UNIVERSIDAD POLITÉCNICA DE MADRID
ESCUELA TÉCNICA SUPERIOR DE INGENIEROS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

DISEÑO E IMPLEMENTACIÓN SOBRE FPGA DE UN PEDAL DE EFECTOS DIGITAL

GRADO EN INGENIERÍA DE TECNOLOGÍAS
Y SERVICIOS DE TELECOMUNICACIÓN

JAVIER OTERO MARTÍNEZ
Madrid, Junio 2019

DISEÑO E IMPLEMENTACIÓN SOBRE FPGA DE UN PEDAL DE EFECTOS DIGITAL

Autor: Javier Otero Martínez

Tutor: Pablo Ituero Herrero
Departamento de Ingeniería Electrónica

Me gustaría agradecer a los que me han apoyado todo este tiempo: mis padres, mi familia, Aida e Impostor mientras trabajaba en el proyecto. También me gustaría agradecer especialmente a los profesores que me han ayudado desinteresadamente cuando he acudido a ellos con alguna pregunta: Alfredo Sanz, Jose Parera, Fernando Gonzalez y por supuesto a mi tutor, Pablo, por toda su paciencia estos meses.

Resumen

El presente proyecto se enmarca en el ámbito de la producción musical en tiempo real. En concreto plantea el diseño, desarrollo e implementación de un pedal de efectos digital. Estos dispositivos permiten añadir en tiempo real efectos y modulaciones a ciertos instrumentos musicales, incluyendo la voz humana. Un octavador es un dispositivo electrónico que recibe un sonido y mediante un procesado de señal, entrega a su salida ese mismo sonido una octava por debajo del original y va a ser el efecto principal que se va a implementar. El proyecto plantea el problema desde cero, utilizando la literatura existente para abordar tanto el diseño como la implementación.

Abstract

This is the Abstract

Índice general

Capítulo 1

Introducción y generalidades

Un pedal de efectos es un dispositivo que se conecta entre un instrumento (normalmente electrófono) y su amplificador, modificando la señal de entrada y sus características fundamentales como pueden ser timbre, tono y volumen. En mi caso particular he optado por un pedal para saxofón, puesto que es el instrumento que yo toco, aunque resulta igual de válido para cualquier otro intrumento que se pueda captar con un micrófono convencional. No es habitual el uso de pedales de efectos en instrumentos de viento debido a que generalmente se busca un sonido limpio y fiel. No obstante, en multitud de producciones se pueden apreciar infinidad de efectos añadidos posteriormente ya sea digital o analógicamente. Algunos ejemplos son el *reverb, chorus u octavador*. Es este último el efecto que se implementará en este proyecto, con la salvedad de que funcionará en tiempo real.

Se ha decidido el combinar un efecto común con la posibilidad de implementarlo en el formato de pedal, que se ha hecho muy popular desde su aparición, dado que los intérpretes pueden activarlo con el pie y no tener que renunciar a tiempo de ejecución. En el caso del saxofón, este no suele ser el problema, ya que los músicos no precisan de estar tocando de forma continuada, si no que suele haber tiempos para descansar suficientemente largos como para operar el dispositivo que se este utilizando. A pesar de ello, se utiliza este formato por analogía con otros instrumentos.

Este proyecto abarca todo el proceso desde la idea inicial de diseño hasta el montaje del prototipo final, por tanto, las especificaciones de funcionamiento que se han utilizado pretenden facilitar un uso profesional del prototipo, de forma que sea compatible con los estándares establecidos tanto musical como ingenierilmente.

En primer lugar se utiliza un transductor para adquirir la señal, en este caso, un micrófono convencional. Una vez que el estímulo externo es transformado en pulsos eléctricos, atravesará un etapa de entrada analógica que pre-amplifica la señal y la adecua a la entrada de la FPGA.

Para el prototipo se ha utilizado la placa proporcionada por el departamento: Nexys A7. Esta placa monta una FPGA *Xilinx XC7A100T-1CSG324C* junto con varios switches, botones, leds y displays de 7 segmentos, que harán más fácil el manejo del prototipo. Esta placa tiene un micrófono integrado, pero es de tan baja calidad que se opta por diseñar la etapa de entrada, analógica completamente, y conectarla con uno de los puertos del *Pmod i2s2*, también de Digilent y proporcionado por el departamento. Este módulo contiene ADC, DAC y los conectores de mini-jack estándar en formato de audio, que servirán para

gestionar la señal de entrada y la de salida.

