UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO

CARRERA DE TELECOMUNICACIONES

DESARROLLO DE FILTROS DIGITALES DE PROCESAMIENTO DE VOZ PARA EL LABORATORIO DE TPS UTILIZANDO RASPBERRY PI Y MATLAB-SIMULINK.

Trabajo de titulación previo a la obtención del
Título de Ingeniero en Telecomunicaciones

AUTORES: LUIS DANIEL SARABIA GUILLERMO
BRAYAN ALEXANDER BRAVO JAYA

TUTOR: LUIS GERMÁN OÑATE CADENA

Quito-Ecuador

2025

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Nosotros, Luis Daniel Sarabia Guillermo documento de identificación N° 1722661244 y Brayan Alexander Bravo Jaya con documento de identificación N° 1725000184; manifestamos que:

Somos los autores y responsables del presente trabajo; y, autorizamos a que sin fines de lucro La Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Quito, 07 de julio del año 2025.

Atentamente,

Luis Daniel Sarabia Guillermo
1722661244

Brayan Alexander Bravo Jaya
1725000184

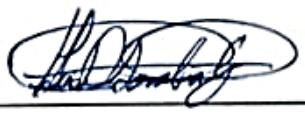
CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Nosotros, Luis Daniel Sarabia Guillermo documento de identificación N° 1722661244 y Brayan Alexander Bravo Jaya con documento de identificación N° 1725000184, expresamos nuestra voluntad y cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del Proyecto Técnico "Desarrollo de Filtros Digitales de Procesamiento de Voz para el laboratorio de TPS utilizando Raspberry Pi y Matlab-Simulink.", el cual ha sido desarrollado para optar por el título de: Ingenieros en Telecomunicaciones en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Quito, 07 de julio del año 2025.

Atentamente,



Luis Daniel Sarabia Guillermo
1722661244

Brayan Alexander Bravo Jaya
1725000184

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Luis Germán Oñate Cadena con documento de identificación N° 1712157401, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: DESARROLLO DE FILTROS DIGITALES DE PROCESAMIENTO DE VOZ PARA EL LABORATORIO DE TPS UTILIZANDO RASPBERRY PI Y MATLAB-SIMULINK. realizado por Luis Daniel Sarabia Guillermo documento de identificación N° 1722661244 y Brayan Alexander Bravo Jaya con documento de identificación N° 1725000184, obteniendo como resultado final el trabajo de titulación bajo la opción Proyecto Técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Quito, 07 de julio del año 2025.

Atentamente,

Ing. Luis Germán Oñate Cadena, MSc

1712157401

DEDICATORIAS

Agradezco profundamente a Dios por mostrarme el camino y brindarme la sabiduría necesaria para seguir esta carrera. A mis padres, Marco Sarabia y Sonia Guillermo, por estar a mi lado en cada etapa de la vida, con amor y un apoyo incondicional que han sido mi mayor fortaleza. A mis abuelos paternos, María Dolores Changotasig y Luis Sarabia, y a mis abuelos maternos, Hugo Guillermo y María Maxi, por su ejemplo, enseñanzas y cariño, que me inspiran a superarme cada día. A mi tía Blanca Sarabia, a mi prima Estefanía y a toda mi familia, por su cercanía, aliento y respaldo constante. A mis amigos y conocidos, que me han acompañado en este camino con su presencia, ánimo y palabras de aliento. A mis jefes de prácticas preprofesionales, quienes me enseñaron con dedicación y me ofrecieron su paciencia en mis primeras experiencias laborales, permitiéndome aprender y crecer con confianza en el ejercicio de los principios fundamentales de mi formación profesional.

Luis Daniel Sarabia Guillermo

Este reconocimiento está dirigido a mi familia, lo más valioso que tengo. Agradezco a Dios por darme la fuerza, la salud y la sabiduría para alcanzar esta meta. A mi tío, Roberto Sarango, por encender la chispa de este camino, por creer en mi potencial desde el inicio y motivarme a dar el primer paso, siendo su apoyo el punto de partida de este logro. A mis padres, Flor Jaya y Alex Bravo, por su amor incondicional, por enseñarme con su ejemplo el valor del esfuerzo y por ser mi mayor inspiración, creyendo en mí incluso en los momentos en que yo mismo dudaba. A mi hijo, Marcelo Bravo, por ser mi alegría, mi fuerza diaria y la razón más profunda para seguir adelante; gracias por tu amor y paciencia a lo largo de este proceso. Y a mí mismo, por no rendirme, por levantarme en cada caída y continuar con determinación. A todos ustedes, gracias por ser parte de este logro que llevaré en el corazón por siempre.

Brayan Alexander Bravo Jaya

Autores

AGRADECIMIENTOS

Quiero dejar constancia de mi sincera gratitud a la Universidad Politécnica Salesiana, sede Quito, y en especial a la Facultad de Ingeniería en Telecomunicaciones, por brindarme una formación académica de calidad y acompañarme en este camino hacia la profesionalización. Agradezco profundamente a los docentes que compartieron sus conocimientos con entrega y vocación, y de manera especial al Ing. Luis Germán Oñate Cadena, tutor de este trabajo, por su valiosa guía y compromiso. Extiendo mi agradecimiento a todas las personas e instituciones que colaboraron en la realización de este proyecto, así como al Grupo ASU de Vóley, por ofrecer un espacio de integración, bienestar y crecimiento personal a lo largo de mi vida universitaria.

Luis Daniel Sarabia Guillermo

A Dios, por darme fuerza, sabiduría y esperanza en cada paso de este camino, porque sin Él, este logro no habría sido posible. A mi tío, Roberto Sarango, por ser el impulso que me motivó a comenzar y confiar en mí desde el inicio; este éxito también es tuyo. A mis padres, por su amor incondicional y apoyo constante, que han sido mi base y motor en todo momento. A mi hijo, mi mayor inspiración y luz en los días difíciles, gracias por tu paciencia y amor incondicional. A mi tutor, por su enseñanza, paciencia y dedicación para guiar esta investigación hasta el éxito, y por su admirable calidad humana. Y a mis queridos ingenieros, quienes me formaron con sabiduría y cariño, y me hicieron amar esta carrera; me siento profundamente orgullosa de pertenecer a esta gran familia de las Telecomunicaciones.

Brayan Alexander Bravo Jaya

Autores

INDICE

ÍNDICE DE FIGURAS

RESUMEN

El presente proyecto tiene como finalidad el desarrollo de un sistema digital de filtros para señales de vos utilizando MATLAB-Simulink junto con una Rasberry Pi, con el objetivo de mejorar el laboratorio de Tratamiento de Señales TPS de la universidad, por medio de esta integración se llevara a cabo el procesamiento de archivos de audio o capturas en tiempo real por medio de un micrófono USB, lo cual es de vital importancia para el estudio practico del comportamiento de las señales en el rango de 0 a 4kHz, que corresponde a la voz humana. Esto será de gran utilidad para los estudiantes ya que permite un aprendizaje más dinámico y real en el diseño de filtros como pasa bajos, pasa altos y notch. La señal se procesa mediante bloques en Simulink, los cuales son cargados en la Raspberry Pi para su ejecución directa, además se puede visualizar la señal mediante bloques gráficos como Scope y analizar su espectro. Los filtros se programan por medio de diagramas funcionales lo cual hace más intuitivo el trabajo, adicionalmente se puede escuchar la salida del sistema en tiempo real mediante el uso del bloque alsa playback conectado a la placa. El control y modificación de los parámetros de los filtros esta al alcance de los usuarios con conocimiento básico de MATLAB, lo cual hace que este sistema sea flexible, educativo y de bajo costo para el aprendizaje en el área de procesamiento de señales de voz.

ABSTRACT

This project aims to develop a digital filter system for voice signals using MATLAB-Simulink together with a Rasberry Pi, in order to improve the TPS Signal Processing laboratory of the university, through this integration will be carried out the processing of audio files or captures in real time via a USB microphone, which is vital for the practical study of the behavior of signals in the range of 0 to 4kHz, which corresponds to the human voice. This will be very useful for students as it allows a more dynamic and real learning in the design of filters such as low-pass, high-pass and notch. The signal is processed through blocks in Simulink, which are loaded into the Raspberry Pi for direct execution, and the signal can be visualized through graphical blocks such as Scope and analyze its spectrum. The filters are programmed by means of functional diagrams which makes the work more intuitive, additionally you can listen to the system output in real time using the alsa playback block connected to the board. The control and modification of the filter parameters is within the reach of users with basic knowledge of MATLAB, which makes this system flexible, educational and low cost for learning in the area of voice signal processing.

INTRODUCCIÓN

Este proyecto muestra el desarrollo de modelos de filtros para el procesamiento de señales de voz en el laboratorio de Tratamiento de Señales y Procesamiento (TPS), utilizando la minicomputadora Raspberry Pi y el programa MATLAB-Simulink. Actualmente, no existe una opción asequible que permita a los docentes y alumnos a desarrollar y comprender el funcionamiento de varios filtros digitales por lo que es necesario este desarrollo, para entender de forma teórica y práctica el funcionamiento y procesamiento de las señales, esto permitirá aprender de mejor manera e incentivar la creación de nuevos equipos que ayuden al aprendizaje. Este trabajo también tiene como objetivo fomentar el uso de herramientas modernas que están disponibles para todos los estudiantes sin necesidad de recurrir a equipos costosos que a veces no están disponibles o son muy difíciles de conseguir o manejar. Al desarrollar este proyecto se espera que se pueda demostrar que es posible implementar varios tipos de filtros digitales como los FIR, IIR y adaptativos, usando herramientas accesibles como la Raspberry Pi, además se hace pruebas para confirmar que los filtros funcionan bien en tiempo real consiguiendo un aprendizaje completo. A través del uso de Simulink se logró realizar varios modelos del sistema que ayuda a entender mejor la lógica y comportamiento de los filtros en las señales de voz, además con este proyecto se da paso a que más personas puedan innovar y crear nuevos sistemas usando la base que aquí se implementa.

CAPITULO 1
ANTECEDENTES

1.1 Planteamiento del problema

En el laboratorio de Tratamiento de Señales y Procesamiento (TPS), se dispone de equipos de alto costo en especial de la marca SIGEX que son reconocidos por su precisión en la emulación de filtros digitales, sin embargo, se debe innovar con nuevas herramientas accesibles y flexibles desarrolladas dentro de las aulas, permitiendo a los estudiantes y docentes diseñar, implementar y probar diversos tipos de filtros digitales de manera práctica y económica para señales en el rango de la voz humana. (Emona Instruments Pty Ltd., 2025)

El uso de plataformas como MATLAB y Simulink permite el diseño y simulación de filtros digitales. No obstante, la falta de integración con hardware accesible, como la Raspberry Pi, dificulta la transición de la teoría a la práctica, lo que impide adquirir experiencia en el desarrollo de filtros digitales en dispositivos embebidos, restringiendo su aprendizaje a entornos de emulación, sin posibilidad de desarrollar y probar nuevos tipos de filtros. (MathWorks, 2025d)

La Raspberry Pi debido a su bajo costo y versatilidad, se ha vuelto como una herramienta educativa eficaz en la enseñanza de sistemas embebidos y procesamiento de señales. (MathWorks, 2025b) Su compatibilidad con Matlab y Simulink ofrece la posibilidad de diseñar, simular y desplegar filtros digitales de manera inmediata mediante el desarrollo de algoritmos que pueden ejecutarse de forma autónoma en este hardware, facilitando la adquisición y procesamiento de señales en tiempo real, lo que la vuelve ideal para que los estudiantes y docentes desarrollen habilidades prácticas enfocadas en el estudio de filtros. (MathWorks, 2025c)

1.2 Justificación

El desarrollo de filtros digitales es una competencia esencial en el procesamiento de señales digitales, ya que permite mejorar la calidad, eficiencia y adaptabilidad de los sistemas de comunicación y su implementación en hardware permite validar su funcionamiento más allá de la simulación. Actualmente, el laboratorio de Tratamiento de Señales y Procesamiento (TPS) cuenta con emuladores de alto costo, pero carece de herramientas accesibles que permitan a los estudiantes diseñar e implementar diversos filtros digitales en hardware real.

El uso de Raspberry Pi en combinación con MATLAB-Simulink ofrece una alternativa de bajo costo, alta flexibilidad y gran potencial educativo enfocado al diseño e implementación de filtros digitales en procesamiento de voz. Esta integración permitirá a los estudiantes tanto crear, simular e implementar filtros digitales en tiempo real lo que fomentará el aprendizaje activo y el desarrollo de proyectos aplicados al procesamiento de señales y electrónica. (MathWorks, 2025e)

En conjunto el uso de software y hardware ayudará a ampliar las herramientas de aprendizaje, facilitando la experimentación con diferentes tipos de filtros digitales, fomentando la integración entre teoría y práctica, permitiendo a los estudiantes implementar variedad de filtros digitales y validar su funcionamiento en tiempo real. Promoviendo la innovación en el laboratorio TPS, al introducir una solución flexible y escalable que pueda adaptarse a diferentes necesidades de investigación y formación.(MathWorks, 2025a)

1.3 Objetivos

1.3.1 Objetivo General

- Desarrollar un equipo en base de Raspberry Pi y MATLAB-Simulink que permita la innovación en el diseño e implementación de diversos filtros digitales para procesamiento de voz en el laboratorio de Tratamiento de Señales y Procesamiento (TPS).

1.3.2 Objetivos Específicos

- Analizar los principios y técnicas de diseño de filtros digitales para procesamiento de voz, así como su implementación en plataformas embebidas.
- Desarrollar diferentes tipos de filtros digitales como paso bajo, paso alto, paso banda y rechazo de banda que sean aplicados a señales de voz.
- Evaluar el rendimiento de los filtros digitales diseñados mediante simulaciones en MATLAB-Simulink y pruebas en hardware con la raspberry pi.
- Realizar el análisis de costos para una futura implementación como producto de venta al público.

1.4 Metodología

Paradigma de Investigación: Positivista

Tipo de Investigación: Experimental

Alcance: Descriptivo

Unidad de Análisis: Correcto funcionamiento de los filtros digitales de procesamiento de voz implementados en Raspberry Pi mediante MATLAB-Simulink.

CAPÍTULO 2
FUNDAMENTACIÓN TEÓRICA

En este capítulo se detalla las fórmulas y fundamentos principales utilizados para diseñar cada uno de los siguientes filtros.

2.1 Filtros Digitales FIR:

Son filtros que poseen una respuesta finita al impulso, es decir, su salida depende únicamente de valores pasados de la entrada. Estos filtros son intrínsecamente estables y presentan una fase lineal, característica valiosa en aplicaciones de voz y audio. Su diseño suele realizarse mediante técnicas como la ventana de Hamming, diseño por frecuencia óptima o el algoritmo Parks-McClellan.

En los filtros FIR utilizados en este proyecto se establecieron parámetros comunes con el fin de mantener coherencia en la simulación y facilitar la comparación de resultados, por lo que en todos los casos se empleó una frecuencia de muestreo (fs) de 8000 Hz la cual supera el mínimo requerido según el teorema de Nyquist, lo que asegura la correcta representación de la señal de voz en el rango de 300 Hz a 3400 Hz como se observa en la ecuación (2.1)

$$fs \geq 2(f_{máxima}) \tag{2.1}$$

$$fs \geq 2(3400)$$

$$fs \geq 6800\ [Hz]\ mínimo$$

Además, se utilizó la ventana de Hamming en cada diseño debido que, al momento de aplicar un corte abrupto en el dominio de la frecuencia, se pueden producir ondas no deseadas llamadas “lobos” o ripples por lo que la ventana de haming ayuda a reducir dichos ripples fuera de la banda de paso con el fin de suavizar la transición entre bandas mejorando el comportamiento del filtro.

Longitud de Filtro (N):

Para calcular la longitud del filtro se debe utilizar las fórmulas 2.2 y 2.3.

$$\Delta f = \frac{|f_{STOP} - f_{PASS}|}{fs} \tag{2.2}$$

$$\Delta f = Ancho\ de\ Banda\ de\ transición\ normalizada$$

$$N_{mínimo} = \frac{3.3}{\Delta f} \quad \textbf{(2.3)}$$

Índice Central (M):

Para calcular el índice central se debe utilizar la fórmula 2.4.

$$M = \frac{N-1}{2} \quad \textbf{(2.4)}$$

Frecuencias de Corte Normalizada (Vc):

Calcula la frecuencia de corte normalizada con la fórmula 2.5.

$$Vc = \frac{2\pi fc}{fs} \quad \textbf{(2.5)}$$

Ventana de Hamming (w[n]):

Según Tan (2008), p.229, en la sección Window Method, indica la siguiente fórmula (2.7) para obtener w[n]. w(n) 0.54-0.46cos(2πn/M)

$$w[n] = 0.54 - 0\ .46\cos\left(\frac{2\pi n}{M}\right) \quad \textbf{(2.6)}$$

Formula desplazada:

$$w[n] = 0.54 - 0\ .46\cos\left(\frac{2\pi n}{N-1}\right) \quad \textbf{(2.7)}$$

Coeficientes de Filtro (b[n]):

Según Tan (2008), p.229, en la sección Window Method, indica la siguiente fórmula (2.8) para obtener b[n].

$$b[n] = h[n] * w[n] \quad \textbf{(2.8)}$$

Respuesta Impulsiva Ideal [h(n)]:

De acuerdo con Tan (2008, tabla 7.1) , cada filtro FIR tiene su fórmula específica para calcular la respuesta impulsiva ideal dependiendo del requerimiento del filtro como se observa en la siguiente Tabla.

