Low-Complexity FIR Digital Filters: Design and Applications in Communications

by

David Ernesto Troncoso Romero

A thesis submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Department of Electronics

National Institute for Astrophysics, Optics and Electronics (INAOE)

June 28, 2013
Tonantzintla, Puebla

Supervised by:

Prof. Gordana Jovanovic Dolecek, PhD
INAOE

©INAOE 2013
All rights reserved

The author hereby grants to INAOE permission to reproduce and to distribute paper or electronic copies of the thesis in whole or in parts
Low-Complexity FIR Digital Filters: Design and Applications in Communications

David Ernesto Troncoso Romero
© INAOE

MMXIII
Abstract

In this dissertation, the efficient design of low-complexity linear-phase Finite Impulse Response (FIR) filters for digital communication applications is investigated. The research developed here relies upon the main two categories of solution, namely, decomposing the overall filter in simple subfilters (subfilter-based solutions) and simplifying the filtering coefficients by eliminating multipliers (multiplierless solutions). For both cases, new proposals have been developed.

The research contributions based on subfilters are focused on both, narrowband and wide-band cases. The Recursive Running Sum (RRS) filter, a useful filter characterized by its low complexity, is employed especially for low-pass narrowband cases. Four proposals that improve the magnitude response of RRS filters are introduced. The proposed schemes provide a more efficient balance between magnitude response improvement and the added complexity to the RRS filter in comparison with previous schemes developed in literature. The efficient design of Hilbert transformers, a special case of wide-band filter, is also investigated. A generalized method to properly combine identical-subfilter-based and periodical-subfilter-based schemes is introduced and it is shown that this method provides low-complexity filters.

Cyclotomic Polynomial Filters (CPF s), a special class of subfilters, constitute low-complexity filtering solutions widely used in the efficient design of FIR digital filters and, because of that, they are studied in this dissertation. An important contribution to design CPF-based filters is
presented, namely, the extension of the search space of CPFs beyond the limits used in literature. From the results of this extension we have developed the theorem of preservation of unitary coefficients, the main contribution on this topic. This theorem enlarges the capabilities of CPFs by showing that any CPF can have a transfer function with unitary coefficients and with the lowest computational complexity.

Finally, our contributions on multiplierless approaches are introduced with the basis on the implementation of constant multiplications as a network of additions and shifts. We develop an extension to the theoretical lower bounds for the adder cost and adder depth in the Single Constant Multiplication (SCM) problem. With this extension, the hidden theoretical lower bound for the number of adders required to preserve the minimum adder depth is revealed. From this study we introduce a general algorithm to design multiplierless filters with minimum number of adders subject to the theoretical lower bound for the adder depth, which is suitable for Field Programmable Gate Arrays (FPGA) implementations.
Resumen

En esta tesis se investiga el diseño eficiente de filtros con Respuesta al Impulso Finita (Finite Impulse Response, FIR) de fase lineal y baja complejidad para aplicaciones en comunicación digital. La investigación desarrollada aquí se basa en las dos principales categorías de solución, a saber, la descomposición del filtro total en sub-filtros simples (soluciones basadas en subfiltros) y la simplificación de los coeficientes del filtro mediante la eliminación de multiplicadores (soluciones sin multiplicadores). Para ambos casos, nuevas propuestas han sido desarrolladas.

Las contribuciones de investigación basadas en subfiltros se enfocan en ambos casos, banda ancha y banda angosta. El filtro de Suma en Funcionamiento Recursivo (Recursive Running Sum, RRS), un útil subfiltro caracterizado por su baja complejidad, es empleado especialmente para casos pasa-bajas de banda angosta. Se introducen cuatro propuestas que mejoran la respuesta en magnitud de filtros RRS. Los esquemas propuestos proveen un balance más eficiente entre la mejora de la respuesta en magnitud y la complejidad añadida al filtro RRS en comparación con esquemas previos desarrollados en la literatura. El diseño eficiente de transformadores de Hilbert, un caso especial de filtro de banda ancha, también es investigado. Se introduce un método generalizado para combinar apropiadamente esquemas basados en subfiltros idénticos y en subfiltros periódicos y se muestra que este método proporciona filtros con baja complejidad.
Los Filtros de Polinomios Ciclotómicos (Cyclotomic Polynomial Filters, CPFs), una clase especial de subfiltros, constituyen soluciones de filtrado de baja complejidad ampliamente usadas en el diseño eficiente de filtros digitales FIR y, debido a eso, son estudiados en esta tesis. Se presenta una importante contribución para diseñar filtros basados en CPFs, a saber, la extensión del espacio de búsqueda de CPFs más allá de los límites usados en la literatura. De los resultados de esta extensión se ha desarrollado el teorema de la preservación de coeficientes unitarios, la principal contribución en este tema. Este teorema amplía las capacidades de los CPFs mediante la demostración de que cualquier CPF puede tener una función de transferencia con coeficientes unitarios y con la más baja complejidad computacional.

Finalmente, se introducen las contribuciones sobre enfoques sin multiplicadores con fundamento en la implementación de multiplicaciones constantes como una interconexión de sumas y corrimientos. Se desarrolla una extensión de los límites inferiores teóricos para el costo de sumadores y la profundidad de sumadores en el problema de Multiplicación por Única Constante (Single Constant Multiplication, SCM). Con esta extensión se revela el límite teórico (hasta ahora oculto) para el número de sumadores requeridos para preservar la profundidad de sumadores mínima. De este estudio se introduce un algoritmo general para diseñar filtros sin multiplicadores con mínimo número de sumadores sujeto al límite inferior teórico para la profundidad de sumadores, el cual es adecuado para implementaciones en FPGA (Field Programmable Gate Array).
Acknowledgment

Deeply thankful to The Lord in first place, I write these lines to express my gratitude to those who have somehow contributed in the successful culmination of my PhD studies.

