PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERÍA

ESCUELA DE SISTEMAS

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE

INGENIERO EN SISTEMAS Y COMPUTACIÓN

“DISEÑO Y DESARROLLO DE UN SINTETIZADOR DE SONIDO DE SOFTWARE”

AUTOR: RONNY FERNANDO CÓRDOVA GUERRÓN

QUITO, AGOSTO 2019

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1: INTRODUCCIÓN

JUSTIFICACIÓN

Es una obviedad asegurar que la velocidad de crecimiento tecnológico a nivel mundial es cada vez mayor. Prácticamente a cualquier lado donde se regrese a mirar, se va a encontrar un aparato tecnológico, ya sea computadora, celular, tablet, etc. La masificación de la tecnología ha sido uno de los logros más grandes de la sociedad actual y, evidentemente, dicha masificación ha significado un avance y un desarrollo social sin precedentes. En cualquier área del conocimiento humano, la tecnología ha traído consigo herramientas que hacen posible plasmar cualquier idea de forma casi inmediata, a un costo relativamente bajo y con la posibilidad de compartirlo con todo el mundo instantáneamente. Sin mencionar que prácticamente toda la información y el conocimiento generado por la civilización humana en toda su historia es accesible desde la "palma de la mano".

En el caso particular que compete al presente trabajo, el campo de la música, conformado, evidentemente, por músicos; ya sean estos profesionales, aficionados o sencillamente personas interesadas en aprender del tema, la tecnología les brinda actualmente las herramientas necesarias para realizar proyectos musicales completos, con una calidad profesional en todos los aspectos, usando solamente un computador; dado que el poder de procesamiento actual permite simular y grabar cualquier instrumento de manera digital, sin necesidad de disponer de equipos analógicos caros y un espacio físico considerable, como se requería en décadas anteriores. Esto ha permitido que millones de personas logren expresarse, sin limitaciones de ninguna índole, a través de la música y puedan, muy fácilmente, compartir su arte con el mundo.

Con el fin de especificar, se puede mencionar a los Digital Audio Workstation o DAW como una de las más herramientas de software más usadas en este campo, los cuales son programas

orientados a facilitar la grabación, mezcla y masterización de proyectos de audio. Dentro de dichos programas existen herramientas desarrolladas con el objetivo de trabajar en conjunto con el DAW -o como programas independientes- y ampliar el abanico de posibilidades de las que dispone el músico de manera prácticamente infinita: samplers, loops, plug-ins, emulación de instrumentos, capacidades MIDI, entre tantos otros disponibles, son algunos de los ejemplos.

Dentro de este último grupo se encuentran los sintetizadores de sonido de software, los cuales, si bien es cierto al compararlos dentro de la gran cantidad de herramientas disponibles, terminan ocupando un espacio reducido; no por eso dejan de ser herramientas sumamente interesantes y que brindan un sinfín de posibilidades musicales a los usuarios. No por nada son instrumentos indispensables en cualquier estudio de grabación y son de los favoritos de músicos, tanto aficionados como profesionales, a la hora de elaborar proyectos musicales.

Los sintetizadores de sonido son capaces de emular instrumentos “reales” y de generar sonidos con timbres únicos, los cuales son, teóricamente, infinitos. Incluso, cada sintetizador es único en sí mismo y, por lo tanto, capaz de producir sonidos y timbres que otros sintetizadores no son capaces. Todo depende del diseño y del tipo de cada sintetizador. Debido a esto se pueden realizar proyectos musicales completos utilizando solamente un sintetizador de sonido. Más aun cuando están disponibles completamente no solo como hardware, sino también como software. Por lo tanto, resultan herramientas muy potentes para cualquier músico o productor musical.

Evidentemente, ya existen en el mercado varias opciones en cuanto a sintetizadores de sonido de software se refiere, cada uno con su diseño y, por lo tanto, sonido particular. Aun así, la bibliografía en cuanto a cómo desarrollarlos es escasa. En el presente trabajo de titulación se plantea diseñar y desarrollar un sintetizador de sonido de software personalizado, que responda a y se adapte las necesidades de un grupo de músicos en específico. Además, entendiendo la

importancia que tiene el libre acceso a la información como pilar fundamental del desarrollo tecnológico y social, el presente trabajo pretende también servir como aporte bibliográfico de libre acceso sobre el tema en cuestión, actuando como base y guía para futuros desarrollos, tanto para el autor, como para cualquier persona interesada en los detalles sobre cómo implementar un proyecto de esta índole, sin tener, necesariamente, un conocimiento muy profundo del tema.

OBJETIVOS

Objetivo General

I. Diseñar y desarrollar un sintetizador de sonido de software

Objetivos Específicos

i. Definir los conceptos, componentes y características de un sintetizador de sonido.

ii. Definir las características y funciones necesarias de un sintetizador de software personalizado, detallando los componentes que lo forman, así como la interacción entre estos.

iii. Desarrollar algoritmos que alteren la forma de onda del sonido.

iv. Desarrollar algoritmos que permitan manipular una señal de audio en tiempo real.

v. Implementar técnicas de procesamiento de audio a través de software que permitan el desarrollo del sintetizador de sonido de software de manera óptima y eficiente.

vi. Desarrollar una interfaz de usuario que resulte intuitiva y fácil de usar, es decir, se explique por sí misma, con una distribución clara de los controles del sintetizador de sonido de software.

vii. Desarrollar presets de instrumentos musicales, utilizando configuraciones definidas de las variables del sintetizador de sonido de software.

CAPÍTULO 2: SINTETIZADOR DE SONIDO

DEFINICIÓN

Es un instrumento musical. Esa es la forma más simple de definir a un sintetizador de sonido. Evidentemente, dicha definición, si bien es técnicamente correcta, puede y debe ser ampliada, con el objetivo de entender a profundidad lo que abarca el concepto general de sintetizador de sonido y encontrar las cualidades que lo diferencian de otros instrumentos musicales.

Para esto, hay que empezar por lo más básico. ¿Qué es síntesis? La Real Academia Española [RAE] (2018) la define como: "Composición de un todo por la reunión de sus partes". En este caso, se puede entender como el "todo" al sonido como tal, lo que significaría que un sintetizador de sonido es aquel artefacto el cual, mediante la unión de sus partes, produce sonido. Evidentemente, esta definición no termina resultando del todo precisa, dado que se la puede aplicar a cualquier otro instrumento musical. Una guitarra, por ejemplo, mediante la unión de su mástil, caja de resonancia y cuerdas -entre otras partes- produce sonido. Por suerte, la misma RAE brinda otra definición para "síntesis", más adaptable a la función que cumple, en la práctica, un sintetizador de sonido, la cual es la siguiente: "Proceso de obtención de un compuesto a partir de sustancias más sencillas" (RAE, 2018). Siendo el sonido una onda que se propaga a través del aire, un sintetizador de sonido "sintetiza" ondas complejas a partir de la combinación de ondas simples; entendiéndose por "onda compleja" a un sonido con un timbre determinado, y por "onda simple", principalmente, a la onda fundamental o básica del sonido, la onda sinusoidal -a pesar de que existen otros tipos de onda que se puede también catalogar como "ondas simples". Esta onda se denomina "fundamental" debido a que es la expresión mínima a la cual se puede reducir el sonido; dicho en otras palabras, "cualquier onda compleja se puede descomponer en la suma de varias ondas sinusoidales" (Shepard, 2013, pág. 66). Este concepto,

relacionado con el funcionamiento básico de un sintetizador, será explicado más a detalle posteriormente en el presente trabajo.

Dicho lo anterior, se puede concluir que un sintetizador de sonido es mucho más funcional que un instrumento musical clásico, dado que, teóricamente, es capaz de sintetizar cualquier sonido -como la voz humana, por ejemplo- sin tener, necesariamente, el objetivo de generar una composición o interpretación musical. Sin embargo, su uso más extendido es, justamente, el de un instrumento musical, dado su enorme aplicabilidad en este campo, pues su capacidad de generar sonidos con timbres únicos y manipularlos a conveniencia lo diferencia del resto de instrumentos musicales. Esto, sumado a que, prácticamente desde sus inicios -y en su gran mayoría- son comercializados junto con un teclado de piano clásico, y por otro lado son accesibles también como herramienta de software; le ha permitido posicionarse como un instrumento esencial en cualquier estudio de grabación.

En resumen, se puede definir a un sintetizador de sonido como aquel aparato que genera, ya sea electrónica o digitalmente, ondas sonoras complejas a partir de la combinación y/o manipulación de varias ondas sonoras simples, con el objetivo de usar el sonido producido en un contexto de composición o interpretación musical. Si bien ya se aclaró que no necesariamente debe un sintetizador de sonido tener un objetivo “musical”, se utilizará dicha definición en el presente trabajo, pues, además de ser uno de los objetivos del desarrollo práctico del mismo, representa de manera más precisa lo que cultural e históricamente se conoce como sintetizador de sonido.

HISTORIA

Orígenes y Evolución del Sintetizador de Sonido

Dado que se ha aclarado que la definición de un sintetizador de sonido no es exacta y tiene, más bien, un significado cultural que literal, resulta difícil determinar su origen exacto. Se puede tratar de abordar este problema buscando el primer aparato capaz de generar sonido de forma electrónica, pues esta una característica fundamental de los primeros sintetizadores. Para esto, es esencial mencionar al teléfono, como una invención clave, no solo para la sociedad, evidentemente; sino también para el desarrollo de los sintetizadores.

> A pesar de que la invención del teléfono demostró que era posible transmitir la voz humana desde un lugar hacia otro por medios electrónicos, esta no fue la única razón para la comercialización del teléfono, (…) una de las fuerzas mayores que impulso la adopción del teléfono fue de hecho musical – el teléfono hizo posible transmitir interpretaciones musicales a muchas personas. (Russ, 2009, pág. 14)

La tecnología básica que permitió el desarrollo de los sintetizadores radica en la rama de las telecomunicaciones, pues dicha rama centró su estudio en el sonido, específicamente en cómo trasmitirlo usando medios electrónicos. La invención del teléfono no solo demostró que el sonido era transmisible electrónicamente, sino también, abrió la puerta a considerar la posibilidad de generar sonidos electrónicamente.

Dicho esto, de acuerdo con Pirkle (2015) “muchos consideran al inventor americano Thaddeus Cahill como el inventor del sintetizador moderno. En 1897, registró una patente para un aparato denominado “Telarmonio”, un instrumento de 200 toneladas que usaba generadores electrónicos accionados a vapor para generar sinusoides” (pág. 1). Según moogfoundation (2013), Cahill transmitía los sonidos que generaba el Telarmonio a través de la red telefónica; en

esencia, creó un sistema de transmisión musical para el teléfono. Este es el primer aparato que usaba la electrónica para aplicar la que se considera, en el presente trabajo, la idea fundamental de los sintetizadores, la cual es generar ondas sinusoidales y combinarlas para generar ondas complejas, y, por lo tanto, sonidos más complejos en cuanto a su timbre. Siendo, justamente, esta idea la base teórica sobre la cual surgen los sintetizadores, resulta oportuno preguntarse: ¿Dónde radica el origen de dicha idea fundamental?

En 1870 se publica el libro "On the Sensations of Tone as a Physiological Basis for the Theory of Music". En él, su autor, Helmholtz (1912) describió que los componentes de los sonidos complejos son una combinación de armónicos de una nota fundamental, siendo los armónicos definidos como frecuencias por encima de dicha nota fundamental. En otras palabras, todo sonido se compone de una nota fundamental -la cual tiene su respectiva frecuencia y es la que se distingue al momento de escuchar el sonido-, y de un grupo de frecuencias que se generan a partir de dicha nota fundamental. Dependiendo de las frecuencias que compongan a un sonido, este tendrá un timbre distinto.

Este trabajo, que buscaba, en palabras de su propio autor: "conectar (…) los límites de la acústica fisiológica, por un lado, y la estética y ciencia musical por otro" (Helmholtz, 1912, pág. 1), tuvo un impacto científico muy grande. Logró, según Kursell (2009), investigadora del Max Planck Institute for the History of Science, que: "la estética de la música (…) colapsará bajo una explicación científica". Helmholtz no solo demostró de que están compuestos los sonidos complejos; sino, además, llegó incluso a explicar por qué ciertos sonidos los percibimos como música y otros como ruido. Con su trabajo, aportó enormemente al conocimiento sobre la composición del sonido y, en consecuencia, definió la teoría fundamental que permitió el desarrollo de los sintetizadores.

Una vez que Helmholtz postuló que todo sonido es la combinación de varias frecuencias, era cuestión de tiempo para que alguien tuviera la idea de desarrollar un aparato que pudiera generar frecuencias individuales -ondas sinusoidales- y combinarlas para generar un sonido complejo, dado que, teóricamente, dicho aparato sería capaz de producir cualquier sonido que el ser humano pudiera imaginar. Con esa premisa, Cahill desarrolló el Telarmonio, y sentó las bases de los sintetizadores de sonido tal como se los conoce hoy en día.

A partir de ahí, hay 3 momentos históricos que se consideran claves, en el presente trabajo, a la hora de hablar sobre la evolución de los sintetizadores: la primera vez que se usó la denominación "sintetizador de sonido" en un aparato orientado principalmente a la composición musical, la popularización de los sintetizadores gracias a Robert Moog, y la primera revolución en las técnicas de síntesis, con el Yamaha DX7.

En 1955, en los laboratorios de la RCA (Radio Corporation of America), empieza a surgir "la idea de eliminar costes de sesión para cada uno de los músicos de orquestra con algún tipo de máquina electrónica que permita contratar a un solo compositor" (alanoneuser, 2012). Basados, nuevamente, en la teoría de Helmholtz, en RCA consideraban que era posible desarrollar un aparato que genere los sonidos de todos los instrumentos clásicos y que además sea programable, de modo que, contratando a un solo compositor, se pudiera grabar discos enteros. Harry Olson y Hebart Belar estuvieron a cargo de llevar a cabo dicha idea, y desarrollaron el sintetizador RCA Mark I -y posteriormente el Mark II. Dicho sintetizador generaba sonidos electrónicos usando osciladores de tubo de vacío, era del tamaño de un cuarto y funcionaba a través de tarjetas perforadas, las cuales contenían la información sobre la composición musical que se quería generar; no podía reproducir música en tiempo real, sino que grababa directamente la composición. Fue el primer sintetizador programable y el primer aparato, orientado directamente

a la composición musical, que usó el término “sintetizador” explícitamente. (AutomaticGainsay, 2018). Fue gracias al RCA Mark II que hubo un cambio de paradigma, y se empezó a relacionar el término “sintetizador” directamente con una herramienta de composición musical.

A pesar de que el Mark II llamó la atención de varios músicos debido a su innovación, dado su tamaño y la complejidad de programar una composición musical entera usando tarjetas perforadas, el sintetizador de RCA no revolucionó el mercado. Además, se hizo evidente que la idea de generar varias ondas sinusoidales individuales para luego combinarlas, si bien es teóricamente correcta, en la práctica presentaba varios problemas, sobre todo la necesidad de tener varios osciladores -generadores de ondas sinusoidales- independientes para poder generar sonidos interesantes. No fue hasta 1963 que Robert Moog desarrollaría el primer sintetizador capaz de ser vendido a las masas y, con ello, causara una revolución sobre la forma en la que se hacía música. Moog se dio cuenta rápidamente que tener varios osciladores no era práctico y abordo el problema desde otra perspectiva. En lugar de sumar varias frecuencias, partió de un sonido inicial complejo al cual le quitaba frecuencias. El aparato que desarrolló estaba conformado por módulos electrónicos independientes que, en conjunto, daban forma al instrumento. Cada módulo realizaba lo siguiente, según Pirkle (2015):

> El módulo de oscilador generaba formas de onda primitivas: sinusoidal, sierra, y ondas cuadradas. El módulo de amplificación usaba un amplificador controlado por voltaje para modificar la amplitud de la señal. El módulo de filtro implementaba varias funciones básicas de filtrado como pasa-bajos y pasa-altos. El módulo de generador de envolvente creaba un valor de salida variable que era usado para controlar las características de amplitud o filtrado de otros módulos. (pág. 1)

La gran ventaja del diseño modular de Moog era que cada módulo -y el sintetizador completo- era relativamente pequeño, dado que solo necesitaba uno o dos osciladores. Su funcionamiento demostró que no solamente sumando ondas sinusoidales se podía sintetizar sonido, abriendo paso al concepto que se conoce como "técnicas de síntesis", las cuales son, justamente, diferentes formas de manipular el timbre de un sonido. El sintetizador de Moog "rompió" el mercado cuando, según Russ (2009): "Walter Carlos lanzó un álbum con composiciones clásicas de Bach ejecutadas en el sintetizador modular. (…) *Switched on Bach* rápidamente se convirtió en un gran éxito, (…) convirtiéndose en uno de los álbumes de música clásica más vendidos de la historia" (pág. 27). Es aquí donde los sintetizadores empezaron a demostrar todo su potencial y finalmente despertaron el interés de la comunidad musical, pues grandes bandas empezaron a experimentar con ellos. "Los Beatles y los Rolling Stones se convirtieron en compradores tempranos de los sintetizadores modulares de Moog" (Russ, 2009, pág. 27).

A partir de este punto, la influencia de los sintetizadores dentro de la música creció exponencialmente. Junto con la conquista del mercado musical por parte de este instrumento, se desarrolló también el que se puede considerar como el avance tecnológico más importante del siglo pasado: el transistor -y, posteriormente, los microprocesadores-. Esto representó un avance tecnológico que dio lugar al mundo digital, del cual los sintetizadores también resultaron beneficiados.

Al momento de hablar de sintetizadores digitales se puede considerar al Yamaha DX7 como el más importante e influyente, dado que no solo usaba un microprocesador para realizar la síntesis de sonido, sino que implementaba una técnica de síntesis completamente nueva, la "síntesis FM". Dicha técnica fue descrita por John Chowning, en una publicación científica titulada: *La Síntesis de Espectros Complejos de Audio por Medio de la Modulación de*

Frecuencias, publicada en 1973 (Russ, 2009). Chowning, al igual que hizo Moog, dejaba atrás la idea fundamental, ya mencionada previamente, de generar varias ondas sinusoidales, que sean armónicos de una onda fundamental, y combinarlas para sintetizar un sonido complejo. Sin embargo, su objetivo final era exactamente el mismo, producir ondas complejas que den como resultado sonidos con timbres únicos. Para lograr esto usaba una onda sinusoidal que modulaba la frecuencia de otra onda sinusoidal, de ahí el término "Modulación de Frecuencia". La técnica se basa en lo siguiente: "Las dos ondas están dentro del espectro audible y, a través de complejos reflejos de frecuencia, inversiones de fase y cancelaciones, se pueden producir una amplia gama de timbres" (Russ, 2009, pág. 9). El resultado fue un sintetizador revolucionario, pues la síntesis FM abrió un nuevo horizonte para modelar timbres. Incluso era capaz de sintetizar sonidos de percusión, como marimbas o campanas (Polyphonic, 2018).

Lo más importante es que Chowning demostró que el campo de las técnicas de síntesis todavía tenía mucho que ofrecer, y se podían expandir más allá de la teoría fundamental descrita por Helmholtz y el sintetizador de Moog, de modo que amplió aún más las posibilidades de un instrumento que ya, de por sí, ofrecía un campo de acción muy amplio.

Si bien es cierto hay muchos otros momentos que se pueden analizar en la evolución de los sintetizadores, como por ejemplo la aparición del órgano Hammond -basado en el Telarmonio-, la aparición del crochet 5 o el desarrollo de la síntesis granular, por nombrar unos ejemplos; a partir del conocimiento de los hechos históricos que se identificaron, en el presente trabajo, como más relevantes; se puede realizar el siguiente análisis, con el objetivo de identificar las ventajas y desventajas de diseñar y desarrollar un sintetizador en la época actual.

Análisis Histórico

La teoría básica, desarrollada por Helmholtz, detrás de un sintetizador es relativamente simple. Lo que la historia enseña es que los sintetizadores aplicaron ese concepto de acuerdo a las posibilidades tecnológicas de cada época, dando como resultado una variedad de técnicas de síntesis -y, por consiguiente, de sintetizadores-, que se describieron previamente y serán detalladas con mayor profundidad posteriormente en el presente trabajo. En la actualidad, las posibilidades digitales prácticamente eliminan estas barreras a la hora de diseñar e implementar un sintetizador, abriendo la posibilidad de seleccionar cualquier cantidad y cualquier tipo de componentes que se necesite, así como definir la interacción entre estos a conveniencia; y aplicar cualquier técnica de síntesis que se desee, o incluso una combinación de varias técnicas; dado que todo se puede emular digitalmente por software. Esto aumenta muchísimo las posibilidades de desarrollo, pero, a su vez, complica de misma forma el diseño. En cierto modo, las limitaciones tecnologías de las décadas pasadas, al brindar un campo de acción reducido, marcaban un camino que seguir del cual no se podía salir. Cahill solo disponía de la tecnología dada por los primeros pasos de las telecomunicaciones, Olson y Belar solo tenían a su disposición tarjetas perforadas para programar las composiciones musicales, Moog contaba con el control de voltaje como base fundamental, y Chowning logró generar un algoritmo que resultó viable dada la capacidad de procesamiento de la época. Al tener pocas opciones al momento de implementar una idea, la creatividad necesariamente se estimula para generar un diseño que se adapte a los límites que existan. Por lo tanto, los límites, sean del tipo que sean, son necesarios. Son la guía que permiten no perderse en un campo abierto de posibilidades. En la actualidad, dichos limites no son tan evidentes. La teoría fundamental en relación a que propiedades del sonido manipular para generar timbres es la misma que hace 130 años, pero el poder de

procesamiento actual permite diseñar cualquier tipo de sintetizador que se pueda llegar a imaginar. Por lo tanto, hay que definir un camino marcado por otro tipo de límites, ya no factores tecnológicos, sino más bien factores un poco más subjetivos, como: mercado objetivo, experiencia musical, uso específico, eficiencia de uso, tendencias de diseño, etc. Entender la necesidad de tener límites y definirlos adecuadamente es clave, de modo que se aprovechen al máximo las capacidades tecnológicas de la presente época y se consiga diseñar un sintetizador que encuentre un equilibrio adecuado entre capacidad de generación y manipulación de sonidos, y facilidad de uso.

TÉCNICAS DE SÍNTESIS DE SONIDO

Cualquier método para manipular o modificar las frecuencias que componen a un sonido -alterando así su timbre- se puede considerar como una técnica de síntesis de sonido. La idea de sumar varias ondas sinusoidales fue la primera técnica de síntesis, y a partir de ella surgieron nuevas y varias formas de sintetizar sonido, cada una con resultados diferentes. En la presente sección se revisarán 3 de las técnicas más usadas en los sintetizadores modernos. Para ello, es necesario empezar profundizando en las propiedades del sonido.

Propiedades del Sonido

Si se empezó el presente capítulo definiendo lo que es "síntesis", resulta prudente empezar la presente sección dando una definición más detallada de lo que es "sonido". Según la RAE (2018), el sonido es la "Sensación producida en el órgano del oído por el movimiento vibratorio de los cuerpos, transmitido por un medio elástico, como el aire." Al ser un fenómeno natural, el sonido tiene una definición más precisa, dado que su comportamiento y sus propiedades han sido estudiadas y explicadas a través de la física. Justamente, el estudio de dichas propiedades -como lo hizo Helmholtz- ha servido para explicar las causas que llevan a que ciertos sonidos sean percibidos, por ejemplo, como musicales y placenteros; y otros como ruido e indeseables.

El medio más común de transmisión del sonido es, evidentemente, el aire. Cuando un objeto determinado vibra, dichas vibraciones -también llamadas oscilaciones- transmiten energía a través del aire generando cambios de presión, dichos cambios de presión llegan al oído humano, el cual envía impulsos nerviosos, los cuales son interpretados por el cerebro como sonido (Shepard, 2013).

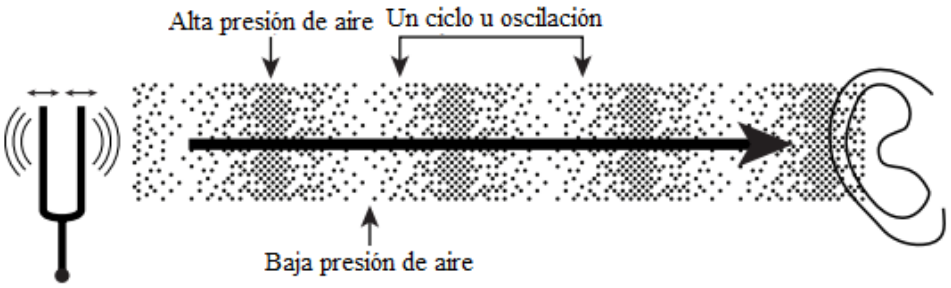

Figura 1. Cambios de Presión en el aire. Shepard (2013).

Dichos cambios de presión se comportan como una onda -o, mejor dicho, son ondas. "Toda onda se compone de dos partes: compresión (cuando la presión de aire generada es mayor a la presión del aire en el ambiente) y rarefacción (cuando la presión de aire generada es menor a la presión del aire en el ambiente)" (Shepard, 2013, pág. 6). La evolución en el tiempo de la onda entre la etapa de compresión y de rarefacción se la representa de la siguiente manera:

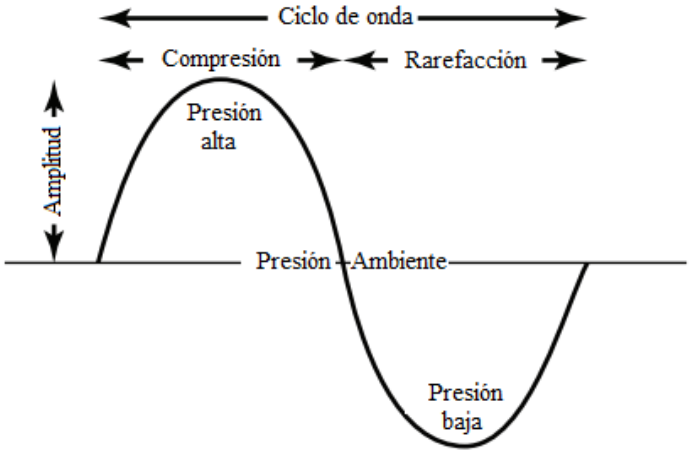

Figura 2. Onda de sonido. Shepard (2013).

Las propiedades fundamentales de la onda de sonido son: amplitud, frecuencia y timbre. Mientras mayor intensidad tengan los cambios de presión en el aire generados por la vibración, mayor es la amplitud y, por lo tanto, el sonido se percibe más alto; mientras más seguidos entre si estén los cambios de presión, mayor es la frecuencia y el sonido se percibe más agudo (Shepard, 2013).

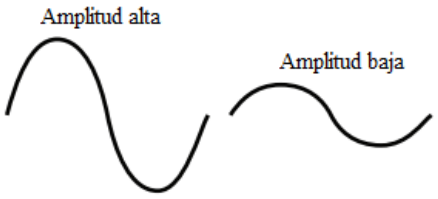

Figura 3. Amplitud de onda de sonido. Shepard (2013).

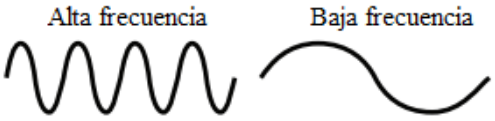

Figura 4. Frecuencia de onda de sonido. Shepard (2013).

El timbre, por su parte, es un concepto un poco más complejo. Es la propiedad que causa que una misma nota interpretada en una guitarra suene diferente a si se la interpreta en una flauta. Justamente, el timbre es la propiedad principal que se busca manipular en un sintetizador. Para entenderlo a detalle, hay que entender el concepto de onda fundamental del sonido.

Onda Fundamental del Sonido

La *Figura 2*, como ya se mencionó, es una representación de los cambios de presión en el aire generados por la vibración de un objeto. Dicha forma de onda es una onda sinusoidal. En la naturaleza, sin embargo, prácticamente ningún sonido tiene esa forma de onda, dado que los cambios de presión no se comportan de una manera tan "perfecta". Por ejemplo, a continuación, se muestra la forma de onda de una guitarra, donde se puede apreciar una forma de onda más "natural":

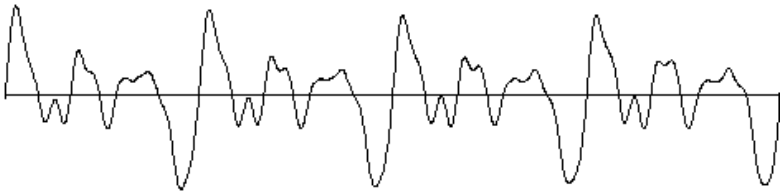

Figura 5. Forma de onda - Guitarra acústica. Elaboración propia.

Justamente, la forma de onda es una característica esencial del timbre de un sonido. En palabras de Helmholtz (1912): "El timbre debe depender de la manera en que el movimiento es realizado en el periodo de cada vibración" (pág. 19). Dado que la amplitud definía el volumen y la rapidez de vibración la frecuencia, la forma de onda era la única propiedad del sonido que se podía atribuir al timbre. Esta fue la premisa sobre la cual Helmholtz basó sus experimentos. Entonces, ¿Por qué se puede considerar a la onda sinusoidal como la onda fundamental del sonido? La respuesta está en el trabajo del matemático y físico francés Jean-Baptiste Joseph Fourier, quien postuló la idea de que cualquier onda periódica compleja podía ser descompuesta en la suma de una serie de ondas sinusoidales simples. Dicha serie de ondas se conoce como la serie de Fourier (Shepard, 2013). Es necesario detenerse un momento en el concepto de "onda periódica", dado que no ha sido detallado hasta el momento. Una onda es periódica cuando tiene

un ciclo que se repite indefinidamente, como se puede apreciar en el ciclo de una onda sinusoidal o la onda de una guitarra:

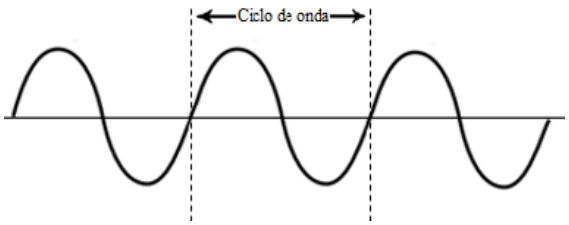

Figura 6. Ciclo de onda sinusoidal. Elaboración propia.

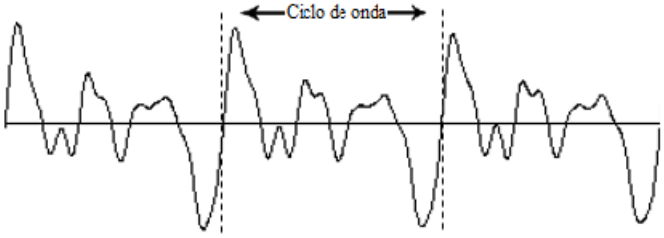

Figura 7. Ciclo de onda - Guitarra acústica. Elaboración propia.

