Implementación de un Filtro Pasobajas Digital en FPGA

Pérez Armenta Andrea [(1)(3)], Portillo Flores Guillermo [(1)(4)], Rodríguez Gutiérrez Morrison Ernesto [(1)(3)], Martínez García Mauricio [(2)], Ambrosio Bastián José [(1)], Rodríguez Torres Octavio [(1)]

***Resumen*— El filtro digital que se analiza en este documento está descrito mediante un algoritmo que se construye en un dispositivo programable, utilizando módulos ya diseñados por terceros. Para este proyecto se realizó un filtro tipo IIR paso bajas de la familia Butterworth, el cual se utiliza para atenuar la amplitud de las componentes de alta frecuencia de una señal, empleando bloques modulares básicos y de fácil reconfiguración. La señal aplicada no requiere de ninguna adaptación ya que se utilizaron entradas y salidas con niveles estándar para audio. Se confirma el correcto funcionamiento del filtro digital mediante mediciones realizadas con un analizador de espectros.**

I. INTRODUCCIÓN

El presente trabajo tiene por objetivo la implementación de un filtro digital en un dispositivo de cómputo reconfigurable (FPGA). Para ello se requiere el diseño teórico del filtro, la descripción modular del circuito, la síntesis del código y la programación de FPGA. Finalmente, se verifica el correcto funcionamiento del sistema mediante el uso o de instrumentos de laboratorio.

La importancia de los filtros electrónicos se debe a su gran diversidad de aplicaciones, como por ejemplo, ecualizadores de audio, cancelación de ruido de una señal, procesamiento de señales biomédicas, voz, imágenes, telecomunicaciones, audio digital, ingeniería sísmica, sólo por nombrar algunas [1][2][3][4][5][6].

El término filtro es usado para describir un dispositivo que discrimina en su salida ciertos elementos. Un filtro electrónico es un dispositivo que permite pasar señales eléctricas a través de él, a una cierta frecuencia [7]. Los filtros electrónicos pueden ser analógicos o digitales y se clasifican según: la familia (Butterworth o Chebyshev), el orden (grado de aceptación o rechazo de frecuencias a partir de la frecuencia de corte) y la respuesta en frecuencia (parámetro que describe las frecuencias que puede reproducir un dispositivo). La función de un filtro analógico o digital es modificar las componentes de la frecuencia, la principal diferencia es que el analógico se construye y el digital se programa. Los filtros digitales se pueden realizar en formas FIR (*Finite Impulse Response*) e IIR (*Infinite Impulse Response*), la diferencia entre ellos radica en la estabilidad y el orden de su función de transferencia [8].

Un filtro es un circuito caracterizado por una entrada y una salida de forma que en la salida solo aparecen parte de las componentes de frecuencia de la señal de entrada. Son por tanto circuitos que se pueden caracterizar por su función de transferencia $H(\omega)$.

$$y(\omega) = H(\omega)x(\omega) \qquad (1)$$

La elección del filtro depende de la naturaleza del problema y de las especificaciones que se le deseen dar a la respuesta en frecuencia. Los filtros se pueden conectar en cascada hasta obtener el orden de la función de transferencia que se necesite.

Los filtros FIR son aquellos cuya respuesta a una señal impulso como entrada, tendrá un número finito de términos no nulos. Los FIR suelen tener un gran número de aplicaciones de audio y son siempre estables [7].

En contraste, los filtros IIR son aquellos en los cuales si la entrada es una señal impulso, la salida tendrá un número infinito de términos que no sean nulos, es decir, no tienen un estado de reposo.

La función de transferencia es aquella que a través de un cociente relaciona la respuesta de un sistema con una señal de entrada y se define por la siguiente ecuación:

$$H(f) = numerador\ (f)/denominador\ (f) \qquad (2)$$

Se dice que las raíces del numerador son los ceros y las raíces del denominador son los polos.