Thanks to CONACYT for providing me financial support through the scholarship No. 204370, which included the funding for a research year in USA. I extend my gratitude to all Mexican citizens, who have made possible the existence of this scholarship. My acknowledgement also to the institute INAOE, for all the resources they have provided for the sake of a fairly nice research environment.

I really appreciate the guidance of my adviser, Prof. Gordana Jovanovic Dolecek, who continuously encouraged me to carry out this research and taught me many tricks of the trade, beyond of the formal research advices. Thanks to Prof. Massimiliano Laddomada, who introduced me to the field of Cyclotomic Polynomial Filters and patiently revised several of my works. Thanks to Prof. Uwe Meyer-Baese, who supervised my work in USA and openly shared with me part of his knowledge. Particularly I thank Prof. Juan Manuel Ramírez, who motivated me to keep going upon letting me know his recognition to my humble efforts, and Prof. Svetlana Mansurova, who evaluated me flexibly in my early days at INAOE. Because of that, I was able to pursue the PhD years later. My acknowledgment is sincerely extended to my dissertation reviewers, for their valuable support in the last stage of this process.

I am intensely thankful to my parents, who have given all for me in an exquisite representation of God’s love. When I told them “I want to pursue the
PhD” they were pleased and sad in equal parts because it meant that I would be away from them for more years. They proudly have expressed their satisfaction of what I have been doing, but secretly have suffered my absence, which can be exemplified in those brief video-callings when I said “I just have 15 minutes to talk. I have to prove a small statement that I know is true, and I need time to figure out how to do it.” Thanks to The Lord for my parents. Thanks to Anita, my sister, and to José, my brother-in-law, for taking care of them and for encouraging me to keep going. Thanks to Miriam, my dear wife, for her patience and support. Her clever advices have always been a blessing. I extend my gratitude to her family, for always considering me another of their dear sons.

Thanks to all the friends that encouraged me and gave me their sincere support in many ways. Thanks to my friends from INAOE: Orlando, Gerardo, Gaudencio, Eric Mario, Oscar Addiel, Vanessa, Wilson and Edel Serafin. Special thanks to Elery (my soul brother) and Orlando for being so close and tolerating my mood. I have lost track of Nacho, but I will always acknowledge his bright mind and singular motivation words in my early days at INAOE.

During my days in USA, I had the blessing to know people that now are a vital part in my life. Special thanks to Yi-Ling, whom I sincerely admire, to Mike and Barbara Lynch, Raúl and María Oquendo, Art and Adele TerMorshuizen, Roger and Elsa Fierro, Aaron and Charis Chen, Jennifer Lee, Laszlo and Jozann Roszol, Carrie Godfrey, Lin and Ying Zhang and all these blessed people that openly received me. Thanks to my granny, Romana, who incessantly prays for me and my wife, and to all those who prayed for us when we started our one-year journey to USA.

David Ernesto Troncoso Romero

June, 2013.
Profundamente agradecido con El Señor en primer lugar, escribo estas líneas para expresar mi gratitud a aquellos que de algún modo han contribuido en la exitosa culminación de mis estudios de doctorado.

Gracias al CONACYT por darme apoyo económico con la beca No. 204370, la cual incluyó la financiación para un año de investigación en Estados Unidos. Extiendo mi agradecimiento a todos los ciudadanos mexicanos, quienes han hecho posible la existencia de esta beca. Mi reconocimiento también para el instituto INAOE, por todos los recursos que ha dispuesto en aras de un ambiente de investigación bastante agradable.

Realmente aprecio la guía de mi asesora, Prof. Gordana Jovanovic Dolecek, quien continuamente me animó a llevar a cabo esta investigación y me enseñó muchos trucos del oficio, más allá de la formal asesoría de investigación. Gracias al Prof. Massimiliano Laddomada, quien me introdujo en el campo de los Filtros CPF y pacientemente revisó varios de mis trabajos. Gracias al Prof. Uwe Meyer-Baese, quien supervisó mi trabajo en Estados Unidos y abiertamente me compartió parte de su conocimiento. Particularmente agradezco al Prof. Juan Manuel Ramírez, quien me motivó a seguir adelante al reconocer mis humildes esfuerzos, y la Prof. Svetlana Mansurova, quien me evaluó con flexibilidad en mis primeros días en el INAOE. Debido a eso, tuve la oportunidad de seguir con el doctorado años después. Extiendo mi sincero agradecimiento a mis revisores de tesis, por su valioso apoyo en la última etapa de este proceso.

Estoy sumamente agradecido con mis padres, que han dado todo por mí en una exquisita representación del amor de Dios. Cuando les dije “Quiero seguir
con el doctorado” estaban felices y tristes en partes iguales, ya que significaba que yo iba a estar lejos de ellos durante más años. Con orgullo han expresado su satisfacción de lo que he estado haciendo, pero en secreto han sufrido esa ausencia que se puede ejemplificar en aquellas breves video-llamadas cuando les decía “Sólo tengo 15 minutos para hablar. Tengo que probar una pequeña declaración que sé que es verdad y necesito tiempo para averiguar cómo hacerlo”. Gracias a Dios por mis padres. Gracias a Anita, mi hermana, y a José, mi cuñado, por cuidar de ellos y por animarme a seguir adelante. Gracias a Miriam, mi querida esposa, por su paciencia y apoyo. Sus consejos inteligentes siempre han sido una bendición. Extiendo mi gratitud a su familia, porque siempre me cuentan como otro de sus queridos hijos.

