UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA

DESARROLLO DE INTERPOLADORES/DIEZMADORES FRACCIONALES PARALELIZADOS PARA SISTEMAS DE COMUNICACIONES POR FIBRAS ÓPTICAS

Autor: Jhon Jairo Cevallos Medina

Tutor: Mª Asunción Pérez Pascual

Cotutor: Mª José Canet Subiela

Trabajo Fin de Máster presentado en el Departamento de Ingeniería Electrónica de la Universitat Politécnica de València para la obtención del Título de Máster Universitario en Ingeniería de Sistemas Electrónicos

Curso 2018-2019

Valencia, enero de 2019

RESUMEN

En el presente documento se desarrolla interpoladores/diezmadores fraccionales para sistemas de comunicaciones por fibra óptica de 16 canales, utilizando técnicas de paralelización para alcanzar una velocidad de funcionamiento superior a los $312.5\ MHz$ por canal y así alcanzar una velocidad de transmisión de $5\ GHz$ para todo el sistema.

En este caso se implementan filtros polifásicos porque nos permiten aplicar técnicas de paralelización de una manera más fácil, se aplican tasas de cambio de $4/3$ y $8/7$ para en el caso de la interpolación, $3/4$ y $7/8$ para el caso del diezmado.

Las estructuras se diseñaron e implementaron en Simulink y se codificaron en lenguaje HDL, se obtuvo la cantidad de recursos empleados y se comprobó que todos los circuitos superaron la velocidad de funcionamiento requerida por el sistema.

RESUM

En aquest document es desenvolupen interpoladors/delmadors fraccionaris per a sistemes de comunicacions per fibra òptica de 16 canals, utilitzant tècniques de paral·lelització per arribar a velocitats de funcionament superiors a 312.5MHz per canal, i així aconseguir una velocitat de transmissió de 5 GHz per a tot el sistema.

En aquest cas s'implementen filtres polifàsics perquè ens permeten aplicar tècniques de paral·lelització d'una manera més fàcil. S'apliquen taxes de canvi de 4/3 i 8/7 en el cas de la interpolació, 3/4 i 7/8 en el cas del delmat.

Les estructures s'han dissenyat e implementat en Simulink i s'han codificat en llenguatge HDL, s'ha obtingut la quantitat de recursos necessaris i s'ha comprovat que tots els circuits superen la velocitat de funcionament requerida pel sistema.

ABSTRACT

The following document develops fractional interpolators / decimators for communication systems based on 16-channel fiber optic, using parallelization techniques to reach an operating speed higher than $312.5\ MHz$ per channel and achieve $5\ GHz$ of transmission speed for all the system.

In this case polyphase filters are implemented because they allow us to apply parallelization techniques in an easier way, exchange rates of $4/3$ and $8/7$ are applied for interpolation, $3/4$ and $7/8$ for decimation.

The hardware architectures were designed and implemented in Simulink and were encoded in HDL language, the amount of resources used was obtained and it was checked that all the circuits reach the operating speed required by the system.

DEDICATORIA

Dedico todo el esfuerzo y sacrifico empleado para la culminación de este trabajo a todas las personas que lo hicieron posible.

A Dios y la Virgen Dolorosa a quienes les debo todo porque siempre me cuidan y protegen.

A mis padres Loly y Fabian quienes con su sacrificio y ejemplo han guiado mis pasos.

A mis hermanos y confidentes Stefy y Anthony quienes con su granito de arena me empujaron a seguir adelante.

A mi sobrinito Emiliano, quien me inyecta esa chispa para poder alcanzar nuevas metas.

A mis abuelitos parte fundamental de mi formación y a quienes les debo muchas enseñanzas.

A mis tíos y toda mi familia que a pesar de que muchos ya no están físicamente presentes, los momentos que compartimos y sus enseñanzas marcaron mi camino

Cada uno de ellos forma parte importante de mi vida y es gracias a ellos que hoy puedo alcanzar un logro más en mi vida.

Jhon Jairo

AGRADECIMIENTO

Dios, tu amor y bondad no tienen fin, me permites sonreír ante todos mis logros que son resultado de tu ayuda. Porque cuando caigo o me pones a prueba, aprendo de mis errores y me doy cuenta de que los pones en frente mío para que mejore como ser humano y crezca de diversas maneras.

A la Universidad Politécnica de Valencia por darme la oportunidad de formarme en sus aulas.

A mis mentoras Asún Pérez y Mª José Canet, que, con sus conocimientos, experiencia, paciencia y motivación, me guiaron a culminar este trabajo con éxito.

Son muchas las personas que han formado parte de esta etapa de mi vida a las que me encantaría agradecerles por su amistad, consejos, apoyo, ánimo y compañía. Algunas se encuentran conmigo y otras permanecen en mis recuerdos y en mi corazón, sin importar en donde estén quiero darles las gracias por formar parte de mí, por todo lo que me han brindado y por todas sus bendiciones.

Jhon Jairo

DESARROLLO DE INTERPOLADORES/DIEZMADORES FRACCIONALES PARALELIZADOS PARA SISTEMAS DE COMUNICACIONES POR FIBRAS ÓPTICAS

CONTENIDO

RESUMEN i

RESUM i

ABSTRACT i

DEDICATORIA ii

AGRADECIMIENTO iii

CONTENIDO iv

ÍNDICE DE FIGURAS vi

ÍNDICE DE TABLAS ix

Capítulo 1. INTRODUCCIÓN 1

1.1 Sistemas de Comunicaciones Ópticos 1

1.2 Objetivos 2

1.2.1 Objetivos Generales 2

1.2.2 Objetivos Específicos 2

1.3 Metodología 2

1.4 Estructura de la Memoria 4

Capítulo 2. BACKGROUND DE PROCESADO DE SEÑAL PARA INTERPOLADORES Y DIEZMADORES FRACCIONALES 6

2.1 Filtros Digitales de Retardo Fraccional 6

2.1.1 Implementación de un filtro de retardo fraccional (FIR usando interpolación de Lagrange) 7

2.2 Cambio de tasa por un factor racional 8

2.2.1 Implementación con un Filtro polifásico 9

2.2.2 Implementación con un Filtro polifásico usando los polinomios de Lagrange 13

2.2.2.1 Optimización del Filtro Polifásico de Lagrange 18

2.2.3 Implementación con la estructura de Farrow 19

2.2.3.1 Optimización del Filtro de Farrow 21

2.3 Comparativa y Recursos Empleados 23

Capítulo 3. ARQUITECTURAS PARA LA IMPLEMENTACIÓN DE INTERPOLADORES /DIEZMADORES FRACCIONALES 28

3.1 Procesado Secuencial 28

3.2 Procesado Paralelo 29

3.2.1 Interpolación por 4/3 paralelizado por Px4 29

3.2.2 Diezmado por 3/4 paralelizado por Px4 30

3.3 Paralelización de Filtros 32
3.3.1 Interpolación por 4/3 paralelizado por Po12-P16 32
3.3.1.1 Verificación Interpolador por 4/3 35
3.3.2 Diezmado por 3/4 paralelizado por Po16-P12 37
3.3.2.1 Verificación diezmador por 3/4 39
3.3.3 Interpolación por 8/7 paralelizado por Po14-P16 41
3.3.3.1 Verificación Interpolador por 8/7 44
3.3.4 Diezmado por 7/8 paralelizado por Po16-P14 45
3.3.4.1 Verificación diezmador por 7/8 48
Capítulo 4. CONCLUSIONES Y LÍNEAS FUTURAS 51
4.1 Conclusiones 51
4.2 Líneas Futuras 52
Capítulo 5. BIBLIOGRAFÍA 53

ÍNDICE DE FIGURAS

Figura 1: Diagrama del Transmisor del Sistema 1
Figura 2: Diagrama del Receptor del Sistema 2
Figura 3: Flujograma del Proyecto 3
Figura 4: Proceso de Verificación por simulación 4
Figura 5: Señal de Entrada retardada por un factor fraccional δ 6
Figura 6: Interpolación de Lagrange para N=1, 2 y 3 8
Figura 7: Modelo Simulink Interpolación de Lagrange para N=3 8
Figura 8: Diagrama de Bloques-Cambio de Tasa por un Factor Racional 9
Figura 9: Estructura Filtro Polifásico para $\boldsymbol{Q = 4/3}$ 9
Figura 10: Diagrama de Bloques cambio de tasa para $\boldsymbol{Q = 4/3}$ 9
Figura 11: Análisis Espectros de señales cambio de tasa para $\boldsymbol{Q = 4/3}$ 10
Figura 12: Parámetros Diseño Filtro (*fdatool*) 11
Figura 13: Respuesta de Magnitud y Fase del Filtro 11
Figura 14: Respuesta al Impulso del Filtro 11
Figura 15: Modelo Simulink Filtro Polifásico 12
Figura 16: Señal Entrada y Salida Filter Designer- Polyphasic Filter 12
Figura 17: Interpolación cúbica por partes 14
Figura 18: Modelo Simulink Filtro Polifásico-Polinomio de Lagrange 16
Figura 19: Estructura Interna Subfiltro-Filtro Polifásico 16
Figura 20: Señal Entrada y Salida-Filtro Polifásico 17
Figura 21: Retardo Fraccionario Filtro Polifásico Interpolador (4 subfiltros) 17
Figura 22: Modelo Simulink Filtro Polifásico de Lagrange Simplificado 18
Figura 23: Cronograma de Tiempo Filtro Polifásico de Lagrange Simplificado 18
Figura 24: Señal Entrada y Salida Lagrange Simplificado 19
Figura 25: Estructura de Farrow- Polinomio de Lagrange 20
Figura 26: Calculo de $\Delta\boldsymbol{k}$ en Simulink 20
Figura 27: Modelo Simulink-Estructura de Farrow para $\boldsymbol{L = 4, M = 3, x0 = 0}$ 21
Figura 28: Señal de Entrada y Salida Estructura de Farrow para $\boldsymbol{L = 4, M = 3, x0 = 0}$ 21
Figura 29: Modelo Simulink-Estructura de Farrow Optimizada 22
Figura 30: Señal Entrada y Salida Farrow Optimizado para $\boldsymbol{L = 4, M = 3, x0 = 0}$ 22
Figura 31: Señales de Salida para un cambio de tasa de $\boldsymbol{3/4}$ 23
Figura 32: Modelo Simulink Filtro Polifásico para $\boldsymbol{3/4}$ 24
Figura 33: Parámetros Diseño Filtro (*fdatool*) para $\boldsymbol{3/4}$ 24
Figura 34: Diagrama de Bloques cambio de tasa para $\boldsymbol{3/4}$ 24

Figura 35: Análisis Espectros de señales cambio de tasa para $\boldsymbol{Q = 3/4}$ 25
Figura 36: Modelo Simulink Filtro Polifásico-Polinomio de Lagrange para $\boldsymbol{3/4}$ 25
Figura 37: Modelo Simulink Filtro Polifásico de Lagrange Simplificado para $\boldsymbol{3/4}$ 26
Figura 38: Generador de Respuesta al Impulso IR..... 28
Figura 39: Diseño de Filtro FIR Paralelo con 4 coeficientes..... 29
Figura 40: Reordenamiento de salidas para $\boldsymbol{4/3}$ paralelizado por $\boldsymbol{Px4}$..... 29
Figura 41: Diagrama de Bloques Filtro Paralelo $\boldsymbol{Px4}$ para $\boldsymbol{Q = 4/3}$ 29
Figura 42: Modelo Simulink Filtro Paralelo $\boldsymbol{Px4}$ para $\boldsymbol{Q = 4/3}$ 30
Figura 43: Señales de Entrada e Interpolada- Filtro Paralelizado $\boldsymbol{Px4}$ para $\boldsymbol{Q = 4/3}$ 30
Figura 44: Reordenamiento de salidas para $\boldsymbol{3/4}$ paralelo por $\boldsymbol{Px4}$ 30
Figura 45: Diagrama de Bloques Filtro Paralelo $\boldsymbol{Px4}$ para $\boldsymbol{Q = 3/4}$ 31
Figura 46: Modelo Simulink Filtro Paralelo $\boldsymbol{Px4}$ para $\boldsymbol{Q = 3/4}$ 31
Figura 47: Señales de Entrada y Diezmada- Filtro Paralelo $\boldsymbol{Px4}$ para $\boldsymbol{Q = 3/4}$..... 31
Figura 48: Modificaciones del Transmisor del Sistema 32
Figura 49: Diagrama Interpolación Fraccional por $\boldsymbol{4/3}$ –Paralelización $\boldsymbol{Po = 12}$ - $\boldsymbol{P = 16}$..... 33
Figura 50: Diagrama Banco de Filtros para $\boldsymbol{4/3}$ –Paralelización $\boldsymbol{Po = 12}$ - $\boldsymbol{P = 16}$ 33
Figura 51: Subfiltros–Paralelización $\boldsymbol{Po = 12}$ - $\boldsymbol{P = 16}$ para $\boldsymbol{4/3}$ 33
Figura 52: Modelo Simulink Subfiltros-Paralelización $\boldsymbol{Po = 12}$ - $\boldsymbol{P = 16}$ para $\boldsymbol{4/3}$..... 34
Figura 53: Señales de Entrada y Salida Filtro (*fdatool*)- $\boldsymbol{Po = 12}$ - $\boldsymbol{P = 16}$ para $\boldsymbol{4/3}$ 35
Figura 54: Diagrama de Verificación Banco de Filtros..... 35
Figura 55: TestBench Banco de Filtros Interpolador por $\boldsymbol{4/3}$-Paralelización $\boldsymbol{Po = 12}$-$\boldsymbol{P = 16}$36
Figura 56: Diagrama RTL Interpolador por $\boldsymbol{4/3}$-Paralelización $\boldsymbol{Po = 12}$ - $\boldsymbol{P = 16}$ 36
Figura 57: Frecuencia Máxima de Funcionamiento Interpolador por $\boldsymbol{4/3}$ 37
Figura 58: Diagrama Diezmador Fraccional por $\boldsymbol{3/4}$ –Paralelización $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 12}$..... 37
Figura 59: Diagrama Banco de Filtros para $\boldsymbol{3/4}$ –Paralelización $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 12}$ 37
Figura 60: Subfiltros–Paralelización $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 12}$ para $\boldsymbol{3/4}$ 38
Figura 61: Modelo Simulink Subfiltros-Paralelización $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 12}$ para $\boldsymbol{3/4}$..... 38
Figura 62: Señales de Entrada y Salida Filtro (*fdatool*)- $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 12}$ para $\boldsymbol{3/4}$ 39
Figura 63: Frecuencia Máxima de Funcionamiento Diezmador por $\boldsymbol{3/4}$..... 39
Figura 64: TestBench Banco de Filtros Diezmador por $\boldsymbol{3/4}$-Paralelización $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 12}$ 40
Figura 65; Diagrama RTL Diezmador por $\boldsymbol{3/4}$-Paralelización $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 12}$ 40
Figura 66: Diagrama Interpolación Fraccional por $\boldsymbol{8/7}$ –Paralelización $\boldsymbol{Po = 14}$ - $\boldsymbol{P = 16}$..... 41
Figura 67: Diagrama Banco de Filtros para $\boldsymbol{8/7}$ –Paralelización $\boldsymbol{Po = 14}$ - $\boldsymbol{P = 16}$ 41
Figura 68: Subfiltros–Paralelización $\boldsymbol{Po = 14}$ - $\boldsymbol{P = 16}$ para $\boldsymbol{8/7}$ 42
Figura 69: Modelo Simulink Subfiltros-Paralelización $\boldsymbol{Po = 14}$ - $\boldsymbol{P = 16}$ para $\boldsymbol{8/7}$..... 43
Figura 70: Señales de Entrada y Salida Filtro (*fdatool*)- $\boldsymbol{Po = 14}$ - $\boldsymbol{P = 16}$ para $\boldsymbol{8/7}$ 44

Figura 71: TestBench Banco de Filtros Interpolador por $\mathbf{8/7}$-Paralelización $\boldsymbol{Po = 14}$-$\boldsymbol{P = 16}$44
Figura 72: Frecuencia Máxima de Funcionamiento Interpolador por $\mathbf{8/7}$ 44
Figura 73; Diagrama RTL Diezmador por $\mathbf{8/7}$-Paralelización $\boldsymbol{Po = 14}$ - $\boldsymbol{P = 16}$ 45
Figura 74: Diagrama Diezmado Fraccional por $\mathbf{7/8}$ –Paralelización $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 14}$ 45
Figura 75: Diagrama Banco de Filtros para $\mathbf{7/8}$ –Paralelización $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 14}$ 46
Figura 76: Subfiltros–Paralelización $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 14}$ para $\mathbf{7/8}$ 47
Figura 77: Modelo Simulink Subfiltros-Paralelización $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 14}$ para $\mathbf{7/8}$..... 47
Figura 78: Señales de Entrada y Salida Filtro (*fdatool*)- $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 14}$ para $\mathbf{7/8}$ 48
Figura 79: Frecuencia Máxima de Funcionamiento Diezmador por $\mathbf{7/8}$..... 48
Figura 80: TestBench Banco de Filtros Diezmador por $\mathbf{7/8}$-Paralelización $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 14}$ 49
Figura 81: Diagrama RTL Diezmador por $\mathbf{7/8}$-Paralelización $\boldsymbol{Po = 16}$ - $\boldsymbol{P = 14}$ 49

ÍNDICE DE TABLAS

Tabla 1: Coeficientes de Lagrange-Retardo Fraccional 7
Tabla 2: Recursos empleados Interpolador de Lagrange 8
Tabla 3: Recursos empleados Filtro 12
Tabla 4: Coeficientes Filtro Polifásico de Lagrange para $\boldsymbol{L = 4}$ 15
Tabla 5: Recursos empleados Filtro Polifásico-Polinomio de Lagrange 17
Tabla 6: Recursos empleados Filtro Polifásico de Lagrange Simplificado 19
Tabla 7: Recursos empleados Filtro con Estructura de Farrow 21
Tabla 8: Recursos empleados Filtro con Estructura de Farrow Optimizada 23
Tabla 9: Coeficientes Filtro Polifásico de Lagrange para $\boldsymbol{L = 3}$ 26
Tabla 10: Cuadro comparativo de memorias empleados por cada metodología 27
Tabla 11: Cuadro comparativo de recursos empleados por cada metodología 27
Tabla 12: Coeficientes Filtro Interpolador (fdatool) para $\boldsymbol{4/3}$ 34
Tabla 13: Recursos empleados Interpolador por $\boldsymbol{4/3}$ 37
Tabla 14: Coeficientes Filtro Diezmador (fdatool) para $\boldsymbol{3/4}$ 39
Tabla 15: Recursos empleados Diezmador por $\boldsymbol{3/4}$ 41
Tabla 16: Coeficientes Filtro Interpolador (*fdatool*) para $\boldsymbol{8/7}$ 43
Tabla 17: Recursos empleados Interpolador por $\boldsymbol{8/7}$ 45
Tabla 18: Coeficientes Filtro Diezmador (*fdatool*) para $\boldsymbol{7/8}$ 48
Tabla 19: Recursos empleados Diezmador por $\boldsymbol{7/8}$ 50

Capítulo 1. INTRODUCCIÓN

1.1 Sistemas de Comunicaciones Ópticos

El elevado ancho de banda que requieren las aplicaciones de comunicaciones actuales hace que los sistemas de comunicaciones alámbricos e inalámbricos estén llegando a su máxima capacidad. Ante este escenario, las redes de comunicaciones que utilizan fibras ópticas ofrecen numerosas ventajas gracias a su bajo coste, alta fiabilidad y fácil mantenimiento, por lo que están siendo adoptadas en muchos sistemas de comunicaciones de alta velocidad. Afortunadamente, muchas de las técnicas de procesado digital que tradicionalmente se han utilizado en los sistemas de comunicaciones alámbricos e inalámbricos pueden adaptarse fácilmente a las necesidades de la transmisión por fibra óptica. En paralelo, el gran avance que se está produciendo en los dispositivos programables, en especial las FPGAs, hace que se puedan desarrollar sistemas específicos para la transmisión por fibra óptica que ofrecerán una gran flexibilidad. El gran reto de estos sistemas será alcanzar las elevadas frecuencias de transmisión necesarias para trabajar con las aplicaciones actuales.

El uso de FPGAs en el diseño de sistemas de comunicaciones ópticas multicanal promueve un incremento en la velocidad y estabilidad de los datos sobre redes a larga distancia, posibilitando la selección de diferentes anchos de banda o el uso de distintos protocolos de transmisión.

La transmisión de información a altas tasas de velocidad demanda que esta sea transmitida con las más altas eficiencias espectrales y la menor interferencia posible, para lograr este objetivo se han estado empleado diversas técnicas de modelado de pulsos como la multiplexación por división de frecuencia ortogonal ***OFDM [1-2],*** la multiplexación de Nyquist y otros esquemas de multiplexación, que aseguren que en el proceso de transmisión (multiplexación-demultiplexación) los espectros de la señal no se superpongan significativamente, lo que comprometería la calidad de la información.

Una de las ventajas que brindan los sistemas de comunicaciones ópticos es la flexibilidad que tienen para combinarse con sistemas eléctricos y así formar sistemas denominados mixtos, los cuales aprovechan las ventajas que cada tecnología les ofrece (velocidad, tipo de modulación, frecuencia de muestreo, forma de pulso) para poder construir sistemas eficientes, capaces de trabajar a altas velocidades y aplicar técnicas de procesado digital DSP para transmitir la información.

Para el desarrollo del presente trabajo, se dispone de un experimento implementando sobre una FPGA**.** La implementación hardware se ha paralelizado en 16 líneas, que trabajarán a una frecuencia de ***312,5 MHz***, con el objetivo de alcanzar ***5 GHz*** de frecuencia de transmisión en la fibra óptica. La ***Figura 1*** y la ***Figura 2*** muestran los esquemas del transmisor y receptor del sistema óptico implementado.

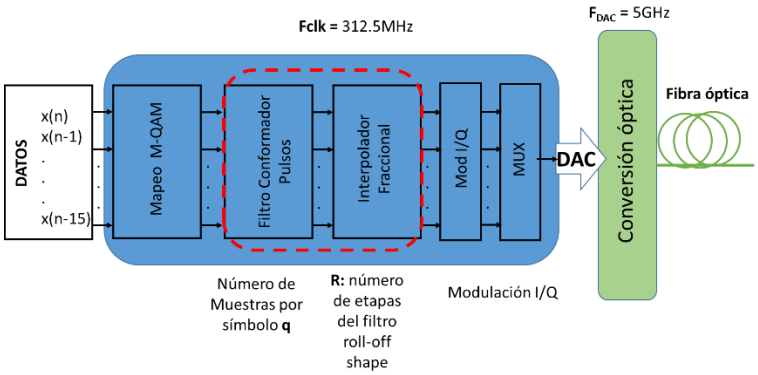


Figura 1: Diagrama del Transmisor del Sistema

Este sistema se implementa de forma que se puede variar fácilmente la modulación de la transmisión y el tipo de interpolación a realizar.

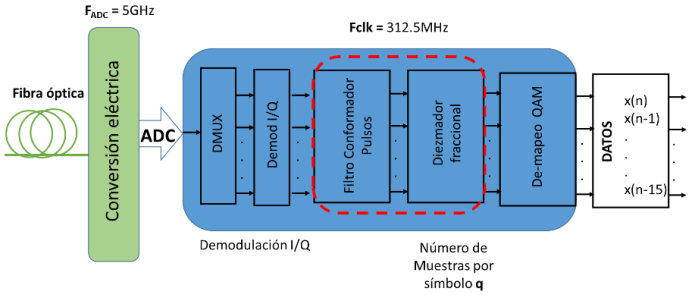


Figura 2: Diagrama del Receptor del Sistema

Al introducir un interpolador fraccional en el sistema se está reduciendo el espectro de la señal y generando bandas de guarda entre las imágenes, mientras mayor sea el factor de interpolación, mayor será la frecuencia que el sistema necesite para trabajar, por lo que el valor de cambio de tasa $\boldsymbol{q}$ debe estar entre $1 < \boldsymbol{q} < \mathbf{2}$. A lo largo del presente proyecto se trabajará sobre el sistema con diferentes valores de cambio de tasa, en concreto $\mathbf{4/3}$ y $\mathbf{8/7}$, debido a que están por debajo de $\mathbf{2}$, y que nos permiten trabajar con $\mathbf{16}$ canales, tal y como demostraremos en el desarrollo del presente trabajo.

El presente proyecto se centra en el desarrollo e implementación de filtros interpoladores/diezmadores fraccionales de muy alta velocidad, buscando el método más eficiente que se adapte a las características del sistema implementado y que emplee la menor cantidad de recursos, además que sea adecuado para trabajar en sistemas de comunicaciones mixtos.

1.2 Objetivos

1.2.1 Objetivos Generales

- Diseñar filtros interpoladores/diezmadores fraccionales aptos para la integración en un sistema de comunicaciones ópticas que alcance una velocidad de transmisión de 5GHz.
- Implementar los filtros anteriores sobre una FPGA para incluirlos en un sistema de transmisión mixto.

1.2.2 Objetivos Específicos

- Estudiar diferentes filtros interpoladores/diezmadores fraccionales con la finalidad de seleccionar la estructura más adecuada para incluirla en un sistema de comunicaciones mixto.
- Diseñar las estructuras anteriores en Simulink.
- Estudiar diferentes técnicas de paralelización de filtros que nos permitan alcanzar una elevada frecuencia de operación.
- Introducir en el sistema de comunicaciones mixto las estructuras diseñadas y seleccionar aquella que permita obtener las mejores prestaciones del sistema.
- Implementar sobre FPGA las estructuras de filtros paralelos diseñadas para distintos valores de interpolación/diezmado fraccional.

1.3 Metodología

La primera tarea a realizar consistió en un estudio teórico previo sobre las técnicas existentes de interpolación fraccional y cómo implementarlas.

El siguiente paso consistió en diseñar e implementar estos filtros en Simulink con la finalidad de comparar su funcionalidad y el número de recursos empleados.

Una vez seleccionado el filtro, se estudió la forma de implementarlo utilizando una arquitectura sistólica con máxima segmentación, para después paralelizar el filtro y así obtener las características de velocidad requeridas por el sistema.

Este proceso de síntesis e implementación se realizó con Quartus II de Altera para diferentes factores de interpolación, diezmado y número de líneas paralelas. De esta manera se realizaron simulaciones de cada caso con el programa Modelsim y se visualizaba su correcto funcionamiento. Con la ayuda de Matlab – Simulink se plasmó un modelo funcional para cada caso, y así se obtuvo un modelo de referencia comparable con el desarrollado en VERILOG.

Los datos relevantes de estas simulaciones fueron la frecuencia máxima de funcionamiento y el área ocupada, constatando así el cumplimiento de los requisitos de velocidad del sistema.

El diagrama de la ***Figura 3*** muestra una visión general del desarrollo seguido para la realización del presente proyecto.

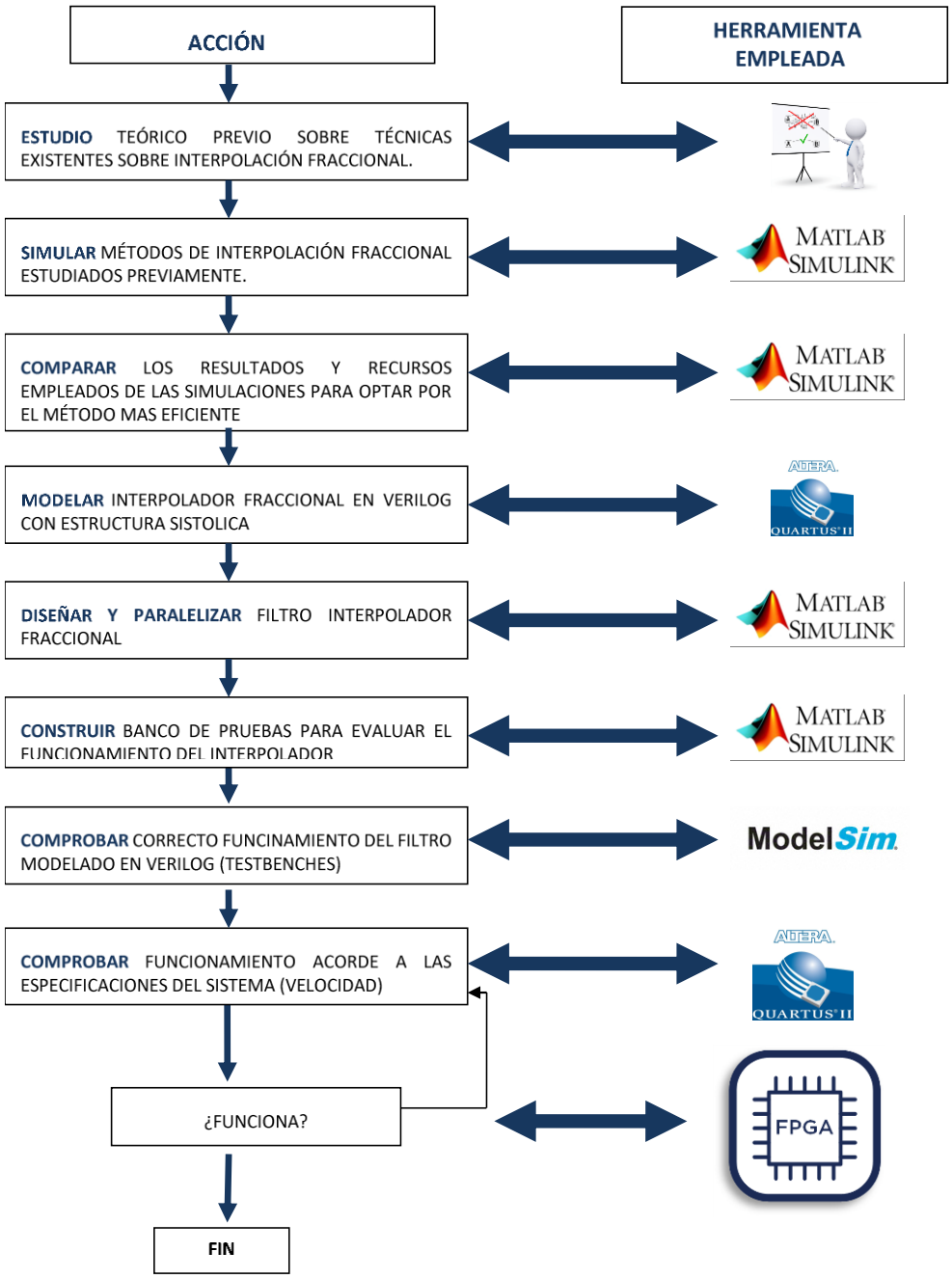


Figura 3: Flujograma del Proyecto

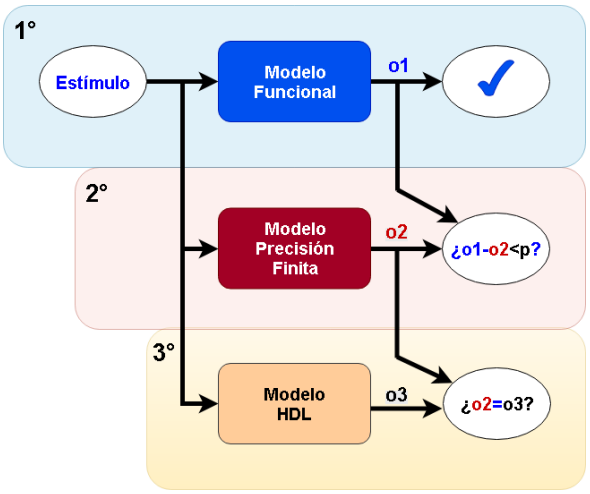


Figura 4: Proceso de Verificación por simulación

El proceso de verificación (***Figura 4)***, comprende tres aspectos fundamentales. El primero delimita el alcance de la verificación (*modelos funcionales*), el segundo describe la infraestructura necesaria (*modelos de precisión finita*) para implementar las pruebas requeridas en la verificación del sistema y el tercero comprende la comprobación de la respuesta, comparando la respuesta del dispositivo con la respuesta esperada para un estímulo en específico.

