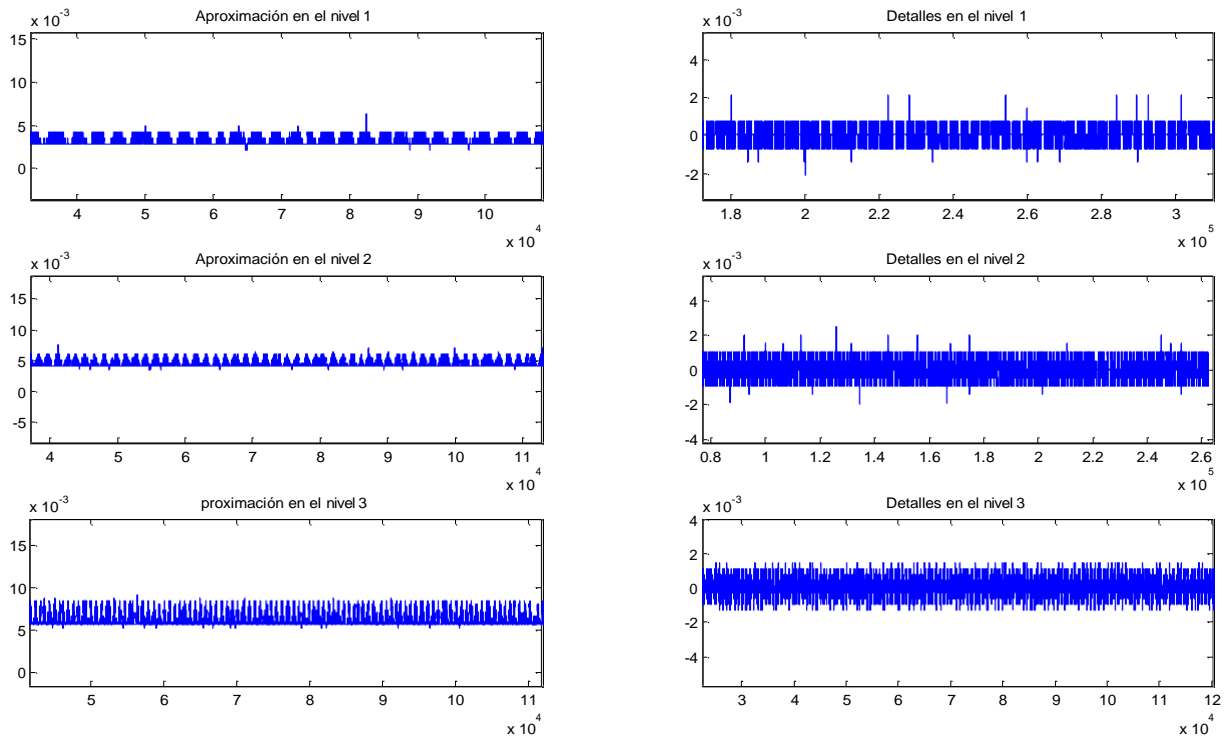


Figura 49 – Multirresolución

Fuente: Propia

Para realizar un estudio de señal primero se prepara la señal para luego realizar estudios diversos como en los ataques de canal lateral (SCA). En la figura 50 se muestra el proceso que sigue una señal para poder posteriormente estudio de SCA como lo menciona en la literatura [12], [34] y [14].

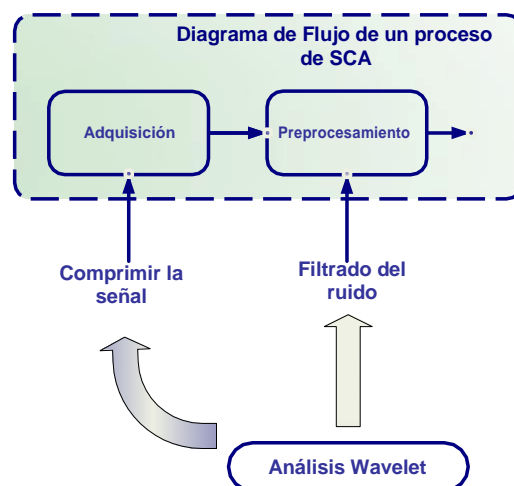


Figura 50 - Aplicación de Transformada Wavelet para distintas situaciones

Fuente: [28]

