UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA
ELÉCTRICA – TELECOMUNICACIONES

IMPLEMENTACIÓN DE UN SISTEMA DE CAPTURA, TRANSMISIÓN Y MEZCLA DE AUDIO DIGITAL

TESIS
QUE PARA OPTAR POR EL GRADO DE:
MAESTRO EN INGENIERÍA

PRESENTA:
ING. PEDRO FELIPE LA ROTTA SANTOS

TUTORA PRINCIPAL
DRA. FATIMA MOUMTADI
FACULTAD DE INGENIERÍA, UNAM.

MÉXICO, D.F. JUNIO DE 2013

JURADO ASIGNADO

Presidente:	Dr. Oleksandr Martynyuk
Secretario:	M. en I. Larry Escobar Salguero
1er. Vocal:	Dra. Fatima Moumtadi.
2do. Vocal:	Dr. Bohumil Psenicka
3er. Vocal:	Dr. Esaú Vicente Vivas

AGRADECIMIENTOS

El soporte y apoyo provisto por mi madre no tiene comparación. Sin ella este trabajo hubiera sido imposible de realizar. Con mucho agradecimiento y admiración, este trabajo va dedicado a Nury Santos, mi madre.

A la Universidad Nacional Autónoma de México, la cuál me brindo unas bases académicas de excelente calidad.

A mi familia, la cual me apoyó, soportó y alentó en cada paso del proceso para cumplir esta meta. En especial a mi padre Jorge La Rotta y a mis tíos Clemencia Santos y Rafael Rodríguez.

A mis amigos, los cuales me ayudaron a los largo de mi formación y alegraron el camino recorrido para llegar hasta aquí.

Por último quiero agradecer a mi tutora, la Dr. Fatima Moumtadi, que me ha apoyado y orientado desde el inicio de la licenciatura.

RESUMEN

En el proyecto de investigación se realizó el desarrollo de un sistema funcional de audio en vivo, el cual puede manejar la información sin retrasos considerables de tiempo y permita mezclar los canales de diferentes fuentes.

El sistema realizado consta de dos partes principales, un micrófono digital y la consola emulada en la computadora; así como un elemento de unión el cual es una red de área local

Se diseñó un sistema 100% digital, el cual se encarga de amplificar y adecuar la señal para ser digitalizada dentro el microcontrolador. Se crean paquetes de diez muestras y se envían a través de una red de área local hasta la computadora.

Dentro de la computadora se programó una consola virtual, la cual permite capturar la información proveniente del micrófono, desempaquetarla, ecualizarla, aplicar control sobre el volumen y el balance y finalmente tener una salida por el hardware de la computadora.

Por último todo el sistema se integró y se realizaron pruebas para conocer el desempeño y la tasa de bits mínima necesaria para su correcto funcionamiento.

ÍNDICE

ÍNDICE DE FIGURAS

Capítulo 1: Estado del arte

Capítulo 2: Desarrollo del Sistema de Captura

Capítulo 3: El Desarrollo del Programa

Capítulo 4: Pruebas y análisis de resultados

OBJETIVOS

El objetivo principal es implementar un sistema que capture una señal de audio y la digitalice directamente en el origen, posteriormente la procese en la computadora y finalmente nos dé una salida con buena calidad. Todos los procesos del sistema van a manejar el audio de manera digital.

Para poder realizar la captura de la señal es necesario desarrollar un sistema de adquisición que tenga la capacidad de digitalizar la señal y que pueda enviar los datos por una red de área local.

Para poder manipular los datos capturados es necesaria la realización de una consola de audio, con una interfaz gráfica operable por el usuario, que asegure la correcta conexión con el micrófono y permita manipular características propias del sonido como el balance, volumen y ecualización.

Finalmente es necesaria la aplicación de una prueba de calidad para dar un valor cuantitativo a la degradación que sufre la señal al pasar por el sistema.

INTRODUCCIÓN

ANTECEDENTES

Desde el origen de las grabaciones de audio, son pocos los equipos que han digitalizado la información desde el primer paso y mantenido su formato digital durante todo el tiempo en el sistema y estos sistemas no manejan la información en vivo, es decir, no se ha planteado un esquema totalmente digital para la transmisión de audio en espectáculos o cabinas de grabación donde se manejen todos los canales con formato digital.

Al controlar el sonido en la computadora podemos establecer un esquema digital desde el inicio hasta el final del proceso, reduciendo con esto pérdidas de potencia y aumento del ruido, situación existente en los esquemas actuales.

En la tesis de licenciatura se elaboró el esquema básico y componentes mínimos del sistema de audio digital, por lo que el desarrollo principal está basado en el trabajo realizado con anterioridad[1].

DEFINICIÓN DEL PROBLEMA

En la actualidad, al momento de administrar los canales de audio en la realización de espectáculos musicales, grabaciones y transmisiones de radiodifusión, se utiliza una consola de audio, generalmente analógica, que controla de manera independiente cada uno de los canales de audio (uno por micrófono en escenario o cabina).

Cada micrófono exige la existencia de un cable para su conexión a la consola, lo que genera problemas como la disminución de la potencia de

[1] (La Rotta Santos 2012)

la señal recibida debido a las pérdidas en la línea, la introducción de ruido y el costo derivado del cableado mismo.

DEFINICIÓN DE LA SOLUCIÓN

El proyecto plantea modificar el esquema de conexión actual, aumentando la calidad, simplificando el sistema de audio, reduciendo las pérdidas de potencia y minimizando los costos del sistema. Esto es logrado a través de la eliminación de la gran cantidad de cables necesarios para poder transportar todas las señales.

La solución planteada para éste problema es digitalizar la señal directamente en su fuente, es decir desde el micrófono, y enviar la información utilizando una conexión de red Ethernet hacia la consola, esto gracias a un conmutador de red al que se conectarán todos los micrófonos, la consola y las bocinas del sistema de audio.

La ventaja de usar un esquema de red, es su posible expansión posterior utilizando los mismos componentes básicos, la facilidad para conectarse a internet y enviar la información a una terminal remota, y el soporte para manejar video, voz y datos sobre la misma infraestructura.

Otro problema actual es la necesidad de tener una consola de audio física para manejar cada canal, lo que reduce la movilidad del sistema e incrementa el espacio mínimo de instalación del sistema.

De esta manera la segunda modificación al esquema actual de conexión lo constituye el uso de una consola virtual: una aplicación que permite manejar cada uno de los canales de audio emulando por completo las posibilidades de ecualización y control del volumen que se tienen actualmente en los equipos mezcladores de sonido.

Tomando en cuenta los requerimientos del proyecto, la solución planteada incluye el desarrollo de los siguientes elementos:

- Digitalizar el sonido: Este paso se logra a través de tres etapas. La primera etapa amplifica la señal proveniente del micrófono y la adapta para ser convertida en el convertidor analógico a digital.
  La segunda etapa es un microcontralador que posee el convertidor analógico a digital en su interior. El microcontrolador es el

responsable de digitalizar la señal y empaquetarla para su posterior transmisión.
Finalmente se necesita un controlador de red, que se encarga de enviar los paquetes generados por el microcontrolador a través de una red de área local.

- Una consola de audio dentro de la computadora: Una vez que los paquetes llegan a la computadora es necesario seleccionar un micrófono (en caso de haber varios) para ser asociado a un canal de la consola y posteriormente se procede a desempaquetar la información proveniente del micrófono.
  Una vez que se tiene la información, se procede a ecualizar la señal, ajustar el volumen, seleccionar el balance del canal y finalmente enviar los datos a una línea de salida para su reproducción.

Los dos elementos están unidos por una red de área local, la cual se compone de un conmutador y cables UTP para la conexión de los dispositivos.

CAPÍTULO 1: ESTADO DEL ARTE

1.1 ACÚSTICA

Los sonidos siempre han estado presentes en nuestro entorno, pero su estudio no se desarrolló hasta la antigua Grecia, donde filósofos como Pitágoras estudiaron las cuerdas vibrantes y sonidos musicales.
Los griegos estaban pendientes de la acústica de sus teatros y en algunos casos se utilizaban cornetas para poder amplificar las ondas sonoras que producían los actores.

La primera persona en escribir respecto al sonido y sus interacciones con los objetos y recintos, fue el arquitecto romano Marco Vitruvio en su obra monumental *De Architectura*, donde se incluye un análisis completo a la acústica de un teatro: "Debemos de escoger un sitio donde la voz caiga suavemente, y no como una reflexión, de tal forma que no adquiera otro significado para el oído."

Después de una gran pausa, los estudios relacionados con el sonido prosiguieron en el renacimiento, cuando físicos de la época comenzaron a estudiar los efectos sonoros causados por las cuerdas, en este terreno los estudios realizados fueron:

- Primero fue Galileo Galilei, el cual comenzó a estudiar la relación existente entre el sonido de una cuerda y su largo.
- Posteriormente Josheph Sauveur continúo con los estudios de la frecuencia con relación al tono.
- El matemático ingles Brook Taylor propuso una solución dinámica para la frecuencia de una cuerda vibrante basado en la forma que toma la cuerda cuando oscila en su modo fundamental.
- Daniel Bernoulli estipuló una ecuación diferencial parcial para la cuerda vibrante y obtuvo resultados que posteriormente d'Alambert interpretó como ondas que viajan en ambas direcciones a lo largo de la cuerda.

Posteriormente se comenzaron a investigar membranas vibrantes y se desarrollaron modelos matemáticos para explicar su comportamiento, comenzando por S. D. Poisson y su membrana circular, pasando por Sophie Germain que escribió una ecuación de cuarto orden para describir las vibraciones en platos (con lo que ganó el un premio por parte del

emperador francés Napoleón), posteriormente Kirchhoff dio un tratamiento más adecuado a las condiciones de frontera de los objetos vibrantes y finalmente Rayleigh trató membranas y platos vibrantes en su celebrado libro *Theory of Sound*.

De forma paralela a estos estudios, se analizaba la frecuencia y su relación con el tono. Iniciando con Marin Mersenme (1588 – 1648) que pudo asociar el tono de cierta nota con su frecuencia. Para lograrlo estudió el comportamiento de una cuerda grande y estableció una fórmula dependiente de la masa, longitud y tensión de la cuerda, posteriormente aplicó la misma fórmula para cuerdas pequeñas de instrumentos musicales.

Posteriormente Joseph Sauveur mejoró los estudios de frecuencia al contar los golpes de dos notas bajas de un órgano y estableciendo la relación existente entre ellas. También acuñó el término *acustics* para referirse a la ciencia del sonido, abarcando un área más grande que la representada por el campo de la música y sus instrumentos.

Desde finales del siglo XVIII se dieron grandes avances en el desarrollo de la acústica, con la invención del teléfono por parte de Alexander Graham Bell y el fonógrafo por Tomas Alba Edison, ya que fue en ese momento cuando un canal de audio se pudo transmitir o grabar como una representación mecánica, magnética o eléctrica de la señal de audio original.

Los dos dispositivos surgieron como una mejora al sistema de telégrafos existente en el momento de la creación de los dispositivos, mismo sistema que no puede existir sin la existencia de una pila que suministre energía, inventada por Alessandro Volta en 1799. La existencia de la pila es también un requisito para el funcionamiento del teléfono.

El primer dispositivo capaz de grabar los sonidos fue creado por Leon Scott, que en el año de 1857 patentó el *phonautograph*. Este dispositivo era capaz de grabar la forma de onda de una señal pero incapaz de reproducirla.

El dispositivo funcionaba con un diafragma, el cual captaba las ondas mecánicas provenientes de la fuente y por medio de un estilógrafo, podía grabarlas en cilindros de papel o en un vidrio humeado, proceso que se puede observar en la figura 1.1[2].

[2] (Rossing 2007)

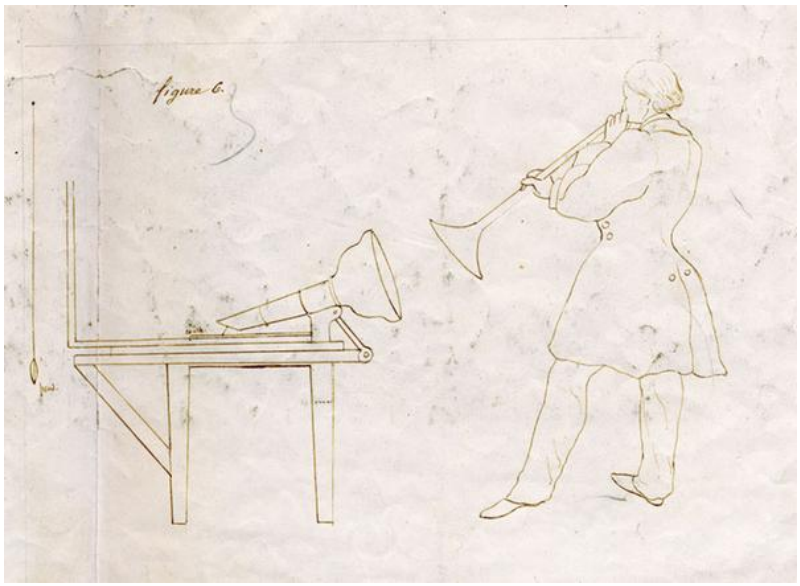

Figura 1.1: Leon Scott y el phonautograph.

Al inicio de las grabaciones el uso de un transductor era innecesario, ya que el almacenamiento se realizaba en un medio físico sensible a las fuerzas mecánicas y la generación de la información tiene un formato de naturaleza mecánica, lo que permitía la grabación de la información sin pasos intermedios de conversión.

1.2 TRANSDUCTORES

El invento del teléfono en 1876, por Alexander Graham Bell (aunque su desarrollo fue alcanzado casi al mismo tiempo por varios investigadores) marcó el inicio de los transductores, ya que era necesaria la conversión de una señal mecánica a una señal eléctrica (y viceversa) para poder lograr la comunicación. Son estos transductores lo que marca el inicio de la vida del micrófono, el cual sigue siendo la clave para el teléfono, la radio y las grabaciones de sonido.

Los primeros micrófonos que se utilizaron en la telefonía fueron los micrófonos de carbón. Estos micrófonos son ruidosos, tienen una respuesta limitada en frecuencia y altos niveles de distorsión, pero los requerimientos impuestos por el teléfono eran básicos, ya que solo era necesaria que la voz fuera entendible[3].

[3] (Eargle 1996)

El principio de funcionamiento de estos primeros micrófonos es sencillo, ya que se basa en los cambios que presenta la resistencia eléctrica propia de los miles de gránulos de carbón presentes en el micrófono, los cuales son comprimidos por el diafragma (que se mueve por acción de las ondas mecánicas propias del sonido), esto hace que la zona de contacto de las partículas de carbón aumente y la resistencia eléctrica disminuye.

Finalmente una fuente de alimentación de corriente directa es conectada al micrófono y a un transformador que se encarga de acoplar las impedancias, esta conexión de realiza en serie. Con esta conexión los cambios del valor de resistencia hacen que la corriente que circula por circuito varié en amplitud y resulte una forma de onda similar a la forma de onda acústica que golpea el diafragma[4].

El principio de operación puede ser observado en la figura 1.2.

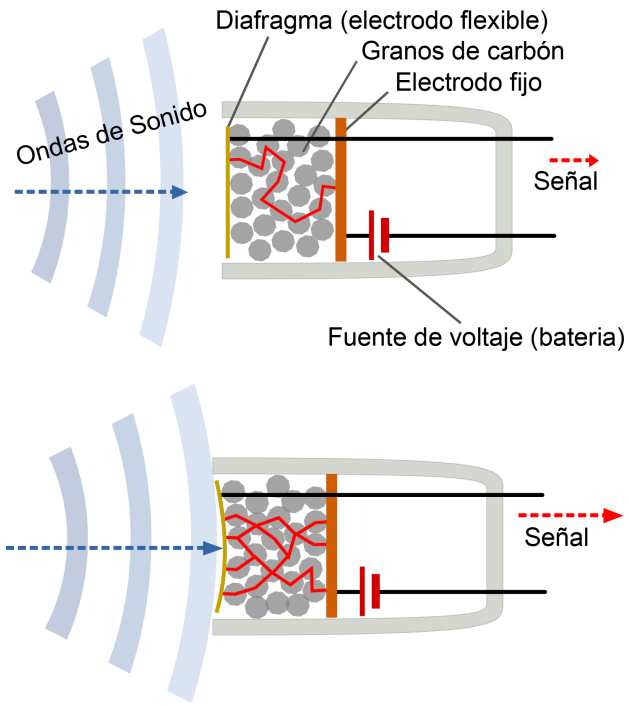

Figura 1.2: Esquema de funcionamiento de un micrófono de carbón.

Al empezar las transmisiones de radio comerciales, a inicios del siglo pasado, fue solicitada una calidad más alta y un desempeño mejorado de los micrófonos, ya que los programas de música y entretenimiento hablado necesitaban una transducción más precisa.

[4] (Ballaou 1991)

Para cubrir esta necesidad fue creado el micrófono de capacitor o condensador, y el principio de inducción electromagnética fue aplicado para crear los micrófonos de cinta y de bobina móvil.

Los micrófonos de bobina móvil funcionan con un diafragma móvil que vibra con las ondas mecánicas del sonido. Adjunto al diafragma se encuentra una pequeña bobina de alambre que se mueve hacia delante y atrás con el movimiento del diafragma. Esta bobina está inmersa en un campo magnético, lo que genera una diferencia de potencial en los extremos de la bobina al existir movimiento de esta dentro del campo magnético.

El voltaje producido por este micrófono se calcula por medio de la ecuación (1.1).

$$E = Blv \qquad (1.1)$$

Donde E es la diferencia de potencial en los extremos de la bobina medido en voltios, B es la densidad de flujo magnético en el espacio donde está contenida la bobina medido en teslas, *l* es la longitud de la bobina en metros, y *v* es la velocidad de la bobina a través del campo magnético medida en metros por segundo.

Debido a las bajas velocidades que desarrolla la bobina y a los tamaños reducidos en las bobinas (el cual es raro que llegue a los 2 cm de diámetro). La solución dada por los diseñadores de micrófonos para incrementar la salida es tener imanes grandes que adecuen la señal. De todas formas lo niveles de salida son de varios milivoltios por mayores que sean las presiones sobre el diafragma.

La ventaja de estos micrófonos es una respuesta en frecuencia plana en un ancho de banda importante. Lamentablemente sufre de problemas de resonancia y *damping*[5].

El micrófono de cinta funciona de manera similar al micrófono de bobina móvil, ya que la salida de voltaje depende del movimiento de un conductor (en este caso la cinta) a través de un campo magnético. La diferencia radica en la construcción física que se puede apreciar en la figura 1.3 y en la función acústica del micrófono.

[5] El *damping* es un decremento en el nivel de las oscilaciones del diafragma como resultado de energía perdida al ser convertida en calor por la fricción y otras fuerzas resistivas.

Para que funcione el micrófono es necesario que los dos imanes ubicados a los lados de la cinta metálica tengan la misma orientación, de tal forma que se genere un campo magnético intenso en la zona donde la cinta vibra.

Las ondas sonoras que tienen la capacidad de mover la cinta pueden llegar por ambos extremos del micrófono, mientras que por los lados es imposible que se produzca efecto alguno por la presencia de los imanes que tapan la cinta. Por este motivo el micrófono maneja una respuesta acústica "en forma de 8", por lo que se dice que tiene un patrón direccional o polar. Éste patrón puede observase en la figura 1.4 y se puede modelar en coordenadas polares como se muestra en la ecuación (1.2), por lo que se conoce como un patrón cosenoidal.

$$\rho = \cos \theta \qquad (1.2)$$

La respuesta en frecuencia del micrófono se puede mantener estable en un gran ancho de banda, donde la frecuencia inferior está dada por la frecuencia de resonancia fundamental y la frecuencia superior está dada por las dimensiones de la estructura magnética. En la mayoría de los diseños tenemos respuestas buenas hasta los 15 o 20 kHz.

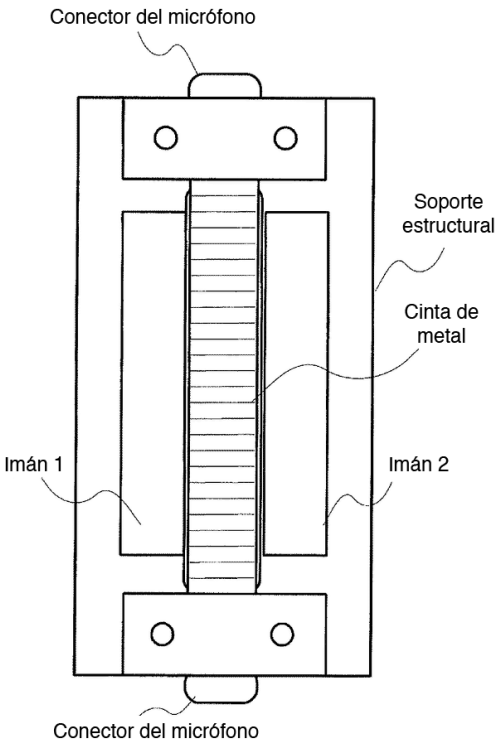

Figura 1.3: Configuración de un micrófono de cinta.

