UNIVERSIDAD DE ANTIOQUIA

Sistema de Control Activo de Ruido basado en el microcontrolador ESP32

Autor(es)

David Trujillo Lopera

Universidad de Antioquia

Facultad de Ingeniería, Departamento de Ingeniería Electrónica y de Telecomunicaciones

Medellín, Colombia

2020

Sistema de Control Activo de Ruido basado en el microcontrolador ESP32

David Trujillo Lopera

Informe de proyecto de investigación como requisito para optar al título de:

Ingeniero Electrónico

Asesor:

Gustavo Adolfo Patiño Álvarez

Ingeniero Electrónico

Línea de Investigación:
Procesamiento digital en sistemas embebidos
Grupo de Investigación:
Sistemas Embebidos e Inteligencia Computacional (SISTEMIC)

Universidad de Antioquia
Facultad de Ingeniería
Medellín, Colombia
2021

Tabla de Ilustraciones

Tabla de Contenidos

Introducción

El ruido es el segundo factor de riesgo ambiental más grande que negativamente afecta la salud humana, después de la contaminación del aire. En aquellas situaciones en las que simplemente no es viable apagar la fuente de ruido, se hace necesario realizar algún tipo de control. Este trabajo trata de una investigación en el área del control activo de ruido que se sirve del uso de componentes activos como parlantes con la finalidad de reducir el ruido y que tiene como objetivo determinar la viabilidad del uso de microcontroladores para tal fin.

Este trabajo está motivado por la reciente irrupción en el mercado colombiano de los audífonos de cancelación de ruido y su promesa de conseguir una experiencia de consumo de audio superior ante la interferencia e inclusive los riesgos de salud asociados al ruido ambiental. Ante una rápida investigación se descubrió que la literatura existente sobre los algoritmos de cancelación de ruido tiende a descuidar la parte práctica, que está conformada por simulaciones o en donde el computador usado para la cancelación se conforma por el uso de procesadores DSP[1] dedicados introducidos por primera vez en la década de 1980. Dado el desarrollo de la microelectrónica, hoy en día existe la tendencia a que algunas aplicaciones de las DPS están siendo absorbidas por los microcontroladores y los computadores de una sola tarjeta[2] que han visto incrementado su desempeño con el paso del tiempo. Además, las DPS presentan un consumo de energía que contrasta con los audífonos de cancelación de ruido que hacen uso de baterías para funcionar. La propuesta de este trabajo es evaluar el desempeño de los algoritmos de cancelación de ruido en un hardware más modesto, como lo es el microcontrolador moderno de 32 bits.

El presente informe se organiza en 5 secciones. En la sección 1 se muestra el marco teórico, aquí se tratan los temas de cancelación de ruido, los componentes que lo conforman, un par de algoritmos que se usan para cancelar ruido y las consideraciones de programación del microcontrolador que debieron realizarse, esto es el conocimiento previo requerido para realizar las implementaciones de la sección 2, que es la metodología, en ella se investigan diferentes bases de datos de ruido y se toma una como ruido de referencia, se codifican los algoritmos requeridos para hacer cancelación y la programación y

---

[1] Digital Signal Processor

[2] *One Board Computer*

cableado del microcontrolador. La sección 3 muestra los resultados obtenidos de los planteamientos de la metodología, se muestran las características del sistema acústico construido, los resultados del filtrado, el análisis de la cuantización, la configuración de los periféricos y las pruebas de comunicación entre los componentes. Con base en los resultados la sección 4 se presentan las conclusiones y en la sección 5 aparecen los trabajos citados, tanto los libros y artículos consultados como el crédito a las imágenes tomadas de la web.

Materias

Cancelación Activa de Ruido
Procesamiento digital de señales
Ruido
Filtrado LMS

Propuesta ante la universidad

El trabajo presente se realiza en respuesta a una propuesta de proyecto realizada ante la facultad de ingeniería de la Universidad de Antioquia, dicha propuesta tiene plantea los objetivos detallados a continuación:

A. General:

Desarrollar un sistema de control de ruido industrial para entornos domiciliarios, utilizando filtros basados en Redes Neuronales Cuantizadas e implementado en un sistema embebido de bajo costo.

B. Específicos:

- Caracterizar el tipo de ruido y el entorno en el cual se llevará a cabo la cancelación de ruido.
- Evaluar el estado del arte de filtros adaptativos basados en inteligencia computacional, a fin de seleccionar el más apropiado para este proyecto.
- Seleccionar la plataforma de programación para desarrollar el filtro de control de ruido y realizar el entrenamiento y parametrización necesario para dicho filtro.
- Evaluar las plataformas de cómputo más apropiadas para la implementación del sistema de control a ser diseñado.
- Implementar todo el sistema en la plataforma seleccionada.

- Validar el sistema de cancelación de ruido propuesto mediante un montaje experimental a modo de prueba de concepto.

Estos propósitos se trabajaron a lo largo del desarrollo de la propuesta, sin embargo, con el advenimiento de la pandemia del SARS-Cov-2 y la subsecuente pérdida de acceso a la mayor parte de la infraestructura que prestaba la universidad con el fin de realizar el proyecto, lo cual repercutió en el aumento no presupuestado del tiempo requerido para realizar las actividades propuestas, por lo que no todos los objetivos específicos pudieron ser consumados en su totalidad y mientras que los primeros 4 objetivos se lograron llevar a cabo, el quinto quedo a medias, puesto que la integración armoniosa de todos los componentes no se pudo obtener y el sexto no se pudo realizar a falta de la finalización del quinto. De allí que el título de este informe debiera ser cambiado para reflejar los objetivos logrados desde "Sistema de Control Activo de Ruido basado en Redes Neuronales Artificiales" hacia "Sistema de Control Activo de Ruido basado en el microcontrolador ESP32".

La manera en la cual se plantea dar resolución a estos planteamientos es con el diseño e implementación de un sistema *ANC*[3] que implemente un algoritmo neuronal y otro algoritmo de reconocimiento dentro del área del ANC como lo es el filtrado LMS o por sus siglas en inglés *fx-LMS*[4].

El sistema de ruido se compone de dos subsistemas, un sistema acústico y un sistema o controlador digital como el propuesto en la Ilustración 1 consiste en un parlante que, actuando como fuente de control, introduce una perturbación secundaria dentro de un sistema físico con el fin de suprimir la presencia de un ruido indeseado que se propaga por el sistema. La señal de control es generada por un procesador de información que filtra los datos obtenidos de un micrófono de referencia en un punto anterior con suficiente antelación y ante el ruido residual del proceso de cancelación obtenido por micrófono de error se realiza un ajuste o adaptación del filtrado con el fin de reducir el error. Los sistemas de cancelación activa de ruido están mejor preparados para cancelar bajas frecuencias, del orden de 500Hz, dado que una mayor frecuencia implica a una mayor capacidad de procesamiento de parte del microcontrolador.

---

[3] Active Noise Control o cancelación activa de ruido

[4] Filtered-x Least Mean Squares o filtrado de mínimos cuadrados

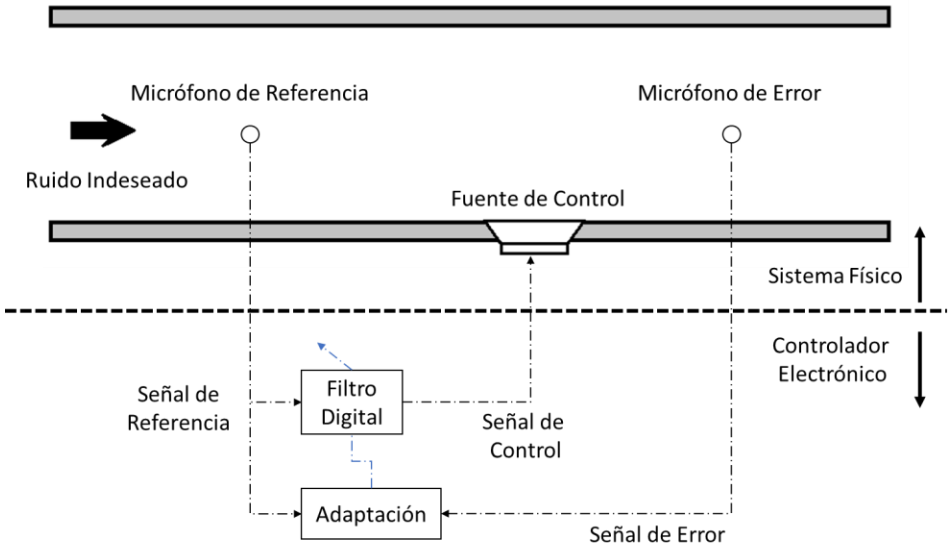


Ilustración 1: Sistema de cancelación de ruido (Imagen Propia)

1 Marco teórico

1.1 El ruido, un problema de salud

De entre las diferentes definiciones que aparecen en el diccionario se resalta la definición del ruido como "*Sonido inarticulado, por lo general desagradable*" (Lengua, s.f.). Hoy en día el ruido es un elemento permanente de la vida moderna, hasta tal punto que el ruido excesivo supone un factor de riesgo ambiental para la salud humana en conjunto con la contaminación del aire, aunque a diferencia de esta, el ruido es directamente detectable por el aparato sensorial humano. Los riesgos para la salud del ruido se manifiestan en forma de pérdida de sensibilidad auditiva, estrés psicofísico que se manifiesta en forma de enfermedades cardiacas (Münzel, y otros, 2018), condiciones de salud mental (Recio, Linares, Banegas, & Díaz, 2016), diabetes (Dzhambov, 2015) y detrimento de la calidad de sueño (Muzet, 2007). Dicha situación se agrava por el hecho de que existe una correlación directa entre el desarrollo económico y la exposición a ruido (Olayinka, 2012). Esto se manifiesta en la cercanía al tráfico terrestre y aéreo, acceso a equipos de sonido, audífonos, equipos de refrigeración, aire acondicionado y el acceso a maquinaria pesada, sumado al hecho de que un elemento fundamental en el proceso de desarrollo de las sociedades involucra el aumento de la densidad poblacional, por lo tanto, la presencia de un ruido especifico afecta a un mayor número de personas y el hecho de que haya más personas aumenta las posibles fuentes de ruido.

La organización mundial de la salud (OMS) en 2018 liberó un documento donde detalla los efectos del ruido ambiental en la vida de los ciudadanos de diferentes países europeos, y las políticas para lidiar con el ruido desde el punto de vista de la salud pública (WHO Regional Office for Europe, 2018). En él, los costos humanos para el ruido ambiental son estimados en más de un millón de AVAD[5], esta aglomera el tiempo de vida sano que una persona o un colectivo de personas pierde debido a la enfermedad, discapacidad o muerte prematura que sucede debido a una causa en particular. Se aclara que el ruido doméstico se diferencia del ruido industrial, debido a que para este ya existen diferentes estándares de seguridad industrial establecidos por las diferentes organizaciones de control gubernamental locales y en donde se requiere que la persona expuesta al ruido

[5] Años de vida ajustados por discapacidad, medida de salud pública que cuantifica el daño ocasionado por una determinada patología.

haga uso de elementos protectores en los ambientes en los que se presenta el ruido con el fin de controlar la exposición y por ende los daños asociados.

1.2 Teoría de señales y control de sistemas

Antes de iniciar a describir el control de ruido es imperativo dejar en claro conceptos relacionados con los sistemas que más adelante son propuestos.

1.2.1 Teoría de señales

Un sistema es un conjunto ordenado de elementos ordenados que interactúan entre sí haciendo parte de un todo, interacciones que suceden mediante el intercambio de energía, información o materia. Los sistemas más frecuentemente tratados por la ingeniería son los sistemas lineales, dada su simplicidad de análisis, para que un sistema sea lineal se debe cumplir que las entradas que generan las salidas cumplen tanto un principio de aditividad como de proporcionalidad y que la influencia de cada entrada en la salida puede ser estudiada por separado, a los sistemas en los que esto no es posible se les llama no lineales (Linear Operators, 2020). La realidad es intrínsecamente compleja y no lineal, siendo los sistemas lineales la excepción o a lo mejor una descripción razonable de las relaciones entre los componentes de un sistema.

Los componentes de un sistema interactúan mediante el intercambio de señales, en sistemas dinámicos las señales alteran su estado con el paso del tiempo, en los sistemas de cancelación de ruido las señales pueden ser tanto variables físicas como digitales. El control de ruido contempla la existencia de incertidumbre en las señales de ruido que se están controlando, para su análisis esto significa que en vez de trabajar con valores concretos se está haciendo uso de las estadísticas entre las señales (Wiener Filters, 2013). Una de esas estadísticas es el coeficiente de correlación dado por la Ecuación 1, donde $\bar{x}$ denota promedio, este informa de la similaridad que existe entre dos señales discretas, siendo su máximo valor 1, que significa que cuando una señal cambia, la otra también lo hace, por lo que la correlación es máxima, mientras que el menor valor es -1 que dice que si una señal hace algo, la otra responde de manera proporcional de contraria, por lo que se dice que la relación es de anti-correlación, por el contrario, cuando el coeficiente está en las cercanías de 0, se dice que la correlación es mínima y que no es posible conocer el estado de una variable conociendo el estado de la otra.

$$\mathrm{r}(\mathbf{X},\mathbf{Y}) = \frac{\sum_{i=1}^{N}(x_i - \bar{\boldsymbol{x}}) \cdot (y_i - \bar{\boldsymbol{y}})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{\boldsymbol{x}})^2} \cdot \sqrt{\sum_{i=1}^{N}(y_i - \bar{\boldsymbol{y}})^2}}$$

Ecuación 1: Coeficiente de correlación (Pairs Of Random Variables, 2006)

Un sistema de control de ruido tiene como objetivo medir el ruido y conseguir una "quietud" resultante que tiene una correlación mínima con el ruido medido.

Para el control de ruido ciertos tipos de señales son controlables mientras que otras no. Una clasificación no extensiva de las señales puede verse en la Ilustración 2. Las señales estacionarias son aquellas a las que sus promedios estadísticos no varían con el tiempo, por lo que es posible hacer estimaciones de los valores de la señal, en caso de que esto no suceda; la señal es no estacionaria, un ejemplo de esto es los transientes de los cuales no es posible hacer estimación alguna. Las señales estacionarias se pueden subdividir en determinísticas y aleatorias, las determinísticas son aquellas que pueden ser definidas por un conjunto de parámetros bien definidos como lo puede ser una senoidal que depende de la amplitud, frecuencia y fase mientras que las aleatorias no están completamente definidas. Las señales estacionarias se dividirán en periódicas y cuasi periódicas, ambas pueden ser descompuestas en tonos senoidales y por tanto tienen transformada de Fourier, se es periódica si las frecuencias presentes se relacionan entre sí por una relación de armónicos de una frecuencia fundamental mientras que si las frecuencias no están relacionadas entre sí la señal es cuasi-periódica (Hansen, Snyder, Xiaojun, Brooks, & Moreau, Spectral Analysis, 2012).

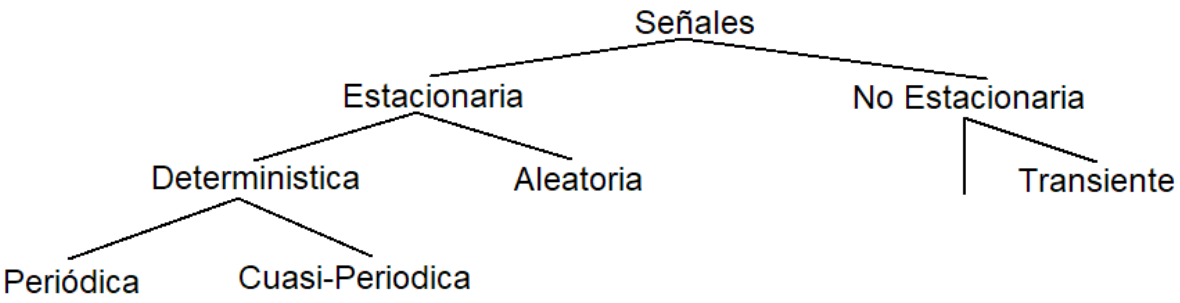


Ilustración 2: Tipos de Señales (Imagen Propia)

1.2.2 Filtrado Óptimo Lineal

El filtrado es el proceso mediante el cual se altera el contenido en el dominio de la frecuencia de una señal que es generada por un proceso ya sea análogo o digital.

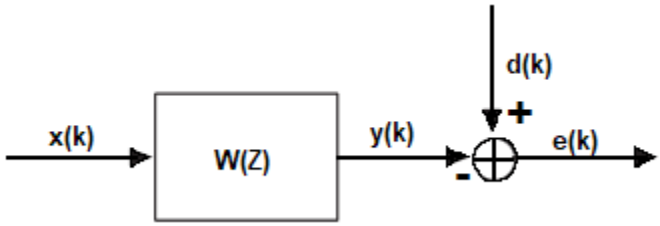


Ilustración 3: Diagrama de bloques de filtrado (Imagen Propia)

El filtrado optimo es el tipo de filtrado que se concentra en resolver la estimación de una secuencia con base en otra secuencia relacionada. Dichos filtros también son conocidos como filtros de Wiener (Wiener Filters, 2013). La situación de filtrado optimo queda definida por medio de la Ilustración 3, en ella se tiene una secuencia $x(k)$ que está siendo filtrada por un filtro $W(Z)$ tal y como lo dispone la Ecuación 2. Dicho filtrado es comparado con una muestra de la señal deseada $d(k)$ y la diferencia es el error que se desea minimizar, para ello se define un criterio de error de tal manera que cumpla las siguientes condiciones:

1. Que tenga un solo punto máximo (o mínimo)
2. Que sea matemáticamente expresable

$$y(k) = \sum_{i=0}^{N-1} x(k-i) \cdot W(i)$$

Ecuación 2: Filtrado FIR (Implementación de Sistemas Discretos en el Tiempo, 2007)

La función de error más simple que cumple los requisitos es el error cuadrático medio expresado en la Ecuación 3, en ella *E* denota el valor esperado de la variable que significa que una vez se mida, la medición se ajusta a las estadísticas matemáticas que se puedan tomar de la variable como podrían ser la media y la desviación estándar. Se anota que el error no necesariamente es cero, sino que puede mayor.

$$\xi = E[e(k)^2]$$

Ecuación 3: Criterio del error cuadrático medio

La superficie del error generada es graficable como un hiperboloide como el que aparece en la Ilustración 4, con dos coeficientes de W.

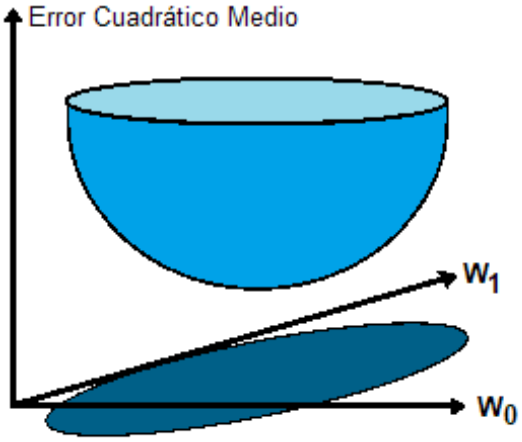


Ilustración 4: Superficie del error (Imagen Propia)

(Farhang-Boroujeny, Adaptive Filters, Theory and Applications, 2013) deduce con lujo de detalles que los coeficientes de $W(Z)$ que minimizan el error para un momento dado se expresan en la Ecuación 4. Para efectos prácticos, el valor esperado *E* genera dificultades con el análisis de datos por lo que se hace la simplificación de que el valor esperado de una variable es esa misma variable.

$$E[\boldsymbol{x}(k) \cdot \boldsymbol{x}^T(k)] \cdot w_{optimo} = E[\boldsymbol{x}(k) \cdot \boldsymbol{d}(k)]$$

Ecuación 4: Coeficientes de error mínimo (Farhang-Boroujeny, Adaptive Filters, Theory and Applications, 2013)

1.2.3 Gradiente descendiente

La sección anterior mostró el valor analítico para el filtrado óptimo expresado en la Ecuación 4. Esta no es la única manera de hallar una solución, otra manera existente consiste en escoger una semilla $\boldsymbol{W_0}$ e ir buscando un valor $\boldsymbol{W_{i+1}}$ que minimice el error de manera progresiva, una manera de lograrlo es usando el criterio de la derivada que queda expresado en la Ecuación 5. Donde μ se conoce como la tasa de aprendizaje para cada gradiente individual, un parámetro que generalmente tiene el rango de (0,1]. El parámetro μ determina la velocidad a la cual se llega a un valor cercano al resultado óptimo del filtrado, entre más grande sea, más rápido se llega al valor óptimo, sin embargo, a medida que μ se hace más grande, el valor óptimo encontrado puede llegar a divergir del valor óptimo real.

$$W_{i+1} = W_i - \mu \cdot \nabla\xi$$

Ecuación 5: Gradiente Descendiente (Search Methods, 2013)

Para el caso del filtrado óptimo se debe cumplir para μ la condición de la Ecuación 6 (Farhang-Boroujeny, Adaptive Filters, Theory and Applications, 2013), que garantiza la convergencia para el método de búsqueda donde *tr* expresa la operación de algebra de matrices *traza*, que es la suma de los valores de la diagonal principal de una matriz cuadrada.

$$0 < \mu < \frac{1}{3 \cdot tr[\boldsymbol{x}(k) \cdot \boldsymbol{x}^T(k)]}$$

Ecuación 6: Condición de convergencia (Search Methods, 2013)

Al filtrado como está expresado en la Ecuación 5 también se le conoce como filtrado LMS y es un tipo de filtrado adaptativo, esto es, que cambia sus valores con el paso del tiempo y a medida que recibe nueva información.

1.3 El control de ruido

1.3.1 Tipos de control de ruido

Con el fin de reducir las emisiones existen dos tipos de control: el control de ruido, y el análisis de vibraciones (Crocker, 2007), una rama del análisis estructural; el primero se encarga de reducir las ondas de presión sonora que se propagan a través del aire, y el segundo estudia y altera los modos de vibración en cualquier otro medio físico donde suceden vibraciones (aceites, concreto, metales, etc.) que al final se traducen en vibraciones sónicas como lo son los motores y los diferentes transductores que generan desplazamiento como los magnéticos, hidráulicos y neumáticos en los diferentes medios materiales de propagación, se anota que este último control no trata únicamente el ruido, sino que generalmente está interesado en aumentar la vida útil de la maquinaria y de los armazones de soporte empleados, utilidad que se reduce a medida que se presenten vibraciones más energéticas y más frecuentemente. Ambos tipos de control pueden ser pasivos o activos (Bies, Hansen, & Howard, 2018), para los propósitos de este estudio, solamente interesará el control de ruido activo acústico.

El control pasivo de ruido hace uso de materiales, barreras físicas y geometrías que siguen los principios de desacople de impedancias para maximizar la reflexión del sonido incidente o bien del principio de disipación de energía en forma de calor en el que una onda incurre al propagarse en un medio material. Considerando que los medios pasivos no añaden energía, se infiere que estos son inherentemente estables, es decir, que tanto todos los posibles estados del

sistema como todas sus posibles salidas se encuentran limitadas y que la respuesta será acorde a excitación proporcionada.

Por otro lado, el control activo hace uso de transductores que tienen por objetivo producir la anti-correlación de la señal física de ruido y reducir su amplitud en el dominio analógico (Elliott, 2007), idealmente hasta cero. El uso de la anti-correlación busca reducir la energía del sonido por medio de la interferencia destructiva, la operación de esta se ve reflejada en la Ilustración 5, en ella se visualiza una señal incidente de ruido, en este caso un tono senoidal, la idea es que la señal de control generada sea el contrario exacto en amplitud del tono incidente para que el resultado sea la quietud, se dice control activo puesto que el control requiere de energía adicional a la de la onda incidente para generar la anti-correlación y para generar la anti-correlación se requiere de un elemento dinámico para generarlo como lo puede ser un parlante. Se anota que este proceso es diferente de 'restar' una señal con otra, esto implicaría que es posible 'quitar' energía de una señal, lo cual es una imposibilidad en la práctica, el proceso se describe mejor diciendo que se está 'sumando el inverso aditivo' de la señal (Vibraciones y Ondas, 2006).

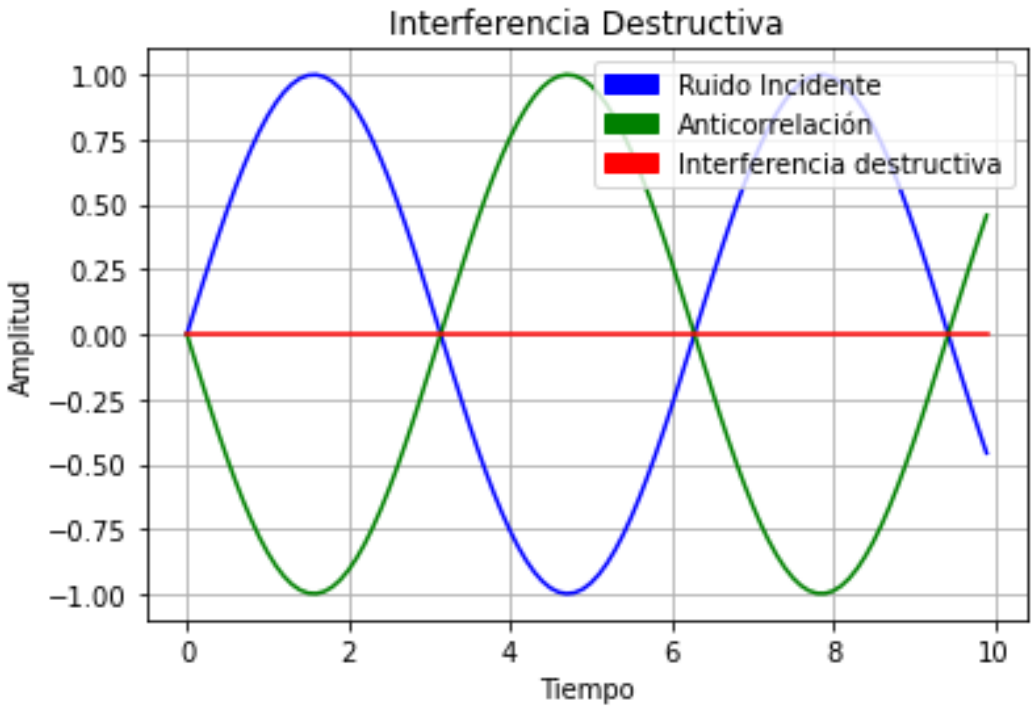


Ilustración 5: Muestra de interferencia destructiva (Imagen Propia)

Comparando ambos tipos de control, el control pasivo de ruido tiene sus limitantes a la hora de lidiar con las bajas frecuencias puesto que el mayor tamaño de las longitudes de onda involucradas en dichas frecuencias implica

que los atenuadores necesarios deben hacerse más gruesos y masivos, lo cual incrementa los costos de atenuación sensiblemente en comparación con la cancelación activa, por lo que no es económicamente viable (Barron, 2003). El control activo aventaja al pasivo por el hecho de que con el uso de transductores es posible operar a bajas frecuencias, sin embargo, este tipo de control adolece de problemas de estabilidad, necesidad de una fuente de energía, necesidad de mantenimiento, limitaciones de ancho de banda fundamentadas en la capacidad del hardware usado (Hansen, Snyder, Xiaojun, Brooks, & Moreau, Background, 2012). En la práctica, un sistema de control de ruido eficaz hace tanto de control del sonido como de las vibraciones mecánicas que lo generan y se sirve tanto del control pasivo como del control activo para lograr una meta de reducción.

Históricamente, Paul Luen (Estados Unidos Patente nº US2043416A, 1936) es considerado como el punto de partida del control activo de ruido al serle otorgado la primera patente de control de ruido. En ella, se hace uso de un transductor para introducir una señal anti-correlacionada que mediante el proceso de interferencia destructiva cancela una onda de ruido medida anteriormente. Dicha aproximación se asemeja mucho a los sistemas modernos de cancelación de ruido, que en tiempos modernos hacen uso sistemas de procesamiento digital de señales con el fin de determinar la anti-correlación de la señal de ruido, idea que fue propuesta inicialmente por (Kido, Abe, & Morikawa, 1984) y los algoritmos han ido mejorando a medida que los desarrollos de la microelectrónica han derivado en computadores más capaces de correrlos.