7.1 Filtrado y Compresión

Los datos son tomados del sistema de adquisición de datos explicado en capítulos anteriores. El rendimiento del método propuesto se evaluó mediante el cálculo de la relación señal a ruido (SNR) y la raíz cuadrada media de error (RMSE) después de eliminación de ruido. Los resultados revelan que el método propuesto ofrece un rendimiento superior que los métodos tradicionales no importan si las señales tienen ruidos pesados o ligeras incrustados. Para el filtro de la señal se utilizó un filtro tipo hard, se utilizó la madre wavelet *coif5* de 9 niveles, con umbral de 0.0015. En la figura 52 se ve la señal original y la reconstruida después de la compresión con los mismos parámetros mencionados líneas anteriores..

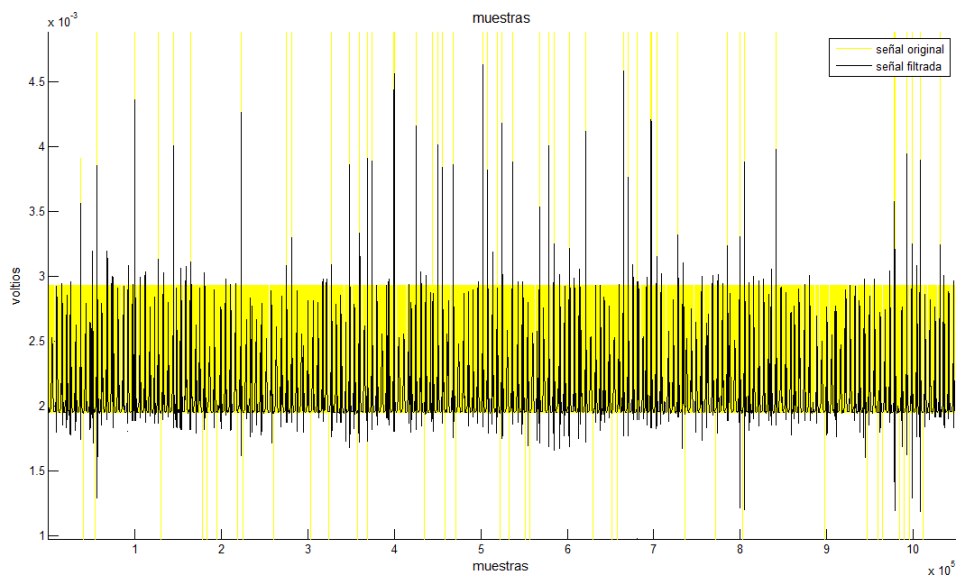


Figura 51 – Filtrado

Funte:Propia

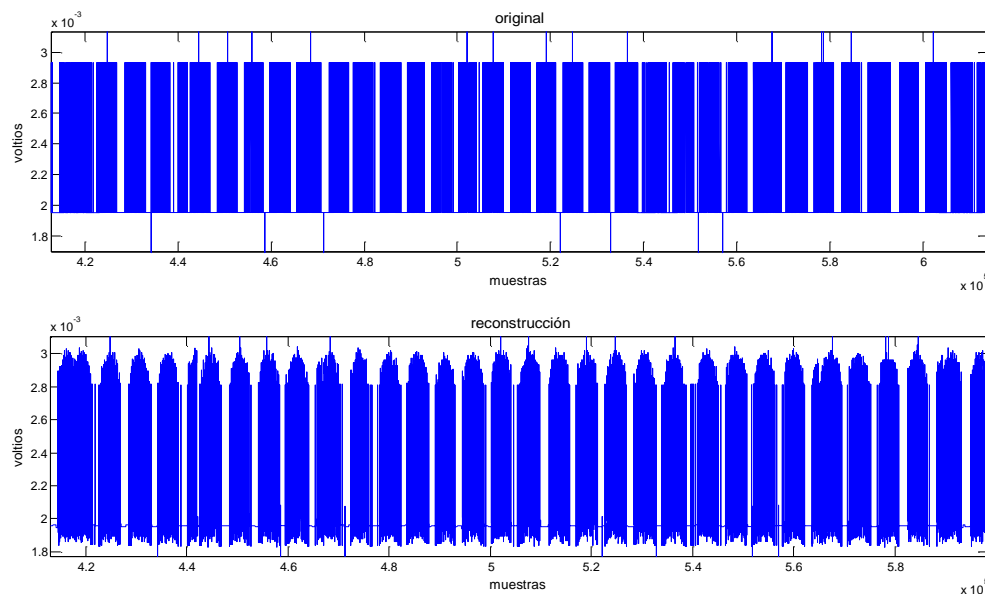


Figura 52 - señal original/ reconstruida

Funte:Propia

Se realizó finalmente filtrado y compresión para 3 familias wavelet distintas:

Filtro		Error Cuadrático medio	
Familia	Orden	Filtrado(db)	Reducción(%)
Coiflet4(coif4)	10	2.1006	28.16
	4	4.9413	27.42
Daubechies4(db4)	5	1.5164	30.45
	4	1.8140	30.29
Symlets(sym4)	20	2.4074	32.04
	10	3.9747	31.61

Figura 53 - MSE(Error cuadrático medio) de los diferentes filtros Wavelet

Fuente:Propia

Para realizar una elección de la función madre indicada se tienen para lograr la eliminación de ruido y también para encontrar una mejor compresión de la data es: $5 \times 10 \times 4 \times 2 \times 3 = 1200$ combinaciones posibles en las que cuentan con las siguientes combinaciones de funciones *Wavelet*, la escala, niveles de *Wavelet*, umbral estimador, umbral transformaciones y cambio de escala de umbral. De los procesos realizados se logró los siguientes resultados:

- El mejor resultado en el filtrado se obtuvo con la transformada *Wavelet Daubechies* de orden 5 ya que el error MSE es menor a la comparación de los demás filtros.