Una buena verificación del sistema es la parte más importante en el proceso de diseño ya que garantiza el funcionamiento y cumplimiento de las especificaciones de este. En el presente trabajo se desarrollaron los modelos funcionales y de precisión finita con la ayuda de Matlab y Simulink para posteriormente comparar estas respuestas con las obtenidas del modelo codificado en Verilog, este proceso se realizó con la ayuda de bancos de tests codificados en lenguaje HDL.

1.4 Estructura de la Memoria

La presente memoria se ha estructurado en cuatro capítulos descritos a continuación:

CAPÍTULO 1: INTRODUCCIÓN

Este capítulo pretende introducir al lector en los sistemas de comunicaciones ópticas de forma breve, se describen los objetivos generales y específicos del trabajo, se incluye la metodología seguida para desarrollar el proyecto y una breve descripción de la memoria.

CAPÍTULO 2: BACKGROUND DE PROCESADO DE SEÑAL PARA INTERPOLADORES Y DIEZMADORES FRACCIONALES

El capítulo dos introduce los conceptos básicos de filtros interpoladores/diezmadores fraccionales, incluyendo la explicación de los diferentes métodos y una descripción de la propuesta de los filtros a implementar en el proyecto.

CAPÍTULO 3: ARQUITECTURAS PARA LA IMPLEMENTACIÓN DE INTERPOLADORES /DIEZMADORES FRACCIONALES

En este capítulo se explican y muestran las diferentes arquitecturas disponibles para la implementación de interpoladores/diezmadores y las técnicas de paralelización existentes, así como las razones por las cuales se escogió la arquitectura adecuada para la aplicación sometida a estudio en este trabajo.

CAPÍTULO 4: CONCLUSIONES Y LÍNEAS FUTURAS

Finalmente, en el capítulo cuatro se muestran las conclusiones en referencia a las características principales de los filtros desarrollados, la síntesis del filtro sobre la FPGA, así como un pequeño apunte personal sobre lo aprendido durante todo el proceso, y líneas futuras de investigación.

Capítulo 2. BACKGROUND DE PROCESADO DE SEÑAL PARA INTERPOLADORES Y DIEZMADORES FRACCIONALES

En este capítulo se introducen brevemente los conceptos básicos para comprender los filtros digitales de retardo fraccional y las estructuras existentes para realizar un cambio de tasa. A lo largo de este capítulo se implementarán las diferentes estructuras estudiadas para un factor de $\boldsymbol{4/3}$ que es uno de los valores que se trabajará en el presente trabajo, posteriormente se realiza una comparativa en función de la cantidad de recursos que requiere cada una y se escoge la más adecuada para implementarla.

2.1 Filtros Digitales de Retardo Fraccional

Los Filtros Digitales de retardo Fraccional nos permiten retrasar una señal de entrada de un sistema por una fracción del periodo de muestreo. Para conseguir un retardo fraccional estos filtros deben implementar una interpolación fraccional en primer lugar, quedándose en última instancia con la muestra interpolada que corresponderá al retardo buscado. Dado que nuestro objetivo en este trabajo es lograr una interpolación fraccional, comenzaremos por estudiar la técnica que se utiliza en este tipo de filtros, los cuales son empleados en diversas aplicaciones del Procesamiento Digital de Señales (DSP) como: codificación y síntesis de voz, conversión de frecuencia de muestreo arbitraria, matrices de antenas eficientes, formación de haz digital, efectos Doppler en realidad virtual, adaptación de fase, sincronización, modelado musical de instrumentos, entre otros.

La elección de la técnica adecuada para la implementación de este tipo de filtros es un desafío para los diseñadores, el documento desarrollado por Laakso, Valimaki, Karjalainen y Laine (Splitting the Unit Delay) ***[4]*** expone una revisión exhaustiva de las distintas herramientas disponibles para el diseño de filtros de retardo fraccional. Este documento es el más citado sobre el diseño y aplicación de Filtros Digitales de Retardo Fraccional.

En tiempo discreto un filtro digital de retardo se expresa como:

$$\boldsymbol{y[n]} = x[n - D] \qquad \boldsymbol{Ecu.\,1}$$

La señal de entrada $\boldsymbol{x[n]}$ es retardada por $\boldsymbol{D}$ muestras, y $\boldsymbol{D}$ es un entero positivo. En el caso de un filtro digital de retardo fraccional, la salida está dada por:

$$\boldsymbol{y[n]} = x[n - \delta] \qquad \boldsymbol{Ecu.\,2}$$

Donde $\boldsymbol{\delta}$ no es necesariamente un número entero. El objetivo del filtro digital de retardo fraccional es interpolar el valor de la señal de entrada entre dos puntos de muestreo como se ilustra en la ***Figura 5*** (los círculos marcan el muestreo inicial y las cruces el muestreo con el retardo fraccional).

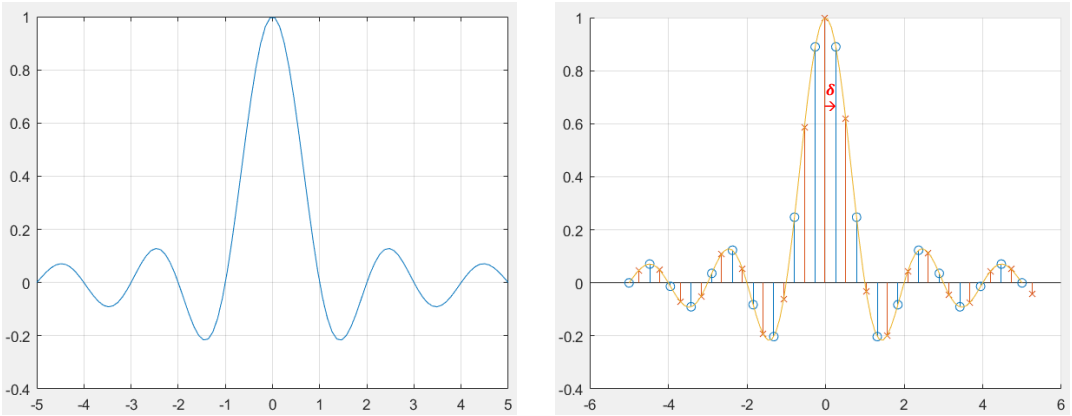


Figura 5: Señal de Entrada retardada por un factor fraccional **δ**

Tanto los Filtros de Respuesta de Impulso Finito ***FIR*** que son estables y presentan fase lineal, como los Filtros de Respuesta de Impulso Infinito ***IIR*** que tienden a tener órdenes de filtros menores pero cuyo diseño puede llevar a obtener filtros inestables son comúnmente empleados para implementar filtros digitales ***[5]***.

Un filtro digital de retardo fraccional ideal tiene fase lineal y respuesta de amplitud plana en la banda de paso, su respuesta al impulso viene dada por una función *SINC* desplazada en el tiempo, es decir es no causal y tiene una duración infinita. El desafío entonces es diseñar un filtro FIR para aproximar la respuesta al impulso ideal tan estrechamente como sea posible, y que el orden del filtro no sea excesivamente grande.

En tiempo discreto no es posible implementar de forma directa un retardo de valor racional (no entero), esto se debe a que los sistemas en tiempo discreto son sistemas muestreados que solo perciben valores de entrada en momentos determinados. Si al sistema se lo retarda por un valor racional se debería obtener el valor se la señal entre dos muestras, lo que es imposible porque el sistema no cuenta con información entre los valores de las muestras que recoge.

Una táctica empleada es separar el valor del retardo $\boldsymbol{D}$ en su parte entera $\boldsymbol{N}$ y su parte decimal $\boldsymbol{\alpha}$. Utilizando aproximaciones se puede aproximar el retardo teórico de valor $\boldsymbol{\alpha}$ y se puede emplear un retardo variable para el coeficiente $\boldsymbol{N}$, esto se describe matemáticamente en la ecuación $\boldsymbol{Ecu.3}$.

$$z^{-D} = z^{-N} * z^{-\alpha} \qquad \textbf{Ecu. 3}$$

En los siguientes apartados se analizarán algunos métodos para implementar un retardo fraccional y se compararán resultados obtenidos.

2.1.1 Implementación de un filtro de retardo fraccional (FIR usando interpolación de Lagrange)

Este método interpola el valor del retardo entre dos muestras diferentes utilizando un Filtro ***FIR***. El cálculo de los coeficientes se realiza empleando la Interpolación Polinómica de Lagrange, que probablemente es la manera más fácil de diseñar un filtro ***FIR***.

Los coeficientes se obtienen ajustando el polinomio para pasar a través de un conjunto dado de datos. La solución puede darse en una forma explícita como:

$$\boldsymbol{h(n)} = \prod_{\substack{k=0 \\ k\neq n}}^{N} \frac{D-k}{n-k} \; Para\; n = 0,1,2,\dots N \qquad \textbf{Ecu. 4}$$

Donde $\boldsymbol{N}$ es el orden del filtro y $\boldsymbol{D}$ es el valor del retardo. Los coeficientes de Lagrange para distintos órdenes del filtro se muestran en la siguiente tabla:

Tabla 1: Coeficientes de Lagrange-Retardo Fraccional

	$\mathbf{h(0)}$	$\mathbf{h(1)}$	$\mathbf{h(2)}$	$\mathbf{h(3)}$
$\mathbf{N=1}$	$1-\mathrm{D}$	D		
$\boldsymbol{N=2}$	$\frac{(D-1)(D-2)}{2}$	$-D(D-2)$	$\frac{D(D-1)}{2}$	
$\mathbf{N=3}$	$\frac{-(\mathrm{D}-1)(\mathrm{D}-2)(\mathrm{D}-3)}{6}$	$\frac{\mathrm{D}(\mathrm{D}-2)(\mathrm{D}-3)}{2}$	$\frac{-\mathrm{D}(\mathrm{D}-1)(\mathrm{D}-3)}{2}$	$\frac{\mathrm{D}(\mathrm{D}-1)(\mathrm{D}-2)}{6}$

La interpolación de Lagrange presenta algunas ventajas ***[4]***:

- Fórmulas explícitas fáciles para calcular los coeficientes.
- Buena respuesta a bajas frecuencias, y una respuesta de magnitud suave.

La ***Figura 6*** muestra un retardo $\boldsymbol{D} = 4/3$ para órdenes del filtro $\mathbf{1}, \mathbf{2}\ y\ \mathbf{3}$, mientras mayor es el orden del filtro, mayor es la cantidad de puntos que se toman para realizar la interpolación.

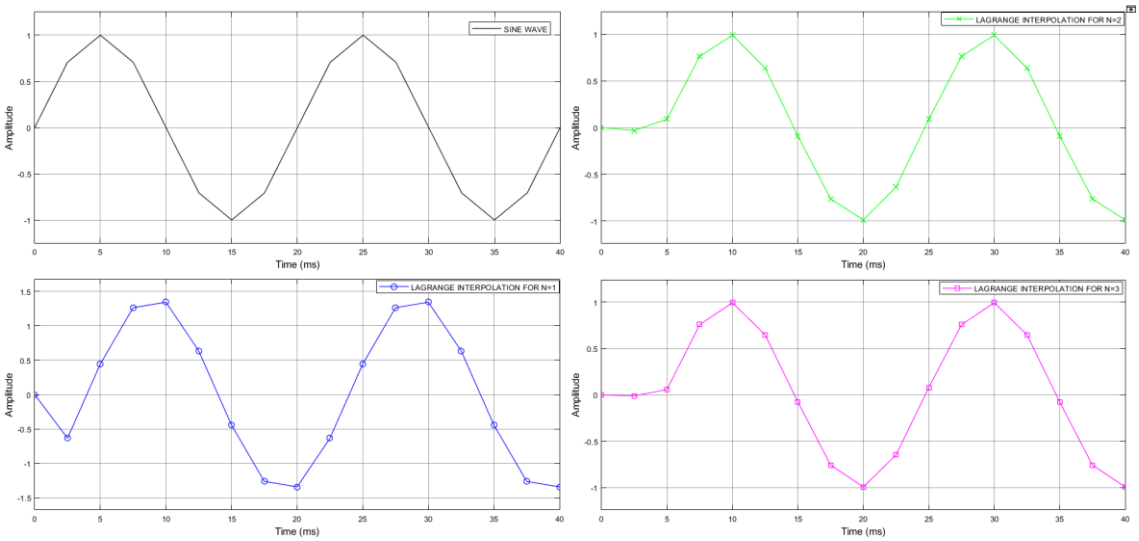


Figura 6: Interpolación de Lagrange para N=1, 2 y 3

En la ***Figura 7*** se muestra el modelo Simulink del Interpolador de Lagrange de orden ***N=3***, cuyos coeficientes empleados se muestran en la ***Tabla 1.***

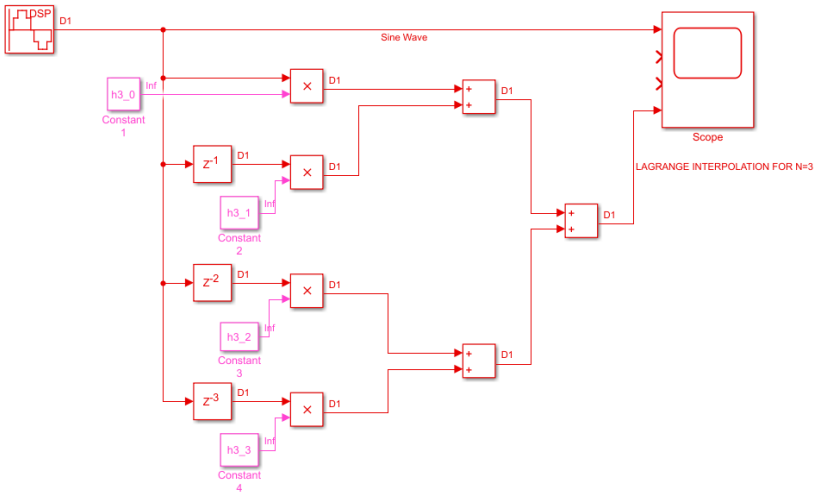


Figura 7: Modelo Simulink Interpolación de Lagrange para N=3

La cantidad de recursos empleados en el modelo Simulink expuesto en la figura anterior se resumen en la ***Tabla 2.***

Tabla 2: Recursos empleados Interpolador de Lagrange

Recurso	Cantidad
Registros	3
Sumadores	3
Multiplicadores	4

2.2 Cambio de tasa por un factor racional

Si el cociente entre la frecuencia de muestreo deseada y la original es un factor racional, se puede emplear un diezmador de orden ***M*** y un interpolador de orden ***L***, expresando el cociente entre las frecuencias como una fracción irreducible (numerador y denominador primos entre sí)

$\boldsymbol{L/M}$. Primero se interpola para evitar la pérdida de información y posteriormente se diezma. Se añade un filtro intermedio que elimina las copias espectrales que aparecen tras la interpolación y limita el ancho de banda para evitar que se produzca aliasing tras el diezmado.

La ***Figura 8*** ilustra el diagrama de bloques para obtener una conversión racional de la frecuencia de muestreo.

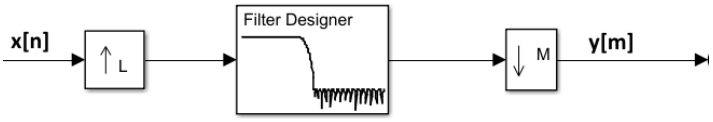


Figura 8: Diagrama de Bloques-Cambio de Tasa por un Factor Racional

En los siguientes apartados se explicarán algunos métodos para implementar un cambio de tasa por un factor racional como: Filtros Polifásicos, Filtros Polifásicos empleando polinomios de Lagrange y la estructura de Farrow, además de algunas optimizaciones con el fin de reducir el uso de recursos.

2.2.1 Implementación con un Filtro polifásico

La implementación de filtros polifásicos busca la reducción del número de operaciones empleadas en el filtro (recursos). No es eficiente en la interpolación filtrar la señal con una muestra de cada $\boldsymbol{L}$ diferente de cero o en el diezmado filtrar la señal después descartar una de cada $\boldsymbol{M}$ muestras, por eso se busca dividir la interpolación/diezmado en varias etapas.

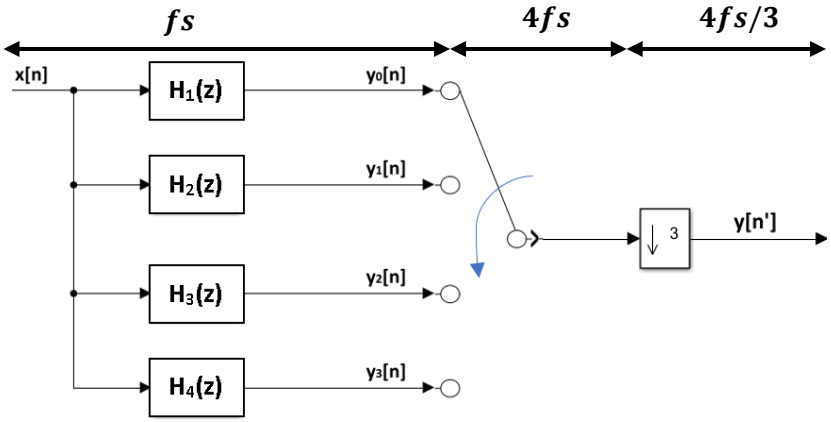


Figura 9: Estructura Filtro Polifásico para $\boldsymbol{Q = 4/3}$

Este tipo de estructuras permiten que los filtros trabajen a una frecuencia $\boldsymbol{fs}$ menor que la frecuencia de salida $4\boldsymbol{fs}$, por lo que las restricciones a la hora de implementar los filtros son menores. Para el cálculo de los coeficientes del filtro necesario para eliminar las copias de la señal debidas a la interpolación y para evitar el aliasing en el diezmado, es necesario establecer los parámetros que debe cumplir este filtro.

La ***Figura 10*** muestra el diagrama de bloques con las frecuencias en cada punto del sistema para un cambio de tasa de $\boldsymbol{Q = 4/3}$, y en la ***Figura 11*** se muestra el análisis de espectros de todas las señales involucradas que nos permite analizar y establecer los parámetros de nuestro filtro para una señal de entrada con un ancho de banda de $\boldsymbol{58\ MHz}$.

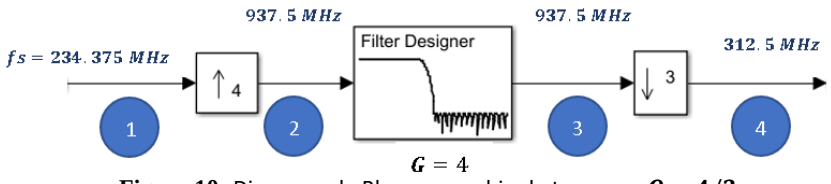


Figura 10: Diagrama de Bloques cambio de tasa para $\boldsymbol{Q = 4/3}$

En el punto ***1*** la señal se muestrea a una frecuencia $\boldsymbol{fs}$**,** en el punto ***2*** la señal se interpola lo que produce **4** copias espectrales por cada ciclo de la señal, en el punto ***3*** la señal pasa por el filtro que elimina estas copias espectrales y limita el ancho de banda para evitar que se produzca aliasing tras el diezmado y en el punto ***4*** se diezma la señal, expandiendo su espectro por un factor **3.**

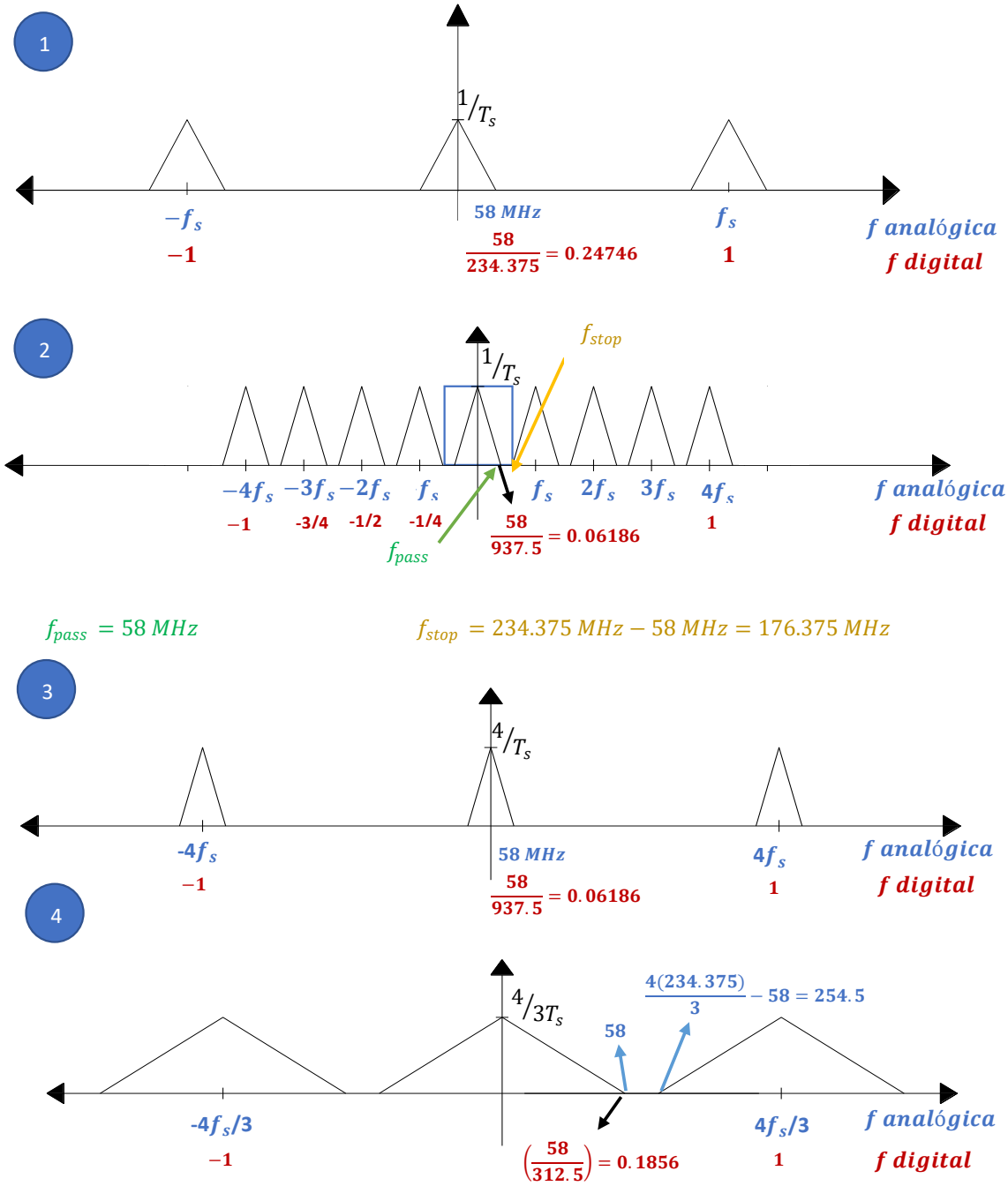


Figura 11: Análisis Espectros de señales cambio de tasa para $\boldsymbol{Q = 4/3}$

Una vez establecidos los parámetros $\boldsymbol{f_{pass}}$(*rango de frecuencias que el filtro dejar pasar*) y $\boldsymbol{f_{stop}}$(*frecuencia a partir de la cual la señal se salida se atenúa significativamente*)**,** también se establece como parámetros de diseño $\boldsymbol{70\ dB}$ de atenuación en la banda eliminada y un rizado de $\boldsymbol{0.1\ dB}$**.** Con la ayuda de ***fdatool*** se procede a la estimación del orden y los coeficientes del filtro aplicando el algoritmo de Parks-McClellan (*FIR: Equiripple*).

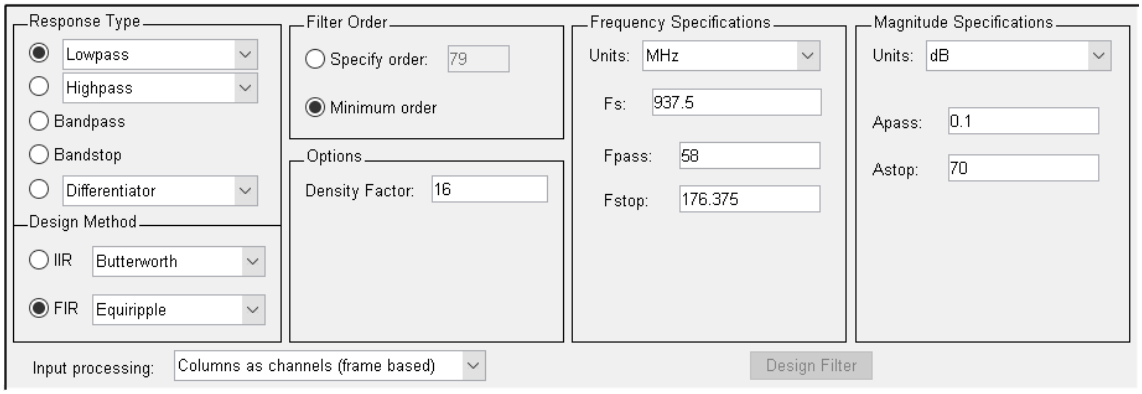


Figura 12: Parámetros Diseño Filtro (*fdatool*)

La ***Figura 13*** muestra el diagrama de magnitud y fase. La respuesta en frecuencia del filtro es plana en la banda de paso, la atenuación en la banda atenuada se encuentra por debajo de $\mathbf{60}\ \boldsymbol{dB}$, es decir que deja pasar señales por debajo de $\mathbf{58}\ \boldsymbol{MHz}$ sin atenuarlas. También el filtro presenta fase lineal en la banda de paso.

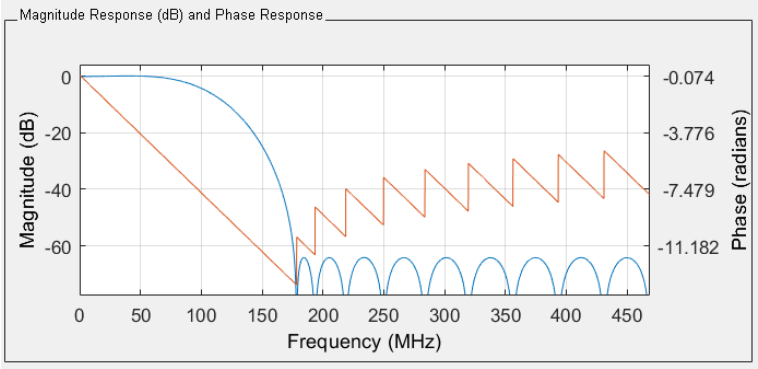


Figura 13: Respuesta de Magnitud y Fase del Filtro

La ***Figura 14*** muestra la respuesta al impulso del filtro diseñado con $\mathbf{24}$ coeficientes.

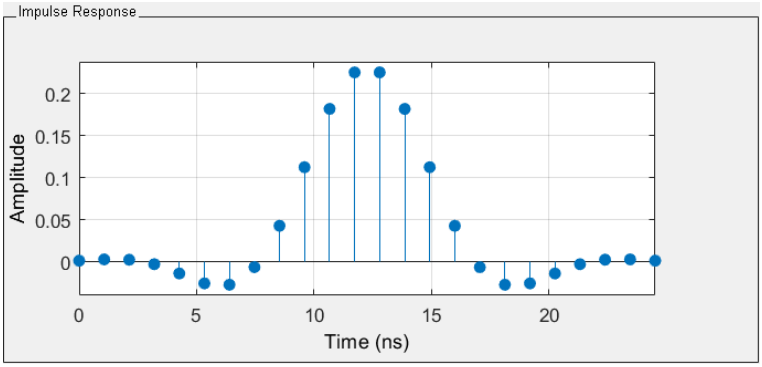


Figura 14: Respuesta al Impulso del Filtro

Para realizar un cambio de tasa por un factor racional utilizando una estructura polifásica para $\boldsymbol{Q} = 4/3$, se reparten los coeficientes en $\mathbf{4}$ subfiltros correspondientes al factor de interpolación, los coeficientes calculados con fdatool ($\mathbf{24}$ coeficientes) se reparten entre los subfiltros con la ayuda de la función **Coef_polif** de Matlab y al final se aplica el diezmador (factor de diezmado).