1.3.2 Aplicaciones del control de ruido

Las aplicaciones del control de ruido pueden ser clasificadas por la geometría en donde suceden y por las áreas en las que se aplica.

Geométricamente existen tres tipos de espacios:

- Ruido en tuberías: Este ruido tiene la característica de ser el sistema más simple y efectivo al tenerse un solo grado de libertad de movimiento, lo cual implica que las ondas sonoras solamente se propagan en el modo transversal. Sucede en ductos de ventilación, chimeneas y tubos de escape de motores de combustión interna (Hansen, Snyder, Qiu, Brooks, & Moreau, Active Control of Noise Propagating in Ducts, 2012).
- Ruido en espacios confinados: cavidades tanto las de un cuarto hogareño, industrial como la que existe en los equipos de protección auditiva, su

principal característica es que es en una región local (Hansen, Snyder, Qiu, Brooks, & Moreau, Active Control of Enclosed Sound Fields, 2012).
- Ruido en el espacio libre: ambientes abiertos (Hansen, Snyder, Qiu, Brooks, & Moreau, Active Control of Free-Field Sound Radiation, 2012).

Desde el punto de vista del área de aplicación se resaltan tres:

- De transporte: En los automóviles, buses, aviones y maquinaria de construcción, en ellos el control se enfoca en las tuberías de escape y en algunos casos se hace control al ruido en el espacio confinado de la cabina (Patente nº US4689821A, 1985), (1993 Patente nº US5245664A).
- Doméstica: Limitación del ruido producido por el aire acondicionado que es arreglado en forma de ducto, audífonos de cancelación de ruido y la limitación de ruido en cuartos (Patente nº JPS6220713A, 1985), (Patente nº US20160353197A1, 2016), (Patente nº JP2005134632A, 2003).
- Industrial: Control del ruido producido por la combustión en las chimeneas, el que se propaga por medio de las chimeneas, el producido por motores y el ruido de los transformadores (Patente nº CN203444957U, 2013), (Patente nº US5097923A, 1992).

1.4 Sistema acústico de ducto de ventilación

1.4.1 El ducto de ventilación

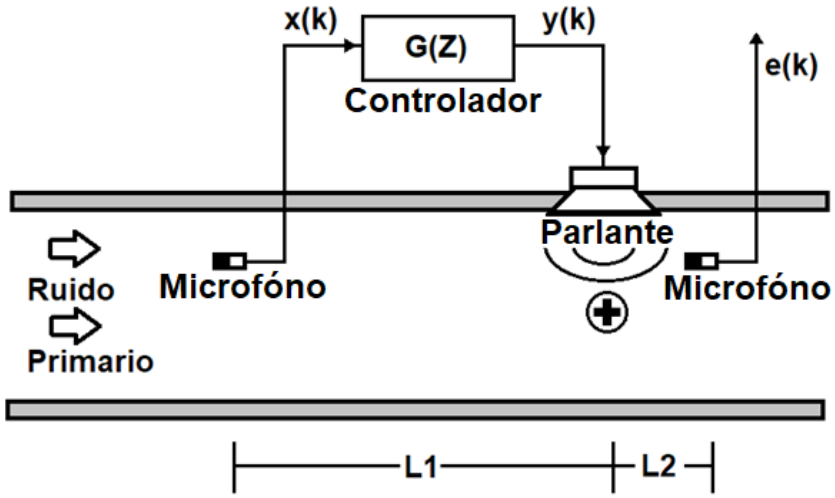
La situación de ducto de ventilación es la escogida para ser estudiada. Las bondades del estudio de este sistema radican en su replicabilidad, simpleza de construcción y almacenamiento, dado el hecho de que este es un sistema de un solo grado de libertad, una vez se cancela una onda de sonido, está queda cancelada por el resto de la extensión del tubo, esta no genera interferencia constructiva debido a un desfase espacial, y que para las frecuencias inferiores a las dadas por la Ecuación 7, donde C_0 se refiere a la velocidad del sonido en el medio y d es el diámetro de la sección transversal, se cumple que el tubo fuerza a que haya un solo modo de propagación, el transversal. Esto implica que solo se requiere de un sistema de cancelación de ruido de un solo canal para cancelar ese modo, en caso contrario se haría necesario hacer uso de un sistema de más canales debido a los otros modos de propagación de las frecuencias superiores.

$$f_{cu} = 0.586 * C_0/d$$ (Hansen, Snyder, Xiaojun, Brooks, & Moreau, Control of Noise Propagating in Ducts, 2012)

Ecuación 7: Máxima frecuencia que se propaga en un modo en área transversal circular

A nivel de diagrama de bloques, (Nelson & Elliott, 1992) describe el sistema de cancelación de ruido tal y como se puede ver en la Ilustración 6. Esta ilustración muestra el modelado del tubo, sin embargo, el uso de un algoritmo específico como el Fx-LMS introduce ligeros cambios para reflejar su funcionamiento interno. La Ilustración 6a muestra un esquema físico mientras que la Ilustración 6b muestra el modelo del sistema físico. El esquema físico es muy similar al ya presentado en la Ilustración 1, en este le da el nombre a los datos que provienen del micrófono de referencia como $\mathrm{x(k)}$ que un controlador $\mathrm{G(Z)}$ al seguir una regla de control pre-estrablecida determina una salida $\mathrm{y(k)}$ que es transmitida al ducto por medio de un parlante y que se cancela con el ruido primario en un punto de suma, más delante de este se encuentra el micrófono de error que mide el ruido residual o el error $\mathrm{e(k)}$. Las distancias L_1 y L_2 denotan la sepación que existe entre el micrófono y el parlante y del parlante al micrófono de error.

El modelo del arreglo físico de la Ilustración 6b describe por medio de funciones de transferencia la situación que sucede. Primero un ruido primario que se desea cancelar se mide y se comunica tanto con el controlador como con el camino primario (el tubo L_1), el controlador determina el anti-ruido que se propaga a través del camino secundario asociado a L_2 y que se suma con el ruido en un punto físico mediante interferencia destructiva, el ruido residual que es medido por el micrófono de referencia es la señal de error cometida por el controlador, sin embargo, la existencia de $F(j\omega)$ implica que parte del anti-ruido generado por el controlador se devuelve a través de L_1, hacia el controlador, se conoce como pseudo realimentación puesto que es indeseada y puede desestabilizar el sistema de control, en la práctica se toman medidas para limitar la injerencia de la realimentación.



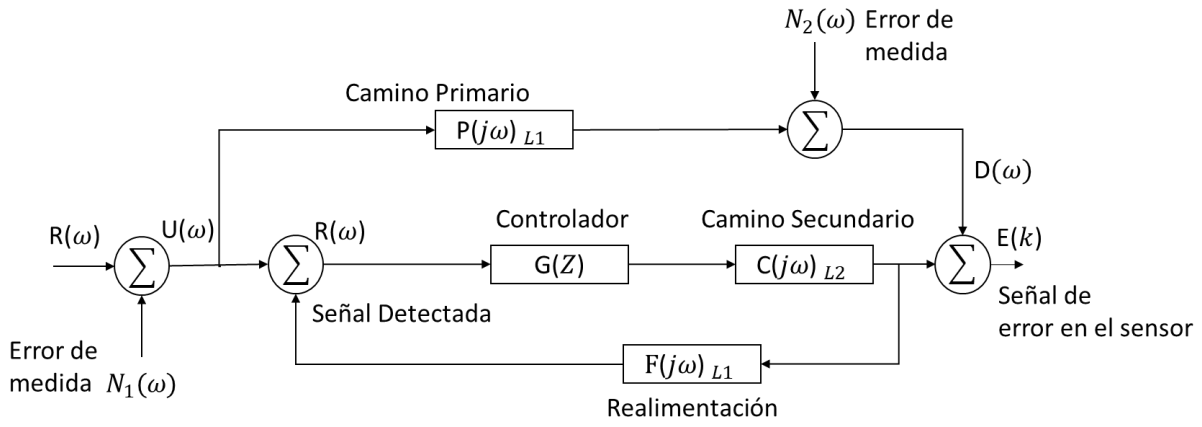


Ilustración 6: (a) esquema cancelación de ruido, (b) diagrama de bloques del sistema de cancelación de ruido. (Imagen Propia)

Las funciones de transferencia de la Ilustración 6b según Nelson y Elliott quedan determinadas como se presenta a continuación:

$G(Z)$: El controlador de ruido discreto, definido por el diseñador del control.

$P(j\omega) = \frac{M_e}{M_r} e^{-jk(L_1-L_2)}$: El camino primario de propagación del ruido a través del ducto.

$C(j\omega) = L_c M_e e^{-jkL_2}$: El camino secundario de propagación de la señal de control.

$F(j\omega) = L_c D_c M_r D_r e^{-jkL_1}$: La realimentación la cual la señal de control reingresa en el sistema.

M_r, M_e: Respuesta en el dominio de la frecuencia de los micrófonos de referencia y de error.

L_c: Respuesta en el dominio de la frecuencia del actuador final de control (Parlante).

L_1: Distancia entre el micrófono de referencia y el actuador.

L_2: Distancia entre el actuador y el micrófono de error.

D_c, D_r: Direccionalidad del actuador y del micrófono de referencia.

1.4.2 Modelado del sistema

De las funciones de transferencia especificadas en la Ilustración 6, se presentan dificultades a la hora de modelarlas, esto se debe a que en un sistema acústico se presenta reflexiones e interacciones con el ambiente circundante de por lo que su comportamiento puede presentar variaciones ante el cambio de ubicación y temperatura. Además, el proceso de modelado requiere espacios especialmente acondicionados para tal fin, espacios que limiten la interacción de los componentes del sistema acústico con el ambiente. De las funciones de transferencia antes mencionadas, son especialmente críticas el camino primario y el camino secundario. El camino primario se compone de un segmento de tubería asociado a L_1 que puede ser fácilmente modelado a partir de los datos técnicos proporcionados por el fabricante. No así para el camino secundario, que se compone del segmento de tubo asociado a L_2 y del parlante que genera la acción de control, dichos parlantes presentan un comportamiento que varía por la dependencia acústica del entorno en la que se realice la reproducción de los sonidos. Ante esta limitación lo más conveniente es realizar un proceso de modelado que no se vea afectado por el entorno, como lo puede ser el modelado numérico por medio del filtrado adaptativo descrito en Filtrado Óptimo Lineal.

La forma de realizar el modelado, que se describe en la Ilustración 7, consiste en "posicionar" un sistema adaptativo discreto $W(Z)$ en paralelo con el sistema físico de interés $W_0(S)$ y una vez hecho esto se excita a ambos sistemas con una señal $v(k)$ que es filtrada en $H(Z)$ para eliminar las frecuencias que no interesan. Tanto $W_0(S)$ como $W(Z)$ darán una respuesta de forma independiente, el propósito es que ambas respuestas sean iguales, por lo que la diferencia de estas $e(n)$ es el error de predicción cometido por $W(Z)$, este error además está contaminado por un ruido N. Como se desea que el resultado sea el mismo se realiza un procedimiento de búsqueda iterativo como el descrito en Gradiente descendiente con el fin de reducir el error. La señal $v(k)$ tiene el requisito de que debe ser variada y debe ser entregada durante un tiempo lo suficientemente largo para garantizar la convergencia. La señal que se adecúa más a este propósito es el ruido blanco gaussiano o distribución estadística normal con media en cero.

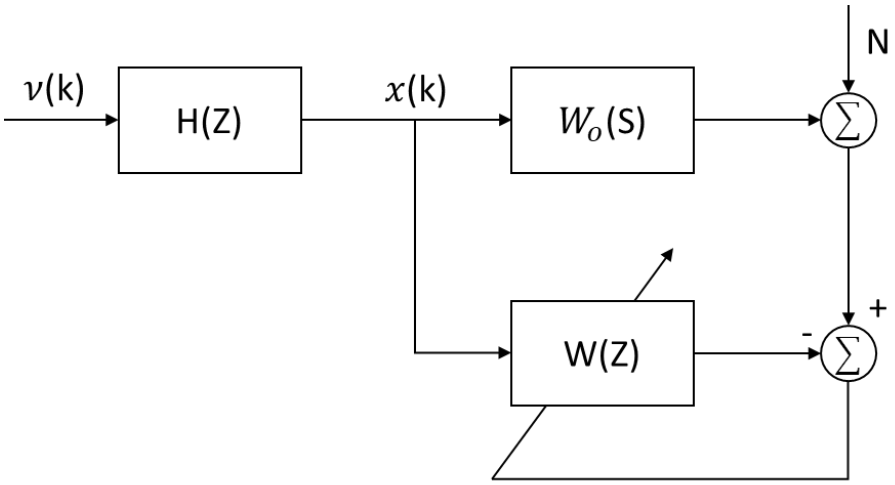


Ilustración 7:Modelado de planta por medio del filtro FIR LMS (Imagen Propia)

1.4.3 Generación de ruido gaussiano por computador

Para conseguir ruido gaussiano se necesita construir un generador de números aleatorios. En internet existen (Damien, 2010) discusiones sobre el tema, en ella se documenta un testeo del desempeño en generación de números de distribución normal en la plataforma de x86 (computadores de escritorio) (Miloyip, 2015) donde se destaca el desempeño del método del algoritmo de Ziggurat (Marsaglia & Tsang, 2000) en la Ilustración 8. Dicho algoritmo se encuentra publicado en C en (Burkardt, 2013).

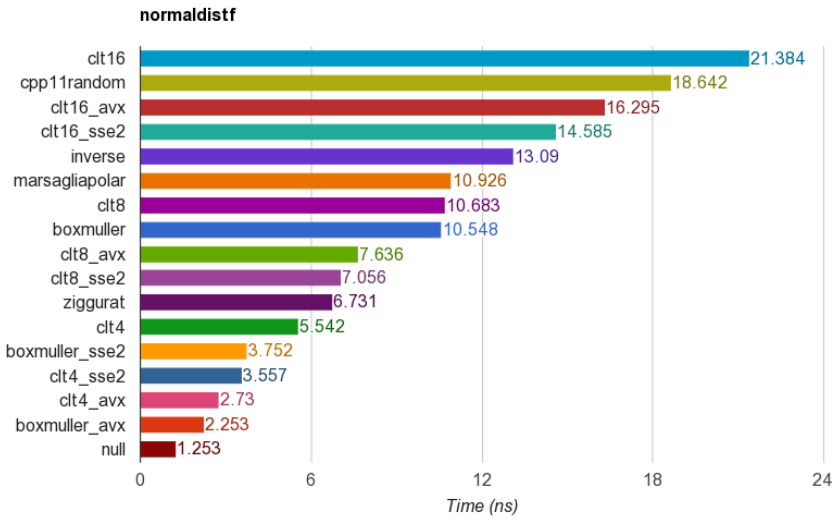


Ilustración 8:Benchmark de generación de números aleatorios (Miloyip, 2015)

1.5 Componentes de un sistema de control de ruido

Un sistema de control activo de ruido se compone de dos subsistemas: uno físico y uno electrónico. El físico se refiere a los elementos acústicos que interactúan entre sí en el proceso de transmisión del sonido, es decir, los transductores que miden la señal, los medios de propagación físicos de las vibraciones, los actuadores que disminuyen el ruido y es la calidad y correcta integración de estos elementos la que limita el desempeño que pueda tener el proceso de control.

1.5.1 Parlantes

Son los componentes encargados de generar una onda de presión que puede ser percibida por el oído humano a partir de una señal eléctrica, en el sistema de cancelación de ruido los parlantes generan el anti-ruido. Los parlantes de uso más extendido son aquellos que funcionan por la interacción de fuerzas magnéticas, una de las implicaciones de su uso es que los parlantes inducen distorsión en el sonido que generan, por lo que aparecen sonidos que en realidad no se están especificando emitir. En el control de ruido la distorsión tiene la consecuencia de que el anti-ruido generado añade ruido por lo que se llega al caso en el cual a pesar de que se registra una reducción de la energía del ruido medido, la percepción de "quietud" no se logra, dada la aparición de dicho ruido adicional. Los parlantes escogidos son parlantes comunes de computador dado su relativo bajo coste y disponibilidad ante la ausencia de guías que informen del requerimiento de un tipo parlante en específico.

1.5.2 Micrófonos

Los micrófonos son un aparato que convierte las ondas de presión sonoras a ondas eléctricas, que es el proceso contrario a los parlantes. Los micrófonos pueden ser clasificados de acuerdo con su direccionalidad (Eargle, 2005), es decir, siguiendo su sensibilidad dependiendo de la dirección del sonido, siendo los patrones más extendidos el omnidireccional y el de cardiode mostrados en la Ilustración 9.

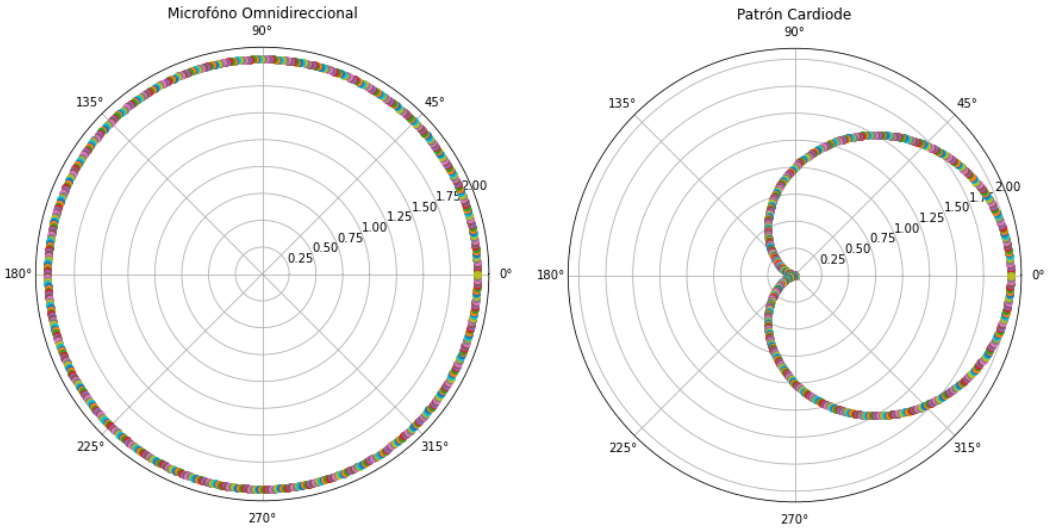


Ilustración 9: Patrones de sensibilidad Omnidireccionales y de Cardiode (Imagen Propia)

La clasificación de los micrófonos también se realiza teniendo en cuenta la tecnología usada para realizar la transducción de sonido a electricidad, se resalta la tecnología de condensador, los dinámicos, piezoeléctricos, etc (Ballou, 2009).

Existe un tipo de micrófonos basados en tecnología de Sistemas MEMS[6], su principio de funcionamiento es el mismo que el de un micrófono de condensador, sólo que el tamaño se disminuye entre 1 y 100 micrómetros. Los micrófonos MEMS tienen la ventaja de ser más ligeros, pequeños y baratos que su contraparte tradicional (Widder & Morcelli, 2014), de allí que se hayan usado en este proyecto. Su estructura se presenta en la Ilustración 10, en ella el sonido ingresa por la apertura inferior izquierda a una cavidad resonante e interactúa con una membrana que se encuentra detrás de una lámina rígida de metal. El sonido cambia la distancia entre la membrana y la lámina, lo que a su vez altera la capacitancia del arreglo lámina-membrana produciéndose una onda de voltaje acorde al sonido que entra al sistema MEMS. Un ASIC[7] se encarga de mantener la carga constante entre la membrana y la lámina y de leer el voltaje.

[6] Micro-Electro Mechanical System o Sistemas Micro-Electro-Mecánicos.

[7] Application Specific Integrated Circuit o circuito integrado de aplicación específica.

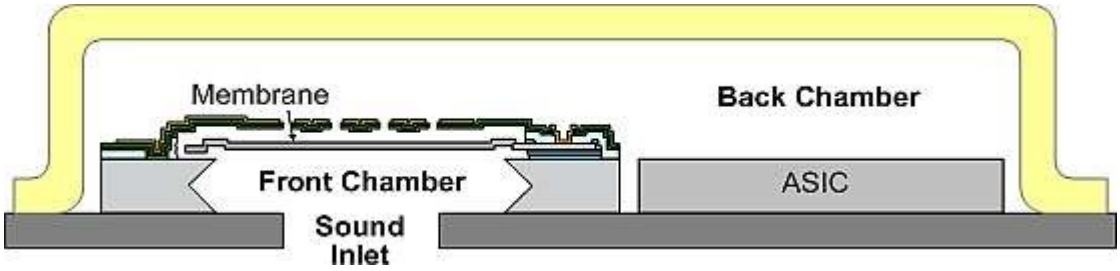


Ilustración 10: Sección Transversal de un micrófono MEMS (Widder & Morcelli, 2014)

El micrófono usado es el SPH0645LM4H-B de la compañía knowles (Knowles, 2015) que se muestra en la *Ilustración 11* y la respuesta en frecuencia de dicho micrófono en la Ilustración 12, en esta última gráfica hay un parámetro que es la frecuencia de reloj, esto es debido a que se está usando un micrófono digital que se comunica por medio del protocolo I2S con palabra PCM[8] de 32 bits y es dependiendo de la frecuencia de este reloj que el micrófono podrá enviar altas frecuencias por el canal de comunicación.

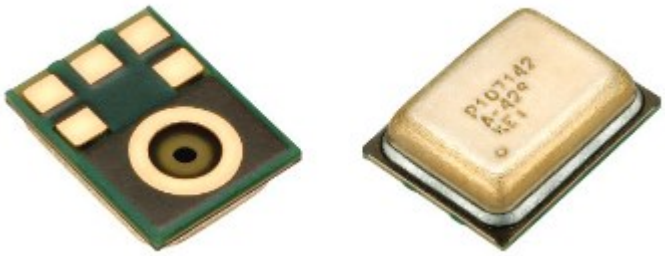


Ilustración 11: Micrófono MEMS SPH0645LM4H-B *(Knowles, 2015)*

[8] Pulsed Code Modulation, es una forma de representar en formato digital variables analógicas.

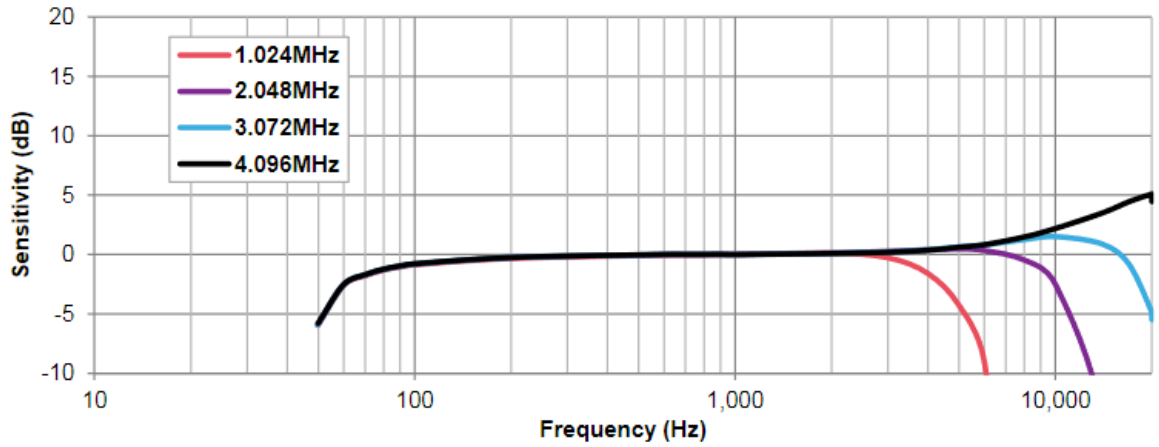


Ilustración 12: Respuesta en frecuencia del micrófono SPH0645LM4H-B (Knowles, 2015)

1.5.3 Medios físicos

Los elementos de propagación de los ductos son emulados mediante el uso de tubo de PVC[9] para cañerías de 2 pulgadas de diámetro, este elemento produce un retardo de propagación entre sus extremos. El otro de los medios físicos usados es la cavidad resonante que incorpora el parlante, el principal requisito que esta tiene es el de evitar que las ondas sonoras se propaguen por el espacio libre y que, en vez de ello, dichas ondas sigan exclusivamente el camino presentado por el ducto, esto se logra mediante el uso de materiales rígidos que presentan una impedancia acústica más grande que la impedancia del tubo, en este caso madera de media pulgada. El sistema acústico puede ser consultado en Sistema Físico.

1.5.4 Dispositivo de procesamiento de la información

También conocido como el computador, es el componente encargado de determinar en un dominio digital la onda de cancelación de ruido. Para este componente existen dos opciones: microcontroladores y DSP. El microcontrolador es el dispositivo de cómputo más simple y su área de aplicación abarca las situaciones que requieren seguir un conjunto de reglas no muy complicadas como la medida de variables físicas e interfaces simples de hardware, sin embargo con el desarrollo de la microelectrónica estos han adquirido una mayor capacidad de cómputo manteniendo su costo, al punto de llegar a desafiar la posición que tienen los computadores embebidos del

[9] Policloruro de Vinilo

siguiente nivel en las áreas de menor demanda, las DSP. Su programación se realiza mediante el lenguaje C, que es un estándar de facto para microcontroladores.

Las DSP, son un dispositivo de cómputo más complejo que el microcontrolador, están hechas con la idea de hacer cálculo numérico a gran velocidad, esto hace que encuentren su área de aplicación en tareas más demandantes que el microcontrolador como el procesamiento de imágenes, video, audio multicanal, radar, el control robótico complejo y otras actividades enmarcadas en el área del HPEC[10] (Wolf, 2014). Con el fin de rendir en estas aplicaciones, este computador hace uso de hardware multinúcleo, mayor frecuencia de reloj y por tanto mayor coste, lo cual desde un principio relega su utilidad un a nicho en donde las tareas son lo suficientemente complejas para justificar su coste y lo suficientemente distantes de un computador central como para validar la necesidad de tener el DSP en el campo. Al igual que los microcontroladores, este dispositivo se programa en C. Dado el objetivo de probar las capacidades de un microcontrolador con los algoritmos de cancelación de ruido que típicamente son ejecutados en DSP, se le hace uso. El microcontrolador usado es el ESP32-WROOM-32U de la compañía china Espressif (Espressif, 2017), con 448 KB de ROM, 512 KB de RAM y operando a una frecuencia de reloj de 240 MHz. Sumado a sus capacidades integradas de comunicación Wi-fi y bluetooth que es relativamente raro que se de en microcontroladores, lo hace idóneo para prototipado que involucre comunicaciones inalámbricas, lo cual ha repercutido en que sea un microcontrolador usado por entusiastas dada su versatilidad y, por tanto, existe una comunidad activa en torno al microcontrolador (Espressif, 2016).

1.5.5 Conversor Digital-Análogo

El microcontrolador hace cálculos en un dominio discretizado y es en ese dominio donde el microcontrolador haya una respuesta matemática para la cancelación de ruido propuesta, sin embargo, para interactuar con variables físicas debe convertir esa solución en una variable física, para esto se hace uso de un conversor Digital-Análogo. Para este caso se usa el *PCM5102A* de la compañía *Texas Instruments* (Texas Instruments, 2012).

[10] High Performance Embedded Computing o computación embebida de alto desempeño

1.5.6 Amplificador de Audio

El amplificador es un componente encargado de dar potencia a una señal para que interactúe con los parlantes y estos a su vez interactúen con los demás sonidos que se propagan por los medios físicos. Se usa un amplificador TDA2030A de la compañía ST Microelectronics (ST Microelectronics, 2000).

1.5.7 Alimentación

En aplicaciones de audio y especialmente en aquellas que se denominan de "alta fidelidad" es requisito que la alimentación de los circuitos sea estable con el fin de evitar distorsión producto de la variación del voltaje de referencia en los conversores Análogo-Digital y Digital-Análogo. La variación del voltaje en una fuente es conocida como rizado y se puede apreciar en la Ilustración 13 en donde el proceso de conversión AC-DC deja una variación en el voltaje DC. Las fuentes DC usadas para audio son del tipo de regulación lineal (Luong, 2018), estas tienden a tener una peor disponibilidad y mayor coste que su contraparte de fuente conmutada. Para subsanar esto se hace uso de baterías de litio 18650[11] de 3.7V con una capacidad de 2000 mAh, lo cual quiere decir que dichas baterías son capaces de suministrar una corriente de 2A durante una hora antes de requerir recarga. En las baterías el rizado tiende a ser menor y este tipo de batería es de amplia disponibilidad dado que se usa en las baterías de computadores.