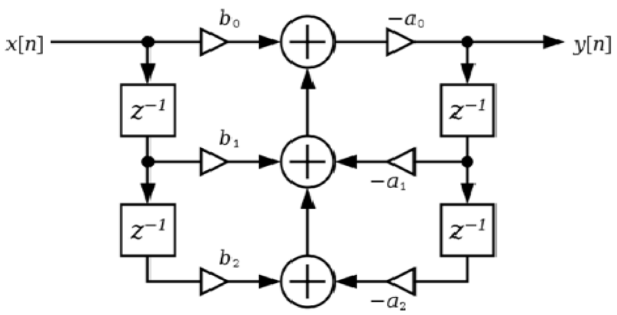


Figura 1. Estructura lineal de un filtro IIR

(1)Pertenecen al GIDi de SISTEMAS EMBEBIDOS (2) Pertenece al GIDI de PROCESAMIENTO DIGITAL DE SEÑALES, (3) Son alumnos de la carrera INGENIERÍA EN ELECTRÓNICA y (4) Es alumno de INGENIERÍA CIBERNÉTICA Y EN SISTEMAS COMPUTACIONALES de la Facultad de Ingeniería, y realizaron el proyecto dentro del curso DISEÑO LÓGICO (Email:gportillo94@hotmail.com).El proyecto fue asesorado por MTRO. OCTAVIO RODRÍGUEZ TORRES, DR. JOSÉ AMBROSIO BASTIÁN y DR. MAURICIO A. MARTÍNEZ GARCÍA.

En la Figura 1 se muestra la estructura de un filtro lineal, donde x[n] es la entrada, b_0, b_1, b_2, a_0, a_1 y a_2 son los coeficientes para determinar la respuesta en frecuencia que se desea, z^{-1} es un retraso para el cual se utilizan *flip flops* tipo D, y[n] es la salida, y se emplean sumadores y multiplicadores. Estas estructuras son realizables de manera relativamente simple, y si se desea un filtro de orden mayor, puede construirse conectando varios filtros de orden dos en cascada en la implementación computacional [7].

Dentro de los filtros IIR se puede hablar, entre otras, de las familias Chebyshev y Butterworth. La primera tiene una caída en la respuesta en frecuencia más pronunciada que la segunda. Los polos se distribuyen sobre una elipse en la cual, sus ceros se encuentran en el eje imaginario. Por el otro lado, el filtro Butterworth es un filtro muy básico diseñado para producir la respuesta más plana que sea posible hasta la frecuencia de corte. Lo que cambia cuando se trabaja con un filtro de orden mayor, es que, la caída va a tener una pendiente más pronunciada para frecuencias mayores a la frecuencia de corte [9].

La respuesta en frecuencia es una medida cuantitativa del espectro de salida que describe el rango de frecuencias que mantiene o atenúa un filtro. La respuesta en frecuencia de cualquier sistema de audio ideal debería de ser plana, es decir, debe tratar igual a todo el sonido entrante. El oído humano puede captar un margen de audiofrecuencia de 20-20,000 Hz. Los sonidos graves son por lo general, los que van de 20-400 Hz, los medios de 400 - 2,000 Hz y los agudos de 2,000-20,000 Hz aproximadamente. Si se desea enfatizar el sonido grave, se requiere de un filtro paso bajas, mientras que para enfatizar los sonidos agudos se requiere de filtro paso altas y para los sonidos medios se utiliza un filtro paso banda. Las gráficas cada uno de los filtros se muestran en la Figura 2, en donde la respuesta o ganancia dada en (dB) es la relación que hay entre la potencia de entrada y de salida [7][9][10].

$$Ganancia = 10\ log_{10}\left(\frac{E\{|y(n)|^2\}}{E\{|x(n)|^2\}}\right) \qquad (3)$$

Donde $E\{\ \}$ denota el operador de valor esperado y $|\ |^2$ denota el módulo cuadrado