Gracias a todos los amigos que me alentaron y me dieron su apoyo sincero de muchas maneras. Gracias a mis amigos del INAOE: Orlando, Gerardo, Gaudencio, Eric Mario, Oscar Addiel, Vanessa, Wilson y Edel Serafín. Un agradecimiento especial a Elery (mi hermano del alma) y Orlando para estar tan cerca y tolerar mi humor. He perdido la pista de Nacho, pero siempre reconoceré su brillante mente y sus singulares palabras de motivación en mis primeros días en el INAOE.

Durante mis días en Estados Unidos tuve la bendición de conocer a quienes ahora son una parte vital en mi vida. Un agradecimiento especial a Yi-Ling, a quien sinceramente admiro, a Mike y Barbara Lynch, Raúl y María Oquendo, Art y Adele TerMorshuizen, Roger y Elsa Fierro, Aarón y Charis Chen, Jennifer Lee, Laszlo y Jozann Roszol, Carrie Godfrey, Lin y Ying Zhang y todas estas personas bendecidas que abiertamente me recibieron. Gracias a mi abuelita Romana, quien incesantemente ora por mí y por mi esposa, y a todos los que oraron por nosotros cuando empezamos nuestro viaje de un año a Estados Unidos.

To my parents: Victoriano Troncoso González and Ricarda Romero Guillén
# Contents

Abstract.................................................................................................................. XIII

Resumen.................................................................................................................. V

Acknowledgment..................................................................................................... VII

Agradecimientos....................................................................................................... IX

Contents.................................................................................................................. XIII

List of tables............................................................................................................ XIX

List of figures.......................................................................................................... XXI

CHAPTER 1: Introduction......................................................................................... 1
  1.1 Introduction...................................................................................................... 1
  1.2 Objective ...................................................................................................... 4
  1.3 Contributions ............................................................................................... 5
  1.4 Dissertation organization ........................................................................... 8
  1.5 References ................................................................................................... 9

CHAPTER 2: Efficient design of digital FIR filters: state of the art.............. 17
2.1 Introduction................................................................. 18
2.2 Subfilter-based methods....................................................... 19
  2.2.1 Identical subfilters approach........................................ 20
  2.2.2 Periodical subfilters approach....................................... 21
  2.2.3 Other approaches.................................................... 23
2.3 Multiplierless methods............................................. 24
  2.3.1 Multiplierless methods over discrete spaces.............. 27
  2.3.2 MCM-based multiplierless methods.............................. 28
2.4 Methods employed in this dissertation........................ 29
  2.4.1 Sharpening method............................................... 30
  2.4.2 Frequency Transformation (FT) method..................... 32
  2.4.3 Frequency Transformation for Hilbert transformers........ 34
  2.4.4 Pipelining-Interleaving (PI) structure..................... 38
  2.4.5 FRM method for Hilbert transformers.................... 39
2.5 References..................................................................... 41

CHAPTER 3: Contributions on the approaches based in subfilters...... 55
3.1 Contributions on narrowband filters................................. 56
  3.1.1 Introduction to Recursive Running Sum (RRS) filters...... 57
    3.1.1.1 Compensated RRS filters................................ 63
  3.1.2 Efficient sharpening-based scheme of compensated RRS filters.................................................... 65
    3.1.2.1 Design examples and discussion of results............. 74
  3.1.3 Optimal sharpening approach for RRS-based filters........ 81
    3.1.3.1 Optimized sharpening of compensated RRS filters....... 87
3.1.3.2 Sharpening the second-stage filter in a two-stage RRS-based structure…………………………………………………… 92
3.1.3.3 Design examples and discussion of results…………………. 99
3.1.4 Changing the paradigm in RRS magnitude improvement: the introduction of a corrector filter……………………………………. 102
3.1.4.1 Calculation of the optimal value $L$………………………………………. 110
3.1.4.2 Design of the basic low-pass filter $G(z)$……………………………… 113
3.1.4.3 Design steps of the corrector filter $F(z)$…………………………………………… 115
3.1.4.4 Proposed structure………………………………………………………… 116
3.1.4.5 Design examples and discussion of results…………………………. 120
3.1.5 Separated magnitude improvement of RRS filters using compensation filtering and Chebyshev sharpening…………………. 127
3.1.5.1 Design and characteristics of Chebyshev RRS filters…….. 128
3.1.5.2 When to use Chebyshev RRS filters…………………………………… 130
3.1.5.3 Compensation filter design………………………………………………… 133
3.1.5.4 Proposed structure………………………………………………………… 144
3.1.5.5 Design steps…………………………………………………………..… 145
3.1.5.6 Design examples and discussion of results…………………..… 145
3.2 Contributions on wideband filters…………………………………… 149
3.2.1 Efficient design of FIR Hilbert transformers using Frequency Transformation (FT)……………………………………………… 150
3.2.1.1 Filtering structure for FT-based Hilbert transformers using the Pipelining-Interleaving (PI) method…………………………… 151
3.2.1.2 Optimal design of Hilbert transformers based on FT method………………………………………………………………………………… 154
3.2.1.3 Multi-level FT method for Hilbert transformers............... 163
3.2.1.4 Design of Hilbert transformers using a combined FRM-FT approach.................................................. 167
3.2.1.5 Design examples and discussion of results............... 172
3.3 Conclusion................................................................. 180
3.4 References............................................................ 183

CHAPTER 4: Contributions on a special class of subfilters:

