DISEÑO E IMPLEMENTACIÓN DE UN PEDAL MULTI-EFECTO PARA GUITARRA ELÉCTRICA

DESIGN AND IMPLEMENTATION OF A MULTI-EFFECT PEDAL FOR ELECTRIC GUITAR

Jorge Álvaro Martínez González Robles
Universidad Autónoma Metropolitana, México
maztergeorge@gmail.com

Javier Alducin Castillo
Universidad Autónoma Metropolitana, México
jac@azc.uam.mx

Recepción: 3/noviembre/2021 **Aceptación:** 7/marzo/2022

Resumen

El diseño e implementación de un pedal multi-efectos de sonido para guitarra eléctrica, es una forma interesante, de crear diferentes efectos acordes al gusto de cada músico y sin estar limitado sistemas comerciales que presentan diversas desventajas como son: portabilidad, costo, configuración, difícil uso, entre otras. En este proyecto se construyó un pedal multi-efectos que combina el uso de hardware (Raspberry) y software libre (Pure Data y código en C), por lo que el pedal es modificable, al tener acceso a su código fuente y su hardware. Se implementan exitosamente efectos de sonido como son: eco, reverberación, overdrive, distorsión, así como un ecualizador de tres bandas analógico. El pedal implementado, brinda de posibilidad de elegir el efecto o los efectos deseados usando botones físicos o a través de una interfaz gráfica (GUI). Otra ventaja es su portabilidad, ya que, al usar un sistema de cómputo en miniatura, es posible transportarlo fácilmente. Por ello, es posible implementar un pedal multi-efectos con una calidad de sonido adecuada. Sin embargo, dicha calidad es mejorable al utilizar un ADC con mayor resolución al utilizado en este trabajo.

Palabras Clave: Efectos de sonido, pedal de efecto, procesamiento de audio, pure data, Raspberry.

Abstract

The design and implementation of a multi-effects sound pedal for electric guitar is an interesting way to create different effects according to the taste of each musician and without being limited to commercial systems that have several disadvantages such as: portability, cost, configuration, difficult to use. In this paper a multi-effects pedal was designed and implemented with hardware (Raspberry) and free software (Pure Data and C code), so the pedal is modifiable, having access to its source code and hardware. Sound effects such as echo, reverb, overdrive, distortion, as well as an analog three-band equalizer are successfully implemented. The implemented pedal provides the possibility to choose the desired effect(s) using physical buttons or through a graphical user interface (GUI). Another advantage is its portability, since, by using a miniature computer system, it can be easily transported. Therefore, it is possible to implement a multi-effects pedal with adequate sound quality. However, this quality can be improved by using an ADC with higher resolution than the one used in this work.

Keywords**:** *Audio effects, audio processing, effects pedal, pure data, Raspberry.*

1. Introducción

Un sistema de procesamiento de señales digital es relativamente sencillo de implementar, modificar, corregir y actualizar en su funcionamiento. Pues basta con reprogramar el programa principal, que permite la ejecución del sistema. Particularmente la ejecución de efectos de guitarra usando sistemas digitales, permite tener múltiples efectos en un solo sistema. Además, es posible contar con características y parámetros ajustables.

Si bien, los sistemas digitales brindan muchas ventajas con respecto a los sistemas analógicos, es claro que los primeros requieren aún de un acondicionamiento analógico. En muchos casos será necesario procesar las señales con etapas analógicas. Convertir las señales a digitales usando, convertidores Analógicos-Digitales (ADC). Y una vez que se tienen las señales digitales, se procesan de forma digital. La señal procesada pasa por un convertidor Digital-Analógico (DAC), y se entrega la información de forma analógica.

Las señales de audio se pueden modificar a través de una serie de parámetros de un sistema digital, de acuerdo con el criterio del usuario; la configuración de los parámetros produce el efecto de sonido deseado. Para poder introducir un efecto de sonido en tiempo real, se utilizan *pedales* principalmente para modificar el sonido de una guitarra. Un pedal de efectos es un dispositivo electrónico capaz de modificar el sonido de una fuente. Dicha fuente puede ser un instrumento musical e inclusive la voz. Este dispositivo se coloca en el piso y funciona mediante el accionar de un botón o un interruptor. Consiguiendo que el músico tenga las manos libres, para tocar su instrumento y lo utilice en cualquier momento utilizando sus pies. Actualmente existen dos tipos de pedales: analógicos y digitales. Los primeros son aquellos que utilizan componentes analógicos y métodos electromecánicos para procesar la señal. Los pedales digitales hacen uso del procesamiento digital.

Es bien sabido que el espectro audible va desde los 20 Hz hasta los 20 kHz, siendo un espectro tan amplio, es necesario contar con herramientas capaces de modificar las frecuencias producidas por un instrumento musical. Obteniendo así un sonido diferente y agradable para el músico y la audiencia. Sin embargo, siempre se busca un sonido propio, que, combinado con la técnica de ejecución, el resultado sea llamativo para todo aquel que esté escuchando al músico, y de esta manera sea asociado ese sonido con el músico. El contar con herramientas como los pedales multi-efectos es bastante útil para modificar el rango audible. De esta manera el músico puede adquirir el sonido que desea.

El diseño e implementación de pedales para guitarra tiene principalmente dos vertientes: pedales comerciales y trabajos aficionados. Por ello, en la revisión de la literatura se realizarán, principalmente, con trabajos comerciales.

En [ElectroSmash, 2020a] se desarrolló un pedal multi efecto programable, usando una ventana de comandos (texto) con una Raspberry Pi. Su construcción física es muy atractiva debido a su portabilidad. Sin embargo, es incapaz de emplear más de un efecto al mismo tiempo, y en caso de querer usar uno diferente es necesario modificarlo manualmente utilizando los botones que trae incorporados. Otra desventaja, se debe a que los botones son pequeños, esto vuelve complicado su utilidad a la hora de una presentación en vivo, ya que se requiere una forma versátil

de poder cambiar un efecto de un momento a otro. Adicionalmente, su tamaño es pequeño, esto dificulta su utilidad debido a que el músico requeriría una precisión para presionar el botón sin mirarlo. Los parámetros modificables de este pedal son únicamente volumen y cantidad de efecto mediante el pulsar de dos botones: uno para bajar y otro para subir volumen.