Se implementará un algoritmo que se encargará de llevar a cabo la octavación de la señal de entrada y de proporcionarla en la salida. Este algoritmo utiliza una aproximación de *Phase Vocoder* muy común en el tratamiento de señales de audio realizando una transformación al dominio de la frecuencia mediante FFT. Durante todo el proceso se priorizará el criterio de la baja latencia, dado que si no, resulta imposible la interpretación para el músico. El algoritmo se describirá en profundidad en su capítulo correspondiente.

1.1. Objetivos

Para la realización de este proyecto de forma satisfactoria se ha establecido la consecución de una serie de objetivos que son los siguientes:

- Diseño de un sistema completo, a partir de una problemática definida previamente mediante el estudio del instrumento y las señales.
- Implementación sobre la FPGA proporcionada.
- Construcción de un prototipo plenamente funcional que supone la consecución de todos los objetivos anteriores.
- Afianzar, debido a todo esto, conocimientos adquiridos durante el grado de diversos ámbitos como el procesamiento de señal y en concreto de la especialidad de Sistemas Electrónicos como la programación hardware, el montaje de un circuito analógico y la correcta comunicación entre todos los módulos.

1.2. Metodología

- En primer lugar se seleccionará el efecto que se quieren implementar, dando prioridad al octavador.
- Como segundo paso, se estudiará la literatura existente y se probarán distintas soluciones algorítmicas empleando MATLAB.
- Estudio y elección de la interfaces de entrada y salida. Selección de micrófono, ADC y DAC.
- Desarrollo y verificación en VHDL empleando la herramienta VIVADO de Xillinx.
- Montaje de un prototipo empleando la placa Nexys A7 de Digilent, el micrófono seleccionado en el punto 3 y los dispositivos necesarios.
- Test y depuración.

Capítulo 2

Sistema Completo

El sistema

Capítulo 3

El algoritmo

El propósito de un octavador es proporcionar una señal una octava inferior a la señal introducida en tiempo real. Para ello, debe haber un algoritmo que implemente una transformación al dominio de la frecuencia analizando cada muestra, posteriormente, se reducirá esa frecuencia entrante a la mitad para realizar la transformada inversa y obtener la señal octavada.

Durante todo el siglo XX se han ido desarrollando diferentes técnicas de tratamiento y codificación para la voz, conforme iba la tecnología en aumento. La consecuencia de ello es la aparición de diversos algoritmos que permiten este tipo de operaciones con una carga computacional relativamente baja. Para este proyecto, se realizará una aproximación de *"Vocoder de fase"*[1] que se podrá aplicar tanto a la voz como a cualquier instrumento, como se verá más adelante.

3.1. Vocoder

Un *Vocoder*[2] es generalemente cualquier aparato que analiza y/o sintetiza la voz humana para lograr algún objetivo concreto, como compresión de datos, multiplexación o encriptación en la mayoría de los casos.

El Vocoder de canal[3], desarrollado por los famosos *Bell Labs* en 1928, utilizaba varios filtros multibanda seguidos por detectores de envolvente cuyas señales de control se transmitían al decodificador del receptor. Estas señales de control son mucho más lentas que la señal original a transmitir, por lo que se puede reducir el ancho de banda permitiendo a un mismo medio de transmisión soportar un mayor número de canales, ya sea por radio o cable. Finalmente, el decodificador amplifica estas señales de control y las introduce en los filtros correspondientes a cada banda para poder sintetizar de nuevo la señal original. Además de las ventajas sobre el ancho de banda, también se ayuda a proteger la señal para que no se pueda interceptar. Encriptando las señales de control y modificando los parámetros de los filtros, se puede hacer muy difícil su correcta reinterpretación si no se sincronizan el codificador y el decodificador. Esto popularizó su uso durante la Segunda

[1] Del inglés "phase vocoder"
[2] Del inglés voice (voz) junto a encoder (codificador)
[3] Del inglés çhannel vocoder"

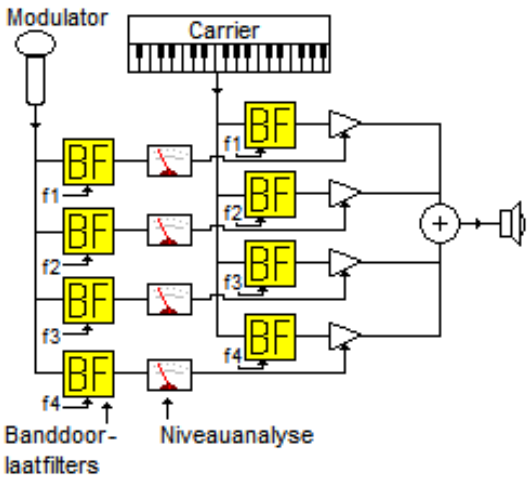

Figura 3.1: Esquema del funcionamiento de un vocoder musical

Guerra Mundial en el bando aliado patentándose diversos diseños.