Cyclotomic Polynomial Filters........................................ 195
4.1 Introduction to Cyclotomic Polynomial Filters (CPFs)........ 196
4.1.1 Definition and usefulness...................................... 197
4.1.2 Problem formulation to design CPF-based filters......... 201
4.2 The advantages of using an extended search.................. 204
4.2.1 Proposed systematic solution procedure.................... 206
4.2.2 Design examples and preliminary observations........... 218
4.3 Theorem of preservation of unitary coefficients.............. 224
4.3.1 Theorem and proof................................................. 225
4.3.2 Improvements on forming a suitable search space........ 238
4.3.3 Design examples and discussion of results................. 241
4.4 Conclusion............................................................. 249
4.5 References............................................................ 251

CHAPTER 5: Contributions on multiplierless SCM and MCM-based techniques........................................... 255
5.1 Introduction............................................................ 256
5.2 Inclusion of prime factors to calculate the theoretical lower bounds in Single Constant Multiplications (SCM)......................... 257
  5.2.1 Key theorems for the current lower bounds................................. 258
  5.2.2 Analysis of the current lower bounds......................................... 261
  5.2.3 Proposed extension of the current lower bounds............................. 262
  5.2.4 Comparison of proposed and current lower bounds....................... 272
5.3 Optimization of high-speed pipelining in FPGA-based FIR filters.. 273
  5.3.1 Pipelining in SCM and MCM blocks.......................................... 275
  5.3.2 Proposed method........................................................................ 283
  5.3.3 Design examples and discussion of results................................. 291
5.4 Conclusion...................................................................................... 295
5.5 References..................................................................................... 297

CHAPTER 6: General conclusions...................................................... 303
  6.1 Conclusion.................................................................................... 303
  6.2 Future work.................................................................................. 306

APPENDIX A: Additional proofs for theorems in Chapter 5............. 309
  A.1 First proof required in Theorem 1................................................. 309
  A.2 Second proof required in Theorem 1............................................. 310
  A.3 Proof required in Theorem 3........................................................ 312
  A.4 Proof required in Theorem 4........................................................ 314
  A.5 Additional proofs......................................................................... 314
    A.5.1 Proof for (A.3)........................................................................ 314
    A.5.2 Proof for (A.12)..................................................................... 315
A.5.3 Proof for (A.20) ........................................................................... 316

APPENDIX B: Publications ................................................................. 317
  B.1 Journals .................................................................................... 317
  B.2 Chapter books ........................................................................... 318
  B.3 International conferences ......................................................... 318
  B.4 National congresses ................................................................. 320

Glossary ............................................................................................ 321
## List of tables

3.1 Typical values of compensation filter parameters................................. 64
3.2 Comparison of results in Example 1........................................................ 76
3.3 Comparison of results in Example 2........................................................ 78
3.4 Comparison of results in Example 3........................................................ 80
3.5 Comparison in terms of APOS in Examples 5 and 6................................. 102
3.6 Comparison in terms of APOS in Example 8.......................................... 125
3.7 Comparison in terms of APOS in Example 9.......................................... 127
3.8 Typical values of the amplitude transformation-based compensation filter parameters........................................................................................................ 140
3.9 Comparison in terms of APOS in Examples 12 and 13............................ 148
3.10 Coefficients of the subfilter $G(z)$ in Example 14................................. 174
3.11 Structural coefficients in Example 14................................................... 174
3.12 Comparison of results for the design of a Hilbert transformer in Example 14............................................................................................................. 175
3.13 Coefficients of $H_{1w}(z)$ in Example 15............................................... 178
3.14 Coefficients of $H_{ma}(z)$ in Example 15............................................... 178
3.15 Structural coefficients in Example 15................................................... 178
3.16 Comparison of results for the design of a Hilbert transformer in Example 15............................................................................................................. 180

4.1 Eligible indexes of CPFs in Example 3..................................................... 220
4.2 Comparison of results in Example 3....................................................... 220
4.3 Eligible indexes of CPFs in Example 4........................................ 222
4.4 Comparison of results in Example 4........................................ 222
4.5 Eligible indexes of CPFs in Example 3 obtained with profit coefficients................................................................. 240
4.6 Eligible indexes of CPFs in Example 5................................. 243
4.7 Comparison of results in Example 5........................................ 244
4.8 Eligible indexes of CPFs in Example 6................................. 245
4.9 Comparison of results in Example 6........................................ 246

5.1 Percentage of constants with improved lower bounds.............. 272
5.2 Synthesis results for the implementation of the constant multiplier 25 on a Xilinx Virtex-4 XC4VSX25-10FF680........................................ 276
5.3 Synthesis results for the implementation of filter FIR3 on a Xilinx Virtex-4 XC4VSX25-10FF680........................................ 293
5.4 Synthesis results for the implementation of first benchmark filters on a Xilinx Virtex-4 XC4VSX25-10FF680........................................ 294
5.5 Synthesis results for the implementation of second benchmark filters on a Xilinx Virtex-4 XC4VSX25-10FF680........................................ 295
## List of figures

2.1 FIR filter in transposed form................................................................. 25

2.2 Multiplierless implementation of products $29x$ and $43x$.................. 26

2.3 Multiplierless implementation of products $29x$ and $43x$ with partial product sharing................................................................. 27

2.4 Piecewise linear ACF and its polynomial approximation.................... 31

2.5 Filtering of two independent sequences using a single filter............. 38

2.6 Filtering of a sequence with $K$ identical cascaded filters, (a) PI architecture with a single filter, (b) equivalent single-rate structure... 39

3.1 Magnitude responses of $K$ cascaded RRS filters with $M = 8$ and $K = 1, 2, 3$ and 4................................................................. 60