Este concepto es importante dado que cuando una onda es periódica, el cerebro humano la interpreta como “música”, y cuando es no-periódica, es interpretada como ruido (Helmholtz, 1912). Por lo tanto, la “periodicidad” de la onda es una característica fundamental del sonido generado por un instrumento musical. Esto no significa que una onda no-periódica no pueda ser descompuesta en una seria de ondas sinusoidales, pero dado que la onda no-periódica no es constante, las ondas sinusoidales que la componen tampoco lo son (Russ, 2009). Por supuesto, en ondas periódicas el cálculo matemático se simplifica y se puede aplicar la serie de Fourier.

Dicho esto, una vez aceptado que toda onda periódica compleja puede ser descompuesta en ondas sinusoidales, también se puede postular que se pueden formar ondas complejas a partir de

la suma de varias ondas sinusoidales. Como cada onda sinusoidal tiene su propia frecuencia, el grupo de frecuencias que forman a un sonido determinan la forma de su onda y, por ende, su timbre. Esta es justamente la idea fundamental de los sintetizadores que ya se ha descrito previamente. Por lo tanto, la onda sinusoidal se considera una onda "fundamental" dado que es la expresión mínima a la que se puede descomponer cualquier sonido.

Otras Formas de Onda

Como ya se mostró, la forma de onda de una guitarra es relativamente compleja, a comparación de una forma de onda "simple" como la onda sinusoidal. Sin embargo, existen otras ondas que se las puede considerar "simples", dada su forma. Por ejemplo, las ondas cuadradas, triangulares o en forma de sierra:

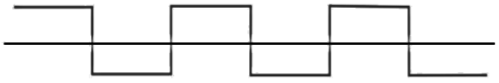

Figura 8. Onda cuadrada. Elaboración propia.

Figura 9. Onda triangular. Elaboración propia.

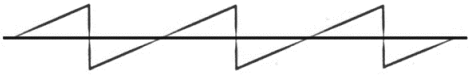

Figura 10. Onda sierra. Elaboración propia.

En el presente trabajo, se las denomina “simples” en el sentido que tienen formas geométricas que resultan fáciles de dibujar; más no son fundamentales, dado que son el resultado de la suma de una serie particular de ondas sinusoidales. Sin embargo, se las puede considerar las ondas básicas sobre las cuales empieza el proceso de síntesis de sonido, dado que son fáciles de generar, ya sea electrónica o digitalmente, motivo por el cual son usadas en gran parte de los sintetizadores modernos.

Finalmente, habiendo repasado las propiedades fundamentales y la composición del sonido, se puede pasar a revisar, a mayor profundidad, como logran las diferentes técnicas de síntesis su objetivo final, el cual es manipular las frecuencias que componen un sonido, cambiando así su forma de onda, y consiguiendo, por ende, modificar su timbre.

Síntesis Aditiva

La idea fundamental de los sintetizadores -referida reiteradamente en el presente trabajo- basada en la demostración de Helmholtz y en el postulado de Fourier, se conoce en la actualidad como “síntesis aditiva”. Se puede sumar ondas sinusoidales -por eso se llama “aditiva”- con distintas frecuencias para generar una forma de onda más compleja y, dado que la forma de onda es esencial en el timbre de un sonido, si se construye un aparato que genere varias ondas sinusoidales y las combine, se tiene un aparato para crear cualquier sonido que se desee (Shepard, 2013). Esto ya se indicó previamente, sin embargo, la pregunta es ¿Qué frecuencia y que amplitud deben tener las ondas sinusoidales que se pretenda combinar? Para responder a dicha pregunta es necesario entender el concepto de “armónico”, el cual ya se ha mencionado anteriormente, pero es oportuno expandir su explicación.

Si la forma de onda de una nota particular interpretada en una guitarra –o cualquier instrumento musical-, se descompusiera en varias ondas sinusoidales, cada onda sinusoidal que la compone tendría una frecuencia y una amplitud específicas. Sin embargo, al escuchar la guitarra, se percibe solamente la frecuencia de la nota interpretada. Dicha frecuencia se denomina "fundamental", esto debido a que el resto de las frecuencias que conforman a la onda, se forman a partir de dicha frecuencia fundamental; cuando una frecuencia es un múltiplo entero de la frecuencia fundamental, se la denomina "armónico" (Russ, 2009). El grupo de frecuencias que forman un sonido se conoce como espectro de frecuencias. En la *Figura 11* se puede apreciar el espectro de frecuencias de la nota de una guitarra, correspondiente a la forma de onda de la *Figura 5*:

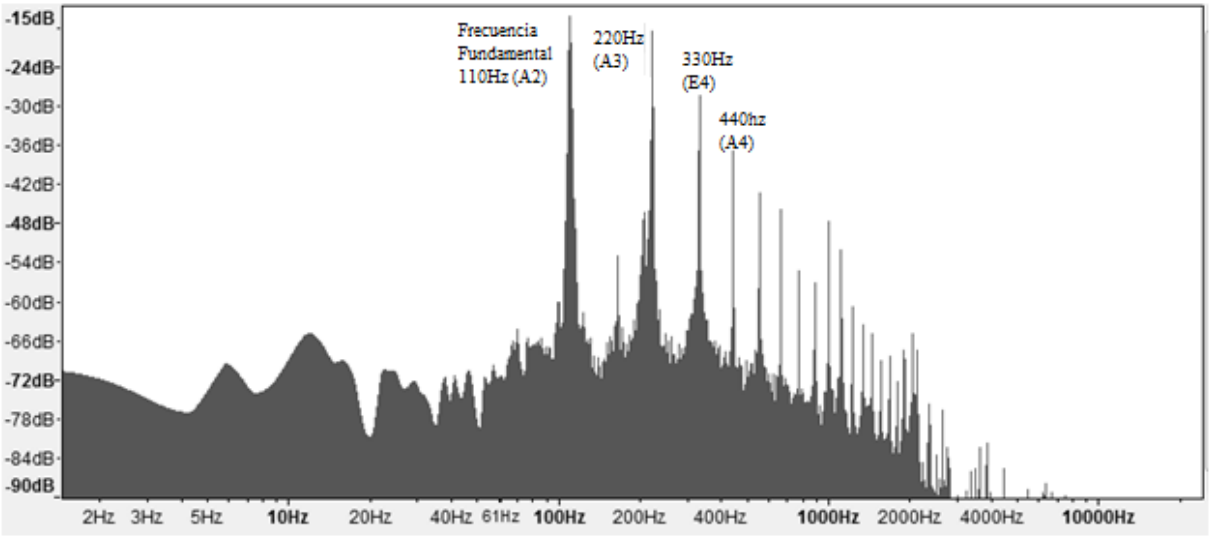

Figura 11. Espectro de frecuencias - Guitarra acústica. Elaboración propia.

Se puede observar como las frecuencias que más resaltan son múltiplos enteros de la frecuencia fundamental, justamente estas frecuencias son los armónicos. Dependiendo del número e intensidad de los armónicos que componen a una nota, el timbre es diferente. Por lo tanto, en la síntesis aditiva se busca generar y sumar varios armónicos a una nota fundamental para manipular el timbre. Por supuesto, no necesariamente se deben usar armónicos, el abanico de frecuencias de cual se puede escoger es prácticamente infinito; se pueden agregar frecuencias que no sean necesariamente múltiplos enteros de la frecuencia fundamental –denominadas "inarmónicos"- lo cual resultará en formas de onda complejas e interesantes. De hecho, la síntesis aditiva produce resultados más "exóticos" y "ricos" cuando se combinan tanto armónicos como inarmónicos (Shepard, 2013).

Síntesis Sustractiva

Si la síntesis aditiva consiste en sumar ondas sinusoidales, con frecuencias distintas, a una onda sinusoidal inicial con una frecuencia fundamental; la síntesis sustractiva es lo contrario, "es el proceso de (…) remover frecuencias de un sonido que ya es complejo" (Shepard, 2013, pág. 66). En otras palabras, quitar frecuencias de una onda que ya tiene inicialmente un contenido armónico importante. Es la técnica usada por Robert Moog en sus sintetizadores. Moog aprovechó que las ondas "simples", las cuales ya se mencionó en el presente trabajo, eran fáciles de generar electrónicamente y, más importante aún, su contenido de armónicos es amplio, lo cual las convirtió en la opción ideal para implementar en su sintetizador. Por ejemplo, en la *Figura 12* se puede apreciar el contenido armónico de una onda cuadrada oscilando a 440Hz:

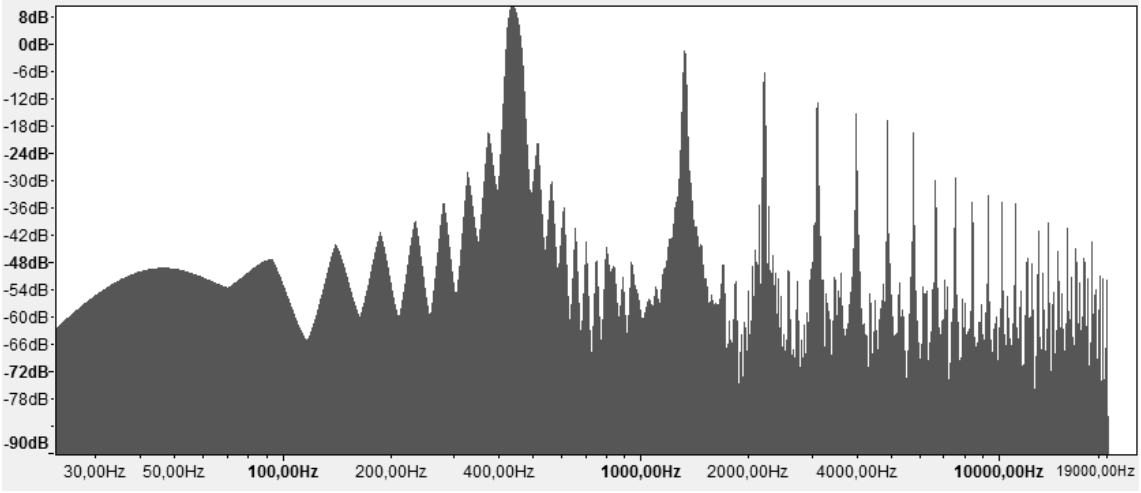

Figura 12. Espectro de frecuencias - Onda cuadrada. Elaboración propia.

Se aprecia claramente que existen una cantidad importante de armónicos que componen al sonido generado por una onda cuadrada. El componente más importante en la técnica de síntesis sustractiva es el filtro, dado que es el componente encargado de eliminar –o, mejor dicho, atenuar– las frecuencias que se requiera. Mientras más opciones brinde el filtro, mayor es la capacidad de manipulación del espectro de frecuencias y, por lo tanto, mayor es la capacidad de modificar el timbre del sonido.

Síntesis FM

La síntesis FM fue la técnica desarrollada por John Chowning e implementada en el Yamaha DX7, como ya se mencionó anteriormente. La técnica se basa en el uso de al menos dos ondas de sonido, una que funciona como "portadora" y otra como "moduladora"; la onda "moduladora" modula, valga la redundancia, la frecuencia de la onda portadora (Shepard, 2013). En palabras más simples, la onda moduladora indica cómo cambia, en el tiempo, la frecuencia de la onda portadora, como muestra la *Figura 13*:

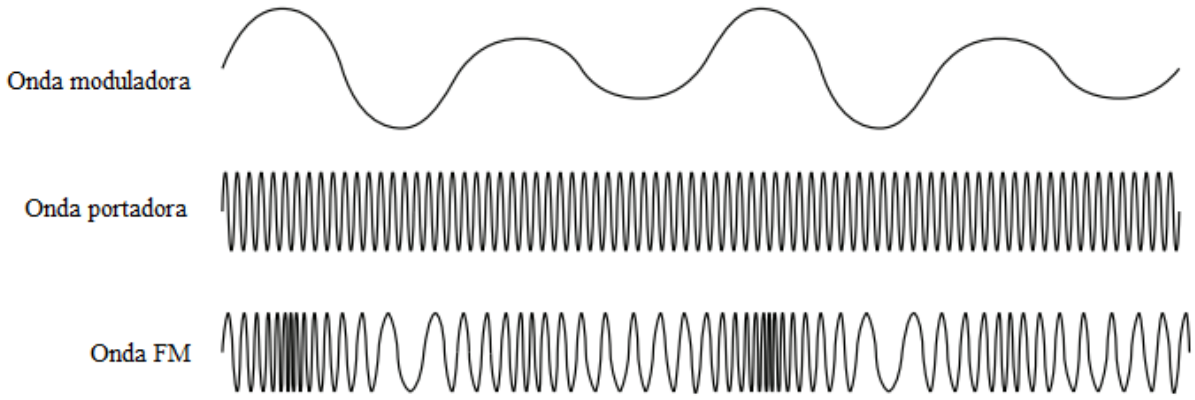

Figura 13. Síntesis FM. Shepard (2013).

"A medida que la onda moduladora causa, a la onda portadora, cambios rápidos en su frecuencia, se genera un conjunto complejo de bandas laterales" (Shepard, 2013, pág. 73), las cuales son la esencia de la síntesis FM. Las bandas laterales son frecuencias que se producen a cada lado de la frecuencia fundamental; el tipo de bandas laterales producidas depende de la relación entre la frecuencia de la onda portadora y la moduladora (Shepard, 2013).

Cuando se crean dichas bandas laterales, cambia el espectro de frecuencias del sonido y, por ende, cambia su timbre. "En la síntesis FM, algunas de estas bandas laterales pueden ser bastante fuertes e incluso ocultar a la frecuencia original de la onda portadora" (Shepard, 2013, pág. 73). Al producirse frecuencias a los dos lados de la frecuencia fundamental, la síntesis FM genera también sub-armónicos, -y sub-inarmónicos-, lo que resulta en sonidos distintos a los de las otras dos técnicas de síntesis previamente repasadas. Así mismo, se puede expandir la idea inicial y usar varias ondas moduladoras que se modulen entre sí y luego afecten a la onda portadora, generando un campo muy amplio de posibilidades para crear sonidos.

La ventaja de la síntesis FM es que puede producir frecuencias inarmónicas, pudiendo generar incluso sonidos de percusión. La desventaja es que, como menciona Shepard (2013), el resultado de la técnica es difícil de controlar dado que el mínimo cambio produce un impacto muy grande en el sonido. A pesar de que dicha desventaja podría ser más bien considerara, en cierto sentido, una ventaja, dado que justamente eso es lo que aumenta considerablemente las posibilidades de crear timbres nuevos a través de la síntesis FM.

Consideraciones

Antes de terminar el presente apartado, es importante considerar lo siguiente. Ya se ha detallado la importancia que tiene el espectro de frecuencias y la forma de onda en el timbre de un sonido, sin embargo, no es la única cualidad que se debe considerar. En realidad, hay otra característica clave a la hora de crear un sonido: el tiempo. La amplitud y el espectro frecuencias de un sonido complejo no son constantes en el tiempo. Dependiendo de la fuente del sonido, ciertas frecuencias disminuyen o aumentan a medida que pasa el tiempo desde que se generó el sonido hasta que se termina (Pirkle, 2015). Así mismo, la amplitud de la onda evoluciona a medida que el sonido avanza. Dicha variación de frecuencias y de amplitud resultan esenciales a la hora de dar forma al timbre final. En la *Figura 14* se ilustra este concepto en un piano que ejecuta la nota A1 (55Hz):

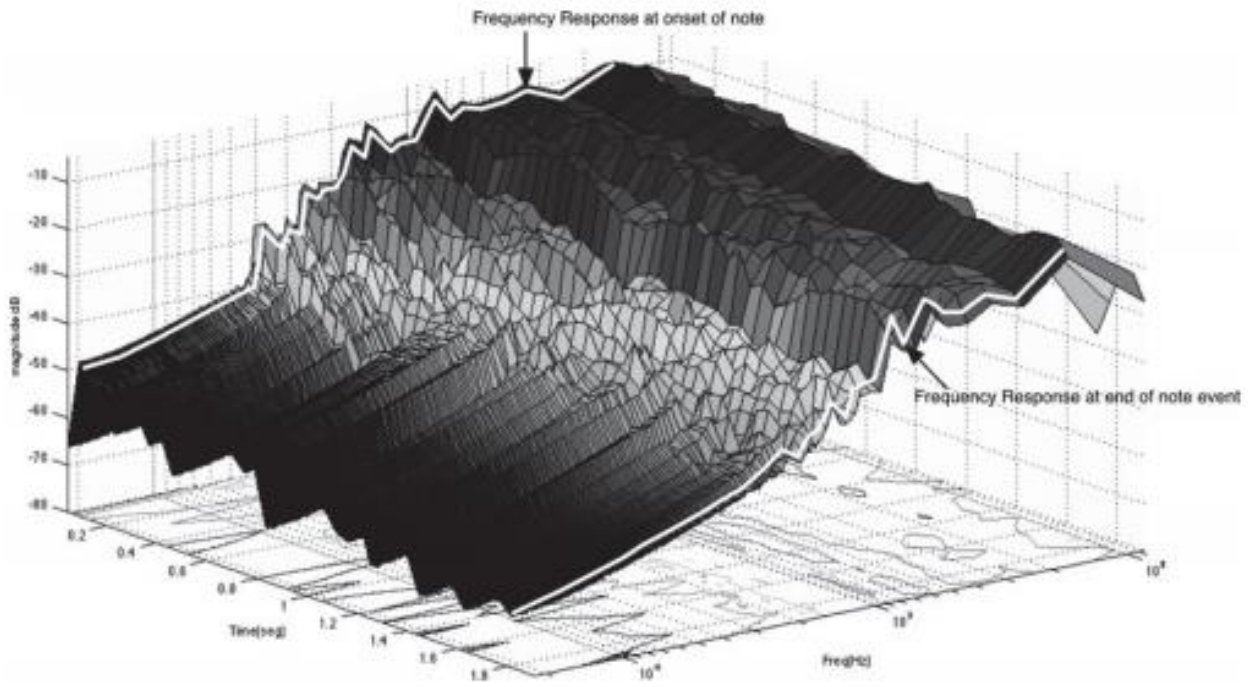

Figura 14. Evolución en el tiempo de amplitud y espectro de frecuencias de una nota de piano. Pirkle (2015).

Se puede apreciar como el perfil del espectro de frecuencias al inicio de la nota es distinto al perfil del final, siendo este cambio esencial en el timbre final, aportando “realismo” al sonido.

Por otro lado, también se mencionó a los armónicos como las frecuencias más importantes en los sonidos generados por instrumentos musicales melódicos. Si bien es cierto que los armónicos son, justamente, las frecuencias que más resaltan; esto no quiere decir que el resto de las frecuencias están completamente ausentes. Entre cada armónico existen infinitas frecuencias que tiene una intensidad baja o muy baja, pero la cual no es cero, y su presencia también aporta al timbre final del sonido. Esto se puede apreciar en la *Figura 11* y *Figura 12*, donde se ve que destacan los armónicos, pero todas las demás frecuencias intermedias también están presentes. Incluso, hay frecuencias con una amplitud casi igual a la del armónico, que son las frecuencias más próximas a este. Por ejemplo, si se parte de una frecuencia fundamental de 440Hz, el segundo armónico seria 880Hz, pero las frecuencias más cercanas, es decir, 878Hz, 879Hz, 881Hz, 882Hz, etc., van a tener una intensidad cercana a la del armónico, como se puede apreciar en los gráficos mostrados. También se puede observar que existen frecuencias por debajo de la frecuencia fundamental, tanto sub-armónicos como sub-inarmónicos.

Esto añade una capa de complejidad muy importante a la hora de tratar de emular el sonido de un instrumento “real”, dado que, por ejemplo, se necesitaría una cantidad demasiado alta de osciladores en la síntesis aditiva; o un filtro sumamente detallista en la síntesis sustractiva. Justamente por esta complejidad, los sintetizadores que buscan recrear, lo más apegado a la realidad posible, a otros instrumentos, no solo que son diseñados con ese único objetivo en mente, sino que se especializan en un tipo particular de instrumentos, ya sea instrumentos de viento, pianos, percusión o cuerdas; y aplican una técnica particular conocida como síntesis de modelado físico (Shepard, 2013).

Es por este motivo que la mayor parte de sintetizadores no se distinguen por emular al detalle a otros instrumentos, sino más bien por generar sonidos cercanos a dichos instrumentos, pero con un toque “artificial” -además de sonidos completamente distintos, evidentemente. Es precisamente por esto que el sintetizador es un instrumento reconocido culturalmente como “electrónico”, dado que, a pesar de tener la posibilidad de crear una gran variedad de sonidos, todos esos sonidos tienen esa particularidad que distingue al instrumento. Incluso, a partir de su invención, surgió un nuevo género musical, la música electrónica.

Por último, hay que destacar que, en la actualidad, gracias a la capacidad de procesamiento disponible, existen nuevas y variadas técnicas de síntesis, como la síntesis granular o la síntesis basada en “samples”, por ejemplo. Sin embargo, las técnicas previamente mencionadas son las bases de la síntesis de sonido y todavía se mantienen muy vigentes en los sintetizadores modernos. Además, serán las usadas posteriormente en el diseño del sintetizador implementado en el presente trabajo.

COMPONENTES DE UN SINTETIZADOR DE SONIDO

Al existir varias técnicas de síntesis, cada sintetizador de sonido puede escoger cual técnica utiliza, o incluso si utiliza una combinación de varias técnicas. Dependiendo de la elección, los componentes que conformen al sintetizador serán distintos. Esto significa que no se puede definir realmente una lista de componentes que sea aplicable a todo sintetizador, pues cada sintetizador tiene un diseño distinto. Sumado a esto, cada fabricante tiende a usar su propia denominación para los componentes que decide usar, generando que no exista una estandarización ni en cuanto a número ni a nombres de los componentes. Sin embargo, para solucionar esto según Pirkle (2015) se pueden agrupar todos los componentes en tres grandes grupos: fuentes de sonido, modificadores y controladores.

A continuación, se presenta una descripción de estos tres grupos, donde se detallan, usando las denominaciones más comunes, algunos componentes específicos que pertenecen a cada grupo. Los componentes descritos están más presentes en sintetizadores que implementan la síntesis sustractiva como su técnica de síntesis fundamental; sin embargo, también son muy comunes en otros sintetizadores. Dichos componentes, a su vez, serán usados en el diseño del sintetizador implementado en el presente trabajo.

Fuentes de Sonido

Se puede considerar cono fuentes de sonido a aquellos componentes que generan las ondas iniciales sobre las cuales se comienza el proceso de síntesis, ya sean estas ondas sinusoidales, cuadradas, triangulares, etc., u ondas más complejas.

Osciladores

"Un oscilador genera una onda sin ningún input" (Pirkle, 2015, pág. 4). Aunque siendo un poco específicos, un oscilador necesita la información de frecuencia de la onda para generarla, ya sea electrónica o digitalmente. Con los osciladores comienza el proceso de síntesis de sonido. Sin embargo, un oscilador puede generar solo una nota, por lo tanto, un sintetizador puede tener uno o varios osciladores, dependiendo de la o las técnicas de síntesis que use. Eso sí, un oscilador es, usualmente, capaz de generar varias formas de onda simples.

Generadores de Ruido

El ruido no es más que fluctuaciones rápidas y completamente aleatorias de la frecuencia y la amplitud de una onda (Shepard, 2013). Existen componente dedicados justamente a generar ruido, dado que este tiene algunas aplicaciones interesantes a la hora de sintetizar sonido como, por ejemplo, "cuando se lo mezcla con un oscilador común, especialmente al inicio del sonido, el ruido puede agregar un poco de realismo a un sonido o aclarar el ataque de forma que el sonido destaque en la mezcla" (Shepard, 2013, pág. 41). Por este motivo, los generadores de ruido son un componente común en cualquier tipo de sintetizadores.

Modificadores

Estos componentes son aquellos que modifican cualquiera de las propiedades de las ondas generadas por las fuentes de sonido, ya sea su amplitud, frecuencia o forma. De estos componentes depende, en gran medida, el sonido final que emita un sintetizador.

Amplitud

La modificación más simple a una onda es manipular su amplitud. Aumentar o disminuir su amplitud se traduce en un cambio de su volumen. Inicialmente se usaba un control por voltaje para modificar la amplitud; en la actualidad es muy fácil hacerlo digitalmente (Pirkle, 2015).

Filtro

El filtro es uno de los componentes más importantes de un sintetizador, dado que sirve para "modificar directamente el espectro de frecuencias, consiguiendo así refinar el timbre del sonido" (Shepard, 2013, pág. 99). Mientras más opciones tenga el filtro de un sintetizador, mayor será la gama de timbres que se puede crear. Por lo tanto, es esencial tener claro los conceptos sobre cómo funciona el proceso de filtrado.

Todo filtro tiene un rango especifico de frecuencias, llamado "passband" o "banda de paso", que pasan a través del filtro inalteradas, y un rango de frecuencias que el filtro impide que pasen, conocido como "stopband" o "banda de rechazo". Dado que ningún filtro puede realizar un cambio brusco desde la banda de paso hacia la banda de corte, existe un área de transición entre las dos bandas en la cual las frecuencias pueden pasar, pero su amplitud es gradualmente reducida o atenuada, a medida que el área de transición se acerca a la banda de rechazo (Shepard, 2013, pág. 100).

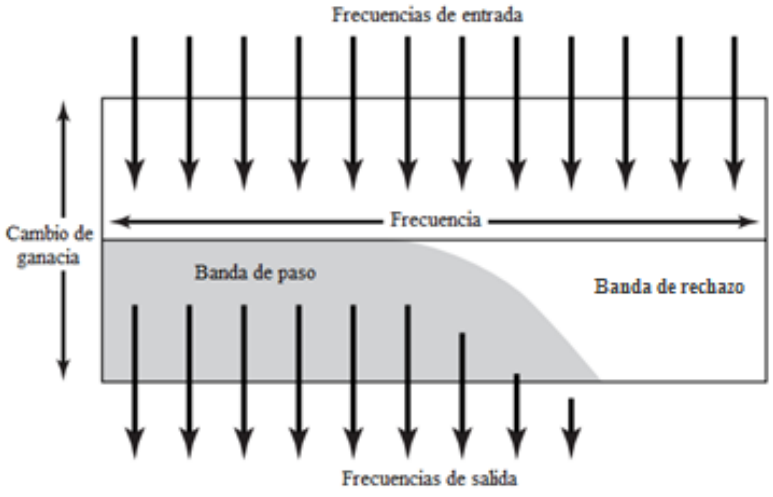

Figura 15. Funcionamiento de un filtro de audio. Shepard (2013).

El tipo de filtro depende de la ubicación de la banda de paso. Los 4 filtros más comúnmente encontrados en un sintetizador son: pasa-altos, pasa-bajos, pasa-banda, banda-eliminada (Shepard, 2013). El nombre del filtro indica las frecuencias que deja pasar o que atenúa, un filtro pasa-bajos deja pasar las frecuencias por debajo de una frecuencia específica, por ejemplo.

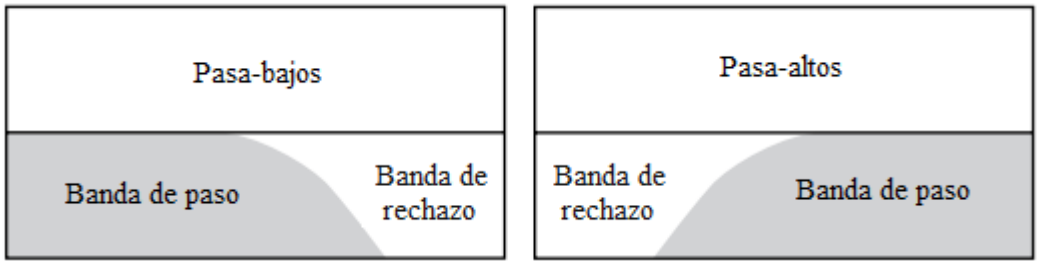

Figura 16. Filtro pasa-bajos y pasa-altos. Shepard (2013).

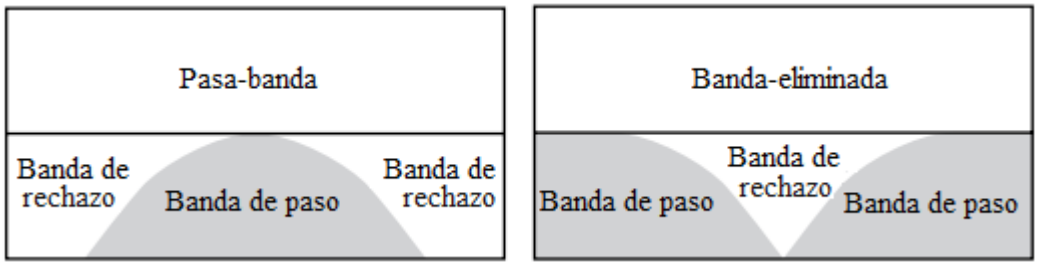

Figura 17. Filtro pasa-banda y banda-eliminada. Shepard (2013).

Basta con observar la forma de cada filtro para entender, al menos de manera general, las frecuencias que atenúan. Por supuesto, dichas diferencias son más apreciadas cuando se escucha a los diferentes filtros actuando sobre una onda. Por otro lado, hay dos características esenciales que todo filtro tiene: la frecuencia de corte y la pendiente.

Como ya se mencionó, existe un área de transición donde las frecuencias se van atenuando gradualmente. La pendiente indica que tan rápido ocurre la transición entre la banda de paso y la

banda de rechazo; en otras palabras, indica el tamaño del área de transición. Dependiendo del tipo de filtro, puede haber una o dos -o incluso más- pendientes. Por ejemplo, el filtro pasa-bajos tiene una pendiente y el pasa-banda tiene dos. Cuando el filtro posee 2 pendientes se usa el término "ancho de banda", lo cual indica la distancia entre las dos pendientes. Cada filtro puede implementar su propio método para controlar la pendiente y el ancho de banda, pero el concepto es el mismo (Shepard, 2013). Evidentemente, dependiendo del valor de la pendiente, el resultado del filtro será distinto.

Dependiendo del número de pendientes sea usa un método distinto para medir el valor de la pendiente. En el caso de una sola pendiente, se usa la medida de decibel por octava o dB/Octava; dicha medida indica el promedio de la tasa de cambio entre el inicio de la transición hasta el final. En el caso de dos pendientes, se mide el ancho de banda usando octavas y fracciones de octavas como medida de distancia; la medición se la realiza desde el punto donde inicia la transición (Shepard, 2013).

Figura 18. Diferencia entre pendientes medidas con dB/Octava. Shepard (2013).