La plataforma Arduino también ha sido utilizada para el desarrollo de pedales comerciales. En [ElectroSmash, 2020b], se desarrolló un pedal a partir del IDE de Arduino. Para implementar modificaciones, en dicho sistema, el usuario requiere un cable con conector tipo B. Incorpora una pequeña pantalla OLED, donde se pueden visualizar algunos parámetros como la cantidad de efecto, el tiempo de muestreo entre otros. Las desventajas siguen siendo las mismas que [ElectroSmash, 2020a]. El pedal es incapaz de emplear más de un efecto al mismo tiempo y para elegir otro efecto es necesario modificar manualmente utilizando los botones que vienen incorporados, complicando su utilidad. Cabe destacar que la pantalla es de un par de centímetros lo que complica la visualización los parámetros del efecto. Los parámetros modificables de este pedal son únicamente volumen y cantidad de efecto mediante el pulsar de dos botones y la personalización de la pantalla. Otro trabajo basado en Arduino [Instructables, 2020], tiene un diseño más profesional que [ElectroSmash, 2020a]; [ElectroSmash, 2020b] aunque tiene un parecido con [Boss, 2020]. Este pedal incorpora una carcasa de metal, dándole mejor robustez. En el interior tiene un material aislante para evitar interferencias. La única desventaja es que sólo tiene programado un efecto de distorsión. Nuevamente los parámetros modificables de este pedal son únicamente volumen y tonos, la modificación de estos parámetros se realiza mediante el uso de perillas.

La marca comercial BOSS desarrolló pedales de efecto analógicos [Boss, 2020]. Pueden variar en color y tamaño, entre sus ventajas se encuentran que son portables, tienen una carcasa de metal lo cual los hace robustos, y su control de perillas es bastante intuitivo, por lo que no es necesario ser un experto para utilizar un pedal de esta marca. Lamentablemente no todos los efectos son tan asequibles, cabe recordar que un pedal analógico de esta marca ronda entre los $1000 y $4000 pesos mexicanos. Otra desventaja es que, en caso de querer otro efecto, se debe

adquirir por separado lo que implicar un costo mayor. Los parámetros modificables únicamente son volumen y tonos. La modificación de estos parámetros se realiza mediante el uso de perillas.

Otro pedal comercial es desarrollado por la marca [Mooer, 2020]. Entre sus ventajas se encuentra una gran cantidad de efectos. Incorpora una caja de ritmos, simuladores de amplificadores, una pantalla LCD con una interfaz gráfica muy sencilla y la capacidad de emplear más de un efecto a la vez, entre otras funciones. Una de las desventajas es que incorpora dos botones tipo foot switch. Estos botones son multitareas lo que complica el uso de sus múltiples efectos. También incorpora una perilla multitareas con la cual puedes modificar ciertos parámetros del pedal como: el brillo de la pantalla, los valores de los efectos pregrabados. El diseño brinda una perilla para navegar en el control del ecualizador de cada efecto. El utilizar este pedal requiere tener los efectos preajustados previamente, de lo contrario el ajuste en vivo es impráctico.

Dadas las ventajas y desventajas previamente mencionadas, en este trabajo se propone implementar un pedal para guitarra con varias etapas. i) Se propone incorporar una etapa analógica de diseño propio, para acondicionar la señal de audio: filtrado y amplificación. ii) Se implementará un sistema de conversión AD. iii) Se procesará la señal digital, incorporando los efectos elegidos por el usuario. Podrá seleccionar los efectos través de botones, de fácil acceso. También contará con la capacidad de aplicar varios efectos a la vez. Esta última característica es un gran diferenciador de los pedales comerciales mencionados. Los efectos que se propuestos son: eco, distorsión, overdrive y reverberación. Adicional, se proponer diseñar una interfaz gráfica, que ayude al usuario a controlar el sistema de procesamiento. iv) El resultado del procesamiento, se entregará a una etapa de conversión DA, para poder reproducir el audio resultante.

Eco

El efecto eco se crea mediante la alteración de la señal original produciéndole un retardo en el tiempo. Esta señal retardada es mezclada con la original, el retardo puede variar entre 10-25 ms [Zölzer, 2002]. En la figura 1, se muestra el diagrama

de bloques que modela el efecto eco y la ecuación 1 describe la ecuación de diferencias de dicho efecto. Donde del son las muestras de retardo y g la atenuación.

$$salida[n] = entrada[n] + g * entrada[n - del] \quad (1)$$

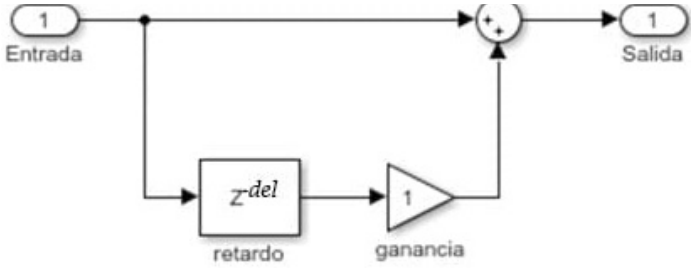


Figura 1 Diagrama a bloques del efecto eco.

Reverberación

El efecto de reverberación se caracteriza por tener tres componentes dependientes del tiempo: *i*) señal original, *ii*) reflexiones tempranas que se producen en un tiempo corto y *iii*) reflexiones tardías. Estas últimas, son aquellas reflexiones provocadas por el rebote de las ondas de propagación del sonido original, que tarda más tiempo en llegar al destino. Los tres tipos de reflexiones se pueden observar en la figura 2. El tiempo de retardo de cada reflexión dependerá del lugar que se desea *simular*, y de los coeficientes de reflexión de las superficies que se deseen modelar. Existen diversos modelos de un sistema de reverberación, sin embargo, el más sencillo es el filtro peine IIR. Por ello, en este trabajo se utilizará una estructura de tres etapas de filtros peine IIR. En la figura 3, se puede observar dicha estructura.