3.2 Proposed structures to improve the magnitude characteristics of RRS filters........................................................................................................ 68

3.3 Flowchart of the algorithm to find the proper $K$ and $n$ to form $H_n(z)$... 72

3.4 Proposed structures for decimation by $M$............................................. 73

3.5 Magnitude responses $|H_0(e^{j\omega})|$ (solution given in [29]) and $|H_1(e^{j\omega})|$ in Example 1................................................................. 75

3.6 Magnitude response of the proposed filter $H_p(z)$ in Example 2........ 78

3.7 Magnitude response of the proposed filter $H_p(z)$ in Example 3........ 80

3.8 CIC-like structure for decimation filtering with sharpened RRS filters........................................................................................................ 86
3.9 CIC-like structure for decimation filtering with sharpened-compensated RRS filters

3.10 Magnitude responses of the filters $H_{sh,1}(z)$, $H_{c,sh,2}(z)$, $H_{sh,3}(z)$ and $H_{c,sh,4}(z)$, presented in Example 4 with $M=16$, $\nu=2$ and $\omega_p=0.6\pi/(M\nu)$.

3.11 Estimated degree of the sharpening polynomial to sharpen RRS filters (dashed line) and compensated RRS filters (solid line), with $\delta_p=0.001$ and $\delta_s=0.001$ (left), $\delta_s=0.0001$ (middle) and $\delta_s=0.00001$ (right).

3.12 Magnitude responses of filters $H_{1,k}(z)$, $H_{2,k}(z^M)$ and $H_{TS}(z)$ for the case $M_1=M_2=5$, with $K_1=K_1=4$.

3.13 Computationally efficient structure for a two-stage RRS-based decimation filter.

3.14 Second-stage sharpened decimation structures for $M_2=2$ with compensation (upper figure) and without compensation (lower figure).

3.15 Magnitude responses of the filters $G_1(z)$ (method [27]), $G_2(z)$ (method [29]) and $G_3(z)$ (proposed), presented in Example 5 with $M=16$, $\nu=4$ and $\omega_p=0.907\pi/(M\nu)$.

3.16 Magnitude responses of the filters $G_1(z)$ (method [29]) and $G_2(z)$ (proposed), presented in Example 6 with $M=M_1M_2=8*17=136$, $\nu=2$ and $\omega_p=0.9\pi/(M\nu)$.

3.17 Magnitude responses of the filters $F_1(z)$, $H_{TS}(z)$ and $H_{C1}(z)$.

3.18 Magnitude responses of the filters $F_2(z)$, $H_{TS}(z)$ and $H_{C2}(z)$.

3.19 Magnitude responses of the filters $F_3(z)$, $H_{TS}(z)$ and $H_{C3}(z)$.

3.20 Magnitude responses of the filters $H_{C1}(z)$, $H_{C2}(z)$ and $H_{C3}(z)$ in the
pasband and the first stopband (where the worst-case attenuation occurs).................................................................................................................. 107

3.21 Pictorial representation of the improvement in the passband and in the first stopband of the RRS filter guaranteed by the corrector filter........................................................................................................ 111

3.22 Initial decimation structure where a two-stage RRS filter is cascaded with the proposed corrector filter.......................................................... 116

3.23 Computationally efficient structure stemming from multirate identities.................................................................................................................. 116

3.24 Second-stage CIC-like structure with polyphase decomposition of the basic low-pass filter when $M_2 > 2$ is a prime number (upper figure) and when $M_2 = 2$ (lower figure)....................................................... 118

3.25 Second-stage structure with embedded corrector filtering, where $M_2$ is a composite number; (a) Initial structure; (b) CIC-like structure obtained after applying noble identities to both, the expanded basic low-pass filter and the comb part of $H_2^{K_2}(z)$; (c) Resulting CIC-like second-stage structure, with the additional computational complexity of the corrector filtering working at lower rate. Note that the structure of Fig. 3.25(c) can be used as a CIC-like overall structure, and not only in the second-stage filtering................................. 119

3.26 Magnitude response of the basic low-pass filter $G(z)$ along with the desired passband and stopband characteristics........................................ 122

3.27 Magnitude responses of filters $H_1^{K_1}(z)$, $H_2^{K_2}(z^{M_1})$ and $(z) = G(z^{L_M})$, with $K_1 = K_2 = 5$, $M_1 = 7$ and $L = 1$, presented in Example 7.............. 122

3.28 Magnitude responses of $K = 5$ cascaded RRS filters and the
3.29 Magnitude responses of the filters $H_a(z)$ (method [27]) and $H_b(z)$ (proposed method) presented in Example 8  
3.30 Magnitude responses of the filters $H_{a,1}(z)$ (method [23]) and $H_{b,1}(z)$ (proposed method) presented in Example 9  
3.31 Magnitude responses of the filters $H_{a,2}(z)$ (method [22]) and $H_{b,2}(z)$ (proposed method) presented in Example 9  
3.32 Worst-case attenuation values of $H_{c,N}(z)$ and $H^k(z)$ vs $\nu$ for $1 \leq N \leq 4$ and $1 \leq K \leq 5$  
3.33 Worst-case attenuation values of $H_{c,N}(z)$ and $H^k(z)$ vs $\nu$; $5 \leq N \leq 10$ and $6 \leq K \leq 10$  
3.34 Linear transformation of a cosine-squared filter. To see how the compensation characteristic arises, proceed counter-clockwise starting on the upper right. Follow the dashed arrows as a reference  
3.35 Second-order transformation-based compensator. Usually $y_0 = 1$  
3.36 Optimal values that the coefficient $|p_1|$ takes to compensate the droop of $K$ cascaded RRS filters for some values $\nu$, with $K$ ranging from 2 to 15 and $\nu$ ranging from 2 to 5  
3.37 Passband detail of magnitude responses of $K = 5$ cascaded RRS filters without compensation ($|H^k(e^{j\omega})|$), compensated with the proposed filter ($|H_a(e^{j\omega})|$), compensated with method [6] ($|H_b(e^{j\omega})|$) and compensated with method [9] ($|H_c(e^{j\omega})|$) in Example 10  
3.38 Passband detail of magnitude responses of $K = 4$ cascaded RRS filters without compensation ($|H^k(e^{j\omega})|$), compensated with the
proposed filter ($|H_a(e^{i\omega})|$), compensated with method [5] ($|H_b(e^{i\omega})|$) and compensated with method [9] ($|H_c(e^{i\omega})|$) in Example 11