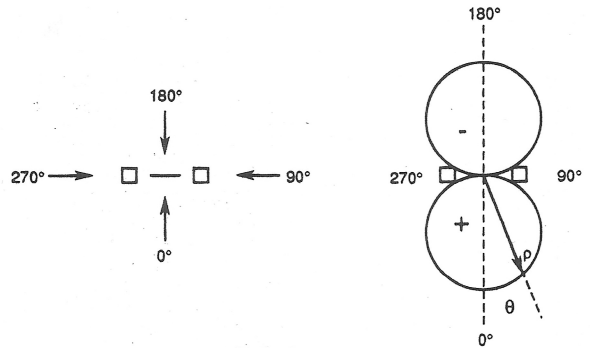

Figura 1.4: Característica direccional del micrófono de cinta.

Una ventaja importante de estos micrófonos es la inversión de la polaridad o cambio de fase (depende del sonido) que se obtiene de los sonidos que llegan en la dirección de 0º y 180º.

Los niveles de voltaje a la salida de la señal son bajos, por lo que se requiere un sistema de amplificación de bajo ruido para tener una salida aceptable. Al captar volúmenes altos (de instrumentos ruidosos) la salida pequeña deja de ser un problema.

Los micrófonos de capacitor (o condensador) son mecánicamente los más simples, ya que su única parte móvil es un diafragma extremadamente ligero. El cual actúa como una placa de capacitor.

Como sabemos los capacitores almacenan energía en forma de carga eléctrica entre dos placas y su comportamiento se rige por la ecuación (1.3).

$$Q = CE \tag{1.3}$$

Donde Q es la carga en coulombs sobre las placas del condensador, C es la capacitancia en faradios, y E es la diferencia de potencial existente entre las placas del capacitor.

La relación existente entre la carga y el voltaje es directa, lo que quiere decir que al tener una carga fija, una reducción en la capacitancia al separar las placas ocasiona un aumento en el voltaje entre ellas.

Este efecto es usado en el micrófono, donde el diafragma se mueve como función de la presión presentada por las ondas sonoras que lo alcanzan, este movimiento ocasiona que la capacitancia varíe.

Para la correcta utilización del micrófono es necesario adaptar un circuito el cual se conecta como se muestra en la figura 1.5, donde la resistencia R es grande y provee una carga constante, lo que hace que el voltaje en las placas cambie de acuerdo a la ecuación (1.4).

$$\Delta E = Q/\Delta C \qquad (1.4)$$

Donde ΔC representa el cambio en la capacitancia y ΔE las variaciones de voltaje entre las placas. La señal producida puede ser amplificada fácilmente con un amplificador que puede estar incluido dentro del mismo micrófono.

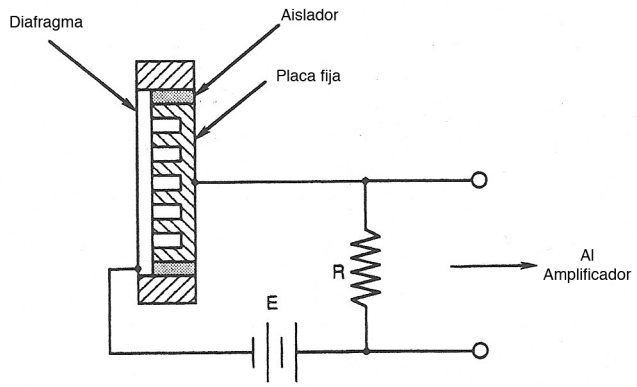

Figura 1.5: Conexión del micrófono de capacitor y sus componentes.

Este diafragma del micrófono se construye de plástico con una pequeña capa metalizada, de esta manera se construye un diafragma ligero y conductivo.

El voltaje suministrado a las placas del capacitor es de 48 voltios y algunas consolas de mezcla incluyen la opción para suministrarlo. Este voltaje también se llama voltaje de polarización.

Es curioso que los micrófonos no se convirtieron en una parte importante de los sistemas de grabación hasta 50 años después de su utilización en los teléfonos, ya que el proceso de grabación era 100% acústico-mecánico, lo que dejaba por fuera a los transductores.
La utilización de los micrófonos comenzó en a mediados de 1925 con la introducción de un grabador eléctrico por la empresa Western Electric.

Al principio de las grabaciones, los micrófonos creados para la radio (de bobina móvil y de cinta) eran útiles y predominaban en el negocio hasta la mitad de la década de 1940.

En este tiempo el mayor desarrollo de micrófonos corrió a cargo de la Western Electric y RCA, principalmente para la transmisión de contenidos por radio y el sonido para las películas.

Los micrófonos de capacitor eran utilizados solamente para medición y calibración de dispositivos dentro de los Estados Unidos, mientras que en Europa tuvieron un gran auge en la década de 1930 y fue después de la segunda guerra mundial que realizaron un gran impacto en las grabaciones estadounidenses.

La introducción de mejores técnicas de grabación, como la grabación en cinta y en LP, a finales de la década de 1940, necesitaban micrófonos con mejor respuesta en frecuencia y los micrófonos de capacitor tuvieron una gran oportunidad para brillar dentro de los estudios de grabación.

Durante los 50's y 60's se vio la entrada al mercado de diferentes micrófonos dinámicos con características de rendimiento fabulosas, lo que ocasionó que los estudios tengan micrófonos tanto dinámicos como capacitivos hoy en día.

Fue hasta el año de 1962 cuando se creó el micrófono Electret, el cual es un micrófono de capacitor con la diferencia de no necesitar una fuente externa de alimentación.
Los materiales utilizados se llaman Electret y tienen la propiedad de mantener una carga eléctrica fija.

Aunque los materiales con características Electret se conocían desde los años 20 y se plantearon para la utilización en micrófonos capacitivos, no fue hasta el año de 1962 que se logró realizar, cuando G.M. Sessler y J.E. West de Bell Laboratories propusieron el diseño de un micrófono utilizando una película delgada y metalizada de teflón. El diseño funcionó y hoy en día es el modelo más utilizado en la fabricación de micrófonos por sus cualidades de alta capacitancia, alta sensibilidad, una buena respuesta en frecuencia, baja distorsión e inmunidad para campos eléctricos y magnéticos externos, así como para vibraciones[6].

El diseño más simple de micrófono Electret es el que se observa en la figura 1.6.

[6] (Lucent Technologies 2000)

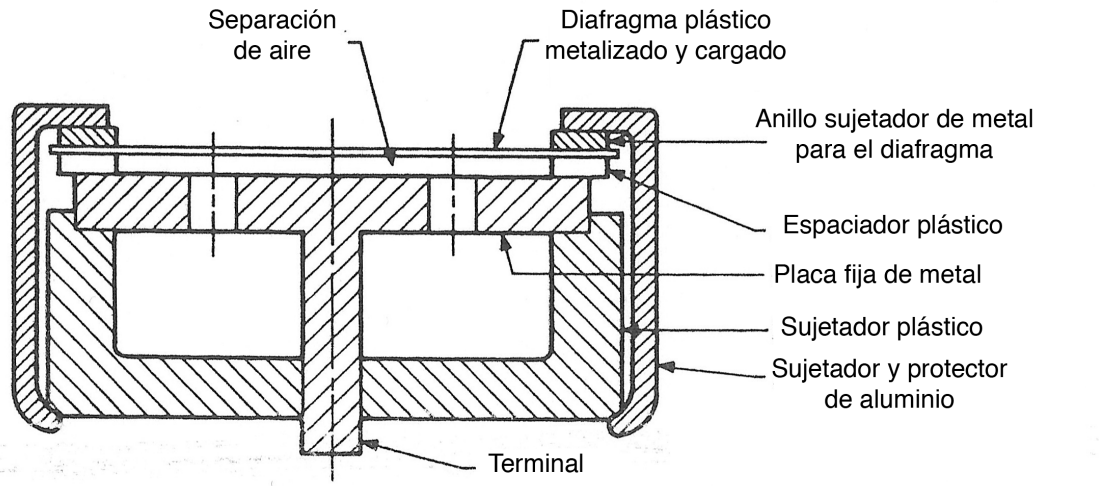

Figura 1.6: Estructura interna de un micrófono Electret sencillo.

Con el fin de seleccionar la hoja Electret se realiza un balance entre sus características eléctricas y las características mecánicas del diafragma, ya que se requiere que sea flexible para ser susceptible a las vibraciones propias de los sonidos.

Materiales hechos de polímeros son buenos para este fin, por lo que actualmente se utilizan materiales como polyacrylonitrile, policarbonato, y algunas resinas con flúor.

Existen varios métodos para fabricar el diafragma, pero estos métodos son conservados como secreto por los fabricantes de micrófonos.

Generalmente el diafragma es recubierto por un lado con un material conductor, como aluminio, oro o níquel; el grosor de esta cubierta es de 50 nanómetros. Una vez recubierto el material se calienta y es cargado con un potencial eléctrico bastante alto, esto hace que al enfriarse la carga se mantenga en su lugar.

Un Electret bien diseñado tiene la capacidad para mantener la carga suficiente para considerar una sensibilidad sin cambios por diez años y una pérdida de 3 dB en un lapso de tiempo de 30 a 100 años.

Las cubiertas plásticas metalizadas no soportan la suficiente tensión para resistir frecuencias de resonancia altas, por lo cual su respuesta en frecuencia cae rápidamente. Para poder contrarrestar este efecto, algunos fabricantes unen el diafragma en varios puntos al sujetador de

plástico para reducir la tensión de las uniones y alcanzar de esta manera frecuencias de operación más altas sin romper la cubierta.[7]

1.3 MEZCLA DE CANALES

A la hora de grabar las pistas de audio en los sistemas mecánicos se producían ciertos efectos indeseados, debido a los pobres métodos de grabación, ya que solo se podía grabar de forma acústica-mecánica y directa, los problemas relacionados variaban, desde problemas con las respuestas en frecuencias de algunas bandas (las bandas medias pueden ser grabadas a diferencia de bandas bajas y altas, donde el sonido no es grabado); hasta algunos problemas con instrumentos no funcionales para esta tecnología, como es el caso del violín.

Cuando se introdujo al mercado la grabación del sonido por medio eléctrico, después de la invención del micrófono y la aplicación de cierta electrónica básica, se pudo crear elementos para trabajar con las señales de audio, como el caso de la amplificación y las consolas de mezcla para modificar el volumen de cada pista de sonido.

La electrónica del momento permitía, por medio del "Audion" un tríodo basado en un tubo de vacío (bulbo), amplificar una señal eléctrica débil, este elemento también se aplicó para las líneas telefónicas de larga distancia y fue la base de los sistemas electrónicos de audio hasta la invención e implementación del transistor[8].

La primera mezcladora y grabadora del mercado fue marca Ampex, empresa creada a finales de la segunda guerra mundial con la finalidad de venderle a la milicia motores pequeños y generadores. Al acabar la guerra, la empresa se entera de una máquina grabadora en cinta magnética alemana, la Magnetophon. Después de ver la demostración de la máquina, la directiva de la empresa tomó la decisión de cambiar el rumbo y dedicarse a la grabación en cinta, diseñando y manufacturando grabadores en cinta con calidad profesional. Al poco tiempo la empresa se da cuenta de la necesidad de agregar un mezclador de sonido a sus productos y así logra colocarse en poco tiempo como empresa líder en grabación en cinta dentro del mercado de televisión, radio, industrias de

[7] (Benson 1988)
[8] (Coleman 2003)

grabación e inclusive dentro de la milicia, para la cual trabajaba en un inicio[9].

A finales de los años 50's, aparecieron mezcladoras pequeñas para radio estaciones, estas "pequeñas" mezcladoras tenían capacidad para manejar hasta ocho canales independientes, en este caso no se contaban con elementos para manipular la señal, consistía solamente en controles de volumen independientes para cada canal. Dentro de estas consolas encontramos la RCA 76-B5, la cual fue usada ampliamente por radio estaciones.

Figura 1.7: Consola de mezcla RCA 76-B5.

Las últimas consolas cuyo funcionamiento se basa en bulbos ya incluían controles para adecuar la señal y contrarrestar efectos de respuesta en frecuencia no planas, es decir, éstas ya incluían un pequeño ecualizador; así como paneles desmontables independientes por cada canal, lo que facilitaba su mantenimiento y remplazo de piezas, estas consolas ya incluían componentes para poder mezclar en vivo y ser utilizadas para espectáculos.

Dentro de esta categoría tenemos la consola de mezcla fabricada por Telefunken y Siemens, con referencia 18 – 2 custom, la cual tenía la capacidad de mezclar hasta 16 canales sin necesidad de recurrir a cintas magnéticas.

[9] (Ampex Corporation 2011)

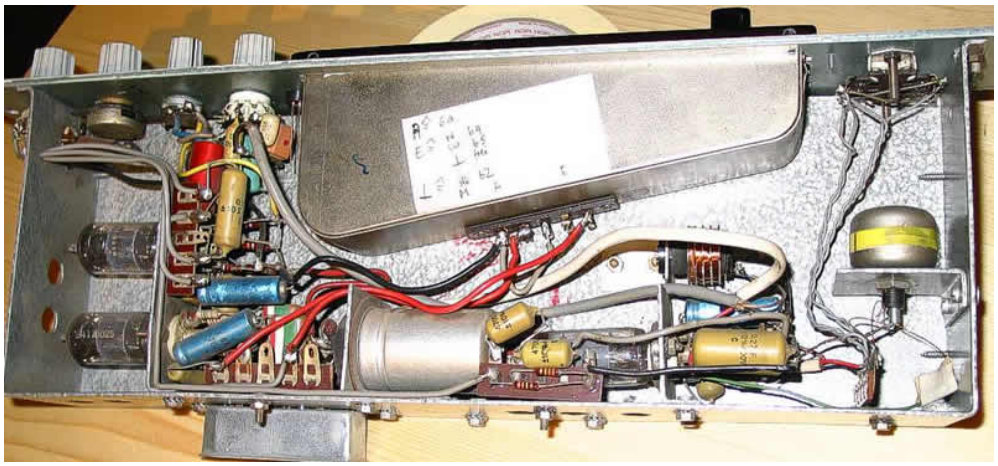

Figura 1.8: Detalle del control de un canal de la consola Telefunken/Siemens 18-2 Custom con electrónica de bulbos.

El desarrollo de consolas mezcladoras se disparó vertiginosamente desde la creación del transistor, ya que la electrónica sufrió un avance gigantesco con este componente y las consolas de audio integraron todos los elementos tecnológicos disponibles para aumentar el número de canales y la fidelidad del sonido.

De esta manera para los años 70's salieron consolas capaces de manejar hasta 32 canales de audio de forma independiente, así como con varios buses internos para poder manejar varias sumas de señales.

La electrónica de estado sólido permitió tener consolas de mezclado más pequeñas y fue éste el momento en donde las consolas salieron de los estudios de grabación y estaciones de audio para poder ofrecer servicios en conciertos y espectáculos.

El siguiente gran paso de las consolas de audio, fue el procesamiento digital de la señal, con los avances de la electrónica y procesadores cada vez más rápidos, fue cuestión de tiempo para que apareciera un procesador digital de señales al precio necesario para que el producto tenga un éxito comercial y con el poder de procesamiento necesario para hacer el trabajo de mezcla y ecualización de canales. De esta manera, en el año de 1982 salió al mercado la primera consola de audio digital, la Neve DSP1. Esta consola fue descrita como la primera consola multipropósito, ya que gracias a estar programada en base a un procesador digital de señales (DSP por sus siglas en inglés), se puede cambiar la configuración y reprogramarla para que preste diferentes servicios, como la grabación de música proveniente de varios canales,

para la post producción o simplemente para la transmisión en vivo de la señal[10].

Figura 1.9: Primera consola de mezcla digital, la NEVE DSP1.

Las consolas para los estudios siguieron evolucionando y mezclándose cada vez más con las computadoras. Es así como se introdujeron avances computacionales importantes, como la grabación en disco duro dentro de la computadora para su posterior exportación a casete y finalmente a mediados de los 90's, la computadora Apple PowerMac introdujo la suficiente potencia de procesamiento para la grabación de audio profesional en el hardware de una computadora común.

Actualmente las consolas profesionales de sonido para estudios son todas digitales, siendo el corazón de la mayoría un DSP, estas consolas tienen un precio bastante elevado, siendo en promedio $8000 dólares por una consola de 16 canales.

Como sabemos, las señales que se pueden obtener de los transductores son señales analógicas, las cuales pueden tener cualquier valor en un intervalo dado. Muchas de las operaciones realizadas en la señal son de una naturaleza analógica, de tal manera que amplificar y mezclar los canales se han realizado con circuitos especializados para ello.

[10] (IBS: The Organisation for Sound Professionals 2010)

En las últimas tres décadas se ha vivido una transformación en la industria, al orientarse hacia el procesamiento de la señal en su forma digital y las ventajas obtenidas en muchas de las aplicaciones son suficientes para dejar de lado el tiempo de latencia agregado y el costo de los componentes.

Esta transformación es mantenida por la reducción de los costos en los componentes, por el aumento y mejora del empaquetamiento de los componentes y los avances en la construcción de los circuitos integrados, lo que permite manejar circuitos con más velocidad y poder de cálculo[11].

Dentro de la evolución tecnológica dada por la aparición del transistor, aparecen circuitos integrados, los cuales tienen una capacidad de cálculo elevada y se especializan en microprocesadores, los cuales se emplean para realizar sistemas completos en base a ellos.

Históricamente los microcontroladores surgieron después de los microprocesadores y siguieron caminos diferentes, dado que un microcontrolador se puede ver como una computadora de tamaño miniatura en un solo chip.

Los microprocesadores se comenzaron a usar en las computadoras personales y estaciones de trabajo, debido a la demanda de alto poder de cómputo y la habilidad para manejar grandes volúmenes de datos e instrucciones, mientras que los microcontroladores se aplican a un gran número de aplicaciones donde se solicita un número fijo de procesos y un bajo costo económico[12].

[11] (Hamdy 2009)
[12] (Rossing 2007)

CAPÍTULO 2: DESARROLLO DEL SISTEMA DE CAPTURA

2.1 REQUERIMIENTOS

Una de las ideas centrales del proyecto, es remover la mayor cantidad de partes donde la señal es analógica dentro del sistema, esto con el fin de lograr una mayor fidelidad y menores pérdidas.

Al hacer esto obtenemos una gran ventaja, ya que tenemos un hardware fácil de reconfigurar para agregar más características al sistema y podemos realizar un procesamiento digital de la información dentro de una consola emulada en la computadora, lo que permite agregar características casi imposibles de lograr con un procesamiento analógico, como mantener un fase lineal y un retraso de grupo constante dentro de los filtros.

Problemas clásicos de los circuitos analógicos, como el acoplamiento de impedancias y los rangos de trabajo de los componentes, no son importantes al trabajar con un sistema digital[13].

El procesador que se escoja va a estar dentro del sistema de captura y va estar especializado en el empaquetamiento de la información y formará parte de un sistema computacional más grande. El procesador va a tener conexiones con hardware externo como la memoria, el convertidor analógico digital, una fuente de alimentación y una salida para la red.

En la mayoría de los sistemas, el procesador y las interfaces deben de operar en tiempo real, lo que significa que las señales de entrada tienen poco tiempo para ser procesadas. Al no cumplir con los requerimientos de tiempo se obtiene una salida ruidosa en aplicaciones de audio y video[14].

Necesitamos que el sistema de captura sea capaz de realizar la digitalización del sonido y manejar su envío por la red de área local, para esto es necesario que el micrófono tenga un elemento que se encargue de la adquisición de datos y su posterior empaquetamiento y envío, para lograr esto los componentes de los cuales requiere el micrófono se pueden ver en la figura 2.1.

[13] (Hamdy 2009)
[14] (Gan and Kuo 2007)

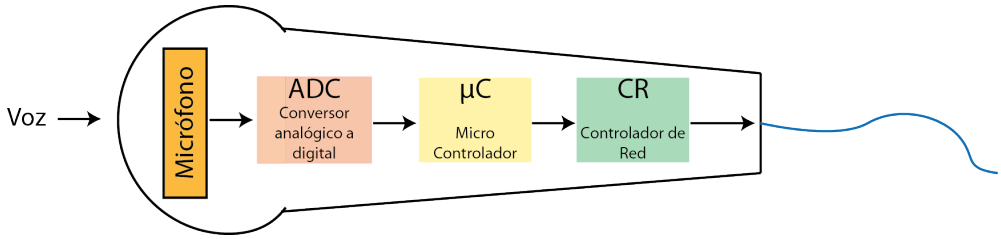

Figura 2.1: Componentes básicos del sistema.

Lo primero por hacer es realizar la selección del elemento central del micrófono, para lo cual se comienza estudiando la posibilidad de manejar un DSP o un microcontrolador (µC).

Para poder determinar qué tipo de dispositivo necesitamos, es necesario saber qué características necesitamos, de esta manera podemos establecer cuáles son los requerimientos mínimos necesarios para el dispositivo.

Para poder trabajar con el audio sin agregar una importante suma de tiempo de latencia, es necesario que la información se envíe sin comprimir, esto hace que el control del micrófono solo necesite empaquetar la información procedente del transductor.

Además necesitamos una conexión Ethernet para poder enviar los paquetes, ya que la conexión con la estación central se realiza a través de una red de área local. De esta manera es necesario que el controlador pueda manejar la conexión de área local, esto quiere decir que se necesita gestionar la asignación de una dirección IP y almacenar la dirección IP de destino para los paquetes.

Para definir el procesador que necesitamos en nuestro es sistema es necesario estudiar las características de los dos sistemas planteados.