Función Matlab Coef_polif

```
function [ FB] = Coef_polif (Num, D)
% Coef_polif Gets the coefficients for a polyphase implementation
% Num is the full filter (should be multiple of D)
```

```
%  D is the interpolation or decimation factor
%  FB is the Polyphase Filter Bank. Each row is a filter
D=4;
L = length (Num)/D;
FB = zeros (D, L);
for i=1: L
   for j=1:D
      FB (j, i) =Num((i-1) *D+j);
   end
end
```

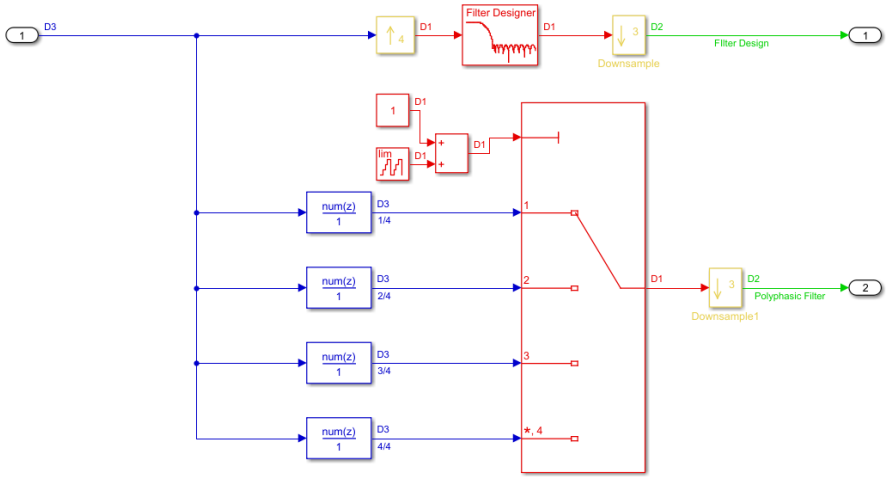


Figura 15: Modelo Simulink Filtro Polifásico

En el modelo de Simulink ***Figura 15***, se compara el filtro diseñado con ***fdatool*** y el filtro con estructura polifásica, obteniéndose el mismo resultado para ambos casos ***Figura 16***.

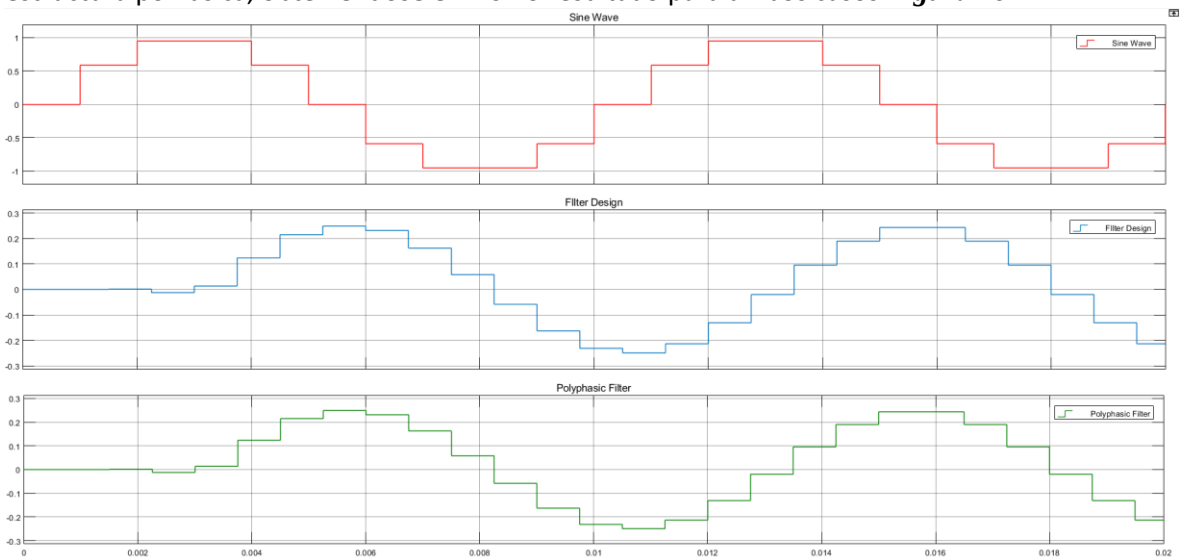


Figura 16: Señal Entrada y Salida Filter Designer- Polyphasic Filter

Los recursos empleados en el filtro diseñado (*Filter Designer*) constan en la ***Tabla 3.***

Tabla 3: Recursos empleados Filtro

Recurso	Cantidad
Registros	23
Sumadores	23
Multiplicadores	24

2.2.2 *Implementación con un Filtro polifásico usando los polinomios de Lagrange*

Como se mencionó anteriormente el uso de los polinomios de Lagrange es muy habitual a la hora de buscar alternativas de implementación de filtros digitales debido a su respuesta frecuencial máximamente plana y su capacidad de aproximar el filtro utilizando filtros con pocos coeficientes.

Un polinomio de Interpolación de Lagrange de grado $\boldsymbol{N-1}$ es aquel que pasa a través de $\boldsymbol{N}$ puntos y se denota de la siguiente manera:

$$\boldsymbol{s(k)} = \sum_{n=0}^{N-1} a_n * k^n \qquad \textbf{Ecu. 5}$$

Donde $\boldsymbol{a_n}$ corresponde a los coeficientes del polinomio $\boldsymbol{s(k)}$, que se calculan para los valores discretos $\boldsymbol{s(n)}$ para $\boldsymbol{n = 0, 1, 2, \ldots}$. Para calcular los coeficientes, se crea un sistema de ecuaciones lineales.

$$\begin{cases} a_0 + a_1.0 + a_2.0^2 + \cdots + a_{N-1}.0^{N-1} = s(0) \\ a_0 + a_1.1 + a_2.1^2 + \cdots + a_{N-1}.1^{N-1} = s(1) \\ a_0 + a_1.2 + a_2.2^2 + \cdots + a_{N-1}.2^{N-1} = s(2) \\ \cdots \\ a_0 + a_1.(N-1) + a_2.(N-1)^2 + \cdots + a_{N-1}.(N-1)^{N-1} = s(N-1) \end{cases}$$

El sistema se puede expresar como:

$$M.a = s \qquad \textbf{Ecu. 6}$$

Siendo $\boldsymbol{M}$ la matriz de coeficientes del sistema de ecuaciones lineales.

$$M = \begin{pmatrix} 1 & 0 & 0^2 & \cdots & 0^{N-1} \\ 1 & 1 & 1^2 & \cdots & 1^{N-1} \\ 1 & 2 & 2^2 & \cdots & 2^{N-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & N-1 & (N-1)^2 & \cdots & (N-1)^{N-1} \end{pmatrix}$$

$$\boldsymbol{a} = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{N-1} \end{bmatrix} \quad \boldsymbol{s} = \begin{bmatrix} s(0) \\ s(1) \\ s(2) \\ \vdots \\ s(N-1) \end{bmatrix}$$

La solución del sistema de ecuaciones se puede expresar como:

$$a = M^{-1}.s \qquad \boldsymbol{Ecu. 7}$$

El cálculo de la matriz inversa $\boldsymbol{M^{-1}}$ es un proceso computacional costoso, lo que plantea el desafío de elegir el número adecuado de muestras de la señal de entrada para aplicar la interpolación (orden del polinomio).

La interpolación cúbica por partes es el método más usado debido a la precisión que ofrece con respecto a su bajo coste computacional. Como se mencionó anteriormente, un polinomio de grado $\boldsymbol{N-1}$ es aquel que pasa a través de $\boldsymbol{N}$ puntos, en este caso son necesarios cuatro puntos para poder construir un polinomio de tercer grado (*cúbico*). El valor de $\boldsymbol{y(k)}$ se puede calcular con la ayuda de las cuatro muestras de entrada $\boldsymbol{s(n), s(n-1)}$, $\boldsymbol{s(n-2), s(n-3)}$

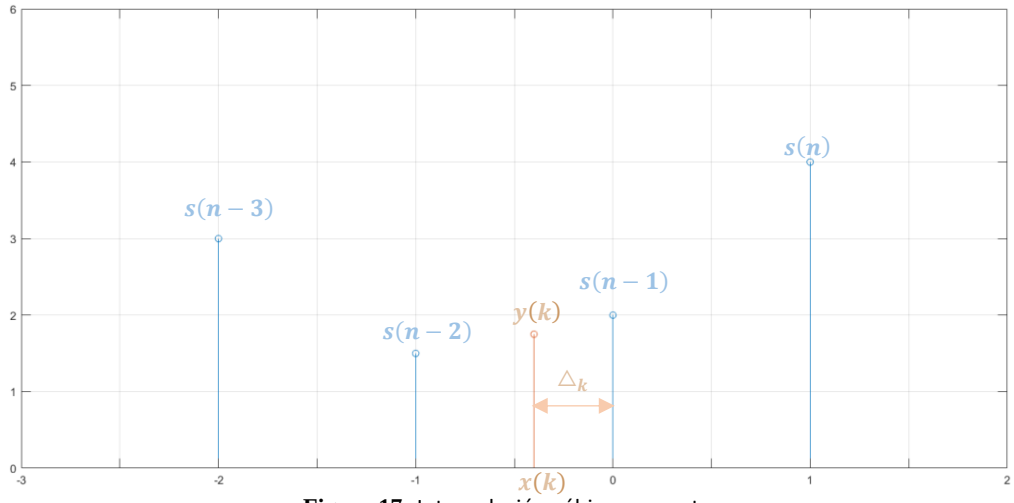


Figura 17: Interpolación cúbica por partes

Si observamos la matriz $\boldsymbol{M}$ está compuesta por los coeficientes de las muestras de la señal de entrada, así que para evitar el cálculo de la matriz $\boldsymbol{M}$ y su inversa para cada valor de $\boldsymbol{k}$ de la señal de salida $\boldsymbol{y(k)}$, en lugar de utilizar $\boldsymbol{x(k)}$ **(***punto de la muestra de salida* $\boldsymbol{k}$ *después del remuestreo***)** se puede usar el valor de $\boldsymbol{\triangle_k}$ **(***valor retraso con respecto a la muestra* $s(n-1)$ **)**. Esto nos permitiría realizar el cálculo de $\boldsymbol{M^{-1}}$ y usarla para cualquier valor de $\boldsymbol{x(k)}$.

Si tomamos los coeficientes de un polinomio con índices fijos ($s(n-3)=-2, s(n-2)=-1, s(n-1)=0\,, s(n)=1$) ***Figura 17,*** podemos armar el siguiente sistema de ecuaciones:

$$\begin{cases} a_0 + a_1.(1) + a_2.1^2 + a_2.1^3 = s(n) \\ a_0 + a_1.0 + a_2.0^2 + a_2.0^3 = s(n-1) \\ a_0 + a_1.(-1) + a_2.(-1)^2 + a_2.(-1)^3 = s(n-2) \\ a_0 + a_1.(-2) + a_2.(-2)^2 + a_2.(-2)^3 = s(n-3) \end{cases}$$

La matriz de coeficientes $\boldsymbol{M}$ y su inversa quedarían de la siguiente manera:

$$\boldsymbol{M} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 \\ 1 & -2 & 4 & -8 \end{bmatrix} \qquad \boldsymbol{M^{-1}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & -1 & \frac{1}{6} \\ \frac{1}{2} & -1 & \frac{1}{2} & 0 \\ \frac{1}{6} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{6} \end{bmatrix}$$

Para poder encontrar la solución del sistema de ecuaciones se aplica la $\boldsymbol{Ecu.7}$**:**

$$\boldsymbol{a} = M^{-1}.s = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & -1 & \frac{1}{6} \\ \frac{1}{2} & -1 & \frac{1}{2} & 0 \\ \frac{1}{6} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{6} \end{bmatrix} . \begin{bmatrix} s(n) \\ s(n-1) \\ s(n-2) \\ s(n-3) \end{bmatrix}$$

La expresión para los coeficientes del polinomio quedaría de la siguiente manera:

$$\begin{cases} a_0 = s(n-1) \\ a_1 = \frac{1}{3}s(n) + \frac{1}{2}s(n-1) - s(n-2) + \frac{1}{6}s(n-3) \\ a_2 = \frac{1}{2}s(n) - s(n-1) + \frac{1}{2}s(n-2) \\ a_3 = \frac{1}{6}s(n) - \frac{1}{2}s(n-1) + \frac{1}{2}s(n-2) - \frac{1}{6}s(n-3) \end{cases} \qquad \boldsymbol{Ecu.8}$$

La ecuación de salida del polinomio cubico será:

$$\boldsymbol{y(k)} = a_0 + a_1(-\triangle_k) + a_2(-\triangle_k)^2 + a_3(-\triangle_k)^3 \qquad \boldsymbol{Ecu.\,9}$$

Reemplazando en $\boldsymbol{y(k)}$ los coeficientes:

$$\begin{aligned}\boldsymbol{y(k)} = s(n-1) &+ \left[\frac{1}{3}s(n) + \frac{1}{2}s(n-1) - s(n-2) + \frac{1}{2}s(n-3)\right](-\triangle_k) \\ &+ \left[\frac{1}{2}s(n) - s(n-1) + \frac{1}{2}s(n-2)\right](-\triangle_k)^2 \\ &+ \left[\frac{1}{6}s(n) - \frac{1}{2}s(n-1) + \frac{1}{2}s(n-2) - \frac{1}{6}s(n-3)\right](-\triangle_k)^3\end{aligned}$$

$$\begin{aligned}\boldsymbol{y(k)} = s(n-1) &+ \left[\frac{-\triangle_k}{6}s(n-3) + \triangle_k * s(n-2) - \frac{\triangle_k}{2}s(n-1) - \frac{\triangle_k}{3}s(n)\right] \\ &+ \left[\frac{\triangle_k{}^2}{2}s(n-2) - \triangle_k{}^2 * s(n-1) + \frac{\triangle_k{}^2}{2}s(n)\right] \\ &+ \left[\frac{\triangle_k{}^3}{6}s(n-3) - \frac{\triangle_k{}^3}{2}s(n-2) + \frac{\triangle_k{}^3}{2}s(n-1) - \frac{\triangle_k{}^3}{6}s(n)\right]\end{aligned}$$

Se agrupan los coeficientes de acuerdo con las muestras que multiplican:

$$\boldsymbol{y(k)} = \boldsymbol{s(n)}\left[\frac{-\triangle_k}{3} + \frac{\triangle_k{}^2}{2} - \frac{\triangle_k{}^3}{6}\right] + \boldsymbol{s(n-1)}\left[1 - \frac{\triangle_k}{2} - \triangle_k{}^2 + \frac{\triangle_k{}^3}{2}\right] +$$

$$\boldsymbol{s(n-2)}\left[\triangle_k + \frac{\triangle_k{}^2}{2} - \frac{\triangle_k{}^3}{2}\right] + \boldsymbol{s(n-3)}\left[\frac{-\triangle_k}{6} + \frac{\triangle_k{}^3}{6}\right]$$

Estos son los coeficientes de los filtros polifásicos. Para cada valor de $\triangle_k$ (entre $\mathbf{1}$ y $\boldsymbol{P}$, siendo $\boldsymbol{P}$ el factor de interpolación).

$$a_0 = \left[\frac{-\triangle_k}{3} + \frac{\triangle_k{}^2}{2} - \frac{\triangle_k{}^3}{6}\right] \qquad a_2 = \left[\triangle_k + \frac{\triangle_k{}^2}{2} - \frac{\triangle_k{}^3}{2}\right]$$

$$a_1 = \left[1 - \frac{\triangle_k}{2} - \triangle_k{}^2 + \frac{\triangle_k{}^3}{2}\right] \qquad a_3 = \left[\frac{-\triangle_k}{6} + \frac{\triangle_k{}^3}{6}\right]$$

A continuación, se desarrolla un ejemplo para un cambio de tasa de $\mathbf{4/3}$ con un filtro polifásico de **4** etapas (factor de interpolación)**:**

Tabla 4: Coeficientes Filtro Polifásico de Lagrange para $\boldsymbol{L = 4}$

Subfiltro	$\triangle_k$	a_0	a_1	a_2	a_3
1	1/4	-0,0547	0,8203	0,2734	-0,0391
2	1/2	-0,0625	0,5625	0,5625	-0,0625
3	3/4	-0,0391	0,2734	0,8203	-0,0547
4	**1**	0,0000	0,0000	1,0000	0,0000

La ***Figura 18*** muestra el modelo Simulink del Filtro Polifásico de **4** etapas (subfiltros) y un diezmado a la salida por **3** para emular el cambio de tasa de $\mathbf{4/3}$**.**

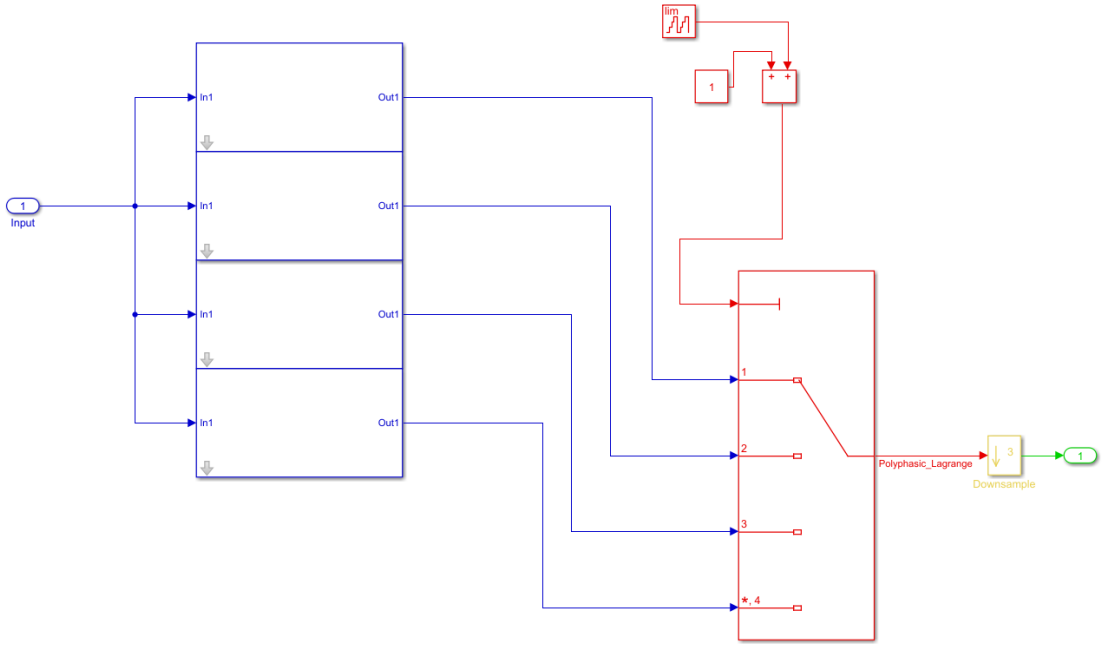


Figura 18: Modelo Simulink Filtro Polifásico-Polinomio de Lagrange

Los coeficientes se reparten en cada uno de los subfiltros del filtro polifásico como se muestra en la ***Figura 19***. La nomenclatura $\boldsymbol{HP(n,y)}$ corresponde a los coeficientes mostrados en la ***Tabla 4***, donde $\boldsymbol{y}$ es el índice del coeficiente $(\boldsymbol{a_0}, \boldsymbol{a_1}, \boldsymbol{a_2}, \boldsymbol{a_3})$ y $\boldsymbol{n}$ es el índice del subfiltro $(1, 2, 3, 4)$.

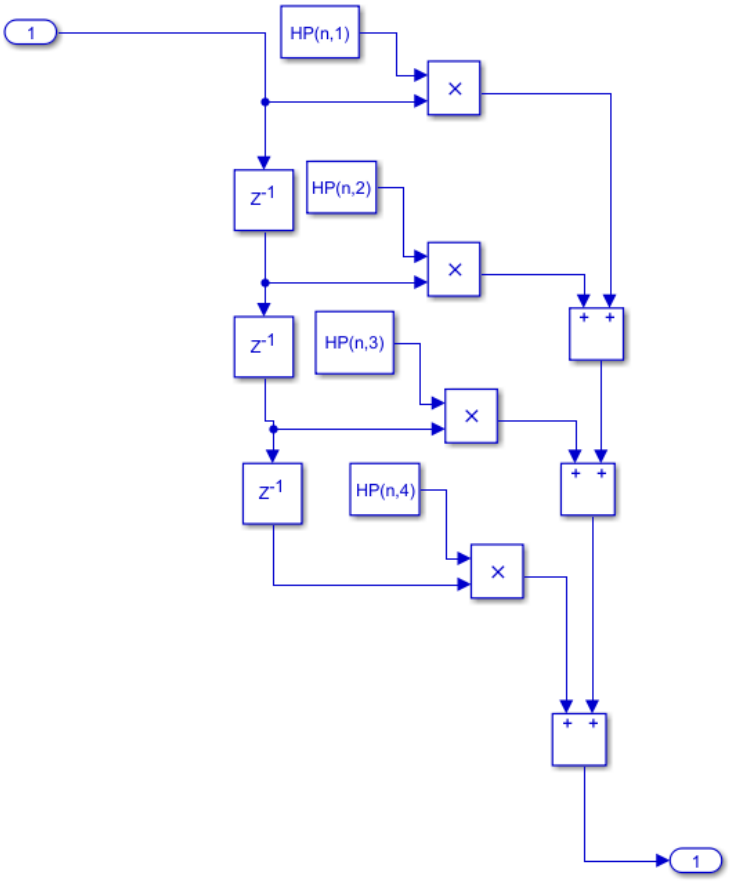


Figura 19: Estructura Interna Subfiltro-Filtro Polifásico

Los recursos empleados para implementar un filtro polifásico utilizando polinomios de Lagrange constan en ***Tabla 5***:

Tabla 5: Recursos empleados Filtro Polifásico-Polinomio de Lagrange

Recurso	Cantidad
Registros	12
Sumadores	12
Multiplicadores	16

En la ***Figura 20*** se observa la señal de entrada con una tasa de muestreo de $\frac{1}{F_s}$ y la señal de salida con un cambio en su tasa de muestreo a $\frac{3}{4F_s}$.

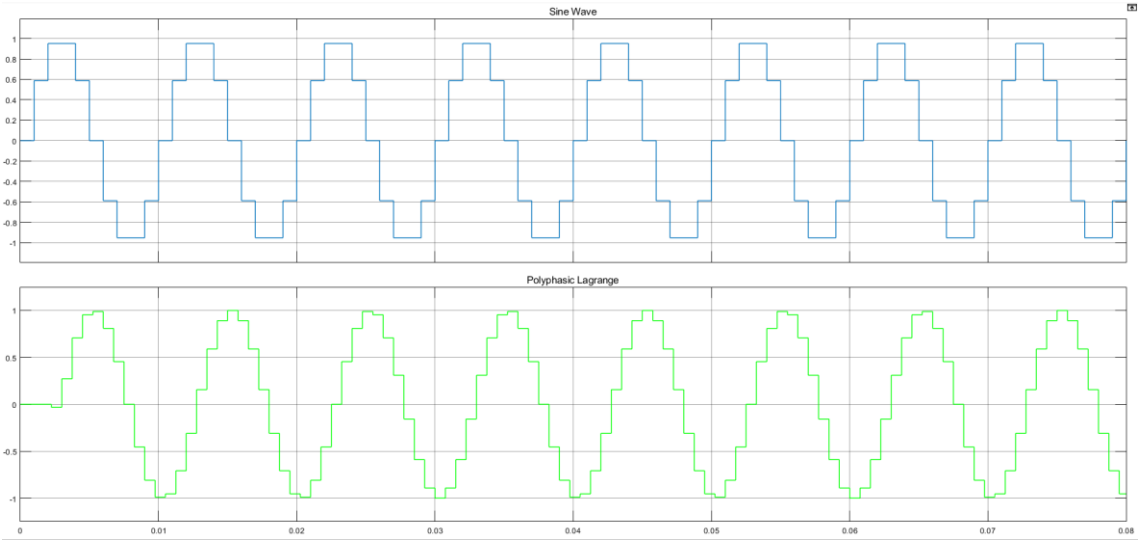


Figura 20: Señal Entrada y Salida-Filtro Polifásico

Como se ha expuesto, los filtros polifásicos se emplean para implementar un cambio de tasa, pero estos también pueden usarse para implementar un retardo fraccionario, esto se puede conseguir con un filtro polifásico interpolador. Si tomamos las salidas de los subfiltros de la estructura de la ***Figura 18***, con el primer subfiltro obtenemos un retardo de $\mathbf{1/4}$, con el segundo subfiltro un retardo de $\mathbf{2/4}$, con el tercer subfiltro un retardo de $\mathbf{3/4}$ y con el cuarto subfiltro un retardo de $\mathbf{4/4}$, esto de aprecia en la ***Figura 21.***

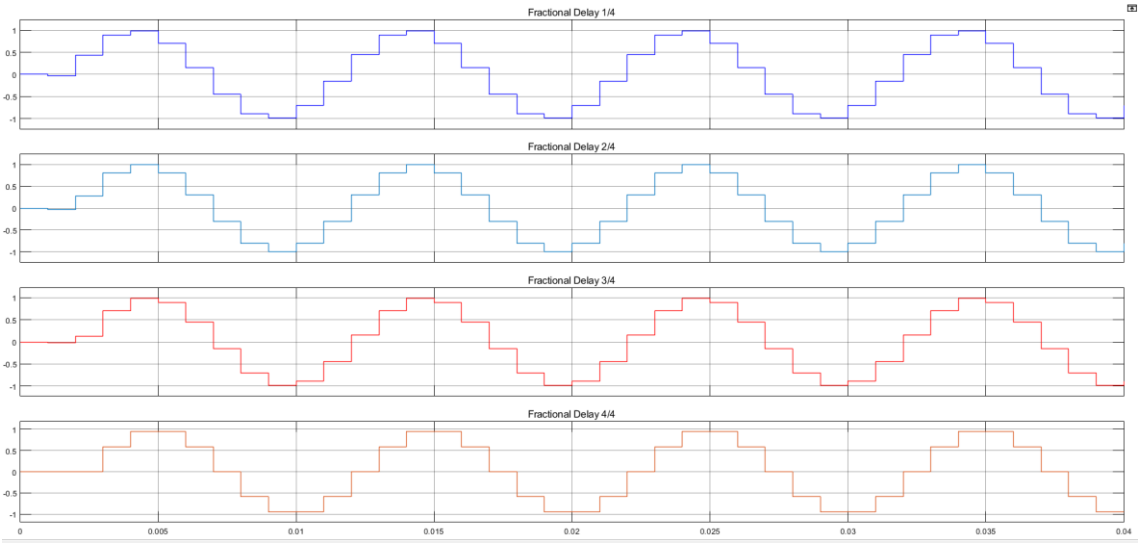


Figura 21: Retardo Fraccionario Filtro Polifásico Interpolador (4 subfiltros)

2.2.2.1 *Optimización del Filtro Polifásico de Lagrange*

Los coeficientes del Polinomio de Lagrange expuestos en la ***Ecu. 8*** se reagrupan en función de las muestras de la señal en el tiempo para obtener los coeficientes de los filtros polifásicos.

El objetivo ahora es disminuir el uso de recursos utilizados, por lo que es necesario desarrollar un control para que cada $\frac{Fs}{L}$ los coeficientes del filtro cambien ya que la estructura de cada subfiltro es exactamente la misma como se muestra en la ***Figura 19***; en Simulink se desarrolló un control para que con un único filtro se pueda obtener exactamente la misma señal de salida que la obtenida con el uso de los subfiltros como se muestra en la ***Figura 22***:

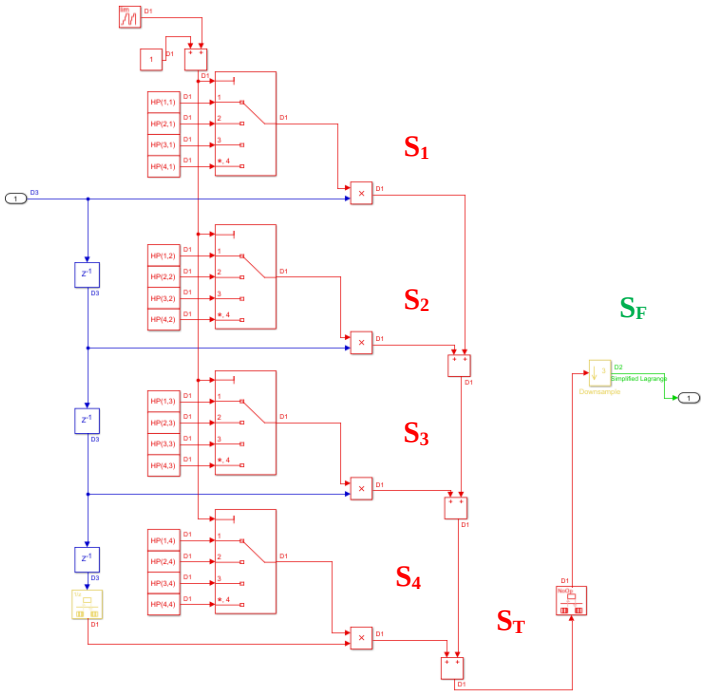


Figura 22: Modelo Simulink Filtro Polifásico de Lagrange Simplificado

Para comprender el funcionamiento de este filtro la ***Figura 23*** muestra un cronograma de tiempo donde se puede apreciar cómo se obtiene la señal de salida en cada parte del circuito mostrado en la ***Figura 22.***

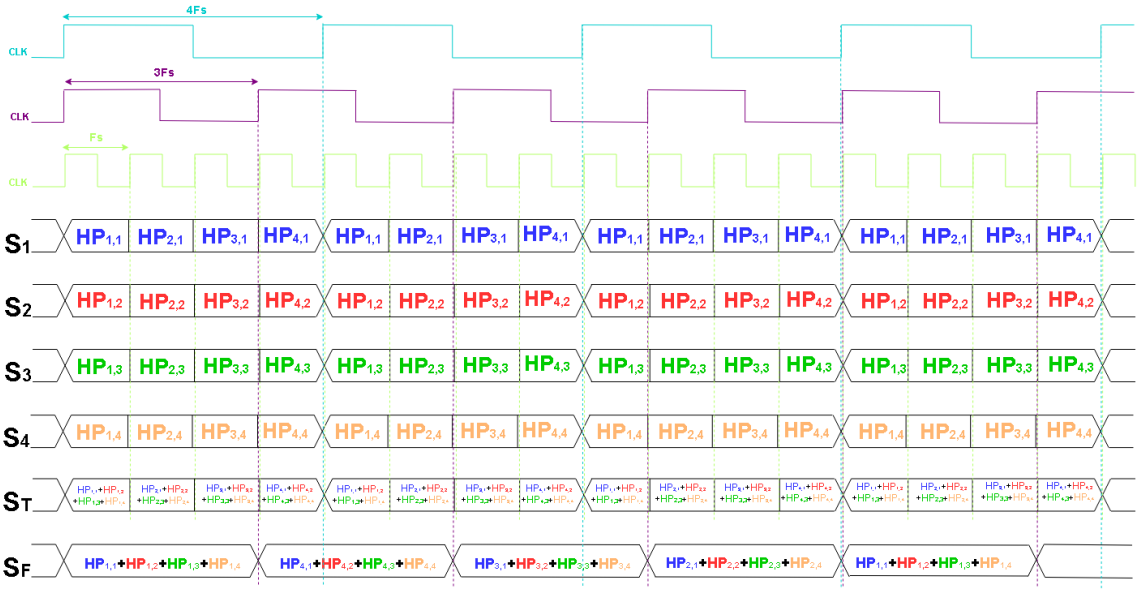


Figura 23: Cronograma de Tiempo Filtro Polifásico de Lagrange Simplificado

Los recursos empleados por el Filtro Polifásico de Lagrange Simplificado del Modelo Simulink constan en **Tabla 6**, cabe recalcar que este modelo utiliza los mismos recursos que una etapa (subfiltro) del Filtro Polifásico de Lagrange.