Tabla 2.1. Fórmulas de h(n)

Tipo de Filtro	N° Ecuación	H[n] Original	N° Ecuación	H[n] Desplazado
Pasa Bajos	2.9	$h(n) = \begin{cases} \frac{Vc}{\pi} & n = 0 \\ \frac{\sin(Vc * n)}{n * \pi} & n > 0 \end{cases}$	2.10	$h(n) = \begin{cases} \frac{Vc}{\pi} & n = 0 \\ \frac{\sin(Vc(n-M))}{\pi(n-M)} & n > 0 \end{cases}$
Pasa Altos	2.11	$h(n) = \begin{cases} \frac{\pi - Vc}{\pi} & n = 0 \\ -\frac{\sin(Vc * n)}{n * \pi} & n > 0 \end{cases}$	2.12	$h(n) = \begin{cases} \frac{\pi - Vc}{\pi} & n = 0 \\ -\frac{\sin(Vc(n-M))}{\pi(n-M)} & n > 0 \end{cases}$

Pasa Banda	2.13	$h(n) = \begin{cases} \frac{Vc2 - Vc1}{\pi} & n = 0 \\ \frac{\sin(Vc2 * n)}{n * \pi} - \frac{\sin(Vc1 * n)}{n * \pi} & n > 0 \end{cases}$	2.14	$h(n) = \begin{cases} \frac{Vc2 - Vc1}{\pi} & n = 0 \\ \frac{\sin(Vc2 * (n - M)) - \sin(Vc1 * (n - M))}{(n - M) * \pi} & n > 0 \end{cases}$
Rechaza Banda	2.15	$h(n) = \begin{cases} \frac{\pi - Vc2 + Vc1}{\pi} & n = 0 \\ -\frac{\sin(Vc2 * n)}{n * \pi} + \frac{\sin(Vc1 * n)}{n * \pi} & n > 0 \end{cases}$	2.16	$h(n) = \begin{cases} \frac{\pi - Vc}{\pi} & n = 0 \\ \frac{\sin(Vc1 * (n - M)) - \sin(Vc2 * (n - M))}{(n - M) * \pi} & n > 0 \end{cases}$

Elaborado por: Luis Daniel Sarabia Guillermo

2.2 Filtros Digitales IIR:

Son filtros que presentan una respuesta infinita al impulso, ya que su salida depende tanto de la entrada como de salidas pasadas. Son más eficientes en términos computacionales y requieren menos coeficientes para una misma especificación que un filtro FIR. Sin embargo, su diseño requiere especial cuidado para garantizar la estabilidad. Métodos como Butterworth, Chebyshev y Elípticos son comúnmente utilizados en su diseño.

En los filtros IIR utilizados en este proyecto se establecieron parámetros comunes con el fin de mantener coherencia en la simulación y facilitar la comparación de resultados, por lo que en todos los casos se empleó una frecuencia de muestreo (fs) de 8000 Hz al igual que en los filtros FIR supera el mínimo según el teorema de Nyquist de la ecuación 2.1.

Además, se utilizó un filtro IIR con el método de Butterworth, ya que este tipo de filtro tiene una respuesta suave, sin ondulaciones, y permite una transición más limpia entre las frecuencias que se quieren dejar pasar y las que se quieren eliminar, para esto al final del paso de Diseño de Filtro de Forma Analógica se debe realizar la transformación bilineal que en este caso se encuentra implícita, ya que Matlab-Simulink ya lo realiza para permitir que la respuesta en frecuencia se traslade correctamente al dominio digital, manteniendo estabilidad y las frecuencias exactas del diseño.

Atenuación Permitida en la Banda de Paso en Filtros IIR:

Este es un límite de atenuación que tendrán las frecuencias aceptadas.

$$dB = 20log_{10}\left(\frac{Atenuación\ de\ Salida}{Atenuación\ de\ Entrada}\right) \quad \textbf{(2.17)}$$

Aplicando -3 dB debido a que en la realidad siempre existen pérdidas:

$$-3 = 20log_{10}\left(\frac{Atenuación\ de\ Salida}{Atenuación\ de\ Entrada}\right)$$

$$\frac{Atenuación\ de\ Salida}{tenuación\ de\ Entrada} = 10^{\frac{-3}{20}} = 0{,}707$$

$$0{,}707(100) = 70{,}79\%$$

Esto significa que la amplitud de la señal para frecuencias aceptadas se mantiene al 70,79% de su valor original, es decir una pérdida del 29,21% permitido dentro de la banda de paso, además las siguientes variables se usarán para todos los filtros.

$$As = 20dB$$

$$Ap = 3dB$$

As: Es la atenuación deseada en frecuencias fuera del rango necesario.

Ap: Es la atenuación deseada en frecuencias dentro de la banda de paso.

Pre-Warping:

Este método es usado para compensar la distorsión de las frecuencias, sobre todo para que las frecuencias importantes se queden correctamente posicionadas.

$Transformación\ Bilineal = \Omega$

$$\Omega = \tan\left(\frac{\pi * fc}{fs}\right) \quad \textbf{(2.17)}$$

Orden de Filtro:

Para cada filtro IIR según Tan (2008, tabla 8.6) , se utiliza la siguientes formulas y constantes de la tabla.

Tabla 2.2. Fórmulas de Orden.

Tipo de Filtro	N°	Contantes	N° Ecuación	Ecuación
Pasa Bajos	2.18	$Vp = 1\ ;\ Vs = \frac{\Omega_s}{\Omega_p}$	2.19	$n \geq \frac{Log_{10}\left(\frac{10^{\frac{As}{10}} - 1}{10^{\frac{As}{10}} - 1}\right)}{2 * Log_{10}\left(\frac{\Omega_s}{\Omega_p}\right)}$
Pasa Altos	2.20	$Vp = 1\ ;\ Vs = \frac{\Omega_p}{\Omega_s}$	2.21	$n \geq \frac{Log_{10}\left(\frac{10^{\frac{As}{10}} - 1}{10^{\frac{As}{10}} - 1}\right)}{2 * Log_{10}\left(\frac{\Omega_p}{\Omega_s}\right)}$
Pasa Banda	2.22	$Vp = 1\ ;\ Vs = \frac{\Omega_{s2} - \Omega_{s1}}{\Omega_{p2} - \Omega_{p1}}$	2.23	$n \geq \frac{Log_{10}\left(\frac{10^{\frac{As}{10}} - 1}{10^{\frac{As}{10}} - 1}\right)}{2 * Log_{10}\left(\frac{\Omega_{s2} - \Omega_{s1}}{\Omega_{p2} - \Omega_{p1}}\right)}$
Rechaza Banda	2.24	$Vp = 1\ ;\ Vs = \frac{\Omega_{p2} - \Omega_{p1}}{\Omega_{s2} - \Omega_{s1}}$	2.25	$n \geq \frac{Log_{10}\left(\frac{10^{\frac{As}{10}} - 1}{10^{\frac{As}{10}} - 1}\right)}{2 * Log_{10}\left(\frac{\Omega_{p2} - \Omega_{p1}}{\Omega_{s2} - \Omega_{s1}}\right)}$

Elaborado por: Luis Daniel Sarabia Guillermo

Diseño de Filtro Forma Analógica:

La ecuación (2.26) fue tomada de la ecuación (8.25) presentada por Tan (2008, p.321)

, mientras que la ecuación (2.27) corresponde a la ecuación (8.28) del mismo libro.

$$|Hp(V)| = \frac{1}{\sqrt{1+\varepsilon^2 * Vs^{2n}}} \quad \textbf{(2.26)}$$

$$\varepsilon^2 = 10^{0,1*Ap} - 1 \quad \textbf{(2.27)}$$

$$\Omega_c = \Omega_p \quad \textbf{(2.28)}$$

La ecuación (2.28) es debido a que Butterworth define como frecuencia de corte analógica del prototipo en el punto donde es -3dB, es quiere decir en 1000 Hz.

Finalmente, la ecuación para conocer cuanto atenúa el filtro en frecuencias no deseadas es la siguiente:

$$A = -20\, Log_{10}(|Hp(V)|) \quad \textbf{(2.29)}$$

2.3 Filtro Adaptativo:

Parámetros de Filtro NLMS:

Según Farhang-Boroujeny (2013, p.496), ũ y Ψ tienen que ser valores positivos.

"u ans Ψ are positive constants"

(Farhang-Boroujeny, 2013)

NLMS: incluye una variación al normalizar automáticamente el paso de la función, por esto el rango de adaptación o u tiene un rango de estabilidad más alto cumpliendo la siguiente condición.

$$0 < \tilde{u} < 2;\ \Psi = 0{,}00001 \text{ ó } 10^{-6} \quad \textbf{(2.18)}$$

ũ = Controla la rapidez de adaptación del filtro siendo:

Tabla 2.3. Valores Filtro FIR Pasa Bajos MATLAB

ũ	**Velocidad de Aprendizaje**	**Estabilidad**	**En la Práctica**
0 a 0.1	Aprende Lento	Muy Estable	Recomendado en señales con ruido.
0.1 a 1	Moderado a rápido	Estable	Buena convergencia sin riesgo
1 a 2	Muy rápido	Riesgo de Inestabilidad	Usad para adaptación rápida
> 2	Muy rápido	Inestable	No recomendado

Elaborado por: Luis Daniel Sarabia Guillermo

Ψ = Este valor "psi" tiene el propósito de evitar divisiones por cero en la formula general de NLMS, por esta razón tiene que ser un valor cercano no igual a cero para evita afectar al cálculo.

Coeficientes del Filtro Adaptativo NLMS w[n]:

w[n]: Es el aprendizaje que ha tenido el filtro hasta ese punto, debido a que al principio

no ha aprendido nada tendremos los siguientes valores.

$w^T[n]$: Es la transpuesta del vector w[n]

$$w(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} ; \ w^T(0) = [0; 0; 0; 0]$$

Vector de Señal de Entrada:

$$x(n) = [x(n), x(n-1); \dots \dots]$$

Este vector tiene variables randómicas, ya que entra desde el micrófono, por esta razón no se puede tener valores fijos.

Señal Deseada:

Este tipo de señal es una señal continua que se representa de la siguiente manera.

$$d(t) = Asin(2\pi ft + \theta)$$

A = Amplitud

f = frecuencia [Hz]

t = tiempo continuo

θ = Fase

$$d(t) = Asin(wt + \theta)$$

Esta señal continua se debe transformar a señal discreta para que los dispositivos usados puedan procesarla, esta tiene que ser variable en el tiempo discreto, esto quiere decir que debe tener variables en instantes específicos por lo que:

$$t = n * Ts$$

$$Ts = \frac{1}{fs}$$

Obteniendo la siguiente fórmula:

$$d(n) = \sin\left(\frac{2\pi * f * n}{fs}\right) \quad \textbf{(2.19)}$$

Según la según Farhang-Boroujeny (2013), en la sección Summary of Normalized LMS, se muestra las siguientes fórmulas a usar ene este filtro.

- **Salida del Filtro:**

$$y(n) = w^T(n).x(n) \quad \textbf{(2.20)}$$

- **Cálculo de Error:**

$$e(n) = d(n) - y(n) \quad \textbf{(2.21)}$$

- **Implementación del Filtro:**

$$w(n+1) = w(n) + \frac{\tilde{u} * e(n) * x(n)}{x^T(n) * x(n) + \Psi} \quad \textbf{(2.22)}$$

2.4 Errores en Medición y Aproximación:

2.4.1. Error Abosluto:

El error absoluto $E_{\propto}$ es definido por Bhatt como "the differrence between the true and the approximate value. ie. absolute error $E_{\propto} = True\ Value - Aproximate\ Value$"(Bhatt, 2022).

$$E_{\propto} = Error\ verdadero - Valor\ Aproximado \tag{2.23}$$

2.4.2. Error Relativo:

El cuál es la diferencia entre el valor verdadero y el valor aproximado, en cambio el error relativo es definido por Bhatt como "absolute error divided by its true value. Ie. Relative Error $Er = \frac{Absolute\ error}{True\ value} = \frac{E_{\propto}}{True\ Valor}$"(Bhatt, 2022).

$$Er = \frac{Error\ Absoluto}{Valor\ verdadero} = \frac{E_{\propto}}{Valor\ verdadero} \tag{2.24}$$

Al obtener este error no es sencillo de interpretar si existe un error grande o pequeño por lo que este valor es más fácil de analizar si este se encuentra en porcentaje tal cual Bhatt define como "The relative percentage error of a number is defined as the relative error miltiplies by 100 with the signo of %"(Bhatt, 2022) ,dando la siguiente formula.

$$Er(\%) = Er * 100 \tag{2.25}$$

A partir de estas bases teóricas para cada filtro, en el capítulo 3 se presenta la implementación computacional en MATLAB/numérica de los filtros descritos, y en el capítulo 4 su validación práctica mediante pruebas en tiempo real en Symulink.

CAPÍTULO 3

Diseño del Sistema

En este capítulo se detalla el diseño de los filtros digitales aplicados al procesamiento de señales de voz, utilizando MATLAB. Se desarrolló filtros FIR, IIR y un filtro adaptativo LMS, los cuales van a ser implementados en software basado en los cálculos con las ecuaciones de la sección **2.2 Filtros Digitales FIR.**

Tabla 3.1. Valores de N, M y Vc

FILTRO	fc	fstop	fpass	Δf	N	M	Vc
FIR PASA BAJO	2000	2500	2000	0.0625	52.8	26	0.5π
FIR PASA ALTO	1000	500	1000	0.0625	52.8	26	0.25π
FIR PASA BANDA 1er CORTE	1000	500	1000	0.0625	52.8	26	0.25π
FIR PASA BANDA 2do CORTE	2000	2500	2000	0.0625	52.8	26	0.5 π
FIR RECHAZA BANDA 1er CORTE	500	500	0	0.0625	52.8	26	0.0875π
FIR RECHAZA BANDA 2do CORTE	700	700	1200	0.0625	52.8	26	0.2125π

Elaborado por: Luis Daniel Sarabia Guillermo

En la Tabla 3.1 se detallan los valores fc (Frecuencia de corte), fstop (frecuencia de paro), fpass (frecuencia de paso) y mediante cálculos en Matlab con las fórmulas de la sección 2.2, el resultado de las variables Δf, N, M, Vc.

3.2 Diseño Filtro FIR Pasa Bajos:

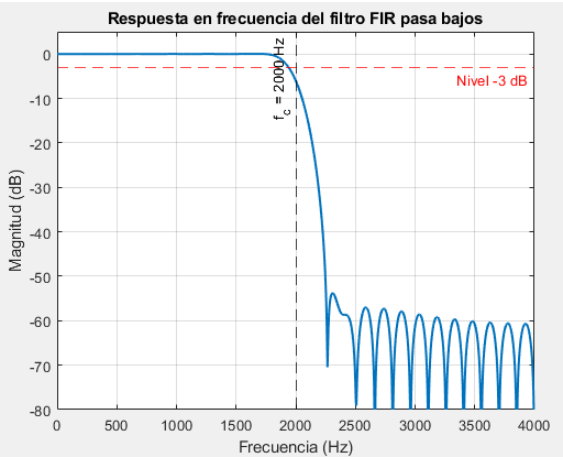


Fig.3.2. Filtro FIR Pasa Bajos

Se diseñó un filtro FIR pasa bajos con una frecuencia de corte de 2000Hz los resultados de los cálculos se describen a continuación.

a) Respuesta Impulsiva Ideal [h(n)]:

Se utilizo la ecuación 2.10 para obtener los resultados que se encuentran en la Tabla 3.2 y en la figura 3.3.

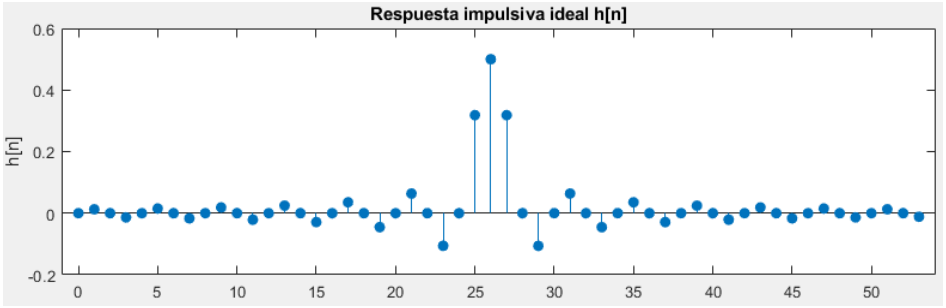


Fig.3.3. h[n] FIR Pasa Bajos

La figura 3.3 muestra la respuesta al impulso, la cual tiene forma de onda donde el valor n = 26 es el más alto, por lo tanto, el punto donde se deja pasar más la señal. En cambio, en los extremos n=0 y n=52, los valores son muy bajos y casi no aportan nada.

b) Ventana de Hamming w[n]:

La ventana de Hamming se calcula en base a la ecuación (2.7) para cada n, los resultados de estos cálculos se encuentran en la Tabla 3.2 y en la figura 3.4.

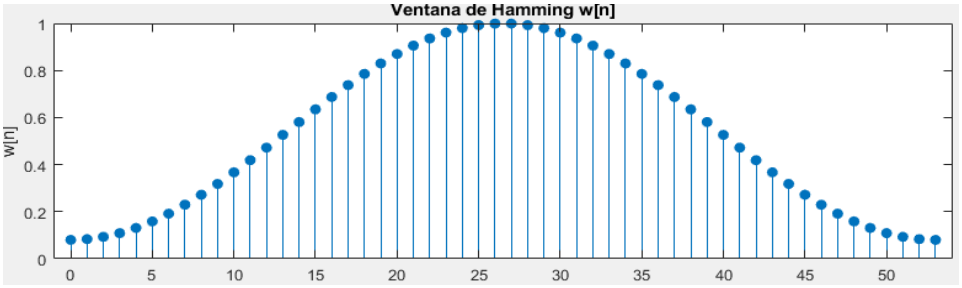


Fig.3.4. w[n] FIR Pasa Bajos

La figura 3.4 indica la ventana de Hamming la cual va de 0,08 hasta 1, como se observa suaviza el filtro y evita saltos bruscos, teniendo más importancia el centro del filtro y menor los extremos.

c) Coeficientes del Filtro b[n]:

Los Coeficientes del Filtro se calcula en base a la ecuación (2.8) para cada n, los resultados se encuentran en la Tabla 3.2 y en la figura 3.5.

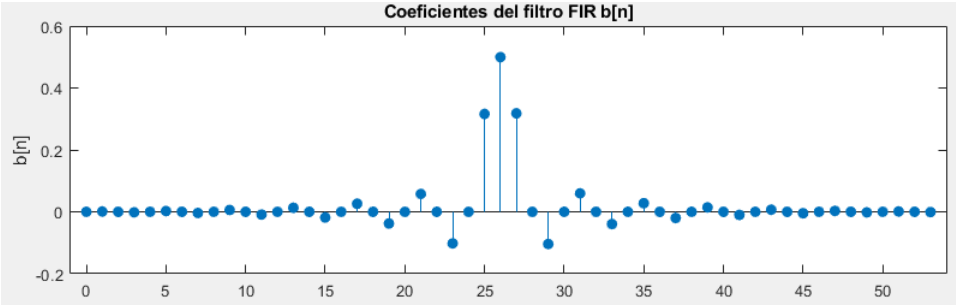


Fig.3.5. b[n] FIR Pasa Bajos

La figura 3.5 muestra el resultado de la señal multiplicada por la ventana de Haming, teniendo una gráfica donde se le da más importancia al centro y nada a los extremos, por esta razón se puede ver que estos tienen menos variaciones a comparación con la figura 3.3.