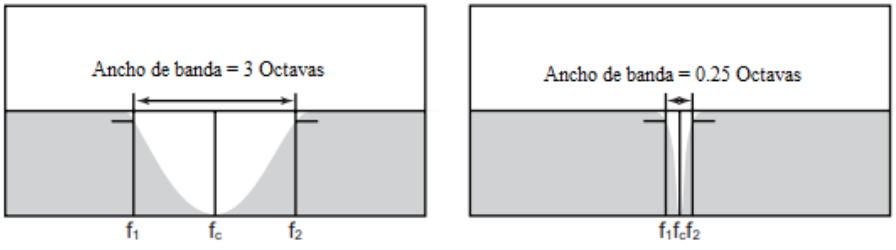

Figura 19. Diferencia entre pendientes medidas con ancho de banda. Shepard (2013).

Por su parte, la frecuencia de corte o “cutoff” es la frecuencia en la cual el filtro empieza la transición desde su banda de paso hacia la banda de corte. Cuando el filtro tiene una sola transición, dado que dicha transición es gradual, es muy difícil determinar exactamente la frecuencia donde empieza dicha transición. Para resolver esto, se definió al punto donde las frecuencias se atenúan por -3dB como el “cutoff”. Si el filtro tiene dos transiciones, la frecuencia que este en el centro del valle o de la cresta se denomina frecuencia central. Sin embargo, muchos sintetizadores usan el término “cutoff” en los dos casos (Shepard, 2013).

Figura 20. Frecuencia de corte y Frecuencia central de un filtro. Shepard (2013).

El último concepto que se considera importante revisar es la resonancia. La resonancia no es necesariamente una propiedad de los filtros, sino más bien una técnica la cual se aplica a un filtro, y genera un sonido muy particular, aumentando las capacidades de manipular el sonido de dicho filtro. Al aumentar la resonancia, lo que se hace es aumentar la amplitud de la frecuencia del punto de corte, causando que la pendiente se vuelva más empinada (Shepard, 2013), como se muestra en la siguiente imagen:

Figura 21. Resonancia de un filtro. Shepard (2013).

Esto produce un efecto sonoro muy interesante, el cual rápidamente se volvió muy popular en los sintetizadores y es parte de su sonido característico.

Envolvente ADSR

Se mencionó previamente que la evolución de un sonido en el tiempo es una característica importante del timbre de dicho sonido. Tanto la amplitud como las frecuencias de una onda cambian desde que el sonido se inicia hasta que este termina. Precisamente, esta evolución en el tiempo es lo que busca emular una envolvente; centrándose, principalmente, en el comportamiento de la amplitud de la onda.

El modelo más usado en las envolventes es el modelo ADSR: Attack, Decay, Sustain y Release. En español: ataque, decaimiento, sostenimiento y relajación. Sin embargo, dado que es mucho más común encontrar estos nombres en inglés en la gran mayoría de sintetizadores, se usará dicha denominación.

Attack es el tiempo que tarda un sonido en alcanzar su máxima amplitud desde el momento en que se inicia; es decir, desde que es generado, donde tiene una amplitud igual a cero. Decay es el tiempo que tarda el sonido desde que alcanzó su amplitud máxima hasta que baja a una amplitud estable. Sustain es, justamente, la amplitud estable a la cual se mantiene el sonido. Finalmente, Release es el tiempo que tarda el sonido en desaparecer completamente una vez que se dejó de producir; por ejemplo, cuando se deja de aplastar la tecla de un piano (Russ, 2009).

Cada instrumento produce sonidos con un comportamiento distinto en el tiempo. Por ejemplo, un piano tiene un tiempo de attack muy corto, un decay igualmente corto, y un sustain bajo, comparado con su amplitud máxima. Un bombo de batería, por otro lado, tiene también un attack muy corto, pero prácticamente no tiene sustain, además de un release muy corto. Por este

motivo, un envolvente ADSR es muy importante a la hora de agregar realismo y dinámicas al sonido final, y constituye un componente prácticamente indispensable en cualquier sintetizador. En la *Figura 22* y *Figura 23* se puede apreciar, justamente, la diferencia de evolución de la amplitud en el tiempo de los ejemplos dados, y como el modelo ADSR representa la evolución de la amplitud de una onda:

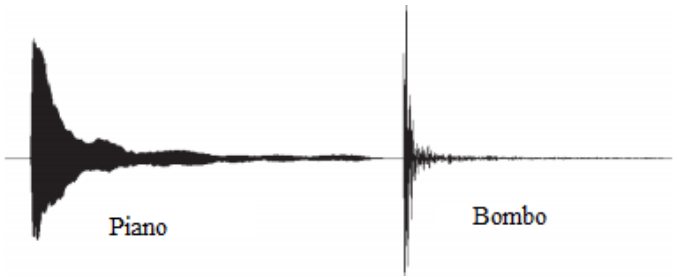

Figura 22. Evolución amplitud en el tiempo - Piano y Bombo. Shepard (2013).

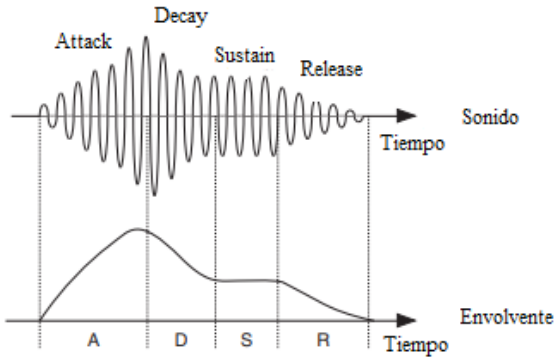

Figura 23. Representación visual modelo ADSR. Russ (2009).

En la actualidad, existen modelos de envolventes que agregan más etapas a parte de las ADSR. Sin embargo, en muchos casos, dichos modelos todavía llaman a sus envolventes como ADSR, debido a la popularidad y familiaridad con la que esta cuenta (Shepard, 2013).

Si bien es cierto el envolvente ADSR es aplicado sobre todo a la amplitud de la onda, es importante considerar que este no es el único parámetro al cual puede modificar. A la final, el envolvente ADSR es un modelo que representa el cambio de un valor en el tiempo, por lo que puede ser aplicado realmente a cualquier parámetro del sintetizador. De hecho, otro uso muy común para el envolvente ADSR es aplicarlo al cutoff del filtro, para generar un efecto de filtrado dinámico.

LFO

Low Frequency Oscillators u Oscilador de Baja Frecuencia. Los LFOs son un tipo particular de osciladores, los cuales no se los considera dentro del grupo de fuentes de sonido por una sencilla razón, no se los puede escuchar. Según Shepard (2013), los LFOs son osciladores con una frecuencia, usualmente, entre 0.25 Hz y 5 Hz, es decir, que están fuera del rango audible de los seres humanos.

Pero entonces, ¿Cuál es su uso? Su objetivo es de modular algún parámetro del sintetizador. De la misma forma que, en la síntesis FM, la onda moduladora indica como cambia en el tiempo la frecuencia de la onda portadora, los LFOs indican como cambia en el tiempo un parámetro determinado. Dependiendo el parámetro al cual se afecte, el efecto será distinto. Por ejemplo, si se lo aplica a la amplitud de un oscilador, se obtiene el efecto conocido como “tremolo”; si se lo aplica a la frecuencia de un oscilador, dado que la frecuencia del LFO es baja, no se generan las bandas laterales de la síntesis FM, sino que se obtiene el efecto “vibrato” (Shepard, 2013).

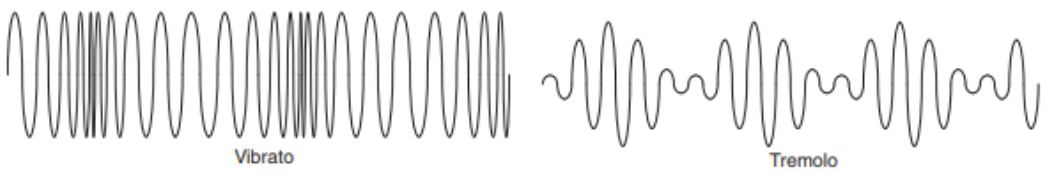

Figura 24. Efecto de vibrato y tremolo sobre una onda. Russ (2009).

Esta capacidad de los LFOs de ser aplicados a cualquier parámetro resulta en que pueden ser considerados tanto modificadores como controladores, dependiendo de si afectan directamente a la onda generada por las fuentes de sonido o si afectan a algún parámetro de los modificadores. Para simplificar el diseño, en el presente trabajo, se los considera solamente como modificadores.

Al igual que los osciladores comunes, los LFOs usualmente pueden generar distintas formas de onda de simples. Los LFOs son componentes muy populares en los sintetizadores dada su facilidad de uso y, sobre todo, posibilidad de experimentación.

Controladores

El último grupo de componentes son los controladores. “Son los artefactos que controlan los parámetros de las fuentes o de los modificadores” (Pirkle, 2015, pág. 2). Son, por ejemplo, los componentes que indican a un oscilador que forma de onda y nota generar, o a un filtro cuál es frecuencia de corte o su resonancia. Los controladores son las partes visibles de un sintetizador y que el usuario puede manipular en tiempo real, como las teclas de un piano o una perilla, para alterar los valores de los demás componentes.

Los controladores que más opciones brindan, en la actualidad, son los controladores MIDI.

MIDI

Musical Instrument Digital Interface o MIDI, por sus siglas en inglés, es un protocolo que “proporciona una interfaz para el intercambio de información entre instrumentos musicales electrónicos y computadoras” (Russ, 2009, pág. 67). MIDI es el conjunto de estándares que permiten que las computadoras y los instrumentos musicales electrónicos puedan comunicarse

entre sí. Un concepto erróneo muy común alrededor de MIDI es creer que funciona transportando una señal de audio. En realidad, “cuando se conectan dos dispositivos a través de MIDI (…) no se está enviando un sonido; se envían instrucciones” (Shepard, 2013, pág. 152).

En otras palabras, a través MIDI no se envía la onda de sonido como tal, se envía la información de dicha onda de sonido. Esa información puede ser la frecuencia, amplitud, el momento en que se presionó una tecla, la intensidad con la que se la presionó, el momento que se la deja de presionar, etc. Dicha información llega a un dispositivo el cual genera un sonido con las características enviadas. Un controlador MIDI no es, por sí solo, capaz de producir sonidos, necesita de un dispositivo receptor que realice esa tarea.

En el caso de los sintetizadores digitales, el uso de un controlador MIDI permite manipular todos los parámetros, tanto de las fuentes de sonido como de los modificadores. Lo más lógico es usar un controlador MIDI que disponga de un teclado de piano para enviar la información de la nota presionada, como su frecuencia, intensidad y duración, a los osciladores para que estos generen el sonido; y varias perillas asignables a los distintos parámetros de los distintos modificadores y fuentes de sonido, de forma que se pueda usar los componentes del sintetizador en tiempo real. El sintetizador de sonido desarrollado en el presente trabajo implementará, justamente, un controlador MIDI para su manejo.

CAPÍTULO 3: DISEÑO DEL SINTETIZADOR DE SONIDO DE SOFTWARE

EJES DE DISEÑO

Una vez finalizado el repaso a la historia, funcionamiento y componentes de un sintetizador de sonido, en el presente capítulo se pretende consolidar toda la teoría, previamente descrita, en el diseño de un sintetizador de sonido de software que cumpla los objetivos del presente trabajo.

Se mencionó, en el apartado de análisis histórico del Capítulo 2, la importancia de definir limites antes de desarrollar el diseño del sintetizador, de modo que se pueda marcar un camino a seguir donde se logre aprovechar al máximo las posibilidades tecnológicas de la época actual, sin que esto signifique elaborar un diseño sumamente complejo. Para lograr esto, se ha definido a dichos límites dentro de tres ejes principales, los cuales, se considera, agrupan las principales necesidades que debe satisfacer un sintetizador de sonido de software.

Pero antes de hablar de dichos ejes, se debe empezar por el límite el cual es, tal vez, el más importante de todos ¿Para quienes va dirigido el sintetizador? Primero se debe definir el público objetivo y a partir de ahí empezar a realizar el diseño. En este caso, es claro que el sintetizador va dirigido a músicos, pues son ese grupo de personas quienes los van a usar, pero ¿Va dirigido a todos los músicos? Evidentemente, no. Existen categorías de músicos, separados por varios factores como: instrumento que ejecutan, género musical de preferencia, nivel de destreza con el instrumento, nivel de conocimiento musical, músicos compositores, músicos intérpretes, músicos aficionados, músicos profesionales, etc. Es esencial delimitar a que grupo de músicos está enfocado el sintetizador de sonido desarrollado en el presente trabajo.

El sintetizador de sonido es un instrumento muy particular, dado que tiene una curva de aprendizaje mayor a la de otros instrumentos, no en cuanto a nivel de destreza, sino más bien en relación con la teoría que hay aprender y manejar para entender realmente como funciona. Ya se

mencionó previamente como cada sintetizador tiene su diseño y componentes particulares, además de que no hay un acuerdo general en cuanto a la denominación de cada componente y cada fabricante usa sus propias denominaciones. Todo esto resulta en confusión en el músico que intenta aprender desde cero a utilizar un sintetizador. Justamente, es a este grupo de músicos al que se pretende llegar con el presente trabajo, músicos aficionados o semiprofesionales que busquen entrar al mundo de los sintetizadores, con poca o ninguna experiencia en el campo de la síntesis de sonido. Lo que se busca es que lo puedan hacer de manera rápida y sencilla, sin sentirse abrumados por la cantidad de información que existe alrededor de este instrumento.

Dicho esto, los tres ejes determinados, alrededor del grupo objetivo descrito previamente, son: manejo intuitivo, gama de sonidos y eficiencia. A continuación, se presenta una descripción de cada eje, detallando los objetivos de cada uno, así como la justificación para su consideración y su impacto en el diseño.

Manejo Intuitivo

Es evidente que si un músico pretende aprender a usar un sintetizador debe, necesariamente, atravesar por un proceso de aprendizaje teórico sobre cómo funciona el instrumento. Un sintetizador orientado a alguien que no tiene conocimientos -o tiene conocimientos muy básicos- sobre síntesis no deber ser un sintetizador que pretenda eliminar el proceso de aprendizaje, sino más bien, debe ser uno que busque guiar al músico de la mejor manera a través de dicho proceso. Entonces, la pregunta aparece por si sola: ¿Cómo se puede guiar a un músico de modo que entienda más rápido como funciona un sintetizador?

Se considera que lo más importante es definir secciones fácilmente identificables. Como ya se explicó en el capítulo anterior, todo sintetizador tiene 3 tipos de componentes: fuentes de sonido, modificadores y controladores. El objetivo de un diseño que sirva como guía de aprendizaje debe ser separar claramente dichos tipos de componentes, de modo que el usuario se familiarice rápidamente con la idea de que la síntesis es un proceso donde cada componente cumple un objetivo específico. Un músico usa su oído como herramienta principal, por lo tanto, no necesita -al menos al empezar en el mundo de la síntesis- entender nombres y conceptos, le basta con distinguir como suena cada componente para empezar a entender cómo usar el sintetizador. Unos componentes generan el sonido inicial, otros lo modifican y otros son usados para controlar la generación y las modificaciones. Sin necesariamente entender a profundidad que es un oscilador, un filtro o un controlador MIDI; el músico puede entender rápidamente que uno genera el sonido, el otro cambia su timbre y con el último controla la nota que quiere que se genere. De esta forma el músico se familiariza con la funcionalidad de cada componente, y no con su nombre; de manera que, si decide usar otro sintetizador, no tendrá confusiones debido a las distintas denominaciones de los componentes.

Secciones bien definidas le permiten al músico manejar intuitivamente el sintetizador y experimentar con los componentes, sin necesidad de un profundo repaso teórico previo. Dichas secciones no necesariamente deben estar separadas por nombres, es más importante que estén separadas visualmente, es decir, que a primera vista se identifique que existen grupos de componentes agrupados de cierta manera, de forma que la idea de que cada grupo es responsable de algo distinto sea interiorizada más rápidamente.

Gama de Sonidos

La cualidad más importante de cualquier instrumento musical es el sonido. Cuando una persona interpreta una guitarra no está pensando en los armónicos que se generan gracias a la madera, cuerdas, caja de resonancia, etc. Solo se interesa en si el producto final, el sonido, le resulta agradable o no. Una persona que escucha por primera vez un sintetizador piensa exactamente lo mismo. Sin embargo, un sintetizador no produce solamente un sonido, produce un conjunto amplio de sonidos. Por lo tanto, ¿Cómo logra un sintetizador ofrecer no solo un sonido agradable, sino una gama de sonidos lo suficientemente amplia de forma que pueda ser usada en distintos contextos?

Para responder a la pregunta se considera que la solución es no usar solamente una técnica de síntesis, sino combinar varias técnicas. Tanto síntesis aditiva, como síntesis sustractiva y síntesis FM producen resultados distintos y pueden ser usadas en conjunto, de modo que se aproveche las posibilidades de cada una y se expanda la gama de sonidos final. Por otro lado, existen parámetros que son muy sencillos de implementar y, con el paso del tiempo, se han ido agregando a los componentes de los sintetizadores, lográndose mantener vigentes, dado que son simples de usar y aportan mucho al sonido final. Ya se mencionó a dos de estos parámetros en el capítulo anterior, la resonancia del filtro y un oscilador que genere ruido. Por supuesto, existen otros que se puede considerar, como, por ejemplo, el llamado “fine tuning” que permite desafinar un poco a un oscilador, produciendo, al combinarlo con otro oscilador sin desafinar, un efecto llamado “chorus”; o la implementación de un sub-oscilador, que es un oscilador que genera una onda cuadrada una octava debajo de la nota presionada, usado para agregar cierta “profundidad” al sonido. Por último, es esencial el uso de LFOs, debido a que al ser osciladores que modulan a otros parámetros, mientras a más parámetros se los pueda asignar, más se podrá experimentar

con el sintetizador y más complejo serán los sonidos resultantes. Todo esto resulta en un sintetizador capaz de ofrecer un amplio conjunto de sonidos posibles, lo que se traduce en un sintetizador que puede ser usado en diferentes contextos y géneros musicales.

Ahora bien, volviendo a considerar el hecho de que el grupo objetivo son músicos que tienen poco o nulo conocimiento en síntesis de sonido, ¿Cómo guiar al músico de modo que pueda explorar toda la gama de sonidos? La respuesta es sencilla: a través de presets. Los presets son configuraciones preestablecidas de los parámetros de los componentes del sintetizador. Se puede tener un preset que simule un bajo, otro donde se tenga el sonido de un piano y otro donde se genere un sonido de nave espacial. La idea básica es definir un número determinado de presets, que recorran la gama de sonidos que el sintetizador pueda generar, de modo que el usuario pueda, fácil y rápidamente, obtener diferentes sonidos y, a partir de estos, empezar a configurar los suyos propios e irse familiarizando con el funcionamiento del proceso de síntesis.

Eficiencia

Ahora bien, ¿Cómo agrupar varias técnicas de síntesis y mantener secciones de componentes definidas sin necesariamente complicar el diseño? La respuesta es: tener un número limitado de componentes, pero optimizar su uso; es decir, que cada componente se encuentre, de una manera u otra, involucrado en el proceso de síntesis, logrando así una alta eficiencia de uso. No es poco común encontrar sintetizadores con una gran cantidad de componentes -los cuales, por supuesto, ofrecen una gama de sonidos sumamente amplia- pero, para el músico que recién empieza en el mundo de la síntesis, un sintetizador con tantas opciones resulta muy abrumador, lo que se traduce en un uso limitado del mismo, dado que muchos de los componentes no son usados en lo más mínimo, desperdiciando todo el potencial del sintetizador. En el peor de los casos, todo lo

anterior genera una pérdida de interés del músico en el instrumento, debido a su aparente complejidad.

Por lo tanto, un diseño donde todos los componentes -o por lo menos la gran mayoría- participen directamente en el proceso de síntesis y en el sonido final, involucra más al músico, dado que le genera la sensación de estar aprovechando todo o gran parte del potencial del instrumento; al existir un número limitado de componentes, esto le permite entender más rápidamente que hace cada componente y lo motiva a experimentar por sí mismo. Un diseño eficiente es, a su vez, un diseño intuitivo y con una gama amplia de sonidos.

Una forma de optimizar el uso de componentes es usar un oscilador tanto para generar sonido, como para modular a otro oscilador, y posteriormente sumar las ondas iniciales y resultantes, implementando así un tipo de hibrido entre síntesis FM y síntesis aditiva. Posteriormente se pasa el sonido por un filtro y un envolvente ADSR y se tiene síntesis sustractiva. Incluso los LFOs se los puede usar para modular la onda moduladora y generar una onda FM más compleja, expandiendo así su funcionalidad. Basta con 5 componentes bien optimizados, para implementar los 3 tipos de síntesis revisadas en el presente trabajo.

Es importante, en este punto, resaltar la ventaja que representa que el sintetizador de sonido del presente trabajo sea desarrollado como software, dado que de esa manera se puede conseguir, de una forma más sencilla, una mayor eficiencia en cada componente, ya que se puede manipular a las ondas generadas de una forma más directa y combinarlas más fácilmente.

ESTRUCTURA GENERAL

A partir de la definición los ejes de diseño, se seleccionaron las siguientes técnicas de síntesis y los siguientes componentes para la elaboración del sintetizador de sonido de software.

Técnicas de Síntesis Implementadas

Síntesis Sustractiva: Esta técnica de síntesis será la principal del sintetizador, dada su facilidad de implementación. Para ello, se usarán dos osciladores principales, los cuales generarán ondas simples como base del proceso de síntesis. Se implementará un filtro para alterar el timbre, así como una envolvente ADSR para modelar la evolución de la amplitud de la onda en el tiempo.

Síntesis FM: Se aprovechará a los dos osciladores principales para también generar una onda FM. Se podrá configurar a uno como onda moduladora y al otro como onda portadora. Así mismo, para expandir más las posibilidades de la síntesis FM y mantener eficiencia de uso, se podrá configurar a los LFOs para que modulen a la onda moduladora, para lograr así más posibilidades de generar sonidos.

Síntesis Aditiva: Por último, tanto las ondas iniciales generadas por los osciladores como la onda FM serán sumadas para obtener un sonido más complejo previo a la aplicación del filtro y de la envolvente ADSR. Así mismo se implementará dos osciladores más, un sub-oscilador y un generador de ruido, de los cuales solo se podrá configurar su amplitud, pero también se sumarán a las ondas previamente mencionadas. Cabe aclarar que sumar todas las ondas generadas como se propone no es estrictamente síntesis aditiva, al menos no como se la definió en el Capítulo 2 del presente trabajo. Sin embargo, se usará dicha denominación dado que el principio es el mismo, sumar ondas simples para generar ondas más complejas.

Componentes

A continuación, se especifican los componentes principales a usar, así como los parámetros que se podrán modificar de cada uno. También se indica el símbolo con el que se identificará a cada componente. Dichos símbolos son basados en la simbología usada en el libro *Designing Software Synthesizer Plug-Ins in C++*.

Fuentes de Sonido

- Oscilador 1: Se podrá seleccionar la forma de onda a generar: sinusoidal, sierra, triangular o cuadrada. Se podrá modificar la nota indicada por el controlador MIDI y generar una onda con una nota superior, dependiendo el semitono escogido.

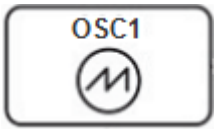

Figura 25. Oscilador 1. Pirkle (2015).

- Oscilador 2: Se podrá seleccionar la forma de onda a generar: sinusoidal, sierra, triangular o cuadrada. Se podrá modificar la nota indicada por el controlador MIDI y generar una onda con una nota superior, dependiendo el semitono escogido. Se podrá desafinar ligeramente a la nota original del Oscilador 2, logrando un efecto de "chorus" al combinarla con el otro oscilador sin desafinar. También se podrá seleccionar si se desea que la onda generada por el segundo oscilador module a la del primer oscilador, generando una nueva onda FM.

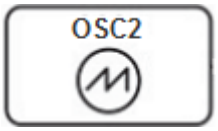

Figura 26. Oscilador 2. Pirkle (2015).

- Sub-Oscilador: Generará una onda cuadrada y su frecuencia es una octava debajo de la dada por el controlador MIDI.

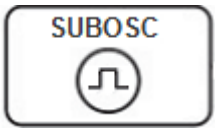

Figura 27. Sub-Oscilador. Elaboración propia.

- Generador de ruido: Es un oscilador que genera frecuencias aleatorias, produciendo un sonido de ruido. Se lo denominará “Noise”.

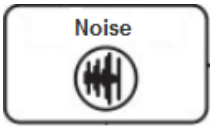

Figura 28. Generador de Ruido. Pirkle (2015).

Modificadores

- Amplitud: Se podrá modificar la amplitud del Oscilador 1, Oscilador 2, Sub-Oscilador y Generador de Ruido previo a sumar todas las ondas.

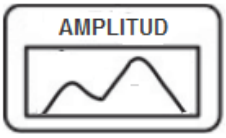

Figura 29. Modificador de amplitud. Elaboración propia.

- Envolvente ADRS – Cutoff: Se podrá modular el “cutoff” del filtro usando esta envolvente ADSR. Se podrá configurar el attack, decay, sustain y release. La envolvente siempre estará activa y se la denominará “Envolvente de corte”.

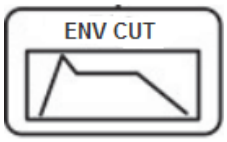

Figura 30. Envolvente ADSR de corte. Pirkle (2015).

- Filtro: Se podrá escoger el tipo de filtro entre: pasa-altos, pasa-bajos y pasa-banda; así como el “cutoff”, y el nivel de resonancia a aplicar. El filtro siempre estará activo y afectará a la suma de todas las ondas generadas por las fuentes de sonido.

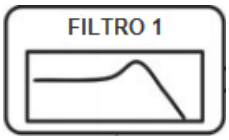

Figura 31. Filtro de audio. Pirkle (2015).

- Envolvente ADSR – Amplitud: Se podrá configurar el attack, decay, sustain y release. La envolvente siempre estará activa y afectará a la amplitud de la suma de todas las ondas generadas por las fuentes de sonido. Se la denominará “Envolvente de amplitud”.

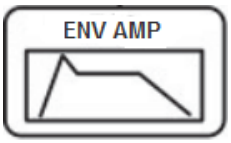

Figura 32. Envolvente ADSR de amplitud. Pirkle (2015).

- Master: Afectará a la amplitud de la onda de sonido resultante de suma de todas las ondas generadas por las fuentes de sonido.

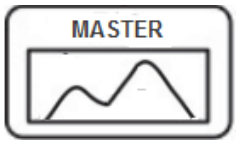

Figura 33. Modificador de amplitud - Master. Elaboración propia.

- LFO1 y LFO2 – Oscilador 1, Oscilador 2, Sub-oscilador, Cutoff y Master: Se podrá modificar la frecuencia o "rate" y la intensidad de cada LFO. Tanto el LFO1 como el LFO2 podrán modular la amplitud y frecuencia de los dos osciladores principales, así como la amplitud del sub-oscilador. Además, Tanto el LFO1 como el LFO2 podrán modular el "cutoff" del filtro y la intensidad del Master. Los LFOs se activarán cuando el usuario lo indique.

Figura 34. LFO1 Y LFO2. Pirkle (2015).

Controladores

- Controlador MIDI: Se usará un input MIDI, generado por un controlador MIDI, para obtener la nota presionada y la duración de esta. Estos datos serán usados por los osciladores principales y el sub-oscilador para generar las ondas iniciales y para empezar el proceso de síntesis.

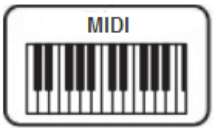

Figura 35. Controlador MIDI. Pirkle (2015).

- GUI: Por último, todos los demás parámetros detallados de cada componente se los controlara a través de la interfaz de usuario del sintetizador.

Figura 36. Símbolo GUI. Elaboración propia.

Interacción entre Componentes

A continuación, se muestra un diagrama donde se puede apreciar el diseño y funcionamiento del sintetizador de sonido. Se detalla la interacción entre los componentes y como se involucra cada uno en el proceso de síntesis, así como el flujo del sonido desde que se manda la señal MIDI hasta que se genera el output final.

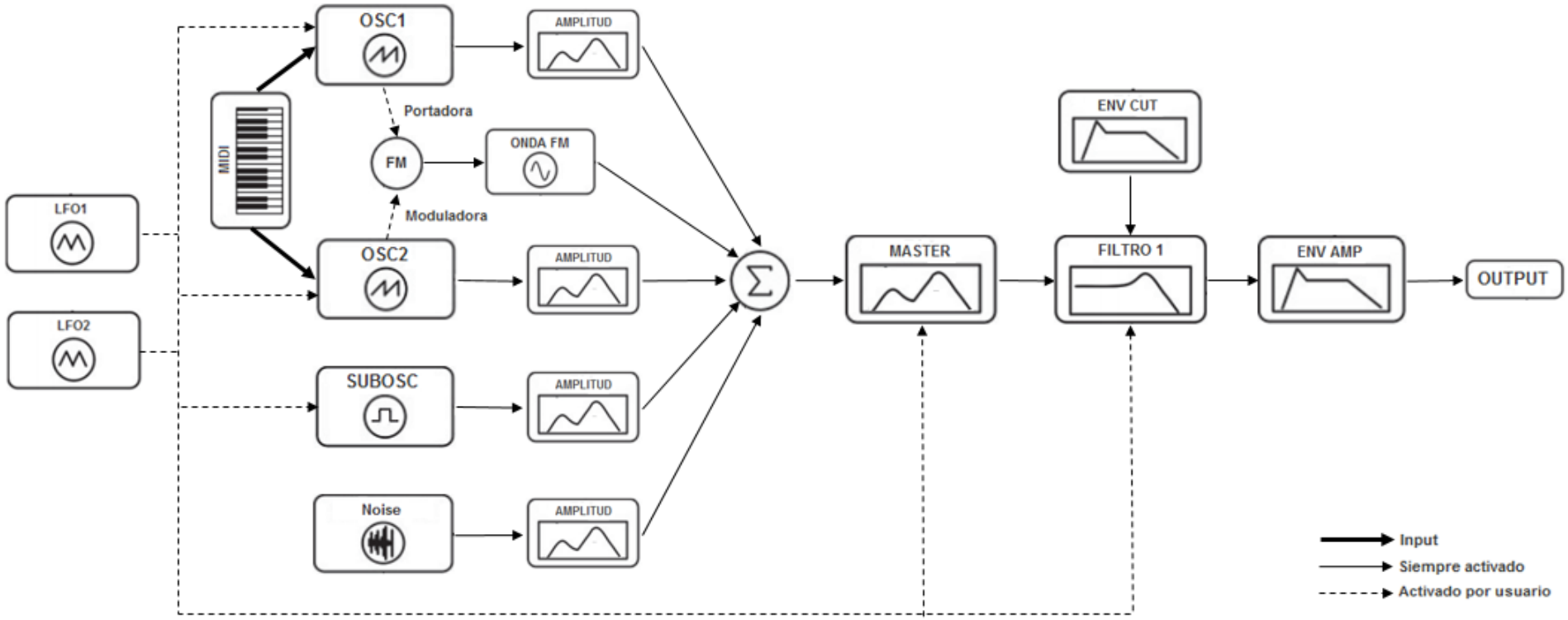

Figura 37. Diseño del Sintetizador de Sonido – Interacción entre componentes, proceso de síntesis y flujo de sonido. Elaboración propia.