Tabla 6: Recursos empleados Filtro Polifásico de Lagrange Simplificado

Recurso	Cantidad
Registros	3
Sumadores	3
Multiplicadores	4

La ***Figura 24*** se aprecia que la salida del filtro es la misma que la ***Figura 20*** con la diferencia que el número de recursos empleados es menor en el Filtro Polifásico.

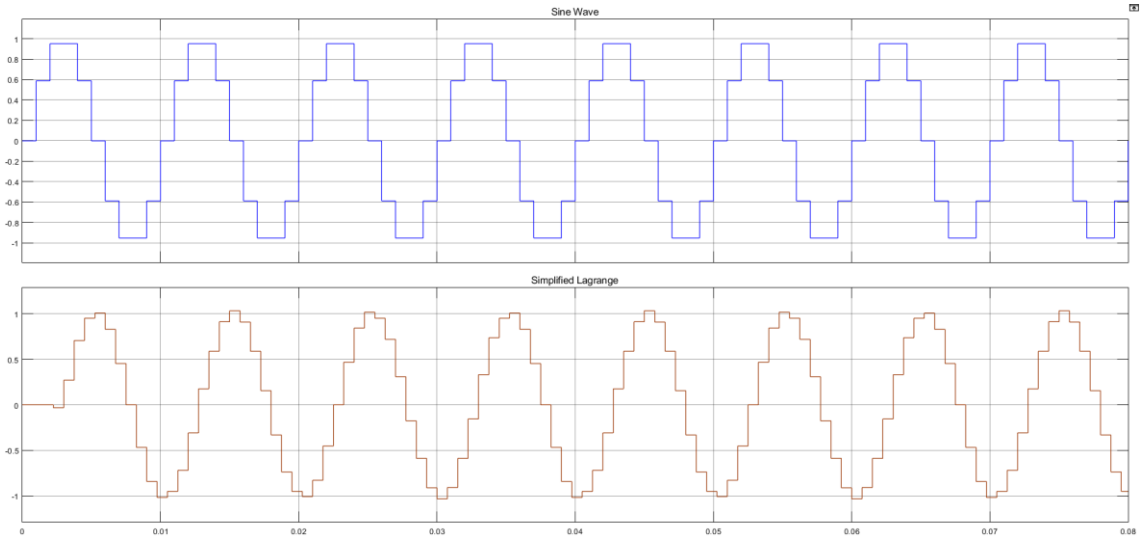


Figura 24: Señal Entrada y Salida Lagrange Simplificado

2.2.3 Implementación con la estructura de Farrow

Si planteamos en términos generales el problema de cambio de tasa de muestreo, se dispone de una señal de entrada $\boldsymbol{s(n)}$ para $\boldsymbol{n = 0, 1, 2, \ldots}$ muestreada a una frecuencia $\boldsymbol{Fs}$, donde cada $\boldsymbol{n}$ corresponde a un instante de tiempo $\boldsymbol{t_n = \frac{n}{Fs}}$.

Para calcular los valores digitales de la señal $\boldsymbol{y(k)}$ para $\boldsymbol{k = 0, 1, 2, \ldots}$ muestreada a $\boldsymbol{F_y = \frac{L}{M} * F_s}$, siendo $\boldsymbol{L}$ y $\boldsymbol{M}$ números enteros. La primera muestra de la señal $\boldsymbol{k}$ $\boldsymbol{(y(0))}$ esta retrasada con respecto a su correspondiente muestra de la señal $\boldsymbol{n}$ $\boldsymbol{(s(0))}$ por $\Delta_t = \frac{x_0}{F_s}$, el valor de $\boldsymbol{x_0}$ está entre $\boldsymbol{-1 \leq x_0 < 0}$.

Si definimos a $\boldsymbol{F_k}$ como frecuencia de muestreo de la señal digitalizada $\boldsymbol{y(k)}$ y tomando en cuenta el retraso Δ_t, podemos plantear la frecuencia de muestreo $\boldsymbol{F_k}$como:

$$\boldsymbol{F_k} = F_y * k - \frac{1}{\Delta_t}$$

$$\boldsymbol{F_k} = \frac{L * F_s}{M} * k - \frac{F_s}{x_0} \qquad \boldsymbol{Ecu.\,10}$$

De acuerdo con la ***Figura 17***, el valor $\boldsymbol{x(k)}$ corresponde a un instante de tiempo $\boldsymbol{t_k = \frac{xk}{Fs}}$. Si consideramos que $\boldsymbol{F_k = \frac{1}{t_k}}$, podemos replantear la ecuación $\boldsymbol{Ecu.\,10}$ como:

$$\frac{xk}{F_s} = \frac{M}{L * F_s} * k - \frac{x_0}{F_s}$$

$$\boldsymbol{xk} = \frac{M}{L} * k - x_0 \qquad \boldsymbol{Ecu.\,11}$$

El valor $\Delta_{\boldsymbol{k}}$ se define como:

$$\Delta_{\boldsymbol{k}} = x_k{}' + 1 - x_k \qquad \boldsymbol{Ecu.\,12}$$

$\boldsymbol{x_k}'$ corresponde al truncamiento del valor de x_k.

La ***Figura 25*** muestra la estructura de Farrow para calcular la salida del polinomio cúbico basado en la interpolación de Lagrange expuesto en la $\boldsymbol{Ecu.\,9}$, donde el valor de $\Delta_{\boldsymbol{k}}$ para este caso se calculará para cada instante de tiempo con la ecuación $\boldsymbol{Ecu.\,12.}$

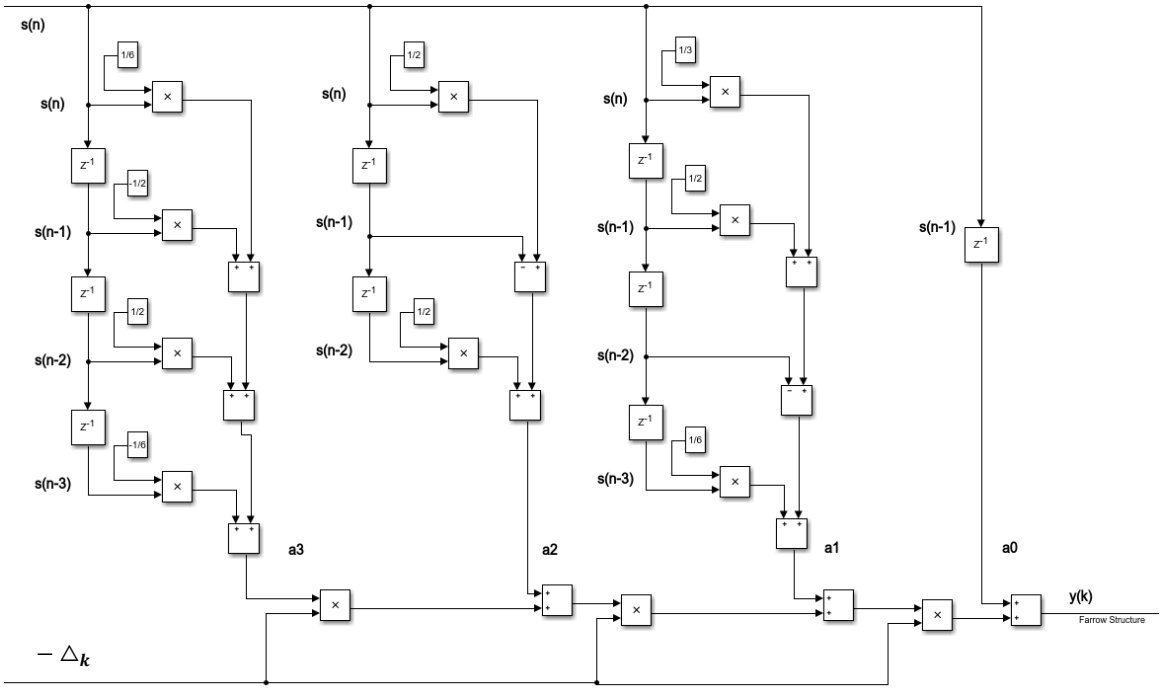


Figura 25: Estructura de Farrow- Polinomio de Lagrange

El ejemplo que se viene desarrollando para un cambio de tasa de $\boldsymbol{4/3}$ no presenta un valor de $\boldsymbol{x_0}$, así que es descartado de la ecuación. La ***Figura 26*** muestra la implementación del cálculo del valor de $\Delta_{\boldsymbol{k}}$ en Simulink, como el valor de $\boldsymbol{x_k}$ no excede la unidad, no es necesario truncar su valor $|\boldsymbol{x_k}|$, por lo que la ecuación quedaría de la siguiente manera:

$$-\Delta_{\boldsymbol{k}} = \boldsymbol{1} - \frac{M}{L} * k$$

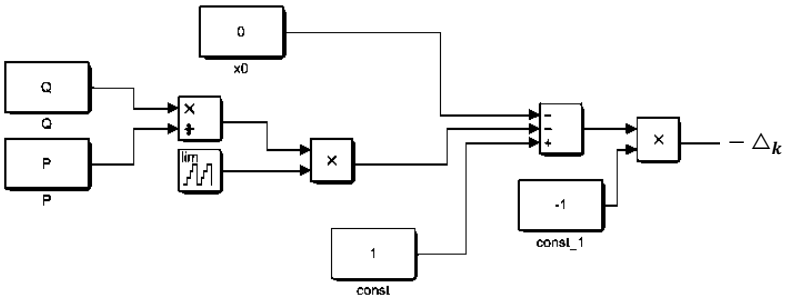


Figura 26: Calculo de $\Delta_{\boldsymbol{k}}$ en Simulink

Se multiplica por **-1** al final ya que buscamos obtener el valor de $-\triangle_{\boldsymbol{k}}$. Para emular el cambio de tasa de $\boldsymbol{4/3}$ en Simulink, se diezma a la salida por $\boldsymbol{3}$ como se muestra en la ***Figura 27.***

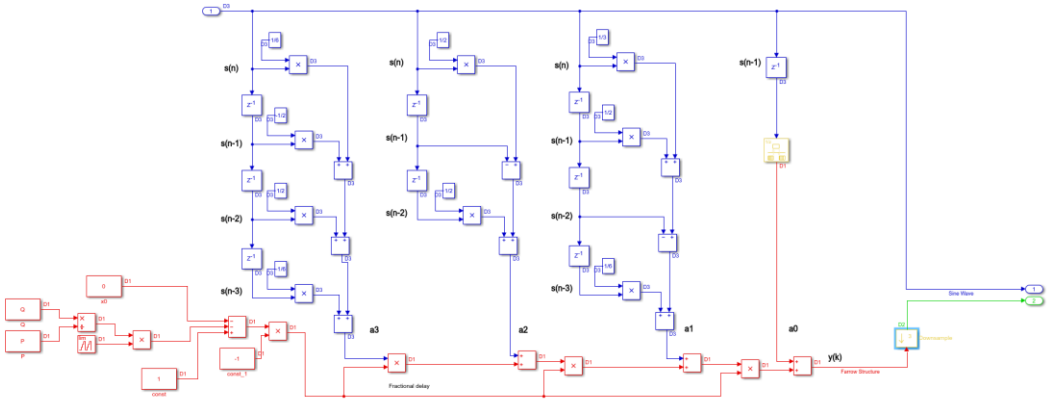


Figura 27: Modelo Simulink-Estructura de Farrow para $\boldsymbol{L = 4, M = 3, x_0 = 0}$

Los recursos empleados por el Filtro con la estructura de Farrow se muestran en la ***Tabla 7***:

Tabla 7: Recursos empleados Filtro con Estructura de Farrow

Recurso	Cantidad
Registros	8
Sumadores	12
Multiplicadores	13

En la ***Figura 28*** se aprecia la salida de la estructura de Farrow para $\boldsymbol{L = 4, M = 3}$ $\boldsymbol{y}$ $\boldsymbol{x_0 = 0}$.

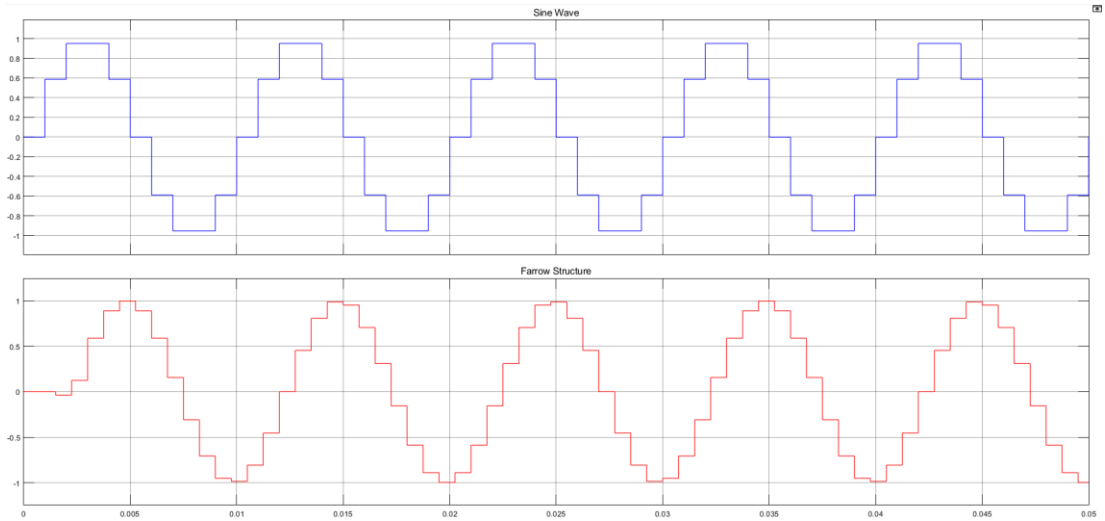


Figura 28: Señal de Entrada y Salida Estructura de Farrow para $\boldsymbol{L = 4, M = 3, x_0 = 0}$

2.2.3.1 Optimización del Filtro de Farrow

Nuevamente el objetivo es minimizar el uso de recursos utilizados para el cálculo de los coeficientes del filtro. Si operamos los coeficientes de la ***Ecu. 8,*** podemos sumar $\boldsymbol{a_3 + a_1}$:

$$a_3 + a_1 = \frac{1}{6}s(n) - \cancel{\frac{1}{2}s(n-1)} + \frac{1}{2}s(n-2) - \cancel{\frac{1}{6}s(n-3)}$$

$$+\frac{1}{3}s(n) + \cancel{\frac{1}{2}s(n-1)} - s(n-2) + \cancel{\frac{1}{6}s(n-3)}$$

$$a_3 + a_1 = \frac{1}{2}s(n) - \frac{1}{2}s(n-2)$$

$$a_1 = \frac{1}{2}s(n) - \frac{1}{2}s(n-2) - a_3 \qquad \textbf{Ecu. 13}$$

También podemos sumar a la ecuación $\boldsymbol{Ecu.13}$ el valor de $\boldsymbol{a_2}$ obtenemos:

$$a_1 + a_2 = \frac{1}{2}s(n) - \frac{1}{2}s(n-2) - a_3 + \frac{1}{2}s(n) - s(n-1) + \frac{1}{2}s(n-2)$$

$$a_1 + a_2 = s(n) - s(n-1) - a_3$$

$$a_2 = s(n) - s(n-1) - a_1 - a_3 \qquad \boldsymbol{Ecu.14}$$

Con las ecuaciones $\boldsymbol{Ecu.13}$ y $\boldsymbol{Ecu.14}$ y si reagrupamos términos para $\boldsymbol{a_3}$, podemos rescribir de forma optimizada el sistema de ecuaciones con solo tres multiplicadores:

$$\begin{cases} a_0 = s(n-1) \\ a_1 = \frac{1}{2}s(n) - \frac{1}{2}s(n-2) - a_3 \\ a_2 = s(n) - s(n-1) - a_1 - a_3 \\ a_3 = \frac{1}{6}\big(s(n) - s(n-3)\big) + \frac{1}{2}(s(n-2) - s(n-1)) \end{cases} \qquad \boldsymbol{Ecu.15}$$

La estructura en Simulink del filtro basado en la estructura de Farrow Optimizada correspondiente a la $\boldsymbol{Ecu.15}$ se muestra en la ***Figura 29.***

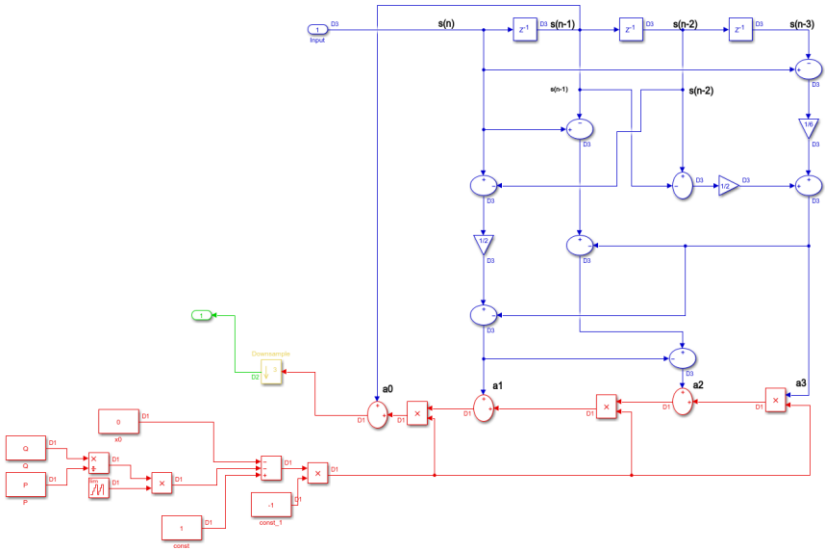


Figura 29: Modelo Simulink-Estructura de Farrow Optimizada

La ***Figura 30*** muestra la salida de la estructura de Farrow Optimizada para $\boldsymbol{L = 4, M = 3}$ y $\boldsymbol{x_0 = 0}$, la salida es similar a la ***Figura 28*** de la estructura de Farrow sin optimizar.

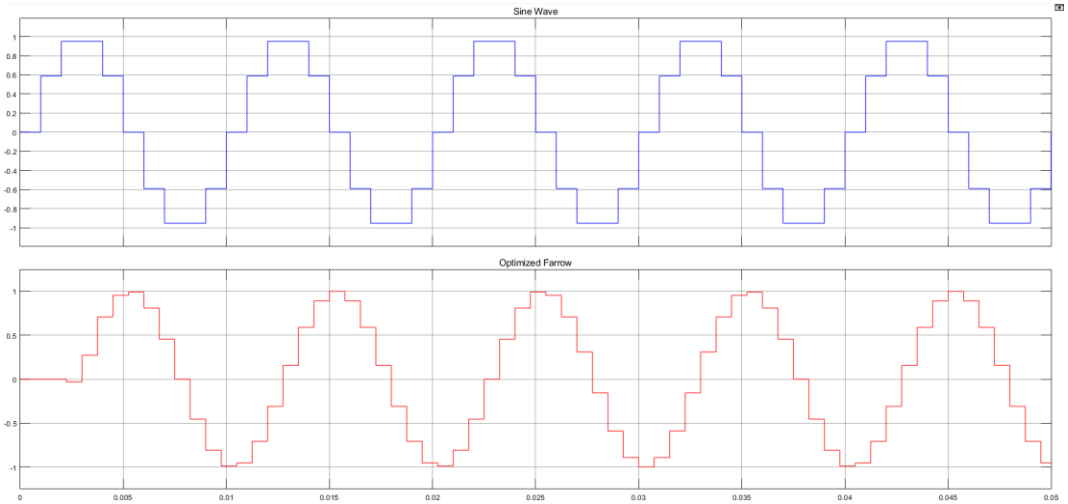


Figura 30: Señal Entrada y Salida Farrow Optimizado para $\boldsymbol{L = 4, M = 3, x_0 = 0}$

Ahora tenemos un sistema que calcula los coeficientes de la interpolación polinómica de Lagrange utilizando un solo multiplicador por $\frac{1}{6}$ y dos multiplicadores triviales por $\frac{1}{2}$, lo que reduce los recursos empleados en el Modelo Simulink que constan en la ***Tabla 8***.

Tabla 8: Recursos empleados Filtro con Estructura de Farrow Optimizada

Recurso	Cantidad
Registros	3
Sumadores	11
Multiplicadores	7

2.3 Comparativa y Recursos Empleados

Tras revisar algunas metodologías para implementar un cambio de tasa por un factor racional, se desarrolló otro ejemplo para un cambio de tasa de $\mathbf{3/4}$ para los métodos anteriormente expuestos. Se realizará una comparativa de la cantidad de recursos utilizados para posteriormente escoger cual es la metodología a implementar sobre el experimento, detallando las razones que nos llevaron a escogerla.

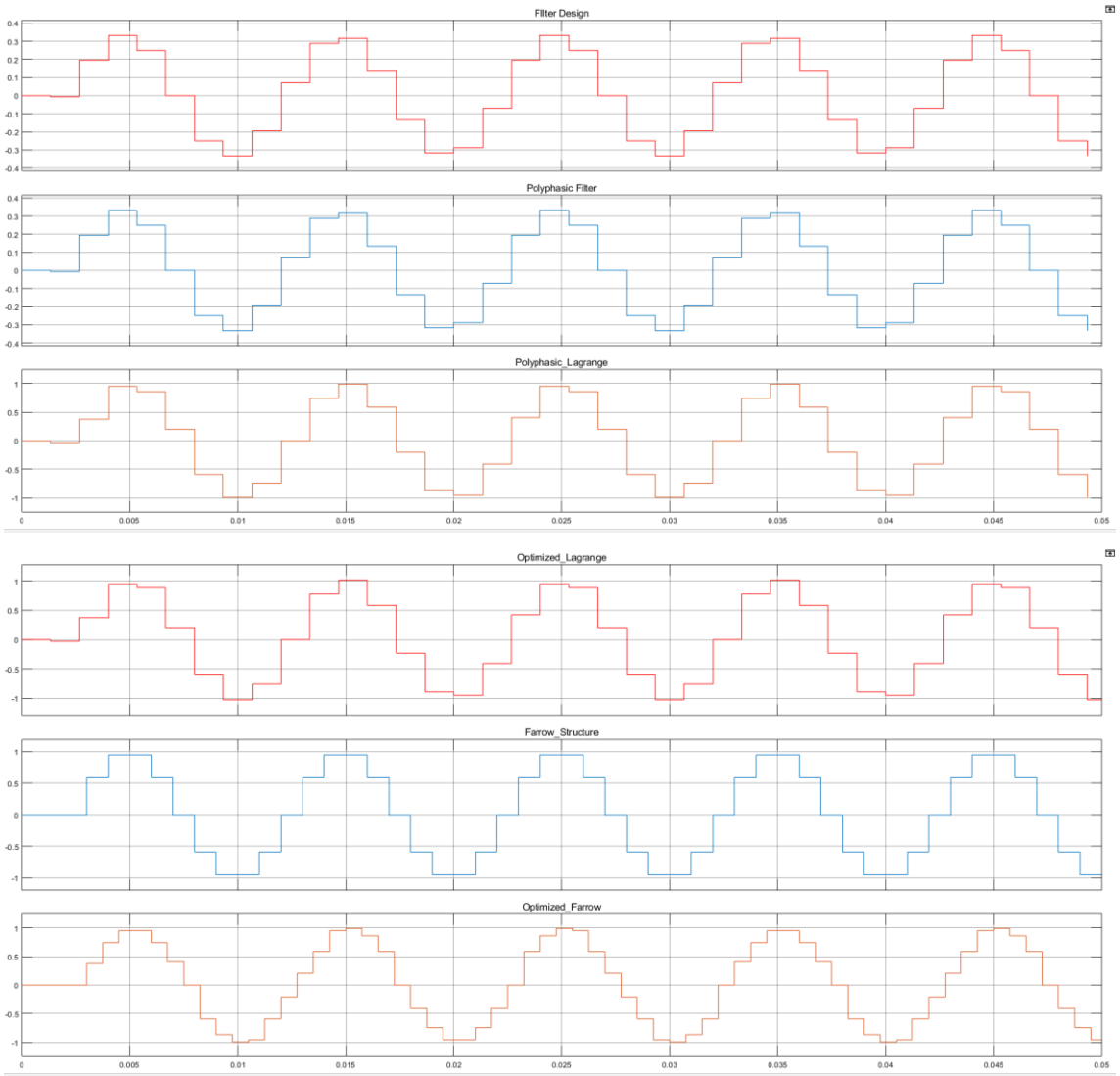


Figura 31: Señales de Salida para un cambio de tasa de $\mathbf{3/4}$

La ***Figura 31*** muestra las señales de salida de cada método aplicado para un cambio de tasa de $\mathbf{3/4}$, para desarrollar este ejemplo se siguió los mismos pasos descritos en las secciones

anteriores modificando los circuitos de Simulink para obtener este nuevo valor de cambio de tasa. Los circuitos de Simulink y los cambios que se realizaron se detallan a continuación:

La implementación del cambio de tasa con un filtro polifásico se muestra en la ***Figura 32***, a diferencia de la ***Figura 15,*** solo son necesarias **3** subfiltros y al final se aplica un down sampler por **4** (factor de diezmado).

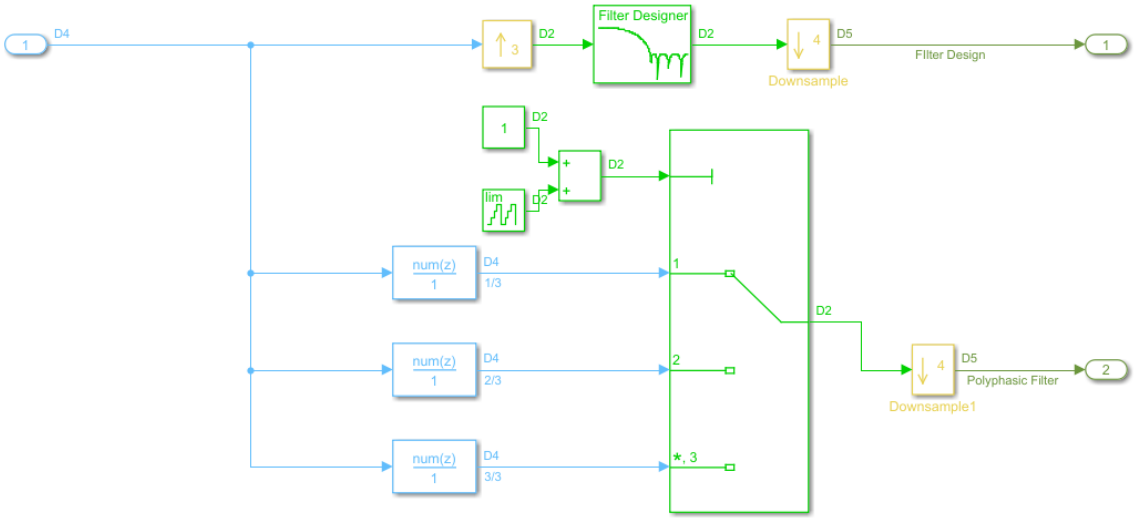


Figura 32: Modelo Simulink Filtro Polifásico para $3/4$

El diseño del filtro con *fdatool* cambia, para este caso se obtienen **13** coeficientes. Los parámetros de diseño se pueden apreciar en la ***Figura 33.***

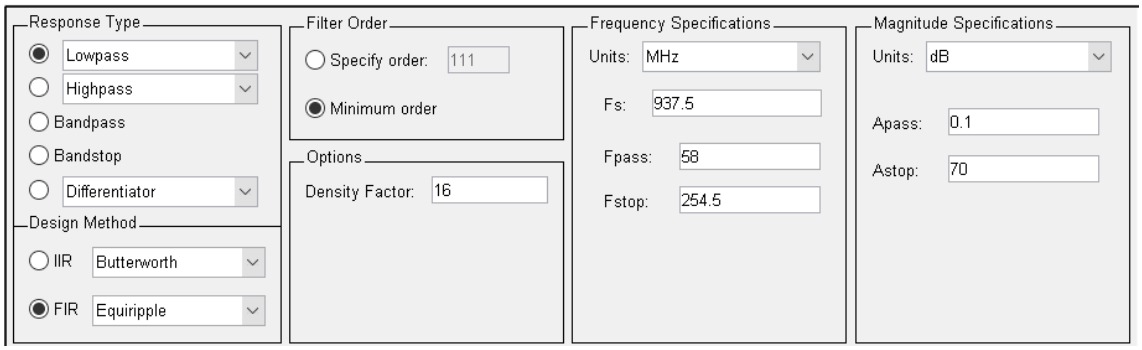


Figura 33: Parámetros Diseño Filtro (*fdatool*) para $3/4$

La ***Figura 34*** muestra el diagrama de bloques con las frecuencias en cada punto del sistema para un cambio de tasa de$3/4$, y en la ***Figura 35*** se muestra el análisis de espectros de todas las señales involucradas lo que nos permite analizar y establecer los parámetros del filtro.