Los procesadores y controladores están optimizados para cómputos de propósito general, lamentablemente sus capacidades están por debajo de las características de alta velocidad necesarias para el procesamiento de señales en tiempo real. Para poder sobrepasar este problema, es necesaria la creación de un hardware especializado en el procesamiento de señales, donde las operaciones de suma y multiplicación son básicas. Es así como se crean los DSP (*Digital Signal Processor*), los cuales pueden realizar las operaciones básicas con gran velocidad. Otras operaciones, como las lógicas y las operaciones de memoria tienen un desempeño normal, ya que la necesidad de ellas para el procesamiento de datos es limitado.

Una opción disponible para el procesador es utilizar un procesador alambrado únicamente para cierta aplicación (ASIC), estos son procesadores con capacidades parecidas a un DSP, la única desventaja es el alto costo de diseño y la imposibilidad de agregar nuevas funciones a medida que avanza la tecnología.

Dentro de las ultimas clases de procesadores para señales están los procesadores llamados *micro signal architecture* (MSA), los cuales fueron diseñados por Intel y Analog Devices Inc. para satisfacer las demandas computacionales de las aplicaciones de audio y video digitales. Estos procesadores incluyen las funcionalidades de un microprocesador y un DSP en un solo núcleo[15].

Los DSP's tienen componentes que son innecesarios dentro de nuestro sistema, como es la unidad de multiplicación, la cual permite hacer operaciones de multiplicación en un solo ciclo de reloj, además de tener una arquitectura Harvard modificada, esto quiere decir que se tienen buses independientes de memoria para los datos y el programa y así ganar velocidad.

Los microcontroladores tienen una arquitectura Von Neuman, en donde solo se tiene un bus que es compartido por los datos y el programa, lo cual reduce su eficiencia por ciclo de reloj si es comparado con un DSP[16].

Los requerimientos de procesamiento de nuestro micrófono son bastante pequeños, por lo cual es recomendable usar un microcontrolador para llevar a cabo las tareas requeridas.

2.1.1 EL MICROCONTROLADOR

Una computadora genérica tiene tres bloques principales: la unidad de procesamiento central, la memoria y el sistema de entradas y salidas. Estos bloques se interconectan por medio de buses, de los cuales existen tres clases: los buses de dirección (transportan datos de la memoria o los datos de entrada y salida), los buses de datos (encargados de la información o instrucciones) y por último los buses de control (manejan señales de control).

[15] (Gan and Kuo 2007)

[16] (Texas Instruments Incorporated 2006)

De esta manera la CPU es el cerebro de la computadora y está a disposición del programa almacenado en la memoria de la computadora. La CPU tiene los circuitos necesarios para poder realizar operaciones lógicas con entradas binarias, este circuito se llama ALU (*Arithmetic and Logic Unit*).

En una pequeña computadora el papel de CPU es desarrollado por un microprocesador el cual se encarga de las operaciones necesarias para correr el programa.

Un microcontrolador se puede ver como una microcomputadora construida en un solo circuito integrado[17].

Se escoge un microcontrolador por las siguientes características:

- Recursos de entradas y salidas: A diferencia de un microprocesador que se basa en la habilidad de computo o un DSP que se centra en el procesamiento de señales, el microcontrolador se centra en la habilidad para las entradas y salidas, de esta manera puede manejar señales analógicas, interruptores, líneas de entrada y salida, etc.
- Optimización de espacio: Se utilizan para poder tener el mayor número de funciones en el menor espacio posible, manteniendo precios bajos.
- Hay gran variedad de microcontroladores con diferencias de entradas o tamaño de la memoria, permitiendo elegir al diseñador el dispositivo más apropiado para la finalidad requerida.
- Protección contra fallas: Los microcontroladores tienen un temporizador que monitorea el sistema para asegurarse que el programa corre correctamente.
- Bajo consumo eléctrico, lo que permite ser usado con baterías. Además tiene un estado de "sleeping" que reduce al mínimo el consumo cuando no se usa el microcontrolador.
- Protección anticopia del programa: La información en la memoria tiene un mecanismo de protección para evitar ser copiado.

Una vez tomada la decisión de utilizar un microcontrolador como el elemento central del micrófono, comenzamos con la búsqueda del dispositivo que cubre nuestras necesidades.

[17] (Valdés Pérez and Pallàs-Areny 2009)

Dentro de las plataformas de desarrollo con las cuales podemos trabajar, tenemos los dispositivos Arduino, ya que nos permiten una programación sencilla y nos dan opciones de hardware interesantes para la implementación de servicios posteriores.

De esta manera escogimos una tableta de desarrollo Arduino UNO, el cual tiene un microcontrolador ATMEL ATmega328, el cual tiene las siguientes características principales:

- Bajo consumo energético.
- Capacidad de alcanzar 20 MIPS de salida.
- Memoria FLASH y EEPROM incluida para almacenar los programas.
- Memoria RAM para ejecutar los programas.
- ADC con resolución de 10 bits y una tasa máxima de 76.9 kSPS.

Se escoge este sistema de tarjeta de desarrollo y microcontrolador por la facilidad existente de tener una conexión de datos por medio de un controlador Ethernet, ya que se puede adaptar una tarjeta encargada de esto con facilidad[18].

Al avanzar en el desarrollo del trabajo, se observó que los valores prometidos como características en la hoja de especificaciones no son alcanzados, esto se debe a que el microcontrolador está trabajando a 16Mhz y por tener un código no optimizado.

Para poder cumplir con el objetivo propuesto es necesario adquirir la voz con la mejor calidad posible, para lo cual es necesario una etapa de amplificación entre el micrófono y el convertidor analógico a digital incluido en el microcontrolador, ya que los valores de voltaje generado por el micrófono no son suficientes para poder ocupar completamente los valores donde el convertidor puede funcionar.

Para amplificar la señal se emplea el circuito mostrado en la figura 2.2.

[18] (Atmel Corporation 2009)

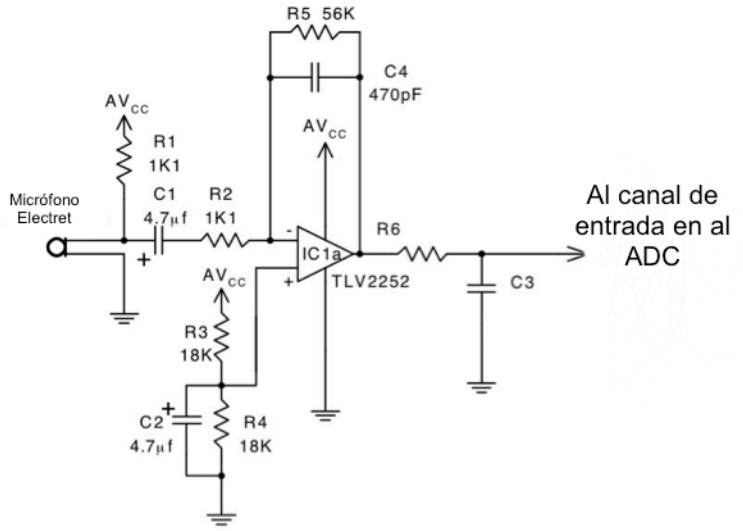

Figura 2.2: Circuito amplificador de señal analógica.

Este circuito también nos soluciona el problema existente por los valores positivos y negativos de salida que nos proporciona el micrófono, ya que al tener un divisor de voltaje en la entrada no inversora del amplificador operacional, obtenemos que la salida del mismo este centrada en el rango de voltajes comprendidos de cero hasta el voltaje de alimentación.

La ganancia que proporciona el amplificador se encuentra con la fórmula (2.1).

$$Ganancia = \frac{R_{entrada}}{R_{lazo\ cerrado}} = \frac{R_5}{R_2} = \frac{56K}{1.1K} = 50.909 \qquad (2.1)$$

De esta manera la ganancia resultante del circuito es de 50.909 veces. Con esto la señal adquiere niveles de voltajes ideales para ser introducida al ADC existente dentro del microcontrolador[19].

La ecuación (2.2) describe el comportamiento de la salida, incluyendo el voltaje agregado por el divisor de voltaje de la terminal no inversora.

$$V_{sal} = \frac{R_3}{R_3+R_4} + \frac{R_5}{R_2} V_{entrada} \qquad (2.2)$$

[19] (Texas Instruments 2001)

2.1.2 INTERFAZ DE RED

Para poder enviar la información a través de una red de área local, es necesaria la utilización de un controlador de red, el cual se conecta al micro controlador ATmega328 por medio de un bus SPI. En este caso el bus se encuentra en los pines 11, 12 y 13 del microcontrolador.

El controlador de red debe de tener la capacidad de manejar conexiones UDP por medio de una red de área local para el envío correcto de los paquetes. Así mismo es necesario tener la capacidad para manejar la dirección IP asignada al micrófono.

También es necesario que tenga la capacidad de introducir las cabeceras de manera automática en los paquetes que indiquen destino y origen de la información para que puedan ser transportados de manera segura en la red.

El controlador de red seleccionado es el WIZnet iEthernet W5100.
Se selecciona este controlador por dos razones principales, la primera son las características propias del mismo (desglosadas en el próximo párrafo) y la segunda es la opción de fácil conexión al microcontrolador principal ya que respeta el orden de conexión por un puerto SPI necesario para conectarse entre sí.

Dentro de las características del controlador de red destacan las siguientes:

- Soporte de hardware para los protocolos TCP/IP: TCP, UDP, ICMP, IPv4 ARP, IGMP. PPPoE y Ethernet.
- Soporta auto-negociación (full o half dúplex).
- Soporta cuatro sockets independientes.
- Memoria interna de 16 kbytes para los paquetes a transmitir y/o recibir.
- Salidas luminosas para indicar la transmisión, recepción, velocidad y conexión.

El diagrama de bloques del dispositivo se puede ver en la figura 2.3[20].

[20] (WIZnet Co., Inc. 2011)

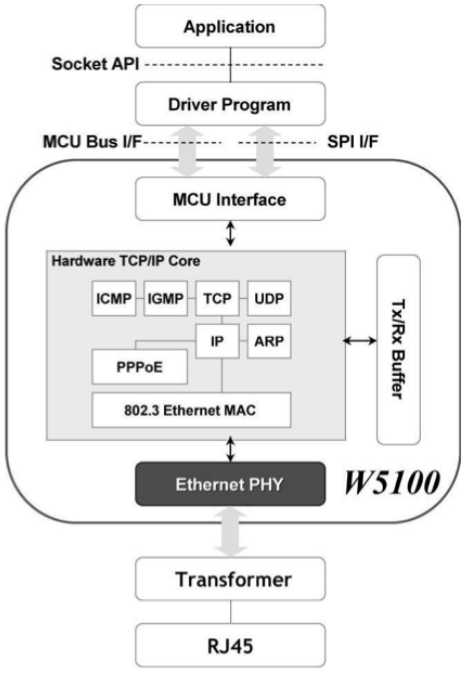

Figura 2.3: Diagrama de bloques del controlador de red[19].

Con la ayuda del controlador de red se pueden establecer las conexiones pertinentes para enviar la información proveniente del micrófono a través de la red.

2.2 IMPLEMENTACIÓN DEL SISTEMA DE CAPTURA

Una vez establecidas las necesidades buscadas en el sistema comenzamos con la búsqueda de los componentes necesarios para realizar la implementación, es decir, el microcontrolador (con una manera factible de programarlo), la conexión con el controlador de red y el diseño del circuito del amplificador de señal de micrófono para poder acoplarlo sin problemas al ADC del sistema.

2.2.1 EL MICROCONTROLADOR ATMEGA328

Como se mencionó con anterioridad, se seleccionó la plataforma de desarrollo Arduino UNO para poder trabajar con el microcontrolador ATMEL ATmega328 de una manera práctica, ya que en la tarjeta viene incluida la parte de alimentación eléctrica, la forma de cargar el programa por medio de un puerto USB y terminales disponibles para agregar los componentes faltantes del sistema.

En la figura 2.4 se puede observar la tarjeta de desarrollo con sus diferentes componentes.

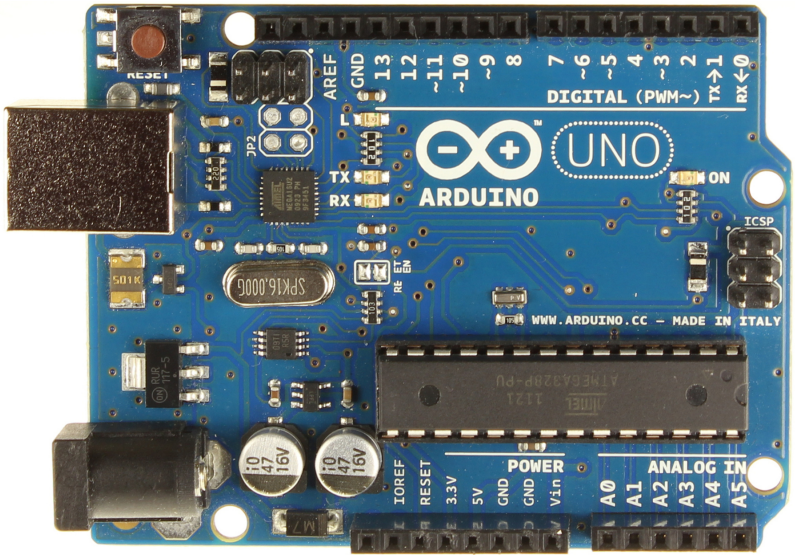

Figura 2.4: Controlador utilizado para la creación del sistema.

Se realizaron pruebas con el microcontrolador para poder establecer la capacidad de trabajo del ADC, el cual operó a 100 Hz debido a la velocidad de transmisión del puerto serial.

Con el fin de realizar las pruebas, se realizó la lectura de mediciones provenientes del canal A0 del microcontrolador, donde estaba conectado un potenciómetro con el fin de variar el voltaje medido. La información fue enviada a través del puerto serial disponible en el microcontrolador, en este caso la información es recuperada en el monitor serial de la computadora.
Los datos obtenidos, así como el monitor serial del programa se pueden observar en la figura 2.5.

Figura 2.5: Monitor serial con los datos recuperados del microcontrolador.

2.2.2 LA INTERFAZ DE RED

Para poder probar el controlador de red de manera sencilla, se selecciona un módulo compatible con el esquema de conexiones Arduino, por lo cual las conexiones entre el microcontrolador y la interfaz de red se hacen de manera automática al ensamblar las diferentes partes.

En este caso la tarjeta de desarrollo trae incluido el puerto RJ45 para el cable de red, indicadores de estado, el controlador WIZnet iEthernet W5100 y todas las conexiones de alimentación y datos compatibles con la tarjeta de desarrollo del microcontrolador.
La tarjeta se puede apreciar en la figura 2.6.

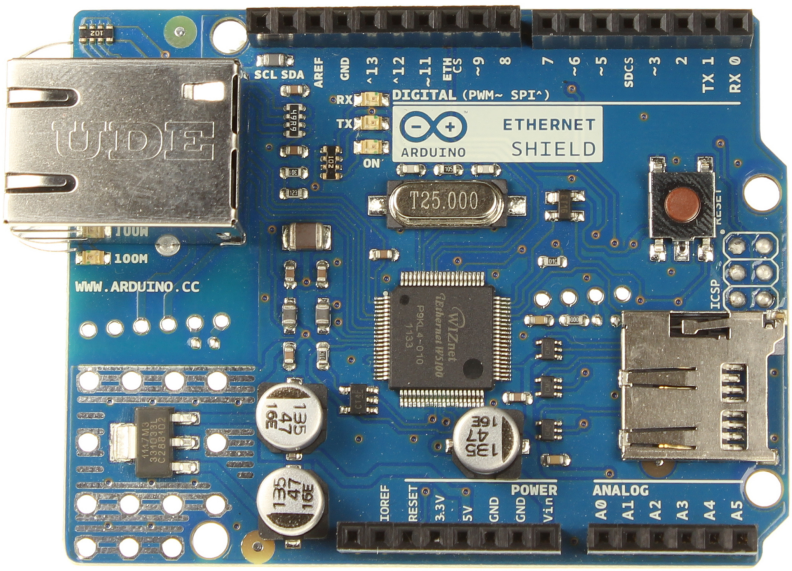

Figura 2.6: Controlador de red adaptable a la tarjeta de desarrollo Arduino.

Con el fin de probar el desempeño del controlador y su correcto funcionamiento al estar operado por el microcontrolador, se realizaron pruebas de envío de información (texto) por medio una conexión TCP para ser visualizadas en un explorador web cualquiera y posteriormente se modificó el programa para poder enviar la información obtenida del micrófono y ser visualizada en el mismo navegador.

Los datos obtenidos a través del explorador web se pueden apreciar en la figura 2.7. El rango de los datos obtenidos es distinto a los obtenidos en la figura 2.6 debido al ajuste del potenciómetro.

Cuando el sistema se pone en funcionamiento con la consola de sonido ejecutándose en la computadora, la información de diez muestras es empaquetada dentro del controlador y finalmente es enviada por la red de área local como un solo paquete UDP.

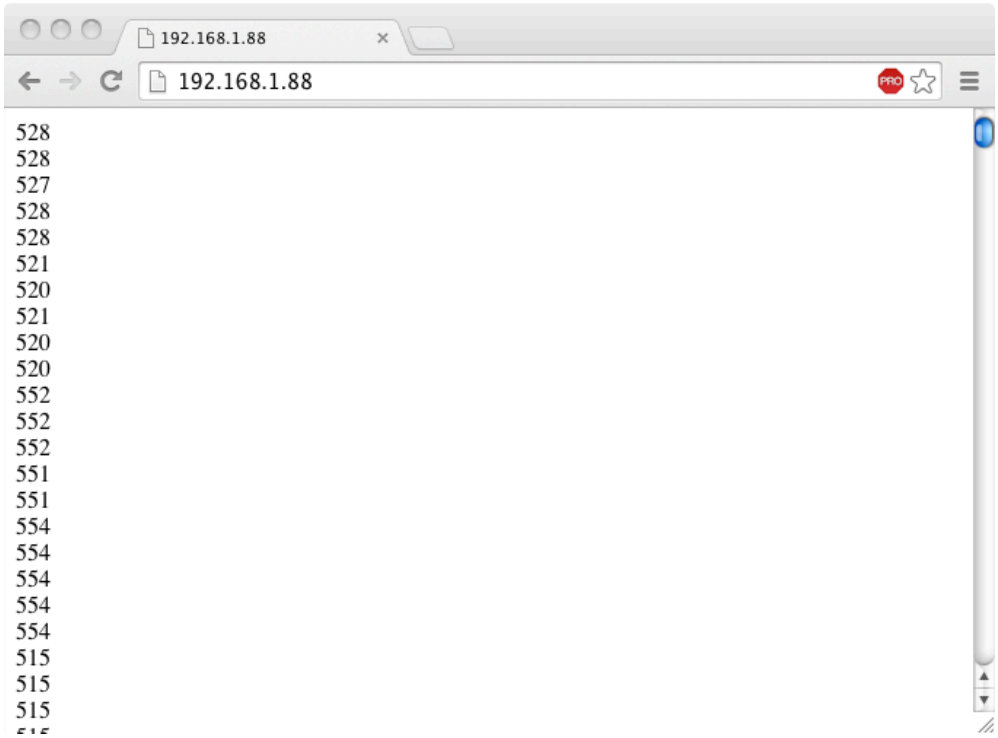

Figura 2.7: Obtención de datos por medio de un explorador web.

Durante la ejecución de estas pruebas las conexiones se establecían con una petición lazada por el navegador al intentar conectarse, este esquema de conexión fue modificado después para agilizar el envío de la información y se eliminó la conexión TCP a favor de una UDP que ahorra bits en el encabezado.

2.2.3 LA ADAPTACIÓN DE SEÑAL

Cuando se quiere introducir la señal proveniente del micrófono en el ADC del microcontrolador es necesaria la amplificación de la señal y adaptación de impedancias entre los componentes. Para la realización de estas actividades se construye el circuito mostrado en la figura 2.2, primero en un protoboard para poder realizar pruebas de respuesta en frecuencia y amplificación de los niveles de voltaje y finalmente en un circuito impreso.

Para poder utilizar el circuito impreso es necesaria la construcción del patrón a crear por medio de un programa de diseño, en este caso el programa escogido es "Eagle", el cual nos permite diseñar el modelo

esquemático y la realización de un esquema con las pistas, agujeros y terminales necesarias para que funcione a la perfección el sistema.

Primero se realiza el modelo esquemático del circuito a fabricar, en donde es necesario buscar los componentes deseados de la librería correspondiente, al incluir cada componente ya se incluyen sus características físicas. El diagrama correspondiente se puede observar en la figura 2.8.

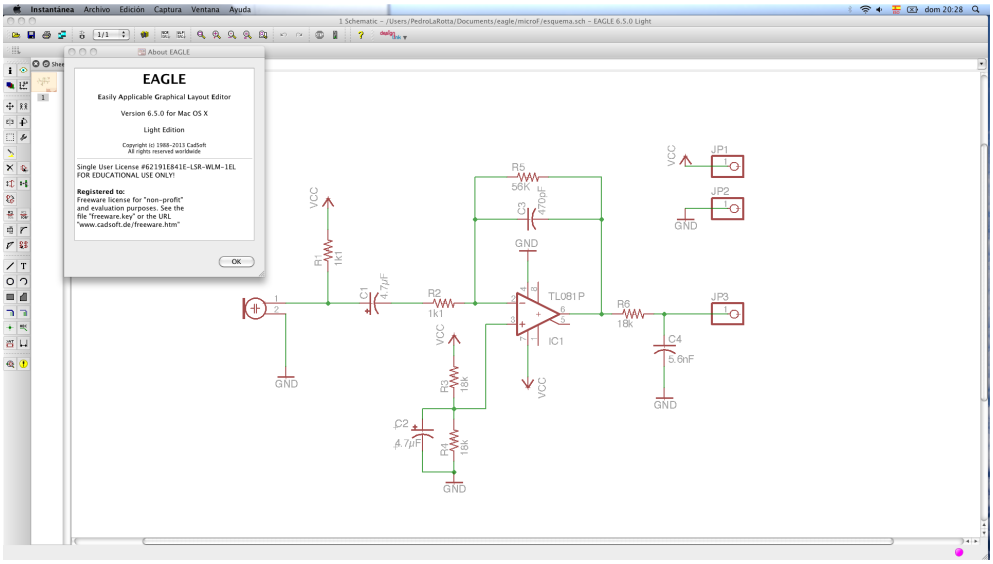

Figura 2.8: Diagrama de conexiones para el adaptador de señal.