- El mejor resultado en la compresión se obtuvo con la transformada *Wavelet Symlets* de orden 20 ya la reducción porcentual es la mayor.

Conclusiones

- El análisis de la transformada *Wavelet* son una herramienta muy potente para el estudio de señales, debido a esto, son útiles en un amplio número de aplicaciones en muy diferentes campos, donde muchas veces se obtienen mejores resultados que con otras técnicas, a pesar de ser una herramienta relativamente nueva en el procesamiento de señales. En muchas ocasiones las *Wavelets* proporcionan características diferentes y más eficaces debido a su análisis local que es uno de sus mayores beneficios a la hora del análisis de señales.
- Se diseñó una arquitectura de Hardware basado en FPGA para el desarrollo de un sistema de adquisición de datos capaz de almacenar datos de señales de frecuencias altas y parte de señales de frecuencias muy altas.
- El trabajo realizado se ha desarrollado una aplicación capaz de integrar equipos electrónicos aplicados a la toma de datos también se ha hecho uso de programación en distintos lenguajes (Python y Matlab), además de configuración de Hardware(VHDL y Verilog), la parte matemática se necesitó mucho, sin embargo también se necesitó conocimientos de índole prácticos como por ejemplo la elección de la señal en estudio, ya que esta fue de manera intuitiva en la elección.
- El objetivo general y los objetivos específicos de las tesis se lograron satisfactoriamente dado que se realizó un estudio tanto de filtrado y compresión de señales dado que estos son parámetros donde se aprovecha este algoritmo.
- Muchas veces para el estudio de estas señales se prevalece que la tasa de muestreo sea alta ya que la señal presenta frecuencias medias a altas por ello se utiliza osciloscopios de muy alta tasa de muestreo, generalmente costosos. Sin embargo en esta tesis da razón a algunos proyectos realizados que indican que se puede lograr resultados buenos con una menor tasa de muestreo.