Finalmente se muestra una ampliación del diagrama previo, donde se incluyen los parámetros de cada componente, y se especifica si se los controla usando el controlador MIDI o la GUI, así como los parámetros susceptibles a ser modulados por los LFOs.

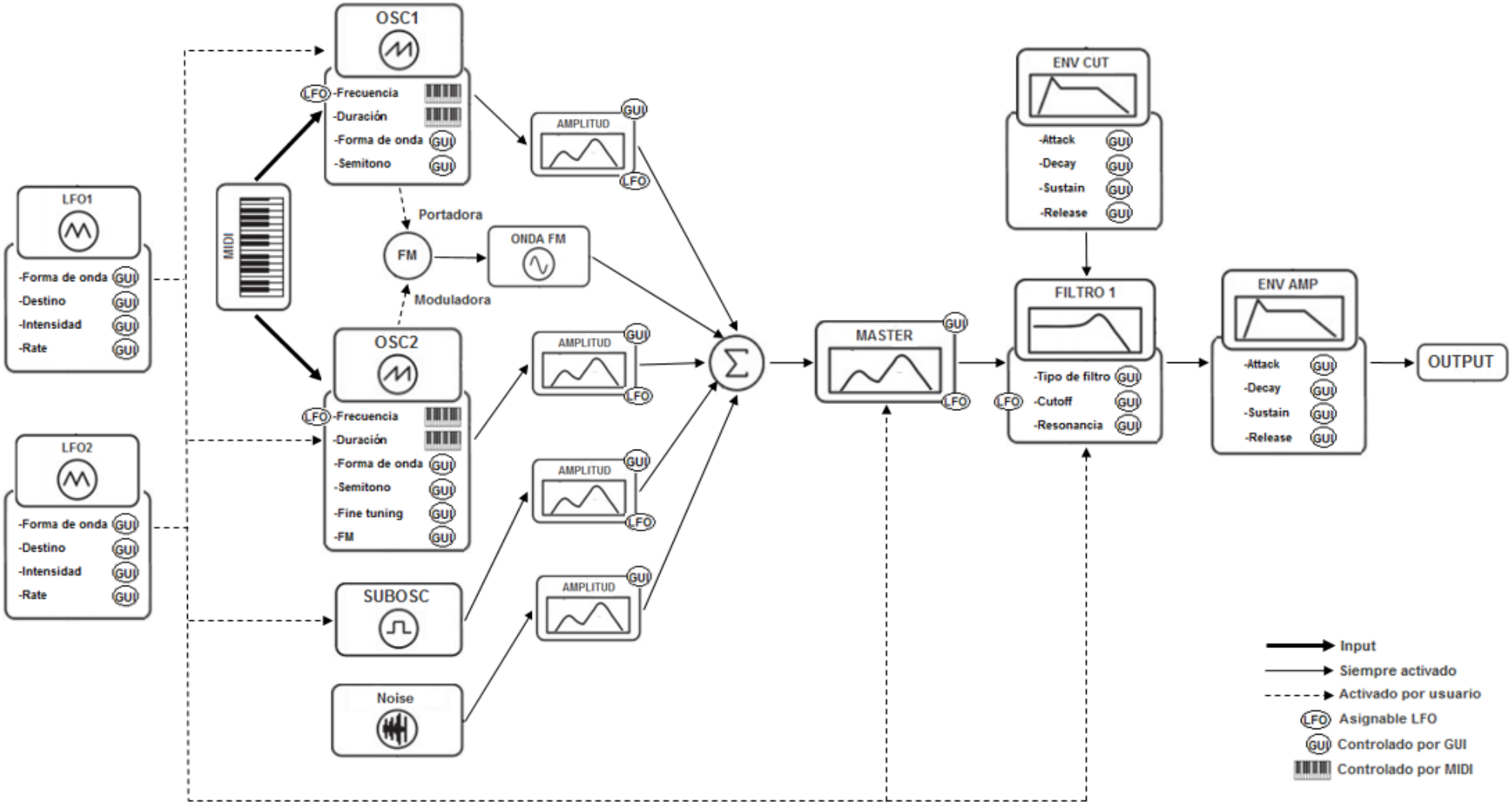

Figura 38. Diseño Detallado del Sintetizador de Sonido – Interacción entre componentes, proceso de síntesis y flujo de sonido. Elaboración propia.

ALGORITMOS DE PROCESAMIENTO DE AUDIO

Algoritmo para Síntesis y Modulación de Sonido

El siguiente algoritmo describe cómo funciona el proceso de síntesis y modulación de sonido en tiempo real, mostrado en los diagramas anteriores. Se usa pseudocódigo para mostrar cómo se implementará el algoritmo en software.

```
Mientras (Nota es presionada en controlador MIDI) {
    Se recibe input MIDI con la frecuencia de la nota presionada
    Se leen los datos de los parámetros de cada componente desde la GUI

    Si LFO1 y/o LFO2 son activados y se asigna(n) a la amplitud o frecuencia de OSC1
        LFO1 y/o LFO2 modula(n) amplitud o frecuencia de OSC1
    OSC1 genera la onda de sonido respectiva

    Si LFO1 y/o LFO2 son activados y se asigna(n) a la amplitud o frecuencia de OSC2
        LFO1 y/o LFO2 modula(n) amplitud o frecuencia de OSC2
    OSC2 genera la onda de sonido respectiva

    Si LFO1 y/o LFO2 son activados y se asigna(n) a la amplitud de SUBOSC
        LFO1 y/o LFO2 modula(n) amplitud de SUBOSC
    SUBOSC genera onda de sonido cuadrada

    NOISE genera ruido

    Si parámetro FM de OSC2 es activado
        OSC2 modula la frecuencia de OSC1
        Se genera onda FM                                   //Síntesis FM

    Se aplica modificación de amplitud a cada onda de sonido generada
    (excepto onda FM)
    Se suman las ondas generadas y se genera una sola onda de sonido    //Síntesis Aditiva
    Se aplica modificación de amplitud (MASTER) a onda de sonido

    Si LFO1 y/o LFO2 son activados y se asigna(n) a cutoff
        LFO1 y/o LFO2 modula(n) valor de cutoff
    Se aplica envolvente de corte (ENV CUT) a cutoff
    Se aplica el FILTRO respectivo a onda de sonido          //Síntesis Sustractiva

    Se aplica envolvente de amplitud (ENV AMP) a onda de sonido

    Se reproduce la onda de sonido final                }
```

CAPÍTULO 4: DESARROLLO E IMPLEMENTACIÓN

Para poder llevar toda la teoría descrita en los capítulos anteriores, y el diseño del sintetizador de sonido hacia un producto de software funcional, es necesario empezar aclarando algunos conceptos esenciales, así como las herramientas básicas necesarias, para entender de mejor manera como se manipula y procesa audio digitalmente a través de una computadora.

CONCEPTOS GENERALES

Conversión Analógico - Digital

Ya se mencionó como el sonido no es más que una onda que representa el cambio de presión en el aire. Dicha onda es continua en el tiempo, es decir que para cualquier punto en el tiempo que se la mida, se obtiene un valor correspondiente. Esto se conoce como analógico –o señal analógica, para ser más precisos-. Ahora bien, una computadora entiende solamente dos estados: paso de corriente o no paso de corriente, lo cual se lo representa como 1 o 0; esto implica que una computadora trabaja solo con datos discretos digitales -o señales digitales-. Por lo tanto, para que una computadora pueda "entender" una onda de sonido, es necesario convertir dicha onda de una señal analógica a una señal digital.

Para aquello, primero se transforma la energía acústica del sonido en energía eléctrica usando transductores. Posteriormente, la señal eléctrica atraviesa un proceso conocido como conversión analógica-digital, donde la amplitud de la señal es medida miles de veces por segundo, y cada medición se almacena en bytes. (Shepard, 2013). En la *Figura 39* se puede apreciar este concepto con más claridad:

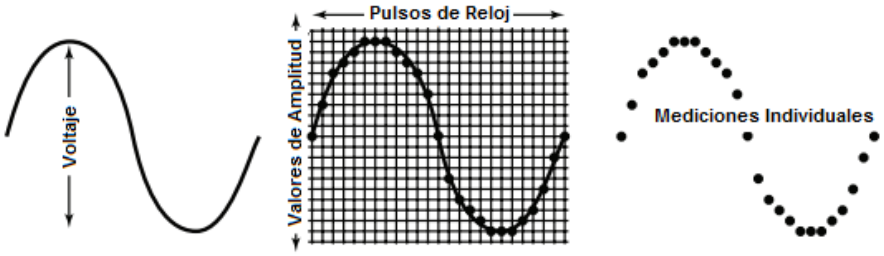

Figura 39. Digitalización de una señal analógica de sonido. Shepard (2013).

La imagen de la izquierda de la *Figura 39* representa la onda de sonido convertida en energía eléctrica. Cada línea vertical de la imagen del medio representa un momento especifico en el tiempo donde se mide el valor de amplitud de la onda. Una vez medida la onda se obtiene la imagen de la derecha, con todas las mediciones realizadas. Se puede apreciar como existen valores solamente para los momentos en los que se midió. La computadora no sabe que valores existen entre cada medición, y trata de reconstruir la onda solamente con los valores de los cuales dispone. Ahora bien, con este método surgen varios conceptos claves los cuales es necesario repasar: a cada medición individual se la conoce como muestra o "sample", la velocidad con la que se realizan las mediciones se conoce como velocidad de muestreo o "sample rate" y el número de bits usados para almacenar el valor de las mediciones se conoce como profundidad de bits o "bit depth". El sample rate es esencial a la hora de obtener una representación lo más apegada a la realidad posible de la onda, dado que mientras más rápido se realicen las mediciones, más mediciones habrá y, por lo tanto, la representación de la onda será más precisa. Además, el sample rate es más importante con frecuencias altas, dado que estas requieren más mediciones para que la onda resultante sea precisa, como se puede apreciar en la *Figura 40*, donde se usa el mismo sample rate para una frecuencia baja y para una frecuencia alta, pero se obtienen resultados distintos en cuanto a la representación final:

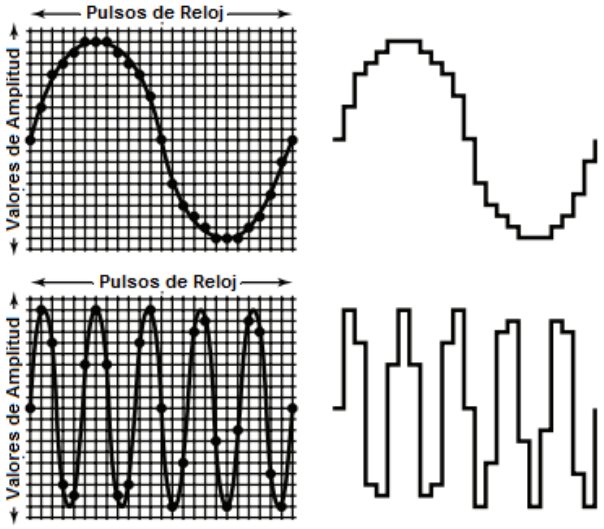

Figura 40. Digitalización de dos ondas con distintas frecuencias usando el mismo sample rate. Shepard (2013).

Para solucionar este problema, se usa uno de los conceptos fundamentales de la teoría de la información, el teorema de Nyquist. Este teorema plantea que el sample rate debe ser al menos el doble de la frecuencia más alta de la cual se va a realizar el muestreo. Dado que el ser humano puede escuchar frecuencias de hasta 20 kHz, según el teorema Nyquist, el sample rate debe ser de 40 kHz -40000 mediciones de la onda por segundo- para poder obtener una representación de onda que mantengan la información original. Ya en la práctica, se estandarizó un sample rate de 44.1 kHz para audio digital, a pesar de que también se usa 48 kHz, 88.2 kHz, 96kHz, 176 kHz o 192 kHz, con el objetivo de obtener un audio de mejor calidad (Shepard, 2013).

Por otro lado, en el mundo real, evidentemente no existe un límite para la amplitud de un sonido. Sin embargo, en el mundo digital, si existe un límite, el cual viene dado por el valor máximo de amplitud que se puede almacenar de acuerdo al bit depth que se usó. Por ejemplo, si el bit depth es de 16 bits, el valor máximo es de 65535; en cambio con un bit depth de 24 bits, el valor máximo es de 16777215. Debido a esta diferencia, el valor máximo, independientemente

del bit depth, se lo representa con 0 dBFS o “decibeles a escala completa”, por sus siglas en inglés (Pirkle, 2013). Esto significa que todos los valores por debajo del valor máximo serán menores a 0 dBFS. Es por esta razón que cuando se analiza una señal de audio digital, los valores de amplitud de la onda de sonido se representan con valores negativos, como ocurre, por ejemplo, en la *Figura 11* y *Figura 12* del Capítulo 2 del presente trabajo.

Si se aumenta el bit depth también se puede obtener una mejora de calidad, dado que se pueden almacenar mediciones más precisas. Sin embargo, a mayor bit depth y también a mayor sample rate, más memoria y más velocidad de procesamiento se necesita en la computadora. Por lo tanto, a la hora de desarrollar programas que manejen sonido, es necesario encontrar el balance adecuado a la hora de digitalizar y manipular audio, para lograr optimizar recursos.

A la final, una onda digital no es más que un arreglo de datos ordenado con sus valores de amplitud, y es sobre dichos valores que se aplican los algoritmos para manipular la onda. Una vez procesado el sonido, los valores del arreglo de datos se los pasa al buffer uno a uno, para posteriormente, a través de la tarjeta de sonido, realizar el proceso que se conoce como “renderización de audio”, que no es más que convertir nuevamente el audio digital a una señal analógica que pueda ser reproducida.

Síntesis con Tabla de Onda

En el caso del sintetizador de sonido del presente trabajo, no se necesita convertir una onda de sonido analógica a digital, dado que las ondas iniciales se las genera directamente en el programa. El concepto es el mismo, sin embargo, se deben generar los valores de amplitud que representen la forma de onda que se desee. Esto se lo puede abordar de varias maneras, no

obstante, la forma que resulta más eficiente, computacionalmente hablando, es la que se conoce como síntesis con tabla de onda. Esta técnica consiste en generar un arreglo de datos con los valores de amplitud de un ciclo de la forma de onda que se necesite -sinusoidal, sierra, cuadrada o triangular- y almacenarlo en memoria. Luego, cuando se desee generar la onda como tal, se procede a leer el arreglo de datos en un loop y, dependiendo de la velocidad de lectura, la frecuencia de la onda generada es distinta.

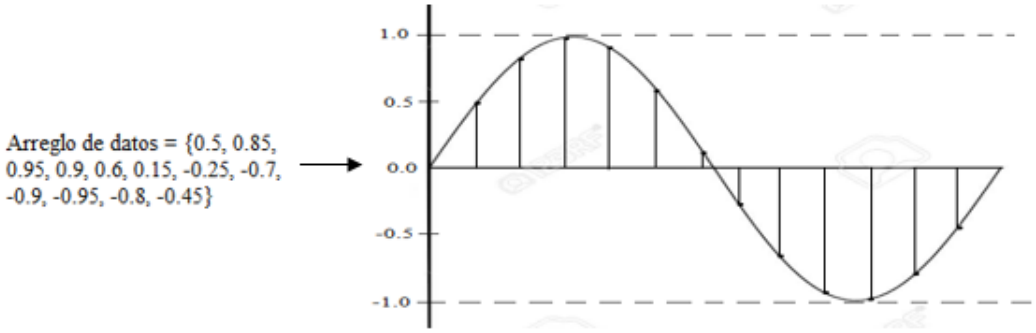

Figura 41. Ejemplo de onda sinusoidal generada a través de síntesis de tabla de onda. Elaboración propia.

El rango de valores usualmente es establecido entre 1 y -1. Posteriormente se los convierte a dBFS para poder manipularlos digitalmente de acuerdo al bit depth que se haya determinado.

La ventaja de este método es que se pueden generar cuantas ondas se deseen usando un solo arreglo de datos, optimizando así el uso de memoria y del procesador. La síntesis con tabla de onda se implementará, en el presente trabajo, a través de la librería Maximillian, que utiliza esta técnica en sus osciladores y de la cual se detallará más adelante.

DAW

Una Estación Digital de Trabajo o DAW, por sus siglas en inglés, es una combinación tanto de software como de hardware orientados a la grabación y edición de audio. La parte del hardware está compuesta por una interfaz de audio, la cual es la responsable de grabar la señal de audio en alta calidad y convertirla de analógica a digital y viceversa; y la parte de software, la cual comprende al o los programas donde se realiza la edición y procesamiento del audio. Popularmente se denomina DAW solamente a la parte de software.

Los DAWs son programas accesibles tanto para profesionales como para aficionados, son usados en las industrias de televisión, cine, videojuegos y, por supuesto, música. Algunos de los más populares son Logic, ProTools, GarageBand y Cubase. Es gracias a estos programas, que cualquier persona, en la actualidad, que disponga de una computadora y de una interfaz de audio -las cuales son muy accesibles- puede realizar proyectos musicales con un nivel de calidad alto.

Plug-In de Audio

Un plug-in de audio es un programa que agrega funcionalidades de procesamiento de audio a un DAW. Aunque también puede funcionar independientemente, lo más común es usarlo en conjunto, justamente, con un DAW, para poder usar el resultado del plug-in en un proyecto. Un plug-in cuenta, usualmente, con su propia interfaz. Esencialmente existen dos tipos, aquellos que reciben una señal de audio y la modifican de alguna manera, como pueden ser plug-ins de efectos como reverb, delay, compresores, equalizadores, etc; y aquellos que directamente generan y procesan la señal de audio. Justamente, el sintetizador de sonido desarrollado en el presente trabajo se encuentra dentro de la segunda categoría.

Cada DAW usa una arquitectura específica para comunicarse con un plug-in, lo que implica que si se quiere desarrollar un plug-in para un DAW en particular, se lo debe desarrollar en una

arquitectura compatible con dicho DAW. Por ejemplo, GarageBand trabaja con Audio Units o AU, Cubase con Virtual Studio Technology o VST y Pro-Tools con Avid Audio eXtension o AAX. Precisamente, AU, VST y AAX son las arquitecturas más usadas a la hora de hablar de aplicaciones de audio.

JUCE

JUCE es un framework orientado al desarrollo de aplicaciones de audio desarrollado en C++. Un framework es un programa que ofrece una base de trabajo estandarizada y módulos de software ya desarrollados sobre los cuales se puede desarrollar diferentes proyectos con mayor facilidad. Se escogió JUCE, en su versión 5.3.2 para el desarrollo del presente trabajo, debido a que es el framework usado por grandes compañías como Korg o Codex. Además, es multiplataforma, es decir, los plug-ins de audio que se desarrollan en JUCE se los puede exportar fácilmente a varias arquitecturas como VST, AU, o AAX; lo que permite que el plug-in sea compatible con prácticamente todos los DAWs del mercado. Por último, JUCE brinda varias facilidades para el desarrollo especifico de un sintetizador de sonido. Todo esto sumado a la gran comunidad online de desarrolladores que trabajan con JUCE, lo convirtieron en el framework ideal para los objetivos del presente trabajo.

Librería Maximillian

Maximillian es una librería de código abierto desarrollada en C++ la cual está enfocada en la síntesis de sonido. Maximillian facilita la implementación de componentes como osciladores o filtros, razón por la cual se decidió usarla en el presente trabajo. La librería está disponible en Github, el enlace se encuentra disponible en la sección de Anexos del presente trabajo.

IMPLEMENTACIÓN A DETALLE

Antes de empezar a codificar, e incluso antes de elegir la herramienta o framework a usar, se debe definir que metodología y/o enfoque de desarrollo se va a utilizar en el proyecto. Esta decisión depende del tipo de proyecto, requerimientos, cliente, recursos, etc. En el caso del presente trabajo, se optó por un desarrollo iterativo e incremental, donde se empieza desarrollando y probando una sección del sintetizador de sonido, obteniendo un producto funcional y, posteriormente, encima de lo ya desarrollado y probado, se aumenta una nueva sección, repitiendo el proceso hasta alcanzar el producto final. Las secciones en las cuales se dividió el desarrollo del sintetizador se detallan más adelante. Así mismo, en el capítulo de pruebas se muestran las pruebas finales realizadas una vez se completaron todas las secciones, dado que son las pruebas que representan el funcionamiento del producto final.

Para poder implementar una metodología de desarrollo y elaborar un proyecto de cualquier índole utilizando un framework, es necesario entender cómo trabaja dicho framework, identificando las particularidades que presenta. Es decir, los estándares que maneja, qué archivos genera, cuáles clases son las más importantes, qué archivos y propiedades se deben configurar, cómo se maneja la depuración y pruebas, etc. En otras palabras, como está estructurado el esqueleto del proyecto. En el caso específico del presente trabajo, es indispensable entender la estructura que maneja JUCE a la hora de desarrollar un proyecto de audio.

Para empezar, es importante mencionar al Projucer, el cual es la herramienta que usa JUCE para la gestión de sus proyectos. A través del Projucer se crea un nuevo proyecto, y todo archivo que se quiera agregar o eliminar del mismo se lo hace a través del Projucer. Básicamente, cualquier cambio a nivel de archivos de un proyecto se lo debe realizar en el Projucer y no en el IDE de desarrollo, para evitar errores al momento de la compilación. A través del Projucer

también se configura los parámetros del proyecto, como, por ejemplo, tipo de proyecto, arquitecturas a exportar, librerías externas, carpeta destino, etc.

Como ya se mencionó, JUCE es un framework enfocado en el desarrollo de aplicaciones de audio. Evidentemente, esto abarca un conjunto muy grande de posibilidades. No todo proyecto de audio es igual, depende del tipo específico de proyecto que se quiera realizar. En este caso, lo que se busca es desarrollar es un plug-in de audio, específicamente un sintetizador de sonido que funcione como plug-in. JUCE, para este tipo de proyecto en particular, presenta varias facilidades, dado que no solo permite especificar, usando el Projucer, que el proyecto es un plug-in; sino también, permite especificar que se trata de un sintetizador de sonido. Dicho de otro modo, JUCE tiene un esqueleto de proyecto solamente enfocado para el desarrollo de un sintetizador de sonido como plug-in, lo que evidentemente representa una gran ventaja para el desarrollo del presente trabajo.

A continuación, se detalla cómo se encuentra estructurado dicho esqueleto, detallando las clases y funciones más importantes, además de cómo se comunican estas entre sí, para lograr generar el procesamiento y renderización de audio en tiempo real, y permitir la implementación del diseño del sintetizador de sonido.

Estructura de JUCE

Para el desarrollo de un plug-in de audio, JUCE divide el proyecto en 2 clases principales: PluginProcessor y PluginEditor. La primera clase se encarga del “backend” del proyecto, es decir de los algoritmos de procesamiento y renderización de audio; y la segunda se encarga del “frontend”, es decir, de la interfaz gráfica que se le presenta al usuario.

Cuando el plug-in es específicamente un sintetizador de sonido, JUCE genera 2 clases adicionales: SynthSound y SynthVoice. SynthSound es la clase encargada de detectar un input MIDI y de pasar la información del sonido a SynthVoice, para que, esta última, use dicha información para generar, procesar y renderizar el sonido. Cada SynthVoice genera una sola nota, por lo tanto, un sintetizador debe tener un arreglo de SynthVoices para que sea polifónico. Así mismo, cada SynthVoice de dicho array puede usar el mismo SynthSound.

Es importante aclarar que, en el caso específico de un sintetizador de sonido como plug-in, realmente el "backend" del proyecto se encuentra en la clase SynthVoice y no en PluginProcessor, dado que es SynthVoice donde se desarrolla el algoritmo de procesamiento de sonido del sintetizador. Sin embargo, PluginProcessor resulta igual muy importante, dado que ahí se definen los parámetros de cada componente del sintetizador -entendiéndose por parámetros a los valores que se pueden seleccionar o modificar a través de la interfaz gráfica. Los valores de dichos parámetros se los pasa a SynthVoice, para realizar su procesamiento, a través de PluginProcessor. Además, la función de SynthVoice para renderizar el audio ya procesado se la llama, también, a través de PluginProcessor.

Por último, es importante mencionar a la clase AudioProcessorValueTreeState, la cual es parte del API de JUCE y se declara como objeto en PluginProcessor. Esta cumple una función esencial, dado que permite asociar los componentes de la GUI -manejados en PlugInEditor- con los parámetros de PlugInProcessor, permitiendo que los valores de dichos parámetros cambien según la interacción del usuario con la interfaz gráfica; para posteriormente pasarlos a SynthVoice y poder realizar el procesamiento y renderización en tiempo real.

A continuación, se detallan las clases previamente mencionadas, así como las funciones más importantes de cada una. No se mostrará el código fuente del programa como tal, sin embargo,

todo el proyecto está disponible en la plataforma de desarrollo colaborativo Github, el enlace se encuentra en la sección Anexos del presente trabajo. Se recomienda revisar el código fuente para entender de mejor manera la interacción entra las clases aquí descritas.

PluginProcessor

Esta clase hereda a la clase AudioProcessor de JUCE, la cual es la clase base para cualquier clase orientada a procesar audio. Como ya se indicó, usualmente el procesamiento de audio de un plug-in se realiza en esta clase, sin embargo, al ser el plug-in un sintetizador de sonido, dicho procesamiento se hace en SynthVoice. Sin embargo, PluginProcessor sigue siendo la clase principal, dado que aquí se inicializa al sintetizador como tal, se definen y pasan los valores de los parámetros de dicho sintetizador -asociados a componentes de la GUI- hacia SynthVoice, además de realizar el llamado a la función de renderización de audio SynthVoice. En otras palabras, PluginProcessor contiene a la clase SynthVoice, lo que implica que SynthVoice no puede funcionar sin PluginProcessor. Para lograr esto, previamente se debe declarar en esta clase 3 objetos esenciales: de tipo Synthesizer, SynthVoice y AudioProcessorValueTreeState, los cuales pertenecen a la API de JUCE. Es importante mencionar que, si bien el archivo generado por JUCE se llama PluginProcessor, el nombre de la clase será el nombre del proyecto acompañado de AudioProcessor. El nombre que se decidió para el proyecto -y para el sintetizador final- es RonSynth, por lo tanto, RonSynthAudioProcessor es el nombre de la clase. Las funciones principales de esta clase son:

- **Constructor:** Aquí se agrega al sintetizador el número de SynthVoices que se quiera que tenga -es decir, cuantas notas puede el sintetizador ejecutar al mismo tiempo-, así como el SynthSound que dichos SynthVoices usaran. Además, se agregan los parámetros del sintetizador al AudioProcessorValueTreeState, definiendo el rango de

valores que puede tener cada parámetro. Dichos valores, posteriormente, son los que se los pasa a la clase SynthVoice.

- **prepareToPlay():** Esta función se usa para limpiar los valores de la nota previa presionada del buffer, además de indicarle al sintetizador el sample rate que se va a usar.
- **processBlock():** Aquí se pasan los valores de los parámetros, modificados por el usuario a través de la interfaz gráfica y obtenidos del AudioProcessorValueTreeState, hacia cada SynthVoice del sintetizador. Además, se llama a la función de renderizado de audio de SynthVoice.
- **getStateInformation() – setStateInformation():** Estas funciones se usan para guardar el estado del programa usando XML, lo cual sirve para el desarrollo de presets. Esto se logra gracias al AudioValueTreeState, dado que contiene el estado del programa en un determinado momento y, dicho estado, se lo puede transformar fácilmente a XML.

AudioProcessorValueTreeState

Esta clase, que se declara como objeto dentro de PluginProcessor, contiene un objeto de tipo ValueTree, el cual es una estructura en forma de árbol que contiene todo el estado del AudioProcessor. Como ya se mencionó, PluginProcessor implementa la clase AudioProcessor y, además, es donde se definen los parámetros del sintetizador . El estado del AudioProcessor no es más que el valor de cada parámetro en un determinado momento. Cada parámetro tiene un identificador único y está asociado a un componente de la interfaz gráfica. Al existir dicha asociación, si hay un cambio en algún componente de la interfaz gráfica, cambia el valor del parámetro asociado a dicho componente y, por lo tanto, el estado del AudioProcessor es diferente. Justamente, en el ValueTree se almacena el estado del AudioProcessor y se actualiza

conforme cambian los parámetros. De esta forma se logra modificar, en tiempo real, los valores de los parámetros del AudioProcessor usando los componentes de la interfaz gráfica, y se puede acceder a dichos valores para realizar el procesamiento del audio.

SynthSound

Es la clase encargada de detectar un input MIDI que va a ser usado específicamente por el sintetizador, y enviar la información a SynthVoice. El uso de esta clase permite que varias instancias de SynthVoice usen el mismo SynthSound, logrando implementar un sintetizador polifónico fácilmente.

SynthVoice

En esta clase se reciben los valores de los parámetros del sintetizador, pasados desde PluginProcessor, y se realiza el procesamiento y renderización de audio. Por lo tanto, esta es la clase más importante del sintetizador de sonido, dado que aquí se implementan las funcionalidades de este. Aquí se genera las ondas iniciales, se las modifica de acuerdo con el diseño del sintetizador y, finalmente, se agrega el sonido final al buffer para ser renderizado de forma que pueda ser reproducido. Las funciones principales, aparte de las implementadas para el funcionamiento del sintetizador como tal, son:

- **canPlaySound():** Esta función se encarga de verificar que se reciba el SynthSound, de forma que pueda ser usado. Vale aclarar que no se instancia un objeto de tipo SynthSound en SynthVoice, pero si se importa la clase para poder verificar al SynthSound generado.
- **startNote() – stopNote():** En estas funciones se determina lo que se realiza una vez empieza la nota -cuando se presiona la tecla del controlador MIDI por parte del usuario- como puede ser, obtener la frecuencia y velocidad, o inicializar variables. Y

que se realiza una vez se termina la nota –cuando el usuario deja de presionar la tecla del controlador MIDI- como, por ejemplo, limpiar los valores que puedan quedar en el buffer.