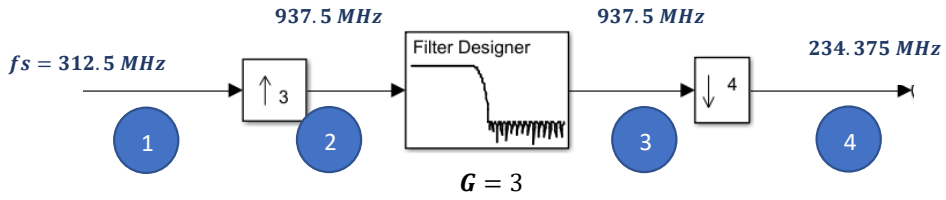


Figura 34: Diagrama de Bloques cambio de tasa para $3/4$

Del mismo modo que en la ***Figura 10*** en el punto ***1*** la señal se muestrea a una frecuencia fs, en el punto ***2*** la señal se interpola lo que produce **3** copias espectrales por cada ciclo de la señal, en el punto ***3*** la señal pasa por el filtro que elimina estas copias espectrales y limita el ancho de

banda para evitar que se produzca aliasing tras el diezmado y en el punto ***4*** se diezma la señal, expandiendo su espectro por un factor **4.**

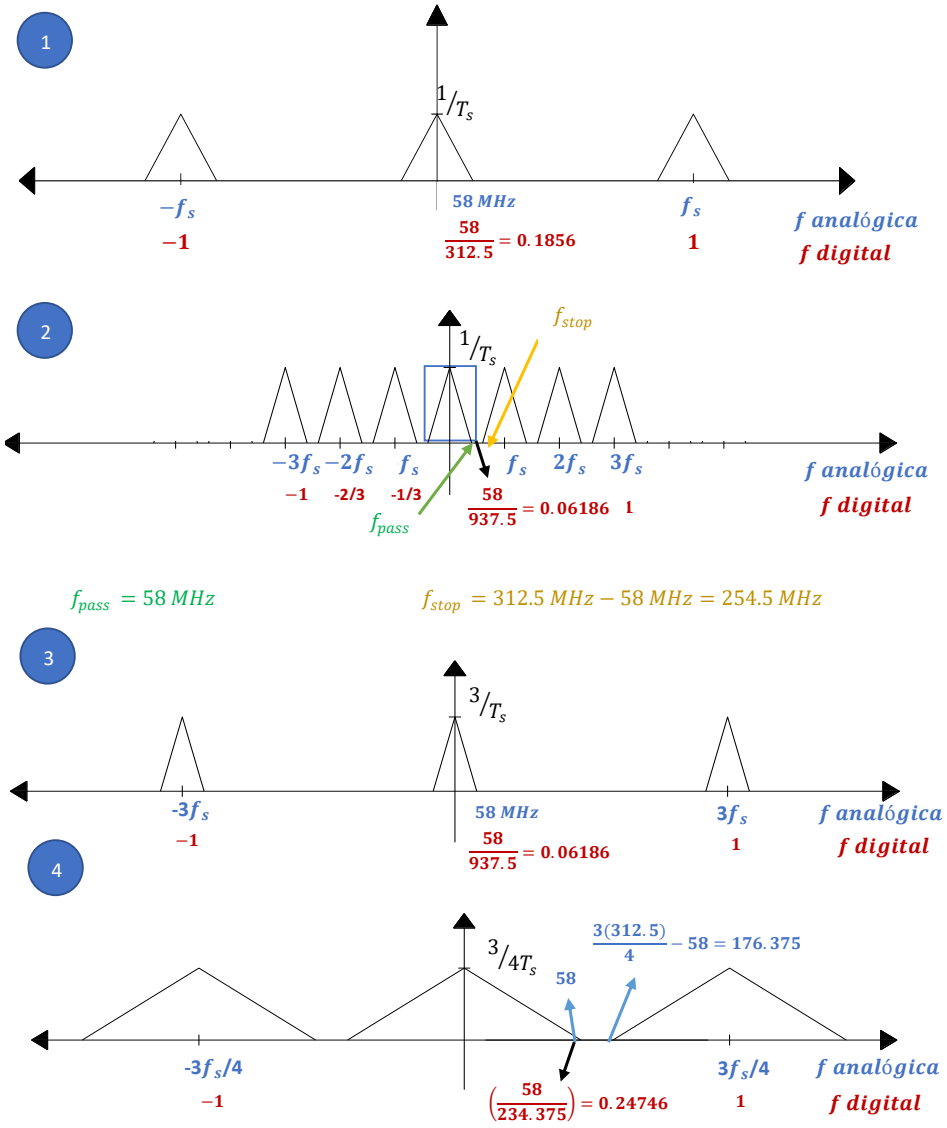


Figura 35: Análisis Espectros de señales cambio de tasa para $\boldsymbol{Q = 3/4}$

La implementación del cambio de tasa con un filtro polifásico usando polinomios de Lagrange se muestra en la ***Figura 32***, a diferencia de la ***Figura 18,*** solo son necesarias **3** subfiltros y al final se aplica un diezmador por **4** (factor de diezmado). La estructura de los subfiltros es la misma que se muestra en la ***Figura 19.***

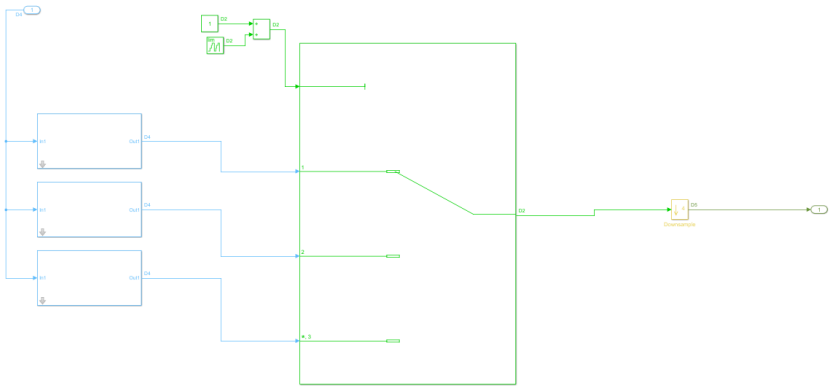


Figura 36: Modelo Simulink Filtro Polifásico-Polinomio de Lagrange para $\boldsymbol{3/4}$

La ***Tabla 9*** muestra los coeficientes para el filtro polifásico de Lagrange de **3** etapas (factor de interpolación)**:**

Tabla 9: Coeficientes Filtro Polifásico de Lagrange para $\boldsymbol{L = 3}$

Subfiltro	Δ_k	a_0	a_1	a_2	a_3
1	1/3	-0,0617	0,7407	0,3704	-0,0494
2	2/3	-0,0494	0,3704	0,7407	-0,0617
3	1	0,0000	0,0000	1,0000	0,0000

La optimización del Filtro Polifásico de Lagrange utiliza la misma estructura mostrada en la ***Figura 22,***la diferencia radica en que para el cambio de tasa de $\boldsymbol{3/4}$ solo son necesarios 3 subfiltros. La ***Figura 37*** muestra los cambios realizados en el modelo de Simulink.

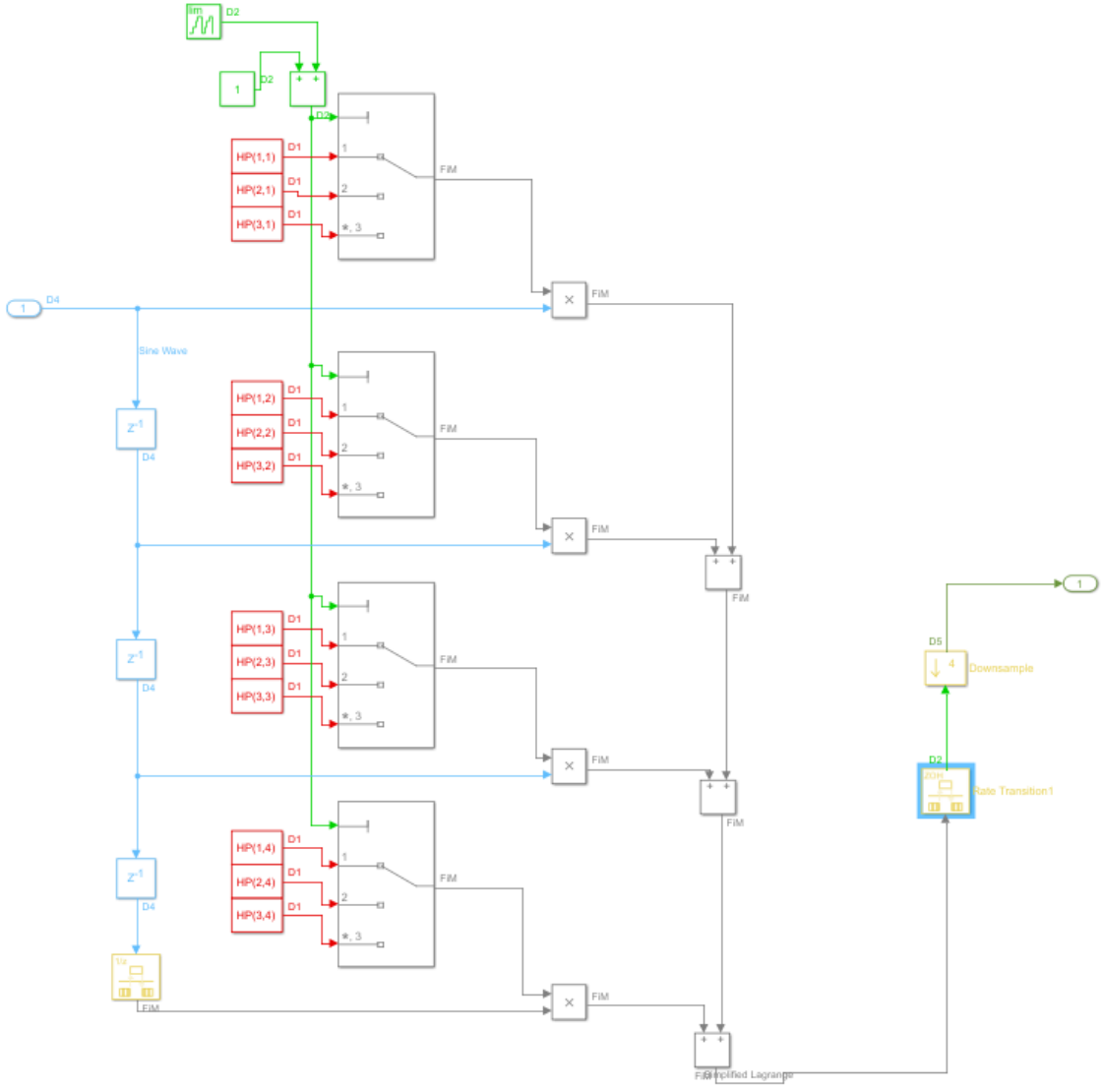


Figura 37: Modelo Simulink Filtro Polifásico de Lagrange Simplificado para $\boldsymbol{3/4}$

Para implementar la estructura de Farrow y su optimización solo fue necesario cambiar el valor de $\boldsymbol{P}$ de ***Figura 26*** porque la estructura que emplea es la misma y solo cambia el proceso para calcular el valor de Δ_k.El valor de $\boldsymbol{P}$ (factor de interpolación) para este caso es **3.**

Con respecto al tamaño de las memorias necesarias para almacenar los coeficientes del filtro se procedió a su cuantificación con la ayuda de las funciones ***quantize*** y ***quantizer*** de Matlab para

conocer el número de bits enteros necesarios para su cuantificación, estos resultados se resumen en la ***Tabla 10.***

Tabla 10: Cuadro comparativo de memorias empleados por cada metodología

ESTRUCTURA	4/3		3/4	
	#COEF	MEMORIA	#COEF	MEMORIA
Filtro Polifásico	*24*	*32 **x** 16 bits*	*11*	*16 **x** 16 bits*
Filtro Polifásico (Inter. Lagrange)	*16*	*16 **x** 9 bits*	*12*	*16 **x** 9 bits*
Optimización Filtro Lagrange	*16*	*16 **x** 9 bits*	*12*	*16 **x** 9 bits*
****Farrow***	-	-	-	-
****Farrow Optimizado***	-	-	-	-

* estas estructuras calculan sus coeficientes para cada muestra (no emplean memorias)

A continuación, se muestra la ***Tabla 11*** donde se resumen la cantidad de recursos empleados en cada una de las metodologías implementadas en Simulink para los dos casos de cambio de tasa $(\frac{3}{4} y \frac{4}{3})$.

Tabla 11: Cuadro comparativo de recursos empleados por cada metodología

ESTRUCTURA	4/3			3/4		
	REG	SUM	MULT	REG	SUM	MULT
Filtro Polifásico	*23*	*23*	*24*	*10*	*10*	*11*
Filtro Polifásico (Inter. Lagrange)	*12*	*12*	*16*	*9*	*9*	*12*
Optimización Filtro Lagrange	*3*	*3*	*4*	*3*	*3*	*4*
Farrow	*8*	*12*	*13*	*8*	*12*	*13*
Farrow Optimizado	*3*	*11*	*7*	*3*	*11*	*7*

Si observamos la tabla anterior, para las tres últimas metodologías (Optimización del Filtro de Lagrange, Farrow y su Optimización), el número de recursos se mantiene para cualquier valor de cambio de tasa.

Con respecto a la implementación del Filtro Polifásico, el procedimiento es único para cada valor de cambio de tasa ya que se tiene que estimar los parámetros para el diseño del filtro en cada caso, esto al momento de programarlo nos limitaría la reutilización de código.

Capítulo 3. ARQUITECTURAS PARA LA IMPLEMENTACIÓN DE INTERPOLADORES /DIEZMADORES FRACCIONALES

A la hora de aplicar técnicas de procesado digital ***DSP***, el problema principal que identificamos es la elevada velocidad de transmisión que cada circuito debe alcanzar, mientras que los dispositivos como las FPGAs poseen recursos que trabajan en torno a los ***500 MHz***, la transmisión por fibra óptica demanda valores cercanos a la decena de GHz. Esto muestra la necesidad de elegir la arquitectura adecuada a emplear para poder alcanzar estas prestaciones de velocidad.

Este capítulo muestra las diferentes arquitecturas disponibles, se detalla las implementaciones desarrolladas para montarlas sobre el experimento y se incluyen resultados de velocidad y área, esto nos permite analizar si cumplen con las restricciones que demanda el sistema que se quiere implementar

3.1 Procesado Secuencial

Una técnica muy empleada para el diseño e implementación de los filtros FIR, tiene en cuenta la descripción de los filtros mediante una secuencia de elementos de retardo $\boldsymbol{R+1}$ muestras, de acuerdo con la convolución discreta, las muestras de señal $\boldsymbol{x[m]}$ se multiplican con los coeficientes del filtro $\boldsymbol{h[r]}$ y suman a la salida $\boldsymbol{y[m]}$ **[6]**:

$$\boldsymbol{y[m]} = \sum_{r=-R/2}^{+R/2} x[m-r]h[r]\; m, r \in Z \qquad \boldsymbol{Ecu.16}$$

Se debe tener en cuenta que cada muestra $\boldsymbol{x[m]}$ a la entrada del filtro genera una respuesta al impulso ***IR-Impulse Response*** $\boldsymbol{x[m]h[r]}$, donde $\boldsymbol{m}$ y $\boldsymbol{r}$ representan puntos discretos en el tiempo. Si se dispone un número limitado de valores diferentes a la entrada del filtro $\boldsymbol{x[m]}$, la salida $\boldsymbol{y[m]}$ se deduce de una superposición lineal de todos los ***IR*** debidamente retardados, es así como se muestra en la ***Figura 38 (***las muestras entran de manera secuencial***).***

Se generan individualmente todos los ***IRs,*** las multiplicaciones requeridas pueden ser calculadas y almacenadas en tablas de búsqueda **LUT [6]**.

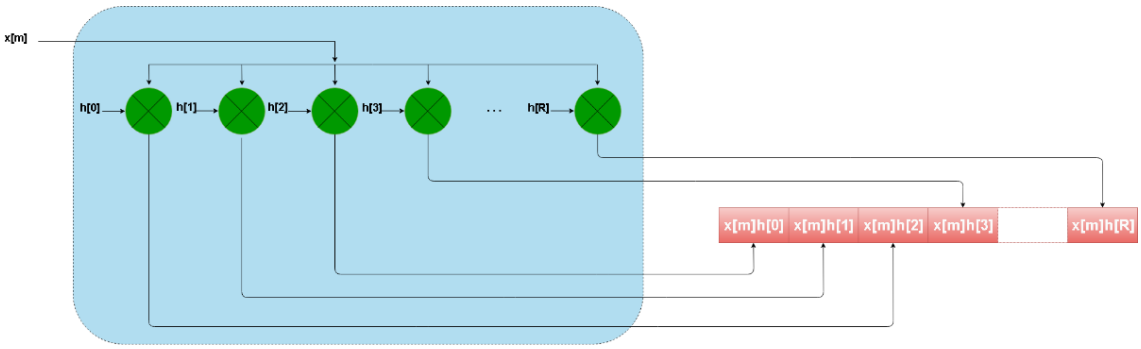


Figura 38: Generador de Respuesta al Impulso IR

A pesar de que se emplean pocos recursos, la velocidad se ve limitada porque es necesario realizar varias iteraciones sobre este operador para conseguir el filtrado completo. Con este tipo de arquitectura la frecuencia de reloj del sistema deberá ser $\boldsymbol{R}$ veces mayor que la frecuencia de muestreo ($\boldsymbol{f_{clk}} = R * f_s$), siendo $\boldsymbol{R}$ el número de coeficientes del filtro a implementar.

Por todo esto descartamos implementar este tipo de procesado porque se ve claramente que no va a ser posible cumplir los requisitos de velocidad del sistema, sin embargo, el generador de Respuesta al Impulso nos servirá de base para poder desarrollar las implementaciones posteriores aplicando otro tipo de procesado.

3.2 Procesado Paralelo

Al implementar un procesado paralelo, este nos permite procesar una señal a la frecuencia de muestreo ($\boldsymbol{f_{clk} = f_s}$). En los siguientes apartados, se describen las implementaciones de $\mathbf{4/3}$ para el transmisor y $\mathbf{3/4}$ para el receptor, estas se desarrollan con filtros de cuatro coeficientes procesados a cuatro muestras por reloj ***Figura 39***.

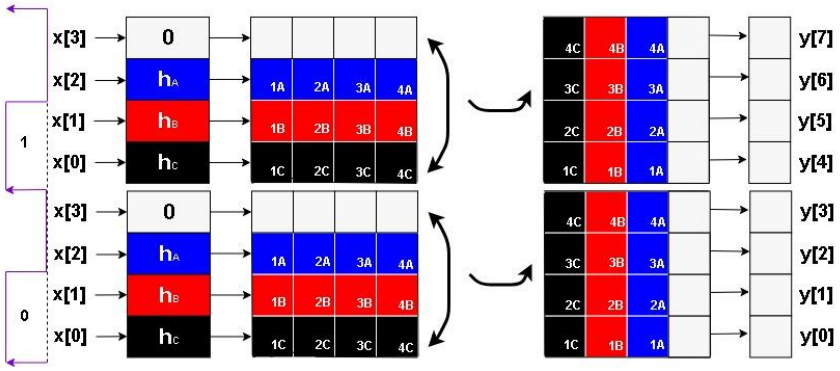


Figura 39: Diseño de Filtro FIR Paralelo con 4 coeficientes

3.2.1 *Interpolación por 4/3 paralelizado por Px4*

La señal de entrada se ingresa de manera secuencial mientras que a la salida se obtiene **4** líneas en paralelo, los coeficientes de la ***Tabla 4*** se reparten en los subfiltros como se mencionó en el apartado ***2.2.1***. Lo que resta es obtener las **4** salidas, cada subfiltro dispone de **4** salidas que corresponden a la etapa de interpolación.

Las salidas del filtro se obtienen sumando todas las muestras del mismo número (*1C, 2C, 3C*) correspondientes a la misma fila de la matriz bidimensional ***2D*** creada a partir de las respuestas al impulso (***Figura 39***)**.** Lo que resta es reordenar las salidas, es decir quedarnos con una de cada **3** muestras (factor de diezmado).

La ***Figura 40*** muestra el reordenamiento de las salidas del filtro paralelo donde la primera línea de puntos ($\boldsymbol{n}$) corresponde a las muestras de entrada, la segunda línea corresponde a las salidas ($\mathbf{1, 2, 3}$ ***y*** $\mathbf{4}$) del filtro interpolador. mientras que la última línea de puntos ($\boldsymbol{m}$) corresponde a las salidas reordenadas del filtro interpolador.

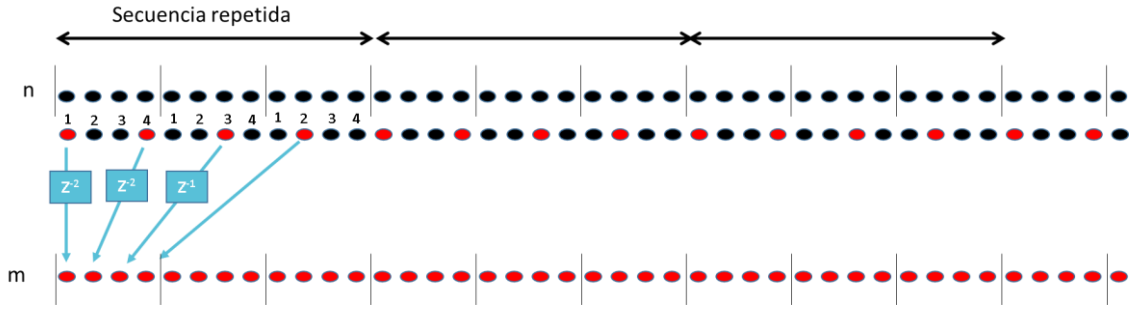


Figura 40: Reordenamiento de salidas para $\mathbf{4/3}$ paralelizado por $\boldsymbol{Px4}$

Para este caso las salidas *4m, 4m+1, 4m+2* y *4m+3* se obtienen retardando las salidas (*1, 4, 3* y *2*) del filtro paralelo como muestra el diagrama de la ***Figura 41***

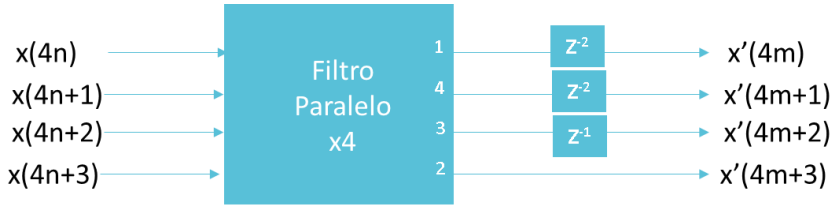


Figura 41: Diagrama de Bloques Filtro Paralelo $\boldsymbol{Px4}$ para $\boldsymbol{Q = 4/3}$

Se implementó el modelo en Simulink ***Figura 42*** para comprobar su correcto funcionamiento. Cada subsistema equivale al generador de respuesta al impulso mostrado en la ***Figura 38.***

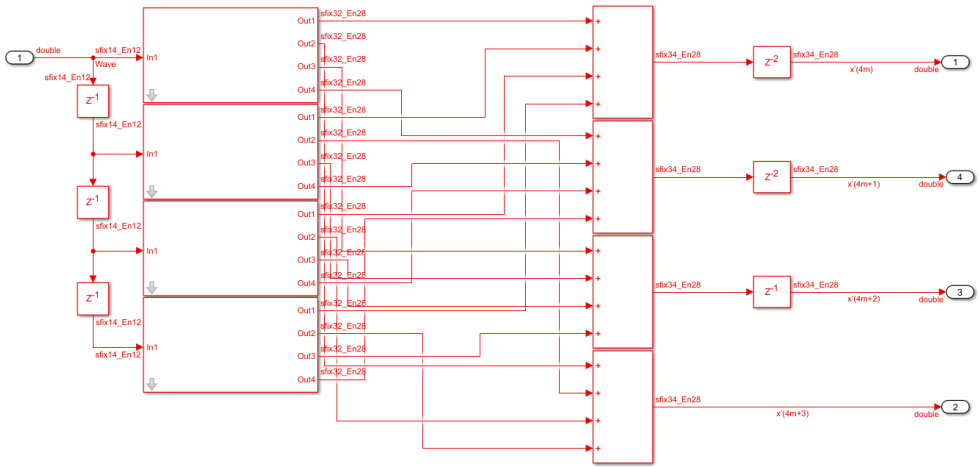


Figura 42: Modelo Simulink Filtro Paralelo $\boldsymbol{Px4}$ para $\boldsymbol{Q = 4/3}$

La ***Figura 43*** muestra las salidas obtenidas a partir del modelo Simulink.

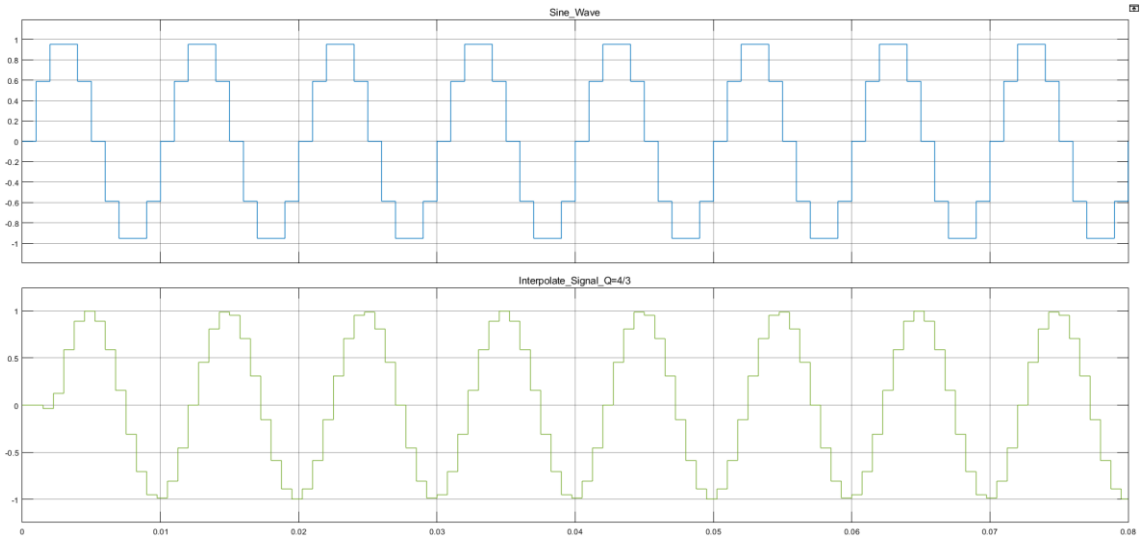


Figura 43: Señales de Entrada e Interpolada- Filtro Paralelizado $\boldsymbol{Px4}$ para $\boldsymbol{Q = 4/3}$

3.2.2 Diezmado por 3/4 paralelizado por Px4

Para este caso se procede de la misma manera que en el apartado anterior con la diferencia que a la salida se obtiene **3** líneas en paralelo, los coeficientes se toman de la **Tabla 9** y se reparten en cada subfiltro.

Para proceder al reordenamiento de las salidas del filtro paralelo nos quedarnos con una de cada 4 muestras (factor de diezmado), esto se observa en la ***Figura 44.***

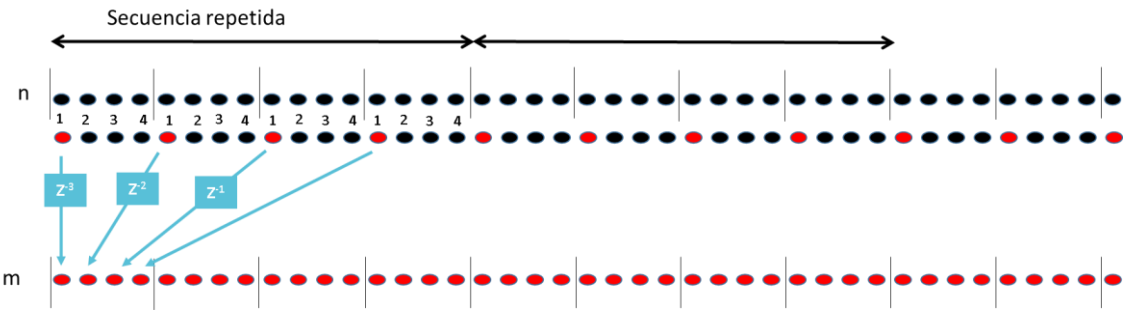


Figura 44: Reordenamiento de salidas para $\boldsymbol{3/4}$ paralelo por $\boldsymbol{Px4}$

Las salidas *3m*, *3m+1*, *3m+2* y *3m+3* se obtienen retardando únicamente la salida (*1*) del filtro paralelizado ***Figura 45*** de acuerdo al reordenamiento de las salidas mostrado en la ***Figura 44***.

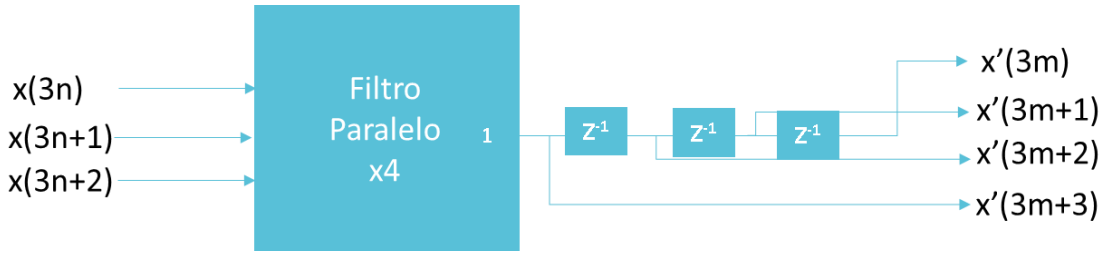


Figura 45: Diagrama de Bloques Filtro Paralelo $\boldsymbol{Px4}$ para $\boldsymbol{Q = 3/4}$

De la misma forma se implementó el modelo en Simulink ***Figura 46*** para comprobar su correcto funcionamiento, la ***Figura 47*** muestra las salidas obtenidas a partir del modelo.