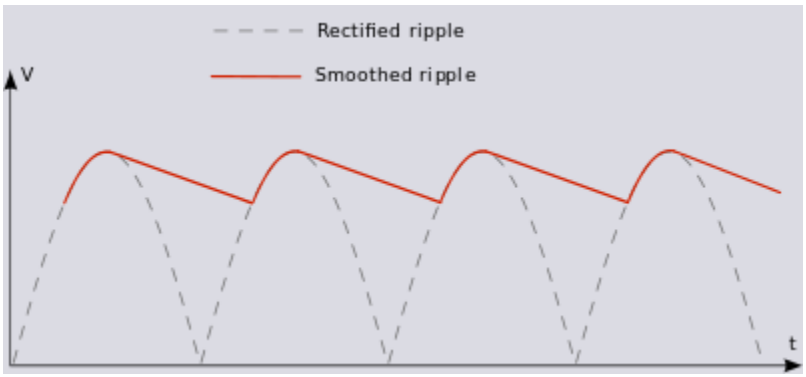


Ilustración 13: Rizado en una fuente de alimentación (Wikimedia, 2008)

[11] Es un cilindro con un ánodo y un cátodo en cada extremo, con dimensiones de 18mm de diámetro por 652 mm de largo

1.5.8 Protocolos de comunicación

La comunicación es uno de los temas esenciales cuando se habla de microcontroladores debido a la necesidad de interactuar con otros sistemas digitales como conversores análogo digital, potenciómetros, memorias, sensores inerciales, otros microcontroladores/computadores, etc. La comunicación por lo general sucede entre dos o más circuitos con un poder dispar: como maestro, es decir, solicitando información o como esclavo, respondiendo a las solicitudes del maestro. Dichas comunicaciones se llevan a cabo por medio de protocolos seriales que contrastan con los protocolos paralelos en que todas las señales que pudieran estar en el protocolo van encaminadas a enviar un solo dato a la vez, a continuación- se detallan los protocolos considerados para el proyecto de investigación.

1.5.8.1 IIS

De las siglas Inter Integrated Sound (I2S), es un protocolo de comunicación síncrono ideado por Philips Semiconductors (hoy NXP Semiconductors) en 1986 y con su última revisión en 1996, para la comunicación de audio PCM entre dispositivos embebidos en los albores de la era del CD (Philips Semiconductors, 1996). La mayor desventaja que posee este protocolo es que dada su especificidad de propósito, no está tan ampliamente implementado como el protocolo UART, I2C o SPI. Una señal típica de I2S se compone de 3 señales mostradas en la Ilustración 14.

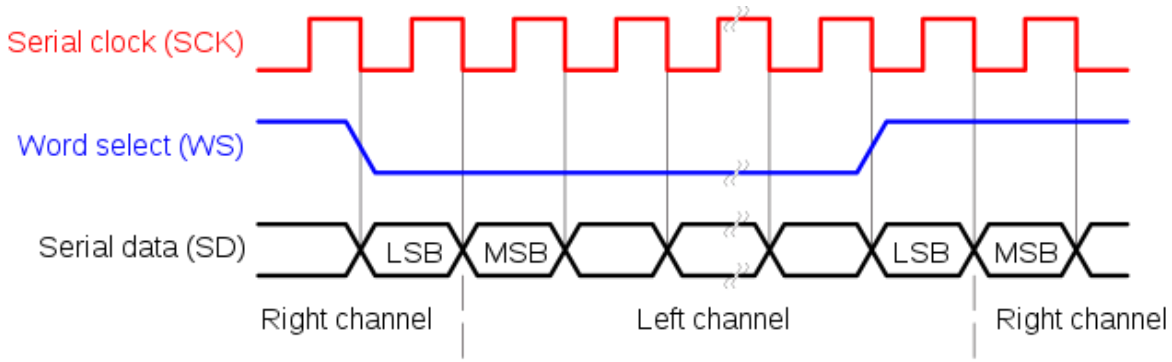


Ilustración 14: Señales del protocolo de comunicación I2S (I2S Timing, 2011)

El reloj serial es quien da los tiempos en esta comunicación síncrona, Word Select indica a que canal, si izquierdo o derecho, pertenece la muestra PCM, mientras que Serial Data son los datos en sí que se están transmitiendo.

1.5.8.2 SPI

Del inglés Serial Peripheral Interface (Motorola, 2000), es un tipo de comunicación síncrona de 4 señales que se implementa para una alta tasa de datos, corta distancia y comunicación bidireccional, introducida en 1979 por motorola, está pensado para la comunicación de un maestro con un esclavo. Con las 4 líneas de datos descritas en la Ilustración 15.

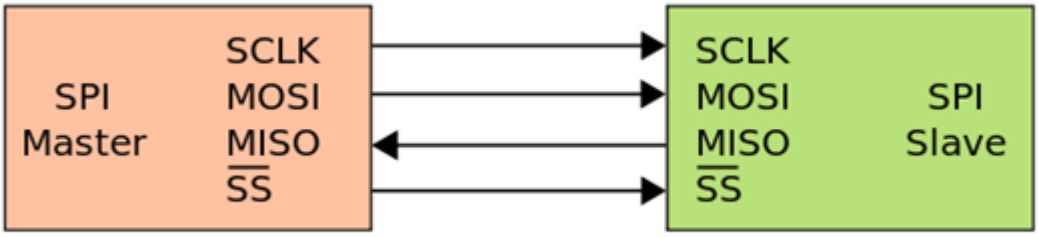


Ilustración 15: Diagrama Conexión SPI (Wikimedia, 2006)

Las señales de SPI se describen a continuación:

- SCLK: Reloj que sirve de compás en la comunicación.
- MOSI: *Master Out Slave In*, línea que lleva los datos desde el maestro a el esclavo.
- MISO: *Master In Slave Out*, línea que comunica datos desde el esclavo al maestro.
- SS: *Slave Select*, Línea que le indica a el esclavo que la información que se encuentra en la línea MOSI está dirigida a este esclavo en particular, puesto que está puede ser compartida con otros esclavos.

1.5.8.3 UART

Por sus iniciales *Universal Asynchronous Receiver-Transmitter*, es un tipo de comunicación diseñada para que los computadores fueran capaces de transmitir información al exterior, para el caso de los computadores y microcontroladores existe la posibilidad de añadir un dispositivo de conversión a un formato de voltaje robusto como el RS-232 o RS485 definidos en los estándares ITU-T V.24 (International Telecomunications Union, 2000) y EIA-485 (Soltero, Zhang, & Cockril, 2002).

1.6 Desempeño de un sistema ANC

Con el fin de determinar el desempeño del sistema de cancelación activa de ruido se hace uso de la medida de decibelios dada la naturaleza logarítmica de la percepción del sonido, tomándose como 0 dB la cantidad de 20 micro Pascales según la norma ISO 226:2003 (ISO/TC 43, 2003) para medidores de sonido, la expresión de decibeles queda expresada en la Ecuación 8 a partir del valor cuadrático medio. Si el sistema de cancelación de ruido es de naturaleza lineal, basta con tomar la lectura en decibelios promediada en todo el espectro o bien hacer uso de uno de los ponderados en frecuencia que existen definidos en el estándar internacional IEC 61672-1:2013 (IEC, 2013). Siendo el tipo de ponderación-A (A-Weighted) o dB(A), el más ampliamente aceptado por las autoridades dado que su formulación se basa en la respuesta aproximada del oído humano ante los diferentes tonos que no es uniforme para cada tono, sino que la sensibilidad del oído depende de la frecuencia, siendo en este ponderado más preeminentes las frecuencias altas que las bajas. Esta medida toma el volumen total y resulta en una medida en el tiempo. Sin embargo, como el comportamiento de los sistemas acústicos es inherentemente no lineal resulta que estos no tratan las frecuencias de la misma manera y la medida de decibelios se queda corta para mostrarlo porque es un solo número, por lo que se plantea hacer uso del espectrograma para dar cuenta de comportamientos no lineales del sistema que pudieran ocurrir. Este se sirve de la FFT[12] para dar cuenta del tratamiento individual y diferenciado que un proceso no lineal da a cada rango de frecuencias, como se puede ver en la Ilustración 17, donde en la parte superior se encuentra una grabación de audio en el tiempo mientras que en la parte superior se encuentra un espectrograma donde el eje horizontal es el tiempo y el vertical es la frecuencia en medio de estos dos ejes se encuentra codificada la intensidad de cada frecuencia por medio de un mapa de color, donde en la medida en que la intensidad es más alta en decibelios, también es más intenso el color.

$$dB = 20 * log(RMS);\ RMS = \sqrt{\frac{x_1^2 + x_2^2 + \ldots x_n^2}{n}}$$

Ecuación 8: Conversión a decibeles (Price, 2007)

[12] Fast Fourier Transform o transformada rápida de Fourier

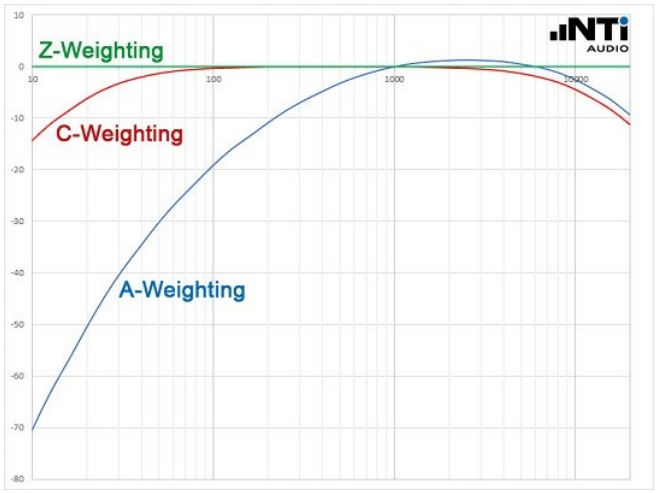


Ilustración 16: Ponderación de frecuencias dB(A), dB(C) y dB(Z) (NTi Audio, 2020)

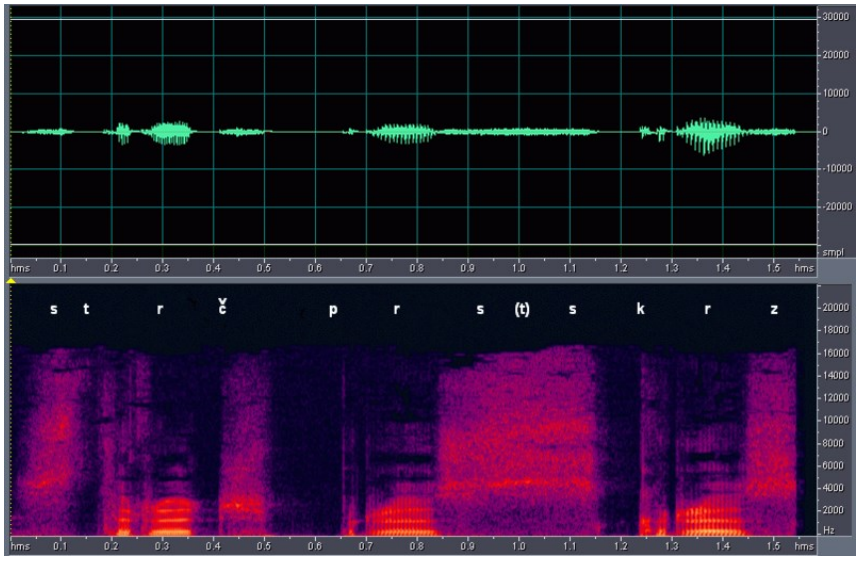


Ilustración 17: Espectrograma (Wikimedia, 2009)

1.7 Algoritmos para un sistema ANC

1.7.1 Fx-LMS

El filtrado Wiener-Hopi (Farhang-Boroujeny, 2013) hace uso de estadísticas de la señal a la que se desea adaptar con el fin de estimar la siguiente muestra. Sin embargo, tiene la limitante de que se requiere conocer a priori dichas estadísticas, lo que hace irrealizable dicho filtro. Por otro lado, el filtro de mínimos cuadrados o LMS realiza un método de búsqueda iterativo con el fin de minimizar el error, este se basa en estimar el gradiente del error y tomar el camino descendente. Dicho error se define por medio del valor esperado del error cuadrático $\xi = E^2(e(k))$ hallado, con el fin de simplificar se dice que el valor

esperado es el mismo valor encontrado o matemáticamente hablando $\xi = E(e^2(k)) \cong e^2(k)$, siendo el error $e(k) = p(k) + s(k)$ con $p(k)$ como la muestra encontrada por el sistema y $s(k)$ la muestra generada por el sistema adaptativo. La Ilustración 18 muestra el sistema de cancelación de ruido para un ducto de ventilación con el filtro LMS tal y como finalmente se plantea en el experimento. El gradiente del error queda expresado en la Ecuación 9 si se da el caso de que la aproximación $f(k) \cong s(k)$ y que $\hat{C}(Z) \approx C(S)$.

$$\Delta\xi = \frac{\partial e^2(k)}{\partial \boldsymbol{w}} = 2e(k)\boldsymbol{f}(k)$$

Ecuación 9: Gradiente del filtro LMS

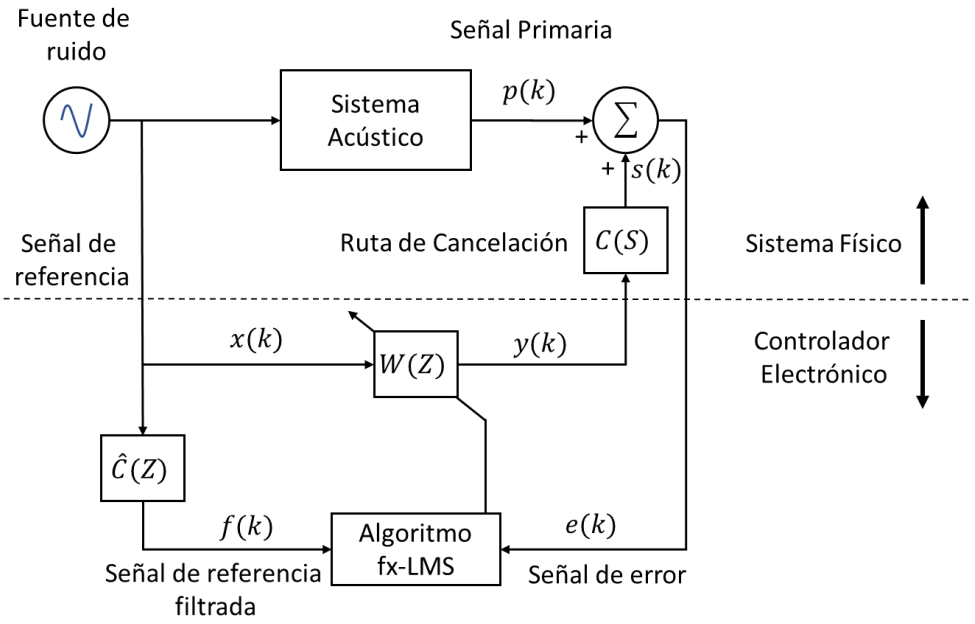


Ilustración 18: Algoritmo de cancelación de ruido fx-LMS (Imagen Propia)

En relación con la Ilustración 6, el ruido que se propaga por el camino primario o sistema acústico y es W(Z) quien predice el negativo del ruido que pasa por dicho canal, reemplazando a G(Z), mientras que la ruta de cancelación C(S) es modelada por su aproximado $\hat{C}(Z)$ por medio del procedimiento mostrado en Modelado del sistema. Por otro lado, la realimentación en el tubo es ignorada, esto es razonable en el caso de que los extremos donde se genera sonido se encuentren en una cavidad sellada o también es razonable en el caso de que la separación L_1 sea lo suficientemente grande como para atenuar el sonido de feedback que se devuelve.

Dicha implementación es conocida en la literatura como algoritmo filtrado LMS o fx-LMS debido a que la actualización de los coeficientes del filtro $W(Z)$ se realizan teniendo en cuenta la aproximación del camino secundario en vez de la señal de referencia $x(k)$ como se haría si fuera un filtro LMS, siendo su ecuación de adaptación de pesos la dada en la Ecuación 10, donde μ es la tasa de aprendizaje. El filtrado adaptativo propuesto por el filtrado Fx-LMS tiene la ventaja de que el sistema resultante mantiene tolerancia ante cambios en los parámetros del sistema como el envejecimiento de los componentes y el cambio de la temperatura.

$$W(k+1) = W(k) - 2\mu e(k) f(k)$$

Ecuación 10: Actualización de pesos del filtro fx-LMS

1.7.2 Control de ruido con Perceptrón Multicapa

Se plantea un segundo algoritmo para el control de ruido haciendo uso de redes neuronales, en relación al algoritmo Fx-LMS, el perceptrón supone una mejora tanto en desempeño de cancelación como en el hecho de que puede dar cuenta de relaciones no lineales entre el ruido que se detecta y el que finalmente llega al punto de cancelación, aunque como lo reporta (Hansen, Snyder, Xiaojun, Brooks, & Moreau, Neural Network-Based Feedfoward Active Control, 2012), el desempeño puede ser mayor sin embargo es inestable y en un momento dado puede desmejorar de forma repentina.

El perceptrón multicapa (Rebala, Ravi, & Churiwala, 2019) es un arreglo de neuronas artificiales como el mostrado en la Ilustración 19, donde a cada columna de nodos se le llama capa, a la roja se le llama de entrada e interactúa con los datos de entrada, a la azul se le llama oculta, puede conformase de varias capas y es el procesamiento interno de información, mientras que la última capa (verde), interactúa con el exterior de la neurona, de ahí su nombre de salida. La principal diferencia entre las capas es la naturaleza de la información que tratan, mientras que las capas de entrada tratan con variables con significado concreto (voltaje, probabilidad, clasificación) la capa oculta trabaja con abstracciones numéricas de la red neuronal. Una neurona, que corresponde en la ilustración a cada nodo de la red, es un arreglo de entes matemáticos llamados entradas $\boldsymbol{x}_n$, pesos $\boldsymbol{w}_i$, sesgo b (también considerado una entrada constante como $\boldsymbol{x}_{n+1}$), función de activación f y salida u, arreglo mostrado en la Ilustración 20.

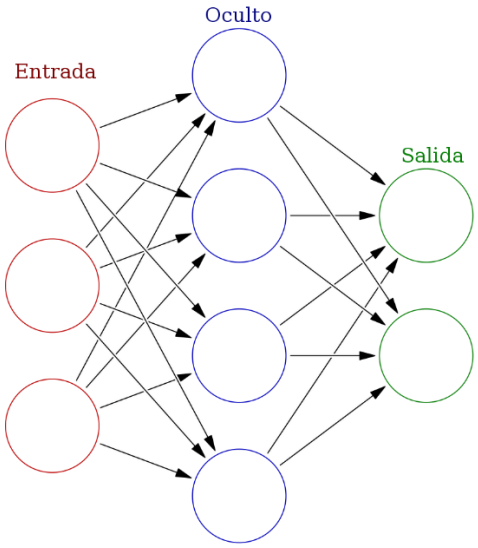


Ilustración 19: Perceptrón Multicapa (Wikimedia, Colored neural network, 2019)

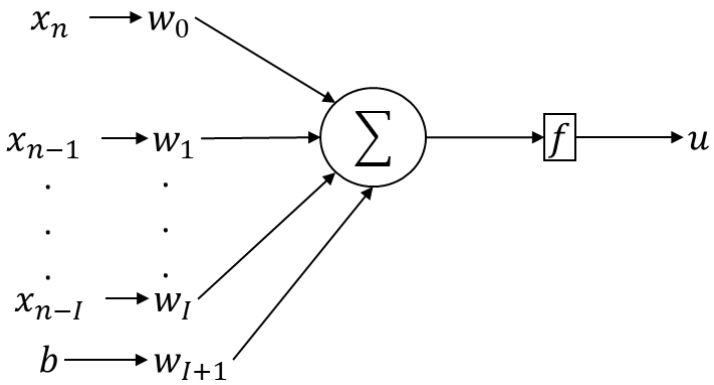


Ilustración 20: Neurona Artificial (Imagen Propia)

Para una sola neurona, la salida viene dada por la Ecuación 11, la función de transferencia f junto con el sesgo (o en inglés bias) b son los componentes que le dan un comportamiento no lineal a la neurona artificial, emulando dicha característica de las neuronas orgánicas. El sesgo como una constante que aparece sin importar la entrada y la función de transferencia se utiliza para hace una relación no lineal entre los valores de entrada y de salida. Una de las relaciones más comúnmente usadas es la sigmoide expresada en la Ecuación 12.

$$y_k = f(\sum_{i=0}^{I+1} w_i \cdot x_i)$$

Ecuación 11: Relación entrada-salida en neurona artificial

$$S(x) = \frac{1}{1 + e^{-x}}$$

Ecuación 12:Función Sigmoide

Se deja como nota que, con una función de activación lineal, sin el sesgo, con cada entrada siendo una muestra retrasada *i* ciclos de reloj y con una neurona se tiene exactamente la misma definición del filtro LMS. Por lo que es de concluirse que la neurona es una generalización no lineal del algoritmo Fx-LMS cuando se aplica al modelo del ducto de ventilación de la Ilustración 18. Para hacer uso de un controlador activo de ruido neuronal, el diagrama de bloques del sistema de la Ilustración 18 no se ve fuertemente modificado, salvo por el hecho de $W(Z)$ se ve reemplazado por la red neuronal y las reglas de actualización del filtro LMS se ven reemplazadas por el desarrollo mostrado por (Zhoua, Zhanga, Lib, & Gan, 2004).

El uso del algoritmo neuronal, en especial el proceso de adaptación tiende a ser intensivo en recursos de máquina, como se plantea el uso de un microcontrolador se recurre a los trabajos de (Vanhoucke, Senior, & Mao, 2011) y de (Zhao, Liu, & Li1, 2020) que proponen el uso de aritmética de punto fijo en vez de aritmética de punto flotante para entrenar la red neuronal, la aritmética de punto fijo se establece mediante la cuantización de los pesos de 8 bits y con la función de activación ReLU cuantizada y limitada.

1.8 Consideraciones de programación del microcontrolador

Cuando se realiza la programación de un microcontrolador, hay que atender realidades propias de su constitución como la representación de los números, que dejan de ser definiciones en papel y pasan a ser cuantizaciones que corren dentro del computador. La frecuencia de muestreo del sistema se debe definir atendiendo limitaciones impuestas por los parlantes, los micrófonos y los límites de convergencia del algoritmo LMS. Finalmente se define la forma en la cual se ejecutan las secciones de código en el microcontrolador, por medio de un sistema operativo para microcontroladores.

1.8.1 Tipos de datos discretos

Por convención, todos los microcontroladores tienen acceso a dos formas de representación de números, el uso de una unidad entera que es imprescindible para que el mismo pueda funcionar y una unidad flotante que no siempre está incluida en el hardware y que se usa para cálculos matemáticos sobre variables

de números reales, existen compromisos al hacer uso de una u otra representación que son presentados a continuación.

1.8.1.1 Formato de Punto Fijo

Las operaciones de punto fijo (Yates, 2020) son un tipo de operación que se puede emular fácilmente mediante el uso de la unidad de enteros del microcontrolador, puesto que dichas operaciones generalmente no son nativamente soportadas por el mismo, su principal característica es que la coma binaria queda fija en un punto del código tal y como aparece en la Ilustración 21 y no cambia a lo largo de la ejecución del código, con el fin de que un microcontrolador implemente punto fijo se requiere hacer consideraciones de truncamiento y saturación en el software.

Ilustración 21: Tipo de dato de Punto Fijo (Wikimedia, 2008)

La saturación (Yates, 2020) sucede de la propiedad inherente de la suma de que, en el peor caso, la suma de dos operandos se desborda en un dígito o en un bit tal y como se muestra en la Ilustración 22a, suponiendo representación sin signo y en donde a falta de lugar para representar el bit 16 se debe desechar el acarreo y la suma de dos números da como resultado un número inferior a cualquiera de los dos sumandos. La mejor forma de minimizar el error es dando el resultado como el mayor número representable por el conjunto de bits tal y como sucede en la Ilustración 22b. El diagrama de flujo de la decisión se encuentra en la Ilustración 23, aclarando que && representa la conjunción.

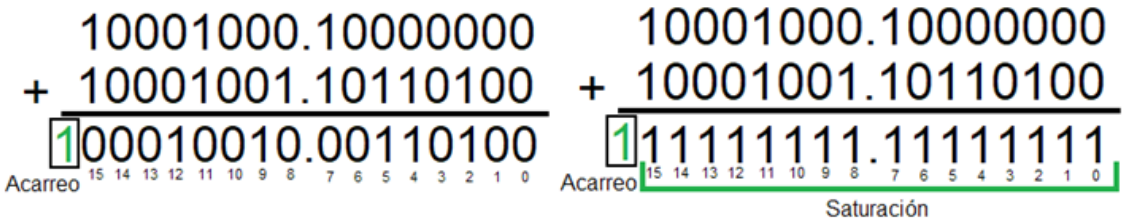


Ilustración 22: (a) Situación de saturación y (b) Corrección (Imagen Propia)

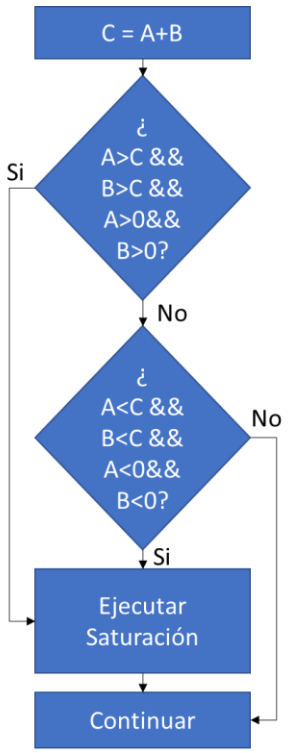


Ilustración 23: Diagrama de flujo para saturación (Imagen Propia)

El truncamiento (Yates, 2020) sucede en la operación de multiplicación, donde también hay que verificar la situación de saturación, en la Ilustración 24a se identifica claramente que la multiplicación requiere de truncamiento puesto que en los 4 bits definidos no se representa la fracción .11, entonces lo que se hace es que se suma al bit menos significativo un 1, quedando como en la Ilustración 24b.

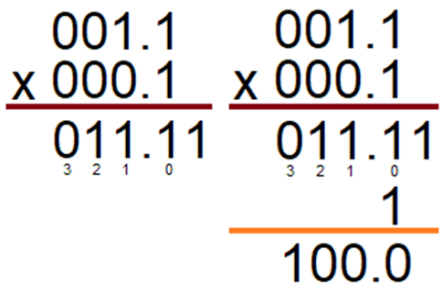


Ilustración 24: (a) Situación de Truncamiento (b) Implementación de truncamiento (Imagen Propia)

El diagrama de flujo que describe el proceso que se incorpora a continuación en la Ilustración 25, en este caso N representa el número de bits de fracción, "&"

representa a la operación lógica "AND" bit a bit y "<" representa a la operación corrimiento de bits a la izquierda.

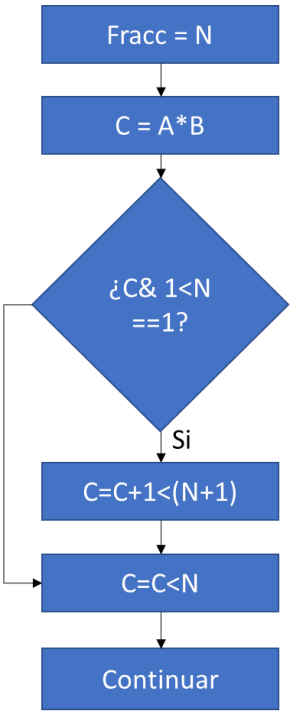


Ilustración 25: Diagrama de flujo de Truncamiento (Imagen Propia)

El formato de punto fijo tiene la ventaja de que es implementable de forma universal para todos los microcontroladores con una bajada en el rendimiento mínima correspondiente a la implementación de las correcciones de saturación y de truncamiento, a diferencia del punto flotante que no está implementado universalmente.