Una vez realizado el esquema, se procede a realizar el diseño de la tarjeta. En este punto se toman en cuenta los diferentes caminos que pueden tomar las pistas para encontrar el trazado óptimo, evitando la unión de diferentes líneas.

Debido a un error en el diseño de la tarjeta se crearon un par de tarjetas imposibles de utilizar debido al grosor de las líneas de transmisión y a la escasa separación de los canales.
Estas tarjetas llevan la versión 1.0 estampadas en cobre.

Para poder corregir éste error fue necesario repetir el diseño de la tarjeta y se procedió a construir un par de tarjetas más, que ahora sí funcionan correctamente.

El estampado de las tarjetas las marca como versión 1.1. Ambos diseños tienen estampadas las iniciales LSPedro.

El diseño correcto realizado para la creación de las tarjetas se puede observar en la figura 2.9.

En este caso el tamaño de la tarjeta donde se estampa el diseño está limitado por las medidas de la tarjeta de desarrollo Arduino que estamos utilizando, ya que buscamos un micrófono pequeño y las medidas mínimas están dadas por las tarjetas de desarrollo.

Por este mismo motivo, los pines donde se conecta la alimentación de energía, el nivel de tierra y la salida amplificada están colocados específicamente para concordar con la posición de los pines de entrada y salida respectivos dentro de la tarjeta Arduino.

Una vez realizado el diseño del circuito, se procede a la creación del modelo físico, para lo cual es necesaria la impresión del diseño en papel couche. Se selecciona este papel por la facilidad existente de transferir el diseño a la tarjeta de baquelita con baño de cobre por medio de calor.

Después de realizar la transferencia, se procede a hacer un retoque de las pistas que sufrieron algún daño en el momento de la transferencia para poder asegurar un resultado correcto a la hora de retirar el cobre sobrante.

Para poder retirar el cobre se emplea una solución de cloruro férrico mezclada con agua, la cual es corrosiva y elimina completamente, mediante oxidación, el metal que está expuesto de la placa de cobre. Este proceso se llevó a cabo en un espacio ventilado para prevenir cualquier interacción no deseada.

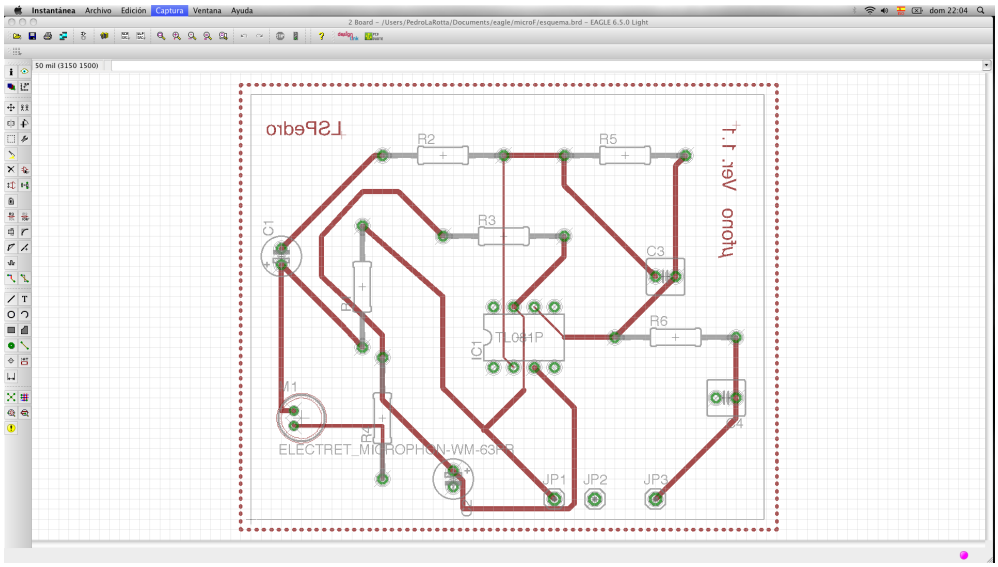

Figura 2.9: Diseño de la tarjeta para adaptar el micrófono al microcontrolador.

Después de la eliminación del cobre, las pistas y el texto introducido en el diseño quedan marcadas sobre la placa, pero protegidas por la tinta, por lo cual es necesario lijar la superficie hasta exponer el cobre que se encuentra debajo de la capa de tinta.

Una vez realizada la exposición del cobre, se procede a realizar los agujeros para la introducción de los componentes.

Este proceso es llevado a cabo con un taladro de mano (conocidos comúnmente como Dremel) y una broca de 1 mm de diámetro para asegurarnos de tener el agujero correcto para que pasen los componentes más anchos. En el caso de las resistencias el agujero es muy grande y es rellenado con soldadura posteriormente.

Ya con los agujeros en su lugar se procede al posicionamiento de piezas y a aplicar la soldadura para unir los componentes eléctricos con las vías y de ésta manera fijarlos mecánicamente a la tableta.

Una vez que se acaba el proceso de soldado, la tarjeta ya queda lista para su incorporación a las otras dos tarjetas y con ello para ser utilizado de manera exitosa por el sistema de audio.

En la figura 2.10 se puede observar el modelo final del adaptador de la señal utilizable por el usuario.

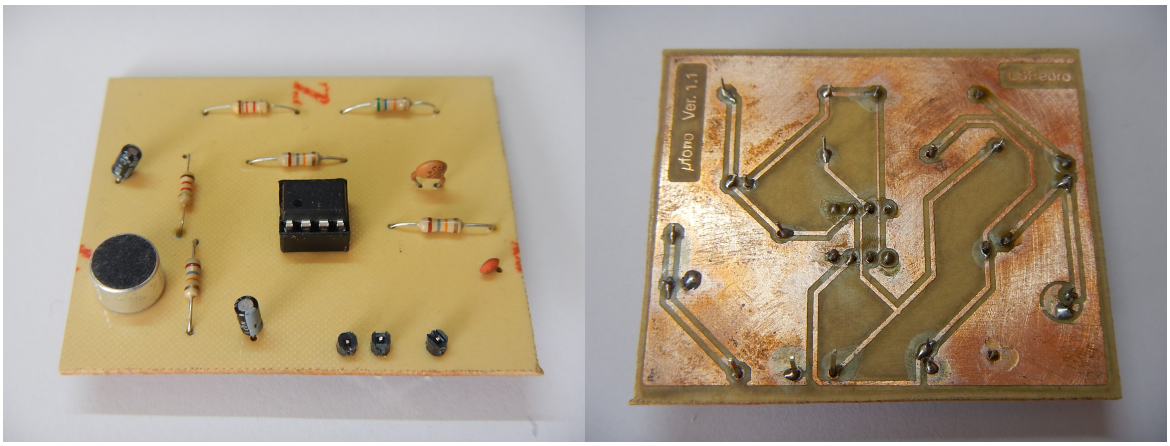

Figura 2.10: Presentación final de las tarjetas terminadas.

Una vez que la tarjeta con el micrófono y el amplificador fue realizada, se ejecutaron pruebas con ayuda de un osciloscopio para poder medir los valores de salida del circuito. Esto con el fin de comprobar que la señal amplificada se adecúa con el conversor analógico a digital contenido dentro del microcontrolador.

El ADC con el cual estamos trabajando debe tener una entrada máxima igual al nivel con el cual se alimenta el microcontrolador. La entrada mínima es igual al nivel de tierra registrado.

Con los parámetros con los cuales se va a trabajar, los valores deben de oscilar entre 0 y 5 voltios. Cuando se presenta el silencio, la salida del circuito debe de estar en la mitad, es decir, en 2.5 volts, de tal manera que al presentarse sonidos capturables por el sistema, los niveles de voltaje oscilen teniendo como media la mitad del rango dinámico del ADC.

Debido a un problema con la calidad del sonido capturado, estas pruebas se realizaron con dos amplificadores operacionales diferentes. El segundo amplificador reportó un mejor comportamiento en el audio y es por esa razón que se utilizó para el desarrollo del circuito.

La señal a la salida del circuito, utilizando el segundo amplificador se puede observar en la figura 2.11.

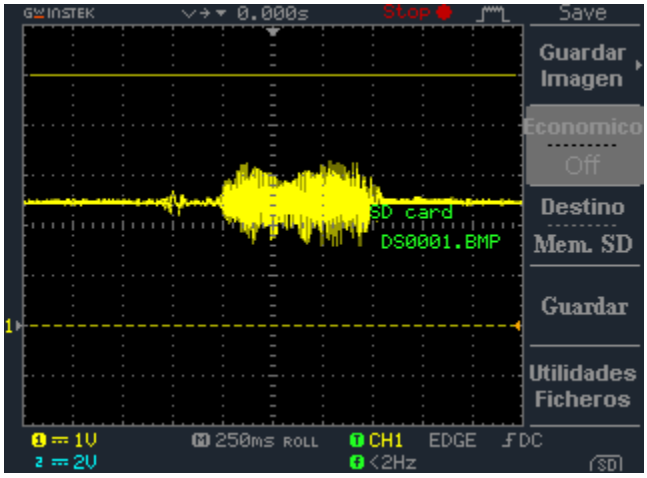

Figura 2.11: Salida analógica del circuito de amplificación.

Una vez que se tiene la salida del circuito amplificador probada en el laboratorio, es necesario conocer cómo va a ser su comportamiento al

estar conectada con el microcontrolador encargado de leer los valores de voltajes de salida. De esta manera se procede a trabajar con las dos tarjetas.

Para poder trabajar con el microcontrolador es necesario contar con el ambiente de desarrollo Arduino instalado en la computadora. Dentro de este ambiente se puede escribir el código a ejecutar, así como obtener ciertos parámetros de regreso provenientes del microcontrolador. Estos datos son obtenidos por medio del puerto serial que incorpora la tarjeta de desarrollo y del lado de la computadora es creado cuando se instala el software.

Con el fin de poder observar los datos enviados por la tarjeta mediante el puerto serial, es necesario emplear un monitor del puerto integrado en el ambiente de desarrollo, es en esta ventana donde se obtienen los valores de regreso escritos por el programa. La velocidad de transferencia del puerto debe de ser ajustada por el programa cargado en el microcontrolador y en la ventana del monitor de la computadora. Es importante que la velocidad sea la misma en los dos lugares, de otra manera se reciben caracteres aleatorios.

En la figura 2.12 se puede observar una imagen del entorno de desarrollo del microcontrolador.

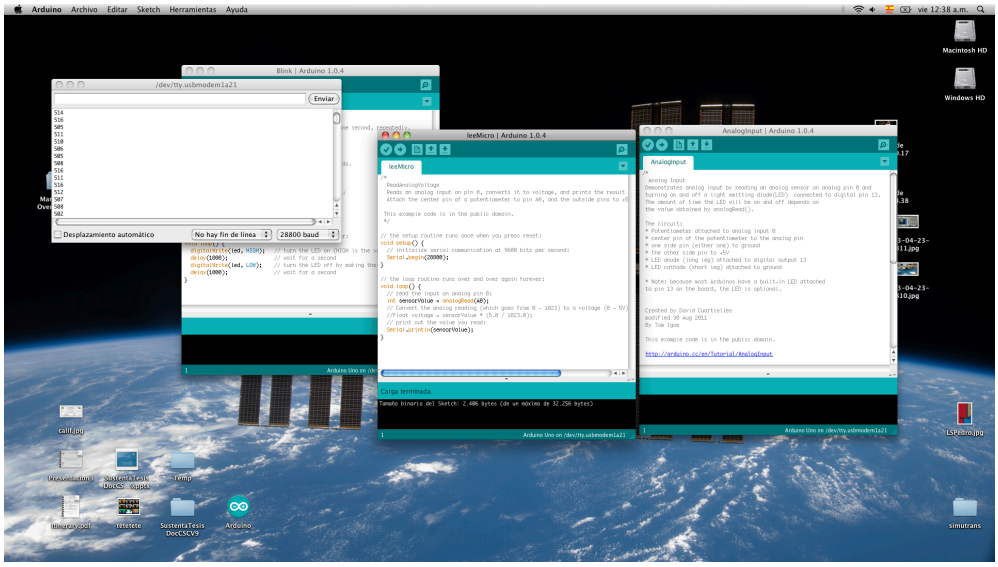

Figura 2.12: Entorno de desarrollo del microcontrolador.

Para poder obtener lecturas del micrófono es necesaria la conexión de las dos tarjetas, posteriormente la ejecución de un programa cargado en el

microcontrolador que se encarga de leer el valor del conversor analógico a digital y escribirlo en el puerto serial, y finalmente la apertura del monitor de puerto serial.

Para la alimentación del circuito (tanto el microcontrolador como el amplificador de señal) se utiliza el voltaje de 5V suministrado por el puerto USB. Posteriormente éste voltaje se puede cambiar o puede provenir de una fuente externa de alimentación.

La señal capturada por el micrófono y enviada por el puerto serial es recuperada del monitor y copiada en un archivo .csv, el cual puede ser introducido a Matlab con el fin de procesar y escuchar la información.

Una vez que se tiene la información en Matlab se procede a eliminar la componente de corriente directa de la señal, ésta componente es necesaria para que el conversor pueda capturar la parte positiva y negativa del voltaje del micrófono. Cuando la componente de corriente directa es eliminada, la señal se puede graficar y reproducir por medio del hardware propio de la computadora.

La forma de onda que adopta la señal después del poco procesamiento realizado en Matlab se puede apreciar en la figura 2.13.

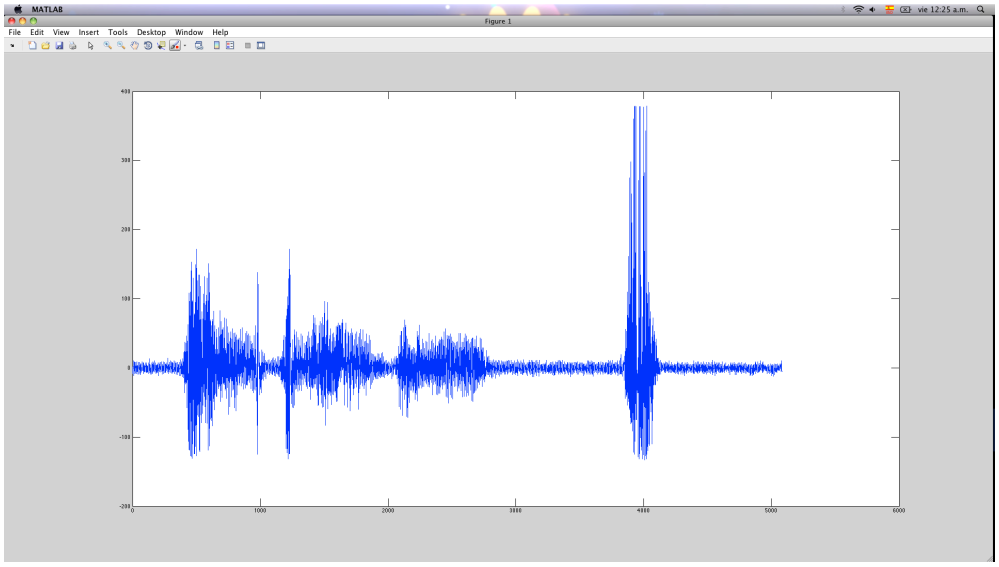

Figura 2.13: Señal de voz obtenida con el microcontrolador y tratada en Matlab.

Al realizar las primeras pruebas la calidad del sonido era muy pobre y la información capturada por el micrófono es sustituida por ruido.

La solución a este problema se alcanzó cambiando el amplificador operacional utilizado en el circuito encargado de adaptar la señal por uno de mejor calidad.

La selección de un nuevo amplificador se llevó a cabo buscando una respuesta en frecuencia sin distorsiones, una mayor tolerancia a impedancias desacopladas y un mejor desempeño hasta frecuencias mayores.

Pensando en satisfacer éstos parámetros se seleccionó el amplificador TL081CP, el cual tiene un desempeño más robusto al ser comparado contra el amplificador utilizado anteriormente (UA741CP).

Una vez remplazado el amplificador operacional, se repitieron las pruebas realizadas con anterioridad y se realizaron nuevas mediciones para poder determinar la respuesta en frecuencia del micrófono ya acoplado con el circuito encargado de adaptar la señal al microcontrolador.

Al rehacer la prueba de Matlab con el nuevo amplificador operacional se obtiene una salida más clara y la información contenida es fácilmente entendible.

Es en este punto del proceso donde se observa que la tasa de muestreo suministrada por sistema es demasiado baja y solo alcanza 2406 Hz.

Con el fin de conocer la respuesta en frecuencia del dispositivo, el micrófono fue remplazado con un generador de señales. Este generador se mantuvo con una señal senoidal de amplitud constante durante toda la prueba pero se realizó un barrido en frecuencia.

Los datos de salida fueron capturados en la terminal que se conecta al conversor analógico a digital del microcontrolador. Para la captura de los datos de salida se utilizó un osciloscopio, el cual nos permitió ver que la forma de onda era constante y nos dejó realizar las mediciones del nivel que tenía la señal a la salida.

Los datos obtenidos para conocer la respuesta en frecuencia se pueden ver en la tabla del anexo 1 y un resumen gráfico ésta disponible en la figura 2.14.

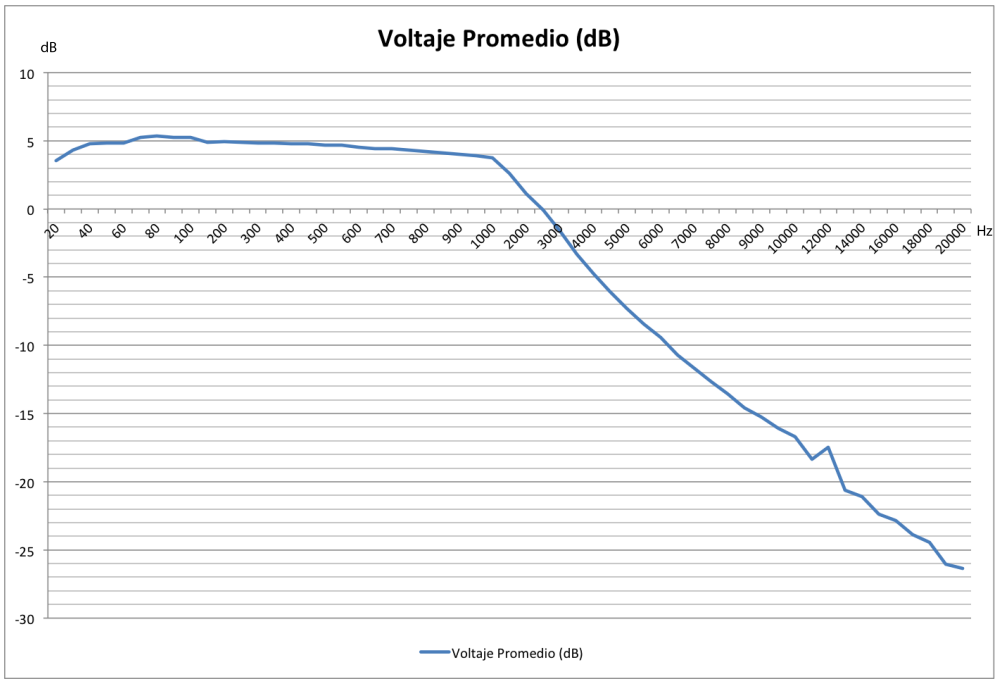

Figura 2.14: Respuesta en frecuencia del sistema.

Debido a la baja frecuencia de muestreo de nuestro dispositivo, el rango de frecuencias captadas por el micrófono se limita a 2000 Hz, esto para evitar efectos contraproducentes como el aliasing.

Una vez que se prueba la tarjeta de captura de sonido y se comprueban sus características, ésta es anexada a las otras tarjetas. Finalmente el bloque completo del micrófono se compone por los tres componentes mencionados durante todo el capítulo y el modelo funcional operado por el usuario, el cual incluye todos los componentes se puede ver en la imagen 2.15.

El micrófono no se puede probar completamente hasta no tener la consola de sonido programada en la computadora, ya que la salida del sonido se va realizar a través del hardware propio de la computadora y el método de escritura para la reproducción del sonido varía bastante de la simulación en Matlab al programa en java terminado.