3.39 Proposed structure. The dotted blocks are included if $N_{\text{min}}$ is odd

3.40 Magnitude responses of the filters $H_a(z)$ (proposed method using a 4th-order optimized compensator), $H_b(z)$ (proposed method using a 2nd-order amplitude-transformation-based compensator) and $H_c(z)$ (method [23]), presented in Example 12

3.41 Magnitude responses of the filters $H_a(z)$ (proposed method using a 4th-order optimized compensator), $H_b(z)$ (proposed method using a 2nd-order amplitude-transformation-based compensator) and $H_c(z)$ (method [22]), presented in Example 13

3.42 Single-rate structure for FT-based Hilbert transformer using a cascaded line of identical subfilters

3.43 Equivalent single-rate structure when the PI technique is used for a cascaded line of subfilters

3.44 Equivalent single-rate structure for FT-based Hilbert transformer when the PI technique is used for the cascaded line of subfilters

3.45 Proposed PI-based structure

3.46 Transition band of the prototype filter and ripple of the subfilter as functions of $\Omega'$. As $\Omega'$ increases, the Prototype filter becomes simpler because its transition band increases. On the contrary, the subfilter becomes more complex because its ripple decreases

3.47 Percentage of absolute error in the estimation of lengths of Hilbert transformers using equations (3.152) and (3.154) for cases with ripples $\delta = 0.0001$ and $\delta = 0.005$, with $\Omega'$ in the range from $0.1\pi$ to
3.48 Percentage of absolute error in the estimation of lengths of Hilbert transformers using equations (3.152) and (3.154) for cases with relative frequencies $\omega L = 0.0001\pi$ and $\omega L = 0.005\pi$, with $\delta$ in the range from 0.1 to 0.9 ............................................................. 161

3.49 Magnitude response of the subfilter $G(z)$ in Example 14 ............ 175

3.50 Magnitude response of the prototype filter $P(z)$ in Example 14 .... 176

3.51 Magnitude response of the desired Hilbert transformer $H(z)$ in Example 14 ............................................................. 176

3.52 Magnitude response of the subfilter $G(z)$ in Example 15 ............ 179

3.53 Magnitude response of the prototype filter $P(z)$ in Example 15 .... 179

3.54 Magnitude response of the desired Hilbert transformer $H(z)$ in Example 15 ............................................................. 180

4.1 CPF-based filter ............................................................. 202

4.2 Order of a CPF, $\phi(p)$, against the index $p$ ............................... 205

4.3 Flowchart of the algorithm to find non-recursive and low-complexity recursive transfer functions for CPFs with indexes $p$ up to 200 ....... 210

4.4 Magnitude responses of two CPFs that are possible candidates to be constituent subfilters of a multiband lowpass CPF-based filter with 12 stopbands; (a) CPF with index $p = 15$ is non-eligible, (b) CPF with index $p = 25$ is eligible ............................................................. 215

4.5 Worst-case magnitude response values in every band of interest of the CPF with index $p_{14} = 25$ ............................................................. 217

4.6 Magnitude response of the CPF-based filter designed in Example 3 ... 219
4.7 Magnitude response of the CPF-based filter designed in Example 4… 221
4.8 Magnitude response of the CPF-based filter designed in Example 5… 243
4.9 Magnitude response of the CPF-based filter designed in Example 6… 246

5.1 Completely Multiplicative (CM) graph……………………………… 261
5.2 CM-based graph with \( p \) \( \times \) operations…………………………………… 263
5.3 A simplified example of a Logic Block (LB) in an FPGA……………… 273
5.4 Fully Pipelined Reduced Adder Graphs (FP-RAGs) for the
implementation of a coefficient multiplier 25: (a) Solution with the
form 25=3×8+1, (b) Solution with the form 25=5×5………………………… 276
5.5 The \( \Phi \)-operation…………………………………………………… 278
5.6 Cost function with vector \( s \) counted in Gray code………………… 289
5.7 FP-MCM block from method [2]. It requires 15 \( \Phi \)-operations……… 291
5.8 FP-MCM block from method [26]. It requires 11 \( \Phi \)-operations……… 292
5.9 FP-MCM block from proposed method. It requires 10 \( \Phi \)-operations… 292
1 Introduction

My mouth shall speak of wisdom; and the meditation of my hearth shall be of understanding.

Psalms 49:3

This chapter provides introductory background on the efficient design of Finite Impulse Response (FIR) digital filters. The objective of the research carried out in this dissertation is presented and the contributions are listed and briefly described. The dissertation organization is introduced at the end of the chapter.