Tabla 3.2. Valores Filtro FIR Pasa Bajos MATLAB

n	h[n]	w[n]	b[n]
0	0	0.08	0
1	0.01	0.08	0
2	0	0.09	0
23	-0.11	0.96	-0.1
24	0	0.98	0
25	0.32	0.99	0.32
26	0.5	1	0.5
27	0.32	1	0.32
28	0	0.99	0
29	-0.11	0.98	-0.1
51	0.01	0.09	0
52	0	0.08	0
53	-0.01	0.08	0

Elaborado por: Luis Daniel Sarabia Guillermo

Como se puede observar, los valores presentados en la Tabla 3.3 coinciden con los cálculos teóricos y las gráficas obtenidas, confirmando la correcta implementación del diseño del filtro FIR pasa bajos.

3.3 Diseño Filtro FIR Pasa Altos:

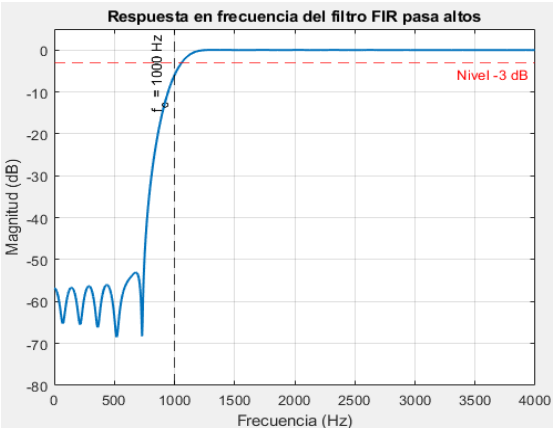


Fig.3.6. Filtro FIR Pasa Altos

Se diseñó un filtro FIR pasa altos con una frecuencia de corte de 1000Hz los resultados de los cálculos se describen a continuación.

a) Respuesta Impulsiva Ideal [h(n)]:

Se utilizo la ecuación 2.12 para obtener los resultados que se encuentran en la Tabla 3.3 y en la figura 3.7.

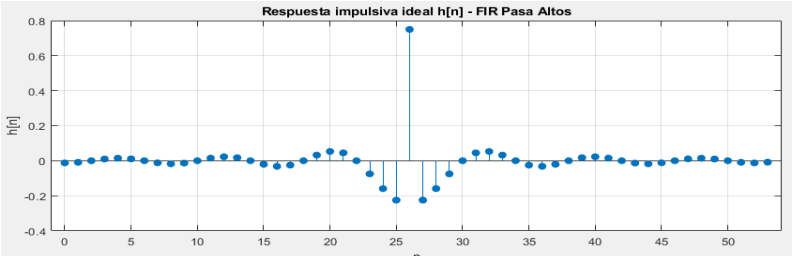


Fig.3.7. h[n] FIR Pasa Altos

La figura 3.7 muestra la respuesta al impulso, donde el valor más alto está en n=26, lo que indica que ahí se deja pasar más la señal. En cambio, en los extremos n=0 y n=52, los valores son muy bajos y casi no aportan nada.

b) Ventana de Hamming w[n]:

La ventana de Hamming se calcula en base a la ecuación (2.7) para cada n, los resultados de estos cálculos se encuentran en la Tabla 3.3 y en la figura 3.8.

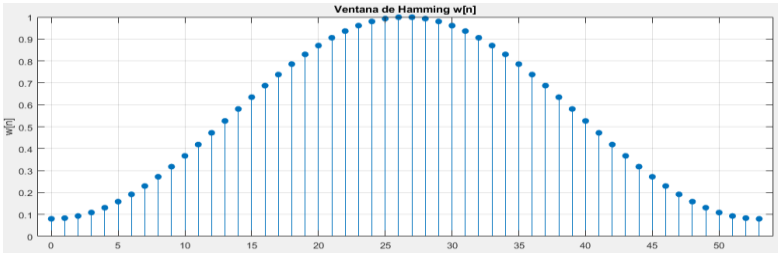


Fig.3.8. w[n] FIR Pasa Altos

La figura 3.8 muestra la ventana de Hamming, que va de 0.08 a 1. Se observa que

suaviza el filtro, dando más peso al centro y menos a los extremos.

c) Coeficientes del Filtro b[n]:

Los Coeficientes del Filtro se calcula en base a la ecuación (2.8) para cada n, los resultados se encuentran en la Tabla 3.3 y en la figura 3.9.

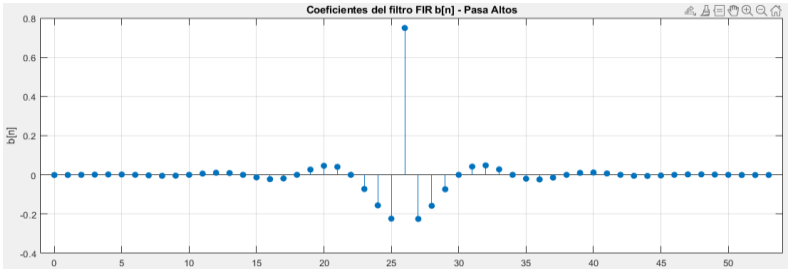


Fig.3.9. b[n] FIR Pasa Bajos

La figura 3.9 muestra el resultado de b[n], que es la señal multiplicada por la ventana de Hamming, dando más importancia al centro y casi nada a los extremos, por eso se nota menos alteraciones en los extremos a comparación de la figura 3.7.

Tabla 3.3. Valores Filtro FIR Pasa Bajos MATLAB

n	**h[n]**	**w[n]**	**b[n]**
0	-0.01	0.08	0.00
1	-0.01	0.08	0.00
2	0.00	0.09	0.00
23	-0.08	0.96	-0.07
24	-0.16	0.98	-0.16
25	-0.23	0.99	-0.22
26	0.75	1.00	0.75
27	-0.23	1.00	-0.22
28	-0.16	0.99	-0.16
29	-0.08	0.98	-0.07
50	0.00	0.11	0.00
51	-0.01	0.09	0.00
52	-0.01	0.08	0.00

Elaborado por: Luis Daniel Sarabia Guillermo

Como se puede observar, los valores presentados en la Tabla 3.3 coinciden con los cálculos teóricos y las gráficas obtenidas, confirmando la correcta implementación del diseño del filtro FIR pasa altos.

3.4 Diseño Filtro FIR Pasa Banda:

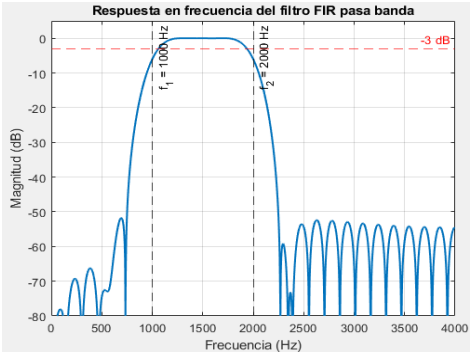


Fig.3.10. Filtro FIR Pasa Banda

Se diseñó un filtro FIR pasa banda con una frecuencia de paso de 1500Hz con cortes en 1000 Hz y 2000 Hz como se muestra en los resultados iniciales en para cada corte en la Tabla 3.1.

a) Respuesta Impulsiva Ideal [h(n)]: (CITAR FORMULA)

Se utilizo la ecuación 2.14 para obtener los resultados que se encuentran en la Tabla 3.4 y en la figura 3.11.

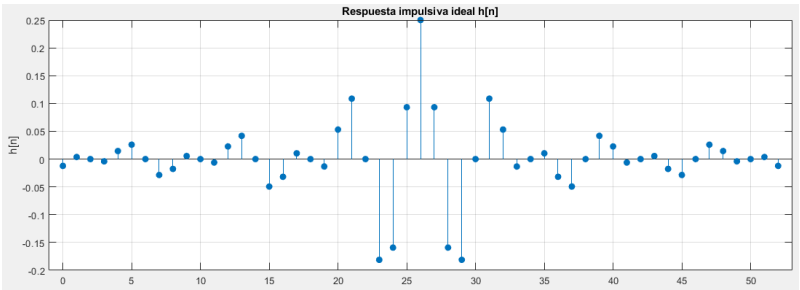


Fig.3.11. h[n] FIR Pasa Banda

La figura 3.11 muestra la respuesta al impulso, donde el valor más alto está en n=26, lo que indica que ahí se deja pasar más la señal. En cambio, en los extremos n=0 y n=52, los valores son muy bajos y casi no aportan nada.

a) Ventana de Hamming w[n]:

La ventana de Hamming se calcula en base a la ecuación (2.7) para cada n, los resultados de estos cálculos se encuentran en la Tabla 3.4 y en la figura 3.12.

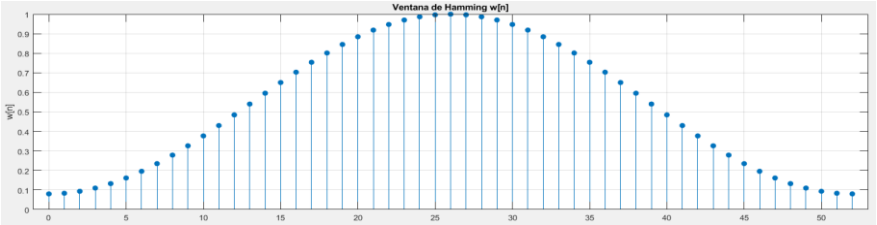


Fig.3.12. w[n] FIR Pasa Banda

La figura 3.12 muestra la ventana de Hamming, que va de 0.08 a 1. Se observa que

suaviza el filtro, dando más peso al centro y menos a los extremos.

b) Coeficientes del Filtro b[n]:

Los Coeficientes del Filtro se calcula en base a la ecuación (2.8) para cada n, los resultados se encuentran en la Tabla 3.4 y en la figura 3.13.

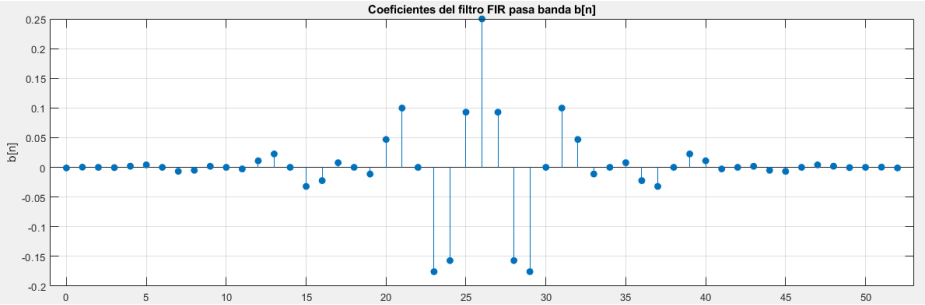


Fig.3.13. b[n] FIR Pasa Banda

La figura 3.13 muestra el resultado de b[n], que es la señal multiplicada por la ventana de Hamming, dando más importancia al centro y casi nada a los extremos, por eso se nota menos alteraciones en los extremos a comparación de la figura 3.11.

Tabla 3.4. Valores Filtro FIR Pasa Bajos MATLAB

n	**h[n]**	**w[n]**	**b[n]**
0	-0.01	0.08	0.00
1	0.00	0.08	0.00
2	0.00	0.09	0.00
23	-0.16	0.99	-0.16
24	0.09	1.00	0.09
25	0.25	1.00	0.25
26	0.09	1.00	0.09
27	-0.16	0.99	-0.16
28	-0.18	0.97	-0.18
29	0.00	0.09	0.00
50	0.00	0.08	0.00
51	-0.01	0.08	0.00
52	-0.01	0.08	0.00

Elaborado por: Luis Daniel Sarabia Guillermo

Como se puede observar, los valores presentados en la Tabla 3.4 coinciden con los cálculos teóricos y las gráficas obtenidas, confirmando la correcta implementación del diseño del filtro FIR pasa banda.

c) Variables en Simulink:

Frecuencia Central (fce)

$$fce = \frac{f1+f2}{2} \tag{3.1}$$

$$fce = \frac{1000 + 2000}{2} = 1500\ Hz$$

Frecuencia Central Normalizada (Vce)

$$Vce = \frac{2\pi * fce}{fs} \tag{3.2}$$

$$Vce = \frac{2\pi * 15000}{8000} = 0.375\pi$$

Ancho de Banda (BW)

$$BW = f2 - f1 \tag{3.3}$$

$$BW = 2000 - 1000 = 1000\ Hz$$

Ancho de Banda Normalizada:

$$Bwn = \frac{BW}{\frac{fs}{2}} \tag{3.4}$$

$$Bwn = \frac{1000}{\frac{8000}{2}} = 0.25$$

3.5 Diseño Filtro FIR Rechaza Banda:

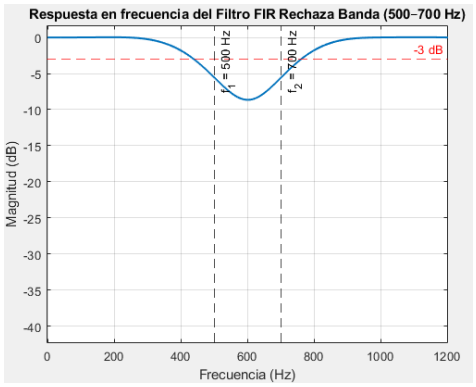


Fig.3.14. Filtro FIR Rechaza Banda

Se diseñó un filtro FIR pasa banda con una frecuencia de rechazo de 600Hz con cortes en 350 Hz y 850 Hz como se muestra en los resultados iniciales en para cada corte en la Tabla 3.1.

b) Respuesta Impulsiva Ideal [h(n)]:

Se utilizo la ecuación 2.16 para obtener los resultados que se encuentran en la Tabla 3.5 y en la figura 3.15.

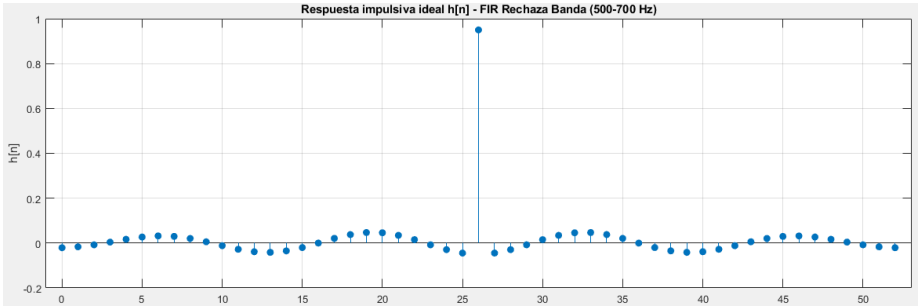


Fig.3.15. h[n] FIR Pasa Banda

La figura 3.15 muestra la respuesta al impulso, donde el valor más alto está en n=26, lo que indica que ahí se deja pasar más la señal. En cambio, en los extremos n=0 y n=52, los valores son muy bajos y casi no aportan nada.

d) Ventana de Hamming w[n]:

La ventana de Hamming se calcula en base a la ecuación (2.7) para cada n, los resultados de estos cálculos se encuentran en la Tabla 3.5 y en la figura 3.16.

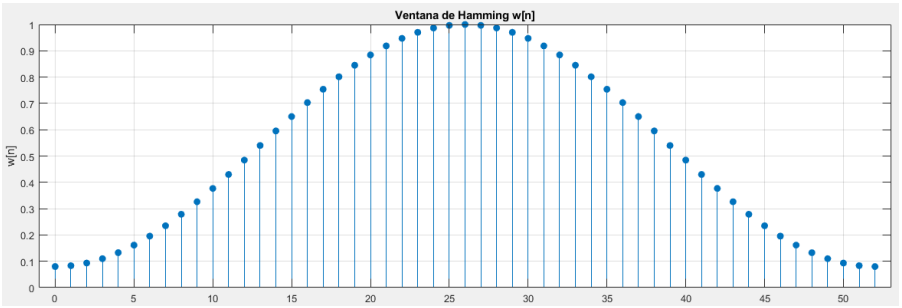


Fig.3.16. w[n] FIR Pasa Banda

La figura 3.16 muestra la ventana de Hamming, que va de 0.08 a 1. Se observa que suaviza el filtro, dando más peso al centro y menos a los extremos.

e) Coeficientes del Filtro b[n]:

Los Coeficientes del Filtro se calcula en base a la ecuación (2.8) para cada n, los resultados se encuentran en la Tabla 3.5 y en la figura 3.17.

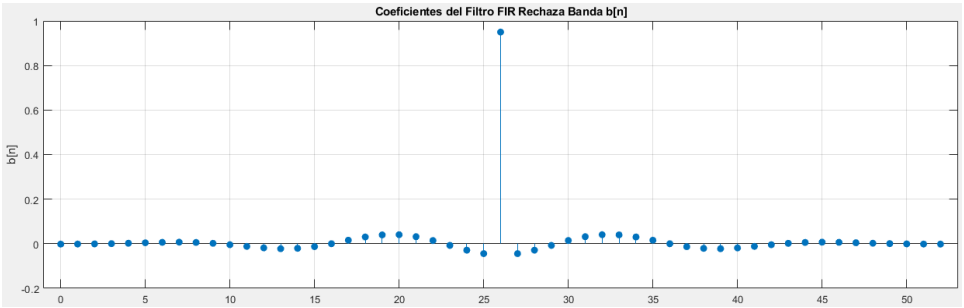


Fig.3.17. b[n] FIR Pasa Banda

La figura 3.17 muestra el resultado de b[n], que es la señal multiplicada por la ventana de Hamming, dando más importancia al centro y casi nada a los extremos, por eso se

nota menos alteraciones en los extremos a comparación de la figura 3.17.

Tabla 3.5. Valores Filtro FIR Pasa Bajos MATLAB

n	h[n]	w[n]	b[n]
0	-0.02	0.08	0.00
1	-0.02	0.08	0.00
2	-0.01	0.09	0.00
23	-0.03	0.99	-0.03
24	-0.04	1.00	-0.04
25	0.95	1.00	0.95
26	-0.04	1.00	-0.04
27	-0.03	0.99	-0.03
28	-0.01	0.97	-0.01
29	-0.01	0.09	0.00
50	-0.02	0.08	0.00
51	-0.02	0.08	0.00
52	-0.02	0.08	0.00

Elaborado por: Luis Daniel Sarabia Guillermo

Como se puede observar, los valores presentados en la Tabla 3.4 coinciden con los cálculos teóricos y las gráficas obtenidas, confirmando la correcta implementación del diseño del filtro FIR pasa banda.

f) Variables en Simulink:

Con la ecuación (3.1) se obtiene el siguiente valor de frecuencia central.

$$fce = \frac{700 + 500}{2} = 600\ Hz$$

Con la ecuación (3.2) se obtiene el siguiente valor de frecuencia central normalizada.

$$Vce = \frac{2\pi * 600}{8000} = 0.15\pi$$

Con la ecuación (3.3) se obtiene en ancho de banda siguiente.

$$BW = 700 - 500 = 200\ Hz$$

Con la ecuación (3.4) se obtiene en ancho de banda normalizada siguiente.

$$Bwn = \frac{200}{\frac{8000}{2}} = 0.05$$

3.6 Diseño Filtro IIR Pasa Bajos:

Se diseñó un filtro IIR pasa bajos con una frecuencia de paso de 1500Hz y de rechazo de 2000Hz, los resultados de los cálculos se describen a continuación.

a) Pre-Warping

Con le ecuación 2.17 se obtiene Ω para fpass y fstop.