1.8.1.2 Formato de Punto Flotante

El tipo de dato de punto flotante usado por la mayoría de los computadores de hoy en día siguen el estándar IEEE-774-2008 (IEEE, 2008), que a la fecha fue reemplazado por la revisión de 2019, sin embargo, esta es una revisión menor que realiza aclaraciones y ampliaciones en vez de cambios, por lo que su consulta sigue siendo válida para los efectos de este estudio. Dicho estándar define varios tipos de datos flotantes con base en el número de bits que el dato ocupa, en este caso es de interés solamente el tipo de datos de 4 bytes o también conocido como de 32 bits, este es el que los microcontroladores generalmente implementan. Los números flotantes son una discretización en números binarios de la representación de notación científica y de los 32 bits disponibles se hace

uso de 24 bits para el significante, es decir, el número con el signo y se usan 8 para el exponente como se muestra en la Ecuación 13.

$$[24\ bits] \cdot 2^{8\ bits}$$

Ecuación 13: Representación IEEE-774 de 4 bytes

En términos generales los tipos de datos flotante tiene sus ventajas al ser capaz de trabajar sobre valores más grandes o infinitesimalmente más pequeños que la representación de punto fijo, sin embargo, el hardware necesario para lograr las operaciones de punto flotante tiende a ocupar mucho más espacio que el hardware para implementar las operaciones enteras, esto repercute en que la unidad de enteros tiende a ser mucho más rápida que la unidad para operaciones de punto flotante y el uso de esta penaliza más fuertemente el consumo de energía que con la unidad de operaciones enteras.

1.8.2 Frecuencias de interés

Ya se sabe que el sistema de cancelación de ruido está apuntando hacia las bajas frecuencias, sin embargo, existen diferentes definiciones de lo que es una baja frecuencia. Estas definiciones se pueden basar en la música, en las capacidades del sistema y en las capacidades de cancelación pasiva del sistema que va a intervenir. La Ilustración 26 muestra los rangos de frecuencias de diferentes instrumentos musicales y define a las bajas frecuencias entre los 20Hz y 300Hz mientras que (Leventhall, 2004) establece el ruido de baja frecuencia entre los 10Hz y 200Hz para las personas. Por otro lado, desde el punto de vista del sistema ANC, el sistema es más exitoso en la medida que este limite la energía del ruido y una regla de la experiencia dice el sobre muestreo de una frecuencia puede llegar a limitar el desempeño del sistema (Hansen, Snyder, Qiu, Brooks, & Moreau, 13.3.1.1 Sample Rate Selection, 2012), por lo que el rango de tonos que se puede cancelar es de 3 octavas o que el rango de frecuencias es $(f_{max}/8, f_{max})$, adicional a esto la respuesta del micrófono a usar es bastante recta, según la Ilustración 12, entre los 100Hz y 6000Hz, +-1dB de sensibilidad y por último están los parlantes, actuadores que limitan la capacidad de cancelación, sin embargo, la experiencia dice que los parlantes de computadores de consumo de este tamaño tienen una frecuencia de respuesta desde los 100-150Hz. Haciendo la última consideración, que radica en que en el sistema no hay cancelación de ruido pasiva y que, al ser un parlante el productor de ruido primario, puede ser regulada su contenido de frecuencia a voluntad por parte del investigador. Con estas consideraciones se define una banda entre los 125Hz y 1000Hz para la cancelación.

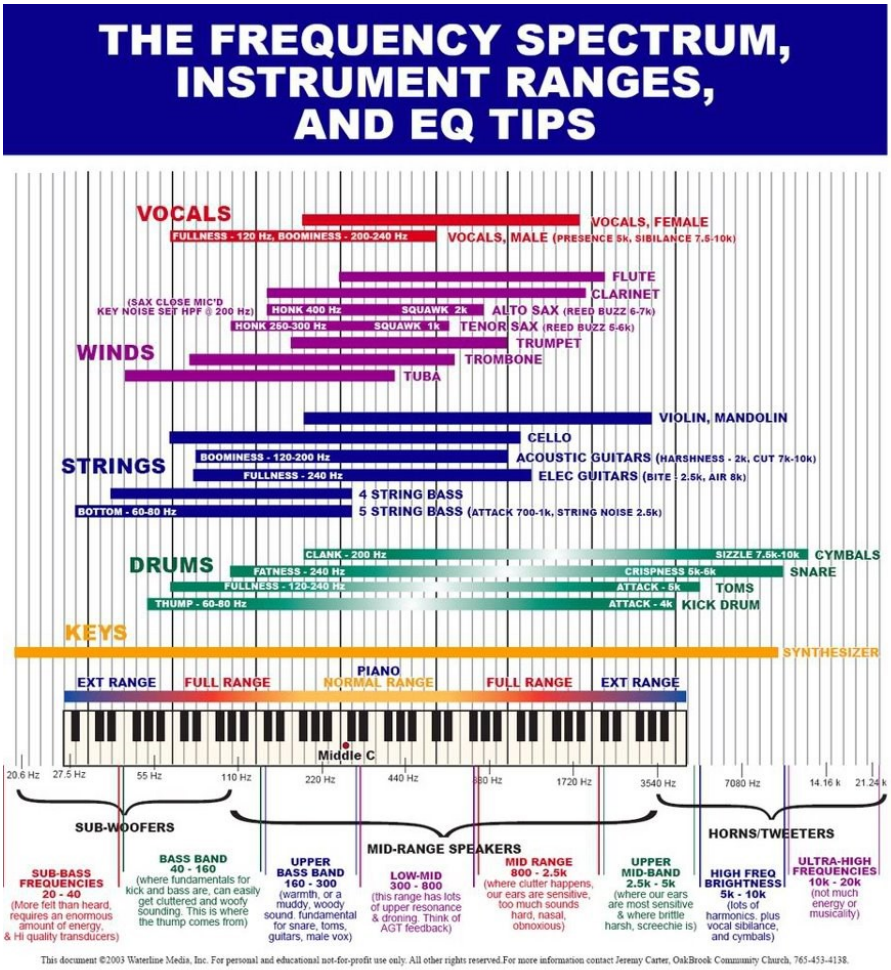


Ilustración 26: Diferentes rangos de frecuencia de los instrumentos musicales (Waterline Media, 2003)

1.8.3 FreeRTOS

FreeRTOS™ (FreeRTOS, 2011) es un sistema operativo de tiempo real para sistemas embebidos distribuido bajo la licencia de código abierto MIT. El sistema operativo está hecho para ser "Pequeño, simple y de fácil uso", el kernel ocupa típicamente entre 6KB y 12KB en el binario y provee una fácil actualización a SafeRTOS, un sistema operativo de tiempo real de pago certificado para trabajo en el sector Industrial, médico y automovilístico bajo condiciones de misión crítica. La principal función del FreeRTOS es administrar la ejecución y comunicación de los procesos de código del microcontrolador, así mismo como de su memoria y los periféricos que este tiene. Aventaja a la programación sin sistema operativo de tiempo real en el hecho de que le ahorra al programador la tarea de administrar el hardware del microcontrolador al ofrecer una capa de

abstracción. FreeRTOS está soportado por el entorno de desarrollo de la mayoría de los microcontroladores, incluido el ESP32, por lo que fue incluido en el proyecto. (Svec, 2012) ofrece un capítulo entero dedicado al tema de FreeRTOS y es accesible libremente a través de la web.

2 Metodología

Este capítulo trata acerca del proceso de diseño y desarrollo del experimento del sistema de cancelación de ruido. Inicia con la búsqueda de unos datos apropiados para tener de referencia y que debieron ser preprocesados, luego se codificó el algoritmo Fx-LMS y el de modelado para después definir unas tareas para el microcontrolador, se prosiguió con un esquema de cableado, se probaron los componentes individualmente y finalmente se evaluó la factibilidad del sistema.

2.1 Base de datos y preprocesado

Con el fin de garantizar la repetibilidad de los resultados fue necesario tener unas ondas de ruido de referencia, para ello se buscaron entre las diferentes bases de datos sonoras documentadas en internet. Los criterios de búsqueda empleados fueron la presencia de ruido tanto periódico como no periódico, una buena accesibilidad a los contenidos de la base de datos por parte de terceros, grabaciones concordantes con ruido ambiental, y el número y duración de las pistas de audio ofrecidas. La base de datos que más se ajustó a los criterios antes mencionados es "*The Diverse Environments Multi-channel Acoustic Noise Database*" (DEMAND) (Zinemanas, Cancela, & Rocamora, 2017). En ella se detallan 18 grabaciones de ruido ambiental de 15 minutos como el que se puede encontrar al lado de un rio, en una cafetería, en medio de la calle, en un restaurante, una plaza, dentro de un carro, etc. Dichas grabaciones fueron recabadas por medio de un arreglo de 16 micrófonos omnidireccionales en audio PCM de 16-bits, y a dos tasas de muestreo 16KHz y 48KHz.

Entre las alternativas que se exploraron está NOISEX-92 (NOISEX-92, 1996), donde se graba el ruido de una fábrica, el sonido de la cabina de un jet, de un tanque, de un automóvil, de la sala de máquinas de un navío de guerra, etc. La grabación es hecha en calidad CD y a pesar de que la base de datos es citada en la bibliografía como un punto importante de referencia, su distribución original es por medio de dos CDs y, por tanto, de acceso limitado para las condiciones del proyecto de investigación. Es de anotar que, aunque dicha base de datos se puede encontrar gratuitamente en repositorios de internet (speechdnn, 2018) (Acousteen, s.f.), la licencia de uso con la que se distribuye no queda para nada clara, puesto que en la página principal del *dataset* se da a entender que todavía se distribuye.

Otra alternativa a disposición es la iniciativa QUT-NOISE-TIMIT (Dean, Sridharan, Vogt, & Mason, 2010), en ella se detallan grabaciones de ruido ambiental enfocado a situaciones de reconocimiento de voz, son 20 grabaciones de duración variable, entre 40-60 minutos con las grabaciones realizadas con una profundidad de 16 Bits y a una tasa de muestreo de 48KHz. En ellas se detallan situaciones de una sala de estar, en un carro, en la calle, en un café, en una piscina, etc.

También existe Aurora (Hirsch & Pearce, 2000), es una iniciativa del instituto europeo de estándares de telecomunicaciones para estandarizar la extracción de características en el habla. Los datos generados de dicha iniciativa son útiles para la cancelación de ruido puesto que incluyen ruido ambiental en diversos escenarios. Siendo la versión más útil Aurora-2, donde se incluye ruido de aeropuerto, estación de trenes, una exhibición, etc.

CHiME (Foster, Sigtia, Krstulovic†, Jon, & Plumbley, 2015)es otra base de datos en el área del reconocimiento de habla enfocado al reconocimiento del habla bajo ruido ambiental, allí se encuentran 6137 grabaciones de 4 segundos en donde se acompaña a una grabación de sonido de habla con ruido ambiental, en calidad 16-Bit/48KHz y finalmente se consideró a MAVD-traffic dataset (Zinemanas, Cancela, & Rocamora, 2017) que alberga grabaciones de ruido de tráfico en diferentes entornos urbanos de la ciudad de Montevideo (Parque, Residencial, Comercial) y con el etiquetado de la presencia de los diferentes componentes del ruido del tráfico (Motor en ralentí, motor acelerando, cambios de velocidad, frenos) de tres tipos de vehículos (Moto, Carro y Bus), las grabaciones fueron realizadas a 24-Bits/48KHz, cada una en torno a los 5 minutos.

El uso de DEMAND obedece a que ofrece variedad de ruido ambiental propio del que se puede encontrar en entornos cotidianos y una longitud apropiada para las pruebas que se plantean, es decir, menor a media hora y mayor a 20 segundos. Y dado que se encuentra bajo la licencia *Creative Commons Atributtion 4.0 International*, lo hace apropiado para los propósitos del proyecto en cuestión por las libertades de distribución que otorga. Para su uso se toman únicamente las grabaciones del primer micrófono del arreglo de grabación.

Para las métricas se hizo uso de decibelios sin peso alguno, puesto que el propósito es conocer la energía restante de la onda sonora, y no medir los posibles efectos en la salud humana del ruido resultante del proceso de cancelación de ruido. Para diferenciar el tratamiento de las diferentes frecuencias, se hizo uso del espectrograma usando la medida de dB.

El preprocesado de las muestras de audio incluyeron la normalización a -1 dB de las grabaciones a usar que son detalladas en la Tabla 1. Se usó la versión de 16KHz que fueron diezmadas a una tasa de 8KHz posteriormente a la normalización. Adicionalmente se incluyó una onda seno de 250Hz a los datos con el fin de probar el desempeño del sistema en presencia de un solo tono.

Grabaciones	Descripción
DKITCHEN.wav	Inclusión de sonidos no periódicos
NRIVER.wav	El ruido es periódico con similaridad al ruido rosa
PCAFETER.wav	Ruido de fondo periódico
STRAFFIC.wav	Ruido periódico y no periódico
TBUS.wav	Ruido periódico y no periódico
METRO.wav	Ruido periódico y no periódico
Sine.wav	Muestra desempeño lineal del sistema

Tabla 1: Grabaciones usadas para el proyecto de cancelación de ruido

Las grabaciones fueron recortadas a los primeros 18 segundos con el fin de realizar un proceso de descripción, limitar el tiempo de ejecución del algoritmo y el análisis respectivo que sucede después de filtrar el ruido. DKITCHEN inicia en silencio, en torno al segundo 13 se abre una puerta y en torno al segundo 17 abre una alacena. *NRIVER* presenta el sonido propio de un rio con eventos en el segundo 9 y 17 de movimiento del micrófono ya sea por parte del viento o por parte del grabador. *PCAFETER* revela el sonido de fondo de una cafetería concurrida, en ella las personas hablan y es posible distinguir el espacio entre las palabras. *STRAFFIC* muestra el sonido de una calle y desde el segundo 3 hasta el 14 se percibe el paso de un camión que acelera hacia el inicio y que se estabiliza en las cercanías del segundo 9. TBUS se graba en el interior de un bus, sin embargo, en esta grabación prevalece el sonido de las voces de los pasajeros, en los segundos 5, 8,11 y 15 suena la bocina de un carro y a lo largo de la grabación un hombre y una mujer dialogan. En *METRO* pasa en el primer segundo que se abren las puertas de un metro y en el segundo, en el segundo 4 un menor de edad habla y su madre le contesta en el segundo 8, en el segundo 12 suena un pitido que indica el cierre de puertas y en el segundo 16 se cierran las puertas. *Sine* es un simple tono de 250Hz, se considera de baja frecuencia y no hay eventos o un entorno ambiental en la grabación.

El proceso de diezmado, mostrado en la Ilustración 27, es el procedimiento mediante el cual se cambia la tasa de muestreo de una señal a otra tasa de muestreo menor, consiste en recabar los datos a diezmar, se determina la frecuencia de diezmado y la frecuencia original de la señal. Primero se determina

si la frecuencia original es divisible por la frecuencia de diezmado, si esto no es posible, se debe realizar un proceso de interpolación y sobre-muestreo con el fin de expandir los datos originales a una frecuencia que sí cumpla el criterio de divisibilidad, una vez hecho esto se realiza un filtrado para eliminar las frecuencias no representables por la frecuencia de diezmado, esto se hace debido a que de no hacerse, los contenidos de frecuencias superiores tomarían la identidad de las frecuencias inferiores en un fenómeno conocido como Aliasing (Oppenheim, 1975) y una vez hecho esto se realiza un proceso de submuestreo donde se eliminan las muestras de tal forma que solo se conserven las muestras correspondientes a la frecuencia de diezmado.

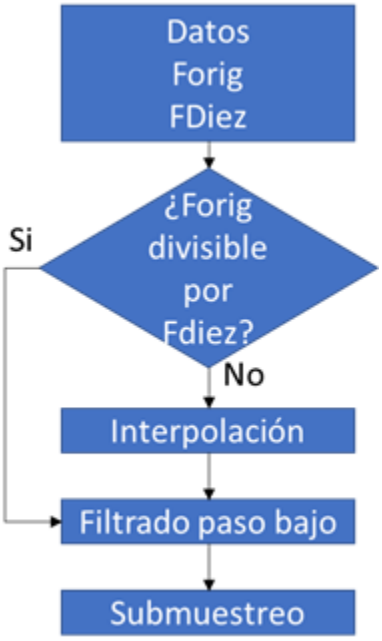


Ilustración 27:Proceso de diezmado (Imagen Propia)

El diezmado se realizó con un filtro FIR diseñado por medio de la herramienta online TFilter (Isza, 2011) como se muestra en la Ilustración 28, con una banda de paso desde 0-3800Hz y un rizado de 0.01dB en dicha banda y tiene una banda de restricción desde 4000Hz a -102.42 dB y después se usó el método *decimate* de la librería de python™ (Python, 2001) "SciPy" (Virtanen, y otros, 2020), con el fin de llevar la frecuencia de muestreo de 16KHz a 8KHz.

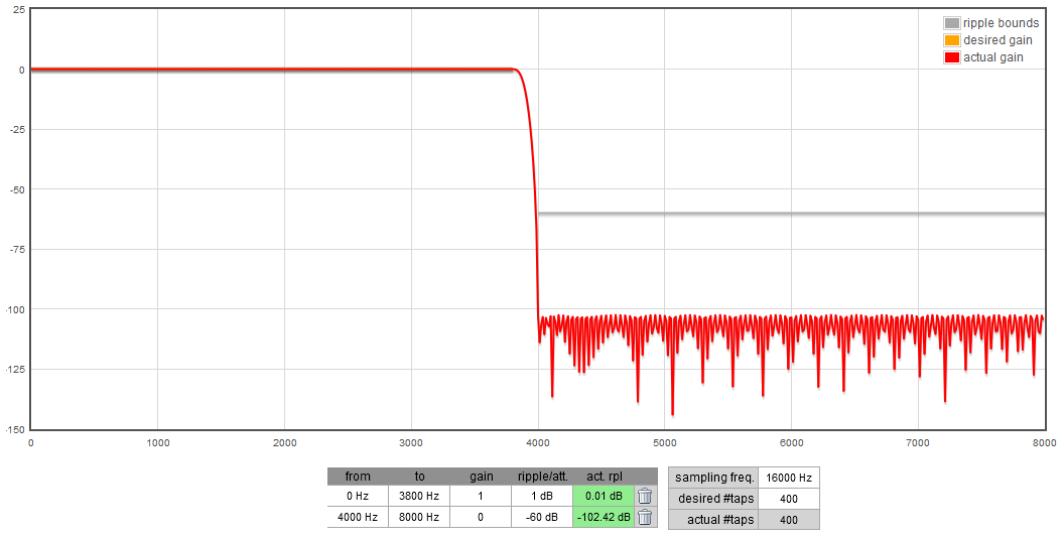


Ilustración 28: Filtro FIR anterior a Diezmado (Isza, 2011)

Para la energía en el tiempo se hizo uso del valor RMS de 1024 muestras en el tiempo convertido a dB FullScale (dB FS) con la Ecuación 14.

$$dBFS = 20 * log(RMS_{n=1024})$$

Ecuación 14: RMS a dBFS (Audio Enginnering Society, 2015)

Con el fin de realizar el espectrograma se hizo uso de la librería Matplotlib (Hunter, Dale, Firing, & Droettboom, 2002), más concretamente el método *specgram* al cual se le especificó un número de muestras de 1024 en escala de magnitud (dB) para la extracción del espectrograma de las muestras, presentado en el Código 1. El espectrograma funciona tomando un número arbitrario de muestras de la señal y realizando la transformada de Fourier se llevan los datos al dominio de la frecuencia, luego se hace uso de una ventana Hanning (Harris, 1978) definida por defecto para suavizar los efectos de haber tomado un pequeño segmento de la información, como efecto secundario de hacer esto la información queda ligeramente alterada en un fenómeno llamado dispersión de espectro; al espectro resultante se le aplica un mapeo en intensidad *viridis* en color que es graficado. Los espectrogramas limitan la cantidad de intensidad que muestran, para el caso del ejemplo se limita a -150 dB en las líneas 3 y 9.

```
fig, ax = plt.subplots(figsize=(16,6))
cmap = plt.get_cmap('viridis')
vmin = 20*np.log10(np.max(x)) - 150
cmap.set_under(color='k', alpha=None)
NFFT = 1024
pxx,  freq, t, cax = ax.specgram(x/(NFFT/2), Fs=fs, NFFT=NFFT,
                                 mode='magnitude',
                                 noverlap=NFFT/2,
                                 vmin=vmin,
                                 cmap=cmap,
                                 )
fig.colorbar(cax).set_label('Intensity [dB]')
```

Código 1: Grafica del Espectrograma

2.2 Codificación de algoritmos

El algoritmo usado para la cancelación de ruido el Fx-LMS. Inicialmente se realizó la codificación en Python debido a su facilidad de testeo y una vez funcional se codifica en lenguaje C, el lenguaje que acepta el compilador del microcontrolador.

2.2.1 Codificación del filtro LMS

El filtro LMS es un filtro FIR con coeficientes variables que fue mostrado en la Ilustración 18 y con la actualización de pesos W(Z) como fue descrita en la Ecuación 10. El filtro FIR se compone de un buffer que almacena los valores anteriores capturados por el filtro, un apuntador al último valor ingresado y los coeficientes W que multiplican a las entradas. Esta configuración en particular se llama "filtro FIR de buffer recirculante", contrasta con otra implementación simple (Ilustración 29) en la cual el último valor ingresado al buffer corresponde al elemento cero del vector de buffer, por lo que se deben correr todos los valores del buffer al momento de ingresar un nuevo valor. En el caso del buffer recirculante de la Ilustración 30, el apuntador (flecha roja) indica cuál elemento debe ser multiplicado con el coeficiente cero del filtro $W(Z)$ y se hacen las multiplicaciones con los coeficientes en el sentido inverso en el cual se estaban agregando las entradas al buffer, para el ejemplo, se están agregando elementos en sentido numérico ascendente y se están realizando las multiplicaciones en sentido numérico descendente.

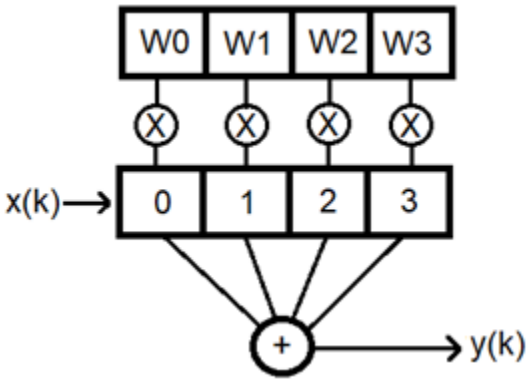


Ilustración 29: Implementación simple del filtro FIR (Imagen Propia)

El orden de ejecución del algoritmo se describe a continuación con base en (Hansen, Snyder, Qiu, Brooks, & Moreau, Feedforward Control System Design, 2012):

1. Avanzar la fila de valores en el buffer del filtro e ingresar el nuevo valor al buffer.
2. Calcular la salida del filtro
3. Calcular el error, la diferencia entre la salida actual del filtro y la salida deseada.
4. Calcular los nuevos coeficientes del filtro LMS
5. Repetir la secuencia.

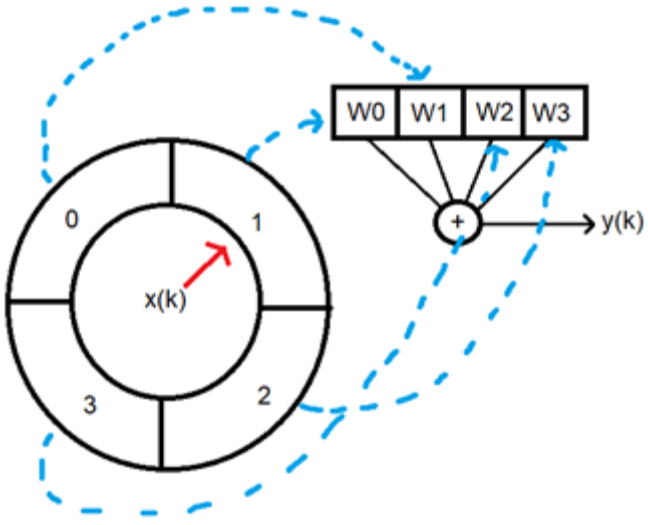


Ilustración 30: Implementación de Buffer Recirculante del filtro FIR (Imagen Propia)

El filtro LMS se puede ver como un filtro FIR dinámico que actualiza los pesos de sus coeficientes, de esta manera se ahorra la tarea de codificar ambos filtros por aparte, siendo el filtro LMS una extensión del filtro FIR. El filtro FIR, dado en el

Código 2 se compone de 4 parámetros, a saber, un apuntador que mantiene los coeficientes $W(Z)$ del filtro, otro apuntador que es el buffer de las entradas $X(k)$, la cantidad de los coeficientes tanto de $W(Z)$ como de $X(k)$ y el apuntador al último valor ingresado del filtro.

```
1.  struct FIR_FIL{
2.      float *taps;
3.      float *buf;
4.      int len;
5.      int off;
6.  };
```

Código 2: Estructura del filtro FIR

La estructura de filtro FIR es inicializada por medio de una función que se muestra en el Código 3, la función se sirve de un apuntador a la estructura FIR_FIL, la longitud del arreglo y un arreglo que almacena los pesos para llenar los campos estipulados por el Código 2, los valores que cambian a cada iteración, es decir *off* y *buf* son forzados a cero para evitar que valores espurios ingresen a la ejecución y prevengan su correcta ejecución.

```
1.  void FIR_FIL_init(struct FIR_FIL *myself, int lenlen, float *array){
2.      int i;
3.      myself->len = lenlen;
4.      myself->taps = array;
5.      myself->buf = malloc(sizeof(float)*lenlen);
6.      myself->off = 0;
7.      for (i = 0; i<lenlen;++i){
8.          myself->buf[i] = 0;
9.      }
10. }
```

Código 3: Inicialización de la estructura de filtro FIR

Los pasos 1 y 2 de la forma de operación para el filtro FIR son realizados con una función llamada FIR mostrada en el Código 4, ella recibe un valor de entrada correspondiente al nuevo valor de $\mathrm{X(k)}$ que entra a reemplazar su último valor tal y como lo muestra la Ilustración 29. A continuación, se define a un acumulador llamado *out_C* que es la salida filtrada. Dos iteradores son posicionados, *i* para iterar en los coeficientes tanto de $\mathrm{W(Z)}$ como de $\mathrm{X(k)}$ e *ind* que es un iterador derivador de *i* que determina el índice en el sentido contrario del buffer recirculante representado por la Ilustración 30 que, para este caso, llena el buffer en sentido numérico descendente y realiza el filtrado en sentido ascendente. Las líneas 4 y 5 realizan el paso 1 de la forma de operación previamente definida, primero se calcula el índice del nuevo dato mediante el operador módulo, este

garantiza que los índices van a estar limitados entre la longitud del arreglo y cero, como se está restando se puede dar la situación de que el módulo opere sobre un número negativo, cuyo resultado desbordaría al arreglo en la región de memoria a la izquierda de su definición en el programa, para subsanar esto se usa la propiedad de que la adición al dividendo del divisor no altera el módulo para números positivos, hecho esto se ingresa el nuevo dato al arreglo. Después se prosigue a calcular la salida del filtro con los interadores antes mencionados en el ciclo *For* de las líneas 6-9, se inicia calculando el índice del buffer de una forma similar a la línea 4 que se usa para multiplicar los coeficientes del filtro con el buffer siguiendo la definición estándar del filtro FIR, la función finaliza devolviendo el acumulador producto del proceso de filtrado.

```
1.  float FIR (float inin,struct FIR_FIL *myself){
2.    float out_C = 0;
3.    int i,ind;
4.    myself->off = (myself->len + myself->off -1)%myself->len;
5.    myself->buf[myself->off]=inin;
6.    for (i=0;i<myself->len;++i){
7.      ind = (myself->off+i)%myself->len;
8.      out_C = out_C + myself->taps[i]*myself->buf[ind];
9.    }
10.   return out_C;
11. }
```

Código 4: Función de avance y cálculo de salida

Existe un buffer camino secundario que solamente guarda los valores ingresados del filtrado $f(k)$ de la Ilustración 18 y que se usan para la actualización del filtro LMS, su implementación aparece en el Código 5, funciona en forma análoga al presentado en las líneas 4 y 5 del Código 4. La función se diferencia de la función *FIR* en cuanto al hecho de que no aplica un filtrado, por lo que se omite la definición de la estructura *FIR_FIL* y la función recibe los valores individuales de la entrada, el buffer recirculante, su longitud y su apuntador.

```
1.  void buff_C_S (float inin,float *x_buff_t,int len_x_buff_t,int *off_buff_t){
2.      *off_buff_t = (len_x_buff_t+*off_buff_t-1)%len_x_buff_t;
3.      x_buff_t[*off_buff_t]=inin;
4.  }
```

Código 5: Buffer que representa a C(z)

Como ya se dijo, el filtro LMS está implementado por medio del filtro FIR con una función adicional, dicha función en el programa se llama *FIR_update* y se describe a continuación en el Código 6. Dicha función tiene como entradas el error, el buffer de la señal filtrada, los coeficientes del filtro FIR y la cantidad de coeficientes del filtro. Para la actualización de los coeficientes sigue las mismas

reglas de implementación del buffer recirculante ya expresadas mediante el doble iterador *i* e *ind* y es una calca en código C del modo de actualización expresado por la Ecuación 10. Siendo BETA y MU la tasa de aprendizaje y olvido respectivamente.

```
1.  void FIR_update(float errorV,float *f_,int off,float *filter_taps_W_t, int len_fil
    ter_taps_W_t){
2.    int i,ind;
3.    for (i=0;i<len_filter_taps_W_t;++i){
4.      ind = (off+i)%len_filter_taps_W_t;
5.      filter_taps_W_t[i] = BETA*filter_taps_W_t[i] - 2*MU*errorV*f_[ind];
6.    }
7.  }
```

Código 6: Función de filtro LMS

La conjunción de todas las funciones del filtro FIR antes mencionadas llevan al código final ilustrado en el Código 7 que da cuenta del sistema propuesto en Fx-LMS, en él, se inicia en la línea 2 capturando la muestra de entrada x(k), prosigue con el cálculo del camino primario $P(Z)$, LMS $W(Z)$, Secundario $C(Z)$y Secundario Aproximado $\hat{C}(Z)$ en las líneas 2-6 tal y como la forma de operación del filtrado LMS lo estipula para los pasos 1 y 2, el paso 3 se realiza en la línea 7, que no es más que el cálculo del error y en la línea 9 consuma el paso 4, que es la actualización de los coeficientes, por lo que esta es la implementación en código C del filtro LMS.

```
1.  for (i=0;i<samples-1;++i){
2.      inin = sound_in[i];
3.      out_P = FIR(inin,&Primary);             // Camino Primario,Elemento Análogo
4.      out_W = FIR(inin,&WFilter);             // Filtro LMS
5.      out_S = FIR(out_W,&Secondary);          // Camino Secundario,Elemento Análogo
6.      out_SA= FIR(inin,&Secondary_Approx);    // Modelado del camino secundario
7.      E=out_S+out_P;                          //Error Simulado
8.      buff_C_S(out_SA,f_buff,len_buf,&off_F_buff); // Buffer de Secondary_Approx
9.      FIR_update(E,f_buff,off_F_buff,WFilter.taps, WFilter.len);//Actualización LMS
10.     EE[i] = E; //Guardado del error, propósitos documentales
11. }
```

Código 7: Ciclo principal del filtro FIR en simulación

Para la simulación del filtro FIR se tomó un camino primario de 15 coeficientes, un filtro LMS de 50 coeficientes y los caminos secundarios de 7 coeficientes.