1.1 Introduction

Digital filters play a central role in modern Digital Signal Processing (DSP) systems. Finite Impulse Response (FIR) filters with fixed coefficients find extensive applications in communication systems to perform functions such as channelization, channel equalization, matched filtering, pulse shaping and
Sampling Rate Conversion (SRC), among others [1]-[7]. This is mainly due to their absolute stability and the fact that FIR filters can be designed to have linear-phase property. Additionally, FIR filters have a low sensitivity to the finite wordlength representation of the coefficients, and the output noise due to multiplication round-off errors in FIR filters is also low [8].

The main disadvantage of conventional FIR filter designs is that they require, especially in applications demanding narrow transition bands, a high number of arithmetic operations and hardware components. This makes the implementation of narrow transition band FIR filters very costly [8]. However, several methods to design low-complexity FIR filters have been developed over the past 3 decades, approximately. Some of the most popular methods, which will be revised with more detail in Chapter 2, can be categorized as follows:

1) Identical-subfilter-based techniques. Techniques that use cascaded identical subfilters as basic building blocks have been proposed in [9]-[19].

2) Periodical-subfilter-based techniques. Techniques that use cascaded periodical subfilters as basic building blocks, where the frequency response of at least one of them is periodical with period $2\pi/M$ and $M \geq 2$, have been proposed in [22]-[28].

3) Multiplierless design techniques. These techniques replace the expensive and power-hungry multipliers by simpler addition-subtraction and shift operations [29]-[36].

4) MCM-based techniques. These techniques are applied to multiplierless filters and they are based in the transposed structure, where the input signal is first multiplied by the constant coefficients and then goes into
the delay elements. This operation is often referred as Multiple Constant Multiplication (MCM) [37]-[48].

The different advantages of each one of the aforementioned techniques are closely related to the application of the desired filter and the platform for implementation. Depending on these cases, any of these methods can be used to obtain a low-complexity filter with a given frequency response specification. As an example, techniques in category (1) are efficient in the design of filters for SRC based on Recursive Running Sum (RRS) filters, where the multiple use of the same RRS filter is common because RRS filters are very simple and can be implemented several times. Additionally, these techniques can be used when an area reduction is required [20], but care must be taken in increasing the clock rate, since the result is higher power consumption [21].

Filters from category (2) are used to implement filters with very narrow transition bands and they are adequate from a low power consumption perspective, since the subfilters have few non-zero coefficients and the overall filter require lower number of computations. However, the number of delay elements is increased, especially in the approaches that require a complementary filter (for example [24], [28]) when the periodical filter is implemented using transposed structure. The use of simple cascaded multiplier-free prefilters [27] has been a good solution to reduce further the computational complexity. Therefore, better results than those of the original techniques have been obtained.

The implementation on Field Programmable Gate Array (FPGA) of designs based in subfilters can take advantage of the embedded Random
Access Memory (RAM) memory blocks. On the other hand, implementations based on full-custom Application-Specific Integrated Circuits (ASICs) have to consider the trade-off between multipliers, adders and registers with regard to area usage [40]. For both FPGA and ASIC-based FIR filters, the multiplierless and MCM solutions from categories (3) and (4) are recommended [37]-[45]. However, a major difficulty encountered in designing an optimum multiplierless filter is its exponential complexity with respect to the filter order, i.e., the design of FIR filter with a minimum number of adders is conjectured to be an NP-hard problem [39], [49]. Due to that, several heuristic methods have been developed [33]-[36], [38], [40]-[41] and in general the multiplierless and MCM-based methods are employed mainly when the desired filter has a low or moderate order.

As we have seen, the design of low-complexity FIR filters is a challenging task since many aspects have to be taken in account. The efficient design of FIR digital filter needs a trade-off between very stringent design specifications, low power consumption, low area requirements, high speed of computations and low time and design effort. An acceptable design should balance the trade-off to a reasonable degree. These observations are the motivation to carry on the present research, where the efficient design of FIR digital filters is addressed.

1.2 Objective

The main goal of this dissertation is developing novel methods to design low-complexity FIR filters taking as a foundation the subfilter-based techniques and the multiplierless and MCM-based techniques.
Particularly, the following special classes of filters have been investigated due to their wide array of applications in digital communications:

- Recursive Running Sum filters, a class of low-pass subfilters useful to design narrowband FIR filters with extensive applications in sampling rate conversion.
- Cyclotomic Polynomial filters, useful subfilters with low-pass, highpass and bandpass characteristics that generalize the properties of RRS filters in terms of placement of zeros over the unit circle, and which are widely used in design schemes based on subfilters.
- Hilbert transformers, allpass filters used to generate complex signals whose spectra are a half wide than the spectra of their real counterparts.

1.3 Contributions

The contributions in this dissertation include novel subfilter-based methods to design narrowband filters specially focused in decimation applications and Hilbert transformers as wideband filters, as well as a novel method to design multiplierless FIR filters. Two important theoretical derivations have been also developed, one related to Cyclotomic Polynomial filters and the other related to multiplierless multiplications by constants, both essential fields in the design of low-complexity FIR filters. The following are brief descriptions of these contributions:

- Two efficient sharpening-based structures to improve the magnitude response characteristics of RRS filters. These structures take advantage
of the passband preconditioning with the simplest compensation filter available in literature. A systematic method is introduced to choose, among both structures, the one that improves the stop-band and simultaneously keeps a minimum increase in the number of required adders. Additionally, from these two structures we have derived the efficient architectures to apply the improved RRS filters in decimation. It is shown that, among the two structures, the one based on the higher order sharpening polynomial provides a lower number of Additions Per Output Sample (APOS) when the number of adders are equal.