Frecuencia de Paso 1500 Hz:

$$\Omega_p = \text{Wap} = \tan\left(\frac{\pi * 1500}{8000}\right) = 0{,}6669$$

Frecuencia de Rechazo 2000 Hz:

$$\Omega_s = \text{Was} = \tan\left(\frac{\pi * 2000}{8000}\right) = 1$$

b) Orden del Filtro:

Remplazar Valores As, Ap, Ω_s y Ω_p en la ecuación 2.19 como se muestra a continuación.

$$n \geq \frac{Log_{10}\left(\frac{10^{\frac{20}{10}} - 1}{10^{\frac{3}{10}} - 1}\right)}{2 * Log_{10}\left(\frac{1}{0{,}6669}\right)}$$

$$\boldsymbol{n \geq 5.67} \text{ mínimo}$$

El valor como mínimo del orden debe ser de 5,67 en este caso se utilizará 6.

c) Diseño de Filtro Forma Analógica:

Remplazar Valores ε^2, Ap, n y Vs en las ecuaciones 2.26, 2.27, 2.28 como se muestra a continuación.

$$\varepsilon^2 = 10^{0{,}1*(3)} - 1 = 0{,}995$$

$$\Omega_c = \Omega_p = 0{,}4142$$

$$|Hp(V)| = \frac{1}{\sqrt{1 + 0{,}995 * \left(\frac{1}{0{,}6669}\right)^{2(6)}}}$$

$$|Hp(V)| = 0{,}087$$

En Decibeles se calcula con la ecuación 2.29.

$$A = -20\, Log_{10}(0{,}087) = 21{,}20\, dB$$

Se verificó que el filtro en frecuencias no deseadas atenúa más de los 20dB deseados.

d) Frecuencia Normalizada:

Con la ecuación (3.2) se obtiene la frecuencia de corte normalizada.

$$Vce = \frac{2\pi * 2000}{8000} = 0.5\pi$$

3.7 Diseño Filtro IIR Pasa Altos:

Se diseñó un filtro IIR pasa alto con una frecuencia de paso de 1500 Hz y de rechazo de 1000 Hz, los resultados de los cálculos se describen a continuación.

a) Pre-Warping

Con la formula (2.17) se realiza pre-warping para la frecuencia de paso y de rechazo.

Frecuencia de Paso 1500 Hz:

$$\Omega_p = \text{Wap} = \tan\left(\frac{\pi * 1500}{8000}\right) = 0{,}6669$$

Frecuencia de Rechazo 1000 Hz:

$$\Omega_s = \text{Was} = \tan\left(\frac{\pi * 1000}{8000}\right) = 0{,}4142$$

b) Orden del Filtro:

Remplazar Valores As, Ap, Ω_s y Ω_p en la ecuación 2.21 como se muestra a continuación.

$$n \geq \frac{Log_{10}\left(\frac{10^{\frac{20}{10}} - 1}{10^{\frac{3}{10}} - 1}\right)}{2 * Log_{10}\left(\frac{0{,}6669}{0{,}4142}\right)}$$

$$\boldsymbol{n \geq 4{,}82} \text{ mínimo}$$

El valor como mínimo del orden debe ser de 4,82 en este caso se utilizará 5.

c) Diseño de Filtro Forma Analógica:

Con las ecuaciones (2.26),(2.27) y (2.28) Remplazar Valores ε^2, Ap, n y Vs:

$$\varepsilon^2 = 10^{0{,}1*(3)} - 1 = 0.9952$$

$$\Omega_c = \Omega_p = 0{,}6669$$

$$|Hp(V)| = \frac{1}{\sqrt{1 + 0.9952 * \left(\frac{0{,}6669}{0{,}4142}\right)^{2(5)}}}$$

$$|Hp(V)| = 0{,}090$$

En Decibeles con la ecuación (2.29):

$$A = -20\, Log_{10}(0{,}090) = 20{,}91\, dB$$

Se verificó que el filtro en frecuencias no deseadas atenúa más de los 20dB deseados.

d) Frecuencia Normalizada:

Con la ecuación (3.2) se obtiene la frecuencia de corte normalizada.

$$Vce = \frac{2\pi * 1000}{8000} = 0.25\pi$$

3.8 Diseño Filtro IIR Pasa Banda:

Se diseñó un filtro IIR pasa banda con una banda de paso de 1000 Hz a 2000 Hz y banda de rechazo de 750 Hz a 2250 Hz, los resultados de los cálculos se describen a continuación.

a) Pre-Warping

Con la formula (2.17) se realiza pre-warping para la frecuencia de paso y de rechazo.

Banda de Paso 1000 Hz a 2000 Hz:

$$\Omega_{p1} = \text{Wap1} = \tan\left(\frac{\pi * 1000}{8000}\right) = 0{,}4142$$

$$\Omega_{p2} = \text{Wap2} = \tan\left(\frac{\pi * 2000}{8000}\right) = 1$$

Banda de Rechazo 750Hz a 2250 Hz:

$$\Omega_{s1} = \text{Was1} = \tan\left(\frac{\pi * 750}{8000}\right) = 0{,}3033$$

$$\Omega_{s2} = \text{Was2} = \tan\left(\frac{\pi * 2250}{8000}\right) = 1{,}2185$$

b) Orden del Filtro:

Remplazar Valores As, Ap, Ω_s y Ω_p en la ecuación 2.23.

$$n \geq \frac{Log_{10}\left(\frac{10^{\frac{20}{10}} - 1}{10^{\frac{3}{10}} - 1}\right)}{2 * Log_{10}\left(\frac{1{,}2485 - 0{,}3033}{1 - 0{,}4142}\right)}$$

$$n \geq \mathbf{5,17\ mínimo}$$

El valor como mínimo del orden debe ser de 5,17 en este caso se utilizará 6.

c) Diseño de Filtro Forma Analógica:

Con las ecuaciones (2.26),(2.27) y (2.28) Remplazar Valores ε^2, Ap, n y Vs:

$$\varepsilon^2 = 10^{0{,}1*(3)} - 1 = 0{,}99$$

$$|Hp(V)| = \frac{1}{\sqrt{1 + 0{,}99 * \left(\frac{1{,}2485 - 0{,}3033}{1 - 0{,}4142}\right)^{2(6)}}}$$

$$|Hp(V)| = 0{,}056$$

En Decibeles con la ecuación (2.29):

$$A = -20\, Log_{10}(0{,}056) = 25{,}03\, dB$$

Se verificó que el filtro en frecuencias no deseadas atenúa más de los 20dB deseados.

d) Frecuencia Normalizada:

Con la ecuación (3.2) se obtiene la frecuencia de corte normalizada.

$$Vce_1 = \frac{2\pi * 1000}{8000} = 0.25\pi$$

$$Vce_2 = \frac{2\pi * 2000}{8000} = 0.5\pi$$

3.9 Diseño Filtro IIR Rechaza Banda:

Se diseñó un filtro IIR pasa banda con una banda de rechazo de 500 Hz a 700 Hz y banda de paso de 450 Hz a 750 Hz, los resultados de los cálculos se describen a continuación.

a) Pre-Warping

Con la formula (2.17) se realiza pre-warping para la frecuencia de paso y de rechazo.

Banda de Paso por debajo de 450 Hz y por encima de 750 Hz:

$$\Omega_{p1} = \text{Wap1} = \tan\left(\frac{\pi * 450}{8000}\right) = 0{,}1785$$

$$\Omega_{p2} = \text{Wap2} = \tan\left(\frac{\pi * 750}{8000}\right) = 0{,}3033$$

Banda de Rechazo de 500 Hz a 700 Hz:

$$\Omega_{s1} = \text{Was1} = \tan\left(\frac{\pi * 500}{8000}\right) = 0{,}3033$$

$$\Omega_{s2} = \text{Was2} = \tan\left(\frac{\pi * 700}{8000}\right) = 0{,}2820$$

b) Orden del Filtro:

Remplazar Valores As, Ap, Ω_s y Ω_p en la ecuación 2.25.

$$n \geq \frac{Log_{10}\left(\frac{10^{\frac{20}{10}}-1}{10^{\frac{3}{10}}-1}\right)}{2 * Log_{10}\left(\frac{0{,}3033-0{,}1785}{0{,}2820-0{,}1989}\right)}$$

$$n \geq \mathbf{5,67\ mínimo}$$

El valor como mínimo del orden debe ser de 5,67 en este caso se utilizará 6.

c) Diseño de Filtro Forma Analógica:

Con las ecuaciones (2.26), (2.27) y (2.28) Remplazar Valores ε^2, Ap, n y Vs:

$$\varepsilon^2 = 10^{0{,}1*(3)} - 1 = 0{,}99$$

$$|Hp(V)| = \frac{1}{\sqrt{1 + 0{,}99 * \left(\frac{0{,}3033-0{,}1785}{0{,}2820-0{,}1989}\right)^{2(6)}}}$$

$$|Hp(V)| = 0{,}0842$$

En Decibeles con la ecuación (2.29):

$$A = -20\ Log_{10}(0{,}0842) = 21{,}49\ dB$$

Se verificó que el filtro en frecuencias no deseadas atenúa más de los 20dB deseados.

d) Frecuencia Normalizada:

Con la ecuación (3.2) se obtiene la frecuencia de corte normalizada.

$$Vce_1 = \frac{2\pi * 500}{8000} = 0.125\pi$$

$$Vce_2 = \frac{2\pi * 700}{8000} = 0.175\pi$$

3.10 Diseño Filtro Selectivo NLMS:

Se diseñó un filtro selectivo con capacidad de dejar pasar la frecuencia que sea necesaria, en este caso de 500 Hz.

a) Cálculo de Orden de Filtro:

Mediante la siguiente ecuación (3.5) se podrá obtener el cálculo del orden mínimo necesario.

$$M = \frac{fs}{f_{señal}} \qquad \textbf{(3.5)}$$

Remplazando fs y $f_{señal} = 500\ Hz$

$$\underline{M = \frac{8000}{500} = 16\ \boldsymbol{mínimo}}$$

A continuación, se va a realizar el ejemplo para 500 Hz con N=4 partiendo de la ecuación (2.19).

$$d(0) = \sin\left(\frac{2\pi * 500 * 0}{8000}\right) = 0$$

$$d(1) = \sin\left(\frac{2\pi * 500 * 1}{8000}\right) = 0{,}382683$$

$$d(2) = \sin\left(\frac{2\pi * 500 * 2}{8000}\right) = 0{,}382683$$

$$d(3) = \sin\left(\frac{2\pi * 500 * 3}{8000}\right) = 0{,}92388$$

b) Ejemplo n=0 del Filtro.

Variables Iniciales:

Ya que los valores de entrada son aleatorios, se va a utilizar los siguientes valores de entrada de orden 4 para poder explicarlo de mejor manera utilizando las siguientes variables de ejemplo.

$$x(0) = [11939, -15578, 8000, -3000]$$

$$x(0) = \begin{bmatrix} 11939 \\ -15578 \\ 8000 \\ -3000 \end{bmatrix};\ w(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix};\ d(0) = 0$$

Salida del Filtro:

Con la ecuación (2.20) se desarrolló la siguiente operación:

$$y(0) = w^T(0).x(0)$$

$$y(0) = [0;0;0;0].\begin{bmatrix} 11939 \\ -15578 \\ 8000 \\ -3000 \end{bmatrix}$$

$$y(0) = 0(11939) + 0(-15578) + 0(8000) + 0(-3000) = 0$$

Cálculo de Error:

Con la ecuación (2.21) se desarrolló la siguiente operación:

$$e(0) = d(0) - y(0)$$

$$e(0) = 0 - 0 = 0$$

Cálculo $\boldsymbol{x^T(n) * x(n)}$:

$$\boldsymbol{x^T(0) * x(0)} = \begin{bmatrix} 11939 \\ -15578 \\ 8000 \\ -3000 \end{bmatrix} * [11939, -15578, 8000, -3000]$$

$$\boldsymbol{x^T(0) * x(0)} = (11939)^2 + (-15578)^2 + (8000)^2 + (3000)^2 = 458213805$$

c) Implementación del Filtro:

Con la ecuación (2.22) se desarrolló la siguiente operación:

$$w(0+1) = w(0) + \frac{\tilde{u} * e(0) * x(0)}{x^T(0) * x(0) + \Psi}$$

$$w(1) = w(0) + \frac{0{,}01 * 0 * x(0)}{458213805 + 10^{-6}}$$

$$w(1) = w(0) + \frac{0 * \begin{bmatrix} 11939 \\ -15578 \\ 8000 \\ -3000 \end{bmatrix}}{458213805 + 10^{-6}}$$

$$w(1) = w(0) + \frac{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}}{458213805 + 10^{-6}}$$

Valor 1 del x (0) = 0

$$\frac{\mathbf{0}}{458213805 + 10^{-6}} = \mathbf{0}$$

Cada valor del vector se debe dividir por separado, en este caso al ser los mismos valores se obtiene un vector de 0, obteniendo el siguiente resultado, además se debe tomar en cuenta que se debe transponer w(0) para poder realizar la operación.

$$w(1) = [0; 0; 0; 0] + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$w(1) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}^T + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Para cada n se debe realizar este proceso remplazando los valores de n por lo que se lo puede desarrollar en Matlab de la siguiente manera:

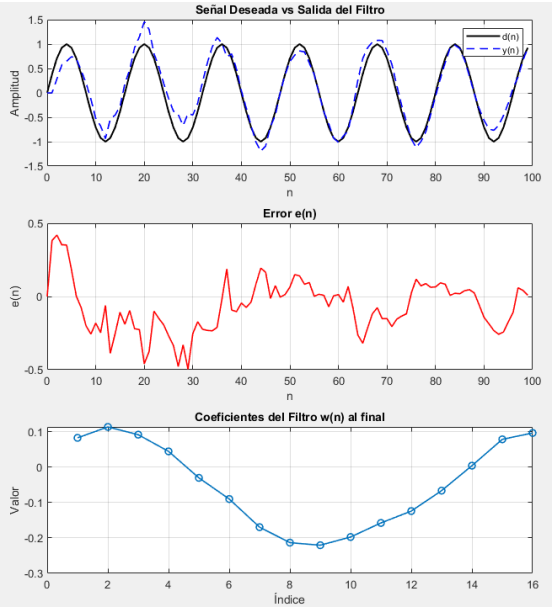


Fig.3.18. Filtro Adaptativo Adaptación Rápida Ideal u = 0,8

En la figura 3.18 con μ=0,8 y N=100, se observa que la salida del filtro se ajusta rápidamente a la señal deseada d(n). El error e(n) presenta cierta irregularidad por las actualizaciones agresivas que genera el valor alto de μ. Los coeficientes adoptan una forma senoidal, ya que se trata de un caso ideal sin presencia de ruido.

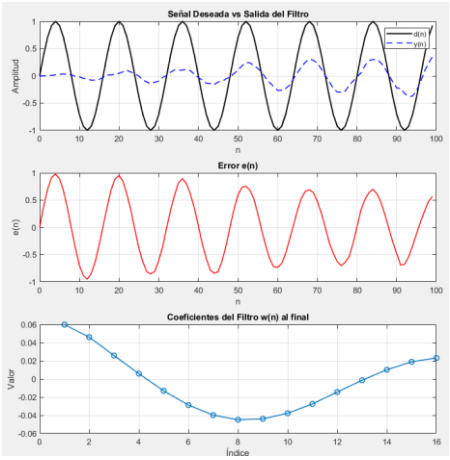


Fig.3.19. Filtro Adaptativo Adaptación Lenta Ideal u = 0,01

En la figura 3.19, con μ=0,01 y N=100, se ve que la salida del filtro no sigue bien a d(n), ya que el valor bajo de μ hace que la adaptación sea muy lenta. El error e(n) se parece mucho a la señal deseada, lo que muestra que y(n) todavía no aprendió casi nada. Los coeficientes apenas se alejan de cero y no forman aún una senoidal. Aunque aprende lento, este valor de μ lo hace muy estable frente al ruido, ideal si se trabaja con señales reales como las de un micrófono.

CAPÍTULO 4
IMPLEMENTACIÓN Y PRUEBAS

En este capítulo se va a mostrar la implementación de cada diseño de los filtros mencionados en el capítulo anterior.

4.1. Modelo Base Utilizado

El modelo base para la implementación de estos filtros parte de este ejemplo:

openExample('raspberrypi/AudioProcessingRaspberryPiExample','supportingFile','raspberrypi_audio_gettingStarted')

Este es un sistema de procesamiento de audio en tiempo real en Raspberry Pi que tiene bloques que se mantendrán en todos los filtros, estos bloques se pueden ver la figura 4.12.

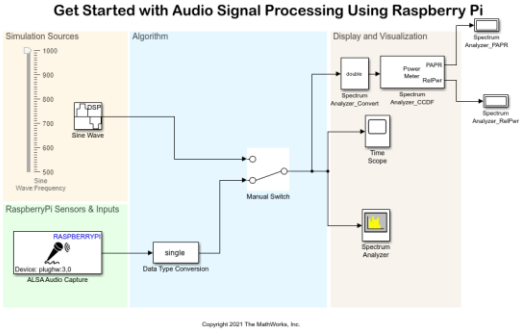


Fig.4.12. Get Started with Audio Signal Processing Using Raspberry Pi

Donde:

- **Audio Capture (ALSA Audio Playback):** Se encarga de capturar el audio en tiempo real desde un micrófono conectado a la Raspberry Pi.
- **Data Type Conversion:** Convierte a un tipo de dato de la señal.
- **Sine Wave:** Genera una señal de onda senoidal es utilizado de prueba.
- **Audio Playback:** Reproduce el audio original o procesado.
- **Manual Switch:** Permite seleccionar manualmente entre dos señales de entrada.
- **Time Scope:** Muestra la señal en tiempo mientras se ejecuta el modelo.
- **Spectrum Analyzer:** Muestra la señal en el dominio de la frecuencia,
- **Digital Filter:** Es el filtro que se va a utilizar dependiendo del modelo.