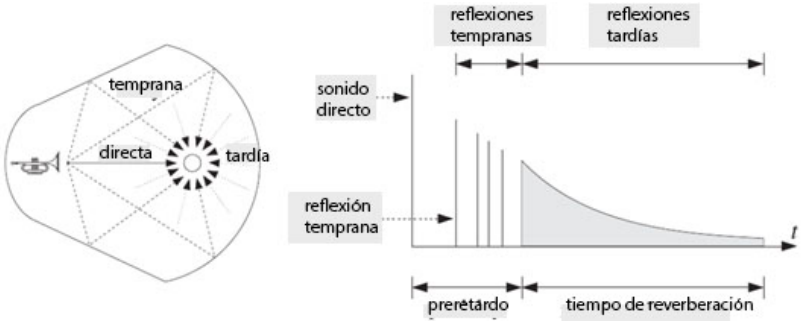


Figura 2 Reflexiones tempranas y tardías [Schroeder, 1962].

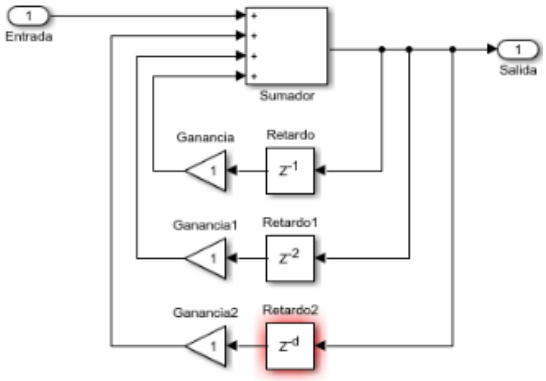


Figura 3 Diagrama a bloques de la estructura de reverberación implementada.

Overdrive

Este efecto se consigue al llevar al límite de corte y saturación la amplitud de la señal de entrada. Esto ocasiona que en los semiciclos positivos y negativos haya recorte de la señal, a ese recorte se le denomina *soft clipping*. Este recorte no se genera de manera brusca, produciendo un sonido más lleno y cálido. En la figura 4, se muestra gráficamente el *soft clipping* a una señal senoidal.

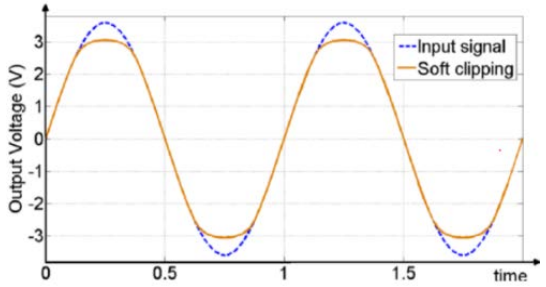


Figura 4 Soft clipping de una señal [Huffenus, 2015].

Distorsión

Este efecto se produce de manera similar al anterior debido a que son de la misma familia. Se produce al generar un recorte más agresivo llamado *hard clipping*. El resultado es una señal casi cuadrada. Al tener ese recorte brusco, se produce un sonido más sucio y pesado. En la figura 5, se muestra una comparación gráfica entre el efecto overdrive y distorsión. La señal original se recorta en ambos efectos, sin embargo, el recorte en el efecto distorsión es más severo al convertir la señal en una onda cuadrada.

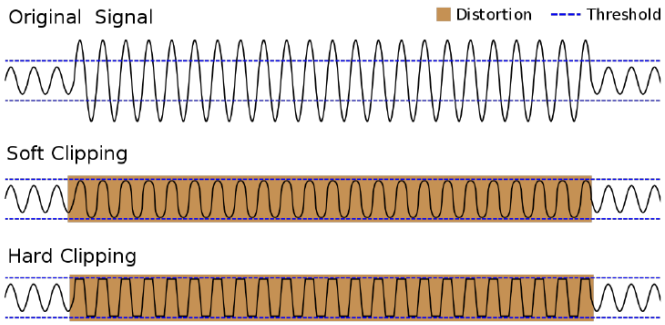


Figura 5 Comparación entre el efecto overdrive y distorsión [Duxans, 2012].

Software Pure Data

Pure Data (Pd) es un entorno/lenguaje de programación gráfica de código abierto, dirigido principalmente hacia el procesamiento digital de audio. La programación en este lenguaje se realiza de manera visual interconectando objetos. Cada objeto puede contar con una o varias entradas y/o salidas, dependiendo de las necesidades del sistema. Los programas realizados en este entorno son mejor conocidos como *patches* o abstracciones. Existe la posibilidad de hacer subprogramas que se encuentran dentro del programa principal, estos son conocidos como *subpatches* o substracciones. Los patches generados, se pueden implementar en sistemas de cómputo miniatura como lo es Raspberry. De esta forma se puede combinar sistemas de software y hardware para crear sistemas de procesamiento de audio.

2. Métodos

El trabajo presentado se divide en varias etapas. En la figura 6, se presenta la secuencia de etapas del sistema desarrollado y se describen a continuación.

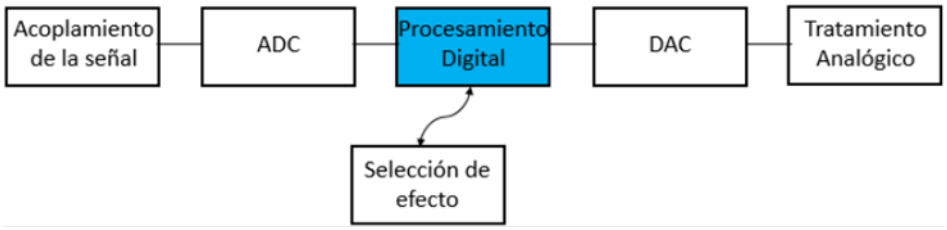


Figura 6 Diagrama a bloques del sistema desarrollado.

Acoplamiento de la señal

El bloque del Acoplamiento de la señal está compuesto por un circuito de diseño propio, para asegurar, una mejor transmisión, sin pérdidas de la señal. En la figura 7, se puede observar el diseño realizado. Se toma en cuenta la impedancia de entrada del OPAMP, y la impedancia de una guitarra tipo Les Paul con pastillas PAF Alnico 2, con un voltaje de 127 mV y una impedancia de 9.5 kΩ.