- **renderNextBlock():** Esta es la función donde se renderiza el audio, es decir, se agrega el audio procesado al buffer para que la tarjeta de sonido o interfaz de audio lo conviertan en señal analógica y pueda ser reproducido. El audio procesado se agrega en samples a la velocidad del sample rate. Esta función es llamada en la función processBlock de PluginProcessor.

PluginEditor

En esta clase se maneja todo lo referente a la interfaz gráfica del sintetizador. Se declaran los componentes que se desean implementar y se los ubica dentro de la interfaz del programa. PluginEditor hereda de la clase AudioProcessEditor que, a su vez, hereda de la clase Component. PluginEditor es el componente gráfico principal y cualquier otro componente pasa a ser un componente hijo de PluginEditor. Es importante mencionar que cuando son varios componentes hijos, una buena práctica de programación es agruparlos en secciones, y crear clases para cada sección, de modo que en PluginEditor se declara un objeto correspondiente a cada sección. Esto permite una mejor organización de la interfaz y facilita el poder realizar cambios. Este concepto se explicará más a detalle en la sección sobre desarrollo de la interfaz gráfica del presente capítulo. De igual forma que con PluginProcessor, el archivo en JUCE se llama PluginEditor, pero el nombre de la clase será el nombre del proyecto acompañado de AudioProcessorEditor. En el caso del presente trabajo, la clase se denomina RonSynthAudioProcessorEditor.

Cada componente hijo debe ser asociado a un parámetro del AudioProcessorValueTreeState, si se quiere usar a dicho componente para manipular el audio en tiempo real. Las principales funciones de PluginEditor son las siguientes:

- **Constructor:** Aquí se define el tamaño de la ventana del programa y se agregan los componentes hijos a la interfaz.
- **paint():** En esta función se determinan parámetros visuales del componente padre -no de los componentes hijos-, como color de fondo, tipo de letra, titulo, etc.
- **resized():** Aquí se determina la ubicación de cada componente hijo dentro del tamaño de la ventana definido en el constructor.

Desarrollo Interfaz Gráfica

Distribución de componentes

En el Capítulo de Diseño del presente trabajo se detalló los 3 ejes sobre los cuales se realizó el diseño del sintetizador de sonido. Uno de esos ejes era el manejo intuitivo, en el cual se describía la importancia de definir secciones claramente identificables para facilitar y agilizar la familiarización del usuario con el sintetizador. Dichas secciones son implementadas en la interfaz gráfica, como se puede apreciar en la *Figura 42*, correspondiente, justamente, a la interfaz principal del sintetizador que se presenta al usuario:

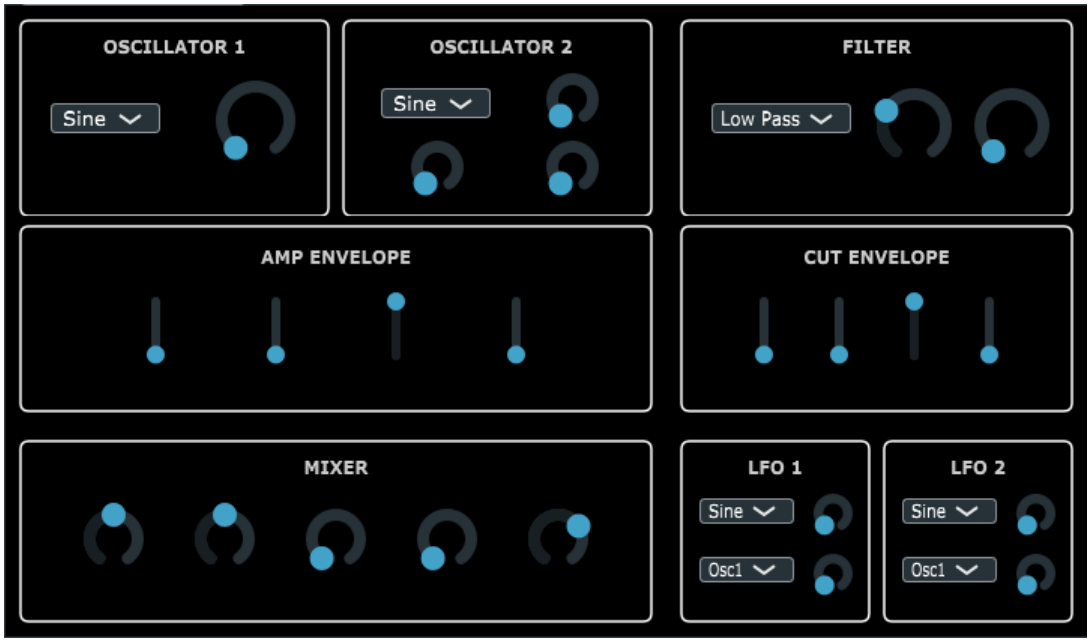

Figura 42. Interfaz gráfica del plug-in de sintetizador de sonido. Elaboración propia.

Los componentes principales: oscilador 1, oscilador 2, filtro, envolvente de amplitud, envolvente de corte, LFO 1 y LFO 2 tienen su propia sección, donde se agrupan los parámetros correspondientes a cada uno. Se definió, además, una sección llamada “Mixer” en la cual se

controla la amplitud de los osciladores, tanto los dos principales, como del sub-oscilador y del generador de ruido, así como de la suma de todas las ondas, denominado Master.

Por otro lado, como se puede apreciar, cada sección está definida por un rectángulo y su título correspondiente, sin embargo, no se muestra el nombre de los parámetros. Esto se debe a una decisión de diseño, dado que así se obtiene una interfaz más limpia y minimalista. Para que se muestre el nombre del parámetro, se debe colocar el mouse encima del mismo y aparecerá un recuadro donde se especifica que parámetro es. Así mismo, cuando se cambie el valor, el recuadro pasa a mostrar el valor correspondiente, como se muestra en la *Figura 43*:

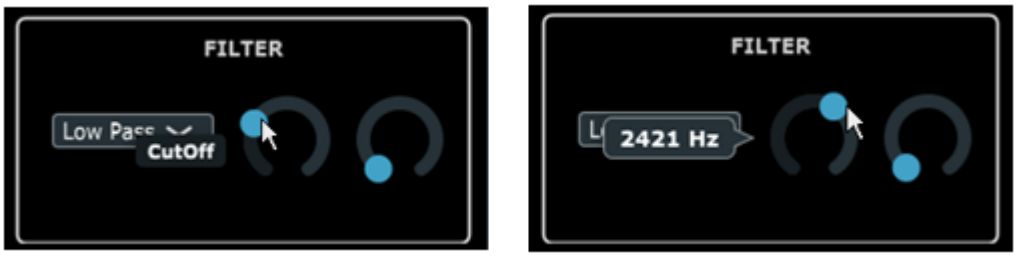

*Figura 43. El n*ombre y valor de parámetros en la interfaz gráfica del plug-in se muestran al posicionar el mouse encima y al hacer clic, respectivamente. Elaboración propia.

A continuación, se muestra en la *Figura 44* la interfaz gráfica indicando el nombre de los parámetros de cada sección, con el objetivo de mostrar la distribución de estos, aclarando nuevamente que así no se muestra la interfaz al usuario:

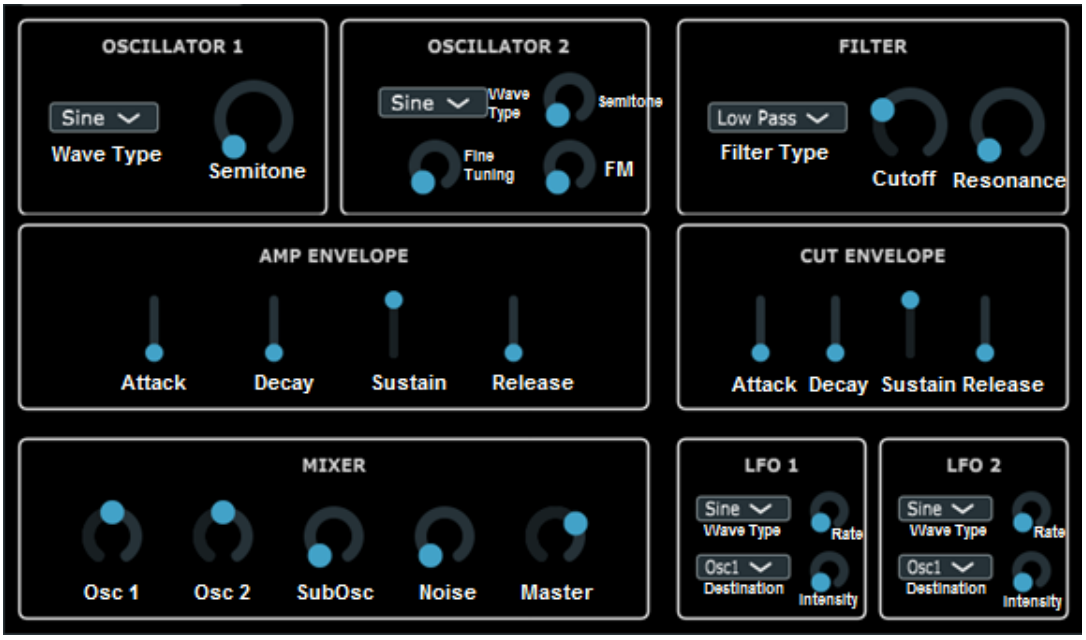

Figura 44. Interfaz gráfica del plug-in de sintetizador de sonido indicando el nombre de los parámetros de cada componente. Elaboración propia.

Se puede apreciar como están distribuidos los parametros de todos los componentes detallados en el capítulo de diseño. La interfaz no resulta sobrecargada con opciones, lo que facilita el uso al usuario. Cabe resaltar que se decidió nombrar en inglés a todos los parámetros y componentes, debido a que la gran mayoria de sintetizadores, ya sean fisicos o de software, se los encuentra en ese idioma. De aquí en adelante, en el presente trabajo, se utilizará los nombres en dicho idioma.

Código Fuente

Con respecto a la implementación de la interfaz gráfica en JUCE, como ya se mencionó, se lo realiza a través de la clase PluginEditor. Sin embargo, cada sección se la implementó en una clase, definida en un archivo independiente, donde se declaró los parámetros correspondientes. Esto con el objetivo de estructurar mejor el código y facilitar la edición de la interfaz. Cada clase, correspondiente a cada sección, tiene las mismas funciones que PluginEditor.

Dentro de cada clase se declaran los parámetros correspondientes. Por ejemplo, en Oscilador1 se declara un ComboBox, para escoger el wave type; y un Slider, para cambiar el semitone. Es

importante mencionar nuevamente que los parámetros de cada clase se deben asociar a los parámetros del AudioProcessValueTreeState. Esto se lo realiza declarando un objeto de tipo `ScopedPointer<AudioProcessValueTreeState::"Componente">`, donde en "Componente" se especifica el tipo de componente que aparece en la GUI, por ejemplo, ComboBox. Posteriormente, en el constructor de clase se inicializa esta variable pasando como argumento el componente respectivo. Así mismo, previamente en cada clase se debe declarar un objeto de tipo AudioProcessor, la cual hace referencia a PluginProcessor, para poder acceder al AudioProcessValueTreeState. De esta forma los elementos de la interfaz gráfica, como ComboBox o Sliders, se asocian y modifican los valores de los parámetros del AudioProcessValueTreeState de PluginProcessor.

Una vez creadas las clases para cada sección y asociados los parámetros al AudioProcessorValueTreeState, en PluginEditor se declaran objetos de cada clase, se los agrega a la interfaz en el constructor y se los ubica en la función resized(). Se puede entender a cada sección como interfaces independientes, donde se crean y se colocan los parámetros correspondientes, y a PluginEditor la interfaz principal, donde se crea y se coloca a cada sección. Nuevamente se recomienda revisar el código fuente del proyecto en Github para entender mejor esta explicación.

A continuación, se muestra un diagrama de clases donde se puede visualizar de manera clara como se conectan entre si todas las clases del proyecto. También se incluyen las clases correspondientes a la librería Maximillian, las cuales serán detalladas más adelante. El diagrama representa la estructura general del proyecto en JUCE y fue generado automáticamente a través de Visual Studio 2015, el cual fue el IDE usado para el desarrollo del sintetizador de sonido:

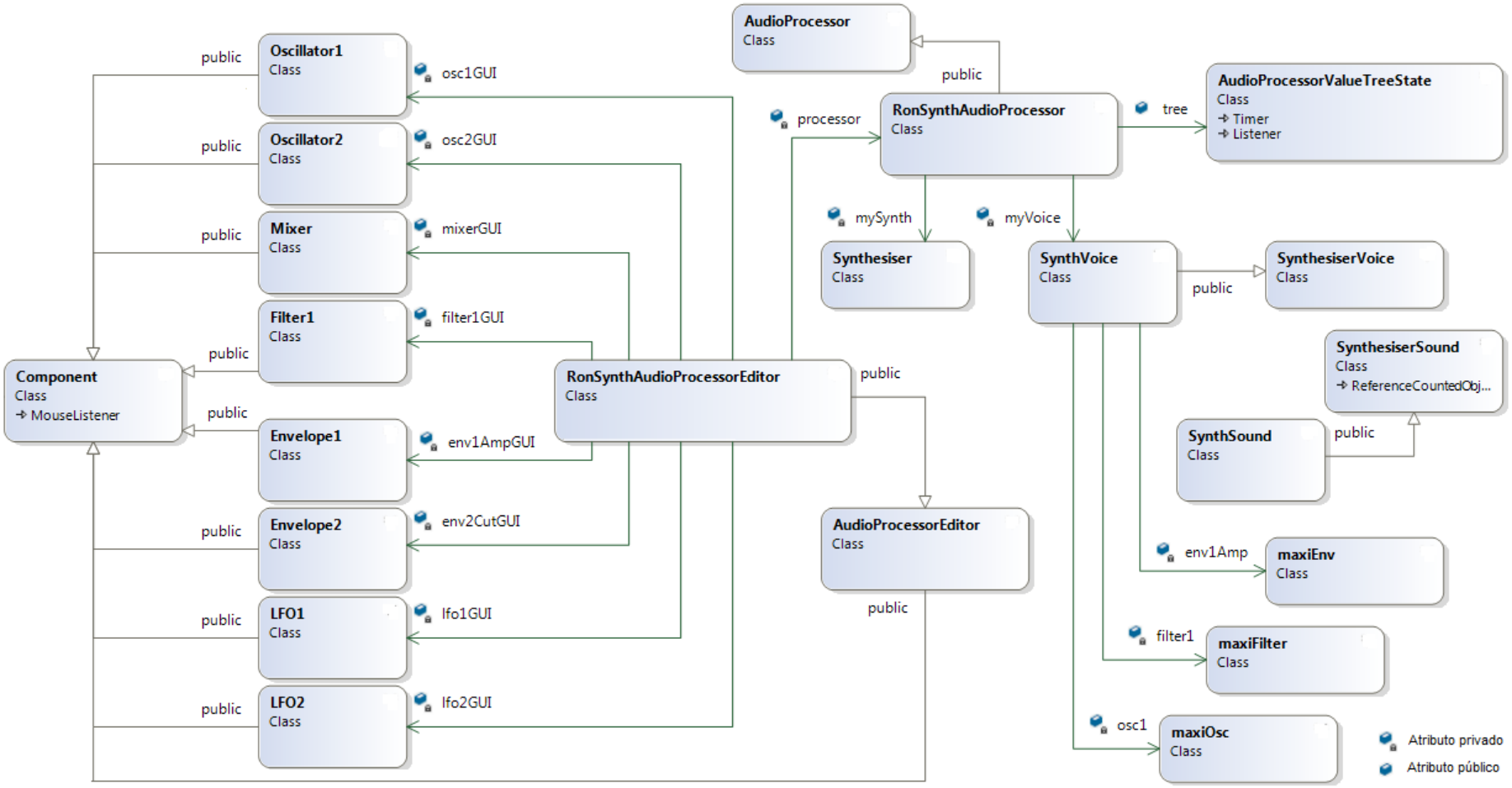

Figura 45. Diagrama de clases - Desarrollo del sintetizador de sonido en JUCE. Elaboración propia.

Implementación de Algoritmo para Síntesis y Modulación de Sonido

Consideraciones Previas

Un sintetizador de sonido como plug-in es un programa que procesa audio en tiempo real. Cualquier cambio en la interfaz de algún parámetro del sintetizador, debe escucharse inmediatamente en el audio que se renderiza. Esto implica que, a la hora de escribir el código fuente, la interfaz gráfica debe desarrollarse primero. Es esencial tener implementada una interfaz, no necesariamente vistosa, pero sí que tenga todos los componentes necesarios disponibles y asociados a los parámetros del AudioProcessValueTreeState. De esta forma se puede empezar a implementar el algoritmo de procesamiento de sonido, dado que se pueden probar los componentes del sintetizador en tiempo real y, por lo tanto, verificar que el algoritmo funcione correctamente.

En el presente trabajo se realizó la implementación justamente en ese orden. Primero se desarrolló la interfaz gráfica con todos los componentes mostrados en el apartado anterior, se asoció cada componente a un parámetro de AudioProcessValueTreeState y posteriormente se empezó a implementar el algoritmo. Como ya se indicó, la implementación del algoritmo para síntesis y modulación de sonido, detallado en el capítulo de diseño, se lo realiza en la clase SynthVoice. Aquí se reciben los datos de la interfaz gráfica, se utiliza el input MIDI, se generan los osciladores, se aplican las técnicas de síntesis, se modifica la onda de sonido y se renderiza el sonido final.

La implementación se la realizó siguiendo el pseudocódigo detallado en el capítulo de diseño. Una vez más, el código fuente se lo puede revisar en Github. A continuación, se explicará cómo se utiliza la librería Maximillian para el desarrollo de los osciladores, del filtro y de los envolventes; así como una breve descripción de las funciones desarrolladas más importantes.

Uso de Maximillian

Se revisó al principio de este capítulo como se representa a las ondas de sonido en el mundo digital, y cuál es la forma más eficiente, computacionalmente hablando, de generarlas. Con ese conocimiento se puede desarrollar los osciladores del sintetizador con relativa facilidad. Sin embargo, desarrollar un filtro y una envolvente resulta un poco más complejo. No es necesariamente un problema muy difícil de resolver, pero dado que no está dentro de los objetivos de este trabajo el necesariamente implementar una síntesis de tabla de onda, o un filtro y envolvente ADSR personalizados, se decidió utilizar la librería Maximillian para esa parte del desarrollo del sintetizador, facilitando así la implementación.

El uso de Maximillian es el siguiente: se declara los objetos `maxiOsc`, para un oscilador; `maxiFilter`, para un filtro; y `maxiEnv` para una envolvente; dentro de SynthVoice. Para los osciladores nada más se especifica la forma de onda que se requiere y se pasa la frecuencia como parámetro. Por ejemplo, si el objeto se llama osc1, se llama a la función `osc1.sinewave(frequency)` para generar una onda sinusoidal. Para el filtro es parecido, nada más se escoge el tipo de filtro y se pasa la onda de sonido a filtrar, el cutoff y la resonancia como parámetros; así, por ejemplo: `filter1.lores(osc1.sinewave(frequency), cutoff, resonance)`. De igual forma con la envolvente, se pasa el parámetro a ser afectado por el modelo ADSR y se especifica si la envolvente esta activada o no: `env1Amp.adsr(osc1.sinewave(frequency), env1Amp.trigger)`. env1Amp.trigger es un variable que puede ser 0 o 1 para indicar, justamente, si la envolvente esta activada o no, su valor se actualiza en las funciones `startNote()` y `stopNote()`.

Cada componente se lo insertó en una función específica, con el objetivo de organizar de mejor manera el código. Por ejemplo, el oscilador 1 usa una función llamado `setOsc1( )`,

donde se hace el llamado a la función de Maximillian para generar la onda, dependiendo de la forma de onda seleccionada por el usuario. Así mismo con el filtro y las envolventes de amplitud y de corte.

Funciones Implementadas

Con el uso de Maximillian se resolvió la implementación de los componentes individuales. El siguiente paso fue lograr que funcionen en conjunto. A continuación, se muestra un resumen de los pasos y funciones desarrolladas para implementar la interacción entre componentes, las técnicas de síntesis y el flujo de audio, según el algoritmo de síntesis y modulación de sonido:

- En primer lugar, todo parámetro obtenido de la interfaz gráfica se lo almacena en variables declaradas dentro de SynthVoice a través de funciones específicas, como, por ejemplo, `getOsc1()` para los parámetros del oscilador 1. Además, se inicializan los valores obtenidos del controlador MIDI en `startNote()`.
- Posteriormente se inicializa los osciladores de Maximillian con los parámetros escogidos por el usuario, como forma de onda o semitono, generando así las ondas iniciales del proceso de síntesis. Esto se realiza dentro de las funciones específicas para cada oscilador.
- Se implementa la función `setFMWave()`, en la cual se usa la onda generada por el oscilador 2 para modular la frecuencia de la onda generada por el oscilador 1 según la intensidad determinada por el usuario, implementando así la síntesis FM y generando una nueva onda denominada “onda FM”. Si el usuario no determina que el valor del parámetro FM sea mayor a 0, no se genera la onda FM.

- A continuación, se implementa la función `setOscillatorsMix()` en la cual se suman todas las ondas generadas por los osciladores, así como la onda FM, de acuerdo a la amplitud determinada por el usuario en la sección Mixer de la interfaz gráfica, implementado así la síntesis aditiva y generando una sola onda resultante.
- Luego, se pasa el parámetro de cutoff del filtro a través de la envolvente de corte en `setCutEnvelope()` y se usa la función `setFilter1()` para filtrar la onda resultante, según el tipo de filtro escogido por el usuario, implementado así la síntesis sustractiva.
- Seguidamente, se usa la función `setAmpEnvelope()` para pasar la onda filtrada por la envolvente de amplitud. Es dicha función la que se llama en la función `renderNextBlock()`, que es donde se renderiza el audio. Es importante mencionar que la renderización se hace agregando cada sample de la onda final al buffer, por lo tanto, la función `setAmpEnvelope()` -y por consiguiente todas las demás funciones-, es llamada por JUCE un número de veces igual al número de samples que posea la onda, a velocidad de sample rate.
- Por último, hay que mencionar que los LFOs se los llama dentro de las funciones correspondientes a los componentes que, según el diseño, tengan parámetros que pueden ser afectados por los LFOs. El LFO correspondiente se activa cuando su parámetro de intensidad de la interfaz gráfica es mayor a 0, modulando el parámetro que el usuario haya escogido.

En el siguiente diagrama de flujo se puede visualizar de mejor manera los pasos previamente detallados. El proceso vuelve a empezar cada que el usuario modifica un parámetro en la interfaz gráfica o presiona una tecla en el controlador MIDI:

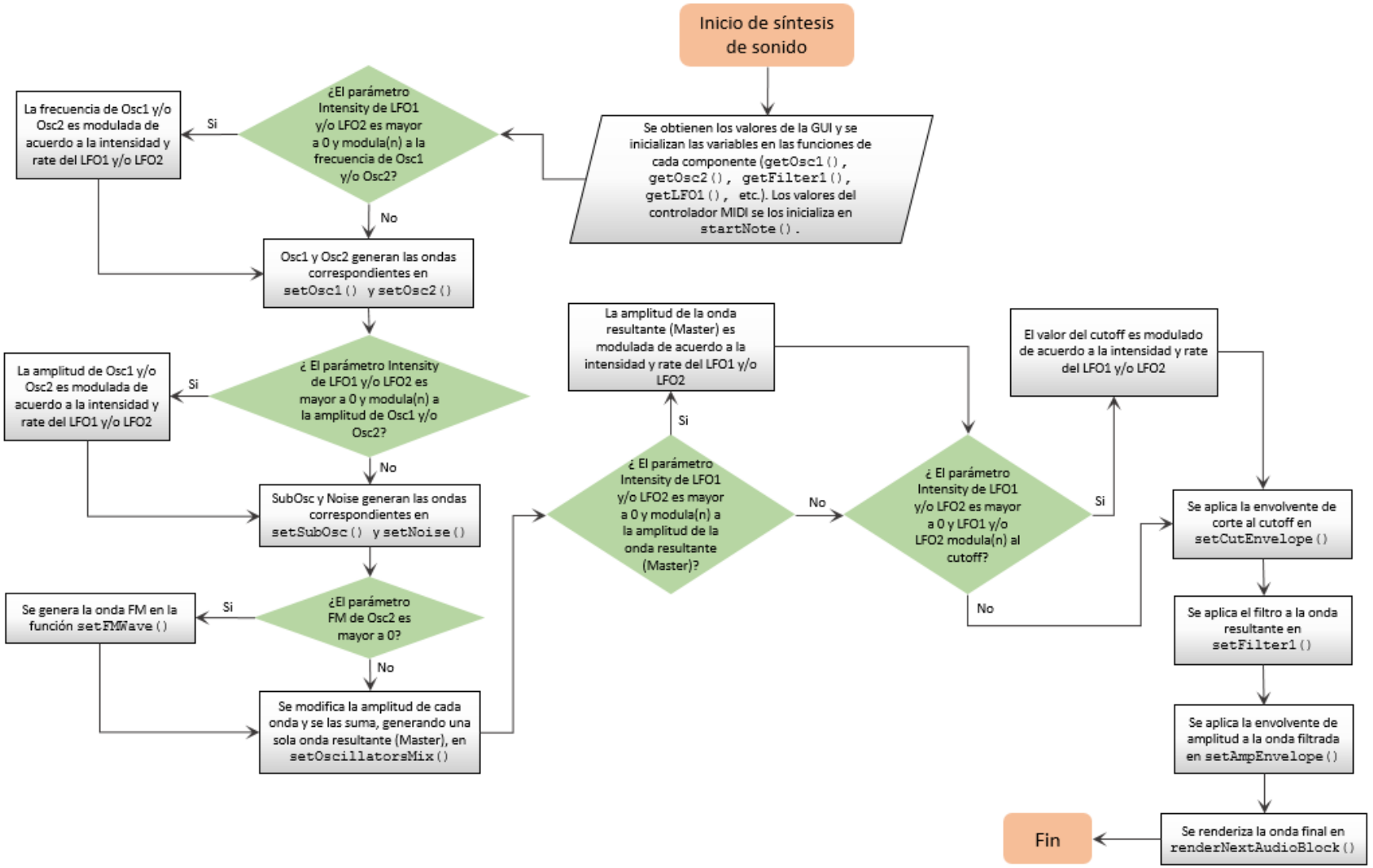

Figura 46. Diagrama de flujo - Proceso de síntesis de sonido. Elaboración propia.

De esta manera se consiguió implementar, utilizando JUCE, el diseño del sintetizador de sonido descrito en el Capítulo 3, obteniendo un sintetizador de sonido de software que funciona como plug-in, el cual puede ser usado en la mayoría de DAWs del mercado.

Se recomienda, por última vez, revisar el código fuente del proyecto en Github, para entender de mejor manera como está estructurado todo el proyecto y poder seguir, paso a paso, las explicaciones previamente detalladas a lo largo del presente capítulo.

CAPÍTULO 5: PRUEBAS

Una vez desarrollado el sintetizador de sonido de software en JUCE, es necesario comprobar el correcto funcionamiento de sus componentes, así como del algoritmo para síntesis y modulación de sonido. Para esto, se realizaron dos tipos de pruebas: pruebas unitarias, donde se comprobó que los 3 componentes principales: osciladores, filtro, envolvente de amplitud y envolvente de corte funcionen, individualmente, de la manera en que se espera; y pruebas de integración, donde se verificó que la interacción entre los componentes sea la correcta, revisando que tanto las técnicas de síntesis, como los LFOs cumplan su función correctamente. Por último, se estableció un rango de valores para cada parámetro, de forma que el usuario pueda aumentar o disminuir el valor de cualquier parámetro sin que existan fallos en el procesamiento o renderización del audio.

Las pruebas se las realizó en un ambiente con las siguientes características y configuraciones:

- Sistema operativo: Windows 7 Professional 64-bits
- Procesador: Intel Core i5-4430 3.00Ghz
- Memoria RAM: 8 Gb
- Disco duro: 1 Terabyte
- Tarjeta de sonido: High Definition Audio Device (Integrada)
- Controlador MIDI: Nektar LX25+

Para los dos tipos de pruebas se usó el plug-in de sintetizador de sonido en conjunto con el DAW Bitwig Studio, en su versión 2.4, el cual dispone de dos plug-ins que fueron usados para el análisis de la onda de sonido: osciloscopio, para observar la forma de onda; y analizador de frecuencias, para observar el espectro de frecuencias de la onda.

PRUEBAS UNITARIAS

Oscilador

El proceso de síntesis de sonido empieza con los osciladores, los cuales son responsables de generar las siguientes formas de onda: sinusoidal, sierra, triangular, cuadrada. Las pruebas están enfocadas a verificar que, efectivamente, los osciladores de la librearía Maximillian generen correctamente dichas formas de onda. Las pruebas se las realizó a través del osciloscopio de Bitwig Studio, utilizando un solo oscilador, renderizando directamente la función donde se genera, sin pasar por el filtro ni la envolvente.

Resultados

Figura 47. Prueba oscilador Maximillian generando onda sinusoidal. Elaboración propia.

Figura 48. Prueba oscilador Maximillian generando onda sierra. Elaboración propia.

Figura 49. Prueba oscilador Maximillian generando onda triangular. Elaboración propia.

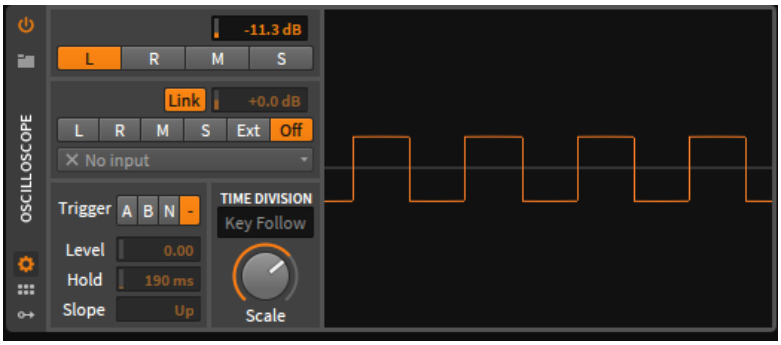

Figura 50. Prueba oscilador Maximillian generando onda cuadrada. Elaboración propia.

Como se puede apreciar en la *Figura 47*, *Figura 48*, *Figura 49* y *Figura 50*; todas las formas de onda son generadas tal cual lo esperado. Cabe recordar que el sub-oscilador es una onda cuadrada, y los LFOs tienen las mismas formas de onda, por lo tanto, con estas pruebas también se verifica su funcionamiento. El generador de ruido, así como los parámetros de semitone y fine tuning de los osciladores, se los probó escuchando su resultado, dado que no se tiene una referencia de cómo debe verse su forma de onda como para poder comprobar su funcionamiento de esa forma.