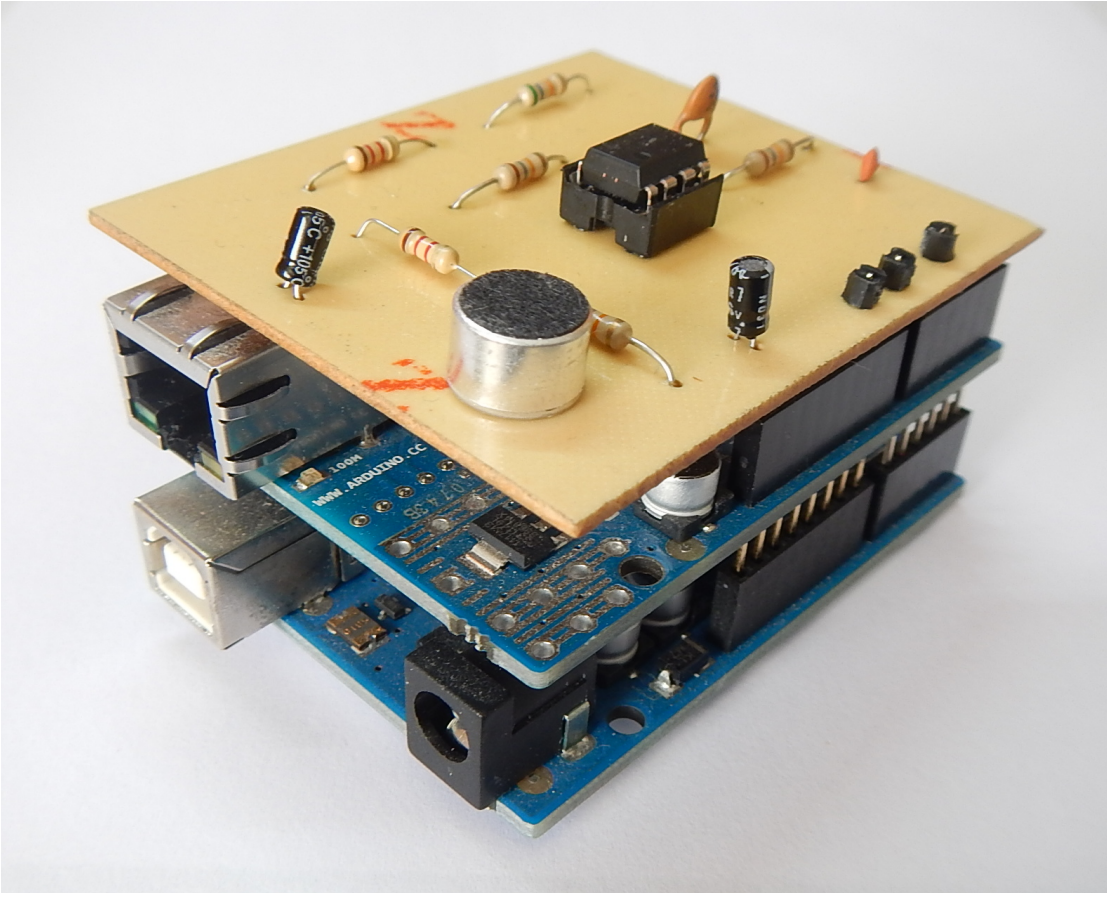

Figura 2.15: Modelo final del micrófono que incorpora todos los elementos.

2.4 CONCLUSIONES

El sistema de captura necesario para poder trabajar con la señal de forma digital dentro de la computadora se desarrolló al utilizar tres tarjetas.

La primer tarjeta se encarga de obtener una representación eléctrica adecuada de la señal para poder ser digitalizada posteriormente. Este tratamiento a la señal se logra por medio de un micrófono y posteriormente de un amplificador.

Posteriormente se empleó un microcontrolador ATMega 328, el cual está encargado de la conversión analógica a digital y del empaquetamiento de la información.

La última etapa del sistema de captura se implementó con una tarjeta de red que se encarga de enviar la información por medio de una red de área local hasta la consola de audio, la cual se ejecuta en una computadora de la misma red.

Se realizaron pruebas independientes a cada etapa del sistema de captura para la corrección de errores y con ello mejorar la calidad del sistema. Fue en este proceso donde se optó por cambiar el amplificador del dispositivo por uno de mejor desempeño.

CAPÍTULO 3: EL DESARROLLO DEL PROGRAMA

3.1 SELECCIÓN DE PLATAFORMA

Dentro de las opciones disponibles para la realización del programa se encuentran diferentes lenguajes de programación, por este motivo es necesario eliminar las opciones menos favorables para la elaboración del mismo y seleccionar el lenguaje que nos dé mayores beneficios.

En el trabajo realizado en la licenciatura se realizó una primera versión del sistema, la cuál fue programada en Cocoa, creando de esta manera una aplicación para correr en el sistema operativo Mac OS X. Esto nos permitía tener herramientas fáciles de programar pero con la imposibilidad de correr en computadores de otra marca, ya que este sistema operativo no se puede instalar de forma nativa en computadores manufacturados por otros fabricantes.

También se presentan problemas a la hora de ejecutar el programa en versiones más antiguas del sistema operativo, ya que las herramientas proporcionadas se basan en bloques ejecutables disponibles solo en la versión actual del sistema operativo.[21]

Para poder solucionar estas limitantes se buscan diferentes opciones para programar la aplicación que se encargue de controlar el sonido.

Dentro de las opciones se pensó programarlo en Visual Basic, lo que acarrea un problema similar al presentado con Cocoa, salvo que en este caso el programa podría ejecutar exclusivamente en Windows.

Para evitar el problema de depender de un sistema operativo y ganar la opción de crear un programa multiplataforma, la decisión tomada es implementar el sistema en Java, el cual puede ser ejecutado en cualquier computadora con el ambiente Java instalado, sin importar su sistema operativo. Esto aumenta la cantidad de usuarios que pueden utilizar el producto, ya que actualmente la plataforma está disponible para todos los sistemas operativos, celulares e inclusive dispositivos hogareños como televisores y reproductores de blu-ray.

[21] (La Rotta Santos 2012)

Además, tanto el ambiente necesario para ejecutar el programa como las herramientas de programación y desarrollo son gratuitas, por lo tanto no hay necesidad de pagar licencias por el uso de software para la creación del programa.

Para poder realizar la interfaz gráfica y tener todos los elementos avanzados de programación, como entorno de desarrollo se escoge el programa NetBeans, el cual nos permite hacer toda la parte de la programación gráfica sin la dificultad de trabajar directamente con código; nos proporciona una gran ayuda a la hora de escribir el conjunto de instrucciones que se encargan de procesar los datos recibidos del micrófono y nos da soporte para generar la unión entre las acciones realizadas en la interfaz gráfica y los cambios en la configuración pertinentes para poder ajustar el sonido de acuerdo a las peticiones del usuario.

3.2 INTERFAZ GRÁFICA

Para poder tener todos los componentes de una consola física emulados en la computadora, es necesaria la creación de una interfaz gráfica que tenga la capacidad de cambiar las variables que controlan el volumen, el balance y la ecualización del canal de audio.

Dichas características a manejar son obtenidas de los controles existentes en las consolas de mezcla disponibles en el mercado, ya que es necesario al menos poder igualar la oferta existente para crear un producto competitivo. También es importante conservar el orden para poder ofrecer el producto sin requerir capacitación adicional por parte del cliente, lo cual disminuye el tiempo entre la compra y operación por parte del consumidor.

Para realizar la programación se fija como meta las características mostradas la figura 3.1, donde se pueden observar los controles de volumen, silencio, ecualización, balance y la lista que permite seleccionar el micrófono a controlar.

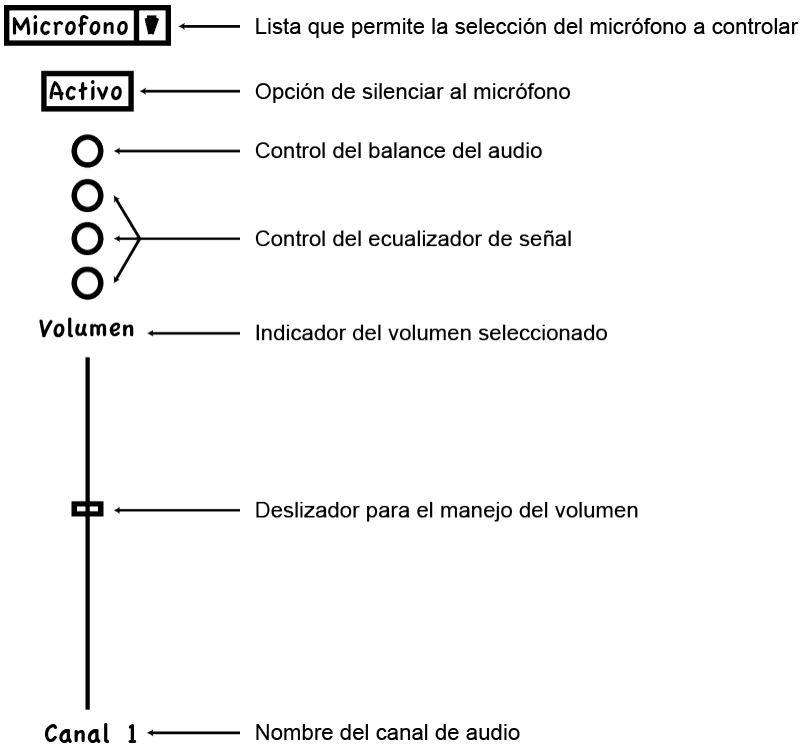

Figura 3.1: Elementos necesarios en la interfaz gráfica.

Una vez que los componentes necesarios son establecidos, se comienza la creación de la interfaz gráfica con una ventana en blanco y se rellena con los siguientes elementos de la librería existente en NetBeans:

- Se agregan los deslizadores encargados de manejar el volumen de los canales de audio. Estos deslizadores están contenidos en la clase JSlider.

  Son programados para que el usuario pueda seleccionar el nivel del volumen en un rango que oscila entre -80 y 13. Estos valores corresponden a los niveles máximos y mínimos en decibeles aceptados por el control de la línea de audio.

- Se crean las etiquetas que indican el nombre del canal y el nivel del volumen. La clase encargada de generar estos elementos es JLabel. En los deslizadores se agrega un par de líneas de código para indicar el nivel de volumen actual en la etiqueta de volumen de cada canal.

- La implementación del control del ecualizador estaba pensada hacerse con deslizadores circulares (existentes en la versión inicial programada en Cocoa), al no encontrarse este estilo de deslizador, fue necesario utilizar un spinner para cada banda.

Este elemento se compone de una caja donde el usuario puede introducir el valor deseado o bien cambiar el valor con las flechas existentes al lado derecho. Se programaran para permitir el cambio de uno en uno en los valores y oscilar entre 0 y 10. Este elemento está contenido en la clase JSpinner.

- Para poder seleccionar el balance del audio asociado a ese canal también fue necesario sustituir el deslizador circular por un deslizador normal, en este caso horizontal el cual nos permite el control deseado.

  En este caso se programa para seleccionar valores entre -10 y 10, este valor será dividido entre diez y pasado al control de la línea para que ajuste el parámetro.

- El control de encendido o pagado del canal es un botón de palanca (toggle button), el cual tiene dos posiciones intercambiables con cada interacción del usuario.

  Al pasar de la posición encendida a apagada, se deshabilita el deslizador de volumen, se silencia la línea y el botón cambia de etiqueta a apagado. Al regresar a su estado original el botón coloca el volumen presente en el deslizador dentro del control de línea, habilita del deslizador y cambia su etiqueta de nuevo.

  Esta herramienta está contenida en la clase JToggleButton.

- Por último se programa la caja de opciones para la selección del micrófono asociado a la línea. En este caso se muestra una lista de los micrófonos conectados, al contar con un solo micrófono esta opción no tiene razón de ser pero de todas formas es incluida en el diseño de la interfaz gráfica.

La consola de audio se programa como una clase independiente de la interfaz gráfica, esto se hace para brindar solidez al programa, poder detectar y corregir errores y para poder añadir funcionalidades nuevas sin dañar las capacidades ya conseguidas.

Dado que son dos clases diferentes, desde la interfaz gráfica solo se pueden cambiar los parámetros importantes dentro de la consola.

Los parámetros que se pueden cambiar son programados en un método público dentro de la clase que contiene a la consola de audio; al ser un

método público se puede acceder a ella desde la interfaz gráfica sin problema alguno.

Para poder cambiar el parámetro se llama el método desde la consola con los valores a cambiar dentro de la invocación.

Finalmente la interfaz gráfica queda implementada con cuatro canales (número fácilmente incrementable), los cuales tienen los elementos presentados en el boceto de la figura 3.1.

La interfaz gráfica terminada se puede observar en la figura 3.2.

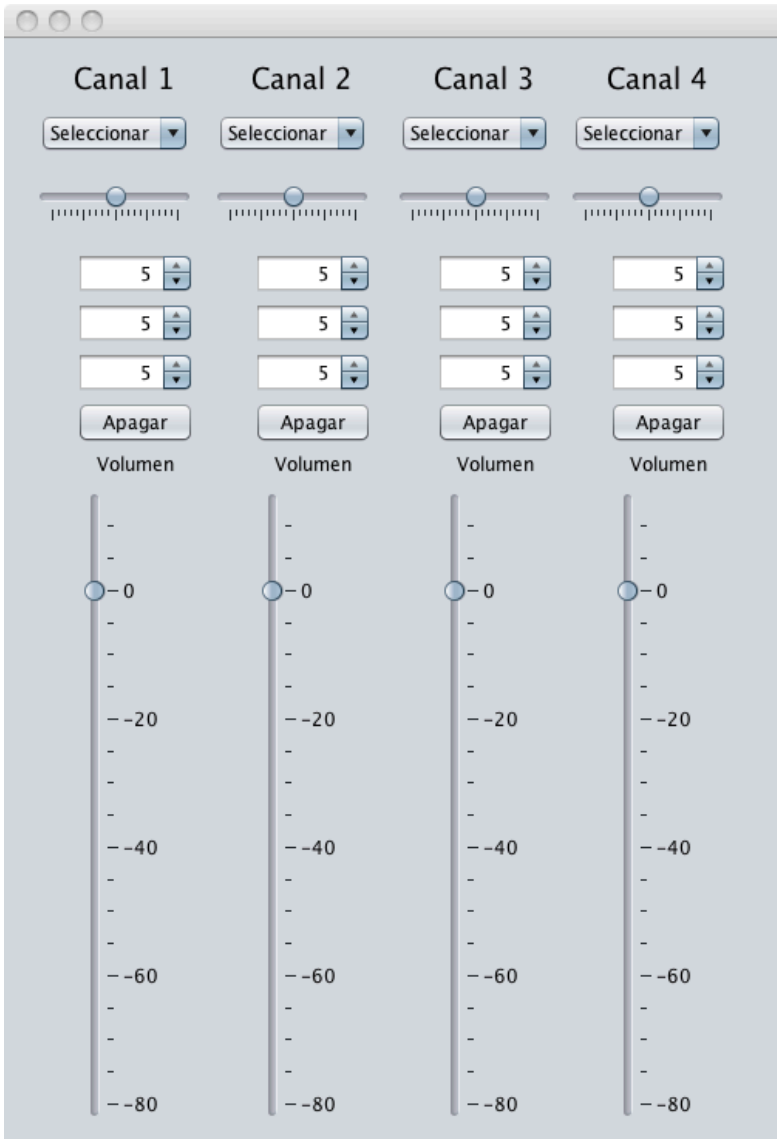

Figura 3.2: Apariencia final de la interfaz gráfica del programa.

3.3 CONEXIÓN CON EL MICRÓFONO

El funcionamiento de todo el sistema se basa en la transferencia de la información en forma digital del micrófono a la consola de audio, por lo cual es necesaria la creación de una conexión por medio de la red de área local entre las dos partes.

El esquema de conexión escogido es UDP, ya que utiliza encabezados cortos (lo cual ahorra ancho de banda), no se preocupa por corregir los errores (en una aplicación en tiempo real no es necesario hacer esto) y finalmente nos permite una conexión fácil de manejar, tanto en el lado del microcontrolador como en el lado de la consola.

Los dos dispositivos que van a interactuar deben de estar programados para ello, es por eso que primero se programa el microcontrolador para el manejo de la información por la conexión UDP y luego el programa de la consola es desarrollado para poder establecer la conexión de forma correcta.

La programación del microcontrolador es bastante sencilla, en parte por la familiarización desarrollada para realizar las pruebas anteriores de envío de información por medio de una conexión TCP (ver sección 2.3).

El código realizado se puede observar en el anexo 2.

El orden de la conexión es la siguiente:

1. Debido a que las direcciones IP no siempre va a ser la misma para la computadora, lo primero que se realiza en el programa es la actualización de la dirección dentro del micrófono, para lo cual se envía un paquete de información al microcontrolador, el cual toma la información de dirección y la almacena para enviar toda la información de sonido a esa misma dirección.

2. Como segundo paso el microcontrolador comienza a tomar muestras de sonido provenientes del micrófono y las envía en un paquete UDP, el cual va orientado a la computadora que ejecuta el programa que emula a la consola. Este proceso se realiza de manera automática hasta el reinicio del microcontrolador.

3. Una vez que se envía el paquete es recibido en la computadora por un puerto UDP especifico, donde cada micrófono realiza la conexión por un puerto distinto, todos los números de los puertos utilizados

tienen valores cercanos a 9000, esto con el fin de evitar colisiones con otros programas. Cuando se utiliza un solo micrófono, se elige el puerto 8888.

Dentro de la computadora la información es desempaquetada, acondicionada para el canal de sonido creado y enviada como una cadena de bits al canal de sonido en Java.

3.4 OBTENCIÓN DEL SONIDO EN JAVA

Para poder crear una pista funcional de sonido es necesaria la obtención de tres objetos en Java: un mezclador, una línea y la información con el formato correcto de audio.

El mezclador viene predefinido por el sistema, ya que un determinado sistema solo tiene disponibles un número fijo de mezcladores, los cuales están marcados por el hardware que posee el equipo con el cual se trabaja.

En algunos sistemas el acceso al hardware del equipo es bastante limitado, por lo cual es necesaria la implementación de un mezclador por software, para lo cual el entorno de java crea un mezclador virtual.

La línea de sonido es solicitada al mezclador con los parámetros fijados por el objeto que contiene las características de los datos que se quieren escuchar a través del sistema de sonido[22].

Para poder ejecutar una aplicación que reproduce audio hay dos maneras de hacerlo: leyendo totalmente el archivo que se quiere reproducir y por medio de una transmisión (o *streaming* en inglés).

En el primer caso es necesario tener el archivo de sonido completo al iniciar el comando, ya que todo el clip de audio es guardado en la memoria y se reproduce de manera lineal.

Cuando se tiene una transmisión, la información que conforma el sonido puede llegar poco a poco al canal de audio, en este caso es necesario

[22] (Oracle Inc. 2013)

fijar un buffer para ir depositando la información de llegada, la información del sonido debe de tener el flujo suficiente para no dejar al buffer vacío ni ser más rápido de lo que dicta la tasa de muestreo del sonido, ya que en este caso el caso el buffer se llenaría y se eliminaría la información que llega después.

Debido a que la aplicación desarrollada en el presente trabajo va a manejar datos en tiempo real y que se necesita ir introduciendo la información conforme se vaya generando en el micrófono, es necesario utilizar el esquema de transmisión. En este caso debemos de crear una línea de sonido en la cual vamos a introducir los datos.

Para poder obtener una línea de audio, primero se requiere seleccionar un mezclador de los que estén disponibles en el sistema los cuales varían de computadora en computadora. En el caso de la MacBook con la cual se está trabajando, se maneja un esquema donde se tienen limitaciones de acceso al hardware, en este caso Java da como solución un mezclador de software, el cual es utilizado para el desarrollo de la aplicación.

Cuando el software es ejecutado en otra computadora, se escoge el primer mezclador que soporte la línea solicitada.

Después se crea un objeto que va a definir las características del sonido, el cual se ensambla con las siguientes propiedades:

- Esquema de codificación (PCM)
- Tasa de muestreo
- Número de canales
- Número de bits por muestra
- Si se trata de información con signo o no
- El orden de los bits de entrada

Una vez se crea este objeto, es posible pedirle al mezclador una línea que sea compatible con el objeto creado.

Es posible que el mezclador no tenga una línea disponible para ciertas configuraciones, de tal forma que toca adecuar los datos de llegada para poder empatarlos a la línea disponible. En este caso la línea disponible es de 16 bits, por lo cual fue necesaria la adaptación de los datos de entrada.

También es necesario realizar el des-empaquetamiento de la información, ya que el micrófono envía paquetes UDP con la información de 10 muestras. Este valor de diez muestras puede ser intercambiable, a mayor

número de muestras por paquete más retraso en la señal y menor ancho de banda necesario (por ahorro de encabezados).

Para poder hacer la adaptación de los datos del micrófono a una línea de sonido disponible, fue necesaria la programación de un método dentro de la clase principal, el cual se encarga de recibir el conjunto de 30 bytes (en el caso de un paquete con diez muestras con tres bytes por muestra), toma muestra por muestra y la adapta para poder encajar en un canal de 16 bits por muestra.

Es en este punto donde la información es enviada al ecualizador, que recibe las muestras individuales y las regresa de igual manera.

Al final del proceso el método arregla los bits resultantes en un arreglo de 20 bytes los cuales ya están listos para ser introducidos en la línea de sonido pedida con anterioridad. Este arreglo de bytes es el elemento que regresa cuando el método es invocado por la función principal.

Para poder adecuar la señal recibida del sistema de captura a una línea de sonido, es necesario realizar una interpolación de las muestras recibidas. La interpolación esta contenida dentro de la librería *Sampled Audio* de Java y es realizada automáticamente por el sistema al introducir las muestras dentro de la línea solicitada.