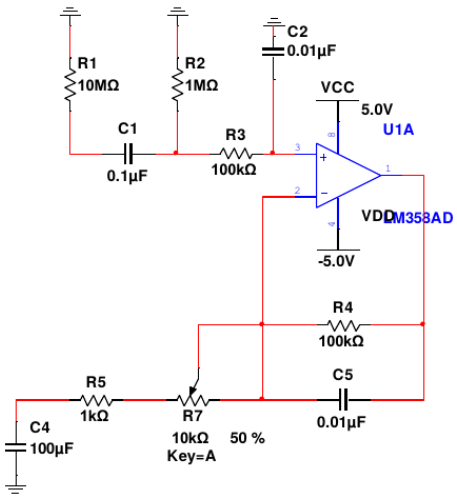


Figura 7 Diagrama del circuito acoplador de impedancias.

Para asegurar que el sonido proveniente del instrumento musical llegué a la salida, se requiere un circuito que sea capaz de mantener el voltaje sin tener pérdidas. Para ello, se utilizan los acopladores de impedancia o mejor conocidos como seguidores de voltaje o buffers. Además, se incluyeron dos filtros RC para limpiar la señal. Y un amplificador que amplifica la señal hasta 3.3 V para que el ADC sea capaz de leer la señal.

ADC

Para introducir la señal analógica proveniente de la guitarra eléctrica, es necesario convertir la señal analógica a digital, mediante un ADC. Se eligió un ADC con matrícula MCP3008. Este circuito integrado es controlado por el bus SPI. Algunas de las características más destacables de este integrado son: voltajes de alimentación de 2.5 - 5 V, resolución de 10 bits y una velocidad de transmisión de

200 kbps. Este circuito ADC se interconecta con un Raspberry Pi Zero, quien recibirá la señal digital. La estructura del código de este bloque se muestra en el siguiente pseudocódigo:

```
#include <bcm2835.h>
#include <stdio.h>
int main(int argc, char **argv)
{
   // Inicia la librería BCM2835 para controlar GPIOS
   if (!bcm2835_init())
   {
     printf("bcm2835_init failed. Estás ejecutando como root??\n");
     return 1;
   }
// Inicializa SPI BUS.
if (!bcm2835_spi_begin())
   {
     printf("bcm2835_spi_begin failed. Estás ejecutando como root??\n");
     return 1;
   }
//define la configuración del SPI bus
   bcm2835_spi_setBitOrder(BCM2835_SPI_BIT_ORDER_MSBFIRST);      // The default
   bcm2835_spi_setDataMode(BCM2835_SPI_MODE0);                   // The default
   bcm2835_spi_setClockDivider(BCM2835_SPI_CLOCK_DIVIDER_64);          //     4MHz
clock with _64
   bcm2835_spi_chipSelect(BCM2835_SPI_CS0);                      // The default
   bcm2835_spi_setChipSelectPolarity(BCM2835_SPI_CS0, LOW);      // the default
         uint8_t mosi[10] = { 0x01, 0x00, 0x00 }; // bits ADC
uint8_t miso[10] = { 0 };
while(1) //Main Loop
{
//lectura del ADC
   bcm2835_spi_transfernb(mosi, miso, 3);
   input_signal = miso[2] + ((miso[1] & 0x0F) << 8);
//**** Deja pasar toda la señal  ***///
   }
//close all and exit
bcm2835_spi_end();
   bcm2835_close();
   return 0;
}
```

Inicialmente, se cargan las librerías para controlar los GPIOS de la Raspberry pi, una vez activados estos pines tiene la capacidad de generar una interfaz entre el Raspberry y el mundo exterior. Posteriormente, se inicializa y se configura el BUS SPI para generar la transferencia de información entre el circuito ADC y el Raspberry. Finalmente, comienza la lectura del ADC, en esta parte únicamente hacemos la lectura en tiempo real de la señal para posteriormente codificarla por el software Pure Data. El diagrama de conexión entre el ADC y la Raspberry se puede observar a detalle en la figura 8.

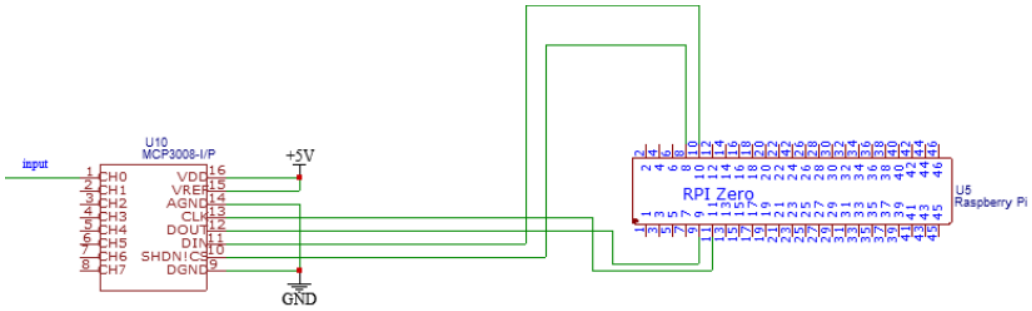


Figura 8 Diagrama de conexión entre el ADC con el Raspberry PI.

Procesamiento digital

Una vez realizada la etapa de conversión AD, se procedió al procesamiento digital de las señales. En este bloque se realiza la introducción de los efectos de sonido, elegidos por el usuario y el sistema que controlará el pedal. Se utiliza un sistema Raspberry para modificar el audio original. En PD, se crearon los *patches*, de manera individual, para asegurar el correcto funcionamiento de cada uno de los efectos. Así, se pueden hacer de manera limpia y ordenada. Además, se creó una de la interfaz gráfica que engloba todos los patches generados por cada efecto mediante subpatches, está interfaz se puede observar en la figura 9.