Filtro

Para comprobar el correcto funcionamiento del filtro se usó el analizador de frecuencias de Bitwig Studio para poder observar cómo cambiaban las frecuencias de la onda. Se uso un solo oscilador que generaba una onda de sierra con una frecuencia fundamental de 246 Hz. Posteriormente dicha onda pasaba por el filtro, sin pasar por la envolvente de amplitud. La prueba se la realizó comprobando los tres tipos filtro: low pass, high pass y band pass, así como la resonancia.

La forma que se espera tenga el espectro de frecuencias después de aplicar cada tipo filtro está determinada de acuerdo a la teoría descrita en el Capítulo 2 sobre el funcionamiento del filtro. Para los filtros low pass y high pass se usaron 3 pruebas, una donde el efecto del filtro sea mínimo, otra donde se aprecie de manera evidente, y una última donde se use la resonancia. Para el caso del filtro band pass solo se realizó una prueba donde se pueda apreciar de forma clara el efecto del filtro.

Resultados Low Pass

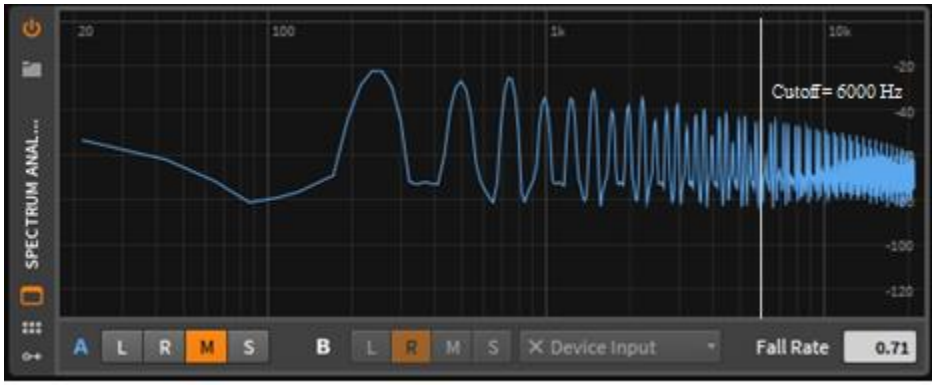

Figura 51. Prueba filtro low pass con cutoff de 6000 Hz, sin resonancia. Elaboración propia.

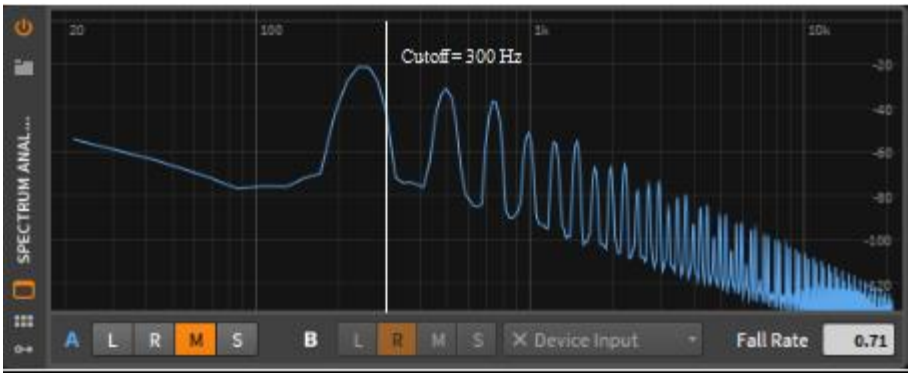

Figura 52. Prueba filtro low pass con cutoff de 300 Hz, sin resonancia. Elaboración propia.

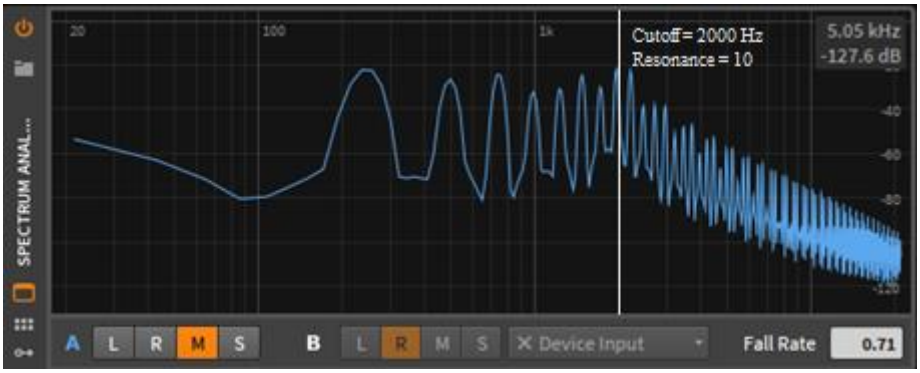

Figura 53. Prueba filtro low pass con cutoff de 2000 Hz, resonancia máxima. Elaboración propia.

Se puede observar cómo, en todos los casos mostrados en la *Figura 51*, *Figura 52* y *Figura 53*; las frecuencias que están por debajo del cutoff pasan inalteradas, en cambio las frecuencias por encima se atenúan gradualmente. Es importante aclarar que los valores de amplitud mostrados en las figuras -así como en el resto de las figuras del presente capitulo- son negativos debido al uso de dBFS, como se explicó en la sección de Conversión Analógico - Digital del Capítulo 4. Se puede determinar que el filtro low pass y la resonancia, como se aprecia, funcionan de acuerdo a lo esperado.

Resultados High Pass

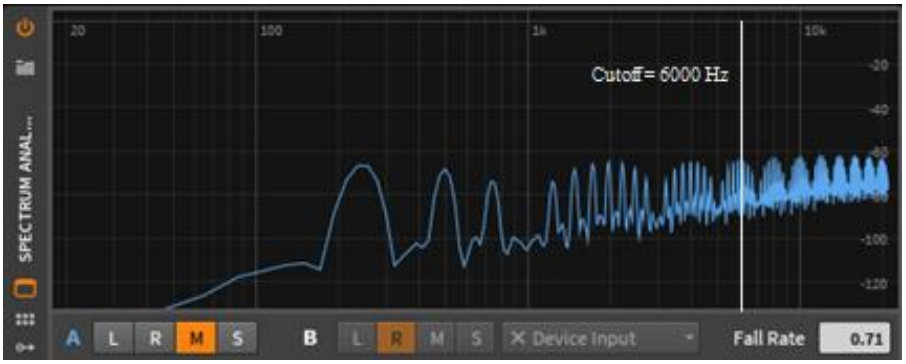

Figura 54. Prueba filtro high pass con cutoff de 6000 Hz, sin resonancia. Elaboración propia.

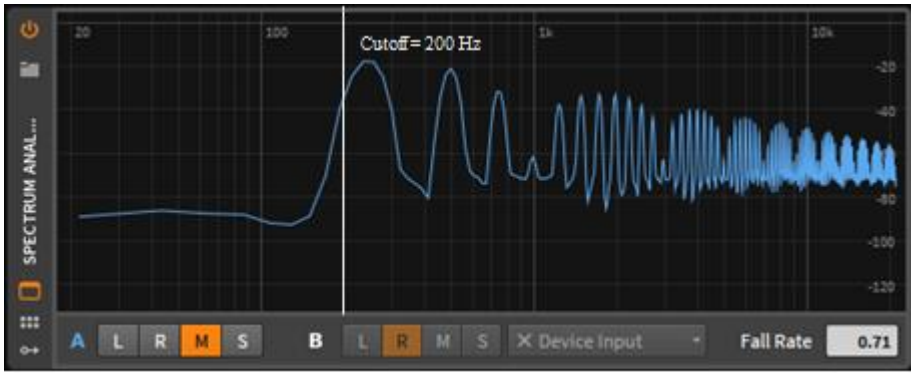

Figura 55. Prueba filtro high pass con cutoff de 200 Hz, sin resonancia. Elaboración propia.

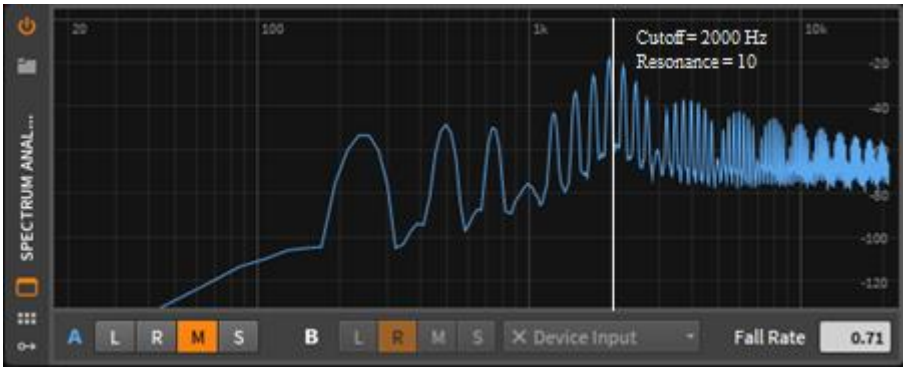

Figura 56. Prueba filtro high pass con cutoff de 2000 Hz, resonancia máxima. Elaboración propia.

Se puede observar en la *Figura 54*, *Figura 55* y *Figura 56* como las frecuencias que están por encima del cutoff pasan inalteradas, en cambio las frecuencias por debajo se atenúan gradualmente. Por lo tanto, se puede determinar que el filtro high pass y la resonancia, como se aprecia, funcionan de acuerdo a lo esperado.

Resultados Band Pass

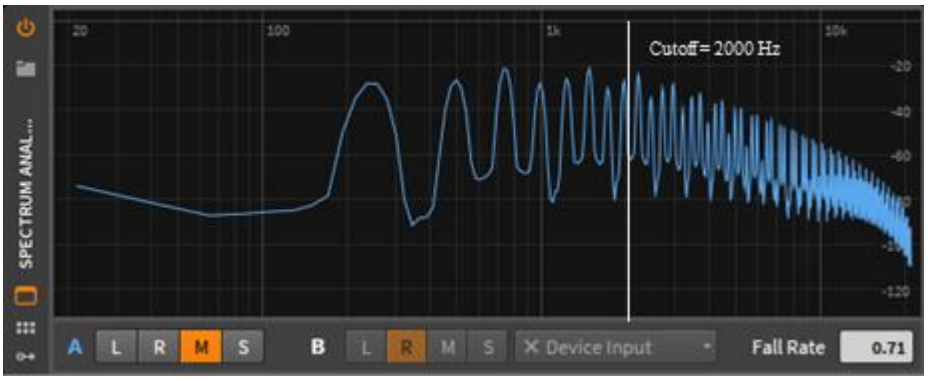

Figura 57. Prueba filtro band pass con frecuencia central de 2000 Hz, sin resonancia. Elaboración propia.

Se puede observar en la *Figura 57* como las frecuencias que están por encima y por debajo del cutoff se atenúan gradualmente, formando una especie de arco, que es lo que espera del filtro band pass. Aunque, si bien es cierto, dicha atenuación no es tan evidente como en los otros filtros. Aun así, se puede determinar que el filtro band pass funciona de acuerdo a lo esperado.

Para terminar, cabe resaltar que el filtro de Maximillian, si bien funciona correctamente como ya se mostró, no permite mucha configuración de sus parámetros, como por ejemplo la pendiente, en el caso de los filtros low pass y high pass; o el ancho de banda en el caso del band pass. Tampoco dispone un filtro de tipo banda eliminada, lo cual limita en parte la funcionalidad final del sintetizador de sonido.

Envolventes ADSR

En el caso de las envolventes ADSR, se probó tanto la envolvente de amplitud como la de corte usando un solo oscilador. Para la envolvente de amplitud se usó varias configuraciones del modelo ADSR y, a través del osciloscopio, se observó la evolución de la amplitud de la onda en el tiempo. En el caso de la envolvente de corte se utilizó al analizador de frecuencias, sin embargo, dado que este no muestra la evolución en el tiempo del espectro de frecuencias, se capturaron imágenes correspondientes a cada fase del modelo ADSR.

Resultados Envolvente de Amplitud

Figura 58. Prueba envolvente de amplitud - Attack: 1000ms; Decay: 50ms, Sustain: 0.8; Release: 50ms. Elaboración propia.

Figura 59. Prueba envolvente de amplitud - Attack: 50ms; Decay: 50ms, Sustain: 0.3; Release: 1000ms.

Elaboración propia.

Figura 60. Prueba envolvente de amplitud - Attack: 400ms; Decay: 1000ms, Sustain: 0.4; Release: 900ms.

Elaboración propia.

Los resultados de la *Figura 58*, *Figura 59* y *Figura 60* son consistentes que el funcionamiento que espera de la envolvente de corte, de acuerdo a la teoría descrita en el Capítulo 2, dado que se aprecia claramente como la amplitud de la onda varia en el tiempo, logrando identificar claramente cada fase del modelo ADSR. Por lo tanto, se puede determinar que la envolvente de amplitud funciona correctamente.

Resultados Envolvente de Corte

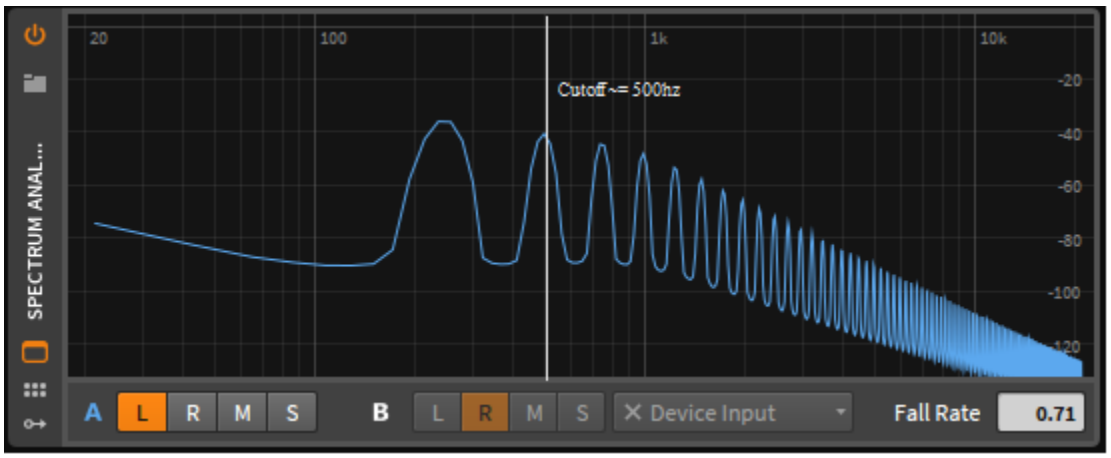

Figura 61. Prueba envolvente de corte - Espectro de frecuencias al inicio de la fase de attack. Filtro low pass con Cutoff ~= 500hz. Elaboración propia.

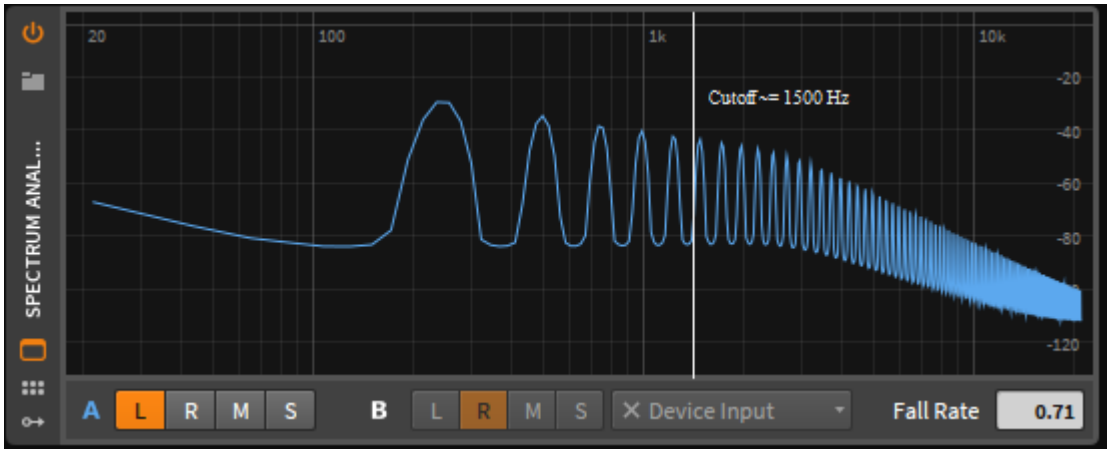

Figura 62. Prueba envolvente de corte - Espectro de frecuencias al final de la fase de attack e inicio de la fase de decay. Filtro low pass - Cutoff ~= 1500hz. Elaboración propia.

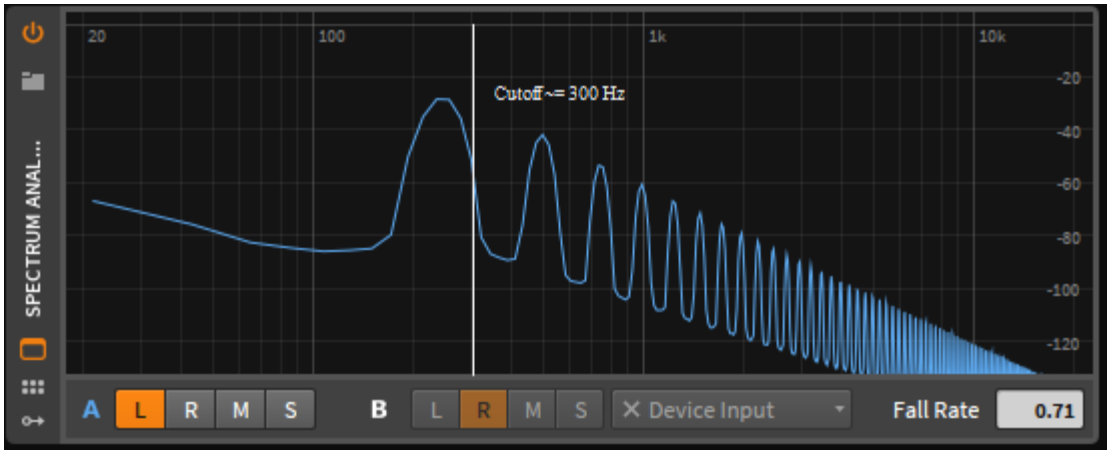

Figura 63. Prueba envolvente de corte - Espectro de frecuencias al final de la fase de decay, donde el cutoff se estabiliza en el valor del sustain. Filtro low pass con Cutoff ~= 300hz. Elaboración propia.

El funcionamiento de la envolvente de corte es más difícil de apreciar a través de las figuras mostradas. Hay que recordar que lo que se espera es que la envolvente cambie el valor del cutoff en el tiempo. La *Figura 61*, *Figura 62* y *Figura 63* corresponden a 3 momentos distintos en el tiempo de una misma onda de sonido, correspondientes a las distintas fases del modelo ADSR, donde se puede verificar como el valor del cutoff, efectivamente, es distinto en cada momento. El valor del cutoff mostrado no es exacto dado que, justamente, varia en el tiempo. Evidentemente, en las figuras no se puede observar el tiempo que toma en cambiar entre cada fase, esto se lo comprobó escuchando el resultado final. De esta forma se puede determinar que la envolvente de corte funciona correctamente.

PRUEBAS DE INTEGRACIÓN

Técnicas de Síntesis

Una vez verificado el correcto funcionamiento de los componentes principales, se procedió a comprobar que las 3 técnicas de síntesis implementadas: síntesis FM, síntesis aditiva y síntesis sustractiva; funcionen correctamente.

Resultado Síntesis FM

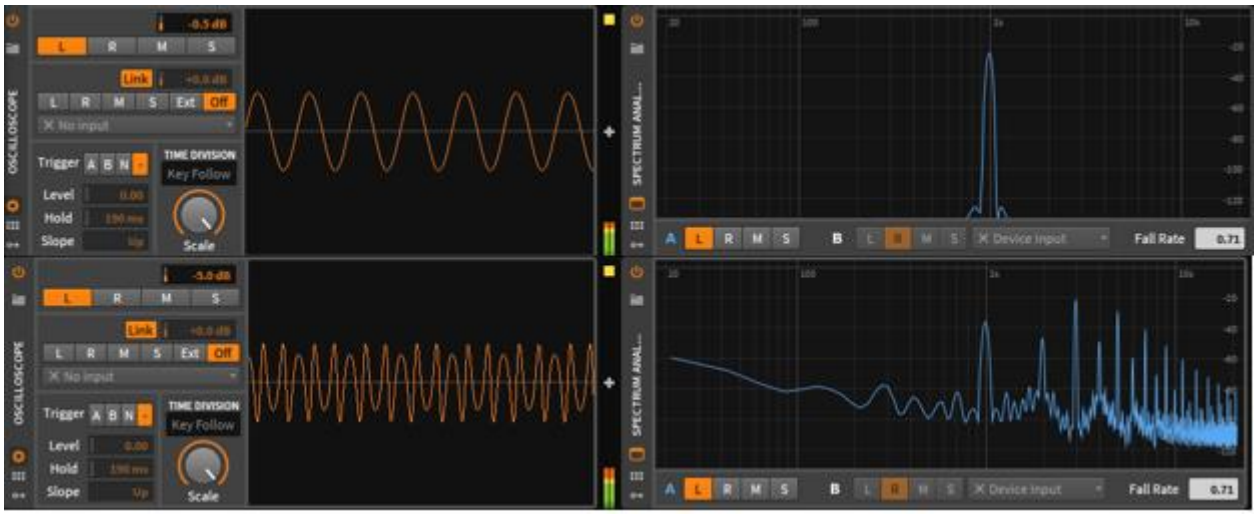

Figura 64. Prueba síntesis FM – Parte superior: onda sinusoidal y su frecuencia fundamental antes de ser modulada. Parte inferior: forma de la onda FM y espectro de frecuencias resultante. Elaboración propia.

Si bien es cierto no se puede saber con certeza la forma de onda resultante de la síntesis FM, si se sabe, gracias a la teoría, que se deben generar frecuencias, tanto armónicos como inarmónicos, a los dos lados de la frecuencia fundamental. Justamente ese es el parámetro escogido para verificar que la síntesis FM esté funcionando correctamente. Se puede apreciar en la *Figura 64* que es precisamente lo que ocurre.

Resultado Síntesis Aditiva

Se partió de la onda FM generada en la prueba anterior, y se sumaron el resto de las ondas generadas por los osciladores. De igual forma, no se tiene una forma de onda esperada, así que solo se comprobó que cambie la forma de onda anterior y aumenten las frecuencias, ya que esto es un indicador claro de que si se están sumando las ondas generadas.

Figura 65. Prueba síntesis aditiva. Configuración de los osciladores, forma de onda y espectro de frecuencias resultante. Elaboración propia.

Se puede observar en la *Figura 65* la amplitud de cada oscilador sumado, así como la forma de onda resultante, la cual se vuelve un poco caótica debido al ruido, y aparecen frecuencias más bajas debido al sub-oscilador, verificando así el correcto funcionamiento de la síntesis aditiva.

Resultado Síntesis Sustractiva

Finalmente, a la onda generada en la prueba anterior se le aplicó un filtro high pass con un cutoff de 1300 Hz y la resonancia máxima, para verificar que la forma de onda cambia y el espectro de frecuencias es modificado a través de la síntesis sustractiva.

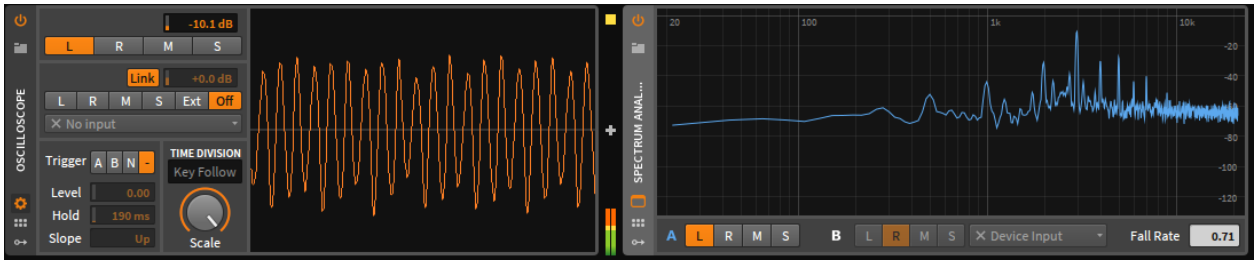

Figura 66. Prueba síntesis sustractiva. Aplicación filtro high pass con cutoff: 1400hz y resonancia máxima. Elaboración propia.

El espectro de frecuencias y la forma de onda final que se puede apreciar en la *Figura 66* cambia debido a la síntesis sustractiva, comprobando así su correcto funcionamiento. La onda mostrada es el resultado de aplicar las 3 técnicas de síntesis en conjunto.

LFOs

Una vez comprobado el correcto funcionamiento de los componentes y de las técnicas de síntesis, se procedió a verificar los LFOs. Para esto, se usó el osciloscopio, de forma que se pueda observar el resultado de los LFOs sobre la onda. Se aplicó un LFO a un solo oscilador. Como recordatorio, hay que mencionar que los LFOs pueden modular a la amplitud y frecuencias de tanto el oscilador 1 como el oscilador 2. Esto resulta en un efecto de tremolo y vibrato respectivamente, los cuales tiene una forma de onda resultante esperada, la cual es justamente la que se aparece en la *Figura 67* y *Figura 68*, verificando así el correcto funcionamiento de los LFOs.

Por último, cabe mencionar que los LFOs también pueden modular al cutoff; el correcto funcionamiento de esto se verificó escuchando y observando el espectro de frecuencias en tiempo real, por lo que no se adjuntan resultados.

Resultado Tremolo

Figura 67. Prueba LFO – Amplitud de onda modulada, generando el efecto de tremolo. Elaboración propia.

Resultado Vibrato

Figura 68. Prueba LFO - Frecuencia de la onda modulada, generando el efecto de vibrato. Elaboración propia.

Rangos de Valores

Comprobado que todos los componentes funcionan, tanto individualmente como en conjunto, de la forma esperada, el último paso consistió en determinar los valores mínimo y máximo que cada parámetro puede tomar, de forma que el usuario pueda modificar los valores en la interfaz gráfica a su gusto, sin que se presenten errores al momento de procesar o renderizar el audio. De esta forma se busca garantizar el correcto funcionamiento de todos los componentes del sintetizador de sonido bajo cualquier configuración de sus parámetros. A continuación, se muestra una tabla con el rango de valores determinado para cada parámetro:

Componente	Valor mínimo GUI	Valor máximo GUI	Unidad de medida	Fórmula aplicada en Backend
Oscillators				
Semitone	0	12	Unidades	Ninguna - Valor directo
Fine tuning	0.00	10.00	-	Frecuencia Osc 2 + FineTuning/10 * [(Frecuencia nota siguiente – Frecuencia nota ejecutada)/2]
FM	0.00	10.00	-	Ninguna - Valor directo
Filter				
Cutoff	20	8000	Hz	Valor directo o Valor modificado por LFO
Resonance	0.0	10.0	-	Ninguna - Valor directo
Envelopes				
Attack	50	3000	ms	Attack * 10
Decay	50	3000	ms	Decay * 10
Sustain	0.00	1.00	-	Ninguna - Valor directo
Release	50	3000	ms	Release * 5
Mixer				
Osc1	0.00	10.00	-	decibelsToGain(-48.0 + (Osc1 * 4.80))
Osc2	0.00	10.00	-	decibelsToGain(-48.0 + (Osc2 * 4.80))
SubOsc	0.00	10.00	-	decibelsToGain(-48.0 + (SubOsc * 4.80))
Noise	0.00	10.00	-	decibelsToGain(-48.0 + (Noise * 4.80))
Master	0.00	10.00	-	decibelsToGain(-48.0 + (Master * 4.80))
LFOs				
Rate	0.0	10.0	Hz	Ninguna - Valor directo
Intensity	0.0	10.0	-	Depende del parámetro a modular

Tabla 1. Rango de valores y fórmula en backend de cada parámetro del sintetizador. Elaboración propia.

Se puede apreciar en la *Tabla 1* que el valor que se presenta al usuario en la interfaz gráfica es posteriormente usado en una fórmula en el backend para cambiar su valor de acuerdo a las necesidades del componente. En el caso de los parámetros semitone de los osciladores, FM del oscilador 2, rate de los LFOs, resonance del filtro y sustain de las envolventes, el valor se usa tal cual se recibe de la interfaz al momento de implementarlo. Para el resto de los parámetros, el valor es cambiado, según la fórmula que se indica, antes de ser usado en la función correspondiente.

Para el fine tuning, se usó una fórmula de modo que cuando el valor de la interfaz sea el máximo, se sume, a la frecuencia original, el valor medio de la diferencia entre la frecuencia original de la nota ejecutada y la frecuencia de la nota siguiente. Por ejemplo, si se ejecuta la nota A4, cuya frecuencia es 440.0 Hz, se identifica la nota siguiente, la cual es A#4, cuya frecuencia es 466.16 Hz. La diferencia entre las dos es 26.16, dicho valor se divide para dos, dando como resultado 13.08. Cuando el fine tuning sea 10.00, la frecuencia final será 453.08 Hz. Por supuesto, si el valor del fine tuning no es 10.00, el valor que se sume a la frecuencia final será proporcional al valor que tenga el fine tuning.

Para los valores de attack, decay y release; antes de implementarlos en la función de envolvente de Maximillian, se los multiplicó por los valores que se indica en la *Tabla 1*, debido a que de esta forma se consiguió que el tiempo que se demora cada fase sea congruente con los valores mostrados en la interfaz. Esto se considera un error en la envolvente de Maximillian, dado que, según su documentación, los parámetros de la función deben ser en milisegundos, pero al momento de realizar las pruebas, se tuvo que realizar el ajuste indicado.

Para los parámetros del Mixer, correspondientes a los valores de amplitud de los osciladores y del master, se cambiaron los valores primero a decibeles de entre -48 y 0 dBFS, dado que, como ya se indicó, de esta manera se representa la amplitud de una onda digitalmente. Posteriormente se usó la función de JUCE decibelsToGain para convertir el valor en decibeles a un valor que sea multiplicable por la onda y genere un cambio en su amplitud apreciable al momento de convertir el audio a analógico. La razón para usar decibeles se debe a que los seres humanos perciben mejor los cambios en volumen de manera logarítmica y no de manera linear, por lo tanto, se debe convertir el valor linear del parámetro a un valor logarítmico para que se perciban los cambios de amplitud de la onda (Pirkle, 2013).

Por último, se puede apreciar como el valor del cutoff se usa directo o modificado por el o los LFOs, así como el valor de la intensidad de los LFOs depende de parámetro que se decida modular. A continuación, se muestra una tabla donde se detalla la formula usada para cada parámetro. Es importante mencionar que las fórmulas fueron determinadas a través del método prueba y error, hasta llegar a un resultado satisfactorio al momento de escucharlo, y que no ocasione errores en el procesamiento ni renderización del sonido.