3.5 CONTROLANDO EL SONIDO

El correcto control de la información proveniente del micrófono se logra al establecer el flujo de datos mostrado en la figura 3.3. Algunos de los pasos mostrados en la cadena son programados como funciones independientes para poder ser utilizado independientemente de línea, otros bloques se programan para estar asociados a la línea de sonido.

Figura 3.3: Flujo para el tratamiento de la señal.

El primer paso se realiza cuando el usuario selecciona de una lista al micrófono que desea asociar al canal. En el caso de tener un solo micrófono este paso es irrelevante.

Con el fin de implementar el ecualizador, es necesario que la información desempaquetada sea introducida a un método que se encargue de ecualizar el sonido. Dentro del método desarrollado, el ecualizador es programado por medio de filtros y los datos que regresan son los datos de sonido, los cuales pueden ser puestos directamente en la línea de sonido.

La implementación del ecualizador esta más detallada en la sección 3.6.

Dentro del paquete de audio muestreado (*sampled audio)* de Java, en el cual nos basamos para poder utilizar los recursos de sonido de la computadora; están incorporados algunos controles básicos que necesitamos para poder trabajar con la consola de audio. Estos controles contienen ayudas para poder manejar el volumen, el balance y la opción de poder silenciar la línea deseada.

Como el sistema está pensado para trabajar con los datos en tiempo real, otras opciones de control relacionadas con el avance de la pista (pausa, reproducir, avanzar, etc.) son irrelevantes.

La integración de la interfaz gráfica desarrollada en la sección 3.2 y el programa encargado del tratamiento y salida del sonido es necesaria para poder controlar el sonido.

El programa necesita tomar en cuenta los cambios realizados por el usuario en la interfaz gráfica, estos cambios son manejados por medio de llamadas a métodos para alterar los parámetros de la línea con la que se está trabajando. Es de esta manera que los valores de amplitud, silencio y balance son alterados.

En el caso del ecualizador, los cambios se realizan directamente a variables que multiplican la amplitud de cada rango de frecuencias y con esto se obtiene un ecualizador funcional.

3.5.1 CONEXIÓN DE LA INTERFAZ GRÁFICA CON LOS CONTROLES

El diseño del programa está pensado para tener varios módulos funcionales programados por separado, los cuales pueden funcionar de manera exitosa sin la necesidad de estar unidos en un programa central.

Por este motivo se realiza la programación de dos clases, las cuales pueden funcionar independientemente, aunque es necesario el correcto funcionamiento de ambas para que el programa se ejecute como es debido.

La primera clase contiene los elementos necesarios para crear una interfaz gráfica que proporcione los medios de control al usuario final. Esta clase se llama panelControl.java y es en esta clase donde se programan los elementos necesarios para poder modificar las variables de control del sistema.

Dentro de la segunda clase, la cual se ejecuta primero al iniciar el programa (por lo que va a ser la clase principal), la cual se llama audioBoom.java, se programan tres métodos públicos, los cuales son llamados desde la primera clase para poder modificar los parámetros de sonido. Estos métodos manejan el volumen, el balance y por último los ajustes de ecualización de cada canal.

Las dos clases se ejecutan al mismo momento, salvo que la interfaz gráfica está en un estado de espera hasta que se interactúa con algún control.

Cuando se interactúa con cualquier elemento de la interfaz gráfica se genera un cambio de estado que es capturado por el sistema, al percibir el cambio de estado el sistema ejecuta un método donde está programada la llamada al cambio de los parámetros dentro de la clase principal.

Los métodos programados dentro de la clase principal tienen como datos de entrada el canal al cual aplicar el cambio y el nuevo valor del atributo dado. Su funcionamiento es sencillo, ya que solo modifican las variables de control asociadas a cada canal, las cuales se declaran como privadas al inicio de la clase para que cualquier método dentro de la clase pueda acceder a ellas. Esto se realiza con el fin de tener varios métodos funcionando de manera paralela sin el problema de tener variables locales para cada método.

Cada método genera un hilo de procesamiento separado, por lo cual pueden correr de manera paralela. Esto proporciona al usuario la capacidad de cambiar los atributos de la señal mientras está siendo procesada.

Cuando el buffer de la línea de sonido está lleno y este tiene un tamaño considerable, los cambios realizados en la interfaz gráfica tienen cierto retraso en ser percibidos. Esto se debe a que los datos contenidos en el buffer fueron procesados con anterioridad y no responden a los cambios efectuados; solo los datos que están siendo o serán procesados son los afectados por los cambios establecidos dentro de la interfaz gráfica.

La clase principal del programa se puede observar en el anexo 3.

3.6 REALIZACIÓN DEL ECUALIZADOR

Dentro de la consola es imprescindible el uso de un ecualizador, el cual se encarga de ajustar los parámetros de amplitud para determinadas frecuencias del sonido. En este caso Java no tiene ninguna herramienta que nos pueda ayudar, por lo que es necesaria la implementación de un método que brinde la capacidad de ecualizar el sonido proveniente del micrófono.

El objetivo es realizar un ecualizador de tres bandas de frecuencia, el cual permita controlar las frecuencias bajas, medias y altas del sonido capturado por el micrófono. Es por ello que es necesaria la implementación de tres filtros, cada uno de ellos diseñado especialmente para cada banda de frecuencias.

Para poder establecer los requisitos necesarios para nuestro método, es necesario tener el filtro con el cual se va a trabajar, ya sea uno con respuesta finita al impulso o con respuesta infinita.

Para la aplicación que se desarrolló, un filtro IIR (*Infinite Impulse Response*) tiene una mayor aceptación por las siguientes razones:

- El filtro puede ser obtenido directamente de su contraparte analógica por medio de la transformada bilineal.

- Son más rápidos de computar que un filtro de características similares implementado como un FIR (*Finite Impulse Response*).

- Se necesita un menor espacio de memoria para poder operar, en nuestro caso este punto es irrelevante, ya que al ser filtros pequeños y contar con una memoria grande tenemos suficiente espacio para operar dentro de la computadora.

La desventaja de utilizar un filtro IIR radica en la inestabilidad creada por los coeficientes al sufrir pequeños efectos numéricos adversos como el truncamiento. También carecen de una fase lineal en la salida, lo que puede ocasionar problemas posteriores[23].

Para la implementación de los mismos se escogen filtros de segundo orden, los cuales tienen una buena respuesta en frecuencia y solo necesitan de dos lugares de memoria para poder guardar los estados pasados.

La configuración del filtro se conoce como IIR de segundo orden de forma directa II. Todos los filtros con estas características se construyen como se observa en la figura 3.4.

[23] (Kahrs and Brandenburg 1998)

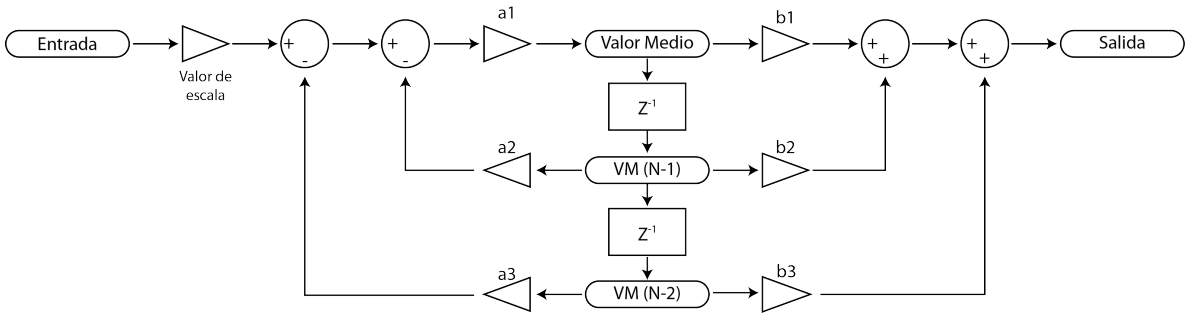

Figura 3.4: Composición de un filtro IIR de segundo orden de forma directa II.

Para la implementación del filtro en Java es necesario guardar las variables que están en los rectángulos con bordes redondeados (casillas de valor medio o VM), la entrada y la salida son arreglos con capacidad para el número de elementos a tratar, también se puede eliminar los arreglos e introducir los valores de uno por uno como escalares sencillos.

Con el fin de almacenar los valores de "valor medio" para las iteraciones siguientes, se establece un arreglo de tres elementos, el cual almacena el valor calculado en la iteración actual y al final de calcular el valor de salida para el ciclo actual, recorre los valores de iteraciones anteriores una posición.

Para poder implementar el filtro se necesitan los valores de los coeficientes, los cuales multiplican el valor de la señales que entran a los triángulos, son estos valores los que indican cómo se va a manejar el filtro y son la única diferencia (junto al valor de escala que adapta la señal antes de entrar al filtro) en la implementación de los tres filtros.

El valor de escala se encarga de ajustar la señal para tener a la salida una señal con una amplitud similar a sus niveles de entrada al filtro.

Una vez que se tienen los elementos necesarios para poder construir el filtro, es necesaria la selección de las bandas en las cuales se va a trabajar. El rango de audición del oído humano comprende todas las frecuencias existentes entre 20 y 20 000 Hz aunque su respuesta no es igual para todas.

Al tener un ecualizador de tres bandas es necesario seccionar el rango auditivo humano en tres bandas. Los limites para cada banda de frecuencias se establecen pensando en filtrar las frecuencias bajas, medias y altas.
Lamentablemente no hay un acuerdo común dentro de la industria para definir el comienzo y final de cada banda, dentro de las implementaciones

comerciales de filtros se observa que las frecuencias más utilizadas para los filtros son:

- Filtro paso bajas: Frecuencia de corte a 250 Hz.
- Filtro paso banda: 250 a 4 000 Hz.
- Filtro paso altas: Frecuencia de corte a 4 000 Hz.

Dentro de este espectro hay que tener cuidado con las frecuencias bajas, ya que muchas veces el sistema de altavoces no tiene el rango dinámico lo suficientemente grande para poder manejar las frecuencias más graves, así que es probable que la señal se atenúe y no sea percibida por el usuario debido a esos efectos contraproducentes en el sistema de salida.

También es el rango de frecuencias que necesita una mayor amplificación y algunos sistemas incluyen un área específica para el manejo de estas señales.[24]

Una vez establecidas las bandas de frecuencia que se deben filtrar y el estilo del filtro, se procede a diseñar los filtros con ayuda de Matlab y su herramienta para el diseño de los mismos, la cual se invoca en la ventana de comando con FDATool.

3.6.1 FILTRO PASO BAJAS

Para tener mejores resultados en el diseño del filtro, se eligió un filtro que deje pasar todas las frecuencias por debajo de 250 Hz, en lugar de un filtro pasa banda con un rango de 20 a 250 Hz. De esta manera la respuesta en frecuencia del filtro se puede observar en la figura 3.5. Los coeficientes necesarios para obtener esta respuesta en el filtro están en la tabla 3.1.

[24] (Apple Inc. 2010)

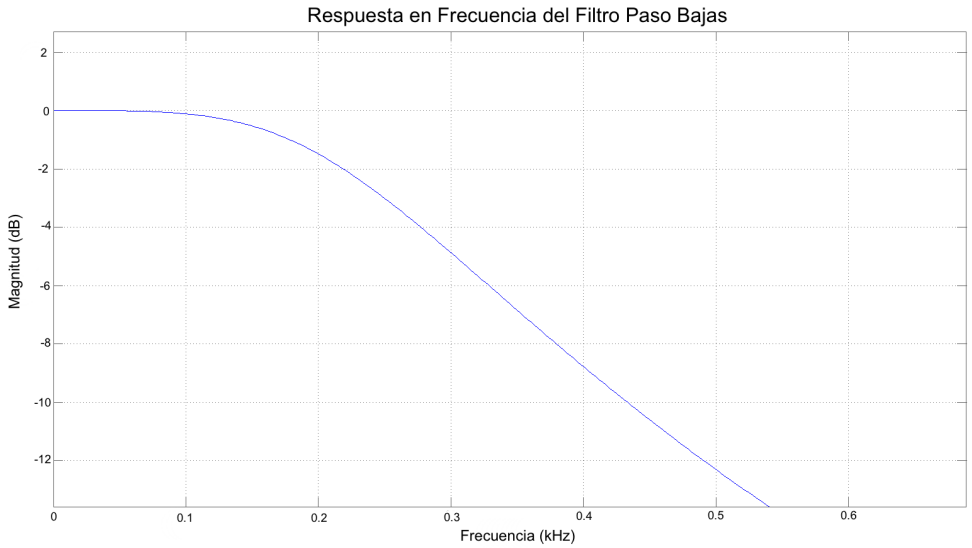

Figura 3.5: Respuesta en frecuencia del filtro paso bajas.

Tabla 3.1: Coeficiente para el filtro paso bajas.

Coeficiente	Valor
A1	1
A2	-1.9496374358791775
A3	0.95087485286562701
B1	1
B2	2
B3	1
Escala	0.00030935424661235632

Una vez establecidos los valores de los coeficientes A1, A2 y A3 se guardan dentro del arreglo de coeficientes en el programa de Java, el cual será llamado por el código del filtro.

Dado que el valor de los coeficientes B1 y B3 es uno, la salida del filtro se simplifica, ya que solo es necesario sumar el valor almacenado en el retraso correspondiente. Para B2, el valor VM(N-1) es multiplicado por dos y es sumado a la salida.
Por este motivo la salida es programada directamente en el código y los coeficientes B's no son almacenados en el arreglo.

3.6.2 FILTRO PASO BANDA

La importancia del rango medio de frecuencias radica en que la densidad espectral de energía de la voz se encuentra mayoritariamente en este rango, por lo cual nos permite manejar el nivel de la voz con facilidad.

Para poder trabajar con esta banda se decidió implementar un filtro pasa banda, el cual deja pasar las frecuencias desde 250 hasta 4 000 Hz. La respuesta en frecuencia del filtro se puede observar en la figura 3.6 y los coeficientes necesarios para obtener dicha respuesta se encuentran en la tabla 3.2.

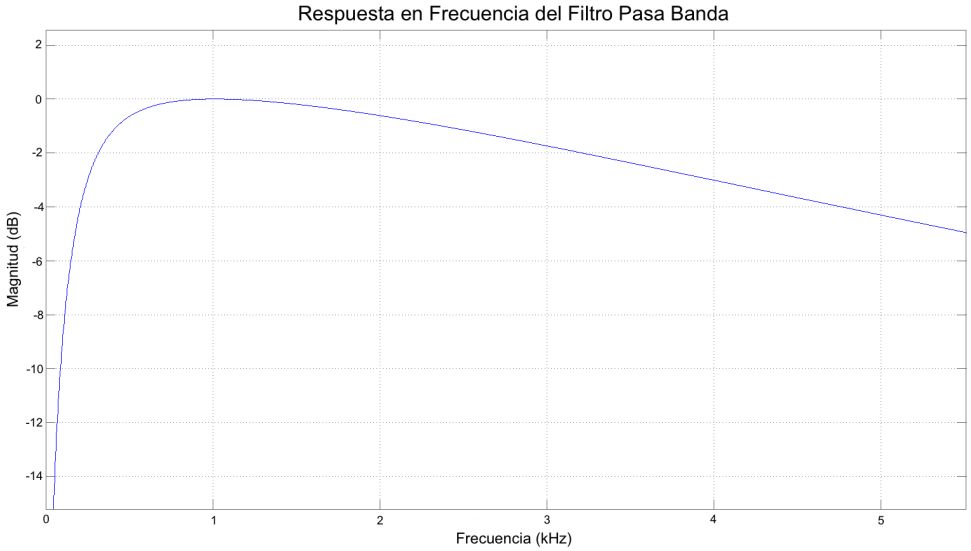

Figura 3.6: Respuesta en frecuencia del filtro paso medias.

Tabla 3.2: Coeficientes para el filtro pasa medias.

Coeficiente	Valor
A1	1
A2	-1.5854125879544436
A3	0.5993693368192391
B1	1
B2	0
B3	-1
Escala	0.20031153315903807

En este caso, como el coeficiente B2 es cero, la salida solo está compuesta de los elementos B1 y B3, ambos elementos son unitarios pero de signos opuestos, lo que significa que para la salida solo se suma el valor medio actual y se resta el valor medio con dos retrasos.
Por este motivo el código es más corto y más rápido de ejecutar.

3.6.3 FILTRO PASO ALTAS

De igual manera que en el filtro paso bajas, no se va utilizar un filtro paso banda, ya que no se obtienen beneficios en el sonido eliminando los elementos arriba de 20 000 Hz y el filtro nos da un mejor corte si utilizamos sencillamente un filtro paso altas.

La respuesta en frecuencia del filtro puede ser observada en la figura 3.7 y los coeficientes necesarios para la construcción del mismo están contenidos en la tabla 3.3.

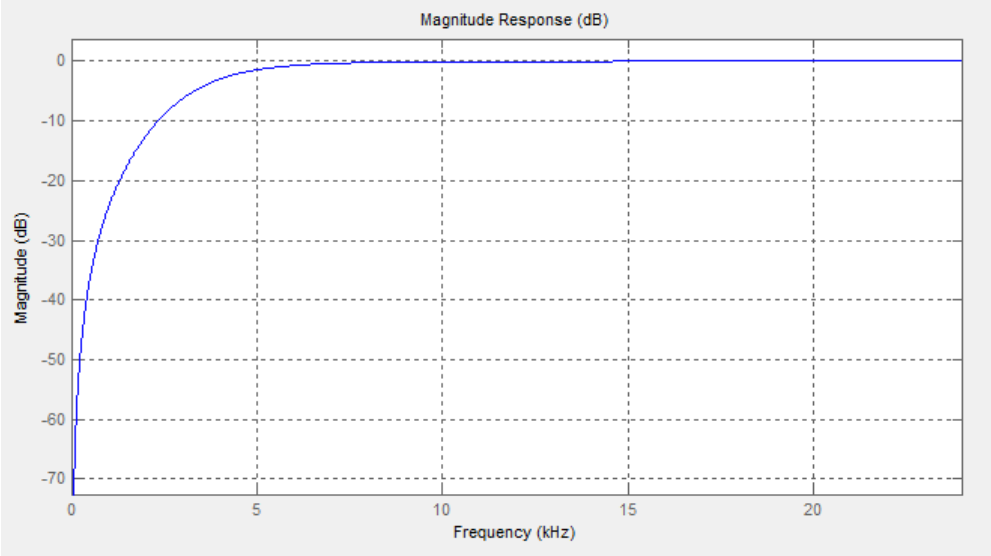

Figura 3.7: Respuesta en frecuencia del filtro paso altas.

Tabla 3.3: Coeficientes para el filtro paso altas.

Coeficiente	Valor
A1	1
A2	-1.279632424997809
A3	0.47759225007251715
B1	1
B2	-2
B3	1
Escala	0.68930616876758155

3.6.4 IMPLEMENTACIÓN DE LOS FILTROS

Todos los filtros fueron implementados con una frecuencia de muestreo del sistema de 44100 Hz, ya que esta es la frecuencia con la cual trabajan la mayoría de equipos comerciales de audio y es la frecuencia ideal para que el equipo desarrollado trabaje con un ADC más robusto.

El resultado obtenido al implementar los tres filtros se puede observar en la figura 3.8. Existe un traslape entre las diferentes bandas de frecuencia debido a que los filtros no son ideales. Al ser los filtros de un grado bajo el traslape es mayor pero la velocidad de cómputo aumenta.

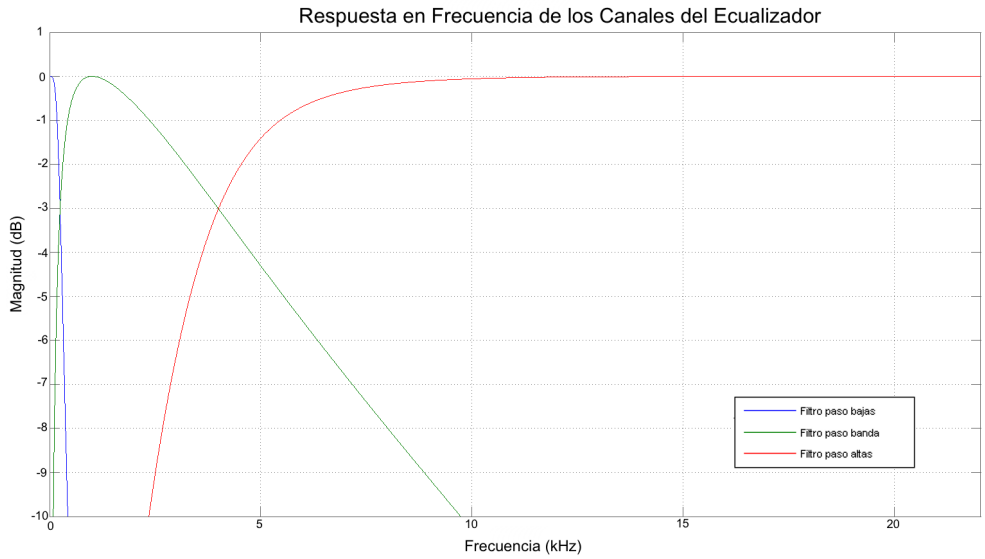

Figura 3.8: Respuesta de las bandas de frecuencia del ecualizador.

Una vez que se tienen los tres canales filtrados, se procede a multiplicar la salida de cada uno por el factor elegido por el usuario a través de la interfaz gráfica, de esta manera se realiza la ecualización, ya que se le puede dar un peso diferente a cada banda de frecuencias.

Los valores que selecciona el usuario están mostrados de cero a diez, lo que ocasiona que el usuario pueda seleccionar desde la eliminación completa de la banda hasta tener el doble del valor de entrada para esa banda.