2.2.2 Representación numérica de los datos

Cuando se hizo la implementación en el microcontrolador, se debió afrontar con la limitación de la capacidad de procesamiento del microcontrolador porque a

pesar de que el ESP32 tiene un puntaje de 1MFlop en Linpack[13] (Ivković & Ivković, 2017), esto puede no ser suficiente. Se anota que el desempeño del microcontrolador para operaciones int32 en Dhraystone[14] 2.1 es de 113 MIPS, dos órdenes de magnitud más grande que para punto flotante de 32 bits. Se decidió aprovechar dicha capacidad de cómputo y se adaptó el algoritmo LMS a un formato de punto fijo.

Para tal fin se debe implementar la saturación para la suma y la saturación y el truncamiento para la multiplicación. Existe una librería para C (Voras, 2020) que implementa el punto fijo sin saturación de valores, no se juzga necesario realizar dicha saturación puesto que en el filtro FIR regular tanto los coeficientes como las entradas están limitadas a un rango de valores de 16 bits y se tienen 32 bits para realizar la representación. Adicional a esto resulta que son de mayor importancia las cifras más significativas por lo que se realiza la normalización del valor de entrada entre -1 y 0.9999695 teniendo en cuenta que el rango de valores discretos de 16 bits es *(-32768,32767)*. El formato de punto fijo escogido es [2,30] en complemento a dos, un bit de signo, un bit entero y 30 bits para la coma binaria.

Con el fin de evaluar si la cuantización [2,30] es suficiente se hizo uso de la herramienta del histograma con los coeficientes de $W(Z)$ que se desarrollan a lo largo de la ejecución del filtro LMS para con todos los datos, hallándose dicha cuantización satisfactoria para el propósito de representar las muestras de audio.

Con el fin de incluir la librería de punto fijo en el algoritmo Fx-LMS es necesario realizar modificaciones en el código, modificaciones como las que están consignadas en el Código 8. En la primera sección se define la cantidad de bits que puede ser o 32 o 64 y se define la cantidad de bits que representan datos enteros que para este caso son 2 y enseguida se incluye la librería de punto fijo. La segunda sección acontece en el ciclo principal, necesita que los datos que se capturan en formato de 16 bits se conviertan al espacio normalizado y a punto fijo, se logra mediante una división de punto flotante y la función *fixedpt_rconst* de la librería, también se muestra en la línea 12 el uso de la suma. La sección 3 muestra las modificaciones en la definición de tipo de dato de punto fijo y en el

[13] (Dongarra, 1988) benchmark de punto flotante que resuelve un sistema de ecuaciones

[14] (Keil, 2020) es un benchmark que no hace uso de operaciones de punto flotante

uso de la multiplicación. Conformándose la implementación de punto fijo, se evaluó el mejoramiento del desempeño y se encontró satisfactorio.

```
{...1}

#define FIXEDPT_BITS 32
#define FIXEDPT_WBITS 2
#include "fixedptc.h"

{...2}

for (i=0;i<samples-1;++i){
    inin = fixedpt_rconst(sound_in[i]/32768.0);//Entrada de datos
    {...}
    E=fixedpt_add(out_S,out_P);
    {...}
    EE[i] =(int) (32768.0*fixedpt_tofloat(E)); Salida de datos
}

{...3}

fixedpt FIR (fixedpt inin,struct FIR_FIL *myself){
  fixedpt out_C = 0;
  {...}
  for (i=0;i<myself->len;++i){
    ind = (myself->off+i)%myself->len;
    out_C = out_C + fixedpt_mul(myself->taps[i],myself->buf[ind]);
  }
  return out_C;
}
```

Código 8: Inclusión de la librería de punto fijo

2.2.3 Modelado de canal

El modelado del canal se usa para el camino de secundario $\hat{C}$ que realiza en el mismo microcontrolador que implementa el controlador, modelado realizado una sola vez antes de la ejecución de la cancelación de ruido. El modelado puede ser descrito por medio del pseudocódigo descrito en el Código 9; Durante un tiempo determinado el microcontrolador va a generar y reproducir un número aleatorio entero de distribución normal con $\sigma^2 = 16384$ y va a saturar en caso de que el valor generado se desborde de los 16 bits de audio, el número generado se reproduce y se pasa por el filtro FIR antes mencionado, una vez hecho esto se lee el micrófono que da lugar al error conseguido por medio de la predicción que se usa para actualizar los valores de los coeficientes del filtro, una vez finalizado el proceso, los valores almacenados deberían ser el modelado del camino secundario.

```
for 100 segundos{
    num = generar_aleatorio()
    reproducir(num)
    filtro_FIR(num)
    result = microfono_leer()
    error = num-result
    actualiza_LMS_C_S(error)
}
```

Código 9: Pseudocódigo del modelado del canal

2.3 Programación del microcontrolador

Para programar la ESP32 se hace uso de un framework llamado ESP-IDF que tiene todas las definiciones, librerías de periféricos, compiladores y debuggers concernientes al ESP32. Con el fin de evitar la realización de tareas repetitivas se hace uso de una integración con Eclipse-IDE, un entorno de desarrollo, se usó la versión compatible más reciente conocida (2020-06) y se usó una metodología de granularidad para programar los componentes del proyecto, programando y testeando cada periférico individualmente para luego integrarlo todo en un mismo programa.

2.3.1 Conexiones y funciones de componentes

El sistema por programar consiste en el mostrado en la Ilustración 31, en él se observa que inicialmente hay un computador con una base de datos de ruido digital que se usa para crear una onda de referencia de ruido que es inicialmente capturada por un micrófono que por comunicación I2S le entrega a un microcontrolador sus datos capturados, este a su vez transmite datos hacia otro microcontrolador que realiza el procesamiento del algoritmo de cancelación de ruido. Están separados dichos microcontroladores puesto que cada microcontrolador solamente tiene dos periféricos I2S que en el caso del microcontrolador de procesamiento ya están ocupados, adicional a esto, la distancia entre el primer micrófono y el segundo microcontrolador puede ser lo suficientemente elevada para generar interferencias con la señal I2S, no así con la señal UART, por eso la separación. El segundo microcontrolador G(z) se encarga tanto de informar al DAC (conversor de señal Digital a Análoga) de la señal de control por medio de un periférico I2S como de recibir la señal de error proveniente del micrófono del extremo derecho. Finalmente, entre el DAC y el parlante existe un amplificador de sonido que se encarga de tomar la señal del

DAC y llevarla a unos niveles de energía necesarios para mover un parlante que idealmente cancela el ruido en el punto de suma.

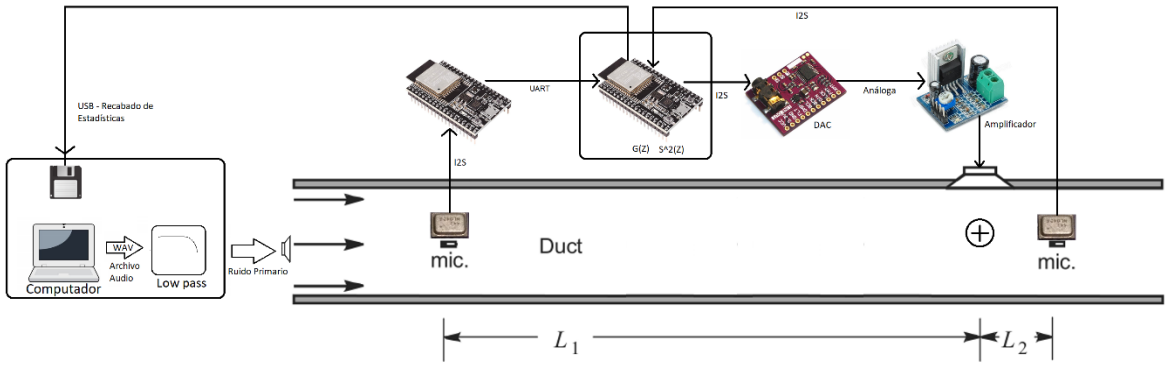


Ilustración 31: Arquitectura física del sistema de cancelación de ruido (Imagen Propia)

El ESP-IDF del microcontrolador soporta directamente el uso de un sistema operativo de tiempo real para microcontroladores llamado FreeRTOS (Svec, 2012). Por lo que el programa se divide en tareas que o bien administran las entradas/salidas de un periférico en específico o bien procesan las entradas/salidas.

2.3.2 Esquema de tareas del microcontrolador

2.3.2.1 Microcontrolador ESP32 1

El microcontrolador 1 tiene la sencilla misión, como esta esquematizada en la Ilustración 32 de recibir los datos de un micrófono, correspondientes al ruido que se desea cancelar, filtrarlos en el rango de las frecuencias de interés, pasar los datos a un formato de 16 bits y comunicarlos al microcontrolador 2. Por esto es llamado el microcontrolador del sensor de micrófono.

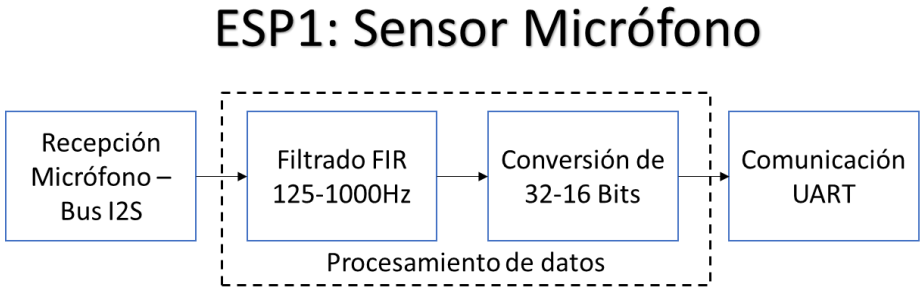


Ilustración 32: Esquema de tareas del microcontrolador ESP32 1 (Imagen Propia)

2.3.2.2 *Microcontrolador ESP32 2*

El microcontrolador 2 se encarga de realizar el proceso de control, para él se definen dos modos de trabajo que coexisten dentro del mismo espacio de memoria, está el modo de modelado de canal definido en la Ilustración 34 y el modo de cancelación definido en la Ilustración 33. El modo de cancelación se compone de recibir los datos del micrófono por el bus I2S y a su vez realizar concurrentemente la recepción del otro micrófono por medio de la recepción UART. Ya con estos datos se realiza la ley de control y se envía al DAC la solución calculada, mientras que se comunican los datos tanto de los micrófonos como de la corrección calculada a un computador mediante USB.

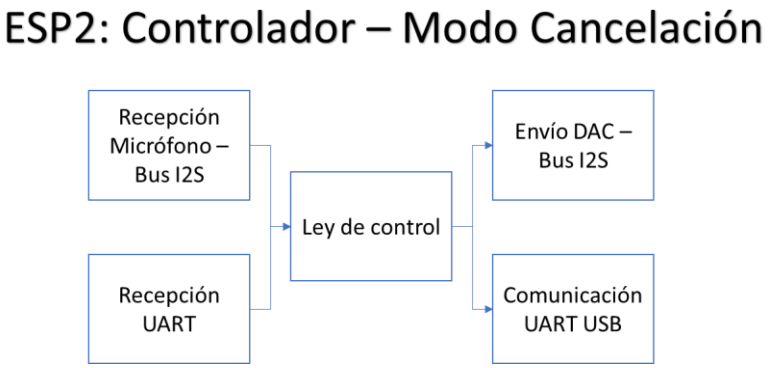


Ilustración 33: Esquema de tareas del microcontrolador ESP32 2 en modo de cancelación (Imagen Propia)

Para el caso de que el microcontrolador este modelando el sistema reciben los datos del micrófono conectado directamente al ESP32-2, se ejecuta el programa de modelado previamente definido y se le comunica la muestra aleatoria calculada al DAC, a su vez de que informa del error actual del filtro y del modelo final por medio de la UART.

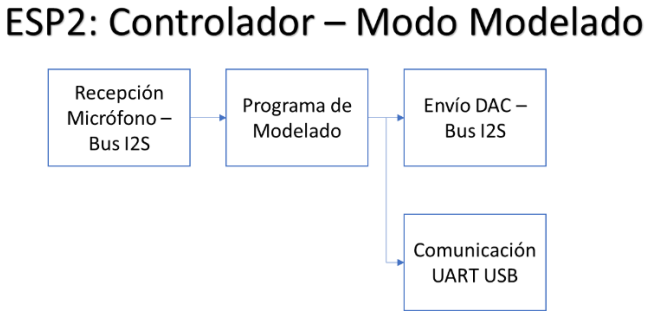


Ilustración 34: Esquema de tareas del microcontrolador ESP32 2 en modo de modelado (Imagen Propia)

2.3.3 Configuración de periféricos

2.3.3.1 UART

La configuración de la UART en cada microcontrolador es detallada en el Código 10. En él se especifica una tasa de 921600 baudios, la máxima tasa que soporta nativamente el driver UART de Windows 10 y por tanto la máxima velocidad que se podía probar sin necesidad de drivers personalizados, esta es 4 veces superior a la tasa de datos generada por el micrófono que es de 256kbps, el tipo de comunicación UART usada es con datos de 8 bits, un bit de stop, sin control de flujo de hardware (pines RTS y CTS) y con un reloj de periférico estipulado por APB que es de 80 MHz. En la instalación se definió un buffer de 512 bytes y una cola de 10 elementos de profundidad para interrupciones.

```
1.  #define BUF_SIZE (512)
2.  {...}
3.
4.  uart_config_t uart_config = {
5.      .baud_rate = 921600,
6.      .data_bits = UART_DATA_8_BITS,
7.      .parity    = UART_PARITY_DISABLE,
8.      .stop_bits = UART_STOP_BITS_1,
9.      .flow_ctrl = UART_HW_FLOWCTRL_DISABLE,
10.     .source_clk = UART_SCLK_APB,
11. };
12. uart_driver_install(UART_NUM_2, BUF_SIZE , BUF_SIZE, 10, &uart2_quenue, 0);
13. uart_param_config(UART_NUM_2, &uart_config);
14. uart_set_pin(UART_NUM_2, GPIO_NUM_17, GPIO_NUM_16, UART_PIN_NO_CHANGE, UART_PIN_NO
    _CHANGE);
```

Código 10: Código de configuración de la UART

2.3.3.2 I2S

La configuración de los periféricos I2S no difieren mucho entre sí, el Código 11 muestra la configuración del micrófono, donde en la línea 8 se define que el microcontrolador es maestro y por tanto genera los dos relojes necesarios para realizar la comunicación I2S y además es receptor, a continuación la tasa de muestreo es 16KHz que es la mínima soportada por los micrófonos, por funcionamiento del micrófono solamente se usa el canal izquierdo y las muestras PCM son de 16 bits, la variante de I2S usada es la estándar donde los datos PCM se envían en complemento a dos, empezando con el bit más significativo. La prioridad de interrupción definida es la más baja mientras que se define un banco de 4 búferes de 1024 bytes cada uno con el fin de almacenar las muestras. En la línea 16 sucede que, puesto que la frecuencia del reloj es calculada de acuerdo con la tasa y profundidad de bit, no se define una velocidad reloj para la transmisión, sino que se deja que el IDE lo calcule, por eso este parámetro es falso. La línea 17 implica que en caso de underflow de buffer, este se va a reiniciar y

puesto que no se está dando un valor específico al reloj, la línea 18 no interesa. Los pines para las señales I2S quedan definidos como lo estipula la estructura de la línea 20. Mientras que la instalación del driver hace uso del periférico I2S0 del microcontrolador y define una cola de 4 valores para su ejecución. Para los micrófonos I2S SPH0645 existe una incompatibilidad en los tiempos de ambos periféricos que está bien documentada en Hackaday (Teachman, 2019), con el fin de lograr el correcto funcionamiento se realiza las configuraciones de las líneas 29 y 30 que realizan retraso en las muestras leídas de 2 ciclos del reloj de la comunicación.

```
1.  #define I2S_MIC         (0)
2.
3.  {...}
4.
5.  // En main
6.  // i2s config
7.  i2s_config_t i2sMemsConfigBothChannels = {
8.      .mode = (i2s_mode_t)(I2S_MODE_MASTER | I2S_MODE_RX),
9.      .sample_rate = 16000,
10.     .bits_per_sample = I2S_BITS_PER_SAMPLE_32BIT,
11.     .channel_format = I2S_CHANNEL_FMT_ONLY_LEFT,
12.     .communication_format = i2s_comm_format_t(I2S_COMM_FORMAT_I2S),
13.     .intr_alloc_flags = ESP_INTR_FLAG_LEVEL1,
14.     .dma_buf_count = 4,
15.     .dma_buf_len = 1024,
16.     .use_apll = false,
17.     .tx_desc_auto_clear = true,
18.     .fixed_mclk = 0};
19. // i2s pins
20. i2s_pin_config_t i2sPins = {
21.     .bck_io_num = GPIO_NUM_26,
22.     .ws_io_num = GPIO_NUM_14,
23.     .data_out_num = I2S_PIN_NO_CHANGE,
24.     .data_in_num = GPIO_NUM_27};
25.
26.
27. i2s_driver_install(I2S_MIC, &i2sConfig, 4, &i2s0Queue);
28. // Solamente con SPH0645
29. REG_SET_BIT(I2S_TIMING_REG(I2S_MIC), BIT(9));
30. REG_SET_BIT(I2S_CONF_REG(I2S_MIC), I2S_RX_MSB_SHIFT);
```

Código 11: Código de configuración del micrófono

2.4 Esquema de cableado de componentes

El esquema de cableado propuesto depende de los dispositivos a los cuales se conecte, para el caso del microcontrolador 1, mostrado en la Ilustración 35, se tiene que solamente recibe el micrófono y lo comunica a la otra ESP32, el microcontrolador recibe su alimentación de una batería de litio de doble celda en serie con un voltaje nominal de 7.4V, lo cual no es un problema, puesto que

el regulador interno de la tarjeta, el AMS1117 regula hasta 15V. Se usan los pines necesarios para alimentar y comunicarse con el micrófono mediante el protocolo I2S que opera a 3.3V y que se comunica con los pines IO26, IO27 e IO14. La comunicación serial se realiza con los pines IO17, IO16 y un nivel de tierra común a los dos microcontroladores ESP32.

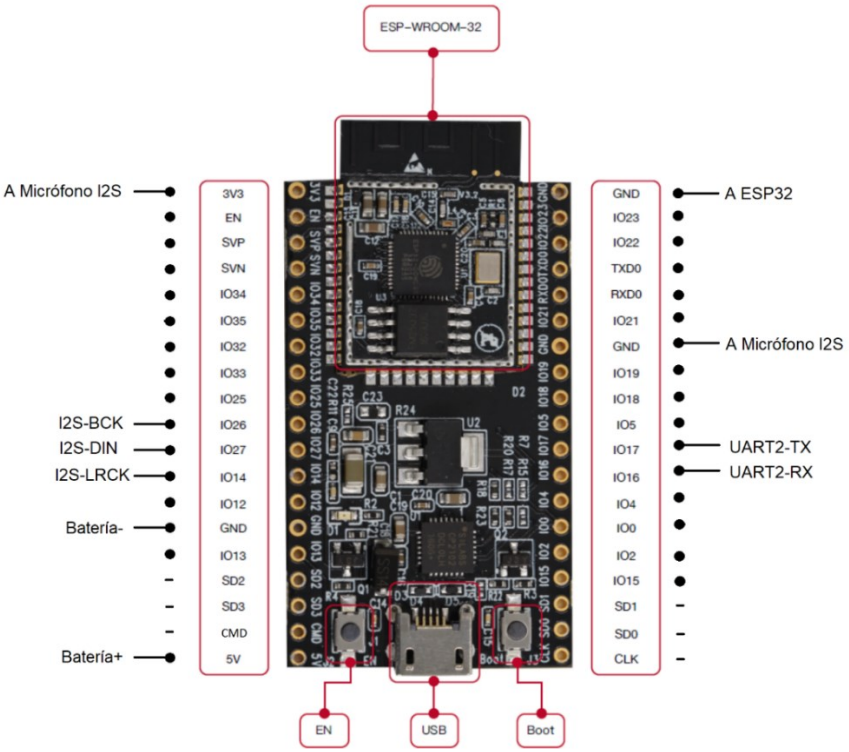


Ilustración 35: Esquema de conexión ESP 1 (Go Jimmy Pi, 2018)

El microcontrolador 2 conecta como lo indica la Ilustración 36 a cuatro dispositivos en la realización de sus funciones, un conversor digital análogo por protocolo I2S, un micrófono por protocolo I2S, una comunicación UART y una comunicación por USB a un computador para fines de documentación que adicional a esto le provee alimentación. La conexión es idéntica a la ESP1 salvo por la batería que no se encuentra en este caso, por las conexiones del DAC I2S que están definidas en los pines IO32, IO33 e IO25 y por la conexión USB al computador.

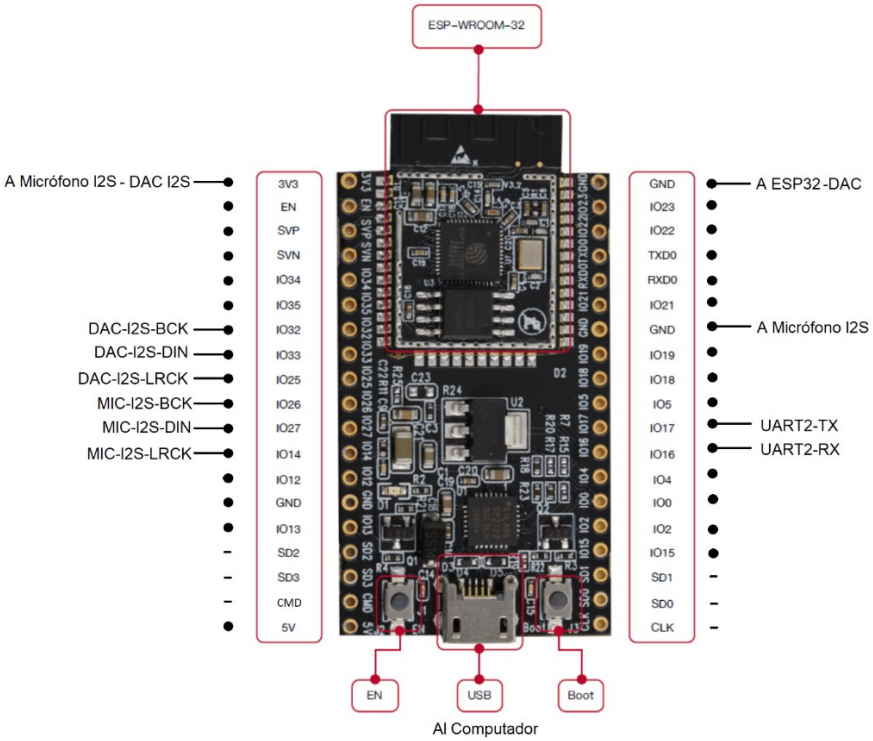


Ilustración 36: Esquema de conexión ESP 2 (Go Jimmy Pi, 2018)

Para los micrófonos se realiza la conexión como dice la Ilustración 37, la variable SEL en tierra, es decir, que el micrófono interpretara que es el canal derecho, las señales LRCL, DOUT y BCLK se conectan a sus contrapartes del microcontrolador en tándem de la siguiente manera: I2S-LRCK, I2S-DIN e I2S-BCK. La tierra y el voltaje de alimentación de 3.3 voltios salen de los pines de la tarjeta de desarrollo del microcontrolador.

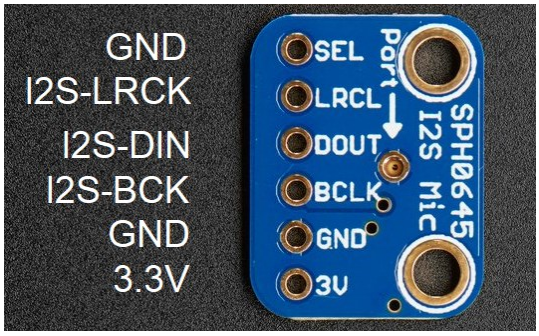


Ilustración 37: Conexión del micrófono SPH0645LM4H (Adafruit, 2017)

Para el DAC se conecta a SCK (El reloj del DAC) a tierra con el fin de que este genere su propio reloj, esto se logra o bien haciendo la conexión a tierra con cable o bien soldando el puente señalado con verde en la Ilustración 38. Las señales BCK, DIN y LCK se conectan a las señales del microcontrolador DAC-I2S-BCK, DAC-I2S-OUT Y DAC-I2S-LRCK. Mientras que la alimentación del dispositivo es traída del microcontrolador. Por otro lado, la salida LINE-OUT de la tarjeta que lleva la señal de audio se conecta con un cable de audio al amplificado, de esta solo es de interés el canal derecho.