Referencias Bibliográficas

- [1] Liran Lerman, Stephane Fernandes Medeiros, Nikita Veshchikov, «Semi-Supervised Template Attack,» Universidad Pública de Bruxelles, Bruxelles, 2013.
- [2] Young Jin Kang, Tae Yong Kim, Jung Bok Jo and Hoon Jae Lee., «An Experimental CPA attack for Arduino Cryptographic Module,» *International Journal of Security and Its Applications*, vol. 8, nº 2, pp. 261 - 270, 2014.
- [3] Utsav Banerjee, Lisa Ho and Skanda Koppula, «Power-Based Side-Channel Attack for AES Key Extraction on the ATMega328 Microcontroller,» *Computer Systems Security*, 2015.
- [4] Youssef Souissi, Nicolas Debande, M. Abdelaziz El Aabid, «Novel Applications of Wavelet Transforms based Side-Channel Analysis,» *Telecom ParisTech, COMELEC*, 2011.
- [5] Colin O'Flynn and Zhizhang Chen, «A Case Study of Side Channel Analysis using Decoupling Capacitor Power Measurement with the OpenADC,» Springer - Verlag, Canada, 212.
- [6] Michel, Wavelet Toolbox User's Guide. 2012. The MathWorks, MISITI, 2012.
- [7] De Mulder, E., «Electromagnetic Techniques and Probes for Side - Channel Analysis on Cryptographic Devices,» 2010.
- [8] P. J.R. Rao, «Towards Sound Approaches to Counteract Power - Analysis Attacks,» *In Proceeding of CRYPTO 99*, vol. LNCS 1666, pp. 398-412, 1999.
- [9] P. Kocher, J.S. Coron, D. Naccache, «Statistics and Secret Leakage,» *In proceeding of Financial Cryptography*, vol. LNCS 1962, pp. 157-173, 2000.
- [10] J.M. Rabacy, Chandrakasan, B. Nikolic, Digital Integrated Circuit-A Design Perspective, Prentice Hall, 2da Ed., 2003.
- [11] J. M. M. Fernández, «Seguridad Física en Sistemas Empotrados,» Madrid, 2012.
- [12] Agrawal, D., B. Archambeault, and J.R. Rao, «The EM Side-Channel(s): Attacks and Assessment Methodologies,» *CHES2002*, pp. 29-45, 2002.
- [13] «The Side Channel Cryptanalysis Lounge-What is already Known?,» [En línea]. Available: <https://www.emsec.rub.de/research/projects/sclounge/>. [Último acceso: 28 2 2016].
- [14] K.H. Rhee and D. Nyang, eds, «Information Security and Cryptology-ICISC 2010,» de *The 13th Annual International Conference on Information Security and Cryptology*, Seoul, Korea, 2010.

- [15] K.Gandolfi, C.Mourtel,and F.Olivier, «Electromagnetic Analysis: Concrete Results.,» *Workshop on Cryptographic Hardware and Embedded Systems — CHES 2001*, vol. LNCS 2162, pp. 251-261, May 2001.
- [16] C.O'Flynn, «OpenADC,» Circuit Cellar, New Orleans, 2011.
- [17] Smith,D., «Signal and noise measurement techniques using magnetic field probes,» *IEEE International Symposium on*, vol. 1, pp. 559-563, 1999.
- [18] D. S. Consultants, «Signal and Noise Measurement Techniques Using Magnetic Field Probes,» IEEE, Oxford, 1998.
- [19] C. O'Flynn, «Power Analysis For Cheapskates». USA 30 7 2013.
- [20] P.P.CHU, FPGA Prototyping By VHDL Examples, New Jersey: WILEY, 2008.
- [21] X. Inc, «Xilinx All Programmable,» 2016. [En línea]. Available: <http://www.xilinx.com/products/silicon-devices/fpga/spartan-6.html>. [Último acceso: 27 Febrero 2016].
- [22] XILINX, 25 Octubre 2011. [En línea]. Available: http://www.xilinx.com/support/documentation/data_sheets/ds160.pdf. [Último acceso: 9 9 2015].
- [23] Avnet Electronics Marketing, 07 27 2011. [En línea]. Available: <http://xilinx.eetrend.com/files-eetrend-xilinx/download/201109/2080-3860-xlxs9lx9fpgamicroboard-ug072711.pdf>. [Último acceso: 9 9 2015].
- [24] Eduardo Mgdaleno Castelló, Manuel Rodríguez Valido, «Tutorial de Xilinx ISE,» España.
- [25] M. Thompson, «FPGAs Accelerate Time to Market for Industrial Designs,» EE Times, Julio 2004. [En línea]. Available: http://www.eetimes.com/document.asp?doc_id=1150608. [Último acceso: 25 8 2015].
- [26] M. A. S. Martínez, Diseño en FPGA de un circuito comparador de imaganes, México D.F: Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, 2005.
- [27] D. M. B. Larrotta y A. J. P. Aranguren, «Diseño VHDL de Sistemas Digitales sobre dispositivos lógicos programables FPGA's,» *Redalyc.org*, nº 4, pp. 37-49, 2004.
- [28] N. Technology, «OPENADC». California Patente OPENADC, 6 Noviembre 2012.
- [29] ISE, «Configuration and Programming Overview,» [En línea]. Available: http://www.xilinx.com/support/ise_c_configuration_overview.htm. [Último acceso: 1 10 2015].
- [30] Rashi Gupta , Making Use of Python, New York: Wiley Publishing, 2002.
- [31] V. Rossum, Python Tutorial Release 2.6.1, Python Software Foundation, 2009.
- [32] I. P. A. Moya, «Análisis de Fourier,» de *Señales y Sistemas- Fundamentos Matemáticos*, Costa Rica, Centro de Desarrollo de Material Bibliográfico (CDMB), 2013, pp. 124-153.
- [33] D.Gabor, «Theory of Communication,» *J.Inst.Elect*, vol. 93, pp. 429-457, 1946.
- [34] Anthony Teolis, Computational Signal Processing with Wavelets, Springer, 1998.
- [35] <http://www.mathworks.com/>, 2 2016. [En línea]. Available: <http://www.mathworks.com/help/wavelet/ug/wavelet-families-additional-discussion.html>. [Último acceso: 25 3 2016].
- [36] Matlab, «<http://www.mathworks.com/>,» 2016. [En línea]. Available: <http://www.mathworks.com/help/wavelet/ug/wavelet-families-additional-discussion.html>.