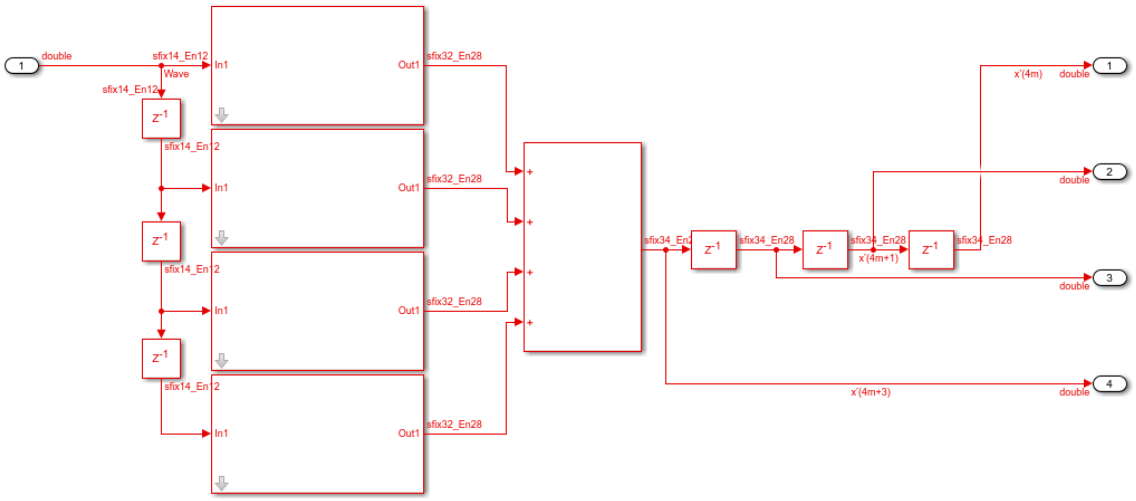


Figura 46: Modelo Simulink Filtro Paralelo $\boldsymbol{Px4}$ para $\boldsymbol{Q = 3/4}$

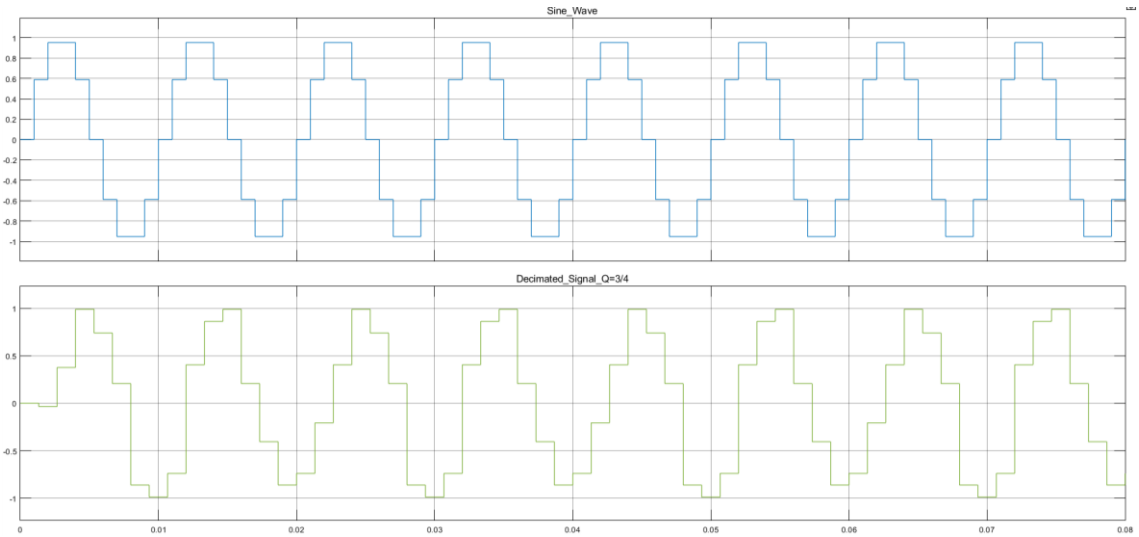


Figura 47: Señales de Entrada y Diezmada- Filtro Paralelo $\boldsymbol{Px4}$ para $\boldsymbol{Q = 3/4}$

El problema ahora radica en que para poder integrar este interpolador/diezmador en el circuito debería ser capaz de trabajar a una frecuencia de reloj ($\boldsymbol{f_{clk}} = 5\ GHz$). Lo que tampoco va a ser posible. Esto nos obliga a buscar una solución que nos permita trabajar a mayor frecuencia, siendo esta la paralelización del filtro

3.3 Paralelización de Filtros

La paralelización nos permite realizar las operaciones en un menor tiempo, es decir trabajar a una frecuencia de muestreo mayor que la de reloj ($f_s > f_{clk}$) y obtener una alta velocidad de procesamiento lo cual genera una mayor eficiencia. Como mencionamos anteriormente el procesado secuencial no nos permite alcanzar la velocidad requerida, por lo que nos vemos obligados a utilizar técnicas de paralelización que producirán un aumento considerable del hardware a utilizar.

El proceso de paralelización visto desde el punto del hardware consiste en replicar el hardware tantas veces como canales paralelos vayamos a utilizar, arreglado la conexión entre muestras y así lograr una velocidad de procesado mucho mayor.

El esquema del sistema mostrado en la ***Figura 1*** sufre ciertas modificaciones expuestas en la ***Figura 48*** donde a la entrada del sistema la señal se paraleliza (las muestras entran de $\boldsymbol{n}$ en $\boldsymbol{n}$) siendo $\boldsymbol{n}$ el número de las líneas paralelas $\boldsymbol{P_o}$ a la entrada del sistema, posteriormente se interpola por un factor de **2** (*filtro conformador de pulsos*).

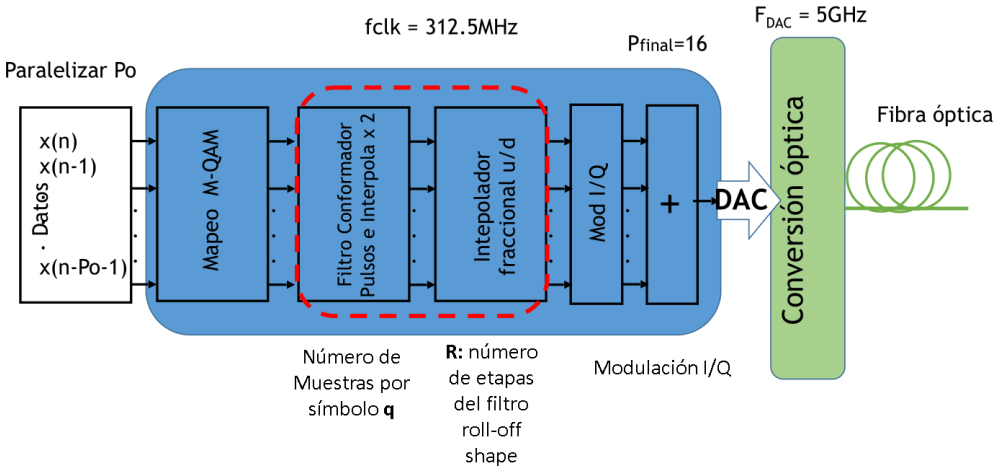


Figura 48: Modificaciones del Transmisor del Sistema

La relación entre el número de líneas paralelas a la entrada del sistema (paralelización de entrada) y el factor de interpolación aplicar en el cambio de tasa viene dado por la ecuación ***Ecu.* 17.**

$$P_o * 2 * \frac{L}{M} = 16 \qquad \textbf{Ecu. 17}$$

Donde $\boldsymbol{P_o}$es la paralelización inicial, $\boldsymbol{L}$ el factor de interpolación y $\boldsymbol{M}$ el factor de diezmado.

La ecuación se iguala a $\mathbf{16}$ que corresponde al número de canales con los cuales queremos trabajar en el sistema y así poder alcanzar los requerimientos de velocidad.

En los siguientes apartados se describe al proceso que se siguió para implementar la paralelización del filtro con valores de cambio de tasa de $\mathbf{4/3}$ y $\mathbf{8/7}$ para el transmisor, $\mathbf{3/4}$ y $\mathbf{7/8}$ para el receptor.

3.3.1 Interpolación por 4/3 paralelizado por Po12-P16

Primero con la ***Ecu.* 17** determinamos el valor de la paralelización inicial (número de líneas paralelas a la entrada del sistema) necesario para poder obtener los 16 canales con los que se requieren trabajar. Siendo $\boldsymbol{L} = 4$ (factor de interpolación) y $\boldsymbol{M} = 3$ (factor de diezmado) reemplazando obtenemos:

$$P_o * 2 * \frac{4}{3} = 16$$

$$4 * P_o = 8 * 3$$

$$P_o = \frac{24}{4} = \mathbf{6}$$

Conociendo el valor de $\boldsymbol{P_o}$ podemos determinar con cuantas de líneas paralelas $\boldsymbol{P}$ **(**paralelización de salida**)** vamos a trabajar luego de pasar por el interpolador por $\mathbf{2}$**.** Para este caso serán $\mathbf{12}$ líneas paralelas, la ***Figura 49*** nos muestra el diagrama.

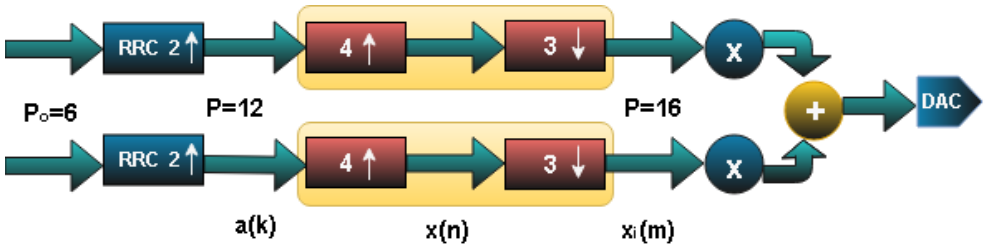


Figura 49: Diagrama Interpolación Fraccional por $\mathbf{4/3}$ –Paralelización $\boldsymbol{P_o} = \mathbf{12}$ - $\boldsymbol{P} = \mathbf{16}$

Una vez paralelizado se aplica el factor de interpolación (para este caso $\boldsymbol{x4}$) por lo que obtendremos a la salida $\mathbf{48}$ muestras. Para obtener estas muestras se crean $\mathbf{12}$ bancos de filtros polifásicos como muestra la ***Figura 50,*** a cada banco de filtros le entran los $\mathbf{12}$ canales y se obtienen cuatro salidas (correspondientes a las fases de la interpolación $\boldsymbol{x4}$).

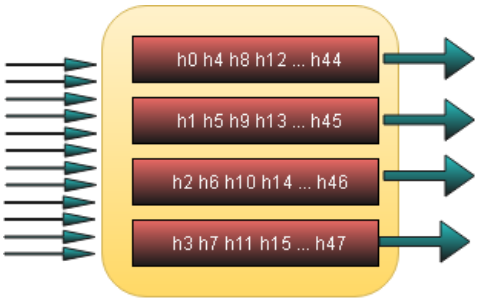


Figura 50: Diagrama Banco de Filtros para $\mathbf{4/3}$ –Paralelización $\boldsymbol{P_o} = \mathbf{12}$ - $\boldsymbol{P} = \mathbf{16}$

Para obtener las $\mathbf{16}$ muestras a la salida aplicamos el factor de diezmado (para este caso $\boldsymbol{x3}$), es decir nos quedaremos con una de cada tres muestras de los bancos de filtros, lo que recudirá el número de salidas de $\mathbf{48}$ a $\mathbf{16}$**.**

Este proceso nos permite reducir la cantidad de hardware empleado al momento de codificarlo en lenguaje Verilog, ya que únicamente identificamos los filtros necesarios para obtener las salidas requeridas ayudándonos a reducir el número de recursos empleados. La ***Figura 51*** muestra los subfiltros necesarios de cada banco de filtros para obtener las $\mathbf{16}$ señales.

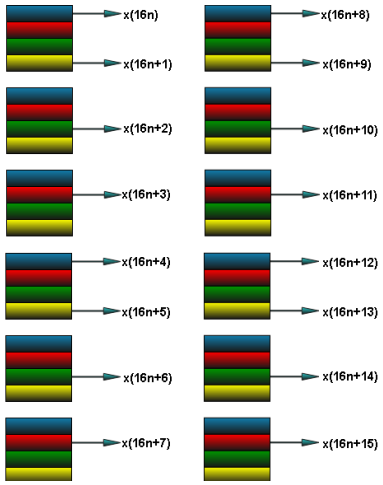


Figura 51: Subfiltros–Paralelización $\boldsymbol{P_o} = \mathbf{12}$ - $\boldsymbol{P} = \mathbf{16}$ para $\mathbf{4/3}$

Cada caja de colores (azul, rojo, verde y amarrillo) de la ***Figura 51*** corresponde a un banco de filtros como el mostrado en la ***Figura 50.***

Al momento de armar el modelo en Simulink nos percatamos que al primer banco de filtros le entran las muestras correspondientes a cada uno los $\mathbf{12}$ canales, para el segundo banco de filtros la muestra de entrada $\boldsymbol{x(12n-12)}$ no es más que la que tenemos en el *canal 1* con un delay, en el tercer banco de filtros las dos últimas muestras de entradas son las del *canal 1* y *2* con un delay cada una y así se procede para cada banco de filtros hasta llegar al doceavo, esto se observa en la ***Figura 52.***

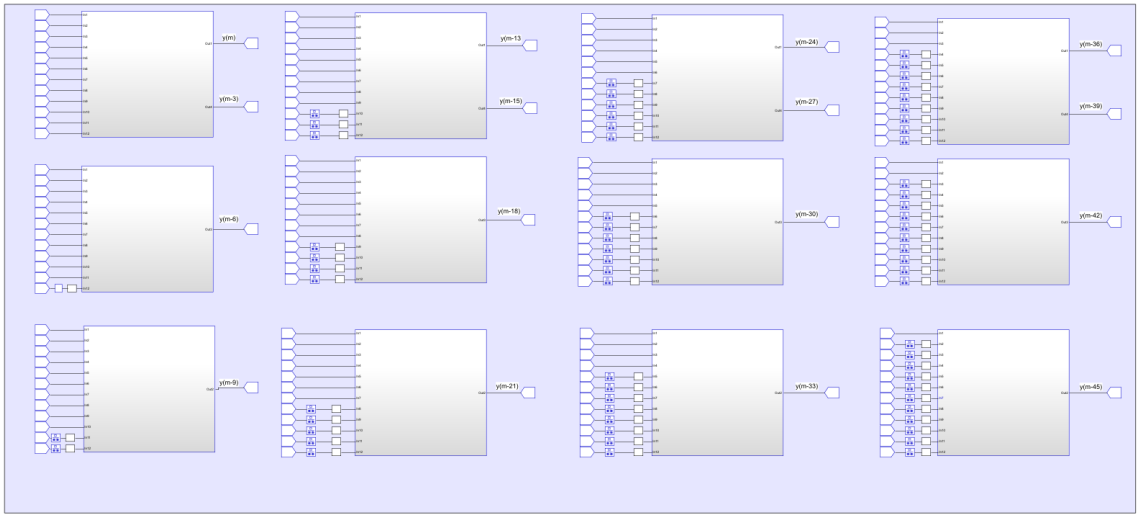


Figura 52: Modelo Simulink Subfiltros-Paralelización $\boldsymbol{P_o = 12}$ - $\boldsymbol{P = 16}$ para $\mathbf{4/3}$

La estructura interna de cada banco de filtros es similar a la mostrada en la ***Figura 18*** y la estructura interna de los subfiltros son similares a la ***Figura 19***.

En el apartado **2.2.1** se mencionó que los coeficientes del filtro se pueden calcular con la ayuda de *fdatool,* siempre que se adapten a las necesidades de paralelización que tenemos ahora, siendo ***48*** el número de coeficientes necesarios para este caso. Los coeficientes calculados se muestran en la ***Tabla 12.***

Tabla 12: Coeficientes Filtro Interpolador (fdatool) para $\mathbf{4/3}$

s(n)	s(n-1)	s(n-2)	s(n-3)	s(n-4)
1	-3,3885E-05	-1,1073E-04	-1,7906E-04	-9,1127E-05
2	2,9976E-04	9,1640E-04	1,2779E-03	6,4905E-04
3	-0,0013	-0,0040	-0,0051	-0,0026
4	0,0043	0,0124	0,0155	0,0078
5	-0,0114	-0,0335	-0,0425	-0,0225
6	0,0327	0,1128	0,1925	0,2422
7	0,2422	0,1925	0,1128	0,0327
8	-0,0225	-0,0425	-0,0335	-0,0114
9	0,0078	0,0155	0,0124	0,0043
10	-0,0026	-0,0051	-0,0040	-0,0013
11	6,4905E-04	1,2779E-03	9,1640E-04	2,9976E-04
12	-9,1127E-05	-1,7906E-04	-1,1073E-04	-3,3885E-05

La ***Figura 53*** muestra las salidas obtenidas al aplicar los coeficientes con una ganancia de $\mathbf{4}$.

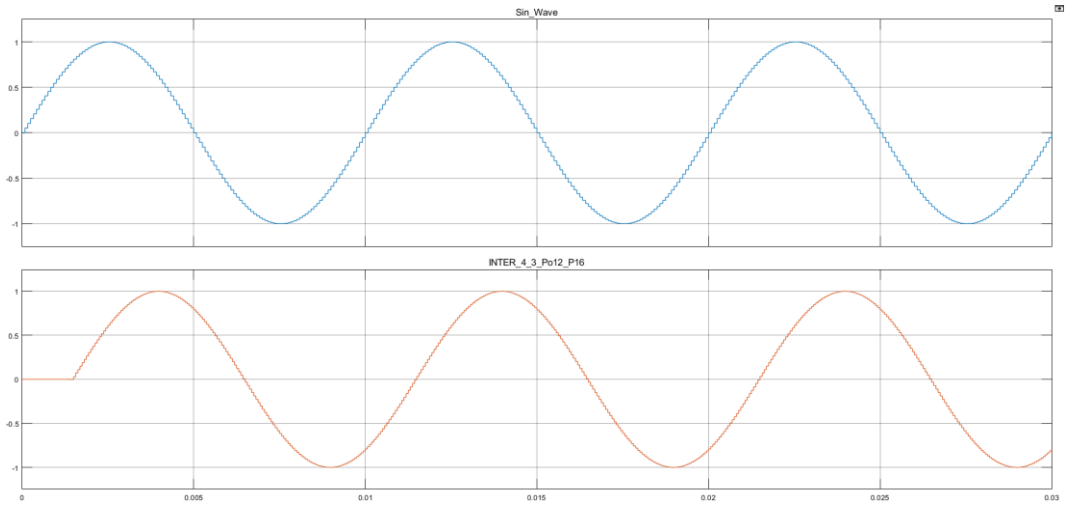


Figura 53: Señales de Entrada y Salida Filtro (*fdatool*)- $\boldsymbol{P_o = 12}$ - $\boldsymbol{P = 16}$ para $\mathbf{4/3}$

3.3.1.1 Verificación Interpolador por 4/3

Una vez implementado el interpolador en Simulink y verificado su funcionamiento, este se codifica en Quartus y se procede al proceso de verificación, comparando las señales de referencia generadas en Matlab-Simulink con las señales del módulo codificado en Verilog.

Para proceder a la verificación del modelo se crea un archivo codificado en Verilog denominado Test Bench, el cual contiene todos los módulos codificados y le proporciona los estímulos necesarios (señal de reloj, habilitación, reset y señales de entrada).

Un Test Bench consta de cuatro procesos esenciales:

1. El proceso de instanciación de los módulos que componen el interpolador.
2. El proceso de lectura de las muestras de la señal de entrada,
3. El proceso de captura de las señales de salida.
4. El proceso de comparación, donde se valora si los datos obtenidos del módulo codificado son los mismos que los datos referenciales generados por MATLAB.

El interpolador consta de 16 bancos de subfiltros, pero se presenta los resultados de la simulación para uno de ellos. donde se observa la señal de reloj (***clk***), la salida del modelo codificado en Verilog (***S_M***), la salida del fichero exportado de Matlab(***S_F***), el contador de errores (***error_cnt***) y el contador de muestras simuladas (***sample_cnt***). La ***Figura 54*** muestra el diagrama de verificación empleado.

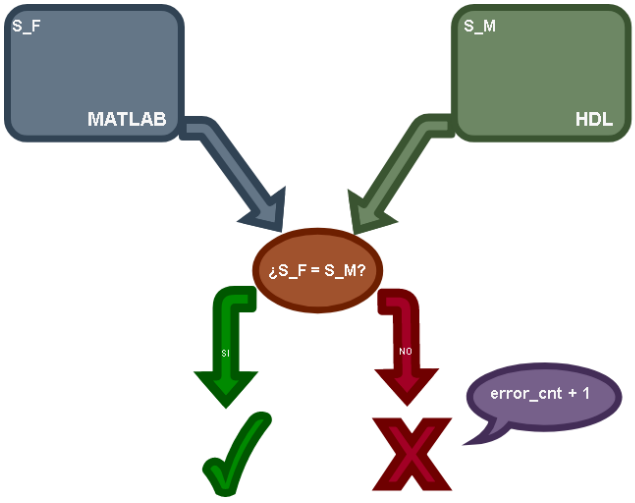


Figura 54: Diagrama de Verificación Banco de Filtros

La ***Figura 55*** muestra el resultado de la verificación del banco de pruebas.

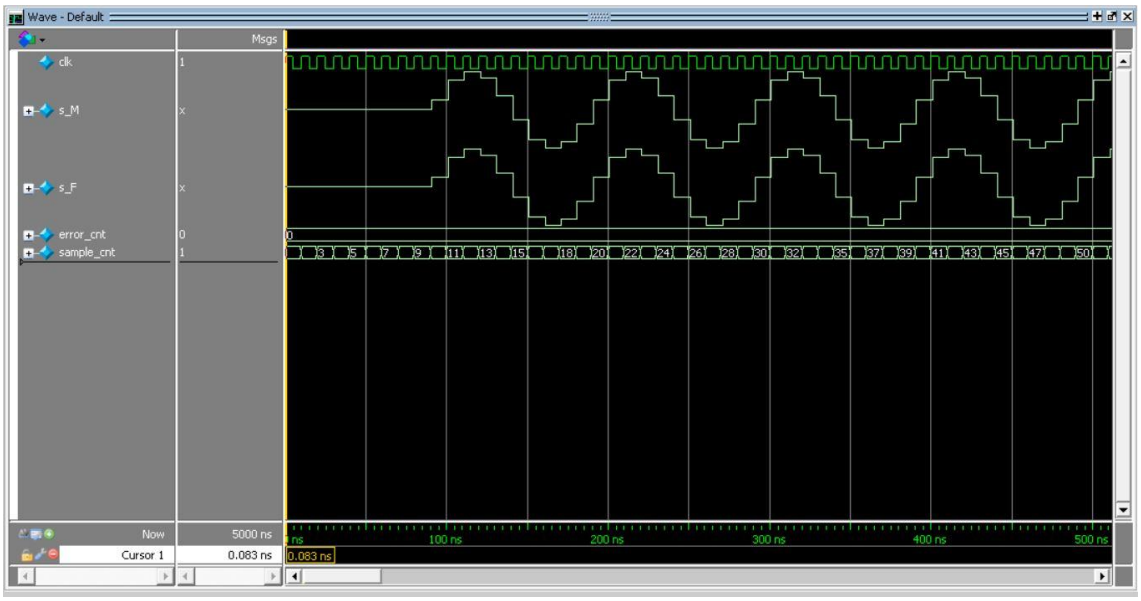


Figura 55: TestBench Banco de Filtros Interpolador por $\mathbf{4/3}$-Paralelización $\boldsymbol{P_o = 12}$-$\boldsymbol{P = 16}$

Una vez verificado el funcionamiento del modelo HDL, se procede a obtener la frecuencia máxima de funcionamiento con la ayuda de ***Time Quest Timing Analyzer***, es importante mencionar que el circuito debe estar segmentado al máximo para poder reducir el camino crítico y así aumentar la velocidad de funcionamiento. La ***Figura 56*** muestra el diagrama RTL de uno de los filtros del banco, donde se observa los sumadores en árbol totalmente segmentados, al igual que todas las multiplicaciones se encuentran registradas.

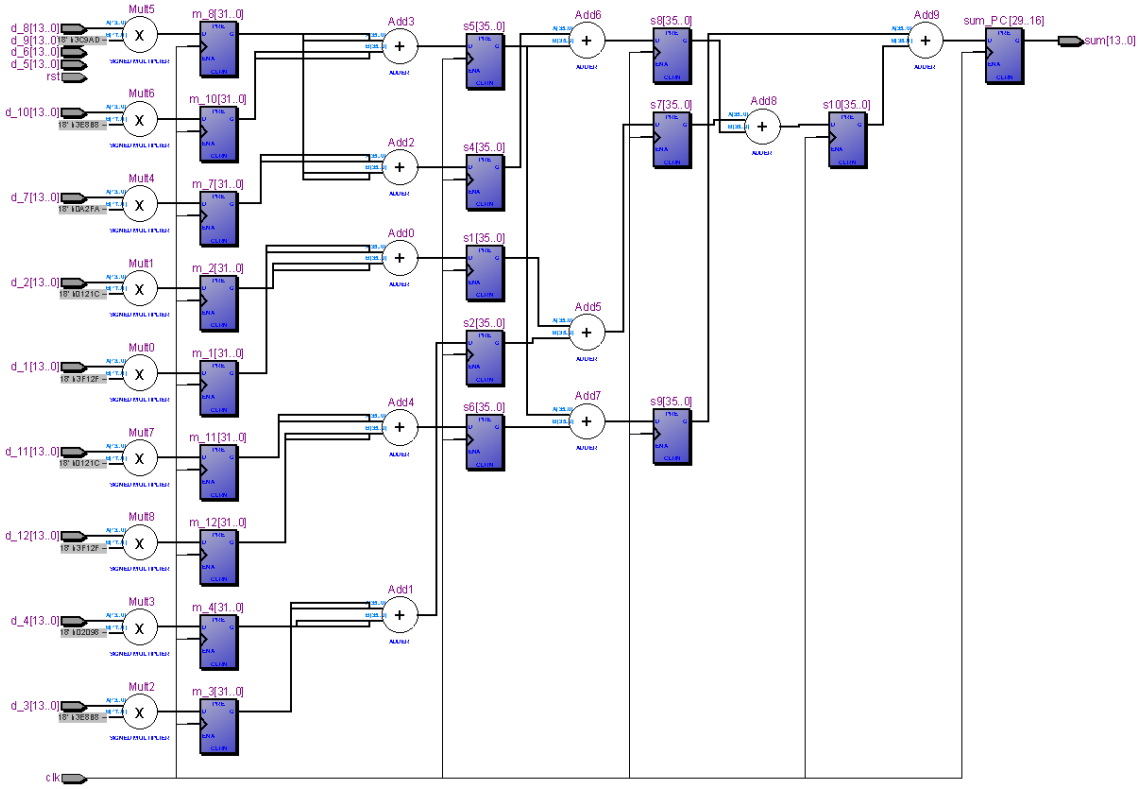


Figura 56: Diagrama RTL Interpolador por $\mathbf{4/3}$-Paralelización $\boldsymbol{P_o = 12}$ - $\boldsymbol{P = 16}$

La ***Figura 57*** muestra que el valor obtenido de frecuencia supera el valor requerido por cada canal de ***312.5 MHz.***

Fmax Summary

	Fmax	Restricted Fmax	Clock Name
1	336.25 MHz	336.25 MHz	clk

Figura 57: Frecuencia Máxima de Funcionamiento Interpolador por $4/3$

Los recursos empleados para implementar el filtro interpolador constan en la ***Tabla 13***. Intel define un ***DSP*** (*Digital Signal Processing*) como un bloque de Procesamiento Digital de Señales ***[7]*** y un ***ALM*** (*Adaptative Logic Module*) como un módulo lógico adaptativo con 8 entradas ***[8].***

Tabla 13: Recursos empleados Interpolador por $4/3$

Recurso	Cantidad
Registros	5008
DSP	96
ALM	2979

Para los siguientes apartados el proceso a seguir es el mismo, lo que van cambiando son los diagramas y algunos cálculos los cuales se irán mencionando en cada uno de ellos

3.3.2 Diezmado por 3/4 paralelizado por Po16-P12

Este interpolador corresponde al receptor del sistema, el proceso es inverso al descrito en el apartado anterior, es decir tendremos una paralelización de entrada por 16 y a la salida obtendremos 12 señales como se observa en la ***Figura 58***

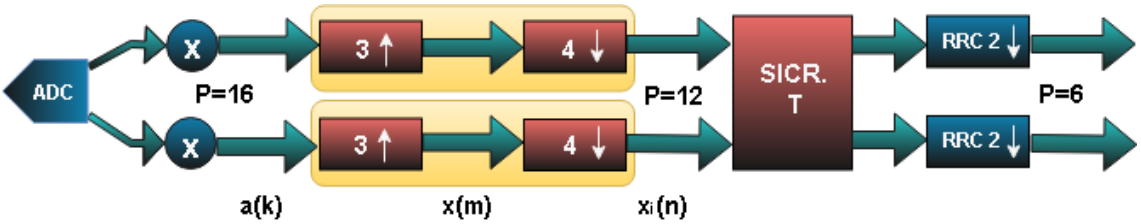


Figura 58: Diagrama Diezmador Fraccional por $3/4$ –Paralelización $P_o = 16$ - $P = 12$

Se aplica el factor de interpolación (para este caso $x3$) por lo que obtendremos a la salida 48 m muestras, para obtener estas muestras se crean 16 bancos de filtros polifásicos como muestra la ***Figura 59,*** a cada banco de filtros le entran los 16 canales y se obtienen tres salidas (correspondientes a las fases de la interpolación $x3$).

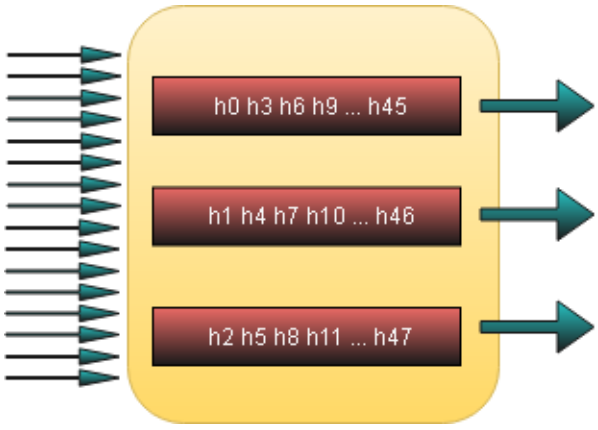


Figura 59: Diagrama Banco de Filtros para $3/4$ –Paralelización $P_o = 16$ - $P = 12$

La ***Figura 60*** muestra los subfiltros necesarios de cada banco de filtros para obtener las 12 señales a la salida. Aplicamos el factor de diezmado (para este caso $x4$) para quedaremos con una de cada cuatro muestras de los bancos de filtros, lo que reduce el número de salidas de 48 a 12.