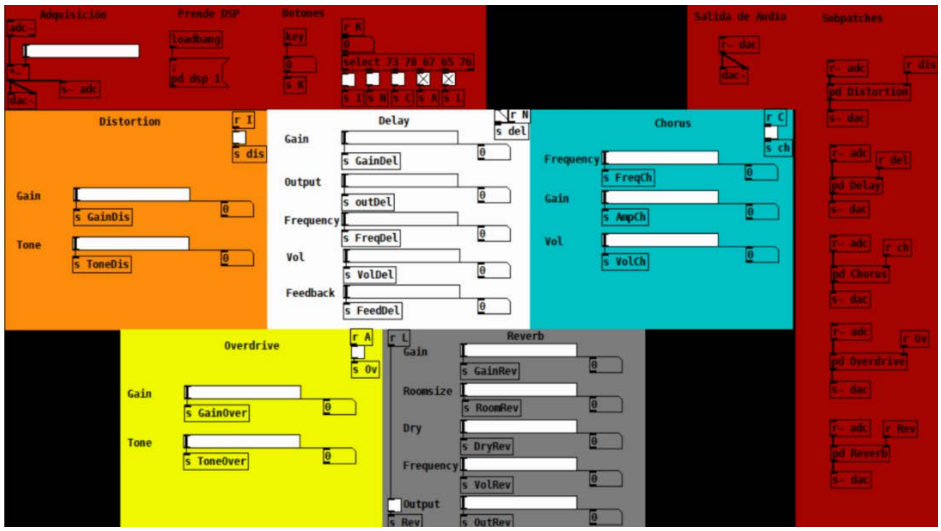


Figura 9 Interfaz gráfica propuesta para el control de los efectos de sonido.

Se implementó que el usuario pueda controlar los diversos parámetros, de cada efecto, a través de dicha interfaz gráfica. Para este propósito se incluyen: paneles,

botones y barras deslizadoras. Esto le brinda al usuario una interfaz gráfica funcional, ordenada y atractiva visualmente. A continuación, se detalla cómo se implementaron los diversos efectos propuestos. En cada efecto, por comodidad se usaron los nombres de los objetos en inglés.

Eco

La implementación del efecto eco, patch en Pd, se describe a continuación y el diagrama de conexión se puede observar en la figura 10. Se crea el objeto adc~ que adquiere la señal proveniente del instrumento. Posteriormente se guardan 5 segundos de la señal de entrada en un buffer (delwrite~ audio-buffer solo genera el retraso 5000).

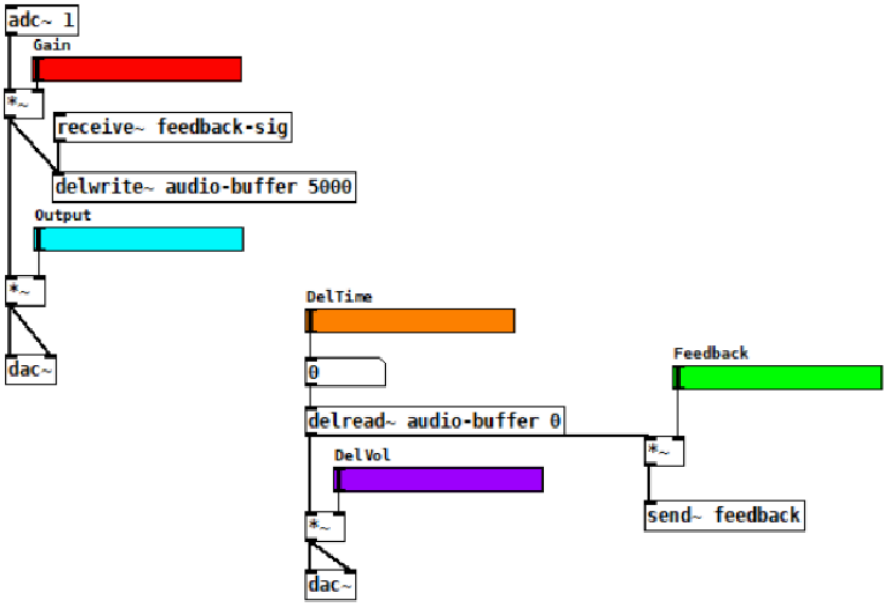


Figura 10 Patch realizado para el efecto de eco.

Una vez guardada la señal en el buffer es necesario volver a leerla, para ello se crea el objeto delread~. Al buffer, se le agrega un tiempo de lectura variable, que se controla con la barra naranja. Esta hace que la lectura sea más rápida o lenta. Para controlar el volumen del efecto, se utiliza una barra (morada). Para obtener múltiples lecturas se agrega el objeto feedback (barra verde). Este recibe la señal modificada del buffer, le agrega una ganancia y la vuelve enviar al buffer con el objeto send~.

El buffer recibe esta señal con el objeto recieve~. De manera resumida, este efecto nos permite obtener la señal de entrada, mezclada con el retardo seleccionado (barra naranja) y con el segundo retardo producido por la barra verde, retroalimentado al mismo buffer.

Reverberación

La implementación del patch para el efecto reverberador se describe a continuación y puede observarse a detalle en la figura 11.