- discussion.html?searchHighlight=rbio. [Último acceso: 25 2 2016].
- [37] Chan, Andrew k., *Fundamentals of Wavelets: Theory, Algorithms, and Applications.*, WILEY, 2011.
 - [38] Leite, F.E.A.; Montagne, R., «Optimal wavelet filter for suppression of coherent noise with an application to seismic data,» *Physica A*, n° 387, p. 1439–1445, 2008.
 - [39] Alberto Rodríguez Gómez, «ANÁLISIS DE LA SEÑAL ECG(ELECTROCARDIOGRAMA), RECONOCIENDO LAS ONDAS P Y T Y EL COMPLEJO QRS USANDO LA TRANSFORMADA WAVELET.,» España, 2014.
 - [40] R.R. Coifman and D.L. Donoho, «Translation - Invariant De - Noising,» *Yale University and Stanford University*, 1995.
 - [41] Han, M.; Liu; Guo, «Noise smoothing for nonlinear time series using wavelet soft threshold,» *IEEE Signal Process*, n° 14, pp. 62-65, 2007.
 - [42] Baili, J.; Lahouar, S.; Hergli, M., «signal de-noising by discrete wavelet transform,» *NDT E. Int*, vol. 42, pp. 696-703, 2009.
 - [43] Liu, C.C.; Sun, T.Y.; Tsai, S.J., «Heuristic wavelet shrinkage for denoising. Appl. Soft.,» *Comput*, vol. 11, pp. 256-264, 2011.
 - [44] Chang, S.G.; Yu, B.; Vetterli., «M. Adaptive wavelet thresholding for image denoising and compression,» *IEEE Trans. Image Process*, vol. 9, pp. 1532-1546, 2000.
 - [45] Ahn, D.; Park, J.S., «Design of the low-pass filter using the novel microstrip defected ground structure.,» *IEEE Tran. Microw. Theory Tech.*, vol. 49, p. 86–93, 2001.
 - [46] Jwo, D.J.; Cho, T.S., «Critical remarks on the linearised and extended Kalman filters with geodetic navigation examples,» *Measurement*, vol. 43, p. 1077–1089, 2010.
 - [47] Wang, G.H.; Li, D.H., «Modified switching median filter for impulse noise removal,» *Signal Process*, vol. 90, p. 3213–3218, 2010.
 - [48] Baykal, B.; Constantinides, A.G., «A neural approach to the underdetermined-order recursive least-squares adaptive filtering,» *Neural Netw*, vol. 10, p. 1523–1531, 1997.
 - [49] Félix Martínez Giménez, Alfredo Peris Manguillot, Francisco Ródenas Escribá, *Tratamiento de señales digitales mediante wavelets y su uso con Matlab*, Valencia: Editorial Club Universitario, 2012.
 - [50] J. Zhang y Q. Zhong, «The Determination of the Threshold and the Decomposition Order in Threshold De-Noising Method Based on Wavelet Transform,» *Proceedings of the Chinese Society for Electrical Engineering*, pp. 118-122, 2008.
 - [51] Donoho, D.L.; Johnstone, I.M., «Adapting to unknown smoothness via wavelet shrinkage,» *Statist. Assoc*, vol. 90, p. 1200–1224, 1995.
 - [52] H.-N. L. Ting-Hua Yi, «Noise Smoothing for Structural Vibration Test Signals Using an Improved Wavelet Thresholding Technique,» *Sensors*, vol. 8, pp. 11205-11220, 2012.
 - [53] j. M. L. Murillo, «Extracción de características mediante criterios basados en teoría de la información,» España, 2007.
 - [54] K. S.K.Pattanaik, «DHT Based JPEG Image Compression Using a Novel Energy Quantization Method,» *IEEE International Conference on Industrial Technology*, pp. 2827-2832, Dec 2006.
 - [55] P. C. Kocher, «Timing Attacks on Implementations of Diffie - Hellman, RSA, DSS, and Other Systems,» *Cryptography Research, Inc.*, 1995.