Todo el proceso de ecualización se programó en un solo método de Java, el cual se encarga de recibir los valores des-empaquetados del micrófono, ecualizar la señal y enviar los valores de salida como la respuesta del método.

Como ayuda adicional para poder manejar un flujo de datos, también se programó un método donde la información que representa a la señal se recibe y se envía como una cadena de bits. Este método sirve para ecualizar el sonido de un archivo almacenado en la computadora. En este caso los bits se transforman a números, se realiza el proceso de ecualización y finalmente se realiza la conversión de números a bits, para poder regresar una cadena similar a la recibida por el método.

Debido a la tasa de muestreo baja del micrófono (2406 Hz), fue necesaria la adaptación de los filtros para poder operar sin problemas con los datos recibidos del micrófono, de esta manera se ajustaron los coeficientes para poder ecualizar tres bandas en el pequeño espacio espectral que se cuenta.

Las bandas para este caso quedaron:

- Filtro paso bajas: Frecuencia de corte a 250 Hz.
- Filtro paso banda: 250 a 750 Hz.
- Filtro paso altas: Frecuencia de corte a 750 Hz.

Las pruebas para los filtros con una tasa de muestreo alta se realizaron con archivos de audio guardados en la computadora, mientras que las pruebas para las bajas tasas de muestreo se realizaron con el uso del micrófono.

3.7 CONCLUSIONES

Con el fin de manejar los datos obtenidos del sistema de captura implementado en el capitulo dos, se programó una consola de sonido en Java para ser ejecutada en cualquier sistema operativo.

La consola permite la interacción del usuario con el sistema completo por medio de una interfaz gráfica, donde el interesado puede cambiar los parámetros de volumen, silencio, ecualización, balance y selección de canal para los datos provenientes del micrófono.

Aparte de la interfaz gráfica la consola de audio establece una conexión funcional entre la computadora y el micrófono por medio de un enlace UDP. La información recibida es desempaquetada e introducida al ecualizador, el cual se encarga de enfatizar ciertas bandas de frecuencia o de disminuir el nivel de otras.

El ecualizador se implementa por medio de tres filtros: paso bajas, paso banda y paso altas. Y se programaron para ser ejecutados en paralelo y al final se realiza la suma de la señal de salida de cada uno de ellos.

Los filtros del ecualizador son del tipo IIR de segundo orden de forma directa II, los cuales obtienen una respuesta en frecuencia aceptable con un tiempo de procesamiento bajo para las muestras.

La salida del ecualizador es introducida en una línea de sonido, solicitada al sistema con anterioridad bajo un esquema de transmisión (streaming), para la reproducción de la señal a través de la tarjeta de sonido de la computadora.

CAPITULO 4: PRUEBAS Y ANÁLISIS DE RESULTADOS

4.1 INTEGRACIÓN DEL SISTEMA

Cuando los elementos del sistema fueron desarrollados se procede a la integración de todos los componentes.
En la figura 4.1 se pueden apreciar todas las partes del proyecto conectadas y trabajando de manera normal.

La unión entre el micrófono y la consola se realizó por medio de un conmutador de red común y cable UTP.

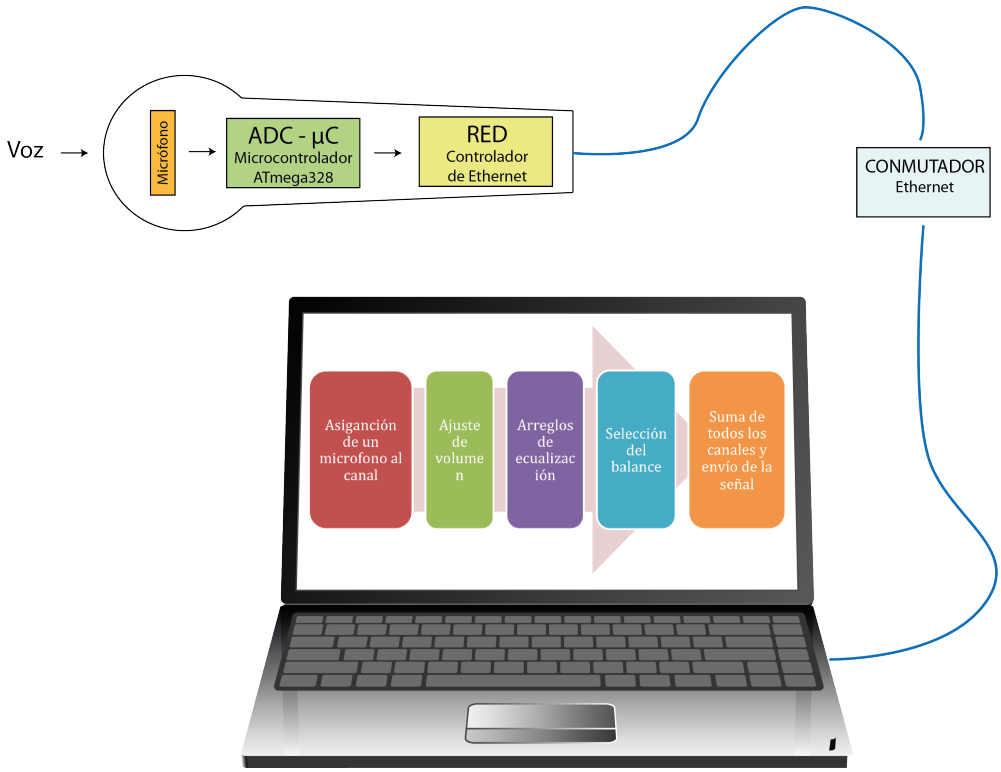

Figura. 4.1: Esquema general del proyecto.

Una vez que el sistema fue ensamblado, se procedió a probar su funcionamiento transmitiendo sonido y escuchándolo sin ningún protocolo fijo.
Cuando se comprobó el correcto funcionamiento se procedió a evaluar la calidad del sistema con un método más robusto.

4.2 MÉTODO DE PRUEBA

Para poder comparar los resultados obtenidos al construir el sistema, es necesaria la realización de pruebas para conocer el desempeño total del sistema en una forma cuantitativa.

Una evaluación objetiva de la calidad del audio, permite catalogar la calidad de un sistema y/o proceso de audio antes de salir al mercado y para poder conocer las características de la señal dentro del mismo sistema.

Tradicionalmente las pruebas se realizan reuniendo las opiniones de las personas que son sometidas a una prueba de escucha en condiciones controladas para asegurar que todas las personas que colaboran tienen el mismo entorno acústico. Éstas pruebas son de alto costo y requieren un elevado tiempo para ser ejecutadas.

Lo ideal es tener un método objetivo que pueda evaluar las características del sonido, sin tener que someter el sistema a pruebas subjetivas dependientes del oído de las personas.

Para poder cumplir con esta meta se comenzaron a explorar las características del sonido y su relación con la calidad percibida por las personas, de esta manera se probaron distintos métodos, como la relación señal a ruido y la distorsión acústica total. Lamentablemente estos métodos no pueden emular los juicios humanos sobre todo cuando la información es comprimida por códigos no lineales y no estacionarios como los manejados actualmente[25].

Con el fin de resolver esta problemática, la Organización Internacional de Telecomunicaciones publicó en el año 2000 una recomendación técnica (Rec. UIT-R BS.1387-1) que contiene un método para mediciones objetivas de la calidad de audio percibida (conocida como PEAQ, por sus siglas en inglés).

Para la elaboración de esta recomendación se estudiaron seis propuestas diferentes, de las cuales se escogieron los aspectos más relevantes de cada una y se desarrolló un método robusto que demostró su desempeño en diversas pruebas de laboratorio.

[25] (Zheng, Zhu and Song 2012)

Las pruebas objetivas deben de estar referidas y comparadas con pruebas subjetivas, ya que para validar sus resultados se deben de tener en comparación los resultados obtenidos con sujetos de prueba (que al final van a ser los usuarios de los sistemas de audio).

En el dominio subjetivo se tiene el grado de diferencia subjetiva (SDG, *subjective difference grade*), la cual está dada en una escala de 0 a -4 y comúnmente se le suman cinco unidades para expresarla en términos positivos. Ésta escala puede ser representada como se ilustra en la figura 4.2.

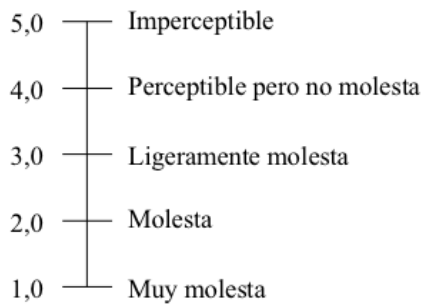

Figura 4.2: Escala del grado de diferencia subjetiva.

En el dominio objetivo ésta escala es substituida por el grado de diferencia objetiva (ODG, *objective difference grade*), la cual es la variable de salida del método planteado en la recomendación y sigue el mismo comportamiento que la SDG.

El método tiene como limitante una sincronía necesaria entre las dos señales, las cuales tienen un límite máximo de tolerancia a la no sincronía de 24 muestras, que a la tasa de muestreo necesaria para la prueba (48 KHz) representan solamente 0.5 milisegundos.

Adicionalmente a la prueba de calidad es necesario conocer la tasa de bits necesaria para que el micrófono funcione de manera correcta. Esta tasa de bits nos va a servir para conocer la velocidad mínima de conexión entre el micrófono y la consola de audio y para conocer cuántos canales de audio pueden ser enviados a través de la conexión a la red de área local con la que cuenta la consola.

4.2.1. PREPARACIÓN DE LA PRUEBA

Para poder realizar una prueba objetiva sobre un dispositivo, es necesaria la adecuación de los componentes de acuerdo a lo presentado en la figura 4.3. Esta misma disposición de los elementos sirve para realizar pruebas digitales y analógicas[26].

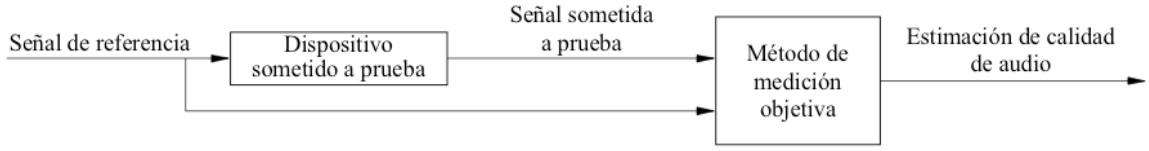

Figura 4.3: Preparación del dispositivo sometido a prueba.

El primer paso en la preparación de la prueba es la creación de un archivo de audio, el cual debe tener una duración aceptable por la recomendación de la ITU. Con este fin se crea un archivo de 10 segundos, el cual contiene dos tonos de frecuencias diferentes (uno a 400 Hz y otro a 900 Hz) y un fragmento hablado.

La prueba requiere de dos señales diferentes para poder trabajar, la señal de referencia y la señal sometida a prueba, las cuales deben de ser conseguidas y procesadas para poder utilizarlas en Matlab.

Para poder capturar las dos señales es necesaria la utilización de un programa que se encargue de capturar las señales de audio en la computadora. Para el desarrollo de la prueba se escoge el programa Audacity, el cual tiene la capacidad de capturar y procesar la señal para adaptarla a la rutina de Matlab que se encarga de la comparación de la señal.

Respecto a la tasa de bits necesaria para la conexión del micrófono se cancelan todas las conexiones que procedan o tengan como destino un lugar externo a la red de área local a la cual está conectada la computadora, dado que en la red solamente existe la computadora y el micrófono, todas las conexiones serán entre estos dos elementos y gracias a ello la velocidad reportada por el sistema corresponde únicamente al micrófono conectado.

[26] (International Telecommunication Union 2001)

Al tener el esquema sin conexiones externas, se procede a conectar el micrófono y enviar información a la consola de audio emulada en la computadora. Dentro de la información de estado del sistema propio del sistema operativo se puede leer la velocidad en bits de las conexiones existentes. Es éste valor el que indica la tasa de bits existente para que se dé la correcta transmisión de datos entre el micrófono y la computadora.

4.2.2. REALIZACIÓN DE LA PRUEBA

Lo primero que se desarrolla en la prueba es la obtención de la señal de referencia con la cual se a trabajar.

La señal de referencia se crea al reproducir el archivo de audio a través de las bocinas con las cuales se va a realizar la prueba, el sonido generado es grabado en la computadora con la ayuda de un micrófono probado y garantizado con anterioridad. Este archivo es guardado y no es sometido a ningún tipo de procesamiento.

En este paso se utilizan las bocinas para incluir su pérdida de calidad dentro del modelo, ya que para obtener la segunda señal es indispensable su utilización.

La señal sometida a prueba se obtiene al capturar la información de salida del programa desarrollado para emular la consola de audio después de que la información ha recorrido todo el sistema, el primer paso es la captura de la información en el micrófono a probar, para lo cual se coloca el micrófono entre las bocinas y se reproduce el archivo de prueba; la información es capturada de manera correcta y enviada por la red de área local hasta la consola con la cual se trabaja.

Cuando el sonido ha recorrido todo el sistema, es necesaria la captura de la información resultante. Esta captura se realiza al grabar la salida de audio del sistema mientras la consola y la prueba se están ejecutando.

La señal sometida a prueba en este caso no es tratada dentro de la consola de ninguna manera, ya que con el ecualizador se puede incrementar su calidad, pero esto queda fuera del alcance de la prueba y por lo tanto se manejan los datos sin procesar.

Una vez obtenidas las dos señales, se procede a realizar su alineación en el tiempo, ya que la recomendación establece que la diferencia máxima de tiempos entre las dos es de 24 muestras.

La alineación se realiza en la interfaz gráfica de Audacity, ya que deja ver con facilidad de forma de onda de la señal y poder alinear los fragmentos con tan solo cortar muestras sobrantes a las señales.

Al terminar la sincronización de las dos pistas, se procede a exportar los archivos de sonido, los archivos resultantes deben de tener formato .wav y una tasa de muestreo de 48000 Hz.

Para poder alcanzar la tasa de muestreo necesaria para evaluar la señal es necesario realizar una interpolación de los datos registrados al llegar las diferentes muestras.
Esta interpolación es realizada por el programa Audacity por medio de una interpolación limitada en banda, la cual utiliza una superposición de funciones sinc, las cuales están desplazadas y escaladas para ajustar sus máximos a los puntos por los cuales pasa la señal original[27].

4.2.2.1. COMPARACIÓN DE LAS SEÑALES

Para poder obtener un resultado es necesaria la comparación de las dos señales obtenidas mediante las grabaciones de la señal.

Para poder efectuar la comparación es necesaria la creación de un programa en Matlab que realice los procedimientos descritos por la recomendación de la ITU para poder establecer las diferentes variables que nos van a ayudar a definir la calidad final del sistema.

La programación se realiza en diferentes bloques, de ésta manera nos permite trabajar el programa por medio de repeticiones sencillas para cada uno de los fragmentos en los cuales es dividido el archivo, ya que se divide en ventanas de 2048 muestras, con una superposición de 1024 para realizar las comparaciones.

De manera similar a la programación de Java, el hecho de tener el programa desarrollado por bloques nos permite corregir los errores de una manera más sencilla e ir ensamblando el programa con facilidad.

[27] (Smith 2014)

El programa analiza cuál de los archivos tiene la menor duración y hace los cálculos de calidad solamente por ese tiempo; también revisa que los archivos tengan la frecuencia de muestreo apropiada (48 000 HZ) para el buen funcionamiento de los filtros que emplea.

En la programación se escoge el modelo definido como básico en la recomendación, ya que el cambio en los resultados es insignificante para la aplicación deseada.

Ciertos bloques del programa se encuentran descritos por medio de un pseudocódigo dentro de la recomendación de la ITU, por lo cual la implementación en Matlab se puede realizar de forma sencilla. En muchas etapas es necesaria la utilización de la transformada rápida de Fourier, en donde Matlab nos provee una gran ayuda, ya que esta es una función ya programada dentro del ambiente de desarrollo.

En algunas etapas de los bloques se presentan problemas por nombres de archivo y variables, los cuales son generados para el uso interno dentro de una sola función, volviendo los datos inaccesibles desde otras funciones del programa. La solución al problema consiste en dejar los valores importantes declarados como matrices globales, donde los datos pueden ser leídos o escritos desde cualquier programa o función en ejecución.

Una vez obtenidos los dos archivos de sonido, se introducen al programa en Matlab. Después de realizar el procesamiento, el programa entrega resultados por cada bloque de 2048 muestras analizadas y al final calcula las variables de salida del modelo (MOV's model output variables), las cuales utiliza para computar el grado de diferencia objetiva existente entre las dos señales.

4.3. RESULTADOS DE LA PRUEBA

Cuando las dos grabaciones necesarias para poder conocer el desempeño del sistema son realizadas e introducidas en el programa creado en Matlab para poder determinar su calidad, como resultado obtenemos las variables de salida del modelo y finalmente el grado de diferencia objetiva existente entre la señal transmitida por nuestro sistema y la señal de prueba.

Para nuestra implementación los resultados obtenidos se pueden apreciar en la tabla 4.1.

Tabla 4.1: Valores de las variables de salida del modelo.

Nombre de la variable	Valor
BandwidthRefB	788.996
BandwidthTestB	57.2542
Total NMRB	1.14979
WinModDiff1B	53.2851
ADBB	3.05867
EHSB	0.145455
AvgModDiff1B	49.6598
AvgModDiff2B	30.9509
RmsNoiseLoudB	1.02666
MFPDB	1
RelDistFramesB	1
Objective Difference Grade	-2.460

BandwidthRefB y BandwidthTestB son las dos primeras variables de salida e indican el promedio lineal de la anchura de banda media de cada trama, una para la señal de referencia y la otra para la señal bajo prueba. Calcular su valor se obtiene la respuesta en frecuencia mediante la transformada rápida de Fourier y de ésta se desprende el ancho en frecuencia de la señal en ese bloque.

Total NMRB es la variable que indica que grado de enmascaramiento de la señal. El valor de esta variable se calcula como el promedio lineal de éste valor dentro de cada trama, el cual está calculado en decibeles.

La variable de salida del modelo WinModDiff1B es el promedio enventanado de la diferencia de modulación calculada a partir del modelo de oído basado en la transformada rápida de Fourier.

Las variables ADBB y MFPDB se refieren a la probabilidad de detección, la primera se enfoca en dar un número específico de tramas distorsionadas y perceptibles por el usuario a lo largo del archivo a estudiar.

La segunda toma en cuenta la respuesta de memoria del ser humano y pondera las distorsiones del archivo en base a eso; con ello nos da la probabilidad máxima filtrada (por la memoria humana) de detección.

La variable EHSB es la magnitud de la estructura armónica, la cual se mide identificando y midiendo el pico más grande en el espectro con la función de autocorrelación, la cual se calcula entre el vector de error y éste mismo vector con un retraso de cierta cantidad de tiempo, el cual depende de la tasa de muestreo y la transformada rápida de Fourier.

De esta manera se calcula la autocorrelación con retrasos desde cero hasta 256 muestras. Al vector resultante se le aplica una ventana de Hann normalizada y, tras excluir la componente de continua sustrayendo el valor medio, se calcula el espectro de potencia con una transformada rápida de Fourier. El valor de cresta máximo del espectro después del primer valle identifica la frecuencia dominante en la función de autocorrelación. El valor medio de este máximo a lo largo de las tramas multiplicado por 1000 es la estructura armónica del error o EHSB.

AvgModDiff1B y AvgModDiff2B se parecen a la variable de salida WinModDiff1B, solo que en éste caso el promedio es lineal en lugar de ser envetandado.

La diferencia existente entre las dos variables es el peso que se le da a las diferencias de modulación global, los cuales son valores fijos predispuestos por la recomendación.

RmsNoiseLoudB es la variable que reporta el comportamiento del ruido, dentro del cálculo de esta variable se toma en cuenta la sonoridad parcial de distorsiones aditivas en presencia de la señal de referencia enmascaradora. Para esto se calcula la media cuadrática de la sonoridad de ruido calculada a partir del modelo de oído basado en la transformada rápida de Fourier.

RelDistFramesB representa la fracción relativa de tramas para las cuales al menos una banda de frecuencias contiene un componente de ruido apreciable.

Algunas de las variables están promediadas de forma lineal, con media cuadrática o de manera enventanada. La descripción de cómo es calculada cada una se puede leer en el anexo 4.

Mediante el uso de estas variables podemos obtener un valor específico para conocer el comportamiento de la señal y evaluar de esta manera sus

cambios de modulación, la sonoridad de la distorsión, las distorsiones linéales, la frecuencia de las distorsiones audibles, la relación ruido enmascaramiento, la probabilidad de detección y finalmente la estructura armónica del error una vez ha sido procesada por el sistema de sonido planteado.

Finalmente tenemos el grado de diferencia objetiva como el resultado final del análisis, el cual toma en cuenta todos las variables de salida y nos da un resultado que puede oscilar entre un resultado excelente (0.22) y un resultado despreciable (-3.98).

Respecto a la velocidad de bits necesaria para la conexión del micrófono con la consola de audio, se obtuvo un valor de 144 kilobits por segundo.

4.3.1 ANÁLISIS DE RESULTADOS

En la anchura de banda media que reportada por el análisis de la información, se puede ver la pérdida sufrida por la baja tasa de muestreo del sistema, lo que genera que el ancho de banda se reduzca de 788 a 57.