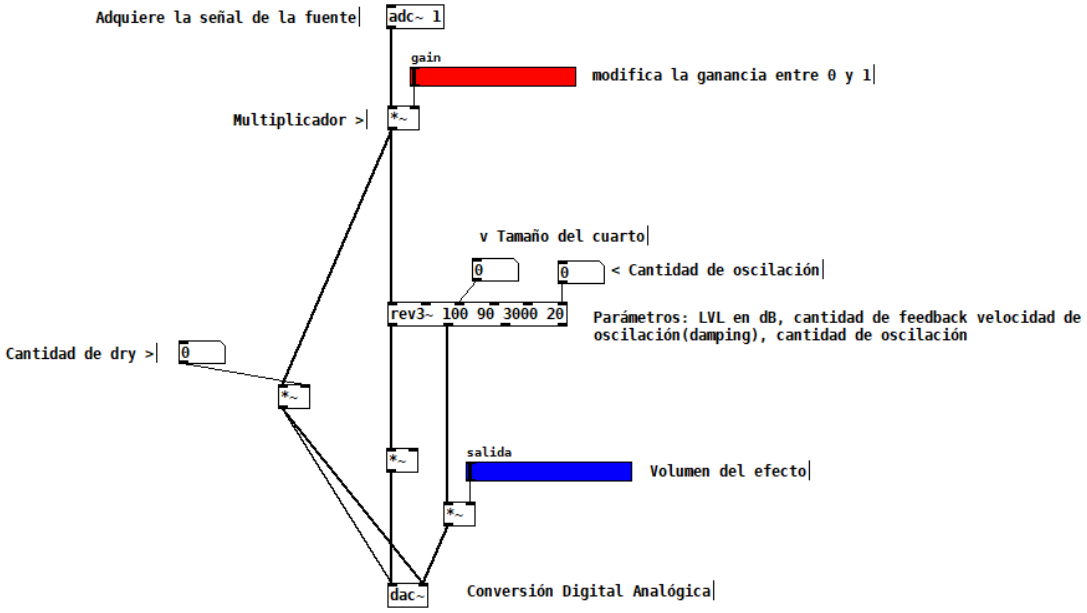


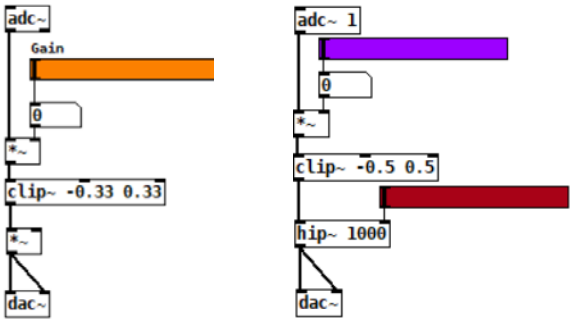
Figura 11 Patch realizado para el efecto de reverberación.

El objeto adc~ 1 adquiere la señal de audio proveniente de la conversión AD. La señal ingresa a un multiplicador que, al mismo tiempo puede ser ponderada por una ganancia variable (barra roja). Esta ganancia, puede tomar valores entre 0 y 1. Se utilizó el objeto rev3~ debido a que este proporciona una mejor calidad en el audio a diferencia de rev1~ y rev2~. Los argumentos del objeto rev3~ son los siguientes: *i*) salida (Output) es la magnitud en dB, se encuentra entre 0 y 100. Se eligió la máxima cantidad para un mayor volumen. *ii)* Tamaño del cuarto (Liveness), con este parámetro se puede simular el tamaño del cuarto, entre más grande el cuarto se tiene mayor cantidad de reflexiones dándole más "color" al efecto. Este parámetro

puede ir de 0 a 100, sin embargo, usualmente se usa en el rango de 85-100, en este caso se usó un valor intermedio. *iii*) Frecuencia de corte (Crossover Frequency), esta frecuencia de corte funciona similar al de un filtro cualquiera, esta nos delimita el rango de frecuencias al que podrá funcionar nuestro efecto. Las unidades de este efecto son los Hz. *iv*) Factor de amortiguamiento (Damping), este factor nos permite obtener una cantidad de oscilaciones, esto se puede traducir en la "duración" del efecto. Entre mayor sea el factor de amortiguamiento menos cantidad de efecto se tendrá. Este valor puede ir de 0 a 100. Por otro lado, se tiene el objeto dry, este nos permite tener un balance entre la reverberación de salida con respecto a la de entrada, entre mayor sea la cantidad de dry a la entrada, menor cantidad de reverb a la salida. Este valor puede ir de 0 a 1 dónde 0 equivale a tener el control del reverb a la salida y conforme va aumentando este valor vamos reduciendo el efecto. Finalmente, todos estos factores, llegan al dac~ que es el encargado de hacer sonar el efecto.

Overdrive

Se generó un patch, figura 12a, para el efecto overdrive donde la señal de entrada es adquirida por el adc~ que entra a un multiplicador de una ganancia entre un rango de 0 a 50. Se definió este rango debido a que, si el pedal es utilizado en un estudio, al usar unos audífonos puede llegar a dañar la audición.



a) Patch efecto overdrive *b) Patch efecto distorsión*

Figura 12 Patchs con la misma estructura, se cambia el rango de recorte.

Posteriormente se realizó un soft clipping de la señal en un rango de -0.33 y 0.33, este rango se obtuvo de forma heurística.

Distorsión

Para realizar este efecto se utilizó una estructura similar del patch del efecto overdrive, variando el rango de recorte de la señal a un hard clipping, figura 12b. El rango de recorte de –0.5 a 0.5 fue elegido heurísticamente, ya que en este rango se adquiere un sonido sucio sin llegar a serlo demasiado. Posteriormente se agregó un filtro pasa altas con una frecuencia variable, este filtro va de 1 a 3 kHz. Este filtro tiene como función variar el tono producido por el efecto. Se eligió este rango de frecuencias debido a que se obtiene un mejor sonido a altas frecuencias.

Creación de la interfaz final

En esta parte se conjuntaron todos los patches anteriores. Para ello es necesario realizar un patch o panel principal, ver figura 9. Los efectos se encuentran como subpatches. Se agregan paneles de colores para delimitar las áreas de cada efecto. Dentro de éstas, se colocan nuevamente las barras de control para cada patch. Se utilizan conexiones inalámbricas para enviar y recibir información, de esta manera se consigue una mayor limpieza y orden en el patch. Los efectos se configuran en *modo true bypass*, para poder ejecutar uno o más efectos simultáneamente. Al activar un efecto determinado, es necesario pulsar el interruptor, que se encuentra en la esquina superior derecha de cada efecto. Otra forma de activar el efecto es través de interruptores tipo *foot switch*. Es decir, se brindan ambas opciones de activación de los efectos.

Etapa de selección

En esta etapa el usuario tiene la capacidad de elegir, a placer, el efecto o los efectos deseados a través de *foot switch.* En la figura 13, se muestra el diagrama de conexión de los interruptores *foot switch* y la Raspberry.

Para la lectura de los botones se utilizaron herramientas de macros programadas en lenguaje Python. De esta forma, se le asigna una letra a cada botón. Posteriormente, se elaboró un patch que detecta la pulsación de una tecla en código ASCII. Si esta corresponde al código de la letra asignada, se marcará la casilla de dicho efecto en la interfaz y comenzará a ejecutar el efecto.