- [56] P. Kocher, «Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. in Advances in Cryptology,» *CRYPTO'96*, 1996.
- [57] W. Schindler, «A timing attack against RSA with the chinese remainder theorem. in Cryptographic Hardware and Embedded Systems,» *CHES 2000*, 2000.
- [58] E. Tromer, «Acoustic Cryptanalysis: on nosy people and noisy machines.,» *Eurocrypt2004*, May 2004.
- [59] M. a. Backes, «Acoustic Side-Channel Attacks on Printers.,» *USENIX Security Symposium*, 2010.
- [60] M. Kuhn, «Optical time-domain eavesdropping risks of CRT displays in Security and Privacy,» *IEEE Symposium on.2002.IEEE.*, 2002.
- [61] Loughry, J. and D.A., «Information leakage from optical emanations,» *ACM Transactions on Information and System Security (TISSEC)*, vol. 5, n° 3, pp. 262-289, 2002.
- [62] Paul Kocher, Joshua Jaffe and Benjamin Jun, «Introduction to Differential Power Analysis and Related Attacks,» de *Cryptography Research*, San Francisco, CA 94102, 1998.
- [63] J.-J. Quisquater and D. Samyde, «Electro Magnetic Analysis (EMA): Measures and Countermeasures for Smart Cards,» de *Research in Smart Cards*, France, 2001.
- [64] M. Droettboom, E. Firing, J. Hunter, Matlab Release 0.985.1, 2008.
- [65] Python.org, «Python,» [En línea]. Available: <http://www.python.org/download/releases>. . [Último acceso: 24 7 2015].
- [66] Sentdex, «<https://www.youtube.com>,» 1 21 ene. 2015, 21 enero 2015. [En línea]. Available: https://www.youtube.com/watch?v=jnpC_Ib_lbc. [Último acceso: 10 octubre 2015].
- [67] trforcelink, «<https://www.youtube.com>,» 18 marzo 2014. [En línea]. Available: <https://www.youtube.com/watch?v=zPMr0lEMqpo>. [Último acceso: 11 octubre 2015].
- [68] Paul Kocher, «Timing Attacks on Implementations of Diffie - Hellman, RSA, DSS, and Other Systems,» de *Cryptography Research*, San Francisco, 1995.
- [69] J. M. Bernardo, «Metodología Basesiana para la toma de desiciones,» España, 2004.
- [70] J. Martínez Escanaverino, «Estrategia para el diseño paramétrico basado en modelos,» *Sistema de Información Científica Redalyc*, vol. 11, n° 3, pp. 39-46, 2008.
- [71] C. O'Flynn, «Asamblea,» 21 7 2013. [En línea]. Available: <https://www.asamblea.com/spaces/openadc/git/source/master/doc>. [Último acceso: 22 11 2015].