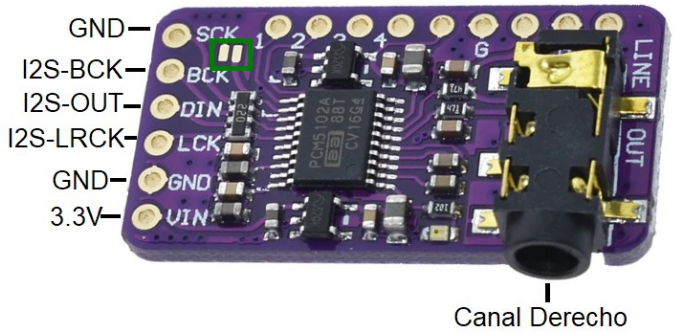


Ilustración 38: Conexión del DAC PCM5102A (Imagen Propia)

El amplificador de audio, como se encuentra en la Ilustración 39 es alimentado por medio de un voltaje de baterías con el fin de garantizar que sea lo más estable posible mientras que las entradas IN+ e IN- se conectan al canal derecho de audio de la salida del DAC, por otro lado, la salida amplificada es conectada al parlante.

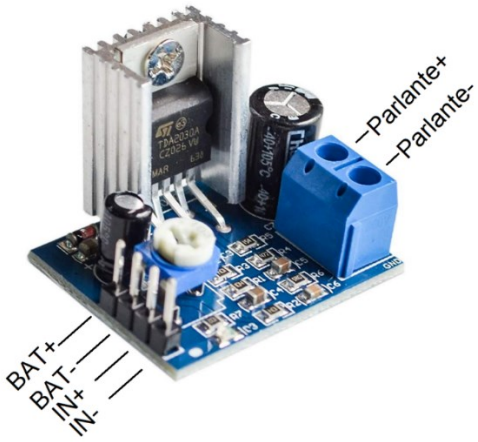


Ilustración 39:Conexión del Amplificador TDA2030A (Circuit.pk, 2020)

Una vez realizada la conexión se probó un primer modo en el cual se hace conexión directa de los dos micrófonos a un mismo microcontrolador, puesto que

un periférico I2S recibe por medio de dos canales, por lo que no era descabellado hacer uso del canal derecho para uno y del izquierdo para el otro. El micrófono que mide el ruido que se desea cancelar se comunicó al microcontrolador mediante cable CAT6 FTP aterrizado de 1.5 metros, con magros resultados. Luego se probó el arreglo final que es con la interfaz UART cuya comunicación fue exitosa.

3 Resultados

Del capítulo de metodología se ofrecen los resultados obtenidos durante la consecución de las acciones mencionadas en la metodología.

3.1 Sistema Físico

Se realizaron averiguaciones sobre la construcción de un sistema acústico donde se pudiera realizar control de ruido en la situación de ducto de ventilación. Con el fin de generar un confinamiento a los parlantes se construyeron dos cajas con planchas de madera aglomerada de 15 mm de espesor. Las dimensiones externas de las cajas tienen unas dimensiones externas de 12cmx25cmx11cm y su función es confinar el sonido dentro de sí y sólo permitir la propagación sonora hacia el tubo, esto se logra mediante las tablas que ofrecen una mayor resistencia al paso del sonido que el aire y mediante un proceso de sellado interno con silicona para vidrios y un sellado externo con masilla para madera. Los resultados pueden ser apreciados en la Ilustración 40 y en la Ilustración 41.

Ilustración 40: Caja Contención del Parlante (Imagen Propia)

Ilustración 41: Caja de madera, vista de perfil (Imagen Propia)

La tubería es conformada por tubería de PVC de 2mm de espesor y 67mm de diámetro interno. La Ilustración 42 muestra el arreglo completo, el tubo más largo es el sistema acústico por controlar con una longitud de 1390mm y el tubo más pequeño de una longitud de 180mm es el camino secundario por el cual viaja la señal de control, tomando a la velocidad del sonido como 340m/s se tiene que la latencia del sistema acústico es de 4ms, que se reduce a 3.56ms contando al camino secundario. Si la tasa de muestreo del sistema de control es de 16KHz se tiene que el camino principal tiene una longitud de 65.4 muestras mientras que el camino secundario es de 8.5 muestras, por lo que los filtros FIR que dan cuenta de dichos sistemas deben al menos tener dicha longitud de muestras para realizar el modelado y la adaptación LMS. Por otro lado, el diámetro del ducto de 67mm implica en la Ecuación 7 que la máxima frecuencia cancelable para un canal es de 2373Hz en este arreglo en específico.

Ilustración 42: Sistema Acústico (Imagen Propia)

3.2 Preprocesado

Para los resultados de Base de datos y preprocesado se tomaron dos medidas para cada audio presentado en la Tabla 1, la del espectro y la de la energía, el comportamiento de la energía se puede apreciar en la Ilustración 43 para STRAFFIC usando la medida de dB FS. Para estos análisis se están tomando los primeros 18 segundos de la grabación con la idea de limitar el tiempo de cómputo y para evitar que la información de las gráficas se amontone y se dificulte el análisis. El análisis de las grabaciones se realizó escuchando a la grabación original y etiquetando los eventos que suceden en la misma se muestra el comportamiento en frecuencia.

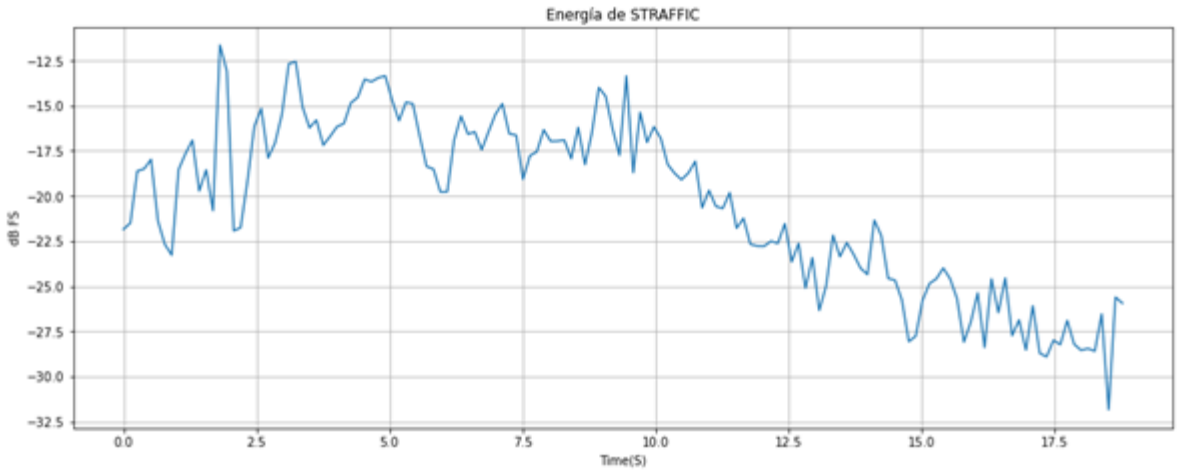


Ilustración 43: Energía de STRAFFIC en dB FS (Imagen Propia)

La situación de la cocina de la Ilustración 44 se muestra bastante quieta salvo en los momentos que alguien mueve un plato como movimientos tienden a ser impulsivos, se representan como una línea vertical en el espectro, como se puede ver en el segundo 17.5 y en torno al 14. Por otro lado, el abrir la alacena cerca del segundo 13 es un movimiento más suave y por ello aparece una frecuencia sostenida en torno a los 1100Hz.

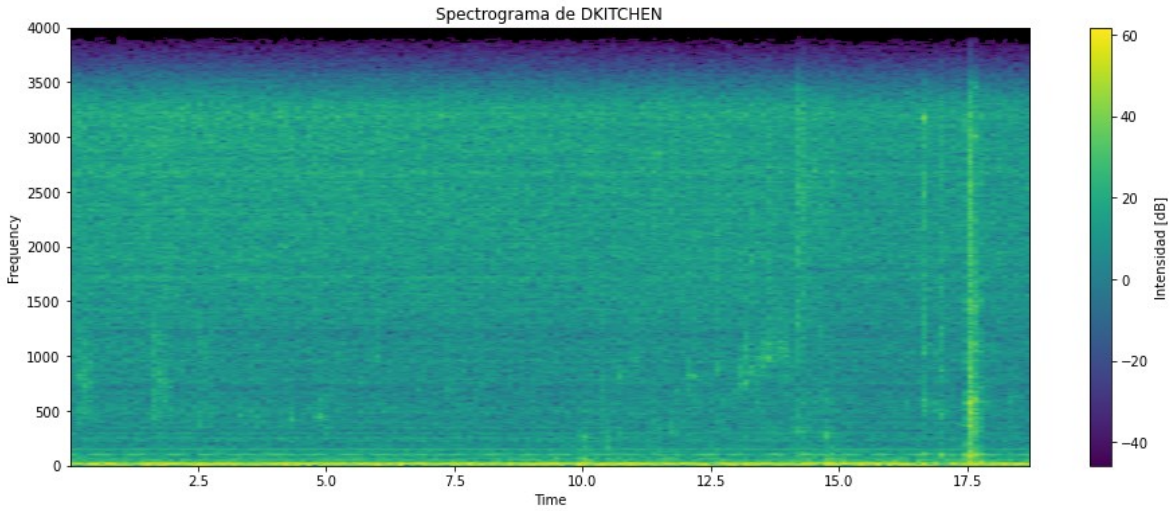


Ilustración 44: Espectrograma de DKITCHEN (Imagen Propia)

El espectrograma del río en la Ilustración 45 es mucho más constante dada la naturaleza del sonido de un arroyo dicho ruido se modela como ruido rosa en el cual la intensidad sigue el inverso de la frecuencia, es decir, entre más alta sea, proporcionalmente menos intensa será su energía. Existe ruido de baja frecuencia en la grabación producto del aire chocando con los micrófonos que no tienen protección alguna, por ello el ruido va tan bajo como 0Hz.

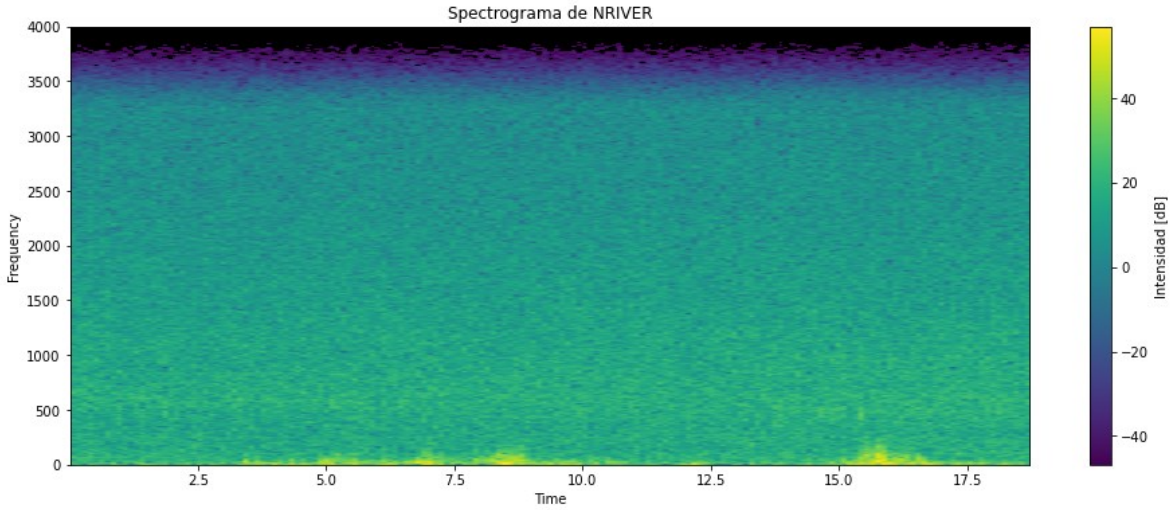


Ilustración 45: Espectrograma de NRIVER (Imagen Propia)

El ruido de fondo de cafetería mostrado en Ilustración 46 se representa en los grupos de barras verticales seguidos de una disminución producto de los silencios que suceden en el habla.

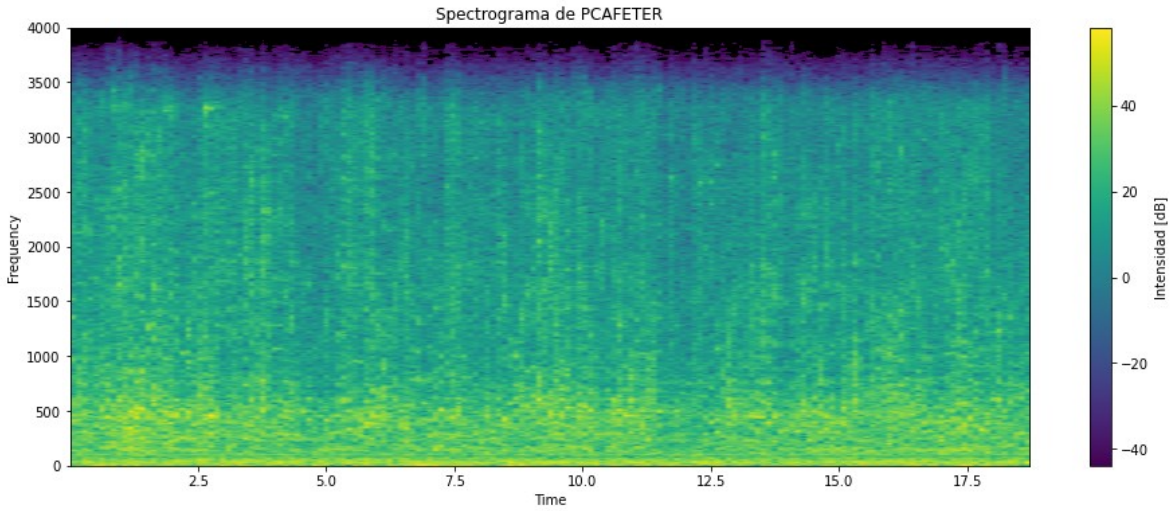


Ilustración 46: Espectrograma de PCAFETER (Imagen Propia)

La onda seno de la Ilustración 47 mantiene una intensidad constante en el tiempo, se resalta que por el uso de la ventana de Hanning existe una dispersión en la frecuencia que se expresa en el gradiente de intensidades, cuando en realidad solo existe la línea en torno a los 250Hz y el resto debería ser de color negro puesto que no hay contenido en dichas frecuencias.

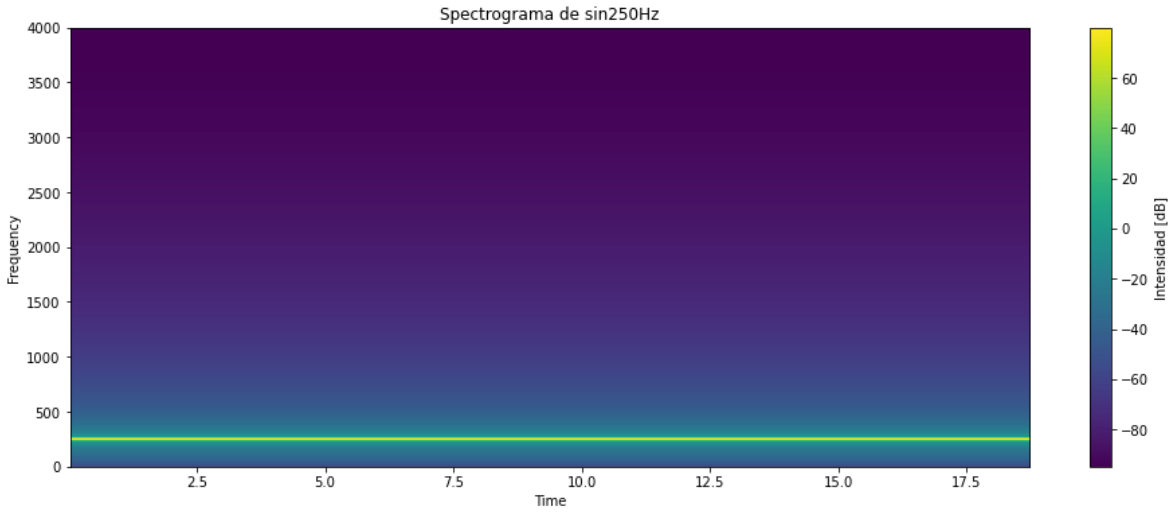


Ilustración 47: Espectrograma de sin250hz (Imagen Propia)

El evento mostrado por STRAFFIC en la Ilustración 48 consiste en la aceleración de un vehículo de carga pesada que está en su máximo en torno al segundo 5, este evento mantiene una subida y bajada gradual de intensidad.

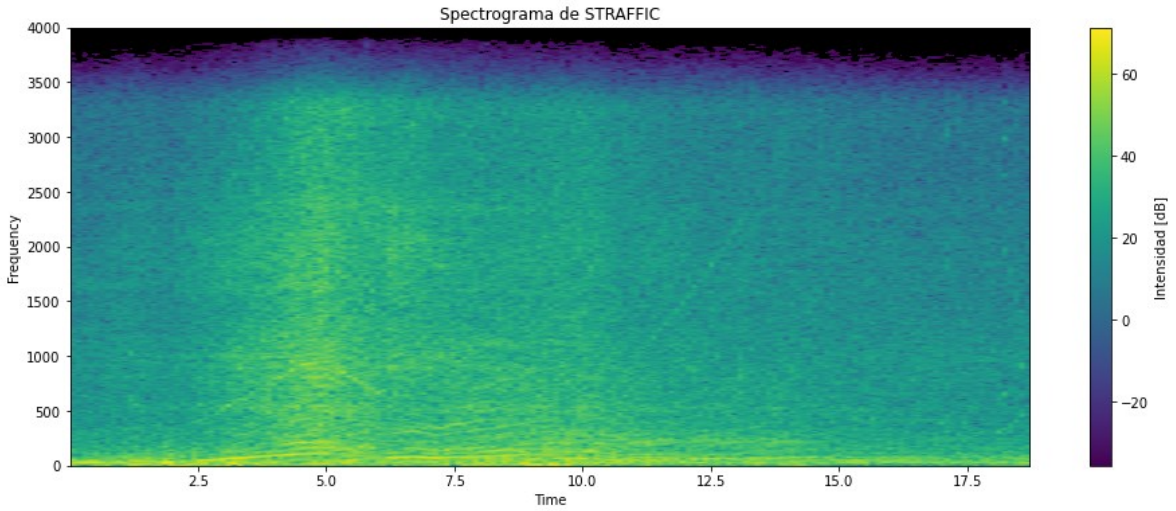


Ilustración 48: Espectrograma de STRAFFIC (Imagen Propia)

En la situación de TBUS en la Ilustración 49, el vehículo se encuentra parado y hay más influencia de las voces de las personas que del motor del bus.

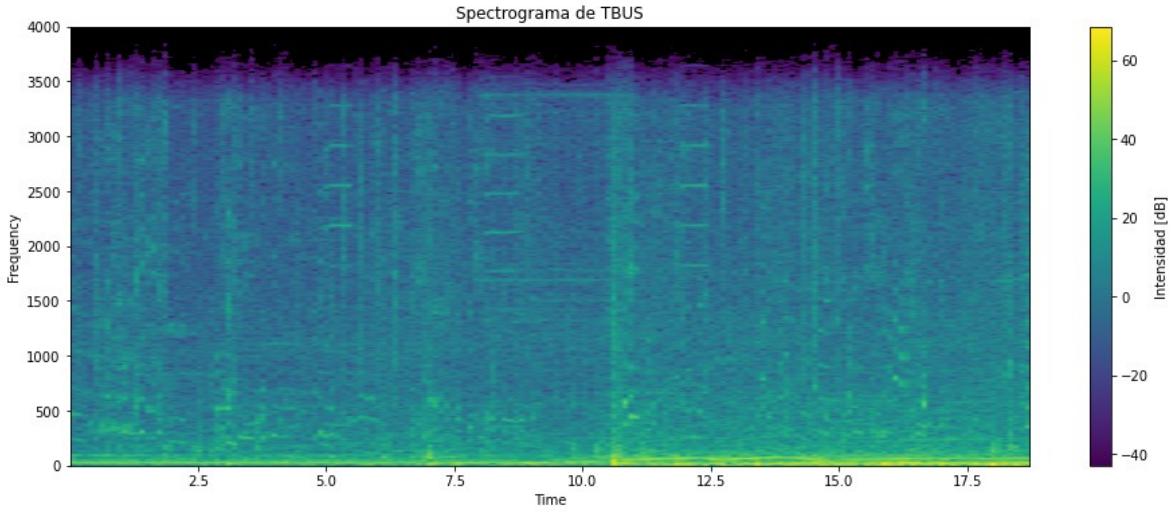


Ilustración 49: Espectrograma de TBUS (Imagen Propia)

La situación descrita en el metro en la Ilustración 50 es el proceso de apertura y cierre de puertas, al inicio de la grabación se abre, luego hay un silencio, que en el segundo 5 se interrumpe por una voz de niño y la respuesta de su madre en el segundo 8, en torno al segundo 13 aparece un tono de cierre de puertas para luego comenzar su cierre en el segundo 15.

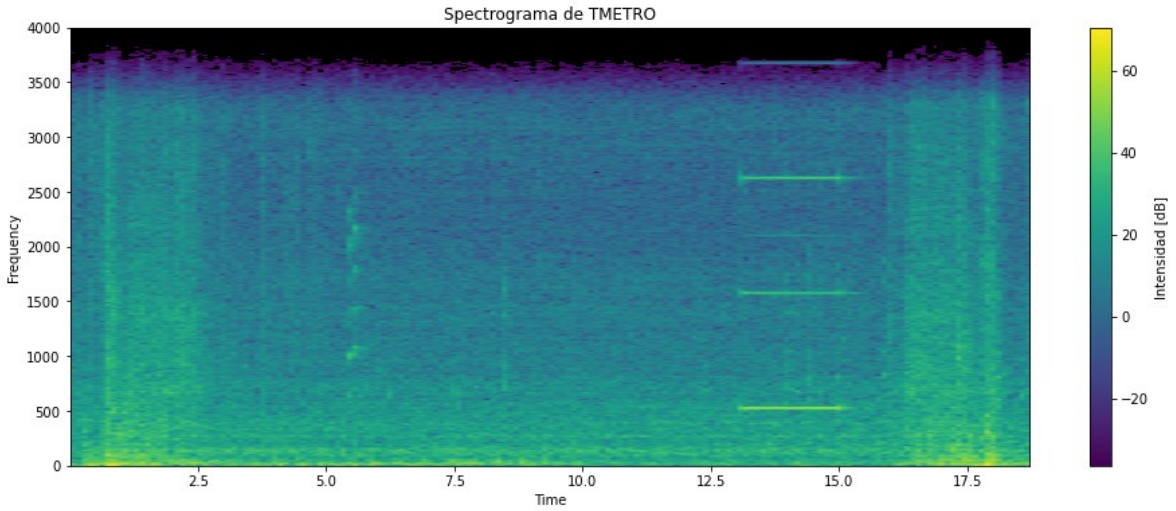


Ilustración 50: Espectrograma de TMETRO (Imagen Propia)

3.3 Filtrado Fx-LMS

Los resultados del filtrado Fx-LMS se detallan a continuación, se requiere que el mapeo de color del espectrograma sea el mismo con el fin de que sean comparables las gráficas de la sección de filtrado con la sección preprocesado, esto no sucede puesto que cuando el proceso de filtrado es exitoso se disminuye la amplitud cambiando el mapeo de color, con el fin equiparar los mapeos se añadió una onda seno de 3.9KHz con una amplitud igual a la amplitud máxima de las grabaciones de la sección de al final los datos, forzando un mapeo de intensidad similar.

La situación de la cocina en la Ilustración 51 y en la Ilustración 52 muestra en su espectrograma una sensible disminución de ruido de baja frecuencia respecto a su contraparte de métricas, sin embargo, la reducción en las otras frecuencias no se nota y se debe recurrir a la gráfica de intensidades con el fin de dar luces acerca de la reducción, que inicia en 0 y que a medida que pasa el tiempo se va aumentando, se nota que en el caso de los eventos impulsivos la reducción se disminuye al punto de que llega a mínimos en el segundo 17.5.

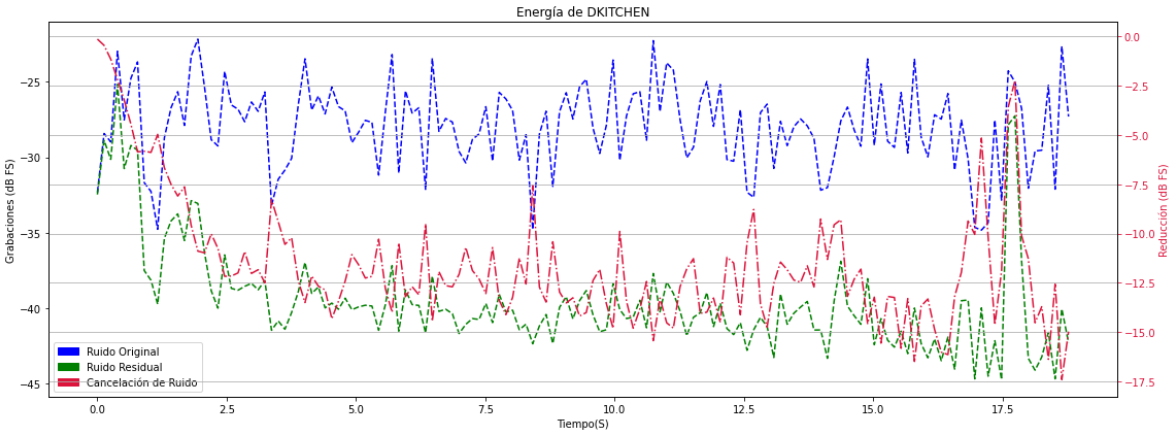


Ilustración 51: Reducción de intensidad en DKITCHEN (Imagen Propia)

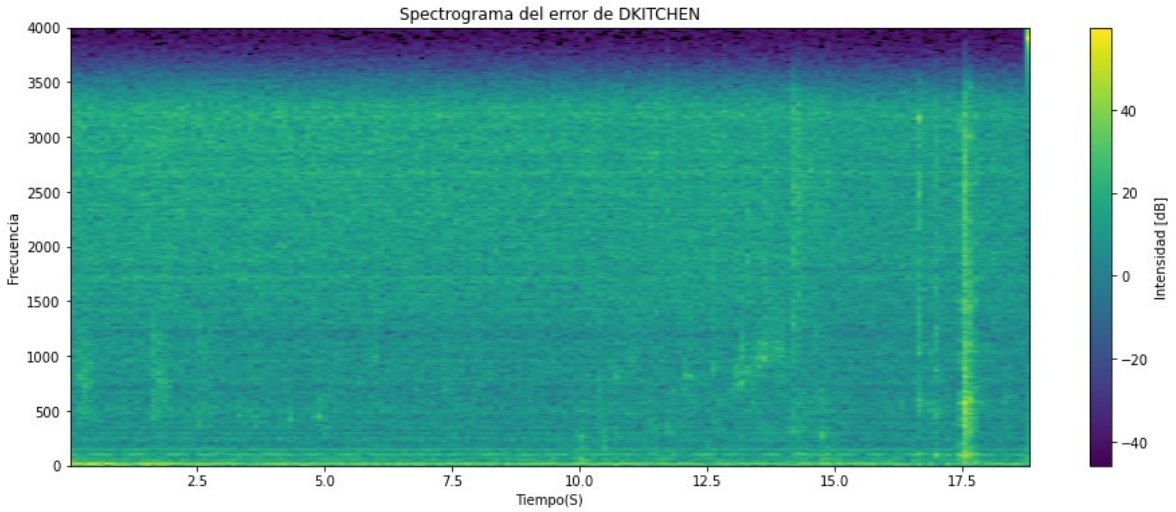


Ilustración 52: Espectrograma del ruido residual de DKITCHEN (Imagen Propia)

La Ilustración 53 y la Ilustración 54 corresponden al filtrado de ruido de NRIVER y muestran una muy pobre reducción de ruido que solamente mejora en los casos en los que aparece el ruido producto del viento chocando con el micrófono. El pobre desempeño podría deberse a que la naturaleza del sonido del arroyo como es ruido rosa, aleatorio, le deja muy complicado lograr alguna reducción al filtro.