El concepto se ha mantenido contante durante todo el siglo hasta nuestros días, donde podemos ver implementaciones modernas de la misma idea, por lo que se ha desarrollado una estandarización. La voz humana posee un rango de frecuencias de entre 200 y 3400 Hz típicamente, por lo que se optó por una frecuencia de muestreo de 8 kHz. Es común que se utilice una codificación con 16 bit por muestra por analogía con el estandar CD, pero con utilizar al menos 12 la mayoría de los receptores será capaz reproducir la señal con una fidelidad razonable. Citando un ejemplo, los codificadores según la norma ITU G.729, que son utilizados en telefonía comercial, tienen una buenísima calidad con una tasa binaria de 8 kbps. Actualmente también se utilizan para desarrollar tecnologías relacionadas con la lingüistica, la física y la neurociencia.

3.1.1. Vocoder en la música

Paralelamente a su utilización en comunicaciones, el vocoder se comenzó a popularizar durante la década de los 70 como método de síntesis. Cabe mencionar, que durante esta década, surge un gran interés en los músicos por experimentar con diferentes timbres y sonidos en instrumentos conocidos o experimentales. Para aplicaciones musicales, se utiliza una frecuencia portadora proveniente de un instrumento en lugar de extraer la frecuencia fundamental del sonido que se esta grabando. El resultado es una deformación del sonido capturado que, por estar afinado en una nota adecuada, produce un resultado agradable al oído. Fue el primer fabricante de sintetizadores y pionero de la música electrónica, Robert Moog, el que desarrollo un prototipo llamado *Farad* en 1968 pero no fue hasta 1970 cuando unieron el funcionamiento de esta máquina con el sintetizador modular *Moog* que había lanzado previamente al mercado. Quedaba ya conformada la esencia de utilizar la señal proveniente de un micrófono como moduladora y la proveniente de sintetizador como portadora para modularla. Algunos ejemplos tempranos de músicos reconocidos que utilizaron estos dispositivos fueron Phil Collins, Mike Oldfield, Stevie Wonder, Herbie Hancock o Michael Jackson.

Estos vocoder proporcionaban sonidos a los que el público estaba poco acostumbrado pero que realmente no mantenían una fidelidad tímbrica respecto el sonido que captaban. Por ello se empezaron a utilizar los vocoder de fase los cuales permiten llevar a cabo expansión o compresión en el tiempo y *Pitch Shifting*, es decir, modificar la altura musical del sonido o afinación sin cambiar la forma de onda que proporciona el timbre característico.

El método para hacerlo es el siguiente. En primer lugar se lleva a cabo una transformada mediante STFT (Short Time Fourier Transform) para posteriormente modificar la afinación mediante sub y sobremuestreo. Este proceso hace que el audio resultante no resulte reconocible, por lo que es necesario ajustar el valor de la fase de cada muestra para mantener la coherencia entre ellas, de ahí el nombre de vocoder de fase. Una vez calculadas las muestras, se transforman de vuelta al dominio del tiempo, donde se rellena con ceros para obtener la misma duración que la señal entrante. A continuación se explican en detalle estas etapas.

3.2. Transformación a frecuencia: STFT

Una STFT se usa para determinar el módulo y fase de muestras próximas de una señal mientras cambia con el tiempo, haciéndola muy adecuada para aplicaciones en tiempo real. Para ello, se divide la señal en segmentos más cortos de la misma longitud y se calcula la transformada de Fourier de cada uno de ellos por separado. El método para calcular la transformada es indiferente pero al priorizar una baja latencia conviene decantarse por el algoritmo de la Transformada Rápida de Fourier o FFT, no obstante, se explica con mayor detalle posteriormente.

La división de las muestras se realiza por dos razones: en primer lugar nos permite acotar el número de muestras para realizar la transformada propiamente dicha (limitado por el Hardware) y en segundo lugar favorece una reconstrucción suave de la señal si se lleva a cabo de una forma adecuada.

3.2.1. Solapamiento y enventanado

Dividir la señal entrante en sucesivas tramas es un proceso sencillo, únicamente se almacenan las muestras en una memoria para introducirlas posteriormente en el módulo que realiza la FFT. Sin embargo, el proceso de reconstrucción puede llegar a ser más complicado, especialmente si se modifica la señal entre medias. Es por ello que conviene estudiar este problema con detalle teniendo en cuenta las numerosas posibilidades que existen ya desarrolladas.