4.2. Implementación:

4.2.1. Filtro FIR Pasa Bajos:

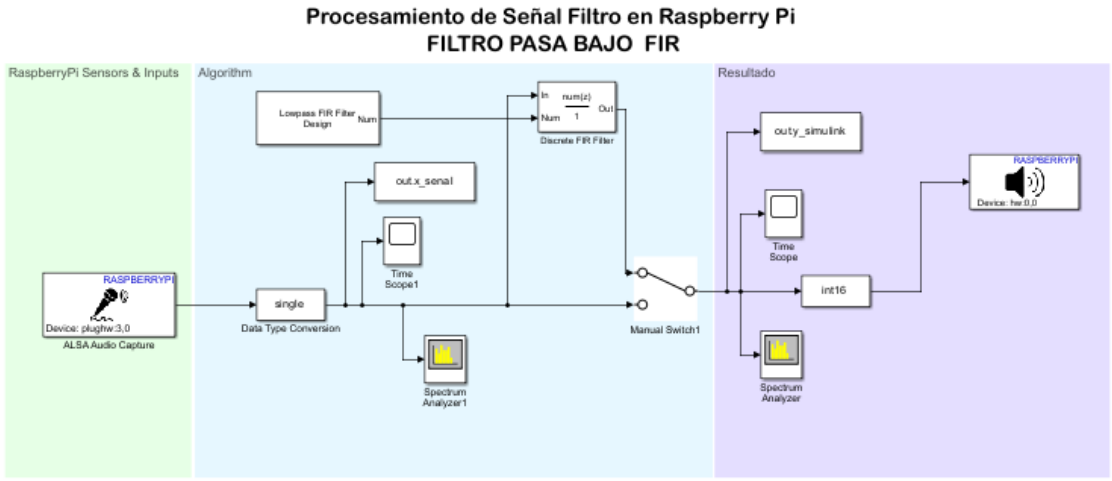


Fig.4.13. Modelo Filtro Pasa Bajo FIR

En la figura 4.13 se puede observar el modelo del filtro pasa bajo con frecuencia de corte en 2000 Hz y sus resultados en tiempo y frecuencia en la figura 4.14.

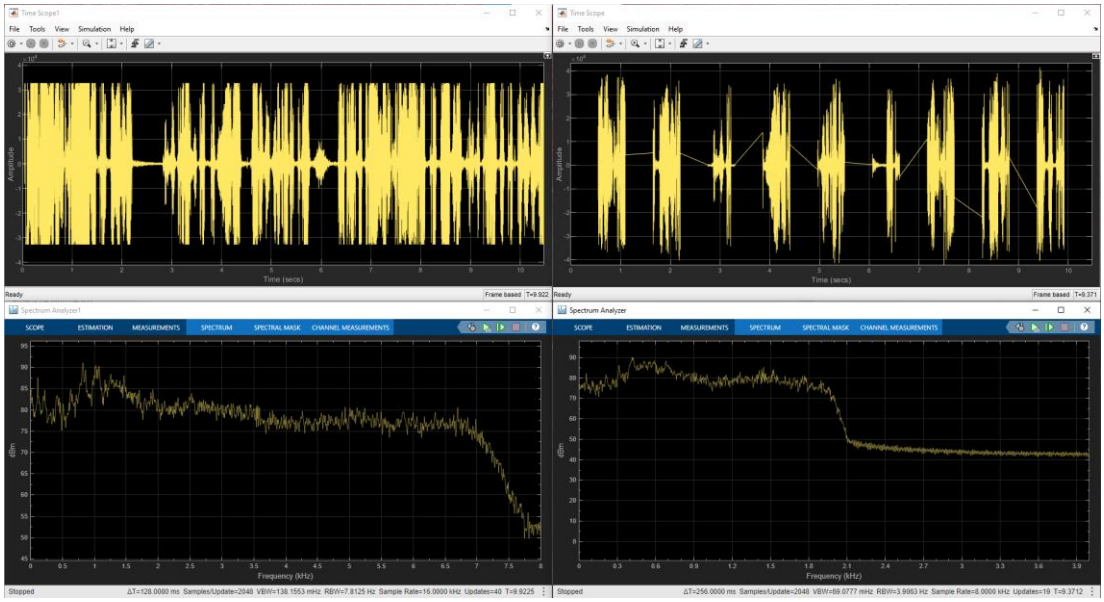


Fig.4.14. Filtro Pasa Bajo FIR antes y después en frecuencia y tiempo.

Los resultados de la fig. 4.14, muestran que el filtro FIR diseñado a un orden N =54 con ventana de Hamming aplicado a una señal de voz real capturada desde un micrófono conectado a la raspberry Pi, atenúa eficazmente las frecuencias superiores al umbral esperado. En el dominio del tiempo la señal filtrada presenta una onda más suavizada, mientras que en el dominio de la frecuencia se observa el corte exacto en 2000 Hz, confirman el funcionamiento correcto de este filtro.

4.2.2. Filtro FIR Pasa Alto:

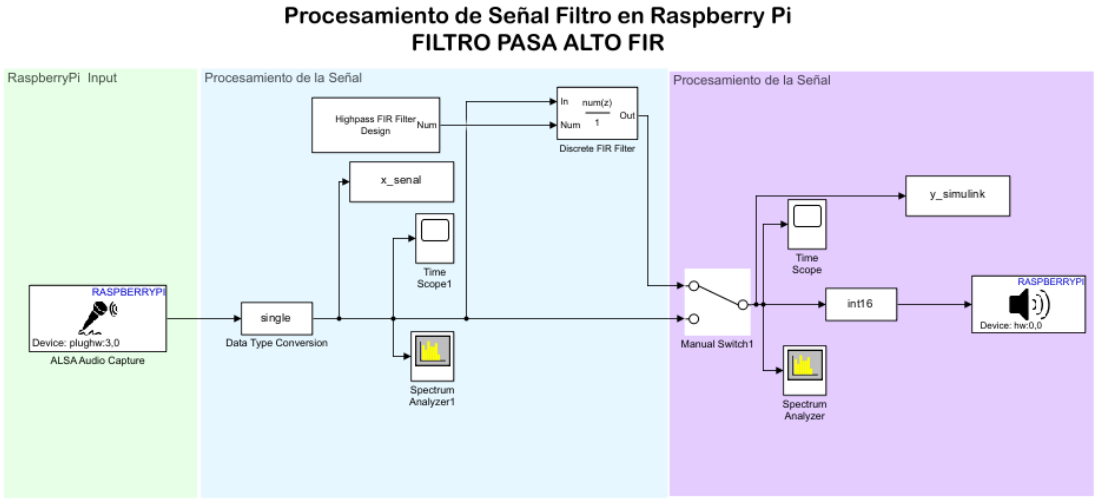


Fig.4.15. Modelo Filtro Pasa Alto FIR

En la figura 4.15 se puede observar el modelo del filtro pasa alto con frecuencia de corte en 1000 Hz y sus resultados en tiempo y frecuencia en la figura 4.16.

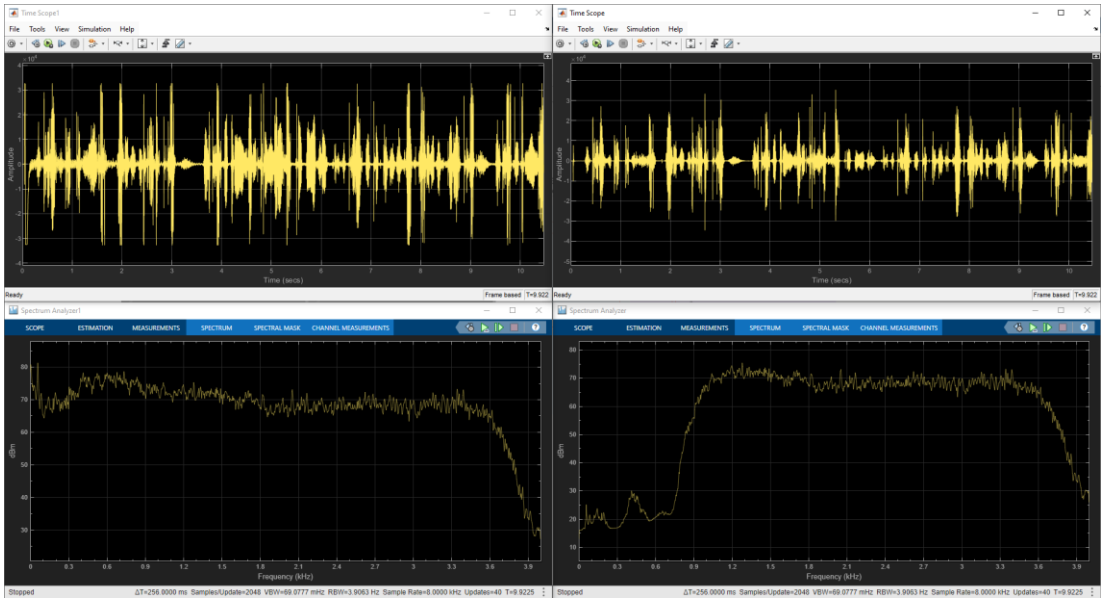


Fig.4.16. Filtro Pasa Alto FIR antes y después en frecuencia y tiempo.

Los resultados de la fig. 4.16, muestran que el filtro FIR pasa alto diseñado a un orden N =54 con ventana de Hamming aplicado a una señal de voz real capturada desde un micrófono conectado a la raspberry Pi, atenúa eficazmente las frecuencias por debajo al umbral esperado. En el dominio de la frecuencia el espectro muestra una clara atenuación a las frecuencias menores a 1kHz confirman el funcionamiento correcto de este filtro sobre la señal de voz real capturada.

4.2.3. Filtro FIR Pasa Banda:

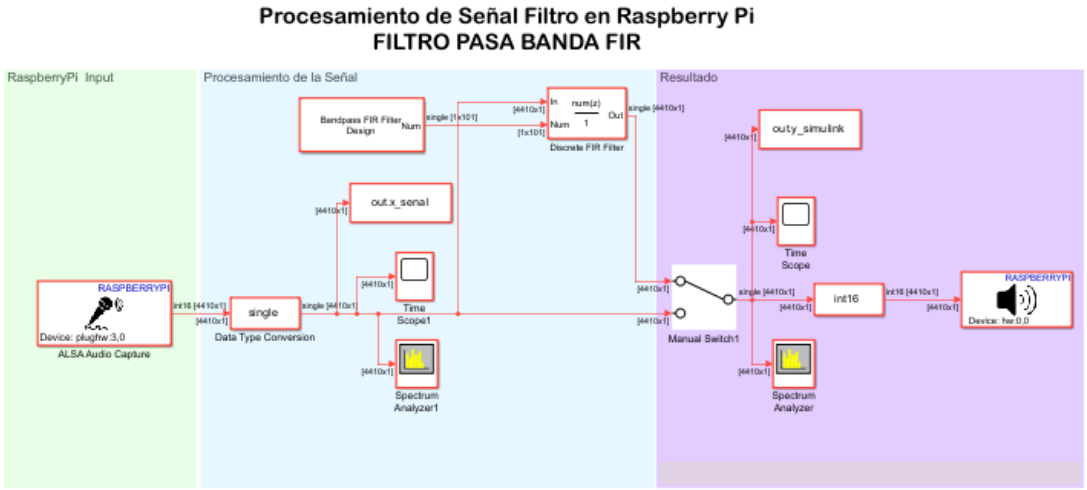


Fig.4.17. Modelo Filtro Pasa Banda FIR

En la figura 4.17 se puede observar el modelo del filtro pasa banda con una banda de paso con centro en 1500 Hz, sus resultados en tiempo y frecuencia en la fig. 4.18.

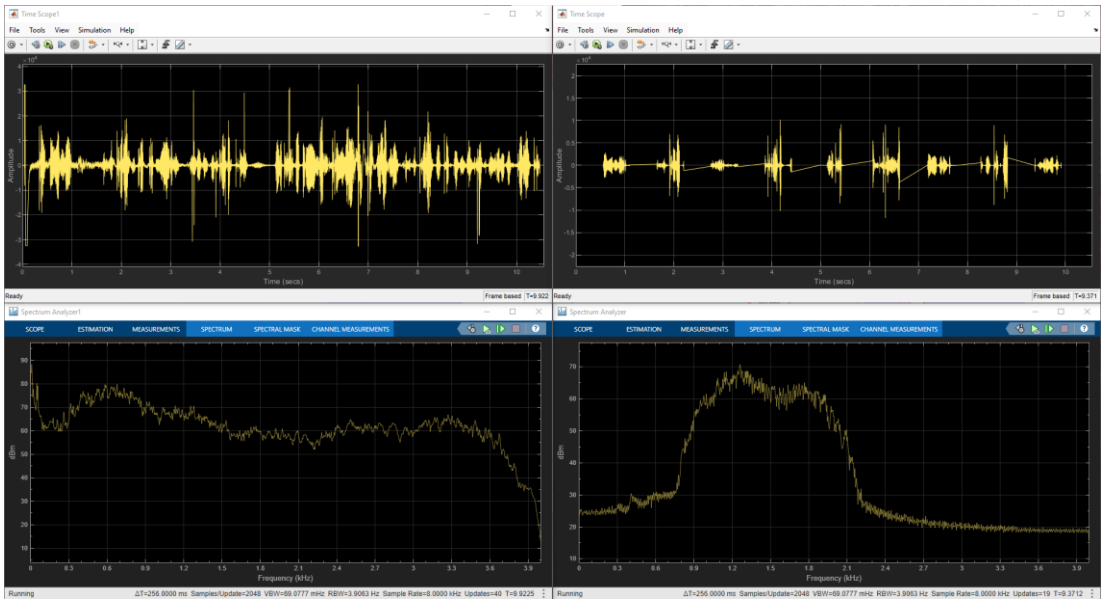


Fig.4.18. Filtro Pasa Banda FIR antes y después en frecuencia y tiempo.

Los resultados de la fig. 4.18, muestran que el filtro FIR pasa banda diseñado a un orden N =54 con ventana de Hamming, donde se muestra una clara atenuación fuera del rango deseado. En el dominio del tiempo se observa una señal suavizada indicando que solo una banda de frecuencias no ha sido alterada. En el dominio de la frecuencia el espectro muestra un grupo de bandas con 1000Hz de ancho de banda centradas en 1500 Hz, además de una fuerte atenuación fuera de estas confirmando el correcto funcionamiento de este filtro sobre la señal de voz real capturada desde el micrófono de la Raspberry Pi

4.2.4. Filtro FIR Rechaza Banda:

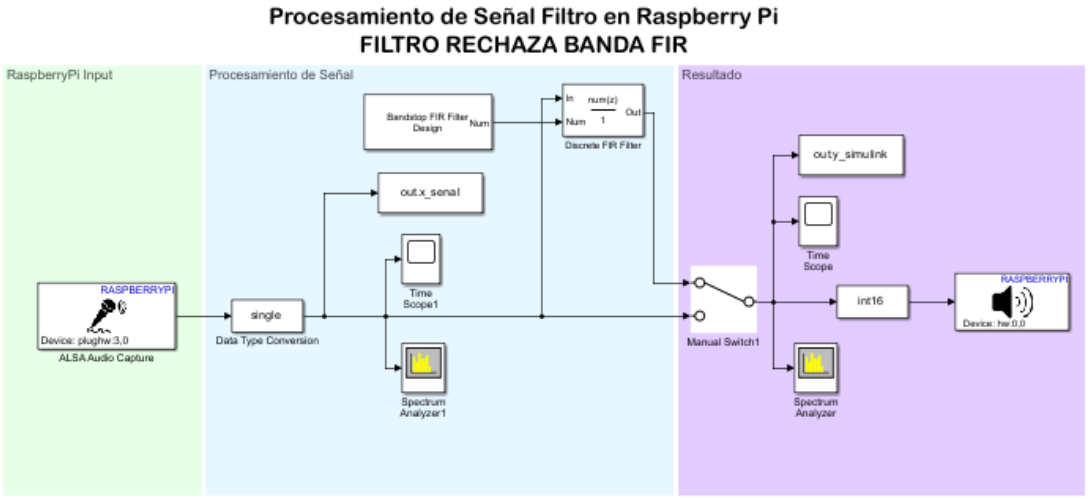


Fig.4.19. Modelo Filtro Rechaza Banda FIR

En la figura 4.19 se puede observar el modelo del filtro pasa banda con una frecuencia de paso de 600 Hz y sus resultados en tiempo y frecuencia en la fig. 4.20.

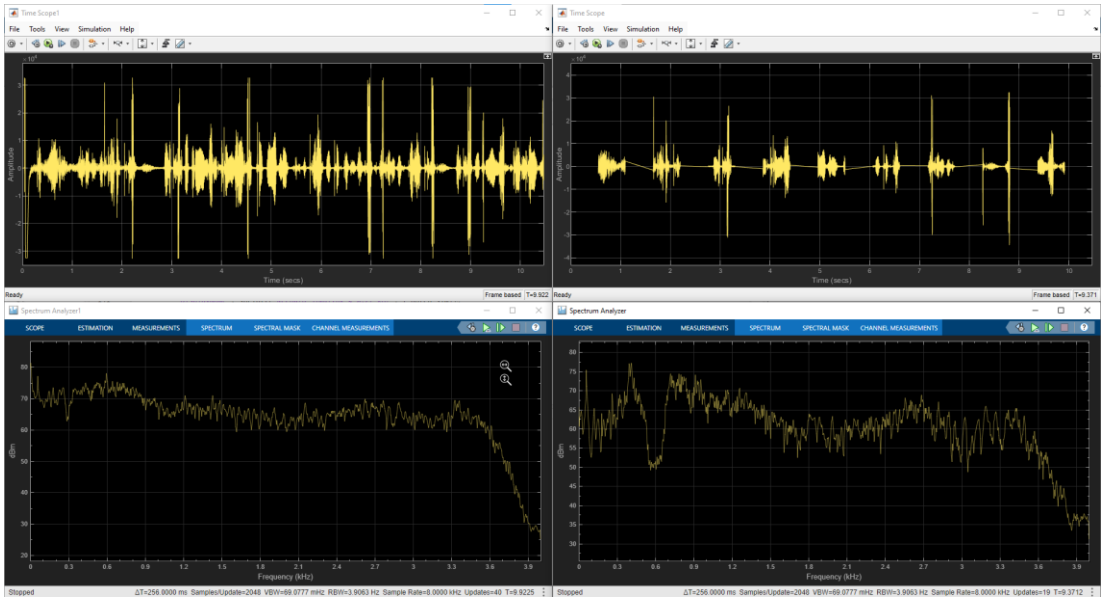


Fig.4.20. Filtro Pasa Banda FIR antes y después en frecuencia y tiempo.