Anexos

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Descomposicion%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% La variable data contiene la señal a analizar

w = 'haar'; % Fijamos la wavelet
n = 3; % Niveles
[C,L] = wavedec(data,n,w); % Descomposicion de la señal

% El vector C contiene la subsenal tendencia a nivel 3 y las
% subsenales fluctuacion a niveles 3, 2 y 1.
% El vector L contiene ordenadas las longitudes de la
% tendencia de tercer nivel y las fluctuaciones de tercero, segundo y
% primer nivel. El ultimo valor de L es la longitud de la
% señal original.

% Extraemos los valores de la tendencia y de
% las fluctuaciones del vector C.

cA1 = appcoef(C,L,w,1);
cA2 = appcoef(C,L,w,2);
cA3 = appcoef(C,L,w,3);
cD1 = detcoef(C,L,1);
cD2 = detcoef(C,L,2);
cD3 = detcoef(C,L,3);

% Representamos estos coeficientes.

figure;
subplot(3,2,1); plot(cA1);
title('Aproximación en el nivel 1');

axis([1 length(data)/2 -1.5 1.5]);
subplot(3,2,2); plot(cD1);
title('Detalles en el nivel 1');
axis([1 length(data)/2 -0.03 0.03]);
subplot(3,2,3); plot(cA2);
title('Aproximación en el nivel 2');
axis([1 length(data)/2 -1.5 1.5]);
subplot(3,2,4); plot(cD2);
title('Detalles en el nivel 2');
axis([1 length(data)/2 -0.03 0.03]);
subplot(3,2,5); plot(cA3);
title('proximación en el nivel 3');
axis([1 length(data)/2 -1.5 1.5]);
subplot(3,2,6); plot(cD3);
title('Detalles en el nivel 3');
axis([1 length(data)/2 -0.03 0.03]);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Descomposicion%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

[illegible]

```

function [y] = cumenergy(x)
%
%y = cumsum(x.^2)/(norm(x)^2);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [e] = rms(x,y)
%Devuelve el error RMS entre x e y.
%Señales x e y deben tener mismo tamaño.

e = sqrt(norm(x-y)^2/length(x));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Fichero denoise.m
%% Eliminación del ruido de una senyal
%% La variable externa s contiene la señal contaminada
s=data;
w = 'coif5'; % Wavelet a utilizar
n = 9; % Niveles de transformada
thr_met = 'h'; % Método threshold: Hard
% Numero de veces la desviación típica en umbral
coef = sqrt(2*log(length(s)));

[C,L] = wavedec(s,n,w); % Transformada wavelet

% Cálculo del umbral (threshold)
cD1 = detcoef(C,L,1); % Extraemos la primera fluctuación
des_tip = std(cD1); % Desviación típica
thr = coef*des_tip; % Umbral (threshold)

Cthr = wthresh(C,thr_met,thr); % Threshold la transformada

s_den = waverec(Cthr,L,w); % Reconstruimos

%% Resultados y representación gráfica
figure;
hold on;
axis([1 length(s) min(s) max(s)]);
title('muestras','FontSize',12);

xlabel('muestras','FontSize',11)
ylabel('voltios','FontSize',11)
plot(s,'y');
plot(s_den,'k');
hold off;
legend('señal original','señal filtrada','line');
error = rms(s,s_den); % Error
[error_men,err] = sprintf('Error: %d \n',error);
ene_ret = 100*energy(s_den)/energy(s); % Porcentaje energía retenida
[ene_ret_men,err] = sprintf('Energía retenida: %2.3f %% \n',ene_ret);
sprintf('%s%s',error_men,ene_ret_men)

```


[illegible]

```
function [y] = energy(x)
%ENERGY Computes the energy of a signal.
%   y = ENERGY(x) returns the energy of x.
%   The energy computed as the sum of the
%   square of the elements of x.
```

```
% ver el archivo CUMENERGY.
```

```
y = norm(x)^2;
```