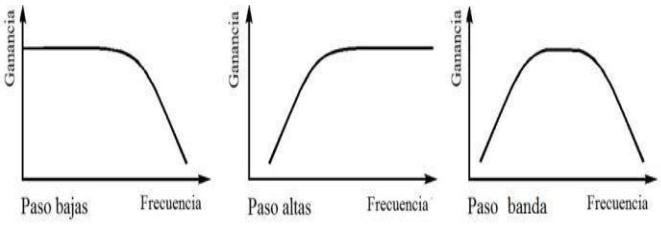


Figura 2. Respuesta en Frecuencia

II. CÓMPUTO RECONFIGURABLE

En electrónica el cómputo reconfigurable se refiere a un microcircuito que permite la reconfiguración del hardware con el simple cambio del algoritmo que se programa. Estos circuitos pueden realizar funciones tan sencillas como compuerta lógica o incluso algo más complejo como un microprocesador. Estos dispositivos tienen celdas que se configuran con una función específica, ya sea como un *flip-flop*, un multiplexor o una compuerta. La labor del usuario es programar el algoritmo que tendrá el dispositivo. Por ende, el usuario debe de especificar la función lógica que éste realizará [11].

Los FPGAs (*Field Programmable Gate Array*) son circuitos integrados que se configuran en campo después de su fabricación. Para configurarlos se requiere un lenguaje de descripción de hardware (*HDL, Hardware Description Language*). Dentro de sus principales ventajas está la capacidad de actualizar su funcionalidad posterior a la venta del producto, de reconfiguración parcial de una porción del diseño y su bajo costo de diseño. Estos dispositivos cuentan con grandes ventajas como programar o diseñar el circuito que uno desea, es fácil detectar y corregir los errores, y no se tiene que construir el circuito de forma física [11][12].

Los FPGAs pueden implementar un diseño a partir de un lenguaje HDL o de un gráfico. La implementación gráfica consiste en armar un circuito a partir de bloques funcionales ya definidos que se deben de interconectar de la forma adecuada. Los lenguajes de descripción de hardware son lenguajes de programación especializados que se utilizan para definir la estructura, configuración, diseño y operación de los PLDs, sobre todo de los FPGAs [12][13].

Los lenguajes HDL más comunes son:

1. **ABEL:** Advanced Boolean Expression Language. Permite describir un diseño concurrentemente mediante tablas de verdad o ecuaciones lógicas.

2. **VERILOG:** es un lenguaje de descripción de hardware (HDL, del Inglés Hardware Description Language) soporta el diseño, prueba e implementación de circuitos analógicos y digitales y de señal mixta a diferentes niveles de abstracción.

3. **VHDL:** Es el acrónimo que representa la combinación de VHSIC y HDL, donde VHSIC es el acrónimo de Very High Speed Integrated Circuit. Aunque puede ser usado de forma general para describir cualquier circuito se usa principalmente para programar PLD (Programable Logic Device - Dispositivo Lógico Programable), FPG), ASIC y similares.

El filtro se implementó en una tarjeta Altera DE2-115. Sus características son:

1. Memoria 8 MB Flash
2. Memoria 128 MB SDRAM
3. Memoria 2 MB SRAM
4. Procesador Cyclone IV

5. Señal de reloj de 50 Mhz

III. METODOLOGÍA

El proceso de diseño e implementación del filtro descrito en este documento, se muestra en la figura 3. Los parámetros del diseño del filtro fueron elegidos de forma arbitraria, utilizando valores estándar. Se seleccionó un filtro paso bajas de Butterworth de orden 2, con frecuencia de muestreo de 48 Khz, frecuencia de corte 1 Khz., datos de 16 bits de los cuales un bit es de signo, uno de guarda y 14 bits de parte fraccionaria.