Los resultados de la fig. 4.20, muestran que el filtro FIR rechaza banda diseñado a un orden N =54 con ventana de Hamming, donde se muestra una clara atenuación dentro del rango deseado. En el dominio del tiempo se observa una señal con una leve pérdida a comparación con la señal de entrada lo que demuestra pérdidas en este caso por la atenuación en 600 Hz por lo que el filtro funciona correctamente a las expectativas planteadas procesando correctamente la señal.

4.2.5. Filtro IIR Pasa Bajo:

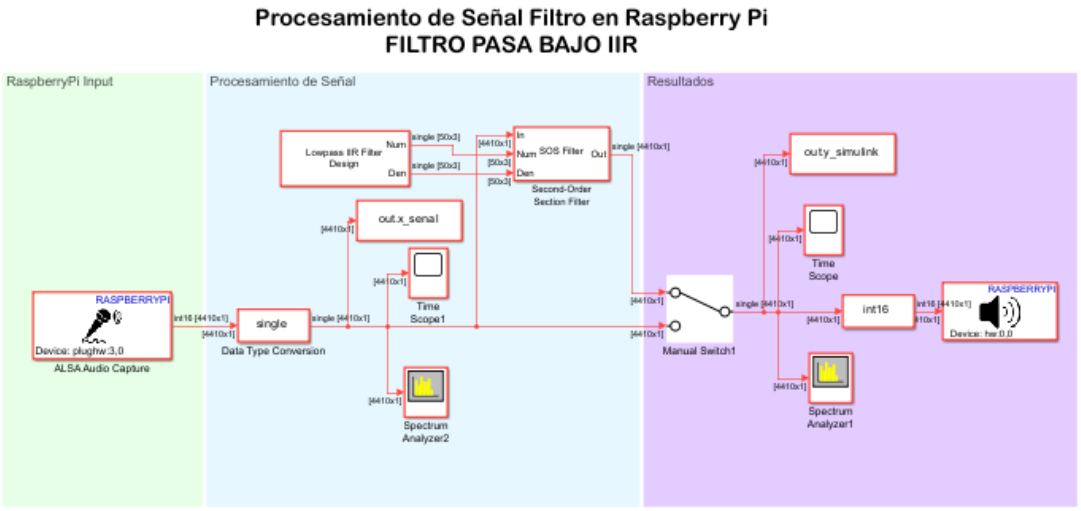


Fig.4.21. Modelo Filtro Pasa Bajo IIR

En la figura 4.21 se puede observar el modelo del filtro pasa bajo con frecuencia de corte en 2000 Hz y sus resultados en tiempo y frecuencia en la figura 4.22.

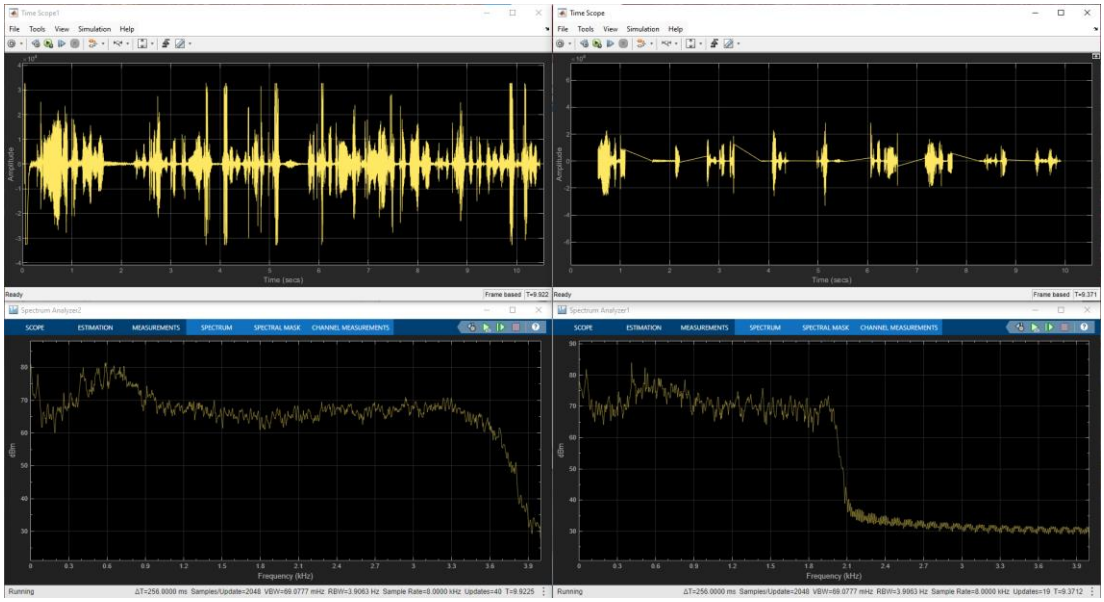


Fig.4.22. Filtro Pasa Bajo IIR antes y después en frecuencia y tiempo.

Los resultados de la fig. 4.22, muestran que el filtro IIR pasa bajas con orden 54 y frecuencia de corte de 2000 Hz aplicado a una señal de voz, se observa que atenúa correctamente las frecuencias, ya que muestra una caída abrupta en estas tanto en espectro como en tiempo puesto que entrega una señal más suave. Además, al compararla con el filtro FIR de igual orden y corte, el IIR es más rápido y usa menos recursos computacionales que el FIR debido a que utiliza Butterworth que reduce las frecuencias altas de forma más brusca, pero puede alterar un poco la señal al realizar este procesamiento.

4.2.6. Filtro IIR Pasa Alto:

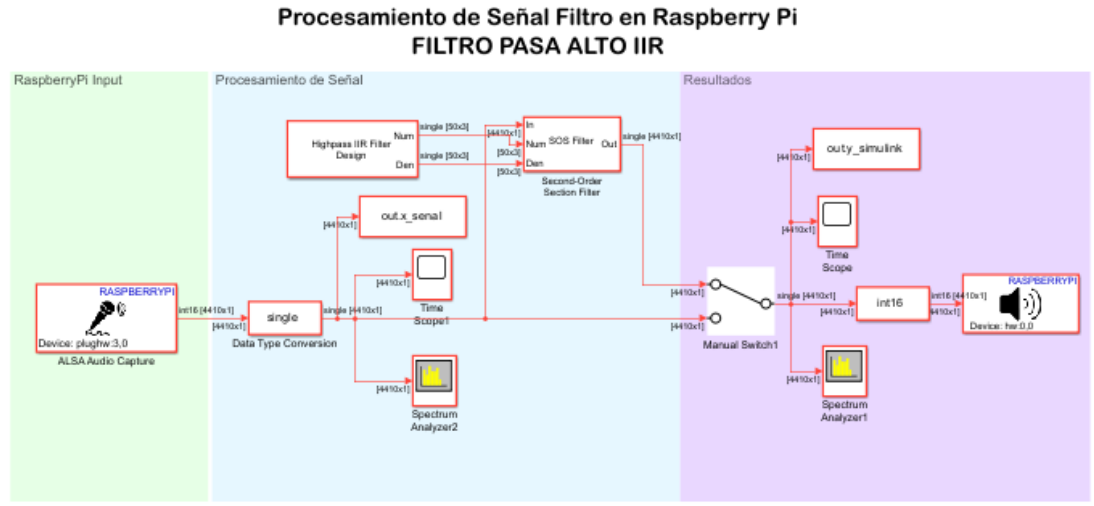


Fig.4.23. Modelo Filtro Pasa Alto IIR

En la figura 4.23 se puede observar el modelo del filtro pasa bajo con frecuencia de corte en 1000 Hz y sus resultados en tiempo y frecuencia en la figura 4.24.

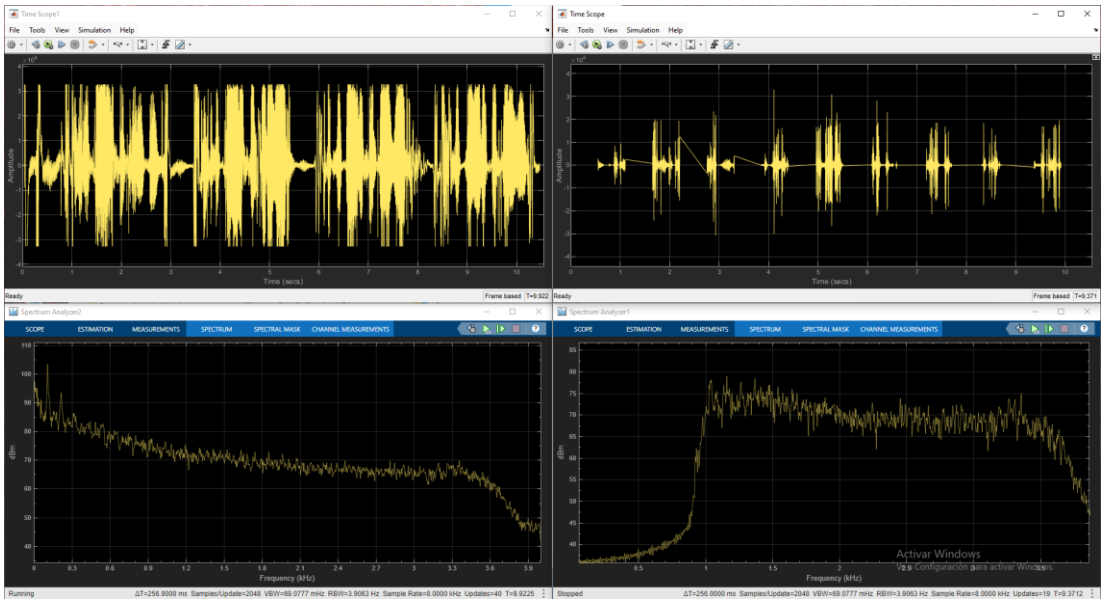


Fig.4.24. Filtro Pasa Alto IIR antes y después en frecuencia y tiempo.

Los resultados de la fig. 4.24, muestran que el comportamiento del filtro IIR pasa altos con Butterworth, el cual es correcto, ya que elimina correctamente frecuencias bajas de 1k HZ en el espectro y en tiempo se demuestra con una señal más suave. Al comparar con el filtro FIR pasa altos con Hamming y misma configuración se muestra que IIR es más eficiente computacionalmente y logra una transición más abrupta en el espectro, teniendo una caída más vertical, además en el dominio del tiempo la señal puede presentar pequeñas deformaciones.

4.2.7. Filtro IIR Pasa Banda:

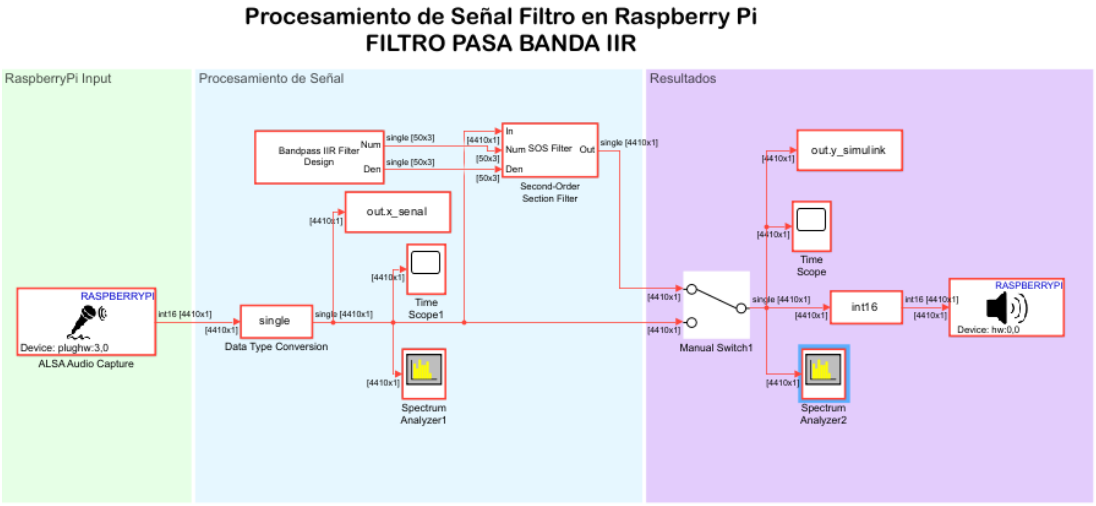


Fig.4.25. Modelo Filtro Pasa Banda IIR

En la figura 4.25 se puede observar el modelo del filtro pasa banda con una banda de paso con centro en 1500 Hz, sus resultados en tiempo y frecuencia en la fig. 4.26.

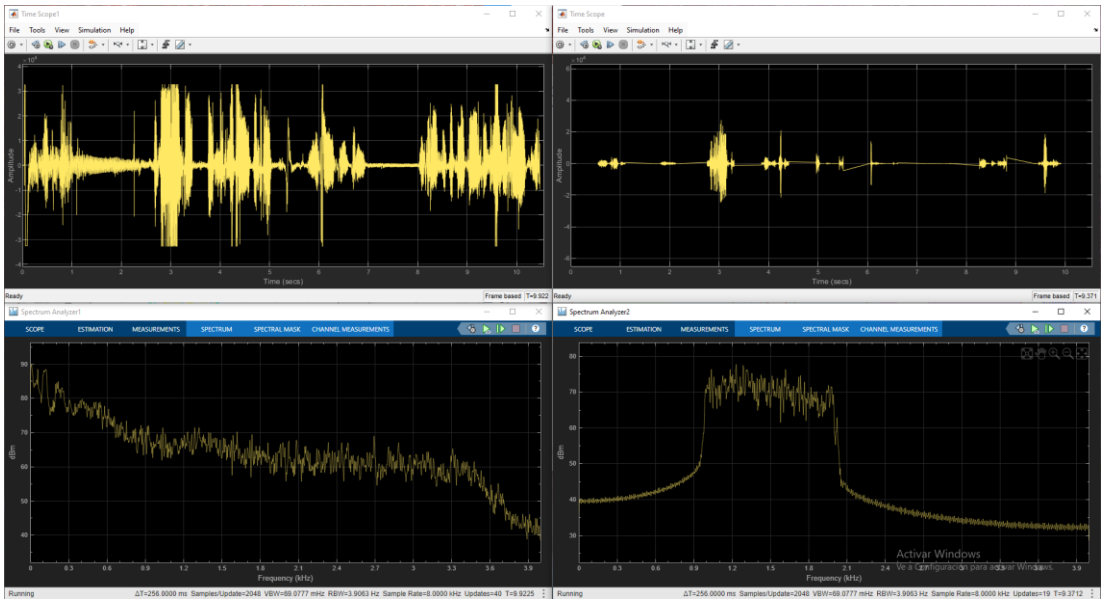


Fig.4.26. Filtro Pasa Banda IIR antes y después en frecuencia y tiempo.

En los resultados de la fig. 4.26, se muestra el comportamiento del filtro IIR pasa banda Butterworth con orden 54, frecuencias de corte de 1000 y 2000 Hz aplicado a una señal de voz. Se observa que la señal filtrada conserva únicamente el contenido de esta banda y atenuando fuertemente fuera de esta, en el espectro se muestra cortes abruptos característico de este tipo de filtro. Al compara con el filtro FIR de similares características se identifica que existe una transición más definida, además en el dominio del tiempo los bordes son más marcados lo que puede indicar una leve deformación en la señal.

4.2.8. Filtro IIR Rechaza Banda:

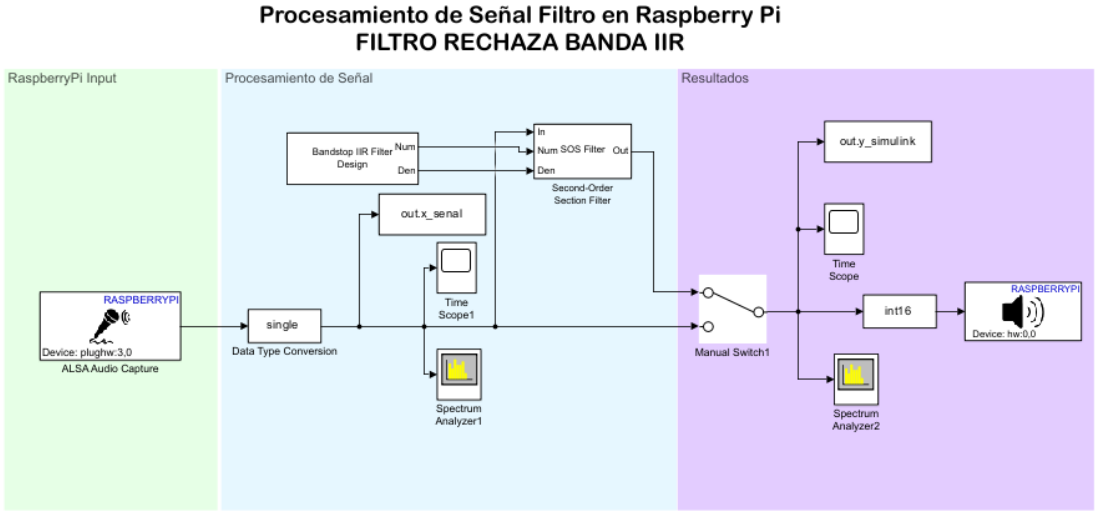


Fig.4.27. Modelo Filtro Pasa Banda IIR

En la figura 4.27 se puede observar el modelo del filtro pasa banda con una frecuencia de paso de 600 Hz, sus resultados en tiempo y frecuencia en la fig. 4.28.