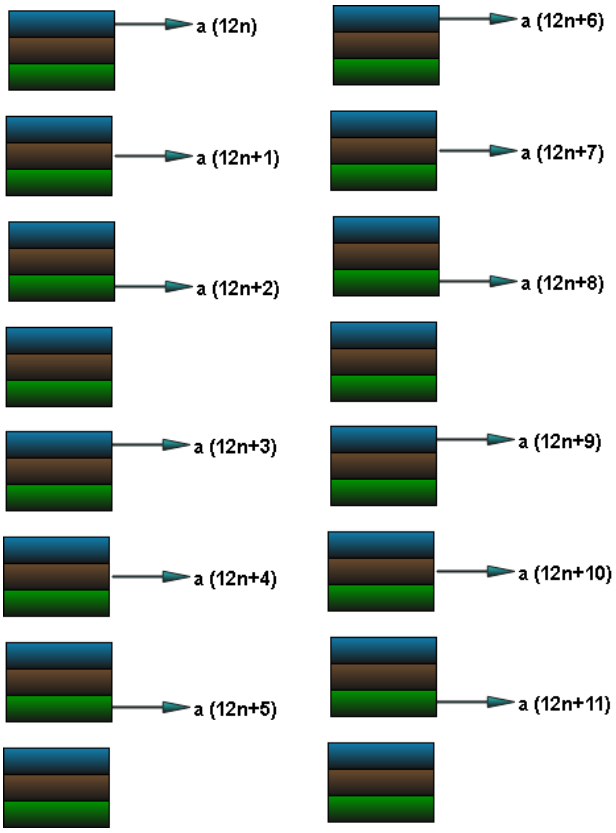


Figura 60: Subfiltros–Paralelización $\boldsymbol{P_o = 16}$ - $\boldsymbol{P = 12}$ para $\mathbf{3/4}$

El modelo Simulink implementado para este caso se muestra en la ***Figura 61***, donde al primer banco de filtros le entran las muestras correspondientes a los $\mathbf{16}$ canales, al segundo banco de filtros la muestra de entrada $\boldsymbol{x(16n-16)}$ no es más que la que tenemos en el *canal 1* con un delay, en el tercer banco de filtros las dos últimas muestras de entradas son las del *canal 1* y *2* con un delay cada una y así se procede para cada banco de filtros hasta llegar al dieciseisavo filtro.

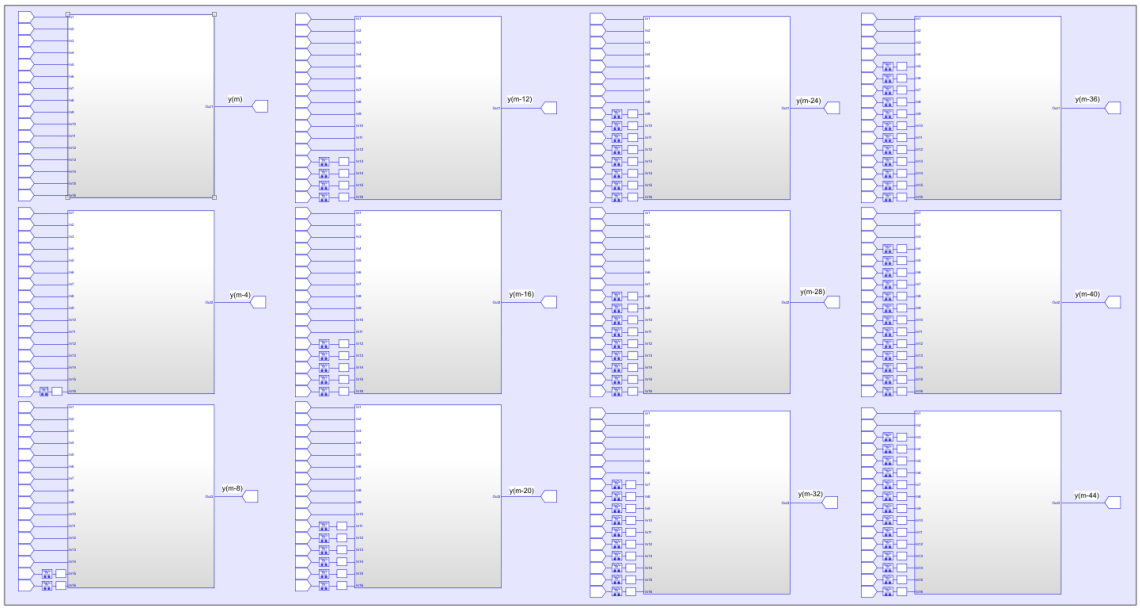


Figura 61: Modelo Simulink Subfiltros-Paralelización $\boldsymbol{P_o = 16}$ - $\boldsymbol{P = 12}$ para $\mathbf{3/4}$

Los coeficientes del filtro son los mismos que se calcularon en la sección anterior pero repartidos de la siguiente manera ***Tabla 14***.

Tabla 14: Coeficientes Filtro Diezmador (fdatool) para $3/4$

s(n)	s(n-1)	s(n-2)	s(n-3)
1	-3,3885E-05	-1,1073E-04	-1,7906E-04
2	-9,1127E-05	2,9976E-04	9,1640E-04
3	0,0013	6,4905E-04	-0,0013
4	-0,0040	-0,0051	-0,0026
5	0,0043	0,0124	0,0155
6	0,0078	-0,0114	-0,0335
7	-0,0425	-0,0225	0,0327
8	0,1128	0,1925	0,2422
9	0,2422	0,1925	0,1128
10	0,0327	-0,0225	-0,0425
11	-0,0335	-0,0114	0,0078
12	0,0155	0,0124	0,0043
13	-0,0026	-0,0051	-0,0040
14	-0,0013	6,4905E-04	0,0013
15	9,1640E-04	2,9976E-04	-9,1127E-05
16	-1,7906E-04	-1,1073E-04	-3,3885E-05

La ***Figura 62*** muestra las salidas obtenidas al aplicar los coeficientes mostrados en la ***Tabla 14*** con una ganancia de 3.

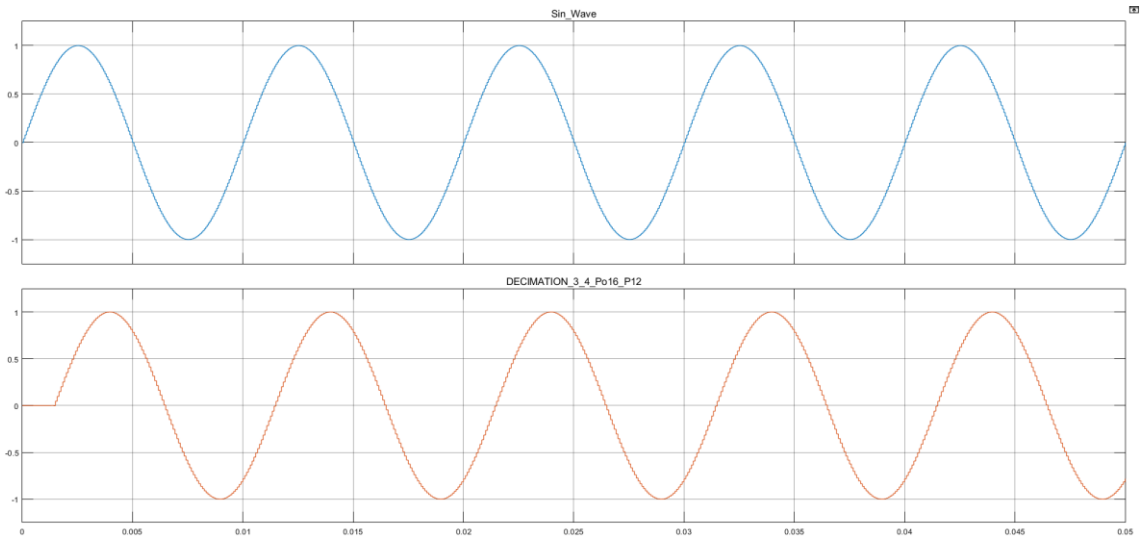


Figura 62: Señales de Entrada y Salida Filtro (*fdatool*)- $P_o = 16$ - $P = 12$ para $3/4$

3.3.2.1 Verificación diezmador por 3/4

La ***Figura 63*** muestra que el valor obtenido de frecuencia supera el valor requerido por cada canal de ***312.5 MHz.***

Fmax Summary

	Fmax	Restricted Fmax	Clock Name
1	315.26 MHz	315.26 MHz	clk

Figura 63: Frecuencia Máxima de Funcionamiento Diezmador por $3/4$

La ***Figura 64*** muestra el resultado de la simulación para uno de los bancos de filtros del diezmador.

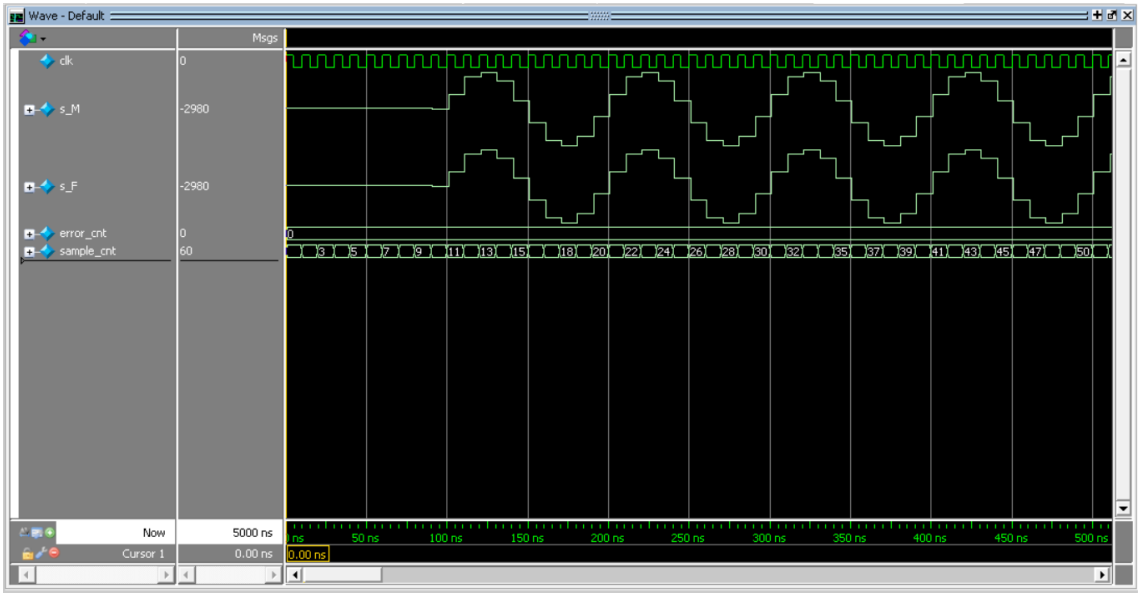


Figura 64: TestBench Banco de Filtros Diezmador por $\mathbf{3/4}$-Paralelización $\boldsymbol{P_o = 16}$ - $\boldsymbol{P = 12}$

La ***Figura 65*** muestra el diagrama RTL del banco de filtros, donde se observa los sumadores en árbol totalmente segmentados, de la misma manera todas las multiplicaciones se encuentran registradas.

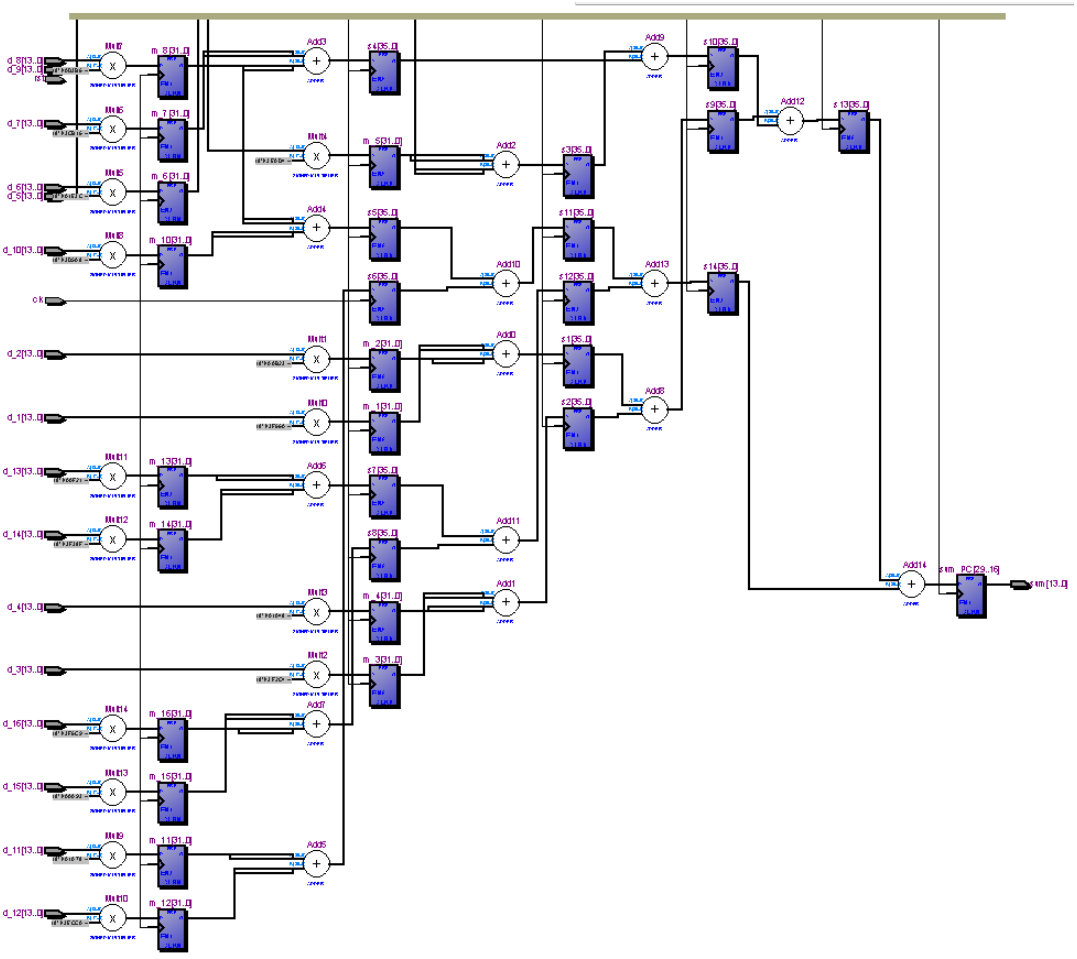


Figura 65; Diagrama RTL Diezmador por $\mathbf{3/4}$-Paralelización $\boldsymbol{P_o = 16}$ - $\boldsymbol{P = 12}$

Los recursos empleados para implementar el filtro interpolador constan en la ***Tabla 15.***

Tabla 15: Recursos empleados Diezmador por $3/4$

Recurso	Cantidad
Registros	5572
DSP	108
ALM	2947

3.3.3 Interpolación por 8/7 paralelizado por Po14-P16

Siendo $\boldsymbol{L} = 8$ (factor de interpolación) y $\boldsymbol{M} = 7$ (factor de diezmado) reemplazando estos valores en la $\boldsymbol{Ecu.17}$ obtenemos:

$$P_o * 2 * \frac{8}{7} = 16$$

$$\cancel{8} * P_o = \cancel{8} * 7$$

$$P_o = \mathbf{7}$$

Conociendo el valor de $\boldsymbol{P_o}$ se determinan las líneas paralelas a trabajar luego de pasar por el interpolador por $\mathbf{2}$. La ***Figura 66*** nos muestra el diagrama, para este caso serán $\mathbf{14}$ líneas paralelas.

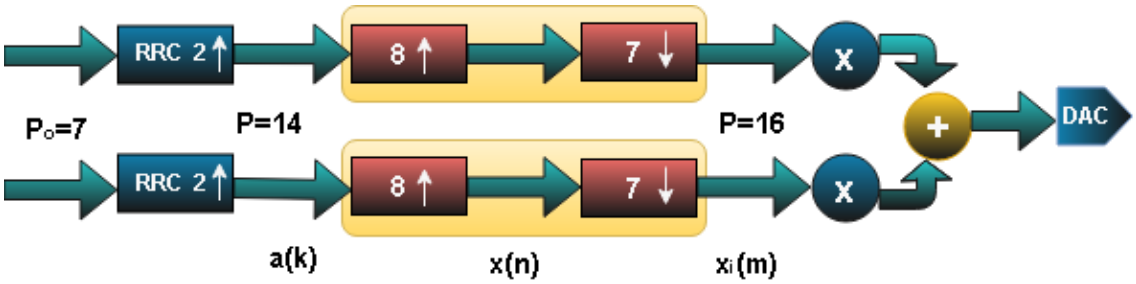


Figura 66: Diagrama Interpolación Fraccional por $8/7$ –Paralelización $\boldsymbol{P_o = 14}$ - $\boldsymbol{P = 16}$

Se aplica el factor de interpolación (para este caso $\boldsymbol{x8}$) por lo que obtendremos a la salida $\mathbf{112}$ muestras. Para obtener estas muestras se crean $\mathbf{14}$ bancos de filtros polifásicos como muestra la ***Figura 67,*** a cada banco de filtros le entran los $\mathbf{14}$ canales y se obtienen ocho salidas (correspondientes a las fases de la interpolación $\boldsymbol{x8}$).

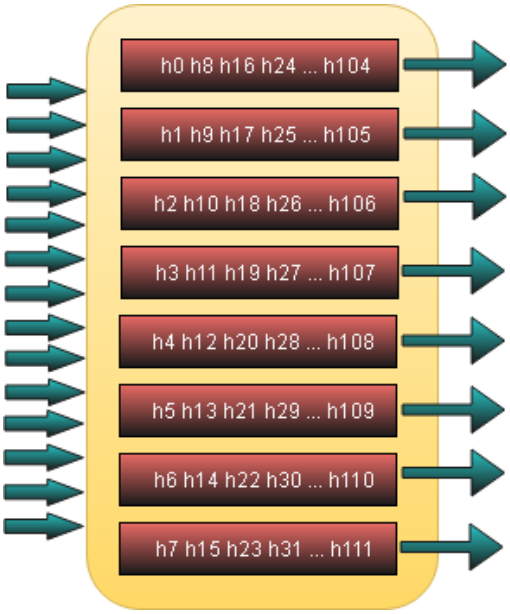


Figura 67: Diagrama Banco de Filtros para $8/7$ –Paralelización $\boldsymbol{P_o = 14}$ - $\boldsymbol{P = 16}$

Para obtener las **16** muestras a la salida aplicamos el factor de diezmado (para este caso $x7$), es decir nos quedaremos con una de cada siete muestras de los bancos de filtros, lo que recudirá el número de salidas de **112** a **16.**

Este proceso nos permite identificar los filtros necesarios para obtener las salidas requeridas ayudándonos a reducir el número de recursos empleados. La **Figura 68** muestra los subfiltros necesarios de cada banco de filtros para obtener las**16** señales.

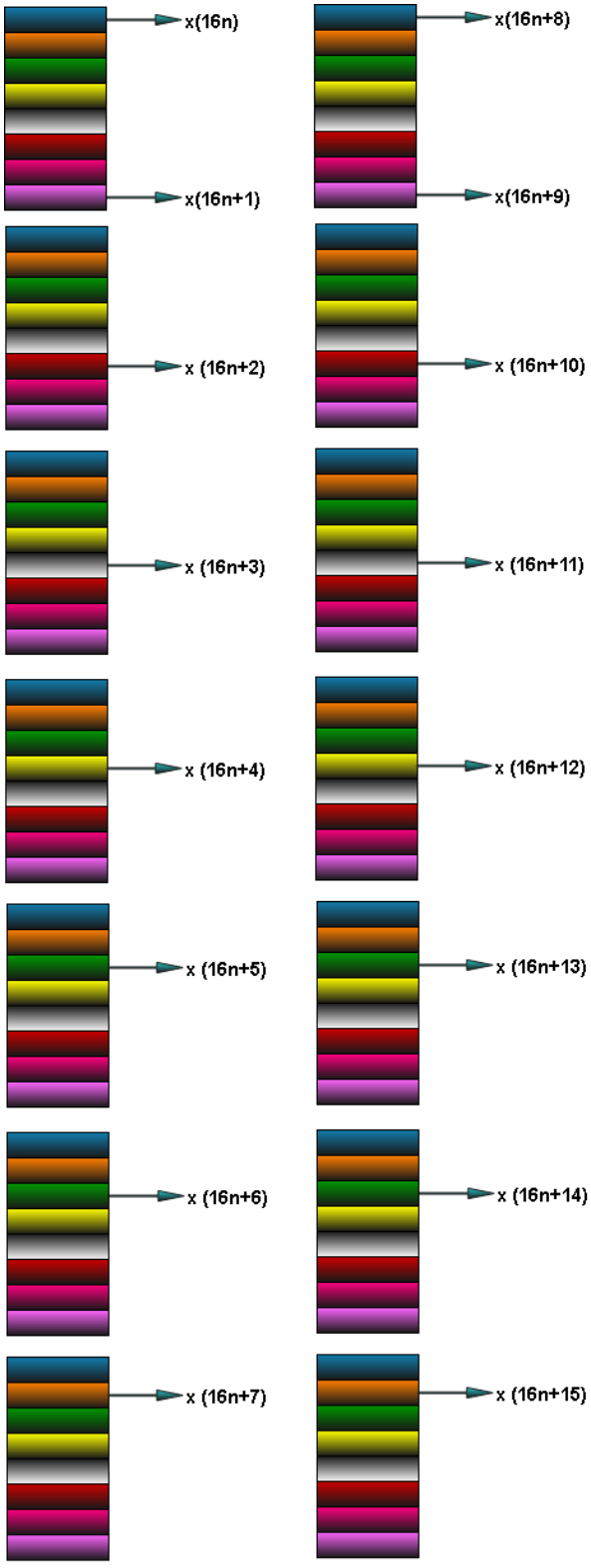


Figura 68: Subfiltros–Paralelización $P_o = 14$ - $P = 16$ para $8/7$

Se arma el modelo en Simulink, al primer banco de filtros le entran las muestras correspondientes a los $\mathbf{14}$ canales, para el segundo banco de filtros la muestra de entrada $\boldsymbol{x(14n-14)}$ es la muestra *canal 1* con un delay, en el tercer banco de filtros las dos últimas muestras de entradas son las del *canal 1* y *2* con un delay cada una y así se procede para cada banco de filtros hasta llegar al catorceavo, esto se observa en la ***Figura 69.***

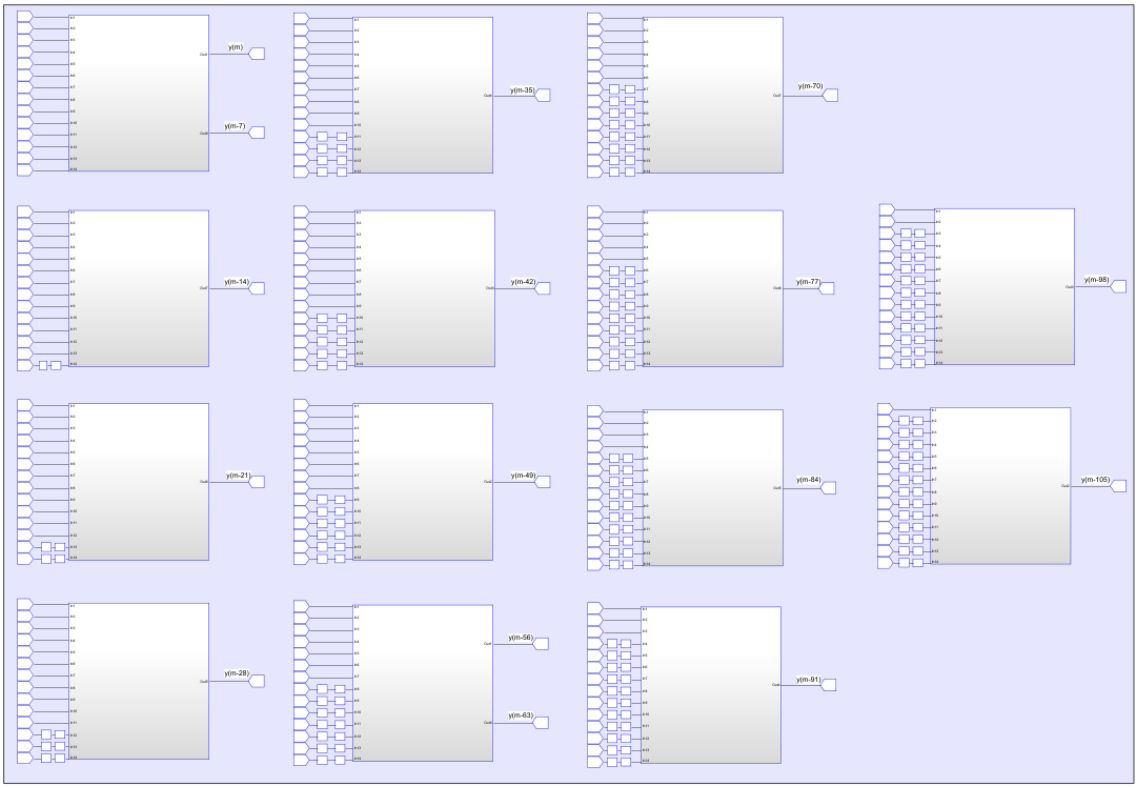


Figura 69: Modelo Simulink Subfiltros-Paralelización $\boldsymbol{P_o = 14}$ - $\boldsymbol{P = 16}$ para $\mathbf{8/7}$

Los coeficientes empleados se muestran en la ***Tabla 16***

Tabla 16: Coeficientes Filtro Interpolador (*fdatool*) para $\mathbf{8/7}$

s(n)	s(n-1)	s(n-2)	s(n-3)	s(n-4)	s(n-5)	s(n-6)	s(n-7)	s(n-8)
1	6,3495E-07	1,8053E-06	3,8751E-06	6,7470E-06	9,8578E-06	1,1975E-05	1,1181E-05	5,1278E-06
2	-8,3635E-06	-3,0369E-05	-5,9745E-05	-9,2112E-05	-1,1933E-04	-1,2994E-04	-1,1078E-04	-5,0115E-05
3	5,8337E-05	2,1089E-04	3,9022E-04	5,6384E-04	6,8604E-04	7,0402E-04	5,6839E-04	2,4677E-04
4	-2,6183E-04	-9,1520E-04	-0,0016	-0,0022	-0,0026	-0,0026	-0,0020	-8,4418E-04
5	8,7602E-04	0,0030	0,0051	0,0069	0,0078	0,0076	0,0058	0,0023
6	-0,0025	-0,0082	-0,0139	-0,0187	-0,0213	-0,0206	-0,0159	-0,0065
7	0,0073	0,0251	0,0454	0,0668	0,0872	0,1047	0,1175	0,1243
8	0,1243	0,1175	0,1047	0,0872	0,0668	0,0454	0,0251	0,0073
9	-0,0065	-0,0159	-0,0206	-0,0213	-0,0187	-0,0139	-0,0082	-0,0025
10	0,0023	0,0058	0,0076	0,0078	0,0069	0,0051	0,0030	8,7602E-04
11	-8,4418E-04	-0,0020	-0,0026	-0,0026	-0,0022	-0,0016	-9,1520E-04	-2,6183E-04
12	2,4677E-04	5,6839E-04	7,0402E-04	6,8604E-04	5,6384E-04	3,9022E-04	2,1089E-04	5,8337E-05
13	-5,0115E-05	-1,1078E-04	-1,2994E-04	-1,1933E-04	-9,2112E-05	-5,9745E-05	-3,0369E-05	-8,3635E-06
14	5,1278E-06	1,1181E-05	1,1975E-05	9,8578E-06	6,7470E-06	3,8751E-06	1,8053E-06	6,3495E-07

La ***Figura 70*** muestra las salidas obtenidas al aplicar los coeficientes con una ganancia de $\mathbf{8}$.

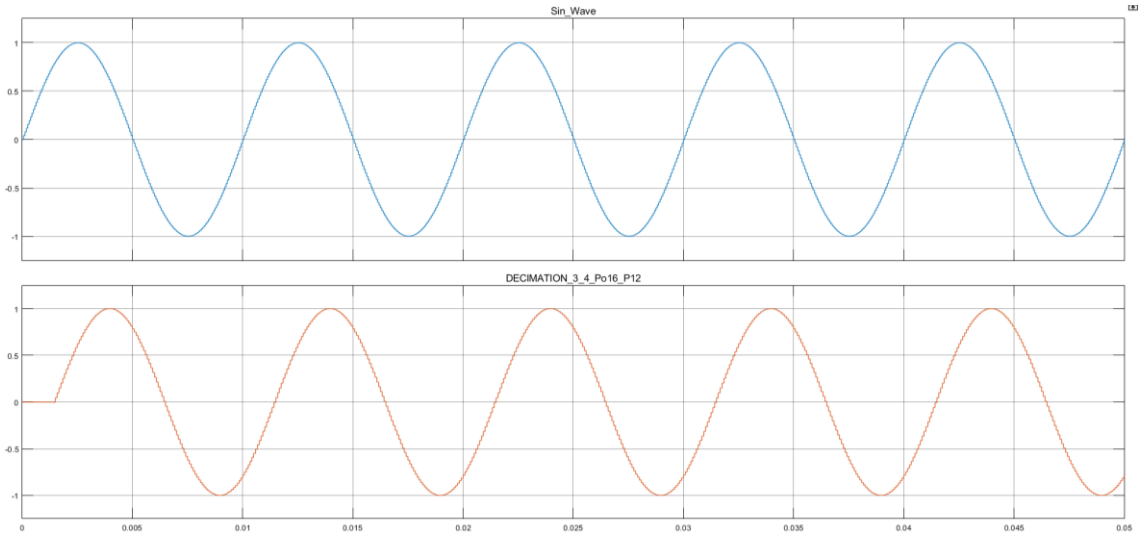


Figura 70: Señales de Entrada y Salida Filtro (*fdatool*)- $P_o = 14$ - $P = 16$ para $8/7$

3.3.3.1 *Verificación Interpolador por 8/7*

El resultado de la simulación para uno de los bancos de filtros del interpolados se muestra en la ***Figura 71.***

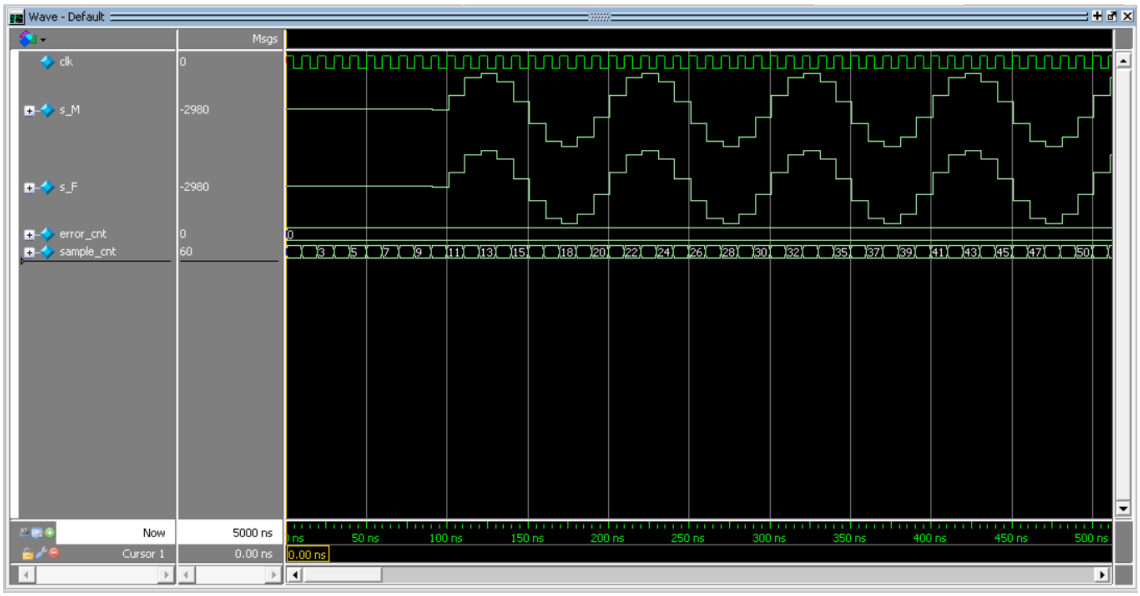


Figura 71: TestBench Banco de Filtros Interpolador por $8/7$-Paralelización $P_o = 14$-$P = 16$

La ***Figura 72*** muestra que el valor obtenido de frecuencia supera el valor requerido por cada canal de ***312.5 MHz.***

Fmax Summary

	Fmax	Restricted Fmax	Clock Name
1	326.05 MHz	326.05 MHz	clk

Figura 72: Frecuencia Máxima de Funcionamiento Interpolador por $8/7$

La ***Figura 73*** muestra el diagrama RTL del banco de filtros, donde se observa los sumadores en árbol totalmente segmentados, de la misma manera todas las multiplicaciones se encuentran registradas.