La idea más intuitiva en este punto es utilizar algún tipo de *solapamiento*, es decir, en lugar de empezar a construir una trama a continuación de la anterior, la empezamos a llenar antes de que se haya acabado de llenar la trama anterior, repitiendo muestras. De esta forma, las tramas están enlazadas entre ellas evitando una discontinuidad abrupt1.

Generalmente se cuantifica este proceso mediante un *factor de solapamiento, fs,* expresado en tanto por ciento. Si una trama t de longitud $n = 100$ muestras tiene un solapamiento de 15 %, las primeras $n * 15\,\% = 15$ muestras de t son idénticas a las 15 últimas de la trama anterior, t-1, y así sucesivamente.

Lógicamente, no se puede aumentar el factor de solapamiento infinitamente. A partir de un 50 % ya no resulta práctico debido a que para reconstruir la trama t se necesitan muestras de las tramas t-1 y t+1. En este caso, el beneficio de introducir este solapamiento es menor que el coste que hay que pagar: la disminución de la eficiencia del algoritmo. Idealmente realizamos el procesado sobre n muestras que luego reconstruimos pero, al introducir solapamiento, solo un porcentaje de ellas, m, formarán parte de la señal reconstruida. En consecuencia $m < n$ siempre, lo que quiere decir que procesamos más muestras de las que utilizaremos en la reconstrucción.

La consecuencia directa de ello es un desperdicio de recursos en el procesado. Por un lado, cuanto más disminuya esta eficiencia, más aumentará la latencia, ya que habrá que esperar al cálculo de la siguiente trama para poder finalizar la construcción de la trama presente. Por otro lado, resulta mucho más complejo de cara a la temporización en su implementación.

Como conclusión, debemos elegir un factor de solapamiento $0 < fs < 50$ para que resulte práctico. Tras un modelado en Matlab, he implementado finalmente un valor $fs = 25\,\%$ tal y como recomienda Ellis [Ell] en su implementación del vocoder de fase.

En general, estos procesos de enventanado (a excepción de la ventana rectangular) tienen la propiedad de que concentran sus transformadas de Fourier alrededor de $\omega = 0$ y además se pueden calcular fácilmente. En la figura 3.2.1 se establece una comparativa entre varias de estas ventanas.

La figura 3.2.1 compara las ventanas desde un punto de vista más analítico. Para este caso conviene prestar especial atención al valor de la anchura del lóbulo principal y la amplitud relativa de los lóbulos laterales. Esta última

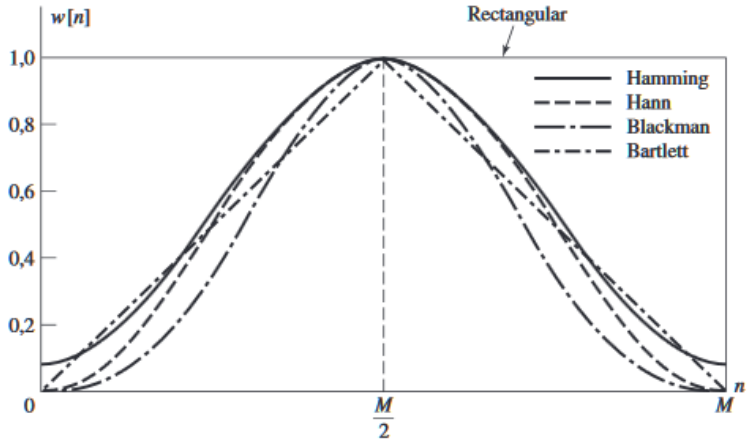

Figura 3.2: Comparativa de las ventanas más utilizadas

Tipo de ventana	Amplitud de pico de los lóbulos laterales (relativa) (dB)	Anchura aproximada del lóbulo principal	Pico del error de aproximación, $20\log_{10}\delta$ (dB)	Ventana de Kaiser equivalente, β	Anchura de la transición de de la ventana de Kaiser equivalente
Rectangular	−13	$4\pi/(M+1)$	−21	0	$1{,}81\pi/M$
Bartlett	−25	$8\pi/M$	−25	1.33	$2{,}37\pi/M$
Hann	−31	$8\pi/M$	−44	3.86	$5{,}01\pi/M$
Hamming	−41	$8\pi/M$	−53	4.86	$6{,}27\pi/M$
Blackman	−57	$12\pi/M$	−74	7.04	$9{,}19\pi/M$

Figura 3.3: Características de las ventanas más utilizadas

Capítulo 4

Implementación

La implementacion

Capítulo 5

Pruebas y depuración

Pruebas y tal y cual

Bibliografía

[Ell] D. Ellis, *A Phase Vocoder in Matlab*, LabROSA at Columbia University, Marzo 2003: http://www.ee.columbia.edu/d̃pwe/LabROSA/matlab/pvoc/