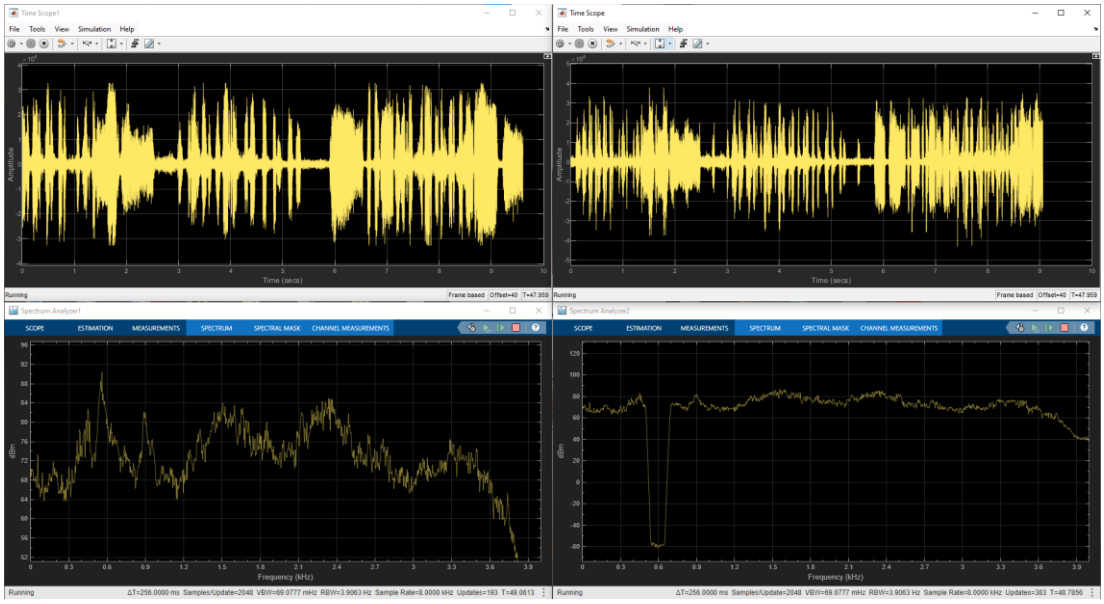


Fig.4.28. Filtro Pasa Banda IIR antes y después en frecuencia y tiempo.

En los resultados de la fig. 4.28, se muestra que el filtro IIR rechaza banda Butterworth con orden 54 y banda eliminada entre 500 y 700 Hz aplicado a una señal de voz, logra atenuar fuertemente esta zona de 600Hz en el espectro obteniendo una caída muy marcada y profunda en el centro de la banda. Al compararlo con el filtro FIR de misma configuración, se observa que IIR tiene un corte más pronunciado y definido, aunque la señal en el dominio del tiempo la señal puede verse afectada en su forma.

4.2.9. Filtro Adaptativo NLMS:

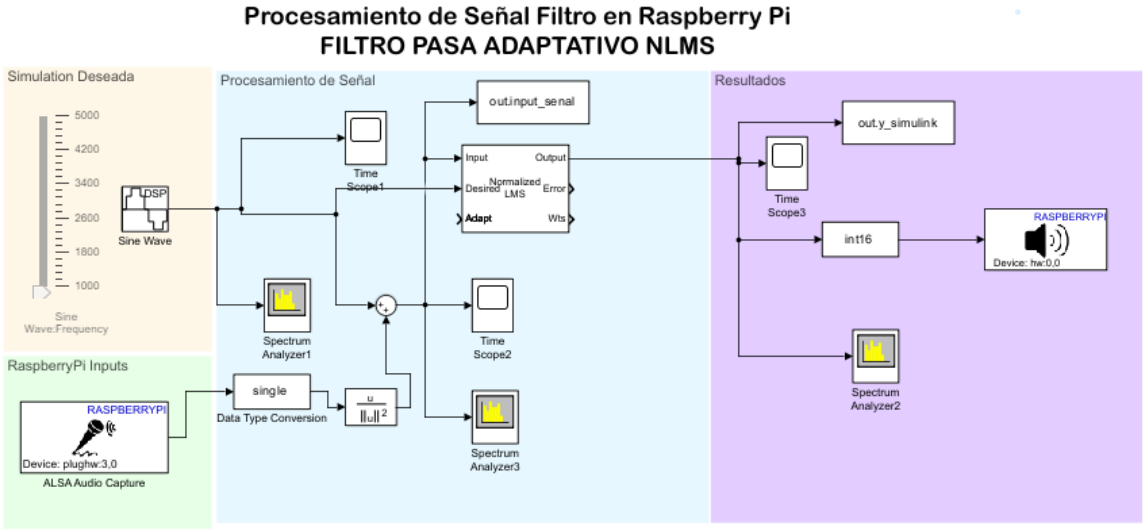


Fig.4.29. Modelo Filtro Adaptativo NLMS

En la figura 4.29 se puede observar el modelo del filtro adaptativo con frecuencia de paso esperada de 500 Hz, sus resultados en tiempo y frecuencia en la fig. 4.30.

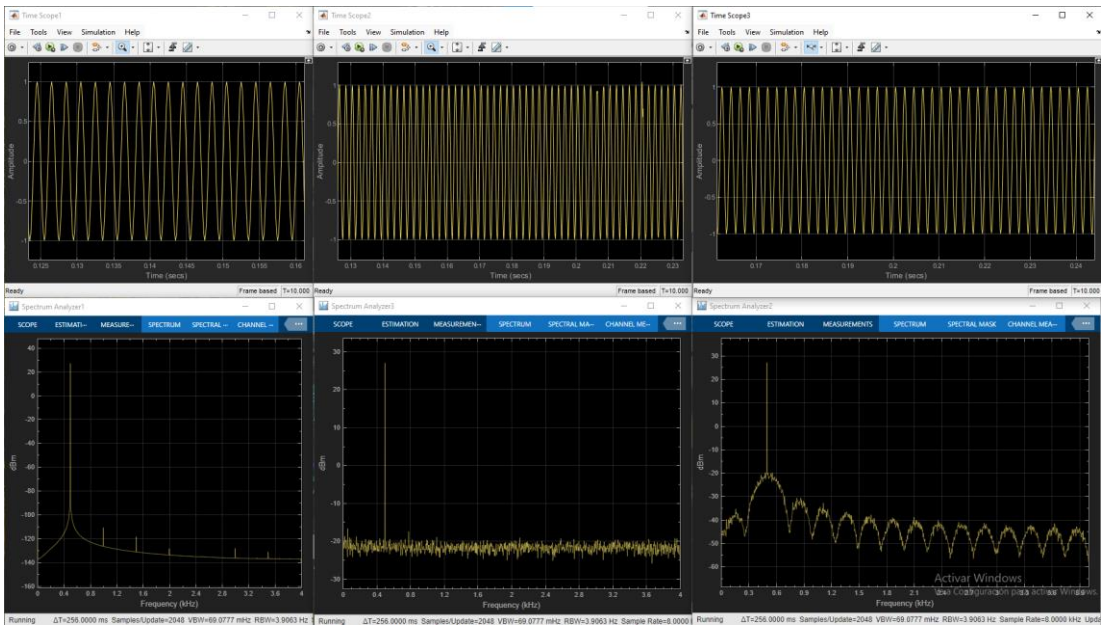


Fig.4.28. Filtro Pasa Banda IIR antes y después en frecuencia y tiempo.

En los resultados de la fig. 4.28, se muestra el funcionamiento del filtro adaptativo NLMS con u = 0,01 y orden 3. En la figura se puede observar tres fases de la señal, primero la señal en tiempo y frecuencia deseada de 500Hz seguido de la señal contaminada con ruido, finalmente la señal pasada por el filtro ya recuperada.

En el *Spectrum Analizer 2* se observa con un pico en 500Hz la cual es la misma señal que se esperaba como se muestra en el *Spectrum Analizer 1*.

4.2.10. Error Relativo y Absoluto para cada Filtro:

En los errores de los siguientes filtros se compara la misma señal extraída del micrófono conectado a la Raspberry en una variable, la cual es filtrada en el codigo implementado Matlab y simulado en Simulink, después comparada en el dominio de la frecuencia.

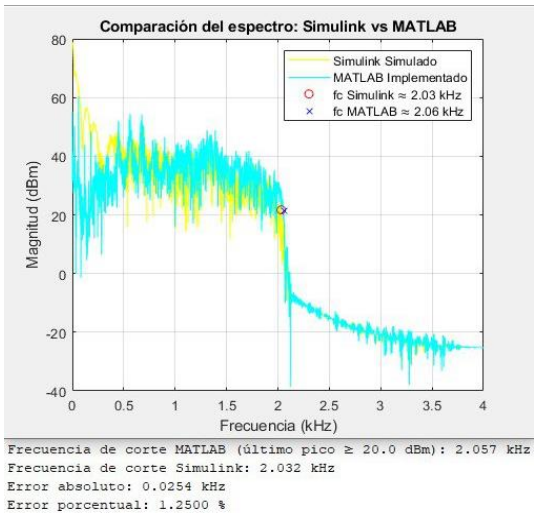


Fig.4.29. Error Filtro Pasa Bajo FIR

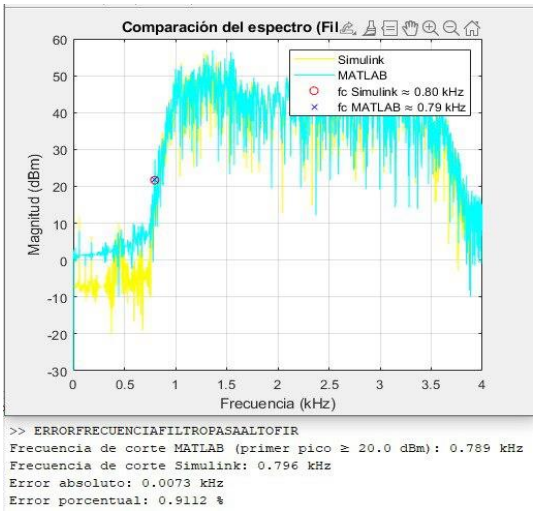


Fig.4.30. Error Filtro Pasa Alto FIR

En la figura 4.29 se observa que se obtuvo un error de 1,25% en la frecuencia de corte que se obtuvo una diferencia de 0.0254 kHz.

En la figura 4.30 se observa que se obtuvo un error de 0,9112% teniendo una diferencia de 0.0073 kHz entre sus frecuencias de corte.

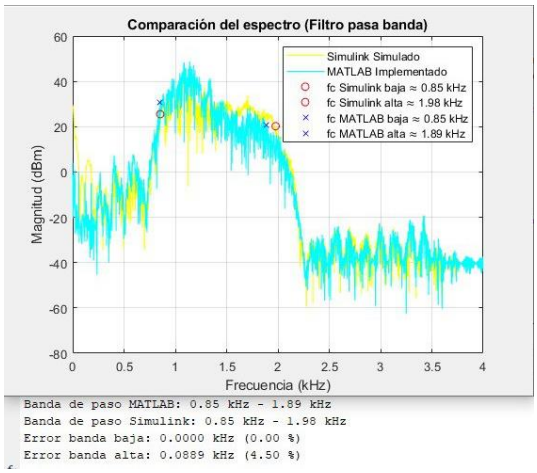


Fig.4.31. Error Filtro Pasa Banda FIR

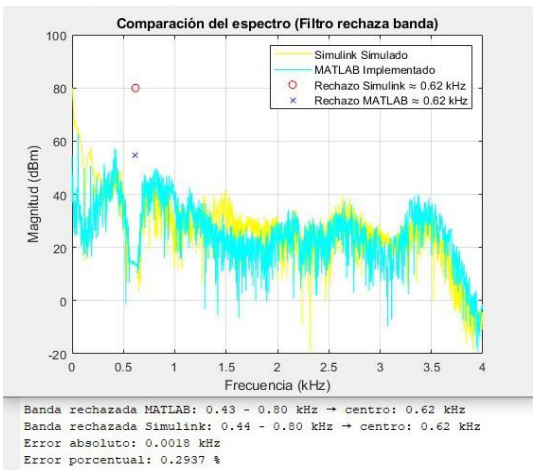
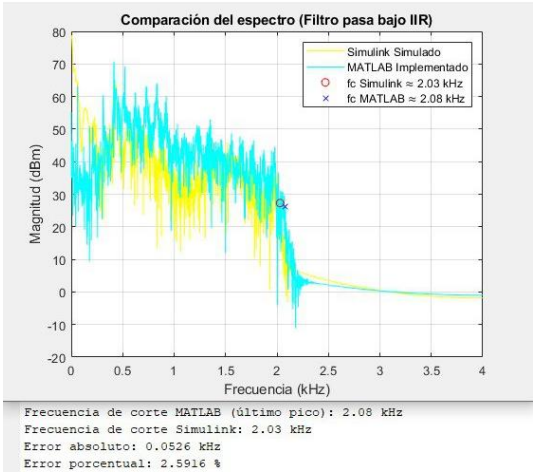


Fig.4.32. Error Filtro Rechaza Banda FIR

En la figura 4.31 se observa que se obtuvo un error de 0% en la frecuencia de corte baja, en cambio en la frecuencia de corte alta se obtuvo 4.50% de error con una diferencia de 0.0089 kHz.

En la figura 4.32 se observa que se obtuvo un error de 0,2937% en la banda baja con un error de 0.0018, en cambio en la frecuencia de corte alta un error del 0%, siendo su corte en el mismo punto.



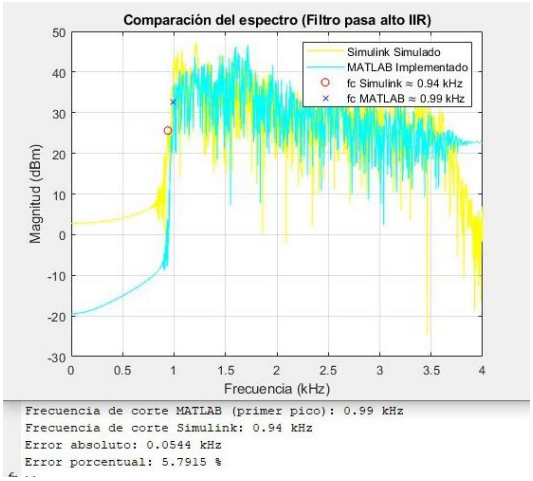
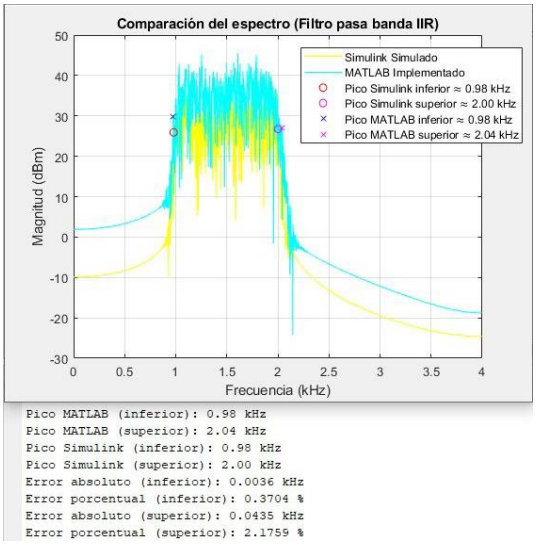


Fig.4.33. Error Filtro Pasa Bajo IIR **Fig.4.34.** Error Filtro Pasa Alto IIR

En la figura 4.33 se observa que se obtuvo un error de 2,5916% en la frecuencia de corte con una diferencia de 0.0526 kHz.

En la figura 4.34 se observa que se obtuvo un error de 5,7915% con diferencia entre Matlab y Simulink de 0,0544 kHz.



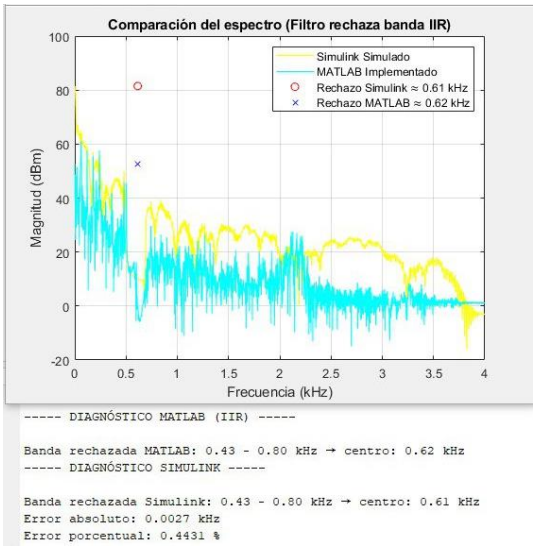


Fig.4.34. Error Filtro Pasa Banda IIR **Fig.4.35.** Error Filtro Rechaza Banda IIR

En la figura 4.34 se observa que se obtuvo un error de 0.3704% en la frecuencia de corte baja con diferencia de 0.036kHz, en cambio en la frecuencia de corte alta se obtuvo 2,1759% de error con una diferencia de 0.0435 kHz.

En la figura 4.35 se observa que se obtuvo un error de 0% en la banda baja, en cambio en la frecuencia de corte alta un error del 0,4431%, teniendo una diferencia de 0,0027 kHz.

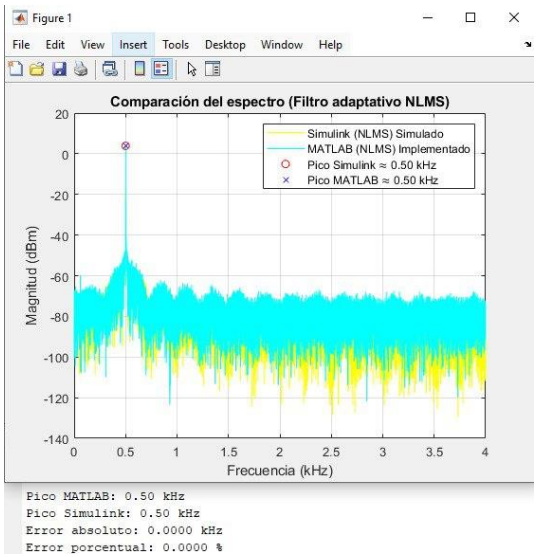


Fig.4.36. Error Filtro Adaptativo

En la figura 4.36 de muestra el filtro adaptativo en el dominio de la frecuencia teniendo un buen filtrado tanto en Matlab como en Simulink, ya que se observa una sola frecuencia a la salida de ambos filtros con un error de 0%.

A continuación, en la Tabla 4.1 se muestran los resultados de frecuencia de corte esperada (fc_E) con los resultados implementados en Matlab y en Simulink.

Tabla 4.1. Valores Error en Frecuencia.

FILTRO	fc_E (kHz)	Fc en Matlab (kHz)	Fc en Simulink (kHz)	Error Absoluto (kHz)	Error (%)
FILTRO PASA BAJO FIR	2.0	2.057	2.032	0.0254	1.2500
FILTRO PASA ALTO FIR	1.0	0.789	0.796	0.0073	0.9112
FILTRO PASA BANDA FIR	1.0	0.85	0.85	0	0
FILTRO PASA BANDA FIR	2.0	1.89	1.98	0.0889	4.5
FILTRO RECHAZA BANDA FIR	0.35	0.43	0.44	0.0018	0.2937
FILTRO RECHAZA BANDA FIR	0.85	0.8	0.8	0	0
FILTRO PASA BAJO IIR	2.0	2.08	2.03	0.0526	2.5916
FILTRO PASA ALTO IIR	1.0	0.99	0.94	0.0544	5.7915
FILTRO PASA BANDA IIR	1.0	0.98	0.98	0.0036	0.3704
FILTRO PASA BANDA IIR	2.0	2.04	2.0	0.0435	2.1759
FILTRO RECHAZA BANDA IIR	0.45	0.43	0.43	0.0027	0.4431
FILTRO RECHAZA BANDA IIR	0.75	0.8	0.8	0	0
FILTRO ADAPTATIVO NLMS	0.5	0.5	0.5	0	0

Elaborado por: Luis Daniel Sarabia Guillermo

CAPITULO 5

Análisis De Costos

En este capítulo se evaluará el análisis de costos de este proyecto educativo, el cual se evaluará la rentabilidad que este tendrá para los usuarios en este caso estudiantes de la universidad politécnica salesiana identificando costos fijos y variables.