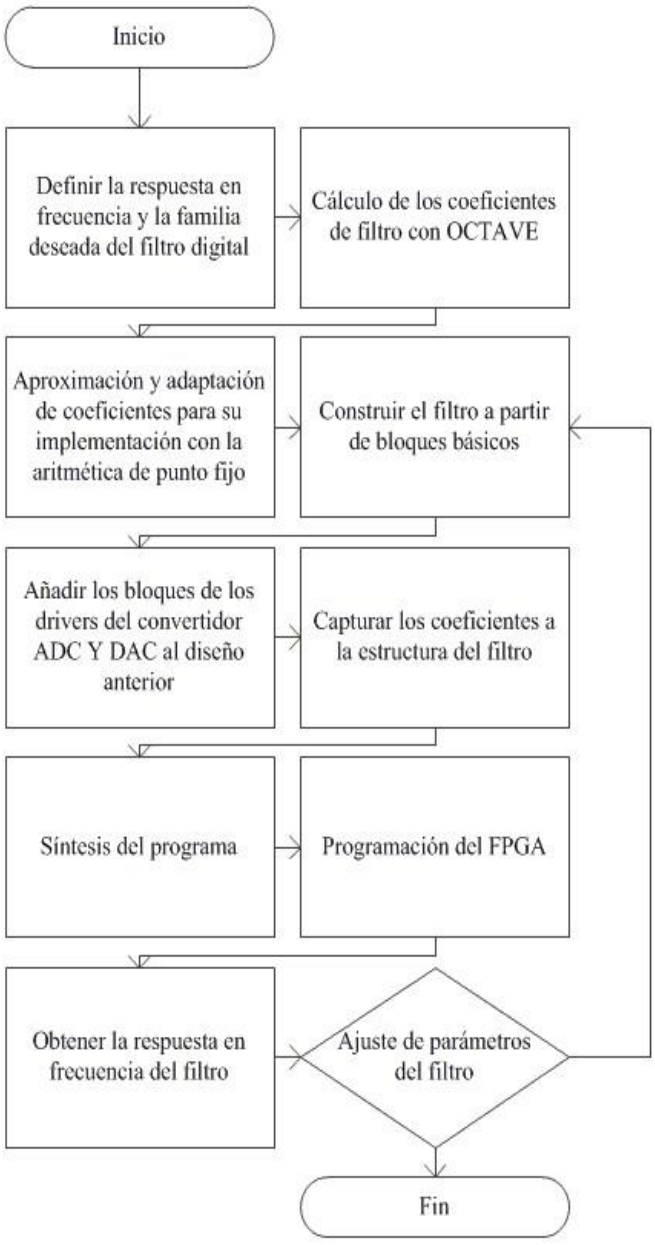


Figura 3. Diagrama del proceso de diseño del filtro.

Para el cálculo de los coeficientes se utilizó el software GNU OCTAVE como se muestra en el Programa 1. Tradicionalmente el diseño de un filtro a partir de sus especificaciones comprende el diseño del filtro analógico equivalente y su discretización con una transformación bilineal, adicionalmente para la implementación en punto fijo se requiere aproximar los coeficientes a dicha aritmética.

El diseño del filtro se realizó principalmente utilizando el lenguaje esquemático como se muestra en las Figuras 4 y 5.

Concretamente se realizó a partir de los siguientes bloques: Funciones primitivas, Megafunciones de Quartus, drivers proporcionados por el fabricante y bloque programados en VHDL o Verilog. Se muestra un ejemplo en el Programa 2.

Programa 1. Función programada en Octave con la definición de la frecuencia de muestreo, frecuencia de corte, cálculo de los coeficientes del filtro y aproximación a la aritmética de punto fijo.

```
function [num den] = calcularCoeficientes()
  orden=2;    %Orden del filtro.
  fc=1000;    %Frecuencia de corte.
  fs=48000;    %Frecuencia de muestreo.
  [num den]=butt er(orden,fc/(fs/2)); _%Calculo
  de los coeficientes. num=num*2^14;
  %Aproximación a la aritmética den=den*2^14;
  % de punto fijo den(2)=65536+den(2);
end
```

Programa 2. Código en VHDL para el bloque de retraso.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY signal_delay IS
      PORT(clk:IN  STD_LOGIC;
           A  :IN  STD_LOGIC_VECTOR(15 DOWNTO 0);
           Z  :OUT STD_LOGIC_VECTOR(15 DOWNTO 0);
           );
END signl_delay