Parámetro	**Fórmula con un LFO**	**Fórmula con dos LFOs**
Amplitud Osciladores	Osc * LFO * lfoIntensity	Osc * [(LFO1 * lfo1Intensity/10) + (LFO2 * lfo2Intensity/10)]
Frecuencia Osciladores	Frecuencia Osc + (LFO * $lfoIntensity^3$)	Frecuencia Osc + [(LFO1 * lfo1Intensity) + (LFO2 * lfo2Intensity)]
Cutoff	Cutoff + (LFO * $lfoIntensity^3$)	Cutoff + [(LFO1 * lfo1Intensity) + (LFO2 * lfo2Intensity)]
Master	Master * LFO * lfoIntensity	Master * [(LFO1 * lfo1Intensity/10) + (LFO2 * lfo2Intensity/10)]

Tabla 2. Fórmula aplicada a cada parámetro que puede ser modulado por los LFOs. Elaboración propia.

CAPÍTULO 6: RESULTADOS

Regresando una vez más al Capítulo de diseño del presente trabajo, uno de los ejes de diseño fue la gama de sonidos. En dicho eje se habló de la necesidad de implementar varias técnicas de síntesis, además de parámetros simples, como el fine tuning, y LFOs para lograr, precisamente, una amplia gama de sonidos. Todo esto, como se comprobó en el capítulo de pruebas, se lo implementó satisfactoriamente en el sintetizador.

Ahora bien, también se mencionó la necesidad de desarrollar presets como manera de guiar al usuario a través de la gama de sonidos. Los presets funcionan, además, como evidencia del resultado final del sintetizador, dado que se usan todos -o al menos la mayoría- de los componentes para generar varios sonidos que pueden ser usados en cualquier composición musical, y demuestran las capacidades de síntesis de sonido del sintetizador. En el presente capitulo se describen los presets desarrollados, indicando la configuración de los parámetros, así como la forma de onda y espectro de frecuencias resultantes, de forma que se evidencie las diferencias entre cada uno. Es importante mencionar que solo se muestran los valores de los parámetros que tienen alguna modificación. Si el valor de un parámetro no se detalla, se entiende que su valor es el mínimo de su respectivo rango, o se aprecia en la figura de la interfaz gráfica. Por último, al igual que los componentes y secciones, se decidió nombrar a los presets en inglés.

Evidentemente, la mejor forma de apreciar las diferencias y características de cada preset es escuchándolos. Debido a esto, en la sección de Anexos del presente trabajo se halla un enlace hacia una carpeta en la nube donde se encuentran archivos de audio de cada preset, los cuales se recomienda revisar. En la misma carpeta se puede encontrar además una composición musical corta que utiliza varios de los presets, mostrando así un pequeño ejemplo de la capacidad y utilidad que presenta el sintetizador de sonido de software desarrollado.

PRESETS DESARROLLADOS

Preset 1: Electronic Bass

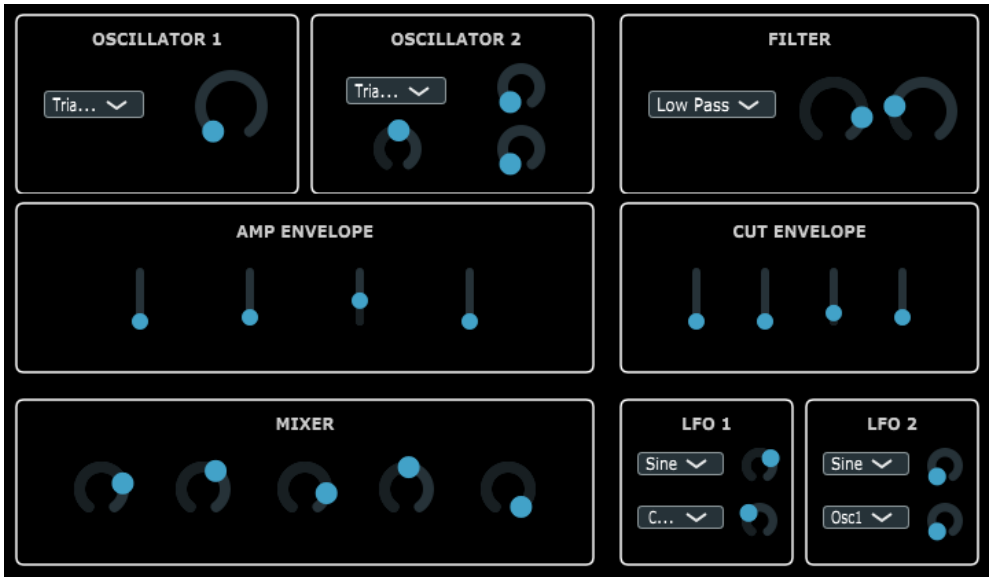

Figura 69. Preset 1: Electronic Bass GUI - Bajo con sonido electrónico. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Oscillators		**Mixer**	
Fine Tuning	5.00	Osc1	7.5
Filter		Osc2	6.0
Cutoff	5500	SubOsc	8.5
Resonance	3.0	Noise	5.0
Envelopes		Master	9.0
Amp Decay	260ms	**LFOs**	
Amp Sustain	0.4	LFO1 Dest	Cutoff
Cut Sustain	0.17	LFO1 Rate	4.0hz
Cut Release	170ms	LFO1 Intensity	2.0

Tabla 3. Preset 1: Electronic Bass - Valores de los parámetros. Elaboración propia.

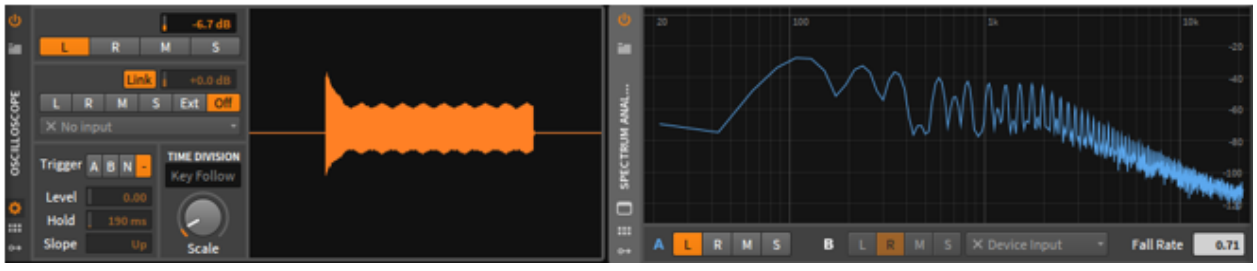

Figura 70. Preset 1: Electronic Bass - Forma de onda y Espectro de frecuencias. Elaboración propia.

Preset 2: Classic Bass

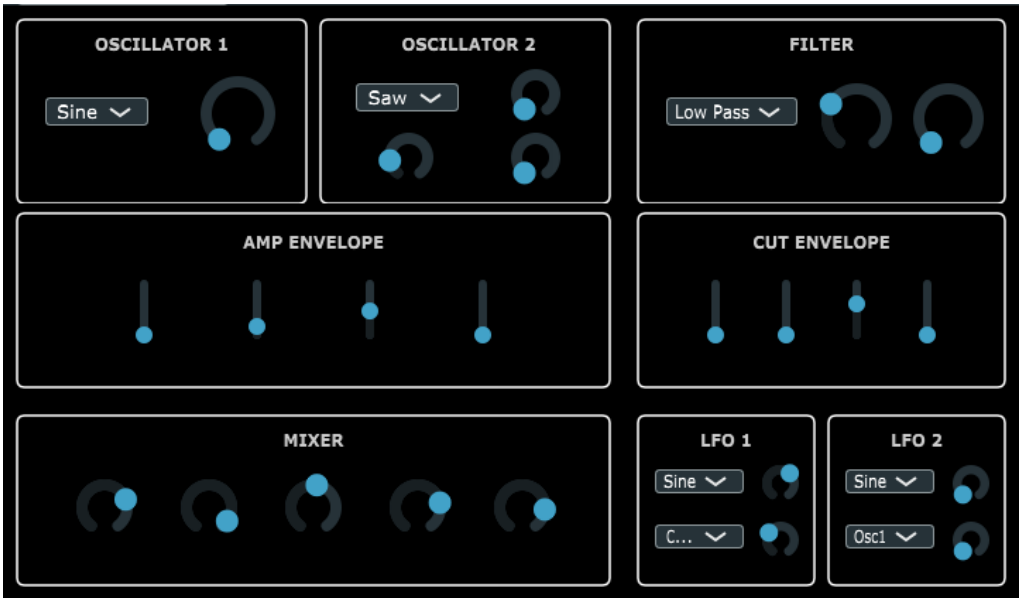

Figura 71. Preset 2: Classic Bass GUI – Sonido de un bajo clásico. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Oscillators		**Mixer**	
Fine Tuning	1.60	Osc1	7.5
Filter		Osc2	9.5
Cutoff	410hz	SubOsc	5.5
Envelopes		Noise	7.5
Amp Decay	340ms	Master	8.5
Amp Sustain	0.45	**LFOs**	
Cut Decay	90ms	LFO1 Dest	Cutoff
Cut Sustain	0.60	LFO1 Rate	4.7hz
		LFO1 Intensity	3.0

Tabla 4. Preset 2: Classic Bass - Valores de los parámetros. Elaboración propia.

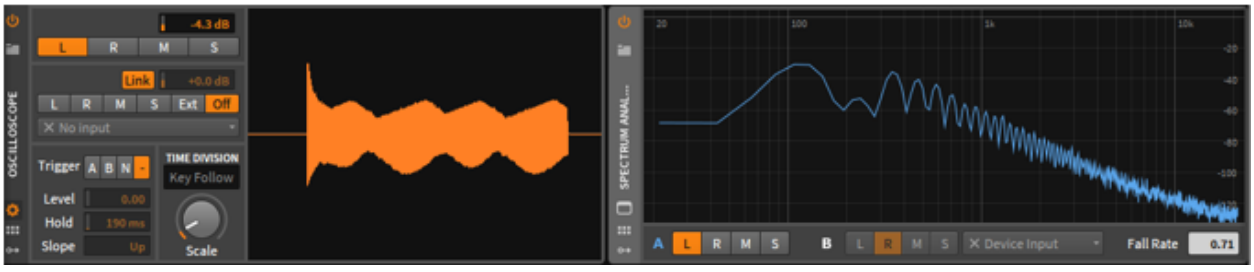

Figura 72. Preset 2: Classic Bass - Forma de onda y Espectro de frecuencias. Elaboración propia.

Preset 3: Distorted Bass

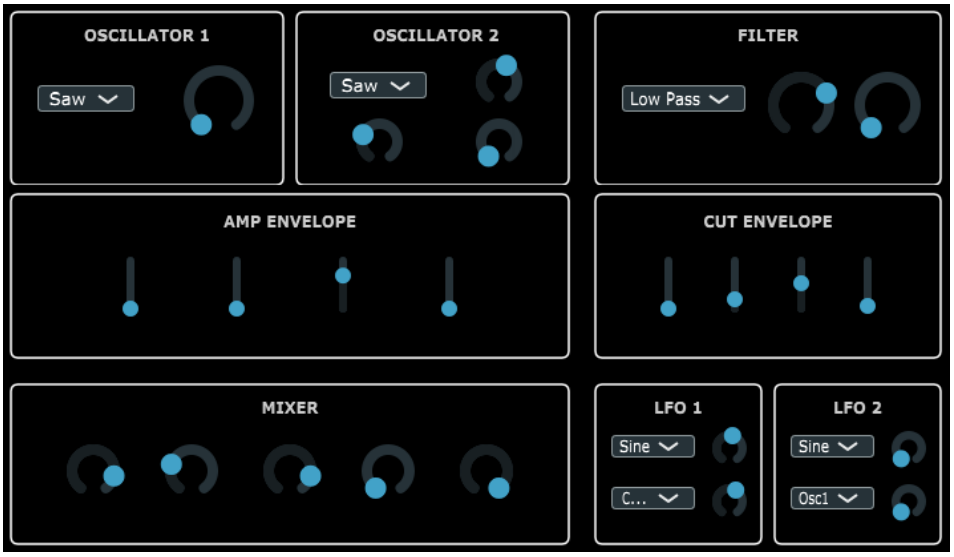

Figura 73. Preset 3: Distorted Bass GUI – Sonido de un bajo eléctrico con distorsión. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Oscillators		**Mixer**	
Osc2 Semitone	7	Osc1	8.5
Fine Tuning	2.70	Osc2	2.5
Filter		SubOsc	8.5
Cutoff	3600hz	Master	8.5
Envelopes		**LFOs**	
Amp Sustain	0.70	LFO1 Dest	Cutoff
Cut Decay	600ms	LFO1 Rate	3.5hz
Cut Sustain	0.55	LFO1 Intensity	6.0

Tabla 5. Preset 3: Distorted Bass - Valores de los parámetros. Elaboración propia.

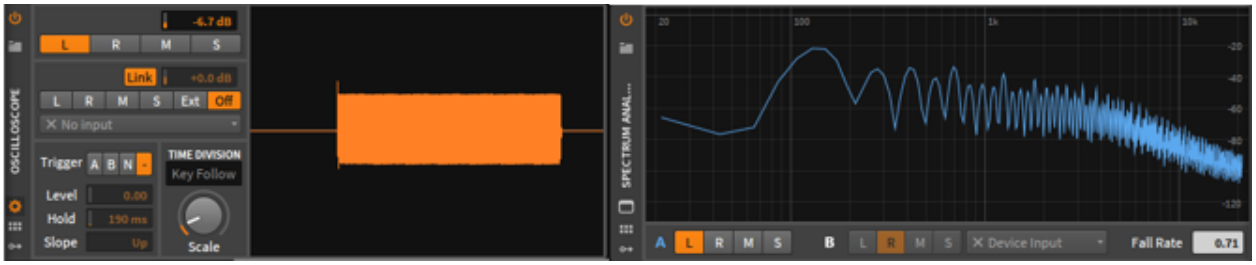

Figura 74. Preset 3: Distorted Bass - Forma de onda y Espectro de frecuencias. Elaboración propia.

Preset 4: Clear Piano

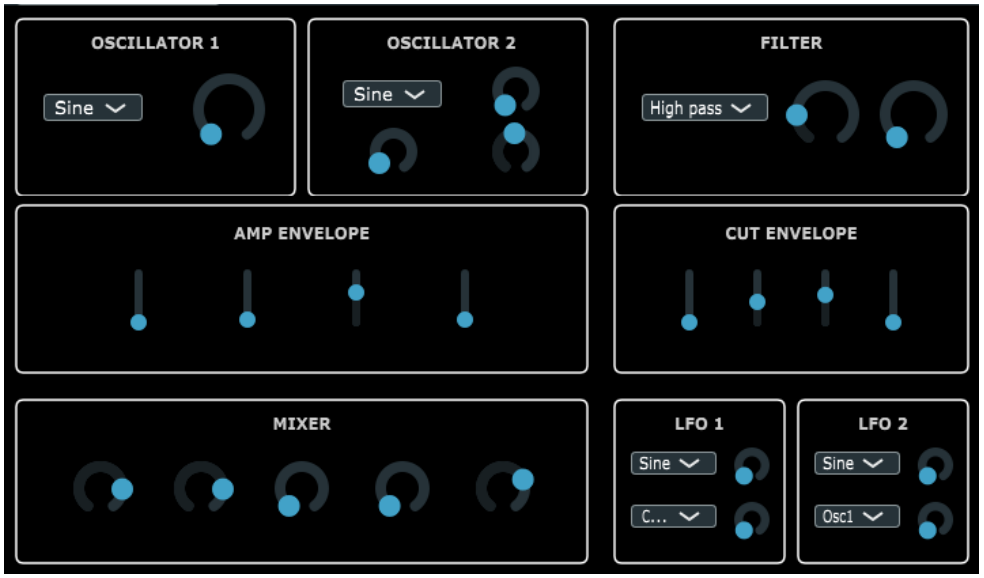

Figura 75. Preset 4: Clear Piano GUI – Piano con sonido limpio. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Oscillators		**Envelopes**	
Fine Tuning	0.5	Cut Decay	1000ms
FM	5.0	Cut Sustain	0.60
Filter		**Mixer**	
Cutoff	150hz	Osc1	8.0
Envelopes		Osc2	8.0
Amp Decay	100ms	Master	7.5
Amp Sustain	0.6		
Amp Release	350ms		

Tabla 6. Preset 4: Clear Piano - Valores de los parámetros. Elaboración propia.

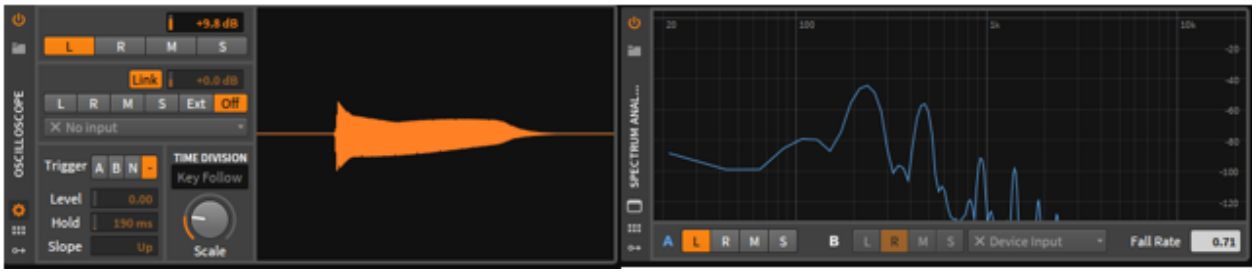

Figura 76. Preset 4: Clear Piano - Forma de onda y Espectro de frecuencias. Elaboración propia.

Preset 5: Soft Piano

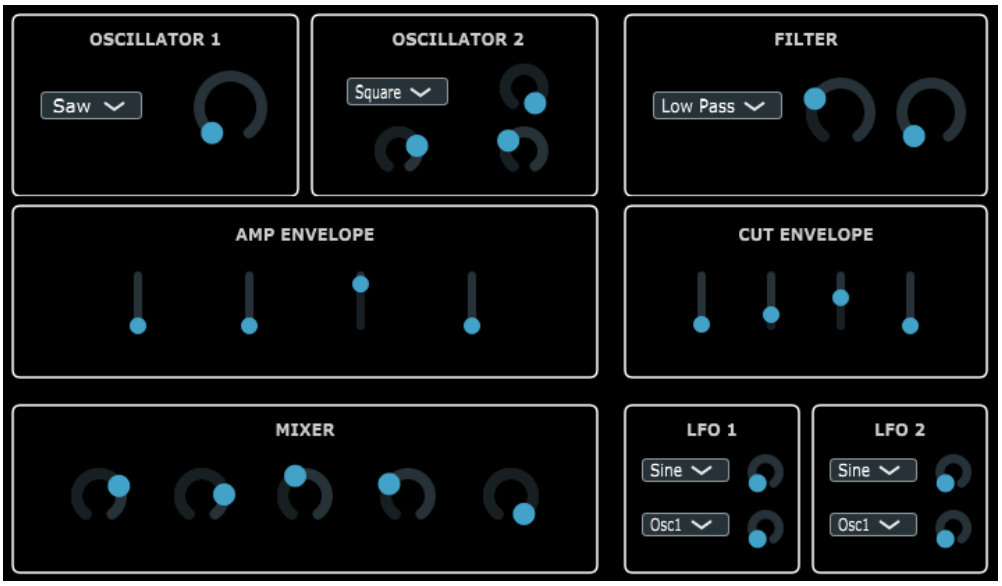

Figura 77. Preset 5: Soft Piano GUI – Piano con un sonido suave y efecto ambiente. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Oscillators		**Envelopes**	
Osc2 Semitone	12	Cut Attack	100ms
Fine Tuning	7.6	Cut Decay	680ms
FM	3.0	Cut Sustain	0.55
Filter		**Mixer**	
Cutoff	400hz	Osc1	7.0
Resonance	3.0	Osc2	8.0
Envelopes		SubOsc	4.0
Amp Attack	100ms	Noise	3.0
Amp Sustain	0.85	Master	8.5
Amp Release	100ms		

Tabla 7. Preset 5: Soft Piano - Valores de los parámetros. Elaboración propia.

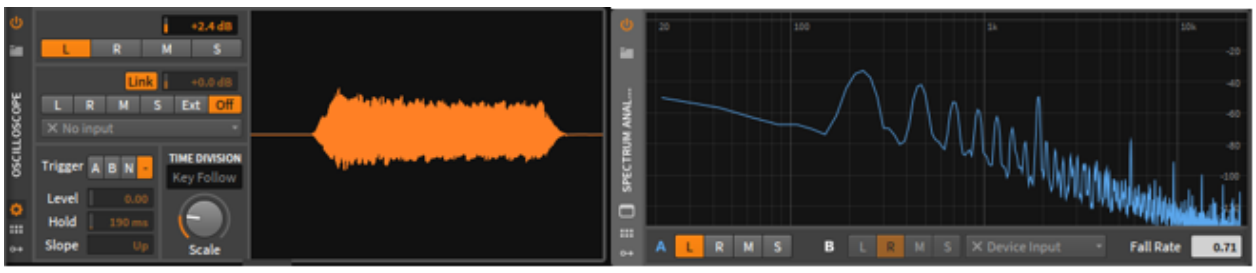

Figura 78. Preset 5: Soft Piano - Forma de onda y Espectro de frecuencias. Elaboración propia.

Preset 6: Magic Xylophone

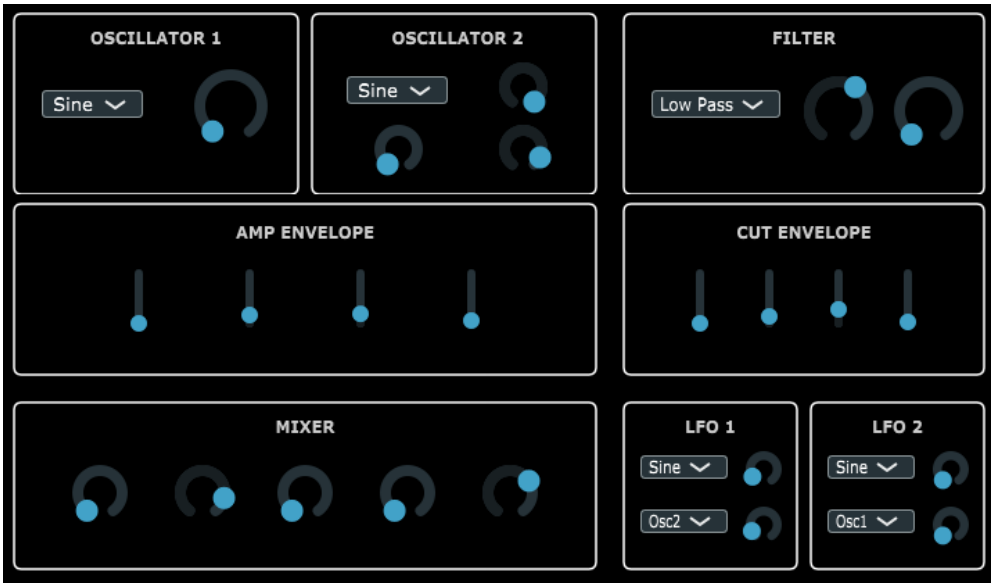

Figura 79. Preset 6: Magic Xylophone GUI – Sonido de un xilófono con efecto ambiente. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Oscillators		**Envelopes**	
Osc2 Semitone	12	Cut Decay	400ms
FM	9.0	Cut Sustain	0.25
Filter		Cut Release	100ms
Cutoff	2500hz	**Mixer**	
Envelopes		Osc2	8.0
Amp Decay	500ms	Master	8.0
Amp Sustain	0.2	**LFOs**	
Amp Release	180ms	LFO1 Rate	0.1hz
		LFO1 Intensity	1.0

Tabla 8. Preset 6: Magic Xylophone - Valores de los parámetros. Elaboración propia.

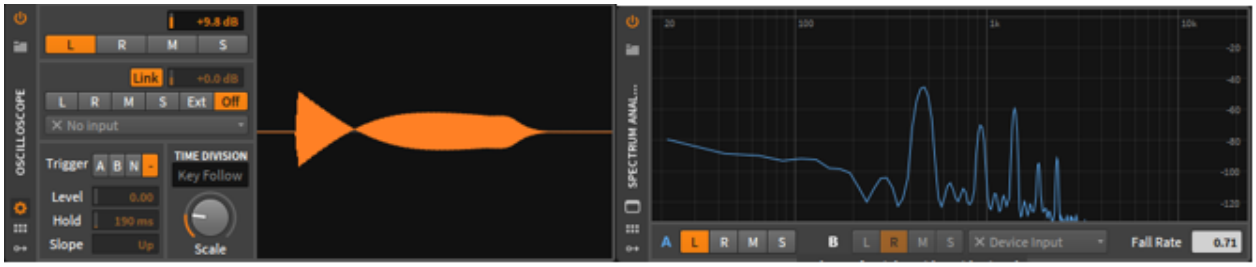

Figura 80. Preset 6: Magic Xylophone - Forma de onda y Espectro de frecuencias. Elaboración propia.

Preset 7: Synth Choir

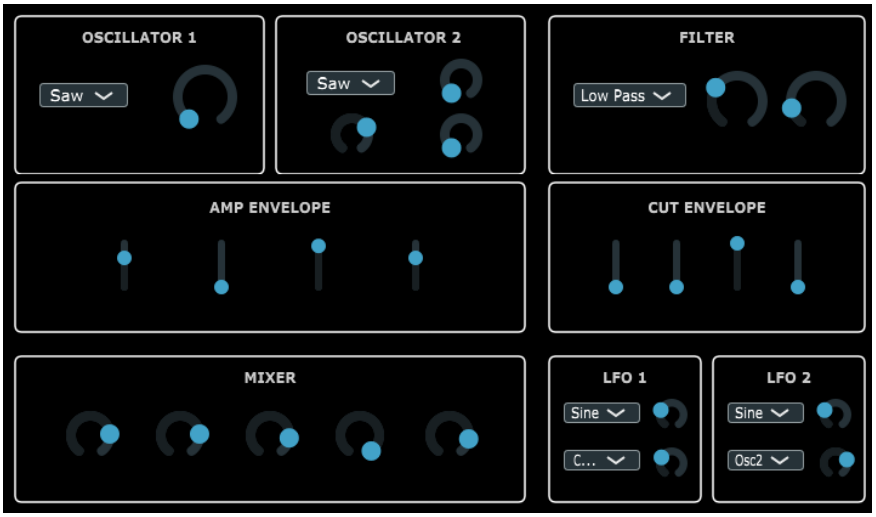

Figura 81. Preset 7: Synth Choir GUI – Sonido de sintetizador con profundidad ambiente. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Oscillators		**Mixer**	
Fine Tuning	7.20	Osc1	8.0
Filter		Osc2	8.0
Cutoff	450hz	SubOsc	8.5
Resonance	2.0	Noise	7.0
Envelopes		Master	8.5
Amp Attack	2000ms	**LFOs**	
Amp Sustain	0.9	LFO1 Dest	Cutoff
Amp Release	2000ms	LFO1 Rate	1.0hz
Cut Attack	200ms	LFO1 Intensity	3.0
Cut Decay	670ms	LFO2 Rate	1.0hz
Cut Sustain	0.8	LFO2 Intensity	7.5
Cut Release	200ms		

Tabla 9. Preset 7: Synth Choir - Valores de los parámetros. Elaboración propia.

Figura 82. Preset 7: Synth Choir - Forma de onda y Espectro de frecuencias. Elaboración propia.

Preset 8: Synth Lead

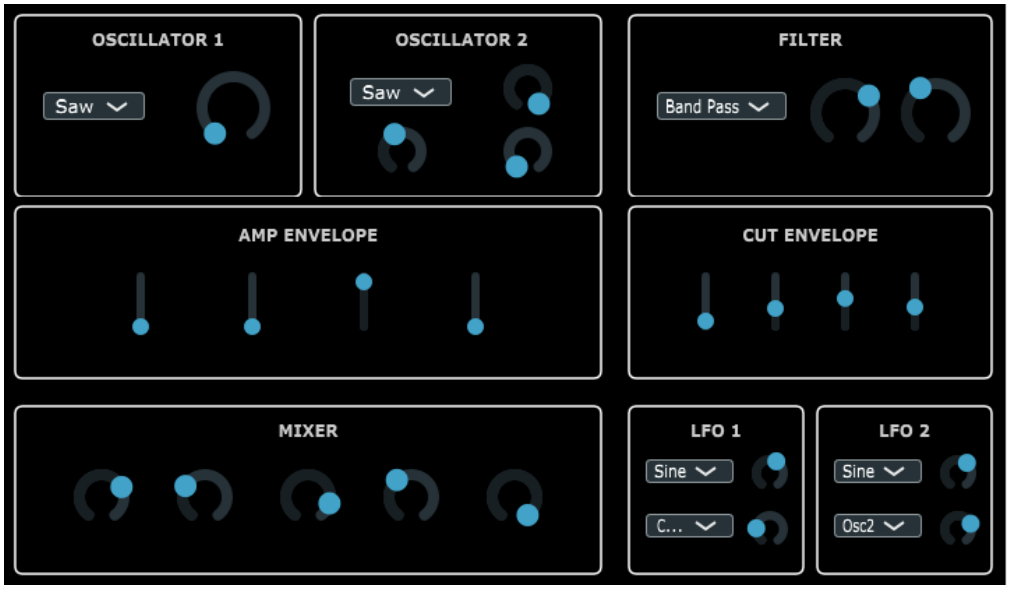

Figura 83. Preset 8: Synth Lead GUI – Sonido de Sintetizador para riffs o solos. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Oscillators		**Mixer**	
Osc2 Semitone	12	Osc1	7.5
Fine Tuning	4.10	Osc2	3.0
Filter		SubOsc	9.0
Cutoff	3200hz	Noise	3.5
Resonance	4.5	Master	9.0
Envelopes		**LFOs**	
Cut Attack	260ms	LFO1 Dest	Cutoff
Cut Decay	1000ms	LFO1 Rate	4.0hz
Cut Sustain	0.6	LFO1 Intensity	2.0
Cut Release	1000ms	LFO2 Rate	4.5hz
		LFO2 Intensity	7.0

Tabla 10. Preset 8: Synth Lead - Valores de los parámetros. Elaboración propia.