A continuación, se detalla el presupuesto estimado para el desarrollo del prototipo del sistema de filtrado de señales de voz basado en Raspberry Pi y MATLAB-Simulink:

5.1. Costos Fijos:

Tabla 5.1. Módulos y Materiales

EQUIPO	DESCRIPCIÓN	CANTIDAD	VALOR TOTAL (USD)
Raspberry Pi 4 Modelo B (4GB)	Microcomputadora principal para procesamiento	1	$80.00
Tarjeta microSD (32 GB Clase 10)	Almacenamiento para el sistema operativo	1	$10.00
Adaptador de corriente	Fuente de energía estable para la Raspberry Pi	1	10.00
Micrófono USB	Dispositivo de entrada de voz	1	15.00
Cables y adaptadores	Cables de Red	2	10.00
MATLAB y Simulink Student Suite.	Licencia estudiantil Matlab y Simulink	1	55.00
Energía Eléctrica e Internet	Costo Simbólico		10.00
		TOTAL	**190.00**

Elaborado por: Brayan Alexander Bravo Jaya

Como se puede observar en la Tabla 5.1, el presupuesto es asequible para un entorno educativo, destacando que a los estudiantes de laboratorio de TPS tienen acceso al préstamo de Raspberry, computadoras, cables de red y la licencia de Matlab lo que reduce significativamente los costos mostrados.

5.2. Costos Variables:

Justificación del Valor por Hora del Trabajo Técnico de Desarrollo:

Según el artículo 7 de la Ley de Pasantías en el Sector Empresarial de Ecuador, el estipendio mensual no puede ser inferior a un tercio del Salario Básico Unificado (SBU), que para el año 2025 corresponde a USD 153.33. Además, el empleador debe asumir la afiliación del pasante al Instituto Ecuatoriano de Seguridad Social (IESS), lo cual representa un costo adicional.

Este valor se usa como punto de partida para calcular el desarrollo técnico de cada filtro. Al igual que las pasantías el proyecto ha sido ejecutado por estudiantes con conocimientos técnicos previos en áreas como diseño de filtros digitales, simulación en MATLAB y manejo de hardware embebido, además de tener un proceso formativo desarrollando habilidades de experimentación y aprendizaje por lo que se considera adecuado adoptar un valor por hora proporcional al de una pasantía técnica de 6$ por hora, ya tomando en cuenta el valor adicional de la aportación del IESS.

Tabla 5.2. Justificación del Valor

FILTRO	HORAS ESTIMADAS	COSTO PARCIAL
FIR Pasa Bajo	3	$18.00
FIR Pasa Alto	3	$18.00
FIR Pasa Banda	4	$24.00
FIR Rechaza Banda	4	$24.00
IIR Pasa Bajo	3	$18.00
IIR Pasa Alto	3	$18.00
IIR Pasa Banda	4	$24.00
IIR Rechaza Banda	4	$24.00
Filtro Adaptativo (NLMS)	5	$30.00
TOTAL	**33**	**$198.00**
Validación y Pruebas	10	$60.00
SUBTOTAL	43	**$258.00**

Elaborado por: Luis Daniel Sarabia Guillermo

Este costo representa únicamente el trabajo de ingeniería y desarrollo técnico por cada filtro desarrollado.

5.3. Punto de Equilibrio:

El punto de equilibrio es calve para determinar el número de unidades del producto se deben vender para cubrir los costos fijos y variables. En este punto, no se genera utilidad ni pérdida, y sirve como indicador para tomar decisiones económicas y de viabilidad comercial (Ara et al., 2020).

$$Precio\ por\ Unidad + 30\% = \frac{258}{1 - 0{,}30} = 368{,}57$$

$$Punto\ de\ Equilibrio = \frac{190}{368{,}57 - 258} = 1.72$$

Eso quiere decir que con dos ventas se lograría cubrir los costos fijos, variables y tener un margen de 30% de ganancia.

Tabla 5.3. Datos Base para Calcular el Punto de Equilibro.

CONCEPTO	VALOR (USD)
Precio de Venta	368,57
Costo Fijo Total	190,00
Costo Variable por unidad	258,00
Ganancia por Unidad **P. Venta – C. Variable**	110,57
Punto de Equilibrio	2 unidades

Elaborado por: Luis Daniel Sarabia Guillermo

Tabla 5.4. Proyección de Utilidad.

CANTIDAD	VENTAS	COSTOS FIJOS	COSTOS VARIABLES	COSTO TOTAL	UTILIDAD
1	368.57	190	258	448	-79.43
2	737.14	190	516	706	31.14
3	1105.71	190	774	964	141.71
4	1474.28	190	1032	1222	252.28
5	1842.85	190	1290	1480	362.85

Elaborado por: Luis Daniel Sarabia Guillermo

La Tabla 5.4 se muestra que, para superar el punto de equilibrio y generar utilidades con un precio de venta de $368,57 por unidad, se tienen que vender 2 unidades.

5.4. Costo de Equipos del Laboratorio de TPS:

Los equipos encontrados en el laboratorio de TPS para el procesamiento de señales son

los siguientes:

Fig.5.1. Elvis II+

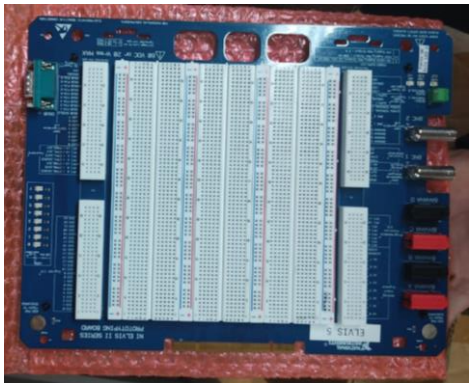


Fig.5.2. Tarjeta Elvis II+

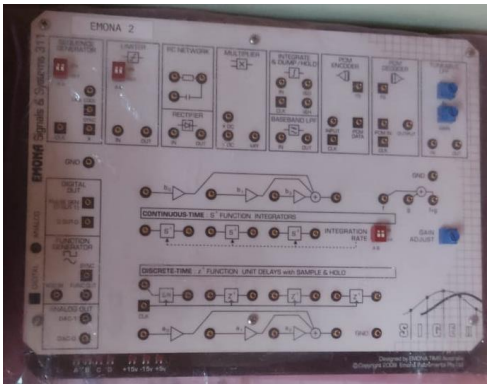


Fig.5.4. Tarjeta EMONA

Estos equipos son muy utilizados en el laboratorio de TPS de la universidad con el fin de enseñar a los estudiantes sobre el procesamiento de señales, además estos equipos en páginas como eBay y Artisan Technology Group tienen un costo de 495$ y 995$, según las condiciones en las que se encuentre el equipo y dependiendo de los accesorios. Estos valores indican que su adquisición representa un costo elevado en comparación con alternativas actuales de más bajo costo.

CAPITULO 6

6.1. Conclusiones

- En el desarrollo de este proyecto se logró profundizar en el diseño teórico y computacional de filtros FIR e IIR utilizando Matlab-Simulink como se muestra en el capítulo 3, donde se detallan los cálculos matemáticos realizados para obtener los parámetros críticos como el orden del filtro y la selección del método ya sea ventana de Haming o Butterworth, lo que permitió optimizar la respuesta en frecuencia y minimizar el rippler garantizando estabilidad y el correcto funcionamiento en aplicaciones de procesamiento de voz.
- Se diseñaron exitosamente los diferentes tipos de filtros digitales FIR e IIR como pasa bajos, pasa altos, pasa banda, rechaza banda aplicándolos a señales de voz real usando la Raspberry Pi. Como se puede observar desde la figura 4.13 a la figura 4.28, se observa que cada filtro sobre todo en frecuencia cumple con cortar o dejar pasar la señal de voz dependiendo su diseño.
- Se comparó mediante el error relativo y absoluto en el dominio de la frecuencia el rendimiento cada filtró en Simulink con su par en Matlab, obteniendo los resultados de la tabla 4.1, teniendo errores relativos y absolutos bajos, lo que indica que hay poca variación entre estos al procesar la misma señal, de igual forma se confirma de forma visual desde la figura 4.29 a 4.36, mostrando que los filtros diseñados son fiables y se comportan conforme lo esperado.
- En el análisis de costos evidenciando que este sistema completo tiene un costo fijo de 190$ y un costo variable por unidad de 258$ teniendo así un punto de equilibrio que se logra con solo vender 2 unidades obteniendo utilidades superiores a 100$ desde la tercera unidad. Estos valores al comparar con equipos comerciales demuestran la viabilidad económica y la accesibilidad del sistema para laboratorios educativos como el de TPS.

6.2. Recomendaciones

- Se debería implementar y comparar algoritmos de filtrado adaptativo más avanzados, como RLS o técnicas basadas en inteligencia artificial, usando el mismo prototipo.
- Se sugiere desarrollar módulos adicionales para el prototipo que permitan implementar y comparar prácticas de laboratorio similares a las ofrecidas por plataformas comerciales como NI ELVIS II+

6.3. Referencias Bibliográficas

Ara, J., Md. Moheuddin, & Hossain, S. (2020). *Un estudio matemático del análisis del punto de equilibrio basado en granjas lecheras en Bangladesh*. https://doi.org/https://doi.org/10.11648/j.ijebo.20200802.13

Bhatt, T. J. (2022). *Numerical Techniques for Engineers (21UHSMA301): Unit 1 – Error Analysis*. https://es.slideshare.net/BhattTushar1/erroranalysispdf?utm_source=chatgpt.com#10

Emona Instruments Pty Ltd. (2025, March 27). *ESSB-30 and Sigex (For ELVIS II & III)*.

Farhang-Boroujeny, B. (2013). *Adaptive Filters: Theory and Applications*. https://abrarhashmi.wordpress.com/wp-content/uploads/2016/02/behrouz-farhang-boroujeny-adaptive-filters_-theory-and-applications-wiley-2013.pdf

Li Tan, J. J. (2008). *Digital Signal Processing: Fundamentals and Applications*. https://www-elec.inaoep.mx/~jmram/Digital_Signal_Processing__LI_TAN.pdf

MathWorks. (2025a, March 27). *Generar e implementar código optimizado para el filtro FIR en Raspberry Pi usando ARM Cortex-A*. 2023. https://www.mathworks.com/help/ecoder/armcortexa/ug/generate-and-deploy-optimized-code-for-ifir-filter-on-raspi-using-armcortex-a.html

MathWorks. (2025b, March 27). *Implementar un filtro de paso de banda en Raspberry Pi*. https://www.mathworks.com/help/simulink/supportpkg/raspberrypi_ref/implement-bandpass-filter-on-raspberry-pi.html

MathWorks. (2025c, March 27). *MATLAB y Simulink para procesamiento de señales*. https://la.mathworks.com/solutions/signal-processing.html

MathWorks. (2025d, March 27). *¿Qué es la programación Raspberry Pi?* https://la.mathworks.com/discovery/raspberry-pi-programming-matlab-simulink.html

MathWorks. (2025e, March 27). *Soporte de hardware para Raspberry Pi*. 2024.

7.1. ANEXOS

Instalación de Software Raspberry Pi OS y configuración:

Fig.7.1. Instalación OS en Raspberry Pi Imager.

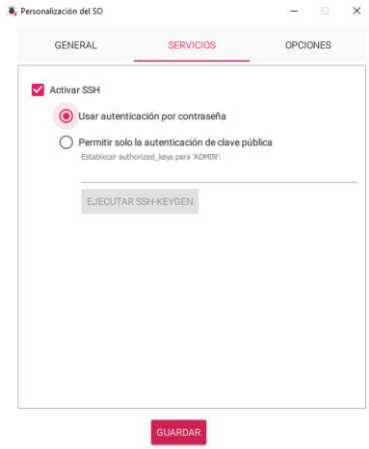


Fig.7.2. Configuración General. **Fig.7.3.** Configuración Servicios SSH.

Encontrar IP de la Raspberry mediante Advanced IP Scanner

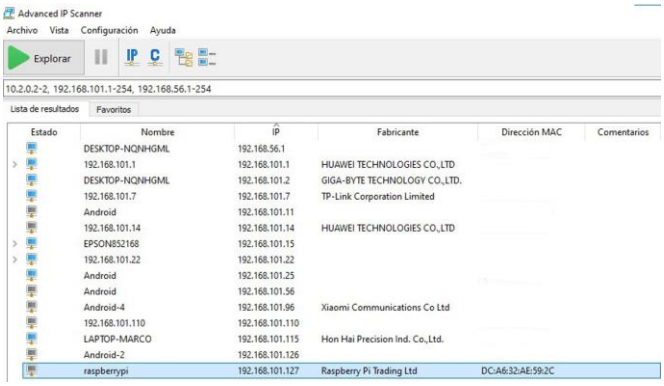


Fig.7.4. IP de Raspberry y MAC Adress

Instalación Matlab – Simulink y paquetes necesarios:

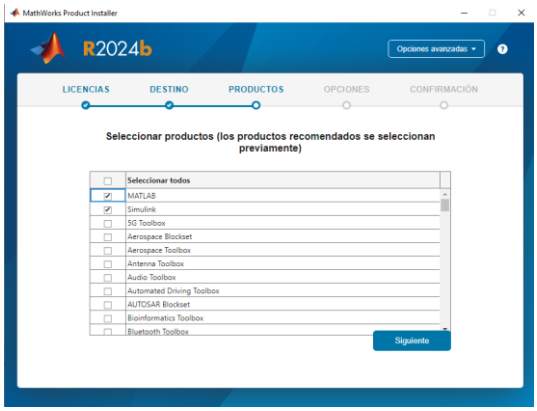


Fig.7.5. MATLAB y Simulink

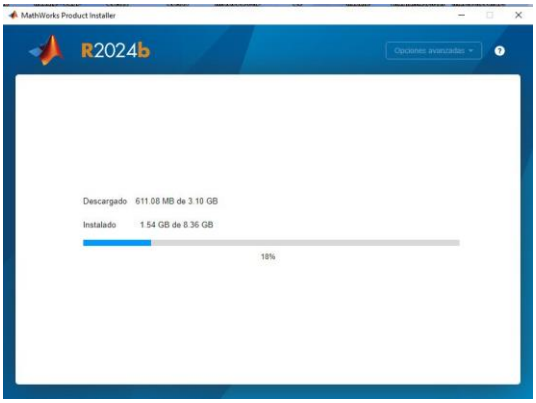


Fig.7.6. Instalación MATLAB y Simulink

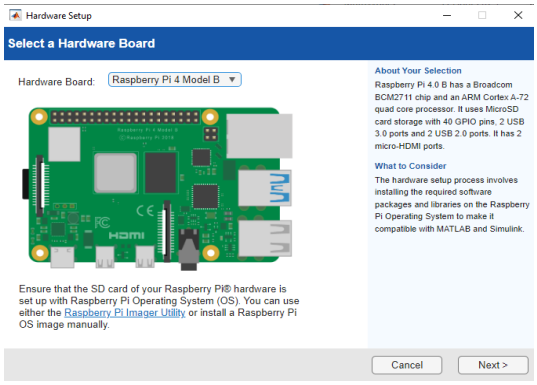


Fig.7.7. Raspberry Pi Model B

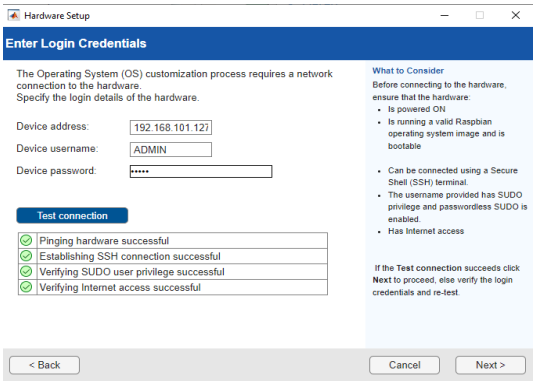


Fig.7.8. Conexión SSH y Simulink

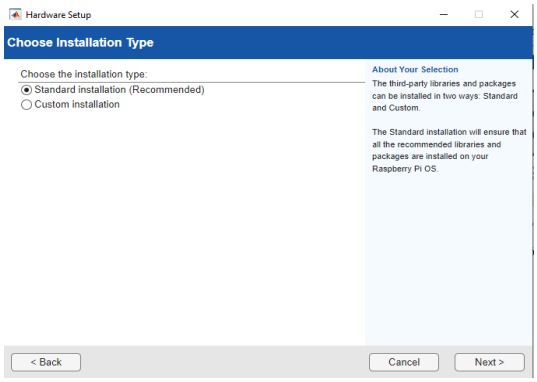


Fig.7.9. Instalación estándar

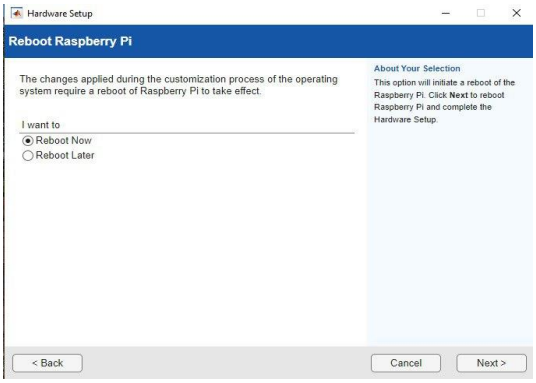


Fig.7.10. Reiniciar Raspberry

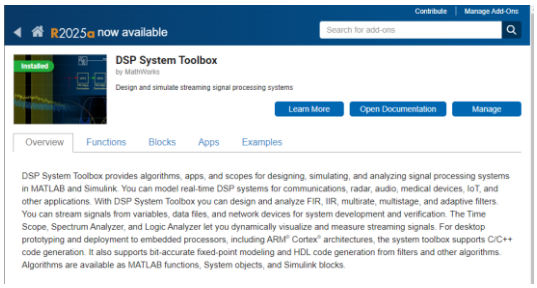


Fig.7.11. Instalación DSP.