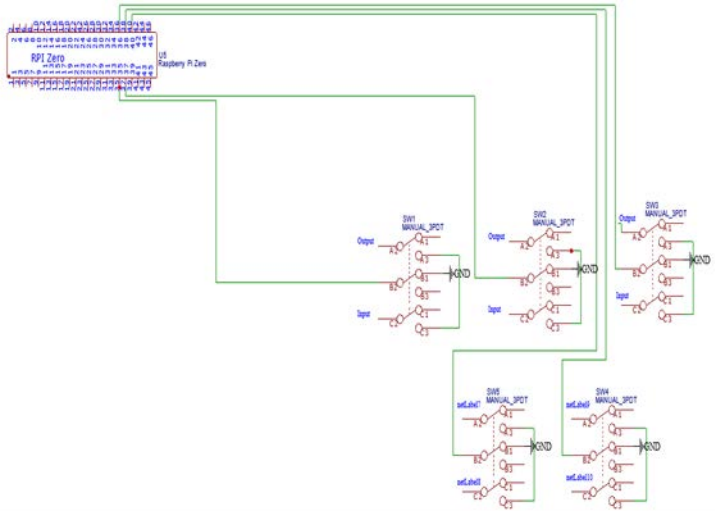


Figura 13 Diagrama de conexión de los botones *foot switch* y la Raspberry.

Etapa conversión digital analógica

Dado que esta versión de la Raspberry Pi no cuenta con un puerto de sonido de salida, es necesario configurar dos canales PWM para realizar la conversión. Posteriormente, se implementó un filtro pasa bajas para obtener una señal de salida puramente analógica [Adafruit, 2015]. En la figura 14, se puede observar que se utilizan las líneas 13 y 33 del Raspberry y a su salida se conecta el filtro pasa bajas pasivo diseñado.

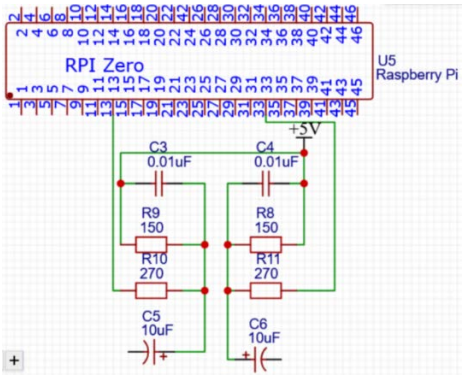


Figura 14 Filtro pasa bajas implementado en las salidas PWM de Raspberry.

Etapa de tratamiento analógico

Después de procesar la señal digital y realizar la conversión DA. Se diseñó e implementó un ecualizador, construido a partir de tres etapas de filtrado basados en

redes Sallen Key. Se eligieron estos filtros activos debido a sus características para operar a bajas frecuencias y no involucrar el uso de inductores. Se eligieron filtros pasa bajas y pasa altas de tipo Bessel, debido a su comportamiento lineal en la banda de paso. Además, es necesario amplificar la señal, para ello se utilizó un circuito amplificador de primer orden. El control de las bandas y del volumen se realiza por medio de potenciómetros o perillas. En la figura 15, se muestra el diagrama de conexión de esta etapa.

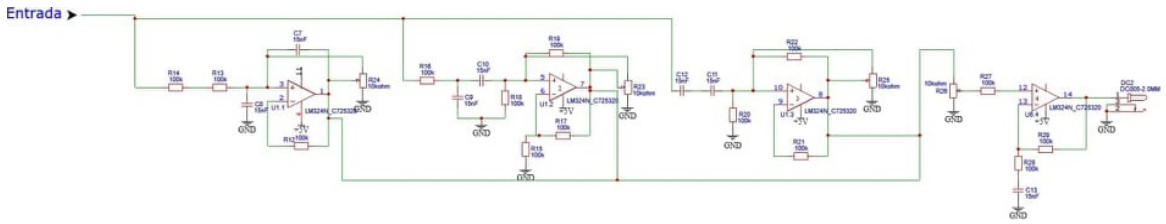
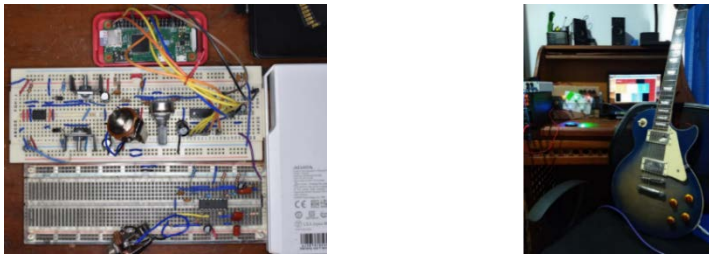


Figura 15 Diagrama de conexión de la etapa del tratamiento analógico.

Sistema completo

En la figura 16a, se muestra el sistema, completo, implementado en una tablilla de conexiones (protoboard). En la figura 16b, se muestra cómo se realizaron las pruebas de funcionamiento del sistema. El diagrama de interconexión, con todas las etapas, se muestra en la figura 17.



a) Plantilla de conexiones físicas del pedal diseñado. b) Conexión de los elementos del proyecto.

Figura 16 Sistema de procesamiento de audio interconectado.

3. Resultados

El desarrollo general del proyecto se llevó a cabo de manera modular, de esta manera se aseguró el funcionamiento de cada una de las partes. Posteriormente, se procedió a la interconexión de cada una de las etapas hasta obtener el sistema

completo en funcionamiento. Para comprobar el funcionamiento del pedal, se realizaron las pruebas de manera auditiva. Se ejecutaron diferentes progresiones rítmicas y melódicas en la guitarra.