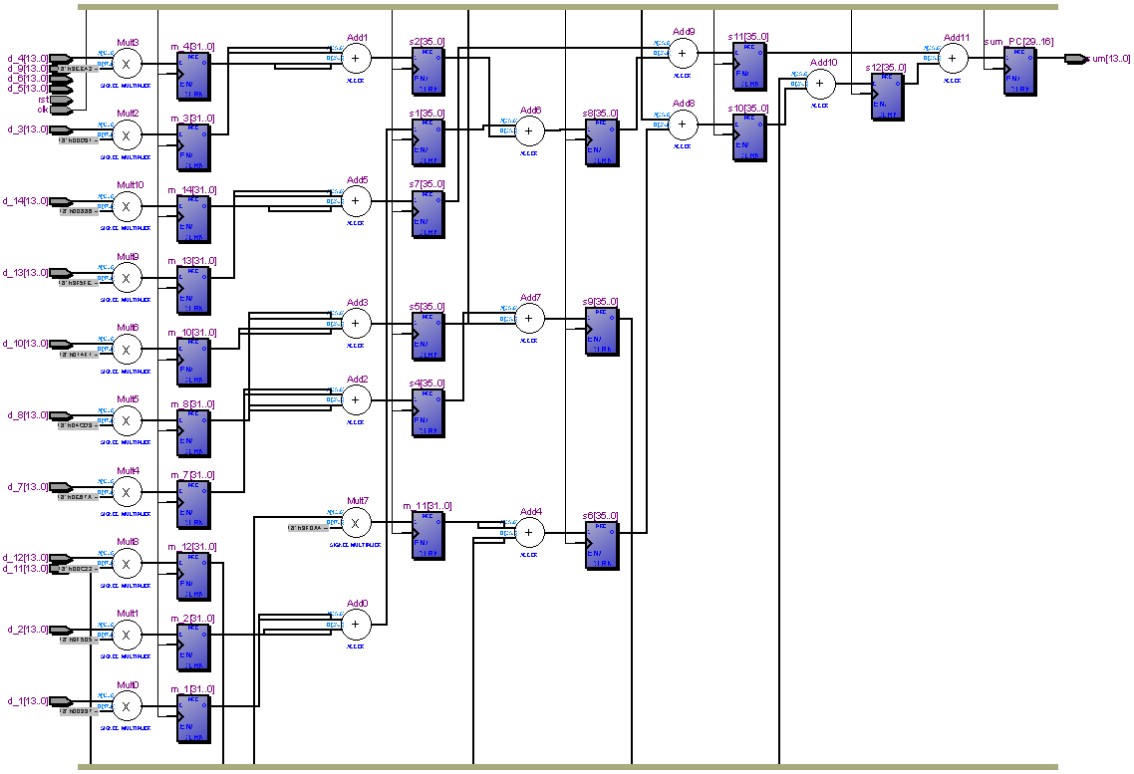


Figura 73; Diagrama RTL Diezmador por $8/7$-Paralelización $P_o = 14$ - $P = 16$

Los recursos empleados para implementar el filtro interpolador constan en la ***Tabla 17.***

Tabla 17: Recursos empleados Interpolador por $8/7$

Recurso	Cantidad
Registros	5993
DSP	112
ALM	3408

3.3.4 Diezmado por 7/8 paralelizado por Po16-P14

Al corresponde este interpolador al receptor del sistema, el proceso es inverso al descrito en el apartado anterior, es decir tendremos una paralelización de entrada por 16 y a la salida obtendremos 14 señales como se observa en la ***Figura 74***.

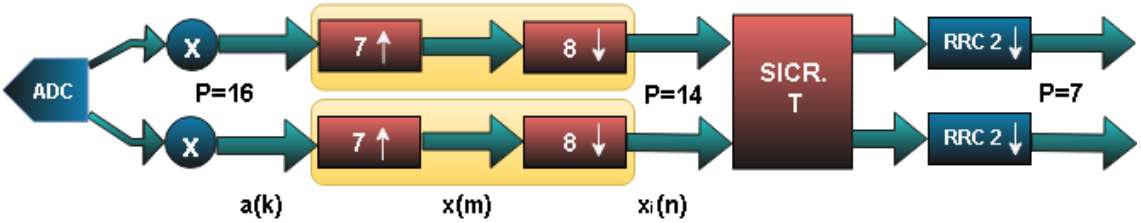


Figura 74: Diagrama Diezmado Fraccional por $7/8$ –Paralelización $P_o = 16$ - $P = 14$

Se aplica el factor de interpolación (para este caso $x7$) por lo que obtendremos a la salida 112 muestras, para obtener estas muestras se crean 16 bancos de filtros polifásicos como muestra la ***Figura 75***, a cada banco de filtros entran los 16 canales y se obtienen siete salidas (correspondientes a las fases de la interpolación $x7$).

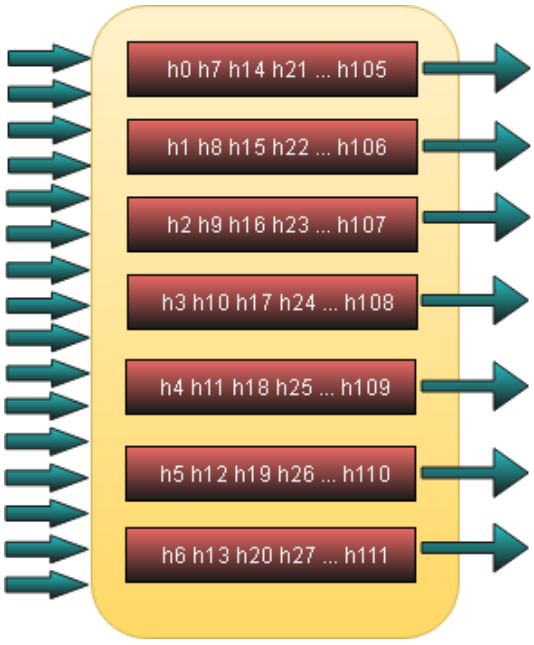
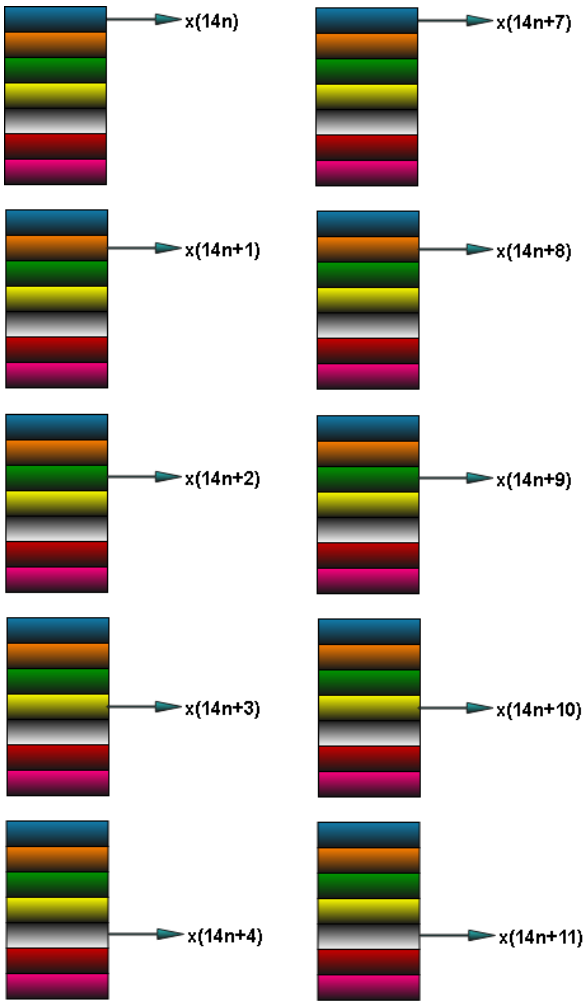


Figura 75: Diagrama Banco de Filtros para $7/8$ –Paralelización $P_o = 16$ - $P = 14$

La ***Figura 76*** muestra los subfiltros necesarios de cada banco de filtros para obtener las 14 señales a la salida. Aplicamos el factor de diezmado (para este caso $x8$) para quedaremos con una de cada ocho muestras de los bancos de filtros, lo que reduce el número de salidas de 112 a 14.



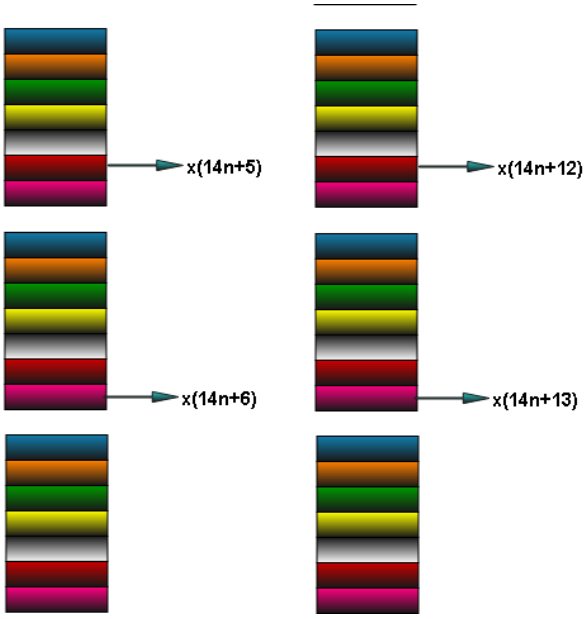


Figura 76: Subfiltros–Paralelización $\boldsymbol{P_o = 16}$ - $\boldsymbol{P = 14}$ para $\mathbf{7/8}$

En el modelo Simulink, al primer banco de filtros le entra las muestras correspondientes a los $\mathbf{16}$ canales, para el segundo banco de filtros la muestra de entrada $\boldsymbol{x(16n - 16)}$ es la muestra *canal 1* con un delay, en el tercer banco de filtros las dos últimas muestras de entradas son las del *canal 1* y *2* con un delay cada una y así se procede para cada banco de filtros hasta llegar al dieciseisavo, esto se observa en la ***Figura 77.***

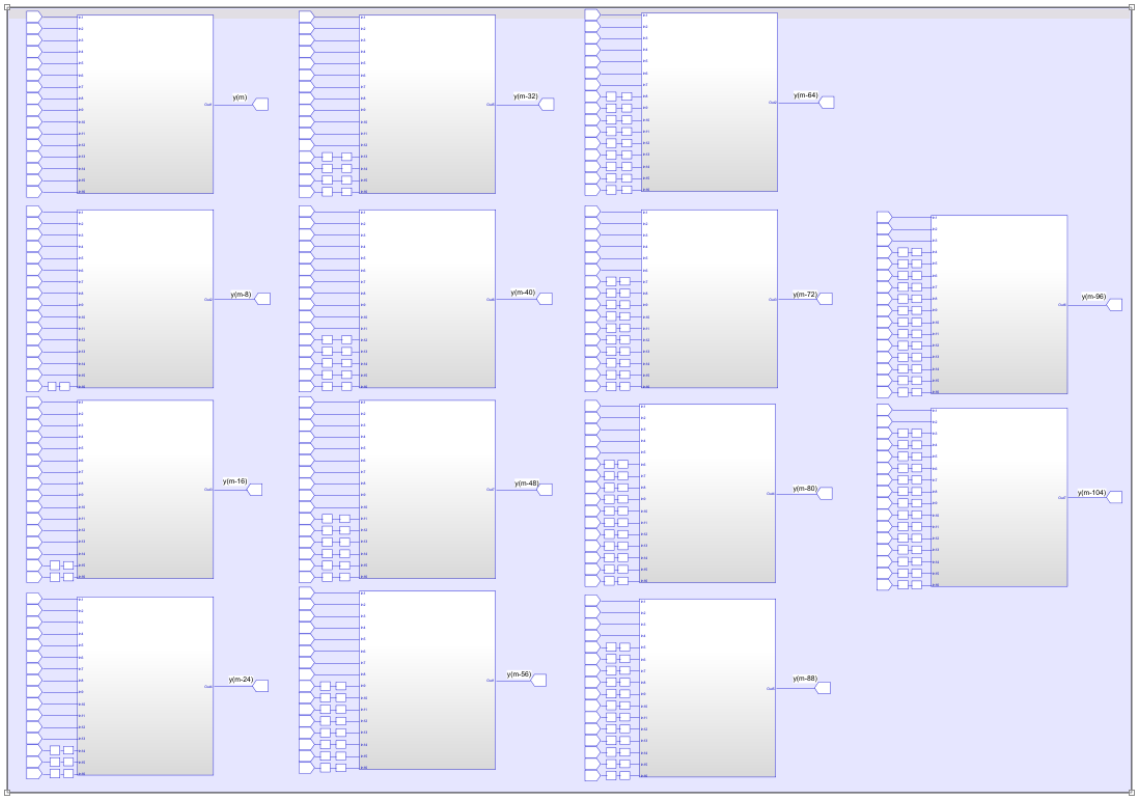


Figura 77: Modelo Simulink Subfiltros-Paralelización $\boldsymbol{P_o = 16}$ - $\boldsymbol{P = 14}$ para $\mathbf{7/8}$

Los coeficientes del filtro son los mismos que se calcularon en la sección anterior pero repartidos de la siguiente manera la ***Tabla 18.***

Tabla 18: Coeficientes Filtro Diezmador (*fdatool*) para $7/8$

s(n)	s(n-1)	s(n-2)	s(n-3)	s(n-4)	s(n-5)	s(n-6)	s(n-7)
1	6,3495E-07	1,8053E-06	3,8751E-06	6,7470E-06	9,8578E-06	1,1975E-05	1,1181E-05
2	5,1278E-06	-8,3635E-06	-3,0369E-05	-5,9745E-05	-9,2112E-05	-1,1933E-04	-1,2994E-04
3	-1,1078E-04	-5,0115E-05	5,8337E-05	2,1089E-04	3,9022E-04	5,6384E-04	6,8604E-04
4	7,0402E-04	5,6839E-04	2,4677E-04	-2,6183E-04	-9,1520E-04	-1,6223E-03	-2,2476E-03
5	-2,6266E-03	-2,5937E-03	-2,0183E-03	-8,4418E-04	8,7602E-04	0,0030	0,0051
6	0,0069	0,0078	0,0076	5,7622E-03	2,3481E-03	-0,0025	-0,0082
7	-0,0139	-0,0187	-0,0213	-0,0206	-0,0159	-0,0065	0,0073
8	0,0251	0,0454	0,0668	0,0872	0,1047	0,1175	0,1243
9	0,1243	0,1175	0,1047	0,0872	0,0668	0,0454	0,0251
10	0,0073	-0,0065	-0,0159	-0,0206	-0,0213	-0,0187	-0,0139
11	-0,0082	-0,0025	0,0023	0,0058	0,0076	0,0078	0,0069
12	0,0051	0,0030	8,7602E-04	-8,4418E-04	-0,0020	-0,0026	-0,0026
13	-0,0022	-0,0016	-9,1520E-04	-2,6183E-04	2,4677E-04	5,6839E-04	7,0402E-04
14	6,8604E-04	5,6384E-04	3,9022E-04	2,1089E-04	5,8337E-05	-5,0115E-05	-1,1078E-04
15	-1,2994E-04	-1,1933E-04	-9,2112E-05	-5,9745E-05	-3,0369E-05	-8,3635E-06	5,1278E-06
16	1,1181E-05	1,1975E-05	9,8578E-06	6,7470E-06	3,8751E-06	1,8053E-06	6,3495E-07

La ***Figura 78*** muestra las salidas obtenidas al aplicar los coeficientes mostrados en la ***Tabla 18*** con una ganancia de **7.**

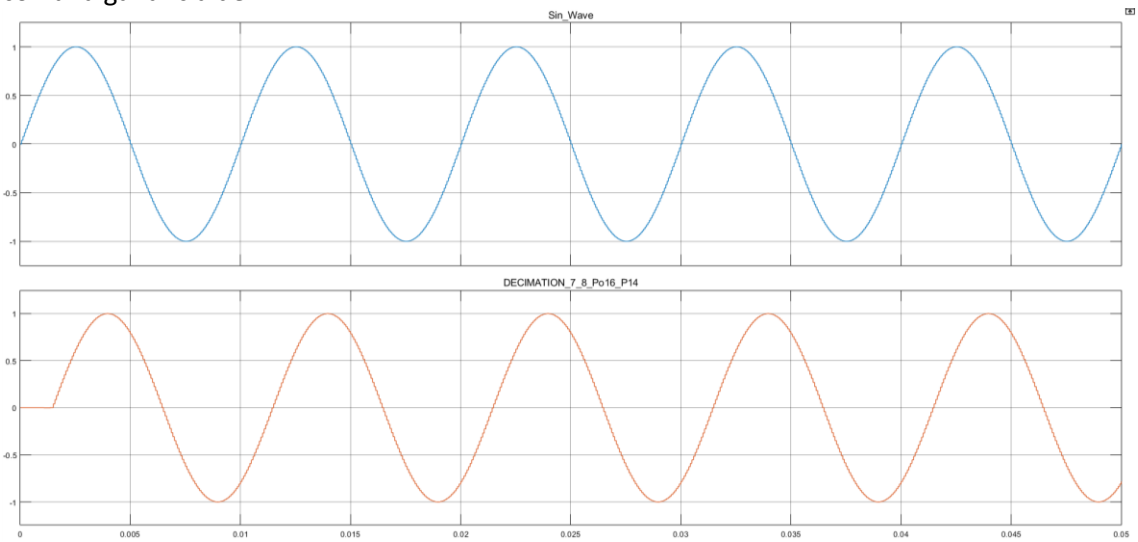


Figura 78: Señales de Entrada y Salida Filtro (*fdatool*)- $P_o = 16$ - $P = 14$ para $7/8$

3.3.4.1 Verificación diezmador por 7/8

La ***Figura 79*** muestra que el valor obtenido de frecuencia supera el valor requerido por cada canal de ***312.5 MHz.***

Fmax Summary

	Fmax	Restricted Fmax	Clock Name
1	338.07 MHz	338.07 MHz	clk

Figura 79: Frecuencia Máxima de Funcionamiento Diezmador por $7/8$

El resultado de la simulación para uno de los bancos de filtros del interpolados se muestra en la ***Figura 80***.

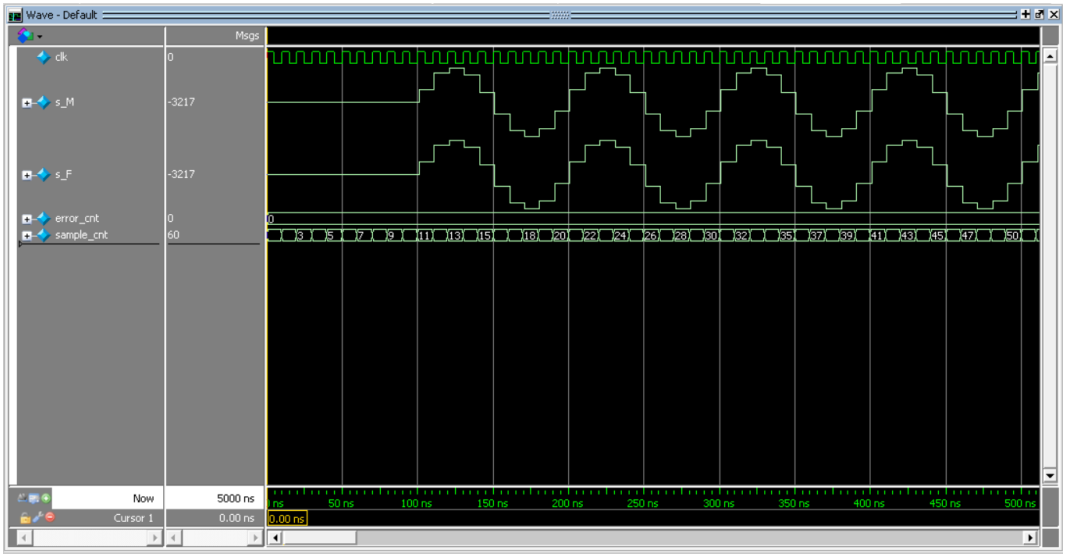


Figura 80: TestBench Banco de Filtros Diezmador por $7/8$-Paralelización $P_o = 16$ - $P = 14$

La ***Figura 81*** muestra el diagrama RTL del banco de filtros, donde se observa los sumadores en árbol totalmente segmentados y de la igual manera todas las multiplicaciones se encuentran registradas.

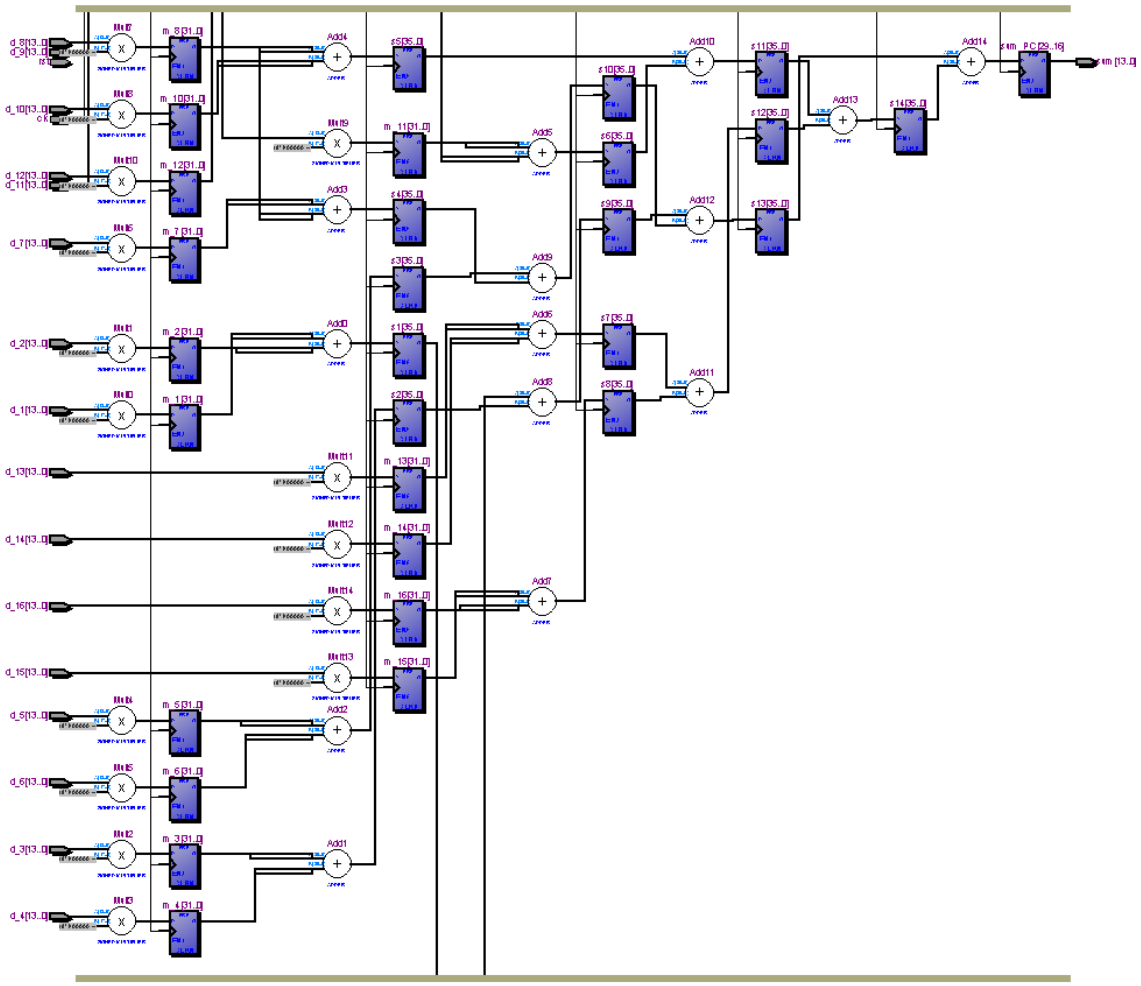


Figura 81: Diagrama RTL Diezmador por $7/8$-Paralelización $P_o = 16$ - $P = 14$

Los recursos empleados para implementar el filtro interpolador constan en la ***Tabla 19***.

Tabla 19: Recursos empleados Diezmador por $7/8$

Recurso	Cantidad
Registros	6468
DSP	126
ALM	3516

Capítulo 4. CONCLUSIONES Y LÍNEAS FUTURAS

En este último capítulo se presentan las conclusiones del proyecto. También se describen las iniciativas propuestas para trabajos futuros.

4.1 Conclusiones

- Se logró diseñar filtros interpoladores/diezmadores fraccionales aptos para ser integrados en un sistema de comunicaciones ópticas que trabaja a una velocidad de transmisión de $\mathbf{5}\ \boldsymbol{GHz}$.
- Se estudiaron diferentes estructuras de implementación filtros interpoladores /diezmadores fraccionales para incluirlas en un sistema de comunicaciones mixto, seleccionando implementar filtros polifásicos porque nos permiten aplicar técnicas de paralelización de manera más fácil que usando otro tipo de filtros cumpliendo con las restricciones planteadas.
- Se diseñaron e implementaron en Simulink todas las estructuras descritas en el presenta trabajo con el fin de realizar una comparativa de los resultados obtenidos y de los recursos empleados por cada una para seleccionar la que se apegue más a las especificaciones del sistema.
- Debido a que no fue posible cumplir los requisitos de frecuencia de operación del sistema y la necesidad de alcanzar una velocidad de procesado muy elevada para muestrear el ***DAC*** aplicando procesado secuencial, se aplicaron técnicas de paralelización de filtros que nos permiten alcanzar la velocidad de operación requerida por el sistema.
- Se codifico en lenguaje ***HDL*** y se comprobó que todos los módulos diseñados tanto para el diezmado como para la interpolación son funcionales y cumplen los requisitos de operación del sistema.
- Se segmento al máximo todos los módulos codificados en lenguaje ***HDL*** para poder reducir el camino crítico y alcanzar la velocidad de funcionamiento requerida por el sistema.
- Se consiguió una velocidad de funcionamiento superior a los $\mathbf{312.5}\ \boldsymbol{MHz}$ para $\mathbf{4/3}$ y $\mathbf{8/7}$ en el caso de la interpolación, $\mathbf{3/4}$ y $\mathbf{7/8}$ en el caso del diezmado.
- Como apunte personal el presente trabajo me ha servido para ampliar mis conocimientos sobre las diferentes técnicas de diseño e implementación de interpoladores/diezmadores fraccionales, a aplicar técnicas de paralelización en el diseño de filtros FIR y a sacar provecho de la codificación HDL para obtener altas prestaciones de velocidad.

4.2 Líneas Futuras

- Integrar los filtros diseñados en el experimento montado, ya que al momento de la finalización del presente trabajo no estaba terminada la parte de canalización del sistema.
- Disminuir la cantidad de recursos empleados manteniendo los parámetros de rendimiento del sistema de comunicaciones dentro de los márgenes aceptables, realizando el estudio de la cuantificación de la ruta de datos, ya que se pretende dentro del proceso de integración realizar medidas con diferentes modulaciones, variando la longitud de la fibra óptica.
- Desarrollar algoritmos capaces de corregir ciertas desviaciones que se producen en el proceso de transmisión por fibra óptica, principalmente la dispersión y otras desviaciones no lineales que provocan una disminución en el rendimiento del sistema, el objetivo de este tipo de sistemas es aprovechar todos los recursos de procesado que nos ofrecen los dispositivos programables *FPGAs* para transmitir la información a largas distancias a una mayor velocidad.

Capítulo 5. BIBLIOGRAFÍA

[1] I.B. Djordjevic, B. Vasic, Orthogonal frequency division multiplexing for high-speed optical transmission, Opt. Express 14 (2006) 3767–3775.

[2] W. Shieh, Q. Yang, Y. Ma, 107 Gb/s coherent optical OFDM transmission over 1000-km SSMF fiber using orthogonal band multiplexing, Opt. Express 16 (2008) 6378–6386.

[3] J. Eslava, H. González. Diseño e implementación de un sistema de verificación funcional utilizando la técnica de aceleración por hardware. (2012) p. 126-145

[4] T. L. Laakso, V. Valimaki, M. Karjalainen and U. K. Laine, "Splitting the unit delay: tools for fractional delay filter design," IEEE Signal Process, vol. 1, no.13, pp. 30-50, 1996.

[5] A. V. Oppenheim and R. W. Schafer, Discrete Time Signal Processing, Prentice-Hall, 1989

[6] R. Schmogrow, M. Winter, M. Meyer, D. Hillerkuss, S. Wolf, B. Baeuerle, A. Ludwig, B. Nebendahl, S. Ben-Ezra, J. Meyer, M. Dreschmann, M. Huebner, J. Becker, C. Koos, W. Freude, and J. Leuthold, "Real-time Nyquist pulse generation beyond 100 Gbit/s and its relation to OFDM," Opt. Express 20(1), 317–337 (2012).

[7] Intel, Cyclone V Device Handbook: Volume 1: Device Interfaces and Integration

[8] Altera, White Paper FPGA Architecture, July 2006.ver. 1.0.