El grado de enmascaramiento de la señal se ubica en un punto medio dentro de la escala a evaluar, ya que la escala va desde -24 hasta 16. Lo cual da un resultado aceptable (resultado = 1.14979).

Las variables WinModDiff1B, AvgModDiff1B y AvgModDiff2B indican los cambios en la modulación que el usuario relaciona con la aspereza del sonido. Estas tres variables se encuentran cerca del centro de los límites de salida, lo que indica que se tiene un resultado aceptable respecto a modulación.

Con relación a la probabilidad de detección del error, que se calcula con ADBB y MFPDB, los resultados son preocupantes, ya que estamos en el límite superior mostrado por la recomendación y esto indica que los errores van a ser fuertemente percibidos por el usuario final del producto, aunque la sonoridad de estos es baja (ver dos párrafos adelante).

Los errores que se registran tienen una estructura armónica baja (de nuevo por la baja tasa de muestreo), ya que se trabaja cerca de la cota inferior de 0.07 y muy lejos de la cota superior de 13.94 (resultado = 0.1454).

En cuanto a la sonoridad del ruido también se está muy bien posicionado, ya que la respuesta obtenida fue bastante baja y se está cerca de la cota inferior de 0.03, manteniéndose lejos de la cota superior de 14.82 (resultado = 1.027).

Respecto a la frecuencia de distorsiones audibles tenemos problema, ya que el valor calculado es el máximo presentado por la recomendación. Esto se debe a que todos los fragmentos tienen errores perceptibles por el usuario, estos errores son debidos a la tasa de muestreo baja presente durante toda la prueba y a las distorsiones que crea.

Finalmente tenemos el grado de diferencia objetiva, el cual evalúa todo el sistema. El resultado obtenido fue de -2.460.

Para poder darle un significado real a este número, es necesario utilizar el índice mostrado en la figura 4.1. Para poder adaptar el número a esa figura solamente es necesario sumar 5 unidades y buscar la calificación general del sistema.

De esta manera se obtiene un resultado de 2.54, lo que en la escala de calidad se demuestra que el sistema presenta un ruido ligeramente molesto.

En la tabla 4.2 se puede apreciar los resultados obtenidos comparados con el rango de resultados que son dados por la recomendación como límites superior e inferior.

Tabla 4.2: Resultados obtenidos y su comparación con los límites dados por la recomendación

Nombre de la variable	Valor obtenido	Recomendación		Resultado
		Mínimo	Máximo	
BandwidthRefB	788.996	-	-	-
BandwidthTestB	57.2542	-	-	-
Total NMRB	1.14979	-24.0451	16.2120	Aceptable
WinModDiff1B	53.2851	1.1106	107.1377	Aceptable
ADBB	3.05867	-0.2066	3.1563	Insuficiente
EHSB	0.145455	0.0743	13.9333	Excelente
AvgModDiff1B	49.6598	1.1136	63.2578	Aceptable
AvgModDiff2B	30.9509	0.9503	1145.018	Aceptable
RmsNoiseLoudB	1.02666	0.0299	14.8197	Excelente
MFPDB	1	0.0001	1	Insuficiente
RelDistFramesB	1	0	1	Insuficiente
Objective Difference Grade	-2.460	0.22	-3.98	Ruido ligeramente molesto

Debido a la tasa de bits medida por el sistema para la conexión del micrófono, la cual fue de 144 kbps, teóricamente es posible la conexión de más de 650 micrófonos de forma simultánea a la consola de sonido, siempre y cuando se utilice un enlace Fast Ethernet, el cual soporta velocidades de 100Mbps.

Al aumentar el número de conexiones y micrófonos conectados es necesario tomar como limitante la capacidad de procesamiento de la computadora donde la consola se esta ejecutando.

CONCLUSIONES Y RESULTADOS

Los objetivos planteados al inicio del proyecto se cumplieron cabalmente, ya que al finalizar el trabajo se tiene un sistema completo de audio, el cual maneja la información de forma digital de principio a fin.

Para la creación del sistema propuesto fue necesario el desarrollo un sistema de captura que se encarga de capturar, digitalizar y enviar la señal a la consola de sonido. El sistema de captura está compuesto por tres tarjetas, cada una de las cuales realiza una labor diferente.

En la primera etapa se trabajó con una tarjeta encargada de la amplificación de la señal, tarjeta que fue diseñada e implementada exclusivamente para este proyecto. Posteriormente se empleó un microcontrolador encargado de la conversión analógica a digital y del empaquetamiento de la información.

La última etapa del sistema de captura se implementó con una tarjeta de red que se encarga de enviar la información por medio de una red de área local hasta la consola de audio, la cual se ejecuta en una computadora de la misma red.

La consola de sonido se programó en Java para ser ejecutada en cualquier sistema operativo. La consola permite la interacción del usuario con el sistema completo por medio de una interfaz gráfica.

La consola de audio establece una conexión funcional entre la computadora y el micrófono por medio de un enlace UDP. La información recibida es desempaquetada e introducida al ecualizador, el cual se encarga de enfatizar ciertas bandas de frecuencia o de disminuir el nivel de otras.

El ecualizador se implementa por medio de tres filtros: paso bajas, paso banda y paso altas. Y se programaron para ser ejecutados en paralelo y al final se realiza la suma de la señal de salida de cada uno de ellos.

La salida del ecualizador es introducida en una línea de sonido, solicitada al sistema con anterioridad bajo un esquema de transmisión (streaming), para la reproducción de la señal a través de la tarjeta de sonido de la computadora.

Fue necesario conocer la calidad alcanzada por el sistema, para lo cual se implementó la recomendación UIT-R BS.1387-1, emitida por la ITU para determinar la calidad objetiva de los sistemas de audio.

Para la correcta implementación de la recomendación se desarrolló un programa en Matlab que compara la calidad de dos archivos de sonido, donde uno de ellos es un archivo de referencia y el otro es la grabación del mismo archivo transmitido a través del sistema. El resultado arroja un grado de diferencia objetiva de -2.460, lo que equivale a tener un ruido ligeramente molesto a la salida del sistema de acuerdo con el grado de diferencia subjetiva estipulado por la recomendación.

Cuando se tuvo el sistema completo funcionando, se procedió a realizar la medición de la tasa de bits necesaria para mantener un canal y con ello determinar el número de canales disponibles a través de la conexión existente entre la red y la computadora, en este caso la velocidad necesaria fue de 144 kbps y nos permite tener hasta 650 conexiones manejando una conexión Fast Ethernet.

El único problema registrado fue la incapacidad del conversor analógico a digital para trabajar a tasas de muestreo altas (se trabaja a 2406 Hz), lo que provocó la pérdida de calidad y distorsiones audibles en la señal, lo que dio como resultado que el sistema obtenga una baja calificación en la prueba de calidad objetiva.

La consola de sonido está preparada para manejar una tasa de muestreo de 44100 Hz, para poder introducir un conversor de mayor velocidad y obtener con ello una calidad de audio superior. La tasa de bits disponible en la red, la cual es de 100 Mbps también permite éste aumento de calidad.

REFERENCIAS

La Rotta Santos, Pedro Felipe. *Realización de un Sistema de Gestión de Audio Digital para Espectáculos.* México DF: Universidad Nacional Autónoma de México, 2012.

Coleman, Mark. *Playback: From the Vitrola to mp3, 100 Years of Musica, Machines and Money* . 2ª Edición. Cambridge: Da Capo Press, 2003.

Lucent Technologies. *Heritage*. 2000. http://www.bell-labs.com/org/1133/Heritage/Foil/ (último acceso: 7 de 6 de 2013).

Altenberg, Bert, Alex Clarke, y Philippe Mougin. *Become an Xcoder: Start Programming the Mac Using Objective-C*. Version 1.15. Cocoa Labs, 2008.

Ampex Corporation. *Ampex History*. 2011. http://www.ampex.com/early-history-ampex/214-early-history-ampex1.html (último acceso: 20 de Noviembre de 2011).

Apple Inc. *Final Cut Pro 7 User Manual.* 2010. http://documentation.apple.com/en/finalcutpro/usermanual/index.html#chapter=59%26section=2%26tasks=true (último acceso: 12 de Marzo de 2014).

Atmel Corporation. *8-bits AVR Microcontroller Datasheet.* 2009.

Ballaou, Glen. *Handbook for Sound Engineerd, The New Audio Cyclopedia.* 2ª edición. SAMS, a division of Macmillan Computer Publishing, 1991.

Benson, K. Blair. *Audio Engineering Handbook.* McGraw-Hill, 1988.

Eargle, John k. *Handbook of Recording Engineering.* 3ª edición. Chapman & Hall of Thomson Publishing, 1996.

Gan, Woon-Seng, y Sen M. Kuo. *Embedded Signal Processing with the Micro Signal Architecture*. New York: John Wiley & Sons, 2007.

Hamdy, Nadder. *Applied Signal Processing.* CRC Press, 2009.

Harman Inc. *Struder - [Struder at a Glance].* 2011. http://www.studer.ch/news/glance.aspx (último acceso: 20 de Noviembre de 2011).

IBS: The Organisation for Sound Professionals. *DSP - 1 - IBS Compendium*. 9 de Marzo de 2010. http://www.ibs.org.uk/audiocompendium/index.php?title=DSP-1 (último acceso: 20 de Noviembre de 2011).

International Telecommunication Union. *Recommendation BS.1387-1 (11/01)*. International Telecommunication Union, 2001.

Kahrs, Mark, y Karlheinz Brandenburg. *Applications of Digital Signal Processing to Audio and Acoustics.* Norwell, Massachusetts: Kluwer Academic Publishers, 1998.

Neve Electronics International. *DSP Digital Desk Brochure*. 1983.

Oracle Inc. *The Java Tutorials: Trail Sound*. 2013.

Peatman, John. *Desing with Microcontrollers.* McGraw-Hill, 1988.

Rossing, Thomas. *Springer Handbook of Acoustics.* New York: Springer Science+Business Media, 2007.

Smith, Julius Orion. *Digital Audio Resampling Home Page*. Editado por Center for Computer Research in Music and Acoustics (CCRMA). Stanford University, 2014.

SOUNDCRAFT HARMAN INTERNATIONAL INDUSTRIES LTD. *4-BUS & 8-BUS PROFESSIONAL MIXING CONSOLES.* 2010.

Texas Instruments Incorporated. *Differences between a DSP and Microcontroller (Micro-controller)*. 2006. http://www-k.ext.ti.com/srvs/cgi-bin/webcgi.exe?Company=%7b5761bcd8-11f5-4e08-84e0-8167176a4ed9%7d,kb=dsp,case=8521,new (último acceso: 25 de Febrero de 2013).

Texas Instruments. *Solid State Voice Recorder Using Flash MSP430.* Dallas, Texas: Texas Instruments, 2001.

Valdés Pérez, Fernando Eudaldo, y Ramon Pallàs-Areny. *Microcontrollers, Fundamentals and Applications with PIC*. CRC Press, 2009.

Watkinson, John. *The art of digital audio.* 3ª edición. Oxford: Focal Press, 2001.

WIZnet Co., Inc. *W5100 Datasheet Versión 1.2.4.* 2011.

Yamaha Pro Audio. *Mixers | Products | Yamaha* . 2011. http://www.yamahaproaudio.com/global/en/products/mixers/ (último acceso: 20 de Noviembre de 2011).

Zheng, Jia, Mengyao Zhu, y Yao Song. «On Objective Assessment of Audio Quality-A Review.» *Language and Image Processing (ICALIP), 2012 International Conference on Audio.* Shanghai: IEEE Xplore, 2012. 777-782.

ANEXO 1: RESPUESTA EN FRECUENCIA DEL CIRCUITO AMPLIFICADOR.

Con el fin de conocer la respuesta en frecuencia que presenta la tarjeta encargada de amplificar la señal del micrófono y adaptarla al conversor analógico digital del microcontrolador, se realizó el procedimiento descrito en la sección 2.3.

Los resultados obtenidos se muestran en la tabla A.1. Con base en estos resultados se obtuvo la figura 2.12, donde se puede observar el comportamiento en frecuencia del circuito.

Tabla A.1: Respuesta en frecuencia del circuito amplificador.

Frecuencia (Hz)	Voltaje Promedio (V)	Voltaje Promedio (dB)
20	1.505	3.550729999
30	1.64	4.296876961
40	1.73	4.760922063
50	1.745	4.835908626
60	1.745	4.835908626
70	1.83	5.249021795
80	1.85	5.343434568
90	1.83	5.249021795
100	1.83	5.249021795
150	1.76	4.910253356
200	1.77	4.959465327
250	1.76	4.910253356
300	1.745	4.835908626
350	1.745	4.835908626
400	1.73	4.760922063
450	1.73	4.760922063
500	1.71	4.659922208
550	1.71	4.659922208
600	1.685	4.531998104
650	1.66	4.402161761
700	1.66	4.402161761
750	1.64	4.296876961

800	1.625	4.217067306
850	1.61	4.136517521
900	1.59	4.027942486
950	1.565	3.890286838
1000	1.54	3.750414417
1500	1.35	2.60667537
2000	1.135	1.099917231
2500	0.99	-0.087296108
3000	0.83	-1.618438152
3500	0.684	-3.298877966
4000	0.576	-4.791550332
4500	0.498	-6.055413145
5000	0.432	-7.290325064
5500	0.378	-8.450164003
6000	0.338	-9.421665994
6500	0.292	-10.69234297
7000	0.26	-11.70053304
7500	0.232	-12.6902403
8000	0.21	-13.55561411
8500	0.187	-14.56316787
9000	0.173	-15.23907794
9500	0.157	-16.08200695
10000	0.146	-16.71294288
11000	0.121	-18.34429259
12000	0.134	-17.45790403
13000	0.093	-20.63034103
14000	0.088	-21.11034656
15000	0.076	-22.38372815
16000	0.072	-22.85335007
17000	0.064	-23.87640052
18000	0.06	-24.43697499
19000	0.05	-26.02059991
20000	0.048	-26.37517525

Con el fin de tener al microcontrolador trabajando de manera correcta, se creó el siguiente código, el cual se encarga de establecer una conexión con la consola de audio y posteriormente envía la información leída del convertidor analógico digital de manera ininterrumpida.

Para el envío de la información emplea paquetes UDP con una muestra contenida en cada uno. Para poder empaquetar un mayor número de muestras es necesario cambiar la última parte del programa.

```
/*
  Programa original y completo para el envio de
  la informacion del microfono al programa
  de java por medio de UDP.

  Por: Pedro La Rotta (pedro.lars@gmail.com)

*/

#include <SPI.h>
#include <Ethernet.h>
#include <EthernetUdp.h>

// Una instancia EthernetUDP la cual nos deja enviar y recibir
paquetes sobre el protocolo UDP
EthernetUDP Udp;

// Se configura la MAC, la direccion IP y el puerto para enviar
byte mac[] = { 0x90, 0xA2, 0xDA, 0x0E, 0x07, 0xE4 };
IPAddress ipLocal(192,168,1,88);    //Direccion IP local
unsigned int localPort = 8888;      //Puerto local para escuchar
IPAddress ipRemota = (0,0,0,0);


void setup() {
  // Iniciamos Ethernet y UDP:
  Ethernet.begin(mac,ipLocal);
  Udp.begin(localPort);
  Serial.begin(9600);

  //Se obtiene la direccion del servidor
  Serial.println("Configurando direcciones");
  int i = 0;
  while (i==0){
  int packetSize = Udp.parsePacket();
  if(packetSize)
  {
    Serial.print("Received packet of size ");
```

```
    Serial.println(packetSize);
    Serial.print("From ");
    ipRemota = Udp.remoteIP();
    for (int i =0; i < 4; i++)
    {
      Serial.print(ipRemota[i], DEC);
      if (i < 3)
      {
        Serial.print(".");
      }
    }
    Serial.print(", port ");
    Serial.println(Udp.remotePort());

    i = 1;
  }
  }

}

//Se envía la información a la consola de audio
void loop() {
  //int valorSensor = analogRead(A0);
  Udp.beginPacket(ipRemota,localPort);
  //Udp.print(valorSensor);
  Udp.print(analogRead(A0));
  Udp.endPacket();
}
```

Con el fin de cubrir las necesidades de la consola de audio, tanto la parte gráfica como el soporte programado para el manejo de audio, se desarrolló el siguiente código de Java.

Dado que el código generado para la clase principal es extenso (388 líneas) aquí solo se muestra el código desarrollado en la clase principal para los métodos correspondientes a la reproducción del audio proveniente del micrófono y el ecualizador necesario para operar la consola.

El código para reproducir el sonido proveniente del micrófono es:

```
private static void microAudio(){
        //Algunas variables
        SourceDataLine linSon = null;
        int Buffer = 20;
        byte[] sale = new byte[Buffer];

        //Obtenemos direeción del microfono
        obtenDir(8888);

        //Preparamos el formato de audio
        AudioFormat formaAudio = new AudioFormat
(2406,16,1,true,false);

        //Obtenemos una linea de sonido
        try{
            DataLine.Info info = new DataLine.Info
(SourceDataLine.class, formaAudio);
            //prueba
            AudioFormat[] prueba = (info.getFormats());
            for (int i = 0; i < prueba.length; i++){
                System.out.println(prueba[i].toString());
            }


            linSon = (SourceDataLine) AudioSystem.getLine(info);
            linSon.open(formaAudio);
            linSon.start();
        }
        catch (LineUnavailableException ex) {
            ex.printStackTrace();
        }

        //Obtenemos los controles sobre la linea de sonido
```

```
        vol2 = (FloatControl)
linSon.getControl(FloatControl.Type.MASTER_GAIN);
        pan2 = (FloatControl)
linSon.getControl(FloatControl.Type.PAN);

        //Obtenemos la información del microfono y la pegamos en
la linea de sonido
        try{
            DatagramPacket paquete = new DatagramPacket(new
byte[30],30);
            DatagramSocket socCliente = new DatagramSocket(8888);
            boolean preparado = false;
            System.out.println("Preparado para recibir");

            while (! preparado) {

                socCliente.receive(paquete);
                //String llega = new String(paquete.getData());
                sale = convierte(paquete.getData());
                linSon.write(sale, 0, Buffer);

            }

            socCliente.close();

        }
        catch(Exception e) {
            System.out.println(e);
        }

        finally {
            linSon.drain();
            linSon.close();
        }
    }
```

El código encargado de realizar la ecualización es:

```
private static int ecualiza2k (int actual){
        //Se programa el filtro paso bajas:
        v[0][0] = (actual*coef2k[0][0]) - (v[0][1]*coef2k[0][1]) -
(v[0][2]*coef2k[0][2]);
        y[0] = v[0][0] + (2*v[0][1]) + v[0][2];
        v[0][2] = v[0][1];
        v[0][1] = v[0][0];

        //Se programa el filtro paso banda:
        v[1][0] = (actual*coef2k[1][0]) - (v[1][1]*coef2k[1][1]) -
(v[1][2]*coef2k[1][2]);
        y[1] = v[1][0] - v[1][2];
        v[1][2] = v[1][1];
```

```
        v[1][1] = v[1][0];

        //Se programa el filtro paso altas:
        v[2][0] = (actual*coef2k[2][0]) - (v[2][1]*coef2k[2][1]) -
(v[2][2]*coef2k[2][2]);
        y[2] = v[2][0] - (2*v[2][1]) + v[2][2];
        v[2][2] = v[2][1];
        v[2][1] = v[2][0];

        //Se unen las diferentes respuestas tomando en
consideración las ganancias
        int              salida              =              (int)
((y[0]*eqParam[1][0])+(y[1]*eqParam[1][1])+(y[2]*eqParam[1][2]));
        return salida;
    }
```

Para poder obtener algunas de las variables de salida del sistema de comparación de la calidad del audio, es necesario realizar el promediado de los valores obtenidos en todos los fragmentos del archivo de sonido.

En algunos casos el promediado se realiza de forma lineal, en este caso se emplea la ecuación A.1 para obtener el valor de la variable de salida.

$$AvgX = \frac{1}{N} \cdot \sum_{n=0}^{N-1} X[n] \tag{A.1}$$

En este caso X es el nombre de la variable de salida deseada y N es el número de muestras temporales para las cuales se han calculado los valores momentáneos de X.

Cuando se requiere la media cuadrática de los valores, se necesita aplicar la fórmula A.2. Este valor también se conoce como RMS por sus siglas en inglés.

$$RmsX = \sqrt{\frac{1}{N} \cdot \sum_{n=0}^{N-1} X[n]^2} \tag{A.2}$$

Para poder obtener el valor numérico de este promedio, las variables representan los mismos valores que en la ecuación A.1.

Para poder obtener el promedio enventanado de las diferentes muestras del archivo, es necesaria la aplicación de la formula A.3. En este caso las variables van a tener el prefijo *Win*.

$$WinX = \sqrt{\frac{1}{N-L+1} \cdot \sum_{n=L-1}^{N-1} \left(\frac{1}{L} \cdot \sum_{i=0}^{L-1} \sqrt{X[n-1]} \right)^4} \tag{A.3}$$

Al obtener el valor numérico de la variable es necesario saber que X es el nombre de la variable de salida del modelo, N es el número de muestras temporales para las cuales se han calculado valores momentáneos de X y

L es la longitud de la ventana temporal deslizante en las muestras temporales.

La longitud L es fijada por la recomendación, la cual estipula su duración a 100 ms, de tal manera que para versión básica, la cual se basa en la transformada rápida de Fourier y fue implementada durante el desarrollo de la tesis, el valor de L es de 4. Para la versión avanzada, la cual se basa en un banco de filtros, el valor de L es de 25.