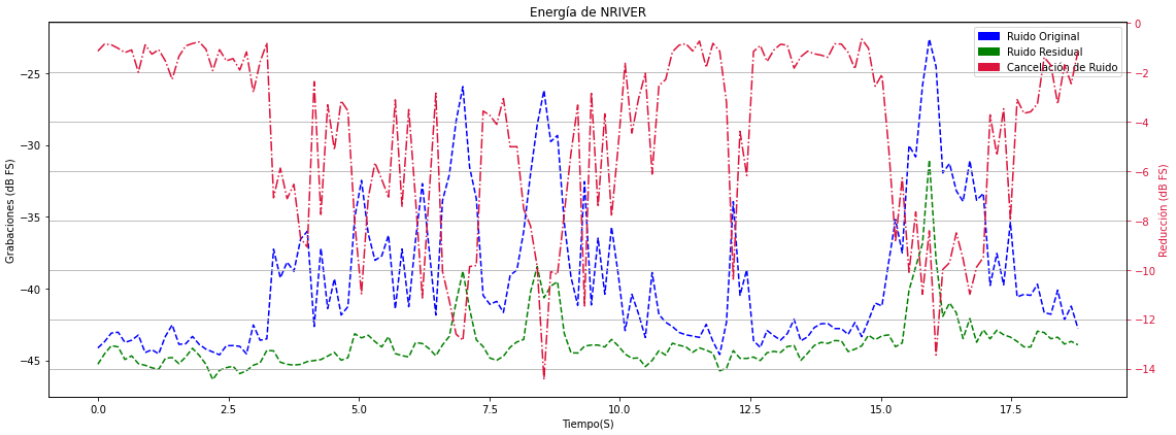


Ilustración 53: Reducción de intensidad en NRIVER (Imagen Propia)

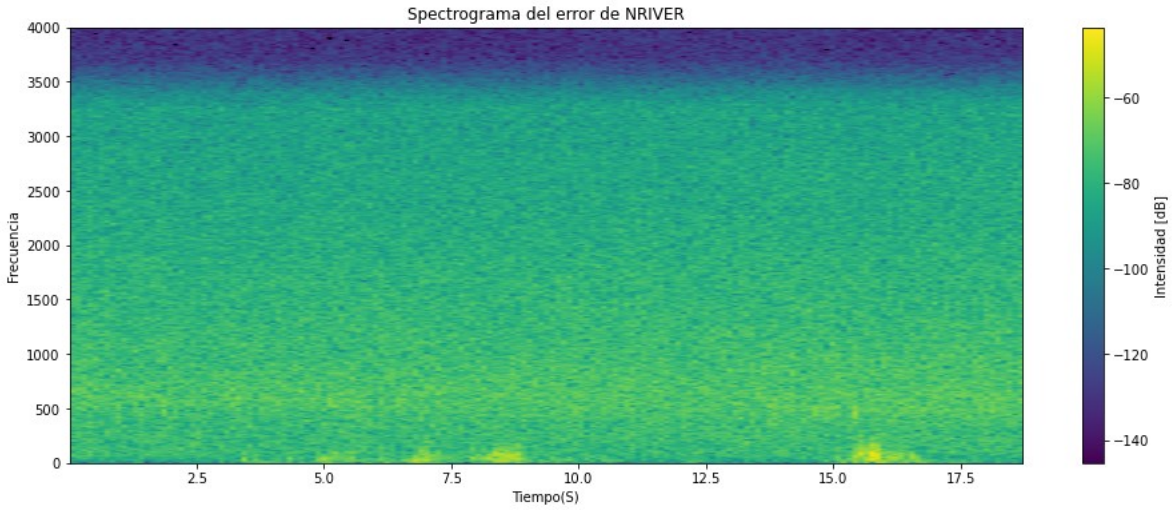


Ilustración 54: Espectrograma del ruido residual de NRIVER (Imagen Propia)

La cancelación de ruido de PCAFETER en la Ilustración 55 y la Ilustración 56 tiende a ser mayor en las situaciones en las que alguien habla que con el ruido de fondo. El espectrograma muestra una notable reducción en los sonidos de baja frecuencia.

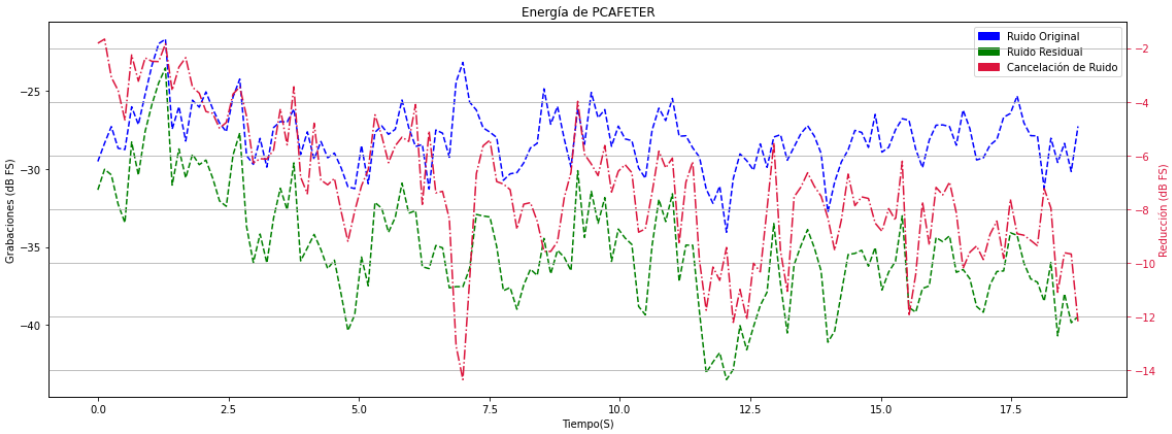


Ilustración 55: Reducción de intensidad en PCAFETER (Imagen Propia)

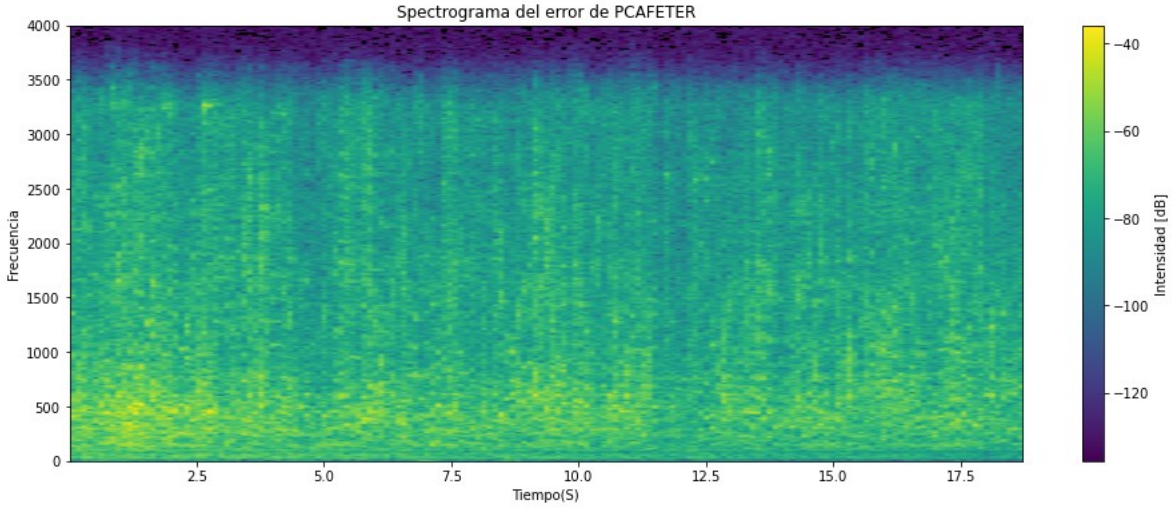


Ilustración 56: Espectrograma del ruido residual de PCAFETER (Imagen Propia)

La onda seno sin ruido de la Ilustración 57 y la Ilustración 58 revela la adaptación a un solo tono y en un entorno ideal muestra una muy rápida y muy buena adaptación, algo extraño que hay que anotar es la presencia de armónicos pares de la señal cancelada que no pueden ser atribuidos a la señal cancelada, aunque anotando que son de muy baja intensidad.

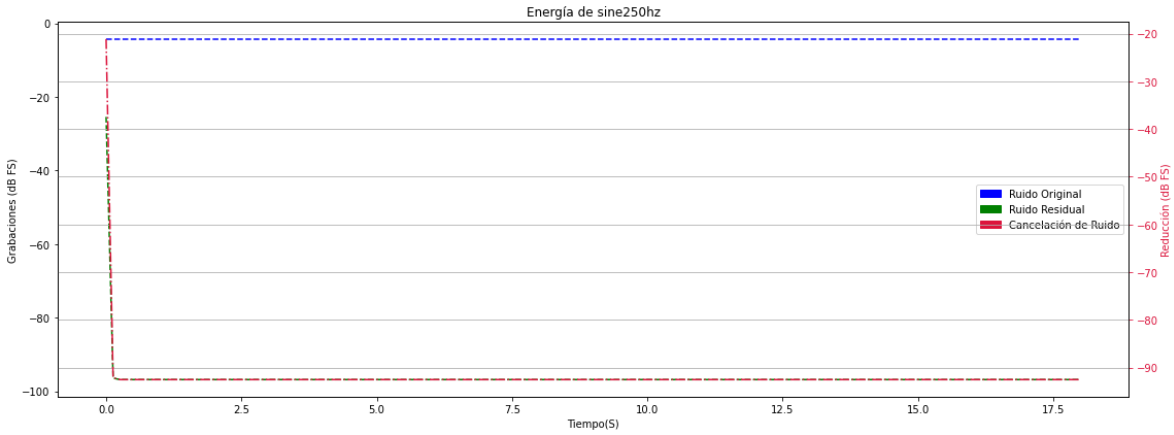


Ilustración 57: Reducción de intensidad en sine250hz (Imagen Propia)

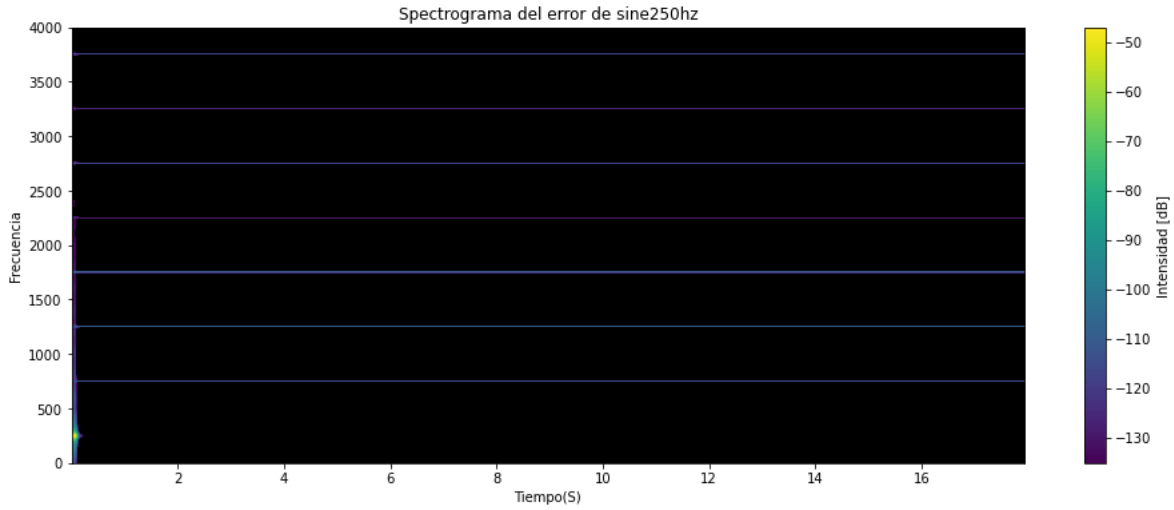


Ilustración 58: Espectrograma del ruido residual de sine250hz (Imagen Propia)

Para STRAFFIC en la Ilustración 59 e Ilustración 60 se muestra que la reducción de sonido inicia bien y esta se desmejora a medida que el camión acelera y se recupera a medida que el camión deja de acelerar, aunque la recuperación de la reducción no es tan grande como la que perdió.

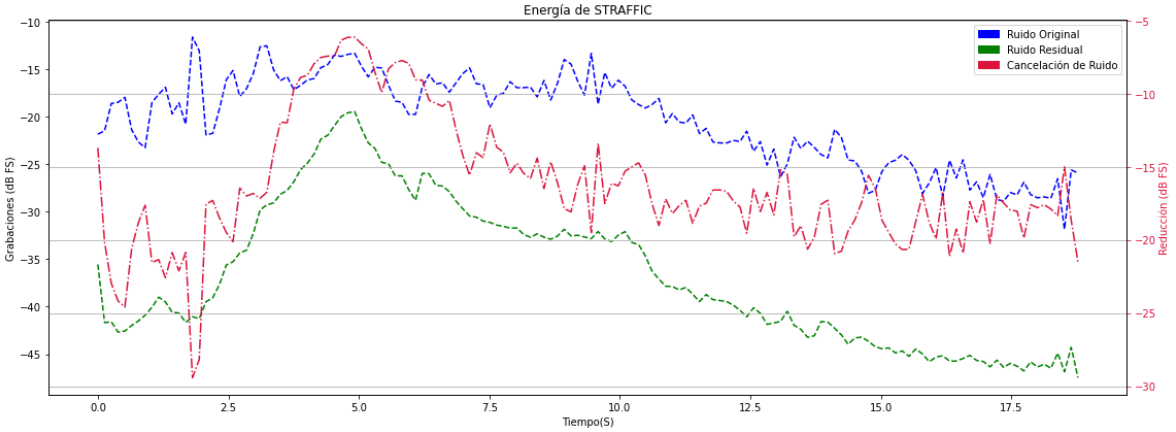


Ilustración 59: Reducción de intensidad en STRAFFIC (Imagen Propia)

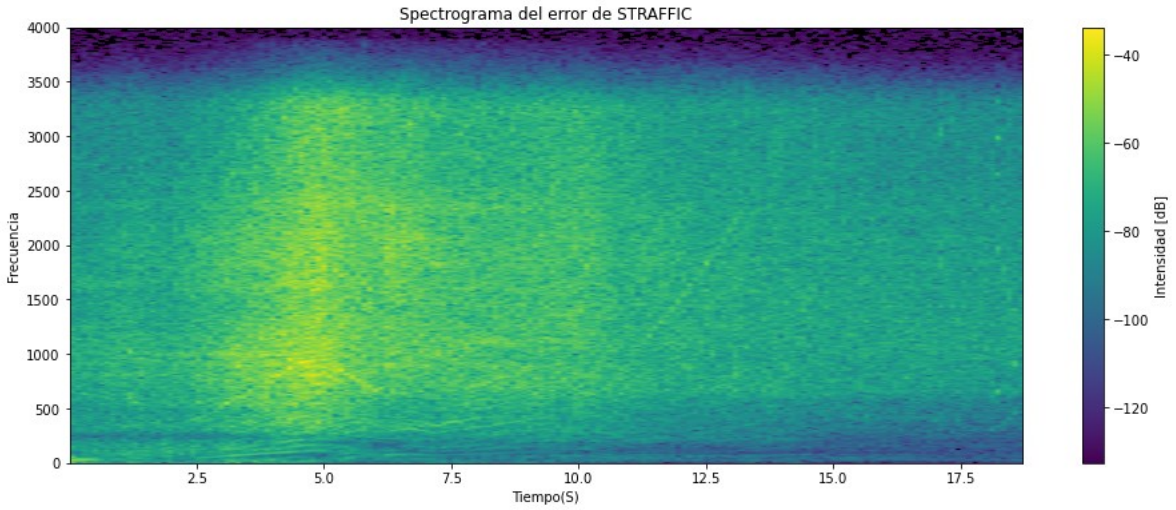


Ilustración 60: Espectrograma del ruido residual de STRAFFIC (Imagen Propia)

La reducción del ruido en TBUS es sustancial como lo revela la Ilustración 61 y la Ilustración 62 pero solo porque la grabación mantiene un volumen muy bajo durante todo el trayecto de esta. En este caso se nota de forma muy patente la característica de que es esta eliminando el ruido de más baja frecuencia y que se está dejando el ruido de alta frecuencia.

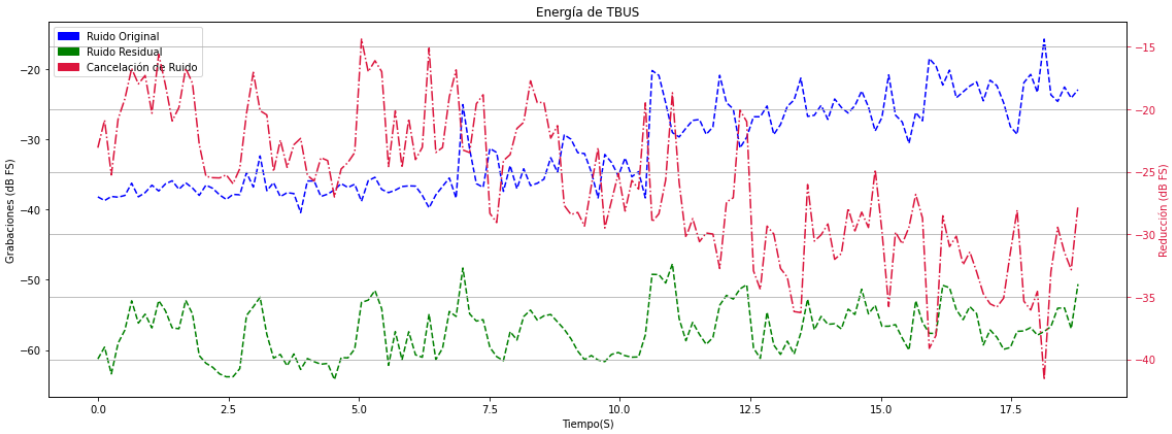


Ilustración 61: Reducción de intensidad en TBUS (Imagen Propia)

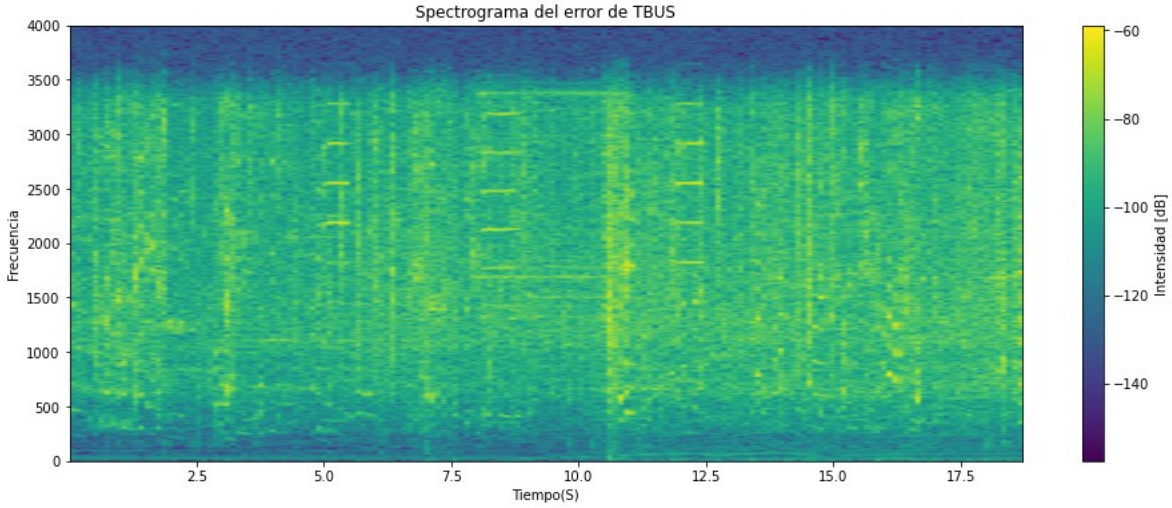


Ilustración 62: Espectrograma del ruido residual de TBUS (Imagen Propia)

Las situaciones acaecidas en TMETRO muestran en la Ilustración 63 y la Ilustración 64 una notable reducción para el caso del movimiento de las puertas, no así para la voz, dada su corta duración ni para el tono de cierre.

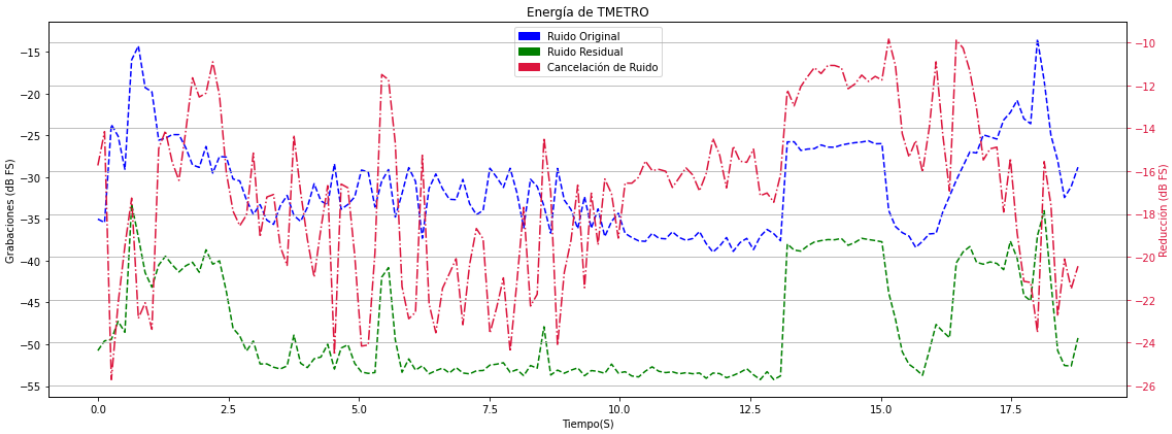


Ilustración 63: Reducción de intensidad en TMETRO (Imagen Propia)

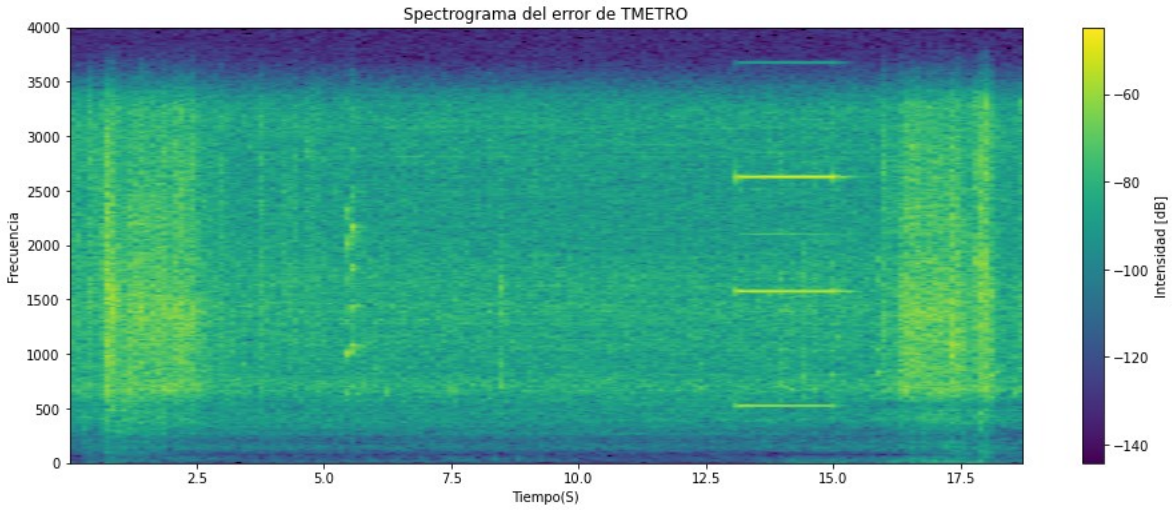


Ilustración 64: Espectrograma del ruido residual de TMETRO (Imagen Propia)

3.4 Cuantización de punto fijo

Era necesario establecer la cuantización de bits para la limitación de los coeficientes, esta fue dilucidada mediante el histograma del filtro LMS con entradas normalizadas entre -1 y 1. Con el fin de determinar que cuantización podría funcionar se corrieron todas las grabaciones por separado en el código de LMS y se hizo el histograma con todos los valores de los coeficientes en la Ilustración 66. Hallándose que la mayoría de los valores están desde el centro hasta el extremo izquierdo como lo muestra la Ilustración 65, salvo por un pico que al hacer una ampliación en la Ilustración 66 se encuentra que está entre los valores 6.33484e-05 y 6.78733e-05, valores que pueden ser representados por 14

bits de coma binaria, por lo que los 30 bits de coma binaria que es lo máximo que soporta el microcontrolador son más que suficientes para representar la mayor parte de los valores con un valor mínimo de representación de 9.31322e-10.

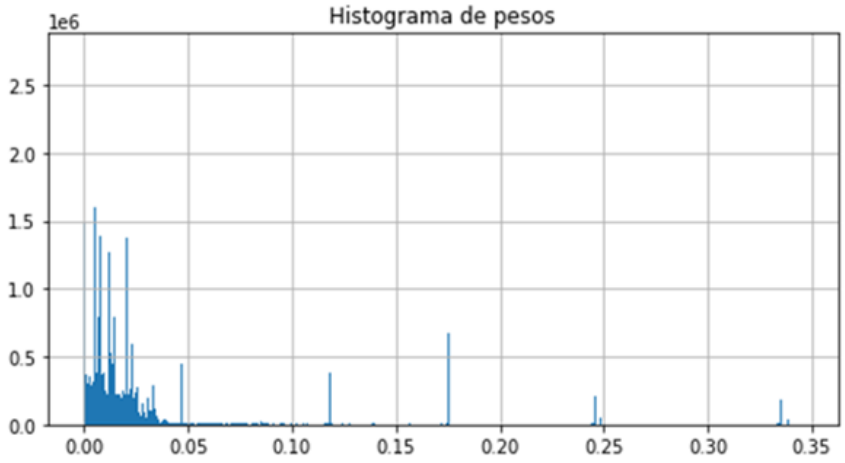


Ilustración 65: Histograma de los pesos W con entrada normalizada (Imagen Propia)

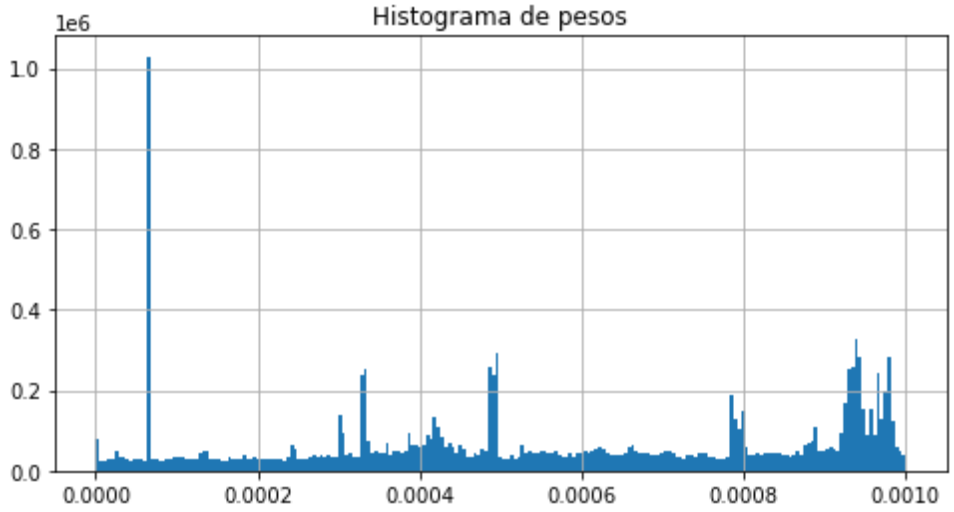


Ilustración 66: Ampliación del histograma de pesos del LMS de entrada normalizada (Imagen Propia)

Se hizo la ejecución de los códigos tanto con el tipo de dato de punto flotante como con el tipo de dato de punto fijo y las diferencias son indistinguibles a la luz de las métricas usadas.