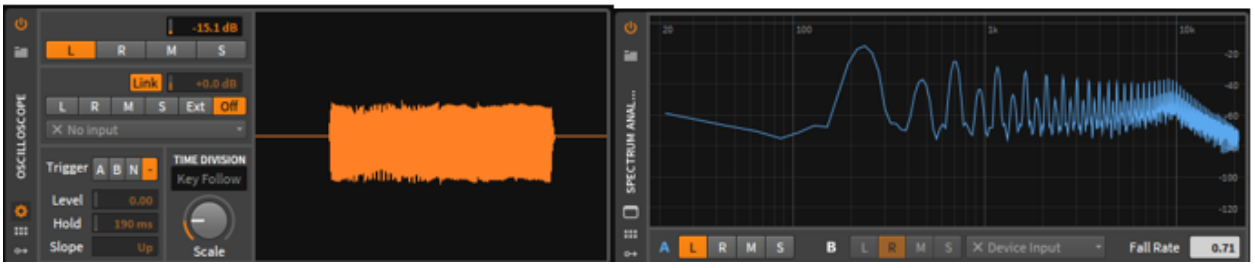

Figura 84. Preset 8: Synth Lead - Forma de onda y Espectro de frecuencias. Elaboración propia.

Preset 9: Bird Sing

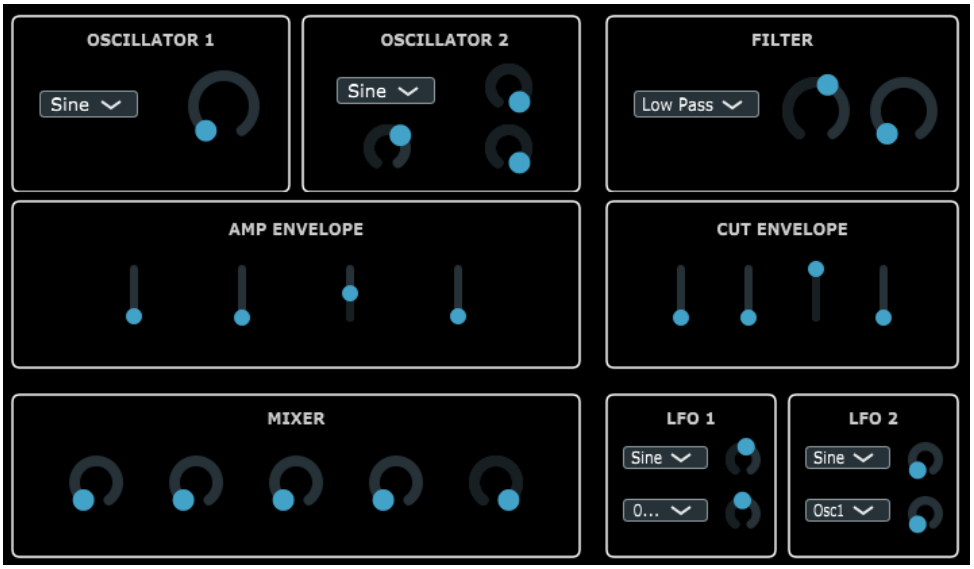

Figura 85. Preset 9: Bird Sing GUI – Sonido similar al canto de los pájaros. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Oscillators		**Envelopes**	
Osc2 Semitone	12	Cut Sustain	0.90
Fine Tuning	6.60	**Mixer**	
FM	10.0	Master	9.0
Filter		**LFOs**	
Cutoff	2200hz	LFO1 Dest	Osc2 Freq
Envelopes		LFO1 Rate	3.5hz
Amp Attack	100ms	LFO1 Intensity	5.0
Amp Sustain	0.50		
Amp Release	100ms		

Tabla 11. Preset 9: Bird Sing - Valores de los parámetros. Elaboración propia.

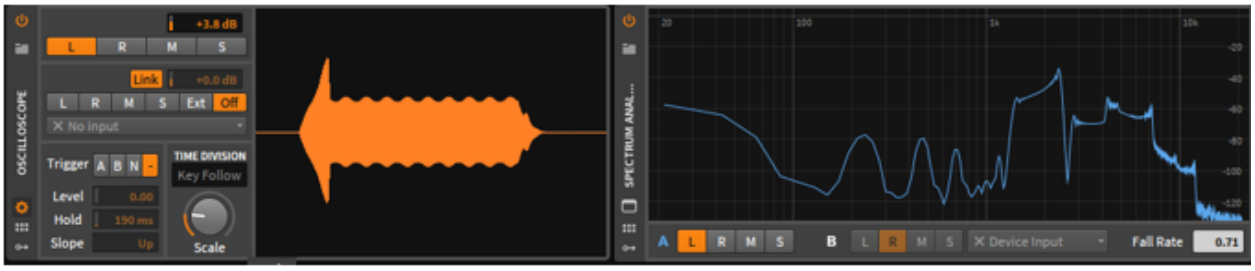

Figura 86. Preset 9: Bird Sing - Forma de onda y Espectro de frecuencias. Elaboración propia.

Preset 10: Smooth Shaker

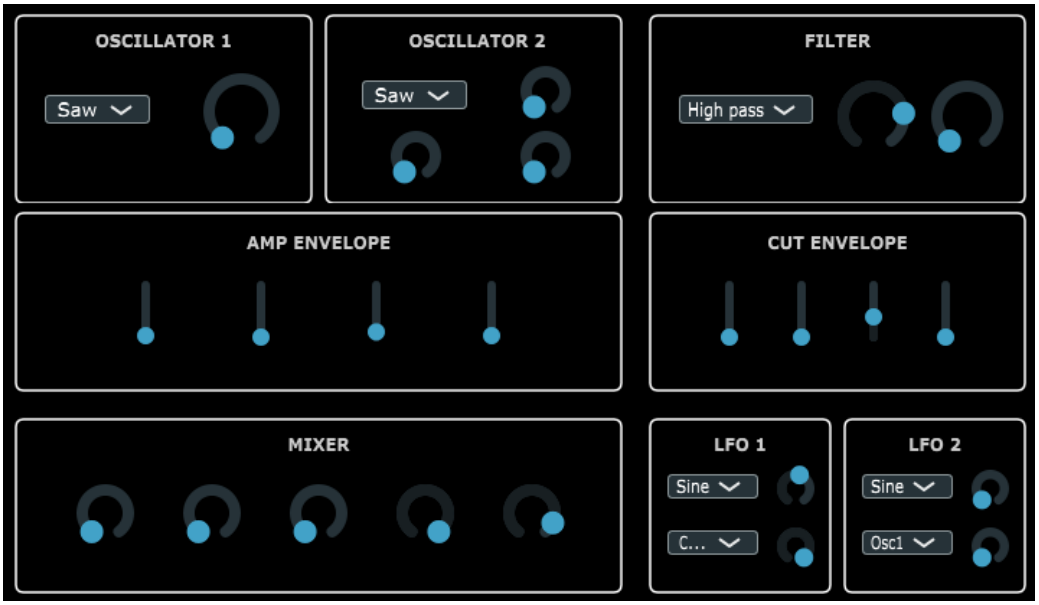

Figura 87. Preset 10: Smooth Shaker GUI – Sonido de percusión shaker. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Filter		**Mixer**	
Cutoff	4500hz	Noise	10.0
Envelopes		Master	9.0
Amp Attack	100ms	**LFOs**	
Amp Sustain	0.10	LFO1 Dest	Cutoff
Amp Release	100ms	LFO1 Rate	3.5hz
Cut Sustain	0.40	LFO1 Intensity	10.0

Tabla 12. Preset 10: Smooth Shaker - Valores de los parámetros. Elaboración propia.

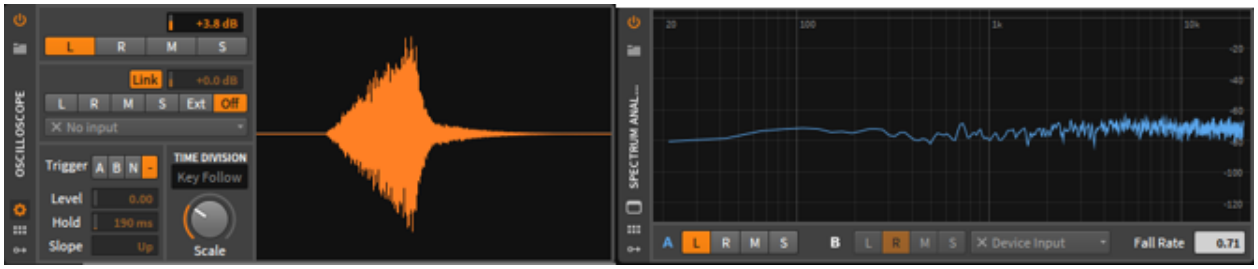

Figura 88. Preset 10: Smooth Shaker - Forma de onda y Espectro de frecuencias. Elaboración propia.

Preset 11: Note Kick

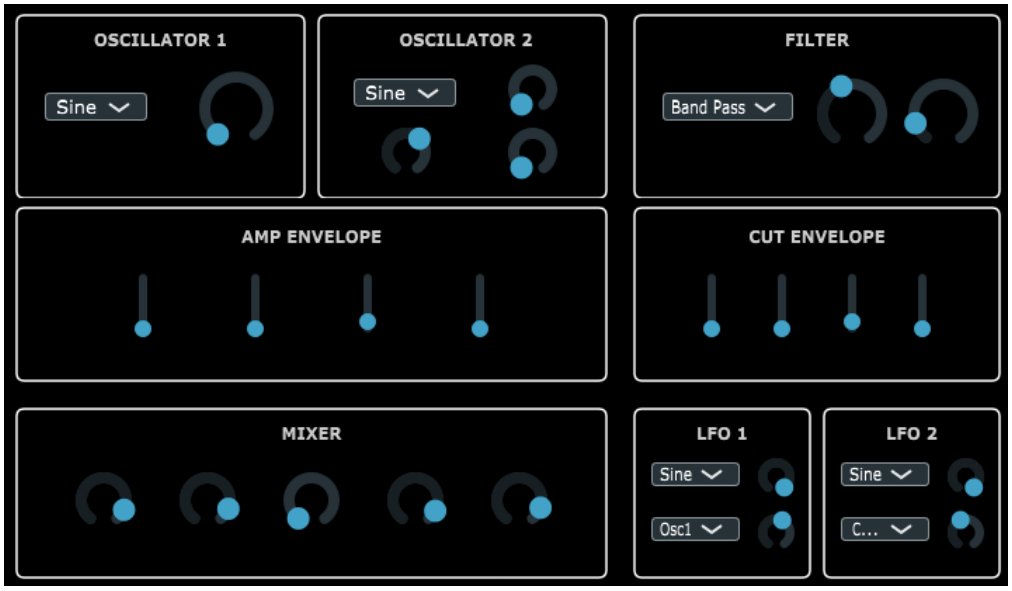

Figura 89. Preset 11: Note Kick GUI – Sonido de percusión de bombo y toms. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Oscillators		**Mixer**	
Fine Tuning	6.5	Noise	9.0
Filter		Master	9.0
Cutoff	1000hz	**LFOs**	
Resonance	2	LFO1 Rate	10.0hz
Envelopes		LFO1 Intensity	6.0
Amp Sustain	0.15	LFO2 Dest	Cutoff
Cut Sustain	0.03	LFO2 Rate	10.0hz
Mixer		LFO2 Intensity	4.0
Osc1	8.5		
Osc2	8.5		

Tabla 13. Preset 11: Note Kick - Valores de los parámetros. Elaboración propia.

Figura 90. Preset 11: Note Kick - Forma de onda y Espectro de frecuencias. Elaboración propia.

Preset 12: Noise Effect

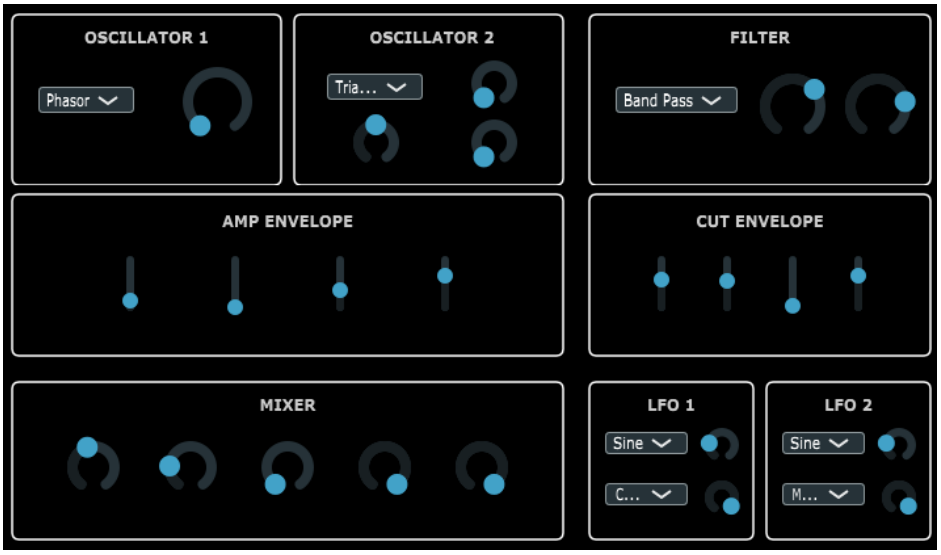

Figura 91. Preset 12: Noise Effect GUI – Efecto de swell usando el generador de ruido. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Oscillators		**Mixer**	
Fine tuning	5.0	Osc1	4.5
Filter		Osc2	2.0
Cutoff	3140hz	Noise	10.0
Resonance	8.0	Master	10.0
Envelopes		**LFOs**	
Amp Attack	500ms	LFO1 Dest	Cutoff
Amp Sustain	0.4	LFO1 Rate	0.7hz
Amp Release	2000ms	LFO1 Intensity	10.0
Cut Attack	1500ms	LFO2 Dest	Master
Cut Decay	1500ms	LFO2 Rate	0.7hz
Cut Sustain	0.08	LFO2 Intensity	10.0
Cut Release	2000ms		

Tabla 14. Preset 12: Noise Effect - Valores de los parámetros. Elaboración propia.

Figura 92. Preset 12: Noise Effect - Forma de onda y Espectro de frecuencias. Elaboración propia.

Preset 13: Ghost

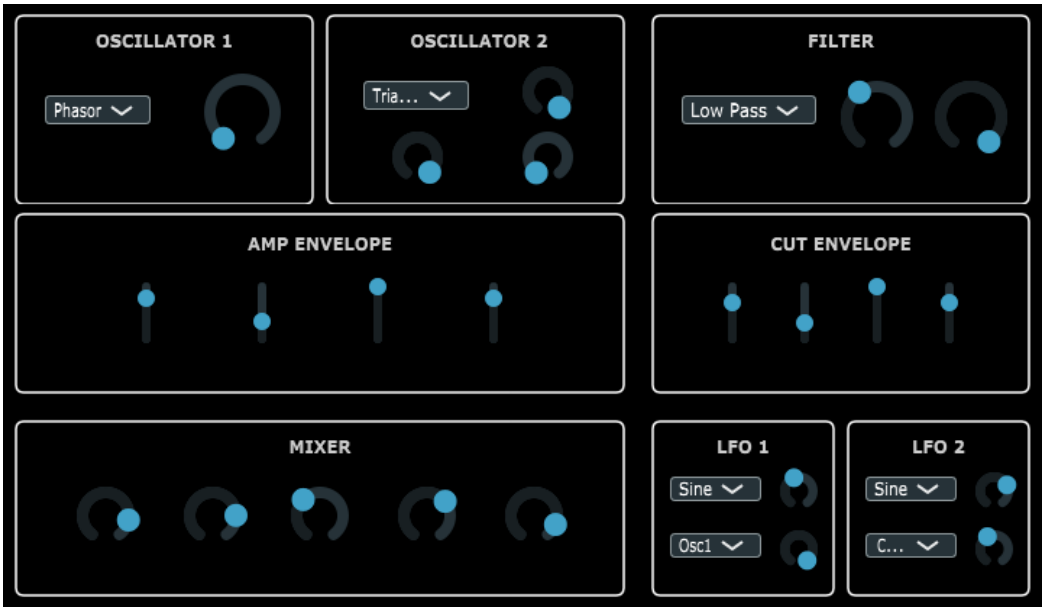

Figura 93. Preset 13: Ghost GUI – Efecto ambiente con sensación de misterio. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Oscillators		**Mixer**	
Osc2 Semitone	12	Osc1	8.5
Fine Tuning	10.0	Osc2	8.0
Filter		SubOsc	3.5
Cutoff	770hz	Noise	7.0
Resonance	10	Master	8.0
Envelopes		**LFOs**	
Amp Attack	2300ms	LFO1 Rate	2.3hz
Amp Decay	1000ms	LFO1 Intensity	10.0
Amp Sustain	0.9	LFO2 Dest	Cutoff
Amp Release	2300ms	LFO2 Rate	5.7hz
Cut Attack	2000ms	LFO2 Intensity	4.0
Cut Decay	1000ms		
Cut Sustain	0.9		
Cut Release	2000ms		

Tabla 15. Preset 13: Ghost - Valores de los parámetros. Elaboración propia.

Figura 94. Preset 13: Ghost - Forma de onda y Espectro de frecuencias. Elaboración propia.

Preset 14: Space Ambient

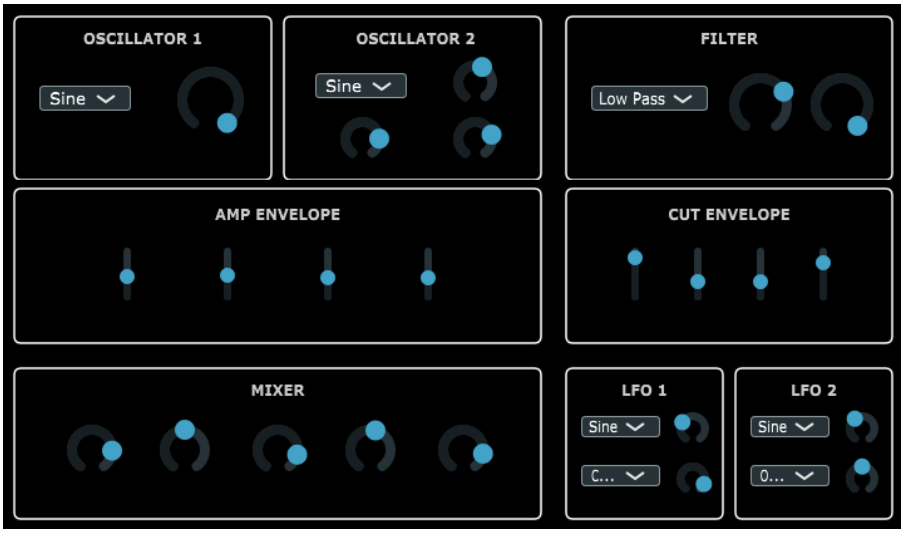

Figura 95. Preset 14: Space Ambient GUI – Efecto ambiente con sensación espacial. Elaboración propia.

Parámetro	Valor	Parámetro	Valor
Oscillators		**Mixer**	
Osc1 Semitone	12	Osc1	8.5
Osc2 Semitone	7	Osc2	5.0
Fine Tuning	8.00	SubOsc	8.5
FM	7.60	Noise	5.5
Filter		Master	9.0
Cutoff	3250hz	**LFOs**	
Resonance	10.0	LFO1 Dest	Cutoff
Envelopes		LFO1 Rate	1.2hz
Amp Attack	1500ms	LFO1 Intensity	9.0
Amp Decay	1500ms	LFO2 Dest	Osc2 Freq
Amp Sustain	0.40	LFO2 Rate	1.8hz
Amp Release	1500ms	LFO2 Intensity	5.0
Cut Attack	2500ms		
Cut Decay	1000ms		
Cut Sustain	0.33		
Cut Release	2300ms		

Tabla 16. Preset 14: Space Ambient - Valores de los parámetros. Elaboración propia.

Figura 96. Preset 14: Space Ambient - Forma de onda y Espectro de frecuencias. Elaboración propia.

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

1. La característica fundamental que diferencia a un sintetizador de sonido de otros instrumentos musicales es que este permite modificar, en poca o gran medida, el timbre de un sonido, usando una o varias técnicas de síntesis, implementadas a través de la combinación de varios componentes, logrando así una gama de sonidos muy variada; razón por la cual es un instrumento que se puede usar en diversos géneros musicales.

2. Para modificar el timbre, la onda de sonido es manipulada a través de varios componentes que modifican distintas propiedades como: su forma inicial, espectro de frecuencias, evolución en el tiempo, etc. Este proceso es secuencial y acumulativo, y cada componente aporta, en mayor o menor medida, al resultado final. Por lo tanto, entender que modificación realiza cada componente individualmente, significa entender de mejor manera su impacto en el resultado final, logrando así comprender todo el proceso de síntesis.

3. La cualidad que diferencia a cada sintetizador de sonido de otros sintetizadores es su diseño. Al existir varias técnicas de síntesis y varios componentes, cada fabricante los escoge y organiza de acuerdo a sus necesidades y objetivos. Esta es la razón por la que existe tanta diversidad de sintetizadores en el mercado, siendo cada uno capaz generar timbres únicos y completamente distintos a los de otros sintetizadores. Cada sintetizador de sonido es un mundo diferente con la capacidad de producir resultados únicos, por lo que, al momento de utilizar alguno, es necesario estudiar sus funcionalidades e identificar sus particularidades, de forma que se pueda aprovechar al máximo sus capacidades.

4. Al momento de diseñar un sintetizador de sonido, lo más importante es tener claro el mercado objetivo al que se pretende llegar, de forma que se puedan plantear objetivos en función de dicho mercado. De esta forma, se facilita la elección, tanto de las técnicas de síntesis, como de los componentes a implementar, así como la elaboración de la interfaz gráfica y las funcionalidades generales del sintetizador. Así mismo, entender las ventajas y desventajas que presenta el mundo digital, comparado con las posibilidades tecnológicas de épocas pasadas, permite marcar un camino claro el cual seguir. Todo esto, sumado a cierto nivel de experiencia y conocimiento musical, se traduce en un diseño final mucho más funcional y efectivo.

5. Antes de empezar el desarrollo como tal, es imprescindible entender cómo trabaja el framework escogido, es decir, como se estructura un proyecto a nivel de archivos, clases y funciones. Esto, sumado a conocer cómo funciona el lenguaje de programación que se va a utilizar, permite planificar de mejor manera el desarrollo del programa, reduce la posibilidad de errores y aumenta la velocidad de codificación, resultando en un producto mucho más estable y que refleja adecuadamente el diseño realizado. Además, es importante que el framework escogido disponga de una comunidad de desarrolladores detrás, de modo que se tenga un respaldo a la hora de encontrar información y soluciones a errores.

6. La decisión de utilizar JUCE como framework de desarrollo demostró ser acertada debido a las grandes facilidades que presenta esta herramienta para la elaboración de proyectos de audio, destacándose sobre todo la intuitiva forma en la que se estructura el proyecto, la velocidad de depuración y de ejecución de pruebas, el eficiente manejo de recursos tanto de hardware como de software, y la gran cantidad de documentación disponible. Todo esto permitió un desarrollo iterativo e incremental muy eficaz y eficiente, dando como resultado un producto final estable y completamente funcional.

7. A nivel de código, independientemente del framework que se utilice, una onda de sonido no es más que un arreglo de datos donde se almacenan los distintos valores de su amplitud, medidos en varios momentos del tiempo. Por lo tanto, todo algoritmo de procesamiento de audio debe ser implementado sobre dicho arreglo de datos. Este concepto, sumado a un entendimiento del proceso de conversión analógico-digital y viceversa, son esenciales para lograr manipular la onda de sonido digitalmente de forma eficaz y eficiente, consiguiendo así resultados satisfactorios.

8. Para verificar el correcto funcionamiento de los componentes del sintetizador, tanto de forma individual como en conjunto, es esencial entender toda la teoría detrás en cuanto a las propiedades del sonido y al proceso de síntesis. De esta forma se puede conocer cómo se debe comportar un componente especifico y se puede establecer parámetros que permitan garantizar que el sintetizador de sonido funcione de la manera esperada.

9. La mejor forma de demostrar la capacidad de un sintetizador de sonido es a través de presets, dado que estos muestran lo más importante del sintetizador: el sonido final. Los presets permiten acceder, de manera rápida, a timbres preestablecidos, mostrando así al usuario la gama de sonidos que el sintetizador es capaz de producir; dándole, además, la posibilidad de crear sus propias configuraciones a partir de los presets, logrando así que se relacione con el proceso de síntesis de manera más sencilla e intuitiva.

10. El desarrollo del sintetizador de sonido del presente trabajo se facilitó, en gran medida, gracias a la disponibilidad de información en internet, tanto teórica como en relación a la implementación. Dicha información, al igual que cualquiera que sea de libre acceso, resultan un aporte incalculable al desarrollo tecnológico y social. Por este motivo, el presente trabajo busca mantener la misma línea y conseguir que los resultados sean accesibles a la mayor cantidad de personas posibles, las cuales puedan trabajar sobre lo realizado y expandir sus funcionalidades, más allá de buscar solamente un redito económico.

RECOMENDACIONES

1. Al ser el sintetizador un instrumento musical, es esencial que el usuario final tenga conocimientos, al menos básicos, sobre teoría musical, para entender claramente conceptos que no fueron explicados a detalle en el presente trabajo, como “semitono” u “octava”. Es muy recomendable, en caso de no tener claro estos conceptos, realizar un repaso sobre teoría musical antes de utilizar el sintetizador de sonido. Así mismo, dado que se usa un controlador MIDI con un teclado de piano para controlar el sintetizador, es preferible -aunque no imprescindible- que el usuario final tenga cierta habilidad para la interpretación de un piano.

2. Para expandir la funcionalidad del sintetizador se recomienda ampliar las capacidades de la síntesis FM, dado que, si bien es cierto se obtuvieron resultados interesantes de la manera en que se la implemento en el presente trabajo, si se utilizan más osciladores, de modo que se puede combinar varias ondas como moduladoras y portadoras, se puede aprovechar el máximo las capacidades de esta técnica de síntesis y ampliar aún más la gama de sonidos disponibles del sintetizador. Así mismo, se puede incluir la síntesis basada en “samples”, la cual es muy utilizada en sintetizadores digitales ya que es relativamente fácil de implementar y aporta más realismo al instrumento.

3. El manejo de ciertos componentes, como el filtro o las envolventes, se puede simplificar si se implementa un manejo visual de los mismos, es decir, que en la interfaz se muestre directamente el espectro de frecuencias y la forma de onda y, sobre estos, se pueda directamente seleccionar la frecuencia de corte o los parámetros ADSR. De esta forma, el uso de estos componentes se vuelve mucho más intuitivo para el usuario.

4. La forma más sencilla de lograr que el sintetizador sea más atractivo para el usuario final es expandiendo la librería de presets del mismo, dado que todavía queda mucha variedad de timbres por explorar, sin necesidad de aumentar el número de componentes. Mientras más presets se tengan, más versátil será el sintetizador y, por lo tanto, más llamara la atención de potenciales usuarios.

5. De modo que el uso del sintetizador sea más funcional en conjunto con un DAW, se puede agregar la posibilidad de sincronizar ciertos parámetros, como, por ejemplo, el “rate” de los LFOs o el “attack” de la envolvente de amplitud; directamente con el beat que se configure en el DAW. De esta forma el sintetizador de sonido se adaptaría mucho más para el uso en producciones musicales.

6. Para aprovechar de mejor manera las posibilidades que ofrece el protocolo MIDI, se puede agregar la posibilidad de incluir secuencias programables al sintetizador de sonido, dado que esta es una de las funcionalidades más usadas de MIDI. Esto permite que se puedan programar y ejecutar melodías sin necesidad de que el usuario las interprete, logrando así implementar otra forma de controlar al sintetizador, más allá de solamente el controlador MIDI como tal, aumentado así su funcionalidad.

ANEXOS

CÓDIGO FUENTE

Librería Maximillian

En el siguiente enlace se encuentra disponible la librería Maximillian desarrollada en C++ para el procesamiento de música y audio, misma que fue usada en el presente trabajo.

https://github.com/micknoise/Maximilian.git

Sintetizador de Sonido de Software – RonSynth

En el siguiente enlace se encuentra el código fuente del sintetizador de sonido de software desarrollado en el presente trabajo. Si se desea compilar el proyecto se debe agregar la librería Maximillian. Se recomienda usar la versión 5.3.2 de JUCE y Visual Studio 2015 como IDE.

https://github.com/RonnyCG/RonSynth-.git

AUDIOS PRESETS

Carpeta General

A través del siguiente enlace se puede acceder a una carpeta en la nube donde se encontrarán archivos de audio MP3 de cada preset detallado en el Capítulo 6: Resultados. El nombre de cada archivo de audio es el mismo de cada preset. Así mismo se encontrará un archivo de audio adicional denominado “Composición Synth” el cual corresponde a una composición musical realizada usando varios de los presets del sintetizador.

https://drive.google.com/open?id=1F_5xlRx_n1Gszdl7E1YQwzt7EvW9d933

BIBLIOGRAFÍA

alanoneuser. (16 de Diciembre de 2012). *The Story of the RCA Synthesizer [archivo de video]*. Obtenido de https://www.youtube.com/watch?v=rgN_VzEIZ1I&t=286s

AutomaticGainsay. (16 de Julio de 2018). *A History of Polyphony: Part 8- Paradigm Shift... the RCA Mark II [archivo de video]*. Obtenido de https://www.youtube.com/watch?v=gJAmQmdzn_Y

Helmholtz, H. (1912). *On the Sensations of Tone as a Physiological Basis for the Theory of Music.* Londres: Longmans, Green, and Co.

Kursell, J. (2009). *Ear and Instrument—Hermann v. Helmholtz's "On the Sensations of Tone as a Physiological Basis for the Theory of Music"*. Obtenido de Max Planck Institute for the History of Science: https://www.mpiwg-berlin.mpg.de/research/projects/DeptIII_Julia_Kursell

moogfoundation. (19 de Febrero de 2013). *The Synthesis of Synthesis- The Telharmonium [archivo de video]*. Obtenido de https://www.youtube.com/watch?v=AV34h-YCMbE

Pirkle, W. (2013). *Designing Audio Effects Plug-Ins in C++*. Burlington, MA: Focal Press.

Pirkle, W. (2015). *Designing Software Synthesizer Plug-Ins in C++*. Burlington, MA: Focal Press.

Polyphonic. (8 de Febrero de 2018). *Yamaha DX7 - El Sintetizador que definió los años 80. [archivo de video]*. Obtenido de https://www.youtube.com/watch?v=Q1Ha0MMT0aA

Real Academia Española. (2018). *síntesis*. Obtenido de Diccionario de la lengua española: https://dle.rae.es/?id=Xzp9ksD

Real Academia Española. (2018). *sonido*. Obtenido de Diccionario de la lengua española: https://dle.rae.es/?id=YMV5Hqd

Russ, M. (2009). *Sound Synthesis and Sampling.* Burlington, MA: Focal Press.

Shepard, B. K. (2013). *Refining Sound: A Practical Guide to Synthesis and Synthesizers.* New York: Oxford University Press.