ARCHITECTURE Behavioral OF signal_delay IS
BEGIN
      PROCESS (clk) BEGIN
               IF (rising_edge(clk)) THEN
                     Z<=A
               END IF
      END PROCESS
END Behavioral
```

CUADRO I
RESULTADOS OBTENIDOS UTILIZANDO DIFERENTES BASES DE DATOS

a0	16384
a1	35793
a2	13615
b0	64
b1	128
b2	64

Nota: la normalización se refiere a, hacer que el coeficiente a0 tome siempre el valor de uno que para una representación con 14 bits equivale a multiplicar todos los coeficientes por 2^{14}

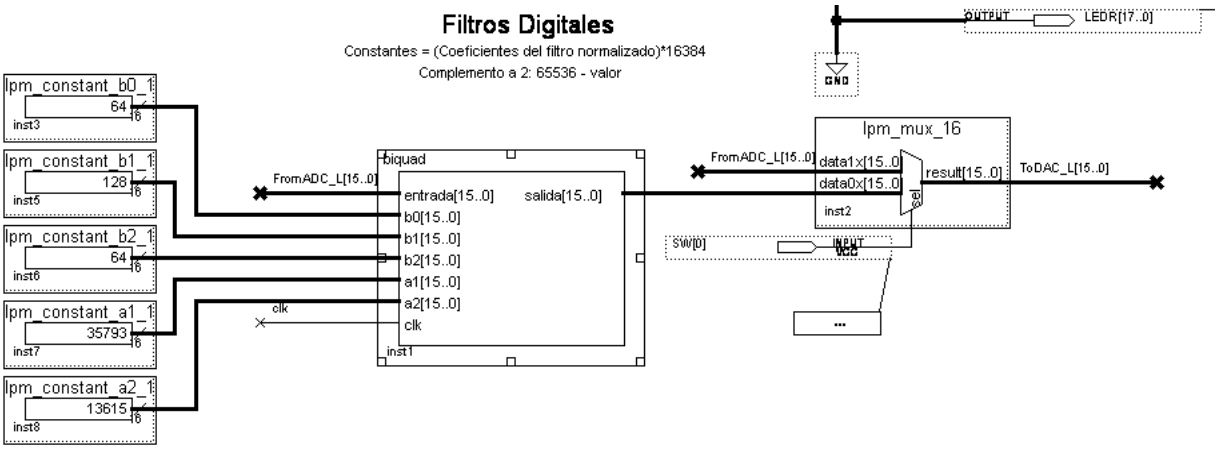


Figura 4. Diagrama esquemático del bloque principal del filtro, mostrando los coeficientes

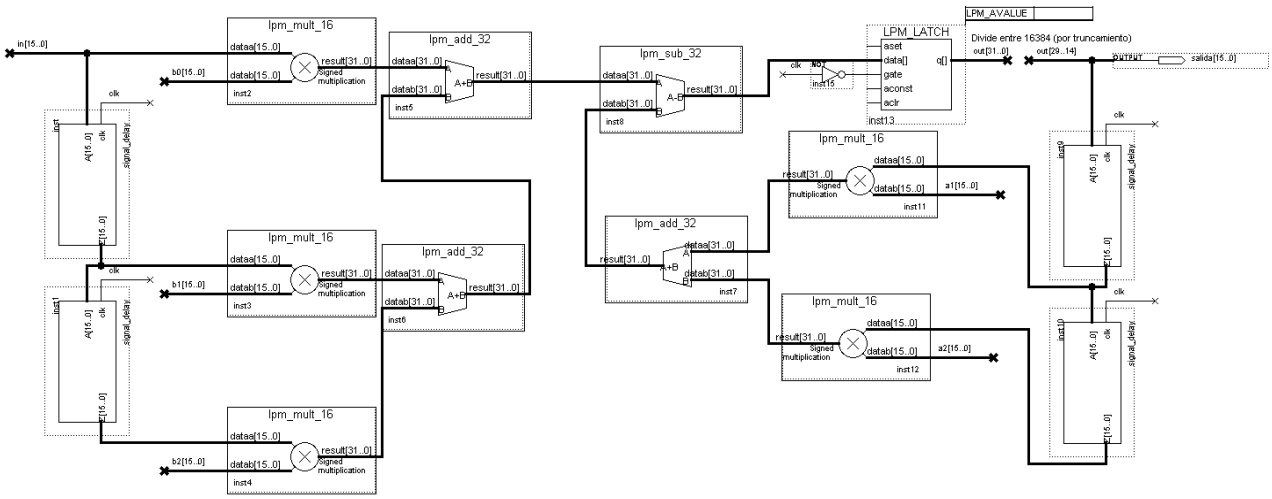


Figura 5. Diagrama esquemático de la implementación del filtro en Quartus

IV. RESULTADOS

Los resultados de la medición de respuesta en frecuencia que se realizó utilizando el analizador de espectros en software VISUAL ANALYSER, el cual genera un barrido cubriendo todo el rango de frecuencias de audio, y mide la ganancia del filtro para cada una de ellas.

Este resultado se visualiza en la Figura 6 a través de una gráfica de respuesta en frecuencia, en la que se puede apreciar la respuesta casi plana del filtro a frecuencias inferiores a la frecuencia de corte, la atenuación de 3 dB en la frecuencia de corte, y la caída monotónica para frecuencias superiores correspondiente a una pendiente de 40 dB por década lo cual corresponde a un filtro de Butterworth de orden 2.

Finalmente en la Figura 7 puede observarse la cantidad de recursos utilizados en la implementación como el número de elementos lógicos, registros, modelo del dispositivo, numero de bits de memoria, los multiplicadores de 9 bits y el número de pines.

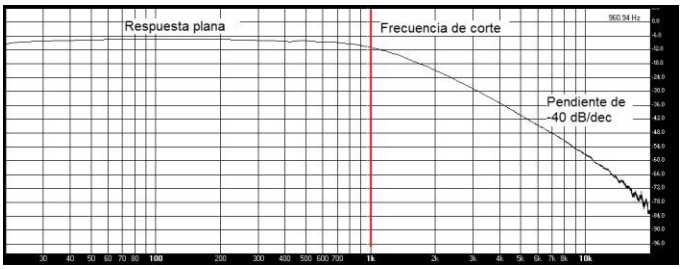


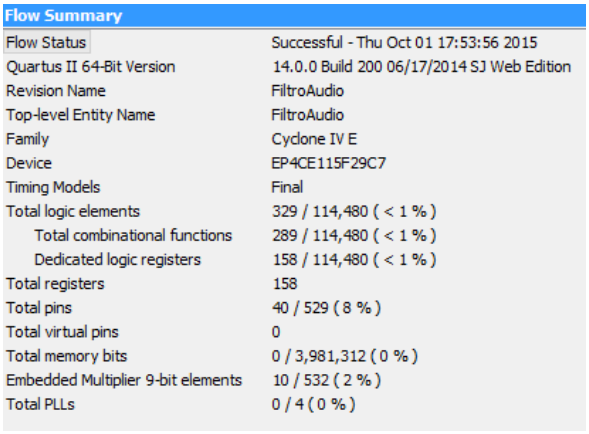
Figura 6. Medición de respuesta en frecuencia del filtro

V. CONCLUSIONES