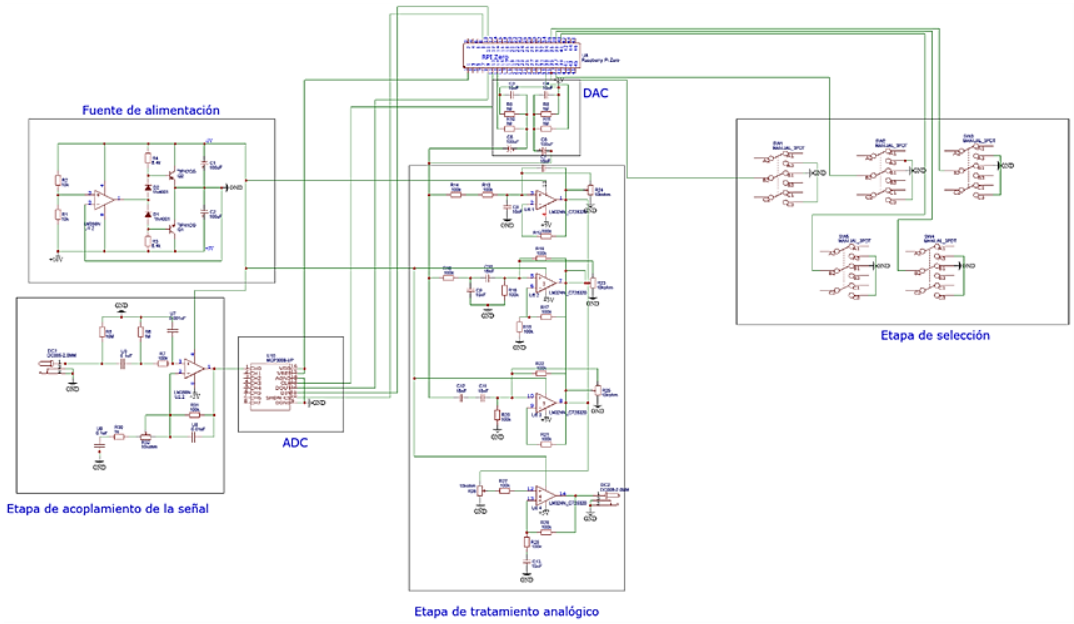


Figura 17 Diagrama del sistema implementado.

En las primeras pruebas, se activaron de manera individual cada uno de los efectos diseñados e implementados, así como el funcionamiento de la interfaz gráfica mediante la variación de los parámetros de cada efecto. Posteriormente, se comenzaron a activar los efectos de manera conjunta y de esta manera escuchar el resultado. Adicionalmente, se probó el pedal utilizando diferentes modelos de guitarras, se obtuvo correctamente cada efecto seleccionado.

En la comprobación del funcionamiento del ecualizador de tres bandas, no se obtuvieron problemas en identificar su efecto sobre el sonido. Al centro se puede observar la interfaz gráfica (que se envía mediante conexión VNC desde la Raspberry a una computadora), así como la guitarra utilizada para generar melodías y aplicar distintos efectos. Cabe mencionar que el funcionamiento de la Raspberry no se vio afectado por las conexiones de todo el sistema. El uso de Pd fue adecuado, ya que mostró un gran desempeño en cuanto al desarrollo de interfaces y también el procesamiento digital de audio. El consumo de recursos es bajo, lo cual

lo hace excelente para la aplicación de este sistema, donde dependemos mucho de los recursos computacionales.

4. Discusión

Si bien Pd fue de gran ayuda para implementar los diferentes efectos de sonido, así como la interfaz gráfica; es claro Pd tiene una gran limitante. Algunas librerías no son compatibles en todas las versiones, esto complica el desarrollo de objetos para darle un aspecto diferente a la interfaz. Algunas de estas librerías son: *flatgui* y *cyclone*. Una librería que funcionó temporalmente fue *Moonlib,* sin embargo, después de abrir y cerrar varias veces el patch, los objetos generados con esta librería perdían sus funcionalidades. Por esta razón se implementó código en C y Python para solventar estas incompatibilidades en Pd. Cabe mencionar que algunos colores en la interfaz hacen referencia a los colores utilizados en el logo de la unidad (rojo y negro) y que los otros colores fueron inspirados en los pedales analógicos BOSS. La programación de los *foot switch* mediante macros fue adecuada, ya que simplifica bastante el uso de los botones, y en caso de querer agregar más botones, basta con asignar nombres de variables al código de Python y modificar el patch en Pd para reconocer más teclas. En la etapa de conversión DA, la programación de las señales PWM para la salida estéreo del pedal fue sencilla con una calidad de audio obtenida es aceptable.

5. Conclusiones

El pedal diseñado e implementado en este trabajo puede realizar uno o más efectos simultáneamente. Además, es utilizable con cualquier tipo de guitarra. Una desventaja es que no proporciona un audio con alta calidad, ya que el ADC utilizado, cuenta con tan solo 10 bits de resolución. Para poder utilizar un ADC con una cantidad mayor de bits, por ejemplo 12 bits, es necesario solucionar un problema de compatibilidad entre el ADC y los patches a utilizar.

Otra ventaja es que el sistema contenido en tamaño, lo que lo hace portable. Tiene la capacidad de conectarse a cualquier dispositivo móvil, que cuente con una conexión VNC. Esta opción nos permite controlar el pedal de manera manual en

caso de querer hacerlo y modificar los patches en caso de querer experimentar con el sonido. El sistema funciona, pero bajo ciertas limitaciones. La principal de estas limitaciones es: la capacidad de procesamiento del ADC, el poder de procesamiento de la Raspberry Pi y de recursos de esta. La cantidad de bits que pueda procesar el ADC afecta mucho en la calidad de audio que proporciona, ya que 10 bits no son suficientes para brindar una alta calidad de audio. Por ello, se propone como trabajo a futuro, mejorar la calidad de audio, mejorando el ADC a utilizar. También, se propone aumentar la calidad de efectos de sonido, lo que haría un sistema con una cantidad de efectos considerable, de fácil uso, modificable y portable.

6. Bibliografía y Referencias

[1] Adafruit, (2015). Adding Basic Audio Output to Raspberry Pi Zero. https://learn.adafruit.com/adding-basic-audio-ouput-to-raspberry-pi-zero/pi-zero-pwm-audio.

[2] Boss, (2020). DS-1 Pedal compacto Distortion. https://www.boss.info/mx/products/ds-1/.

[3] Duxans H. B., Ruiz M., Efectos digitales de la señal de audio, Universidad Oberta de Catalunya, 2012.

[4] ElectroSmash, (2020a). Pedal Pi- Rasberry Pi Zero Guitar Pedal. https://www.electrosmash.com/pedal-pi.

[5] ElectroSmash, (2020b). PedalShield Mega Arduino Guitar Pedal. https://www.electrosmash.com/pedalshield-mega.

[6] Huffenus A. & Pillonnet G., (2015). Digitally Assisted Analog: An Anti-Clipping Function for Class-D Audio Amplifier. Journal of Low Power Electronics. 11.10.1166/jolpe.2015.1360.

[7] Instructables, (2020). Arduino Guitar Pedal. https://www.instructables.com/Arduino-Guitar-Pedal/.

[8] Mooer, (2020). GE100 Guitar Multi-Effects Processor. http://www.mooeraudio.com/product/GE100-168.html.

[9] Schroeder M. R., Natural Sounding Artificial Reverberation. Journal of the audio engineering society. No. 10(3), 1962.