3.5 Desempeño de los algoritmos en el microcontrolador

Con el fin de medir el desempeño de los algoritmos se define la tarea de la Ilustración 67, un timer configurado para interrumpir el sistema cada segundo

trabaja en paralelo con una tarea que tiene la mínima programación requerida para ejecutar los algoritmos de interés, los programas son compilados con la bandera de optimización -O2 y con la frecuencia del microcontrolador establecida en 240MHz. Los resultados de las pruebas con los algoritmos de Ziggurat y los algoritmos LMS punto flotante y LMS de punto fijo con un camino primario de 500 y un camino secundario de 100 se revelan en la Tabla 2.

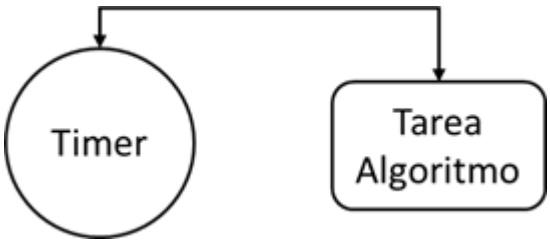


Ilustración 67: Tarea para prueba de algoritmos (Imagen Propia)

Programa	Ejecuciones por Segundo
Algoritmo Ziggurat	580000
LMS en punto flotante	873
LMS en punto fijo	94000

Tabla 2: Ejecuciones de programas de interés

Se puede apreciar de los resultados que el generador de números aleatorios es más que satisfactorio para realizar el modelado y que la ganancia ente el algoritmo de punto fijo respecto al punto flotante es de 107 veces, lo cual está dentro de lo esperado según el benchmark de Dhraystone 2.1.

3.6 Pruebas de interconexión

El micrófono I2S conectado con cable de 20 cm exhibe su comportamiento desde que se energiza el microcontrolador en la Ilustración 68, lo primero que se nota es que hay un tiempo de estabilización del micrófono y es por ellos que los primeros segundos son transientes e incorrectos, inclusive se puede ver como una parte de la grabación pasa desde un máximo al valor de estabilización hacia el inicio, del valor de estabilización se dice que no es cero y que está contaminado por ruido DC, . Se anota que el comportamiento con un más corto cable de 10 cm es indistinguible del desempeño con 20 cm.

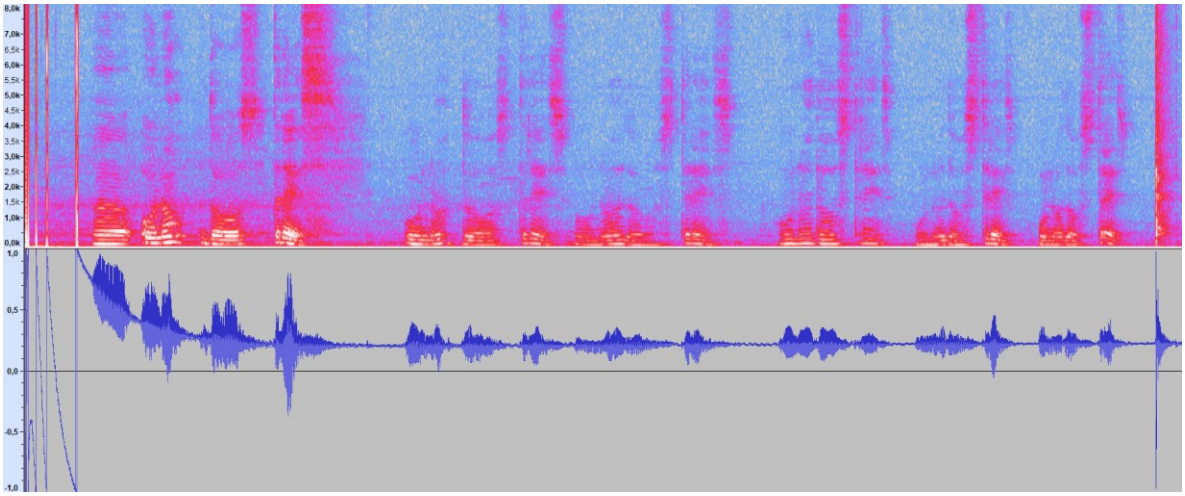


Ilustración 68: Señal I2S con cable de 20 cm (Imagen Propia)

Para las pruebas de interconexión del micrófono donde se usó el protocolo I2S y un cable de CAT6 de 1.5 M se muestran los resultandos en la Ilustración 69: Prueba I2S con cable de 1.5M. Se puede visualizar la presencia de un ruido de baja frecuencia superior al que se encontraba antes mientras el micrófono está en silencio y hay presencia de ruido de alta frecuencia, especialmente hacia el final de la grabación donde se inducen sonidos audibles. De este comportamiento se deduce que el ruido eléctrico está resultando en que la comunicación es infructuosa. Las causas de este ruido pueden ser tres: que debido el ruido introducido por la diferencia del potencial de las tierras del ESP32 y el micrófono hace que los niveles de tensión lleguen a un punto de indeterminación en la comunicación, por eso, un cero se interpreta como uno y un uno como cero, de allí el ruido de alta frecuencia, otra opción radica en la presencia de ruido electromagnético que dificulta la diferenciación de unos y ceros en la comunicación y finalmente, es posible estar en frente de un problema de compatibilidad electromagnética debido a una posible diferencia de impedancias que evita que la energía de la comunicación llegue a el receptor. Esto podría ser determinado mediante el uso de un osciloscopio, un medidor de voltajes en el tiempo, sin embargo, ante la carencia de dicho equipo se descarta el uso del bus a larga distancia y se apela al uso de un protocolo de comunicación más robusto como UART para realizar la comunicación a 1.5 M.

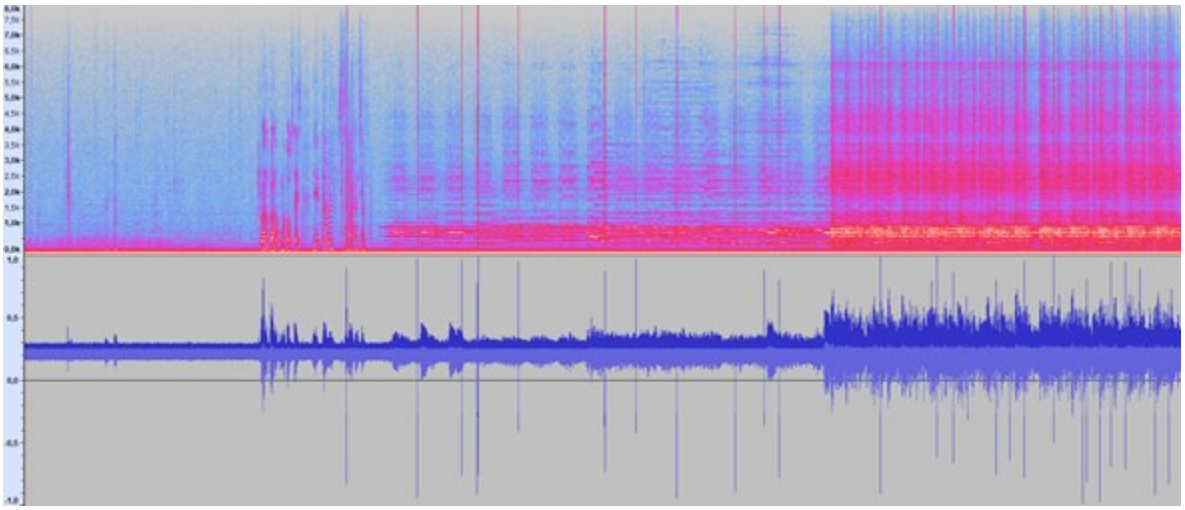


Ilustración 69: Prueba I2S con cable de 1.5M (Imagen Propia)

Con el fin de probar a la conexión UART se estableció la conexión por medio del cable CAT6 de 1.5M y se comunicaron 500 caracteres, recuperándose la totalidad de ellos sin problema, por lo que este protocolo es el que finalmente se usó.

4 Conclusiones

Durante el desarrollo del trabajo se llegó a una base de datos en la sección de Base de datos y preprocesado que comparada con las demás resultó ser variada y de una duración ni muy larga ni muy corta, ideal para el trabajo de cancelación de ruido, a pesar de que dicha base de datos es ignorada en favor de otras bases de datos más establecidas como NOISEX-92 y Aurora-2, se considera que es idónea para trabajos que involucren ruido ambiental.

El modelo de programación del microcontrolador presentado en Consideraciones de programación del microcontrolador y en Esquema de tareas del microcontrolador supone una barrera para la implementación, puesto que cada microcontrolador requiere de una configuración ligeramente distinta por lo que las configuraciones de periféricos y las tareas definidas son dependientes de hardware usado y han de tenerse en cuenta a la hora de programar el microcontrolador, aún con el uso de una capa de abstracción como lo es el sistema operativo embebido FreeRTOS.

El proceso de cancelación de ruido implementado con el Filtrado Fx-LMS se encuentra exitoso, puesto que muestra una respectiva reducción la energía en decibelios, sin embargo, parece que la cancelación tiene preferencia por las bajas frecuencias aún en el algoritmo, lo cual contradice el hecho de que el filtrado LMS es lineal y esto es un absurdo dado que este proviene de la estimación lineal ideal, esto se puede dar puesto que el proceso de análisis es manual y está sujeto al sesgo del evaluador y por ello se requiere de otra métrica que compare un par de espectrogramas de forma analítica.

La Cuantización de punto fijo resultó exitosa, puesto que logró aumentar la capacidad del microcontrolador para ejecutar el algoritmo Fx-LMS en los dos órdenes de magnitud predichos por Dhrystone2.1 respecto a los resultados de Linpack y llevó al microcontrolador hasta una posible tasa de muestreo que implica frecuencias ultrasónicas, por lo que su ejecución en un hardware más modesto es factible. Por otro lado, también se concluye que, para unos datos de audio cuantizados en 16 bits, un tipo de dato de 32 bits resulta suficiente para realizar el procesamiento digital.

La comunicación embebida a larga distancia hecha en Pruebas de interconexión presenta dificultades puesto que la mayoría de los protocolos no están pensados para distancias más largas que las que se pueden encontrar en una tarjeta electrónica y estos caen presa del ruido y de la pérdida de potencial

de las tierras, cuando se requieran realizar comunicaciones a largas distancias se requiere o bien usar los protocolos que lo soporten, también conocidos como protocolos de comunicación hacia afuera de la tarjeta o de hardware específico que convierta las señales a protocolos que soportan la comunicación de larga distancia como RS-232 ó RS-485.

5 Trabajos Citados

Acousteen. (s.f.). Recuperado el 5 de Diciembre de 2020, de http://www.steeneken.nl/7-noise-data-base/

Adafruit. (22 de Febrero de 2017). *Adafruit I2S MEMS Microphone Breakout*. Obtenido de Adafruit: https://learn.adafruit.com/adafruit-i2s-mems-microphone-breakout/

Audio Enginnering Society. (2015). AES17-2020. New York. Recuperado el 7 de Julio de 2020, de https://www.aes.org/publications/standards/search.cfm?docID=21

Ballou, G. (2009). Types of Transducers. En *Electroacoustic Devices: Microphones and Loudspeakers* (págs. 23-48). Boston: Elsevier.

Barron, R. F. (2003). Silencer Design. En R. F. Barron, *Industrial Noise Control And Acoustics* (págs. 341-416). New York: Marcel Dekker.

Bies, D. A., Hansen, C. H., & Howard, C. Q. (2018). *Handbook of Engineering Noise Control.* Miami: CRC Press.

Burkardt, J. (2013 de Octubre de 2013). *Ziggurat Random Number Generator (RNG)*. Obtenido de Florita State University - Department of Scientific Computing: https://people.sc.fsu.edu/~jburkardt/c_src/ziggurat/ziggurat.html

Circuit.pk. (2020). *TDA2030A*. Obtenido de Circuit.pk: https://circuit.pk/wp-content/uploads/2018/10/Single-Power-Audio-Amplifier-Module-TDA2030A-1.jpg

Crocker, M. J. (2007). *Noise And Vibration Control.* New Jersey: Wiley.

Damien. (24 de Febrero de 2010). *Generate random numbers following a normal distribution in C/C++*. Obtenido de Stack Overflow: https://stackoverflow.com/questions/2325472/generate-random-numbers-following-a-normal-distribution-in-c-c

Dean, D., Sridharan, S., Vogt, R., & Mason, M. (2010). The QUT-NOISE-TIMIT corpus for evaluation of voice activity detection algorithms. *Proceedings of Interspeech*, (págs. 3110-3113). Makuhari.

Dongarra, J. (1988). The Linpack Benchmark, and explanation. Recuperado el 13 de Septiembre de 2020, de http://www.netlib.org/utk/people/JackDongarra/PAPERS/The-LINPACK-Benchmark-An-Explanation.pdf

Dzhambov, A. M. (2015). Long-term noise exposure and the risk for type 2 diabetes: A meta-analysis. *Noise and Health, 17*(74), 23. doi:10.4103/1463-1741.149571

Eargle, J. (2005). First Order Directional Microphones. En *The Microphone Book* (2nd ed., págs. 66-72). Boston, Estados Unidos de América: Elsevier.

Elliott, S. J. (2007). Active Noise Control. En M. J. Crocker, *Handbook of Noise and Vibration Control* (págs. 761-769). Miami: Wiley.

Espressif. (1 de Septiembre de 2016). *ESP32 Forum*. Recuperado el 17 de Septiembre de 2020, de https://esp32.com/

Espressif. (Noviembre de 2017). *ESP32 Overview*. Obtenido de Espressif: https://www.espressif.com/en/products/socs/esp32/overview

Farhang-Boroujeny, B. (2013). *Adaptive Filters, Theory and Applications.* Chichester, UK: WILEY.

Foster, P., Sigtia, S., Krstulovic†, S., Jon, B., & Plumbley, M. D. (2015). CHiME-HOME: A Dataset for Sound Source Recognition in a Domestic Environment. *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*.

FreeRTOS. (2011). Obtenido de https://freertos.org/

Go Jimmy Pi. (2018). Obtenido de Github: https://external-content.duckduckgo.com/iu/?u=https%3A%2F%2Fraw.githubusercontent.com%2Fgojimmypi%2FESP32%2Fmaster%2Fimages%2FmyESP32%2520DevKitC%2520pinout.png&f=1&nofb=1

Hansen, C., Snyder, S., Qiu, X., Brooks, L., & Moreau, D. (2012). 13.3.1.1 Sample Rate Selection. En *Active Control Of Noise And Vibration* (2 ed., Vol. II, pág. 408). Miami: CRC Press.

Hansen, C., Snyder, S., Qiu, X., Brooks, L., & Moreau, D. (2012). Active Control of Enclosed Sound Fields. En *Active Control of Noise and Vibration* (págs. 983 - 1092). Miami: CRC Press.

Hansen, C., Snyder, S., Qiu, X., Brooks, L., & Moreau, D. (2012). Active Control of Free-Field Sound Radiation. En *Active Control of Noise and Vibration* (2 ed., Vol. 2, págs. 823-982). Miami: CRC Press.

Hansen, C., Snyder, S., Qiu, X., Brooks, L., & Moreau, D. (2012). Active Control of Noise Propagating in Ducts. En *Active Control of Noise and Vibration* (2 ed., Vol. 1, págs. 697-816). Miami, Florida, Estados Unidos de América: CRC Press.

Hansen, C., Snyder, S., Qiu, X., Brooks, L., & Moreau, D. (2012). Feedforward Control System Design. En *Active Control of Noise and Vibration* (Vol. I, pág. 410). Miami: CRC Press.

Hansen, C., Snyder, S., Xiaojun, Q., Brooks, L., & Moreau, D. (2012). Background. En *Active Control Of Noise And Vibration* (págs. 1-3). CRC Press.

Hansen, C., Snyder, S., Xiaojun, Q., Brooks, L., & Moreau, D. (2012). Control of Noise Propagating in Ducts. En *Active Control of Noise and Vibration* (págs. 697-882). CRC Press.

Hansen, C., Snyder, S., Xiaojun, Q., Brooks, L., & Moreau, D. (2012). Neural Network-Based Feedfoward Active Control. En *Active Control of Noise and Vibration* (2nd ed., Vol. 2, págs. 662-677). Miami: CRC Press.

Hansen, C., Snyder, S., Xiaojun, Q., Brooks, L., & Moreau, D. (2012). Spectral Analysis. En *Active Control of Noise and Vibration, Volume I* (págs. 190-211). CRC Press.

Harris, F. J. (1978). On the use of Windows for Harmonic Analysis with the Discrete Fourier Transform. *Proceedings of the IEEE*, 51–83.

Hirsch, H.-G., & Pearce, D. (18 de Septiembre de 2000). *Aurora-2*. Recuperado el 5 de Diciembre de 2020, de Aurora - Audio Speech Recogniton: http://aurora.hsnr.de/index-2.html

Hunter, J., Dale, D., Firing, E., & Droettboom, M. (2002). *Matplotlib*. Obtenido de https://matplotlib.org/

I2S Timing. (19 de Septiembre de 2011). Obtenido de https://commons.wikimedia.org/wiki/File:I2S_Timing.svg

IEC. (Septiembre de 2013). *International Electrotechnical Commission Webstore*. Obtenido de International Electrotechnical Commission : https://webstore.iec.ch/publication/5708

IEEE. (12 de Junio de 2008). *IEEE Standards*. Obtenido de https://standards.ieee.org/standard/754-2008.html

Implementación de Sistemas Discretos en el Tiempo. (2007). En J. G. Proakis, & D. G. Manolakis, *Tratamiento Digital de Señales* (págs. 505-520). Madrid: Prentice Hall.

International Telecomunications Union. (17 de Febrero de 2000). Obtenido de https://www.itu.int/rec/T-REC-V.24-200002-I/en

ISO/TC 43. (Agosto de 2003). *Acoustics - Normal equal-loudness-level contours*, 2. Obtenido de International Organization for Standardization: https://www.iso.org/standard/34222.html

Isza, P. (2011). *TFilter*. Recuperado el 6 de Diciembre de 2020, de http://t-filter.engineerjs.com/

Ito, K., Kato, M., Kuwakado, S., & Takagi, M. (1985). *Patente nº JPS6220713A*.

Ivković, J., & Ivković, J. L. (2017). Analysis of the Performance of the New Generation of 32-bit Microcontrollers for IoT and Big Data Application. *7th International Conference on Information Society and Technology ICIST*. Kopaonik.

Keil. (2020). *Dhrystones*. Recuperado el 13 de Septiembre de 2020, de arm Keil: https://www.keil.com/benchmarks/dhrystone.asp

Kido, K., Abe, M., & Morikawa, S. (1984). Suppression of acoustic noise by computer generated sound. *Proceedings of Internoise 84 conference*, 463-468.

Kinoshite, A., & Aoki, H. (s.f.). *1993 Patente nº US5245664A*.

Knowles. (6 de Septiembre de 2015). Datasheet SPH0645LR4H-B Rev C. Chicago, Illinois, Estados Unidos de América. Obtenido de https://www.knowles.com/docs/default-source/model-downloads/sph0645lm4h-b-datasheet-rev-c.pdf?Status=Master&sfvrsn=c1bf77b1_4

Lengua, R. A. (s.f.). *Diccionario*. Obtenido de https://dle.rae.es/ruido

Leventhall, H. (2004). Low frequency noise and annoyance. *Noise & Health, 6*(23), 59-72.

Linear Operators. (2020). En M. V. Markin, *Elementary Operator Theory* (págs. 161-206). Berlín, Alemania: De Gruyter.

Lueg, P. (9 de Junio de 1936). *Estados Unidos Patente n° US2043416A*.

Luong, J. (29 de Junio de 2018). *Linear Power Supplies for Audiophiles – Getting Closer to Live*. Recuperado el 20 de Noviembre de 2020, de Audio Bacon: https://audiobacon.net/2018/06/29/linear-power-supplies-for-audiophiles-getting-closer-to-live/

Marsaglia, G., & Tsang, W. W. (2000). The Ziggurat Method for Generating Random Variables. *Journal of Statistical Software, 5*(8).

Miloyip. (10 de Julio de 2015). *Normally Distributed Random Number Generator Benchmark*. Obtenido de Github: https://github.com/miloyip/normaldist-benchmark

Mitsuhata, S. (2003). *Patente n° JP2005134632A*.

Motorola. (21 de Enero de 2000). SPI Block Guide V4.01. Recuperado el 23 de Mayo de 2020, de https://www.nxp.com/files-static/microcontrollers/doc/ref_manual/S12SPIV4.pdf

Münzel, T., Schmidt, F. P., Steven, S., Herzog, J., Daiber, A., Sørensen, M., & Cardiol, J. C. (2018). Environmental Noise and the Cardiovascular System. *Journal of the American College of Cardiology, 71*(6), 688-697. doi:10.1016/j.jacc.2017.12.015

Muzet, A. (2007). Environmental noise, sleep and health. *Sleep Medicine Reviews, 11*(2), 135-142. doi:doi:10.1016/j.smrv.2006.09.001

Nelson, P., & Elliott, S. (1992). *Active Control of Sound.* London: Academic Press.

NOISEX-92. (13 de Agosto de 1996). Obtenido de http://www.speech.cs.cmu.edu/comp.speech/Section1/Data/noisex.html

NTi Audio. (2020). Frequency-Weightings for Sound Level Measurements. Recuperado el 29 de Abril de 2020, de https://www.nti-audio.com/en/support/know-how/frequency-weightings-for-sound-level-measurements

Olayinka, O. S. (2012). Noise Pollution in Urban Areas: The Neglected Dimensions. *Environmental Research Journal,* 6(4), 259-271. doi:10.3923/erj.2012.259.271

Oppenheim, A. V. (1975). Obtenido de MIT Open Course Ware: https://ocw.mit.edu/resources/res-6-008-digital-signal-processing-spring-2011/video-lectures/demonstration-1-sampling-aliasing-and-frequency-response-part-1/

Pairs Of Random Variables. (2006). En J. A. Gubner, *Correlation and covariance* (págs. 91-92). New York, Estados Unidos de Norteamerica: Cambridge University Press.

Philips Semiconductors. (1996). *I2S bus specification.*

Price, J. (7 de Julio de 2007). *Understanding dB*. Recuperado el 12 de Mayo de 2020, de Jim Price: http://www.jimprice.com/prosound/db.htm

Python. (2001). (Python Software Foundation) Obtenido de https://www.python.org

Rebala, G., Ravi, A., & Churiwala, S. (2019). *An Introduction to Machine Learning.* Cham, Suiza.

Recio, A., Linares, C., Banegas, J. R., & Díaz, J. (2016). Road traffic noise effects on cardiovascular, respiratory, and metabolic health: An integrative model of biological mechanisms. *Environmental Research, 146*(1), 359-370. doi:10.1016/j.envres.2015.12.036

Ring, M. (2016). *Patente nº US20160353197A1.*

Salikuddin, M., & Tanna, H. K. (1985). *Patente nº US4689821A.*

Search Methods. (2013). En F.-B. Behrouz, *Adaptive Filters - Theory and Practice* (págs. 119-130). Chichester: Wiley.

Soltero, M., Zhang, J., & Cockril, C. (Junio de 2002). RS-422 and RS-485 Standards Overviewand System Configurations.

speechdnn. (Agosto de 2018). *Github*. Recuperado el 5 de 12 de 2020, de https://github.com/speechdnn/Noises

ST Microelectronics. (Octubre de 2000). TDA2030A datasheet Rev 2. Recuperado el 10 de Mayo de 2020, de https://www.st.com/resource/en/datasheet/CD00000129.pdf

Svec, C. (2012). FreeRTOS. En A. Brown, & G. Wilson, *The Architecture of Open Source Aplications.*

Teachman, M. (20 de Marzo de 2019). *New I2S Microphone*. Obtenido de Hackaday: https://hackaday.io/project/162059-street-sense/log/160705-new-i2s-microphone

Texas Instruments. (Mayo de 2012). *PCM5102A datasheet, product information and support*. Obtenido de Texas Instruments: https://www.ti.com/product/PCM5102A

Vanhoucke, V., Senior, A., & Mao, M. Z. (2011). *Improving the speed of neural networks on CPUs.* Google.

Vibraciones y Ondas. (2006). En P. G. Hewitt, *Física Conceptual* (págs. 369-370). Ciudad de México: Pearson.

Virtanen, a., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., . . . al. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods, 3*(23), 261-272. Obtenido de https://www.scipy.org/

Voras, I. (1 de Marzo de 2020). *Fixed Point Math Library for C*. (SourceForge) Obtenido de https://sourceforge.net/projects/fixedptc/

Waterline Media. (2003). The Frequency Spectrum, Instrument Ranges, and EQ Tips. Recuperado el 1 de Noviembre de 2020, de http://www.guitarbuilding.org/wp-content/uploads/2014/06/Instrument-Sound-EQ-Chart.pdf

WHO Regional Office for Europe. (2018). *Environmental Noise Guidelines for the European Region* . http://www.euro.who.int/en/health-topics/environment-and-health/noise/publications/2018/environmental-

noise-guidelines-for-the-european-region-2018: WHO Regional Office for Europe.

Widder, J., & Morcelli, A. (14 de Mayo de 2014). *EDN*. Obtenido de https://www.edn.com/basic-principles-of-mems-microphones/

Wiener Filters. (2013). En B. Farhang-Boroujeny, *Adaptative Filters Theory and Applications* (págs. 48-50). Chichester: Wiley.

Wiener Filters. (2013). En B. Farhang-Boroujeny, *Adaptive Filters Theory and Applications* (págs. 48-58). Chichester: Wiley.

Wikimedia. (30 de Abril de 2005). *Artificial neuron*. Obtenido de Wikimedia: https://en.wikipedia.org/wiki/File:Artificial_neuron.png

Wikimedia. (19 de Diciembre de 2006). File:SPI single slave.svg. doi:https://commons.wikimedia.org/wiki/File:SPI_single_slave.svg

Wikimedia. (24 de Abril de 2008). *FixedPoint.jpg*. Obtenido de NYCRESISTOR: https://wiki.nycresistor.com/wiki/File:FixedPoint.jpg

Wikimedia. (12 de Octubre de 2008). Smoothed ripple.svg. Obtenido de https://en.wikipedia.org/wiki/File:Smoothed_ripple.svg

Wikimedia. (Mayo de 2009). File:StrcPrstSkrzKrk.png. Obtenido de https://commons.wikimedia.org/wiki/File:StrcPrstSkrzKrk.png

Wikimedia. (7 de Mayo de 2019). *Colored neural network*. Obtenido de Wikimedia: https://commons.wikimedia.org/wiki/File:Colored_neural_network.svg

Wolf, M. (2014). *High-Performance Embedded Computing - Architectures, Applications, and Methodologies* (2nd ed.). Miami: Elsevier. doi:https://doi.org/10.1016/C2012-0-07058-5

Yates, R. (15 de Septiembre de 2020). Fixed-Point Arithmetic: An Introduction.

Yuko, S., Susumu, S., & Sekiguchi, Y. (1992). *Japón Patente n° JP23978491*.

Yunshuang, J., Li, Y., & Xuan, Z. (2013). *Patente n° CN203444957U*.

Zhao, H., Liu, D., & Li1, H. (2020). *Efficient Integer-Arithmetic-Only Convolutional Neural Networks.* Obtenido de https://arxiv.org/pdf/2006.11735v1.pdf

Zhoua, Y.-L., Zhanga, Q.-Z., Lib, X.-D., & Gan, W.-S. (2004). Analysis and DSP implementation of an ANC system using a filtered-error neural network. *Journal of Sound and Vibration*, 1-25.

Ziegler, E. W., & Gardner, J. W. (1992). *Patente nº US5097923A*.

Zinemanas, P., Cancela, P., & Rocamora, M. (17 de Julio de 2017). *MAVD-traffic dataset*. Recuperado el 5 de Diciembre de 2020, de https://zenodo.org/record/3338727