Por medio de este trabajo se pudo demostrar que utilizando cómputo reconfigurable con bloques modulares básicos y conocimientos esenciales de filtros, se puede realizar un proyecto complejo, con cierta facilidad y con aplicaciones en diversas áreas. Los resultados pueden variar dependiendo de los parámetros que seleccionen para el filtro. Para la propuesta presentada, al hacer una respuesta en frecuencia pasobajas y con una frecuencia de corte de 1 kHz, se logró obtener el resultado esperado, pero esto podría fallar debido

a limitaciones de los instrumentos de medición, en cuyo caso se tendría que hacer una compensación.

Es importante mencionar que la alternativa de implementación de un filtro digital es utilizar un DSP (Digital Signal Processor) el cual tendría ciertas ventajas como el costo, consumo de energía, entre otras. Sin embargo, la propuesta aquí planteada aprovecha las ventajas de la programación gráfica con la posibilidad de implementar varios filtros en paralelo. Finalmente, la implementación de filtros digitales más complejos depende de la capacidad del FPGA.



Flow Summary	
Flow Status	Successful - Thu Oct 01 17:53:56 2015
Quartus II 64-Bit Version	14.0.0 Build 200 06/17/2014 SJ Web Edition
Revision Name	FiltroAudio
Top-level Entity Name	FiltroAudio
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	329 / 114,480 (< 1 %)
Total combinational functions	289 / 114,480 (< 1 %)
Dedicated logic registers	158 / 114,480 (< 1 %)
Total registers	158
Total pins	40 / 529 (8 %)
Total virtual pins	0
Total memory bits	0 / 3,981,312 (0 %)
Embedded Multiplier 9-bit elements	10 / 532 (2 %)
Total PLLs	0 / 4 (0 %)

Figura 7. Resumen de recursos utilizados en el FPGA Cyclone IV

VI. REFERENCIAS

[1] A. F. Shalash, "Hybrid FIR-IIR Adaptive Echo Canceller for Wireline Applications," in *Conference Record of the Thirty-Ninth Asilomar Conference on Signals, Systems and Computers, 2005*, 2005, pp. 364–368.

[2] J. Ambrosio Bastian, "Desarrollo de sistemas lineales y no lineales para la eliminacion de interferencias en canales de comunicacion," Thesis, 2008.

[3] D. Xu and J. Chiu, "Design of a high-order FIR digital filtering and variable gain ranging seismic data acquisition system," in *, IEEE Southeastcon '93, Proceedings*, 1993, p. 6 p.–.

[4] K. M. Gaikwad and M. S. Chavan, "Removal of high frequency noise from ECG signal using digital IIR butterworth filter," in *2014 IEEE Global Conference on Wireless Computing and Networking (GCWCN)*, 2014, pp. 121–124.

[5] E. G. Lee, J. A. Bastian, M. N. Miyatake, and H. P. Meana, "Decision feedback equalizers for high speed data communications," in *4th International Conference on Antenna Theory and Techniques, 2003*, 2003, vol. 1, pp. 341–344 vol.1.

[6] M. Nakano, E. González, J. Ambrosio, and H. M. Pérez, "Eliminación de Interferencias en Canales Telefónicos mediante Ecualizadores de Decisión Retroalimentada," *Inf. Tecnológica*, vol. 16, no. 6, pp. 51–62, 2005.

[7] J. G. Proakis and D. G. Manolakis, *Tratamiento digital de señales*. Pearson Educación, 2007.

[8] S. Lapatine, *Electrónica en Sistemas de Comunicación*. Editorial Limusa S.A. De C.V., 1996.

[9] S. K. Mitra, *Procesamiento de señales digitales: un enfoque basado en computadora*. McGraw-Hill-Interamericana, 2007.

[10] A. Ambardar, *Procesamiento de señales analógicas y digitales*. Thomson, 2002.

[11] I. Grout, *Digital Systems Design with FPGAs and CPLDs*. Newnes, 2011.

[12] S. Brown and J. Rose, "FPGA and CPLD Architectures: A Tutorial," *IEEE Test*, vol. 13, no. 2, pp. 42–57, Jun. 1996.

[13] "Implantación del Algoritmo de Filtrado Adaptable QR-RLS en FPGA." [Online]. Available: http://docplayer.es/538795-Implantacion-del-algoritmo-de-filtrado-adaptable-qr-rls-en-fpga.html. [Accessed: 01-Oct-2015].