- An optimization-based sharpening method for RRS filters. We provide an algorithmic way to find sharpening polynomials with the minimum order necessary to preserve passband and stopband characteristics of RRS filters under given specifications. The polynomial coefficients are optimized over the powers-of-two coefficients space, thus it is guaranteed that the sharpening is an optimal solution that does not require multipliers. The sharpening of previously compensated RRS filters has been explored and it has been shown that, mainly for strict magnitude specifications, the resulting filter has lower computational complexity in comparison to sharpening of traditional RRS filters.

- A new method to improve the magnitude response characteristics of RRS filters by introducing the concept of a corrector filter. The corrector filter provides both, passband compensation and improvement in the frequency region where the worst-case attenuation occurs, while require only shift-and-add operations. The
use of corrector filters is especially helpful for two-stage RRS-based
decimation filters.

- A simple and efficient method to design multiplierless two-stage RRS-
  based decimation filters that takes advantage of the Chebyshev
  sharpening, recently introduced in literature, to obtain higher
  selectivity in comparison to traditional RRS filters. In addition, we
  introduce the design of compensation filters for RRS and Chebyshev-
  RRS decimators using amplitude transformation. It is shown that the
  transformation of cosine-squared filters provides good compensation
  characteristics. For a first-degree polynomial, the slope of the
  transformation line is explicitly set as the unique compensator’s
  multiplierless coefficient that changes proportionally with the increase
  of the passband droop. The proposed approach provides an intuitive
  and easy way of designing compensation filters, and the resulting
  filtering scheme has better magnitude response characteristics and
  fewer Additions Per Output Sample (APOS) with respect to other two-
  stage comb-based filters available in literature.

- A generalized optimization-based approach to design FIR Hilbert
  transformers based on the proper combination of Frequency
  Transformation (FT) method and Frequency-Response Masking (FRM)
  method, applicable either to a single-level or to a multi-level FT
  structure. The method allows minimizing the overall number of filter
  coefficients and the introduction of weighting factors makes the
optimization flexible to minimize the overall estimated complexity, considering also the cost contributions of adders and delay elements.

- The extension of the search space of Cyclotomic Polynomial Filters (CPFs) beyond the limits used in literature. From the results of this extension we have developed the theorem of preservation of unitary coefficients. This theorem enlarges the capabilities of CPFs by showing that any CPF can have a transfer function with unitary coefficients and with the lowest computational complexity.

- The extension of the theoretical lower bounds for the adder cost and adder depth in the Single Constant Multiplication (SCM) problem. With this extension, the hidden theoretical lower bound for the number of adders required to preserve the minimum adder depth has been revealed. From this study we introduce a general algorithm to design multiplierless filters with minimum number of adders subject to the theoretical lower bound for the adder depth, which is suitable for Field Programmable Gate Arrays (FPGA) implementations.

1.4 Dissertation organization

Six chapters are included in this dissertation. Following this introduction, Chapter 2 provides a review of the state of the art with respect to methods for efficient design of FIR digital filters and presents the basis for the proposals developed in this research. Chapter 3 presents the proposed methods developed to design narrowband and wideband FIR filters. The study and contributions on the design of FIR filters based on Cyclotomic Polynomial
filters is presented in Chapter 4. Chapter 5 presents the research contributions on the solution of the multiplierless Single Constant Multiplication (SCM) and Multiple Constant Multiplication (MCM) problems, with an efficient design method for the MCM block of FIR filters. Finally, conclusions and suggestions for future work are presented in Chapter 6. An appendix is included to introduce some proofs required in Chapter 5.

1.5 References


This chapter presents an introduction to linear-phase Finite Impulse Response (FIR) digital filters. For narrow transition bandwidths, the direct design of these filters demands a high computational complexity. Thus, several special structures together with design methods have been developed in literature to efficiently design FIR filters. These methods are classified as subfilter-based methods and multiplierless methods and the most relevant of them are revised in this chapter. Finally, the methods used as starting point to develop the research in this dissertation are explained.
2.1 Introduction

A Finite Impulse Response (FIR) filter with constant coefficients is a Linear Time Invariant (LTI) system whose transfer function is given by

\[ H(z) = \sum_{n=0}^{N} h(n)z^{-n}, \quad (2.1) \]

where \( N \) is the filter order and \( h(n) \) is the \( n \)-th filter coefficient. The values \( h(n), 0 \leq n \leq N \), are the impulse response of the filter. A FIR filter has a linear phase if its impulse response accomplishes the following condition,

\[ h(n) = \pm h(N - n), \quad (2.2) \]

for \( 0 \leq n \leq N \). If the right-hand term in (2.2) has a positive sign, the condition is called symmetry. If the sign is negative, the condition is called anti-symmetry.

Replacing (2.2) in (2.1) we obtain for a linear-phase filter

\[
H(z) = \begin{cases} 
  z^{-N/2} \left[ h\left(\frac{N}{2}\right) + \sum_{n=0}^{(N-2)/2} h(n)\left[z^{(N-2n)/2} \pm z^{-(N-2n)/2}\right]\right], & \text{N even,} \\
  z^{-N/2} \sum_{n=0}^{(N+1)/2-1} h(n)\left[z^{(N-2n)/2} \pm z^{-(N-2n)/2}\right], & \text{N odd.}
\end{cases} \quad (2.3)
\]

The frequency response is

\[
H(e^{j\omega}) = \begin{cases} 
  H(\omega)e^{-j\omega N/2}, & \text{symmetric,} \\
  H(\omega)e^{j\omega N/2}, & \text{anti-symmetric.}
\end{cases} \quad (2.4)
\]

where \( H(\omega) \) is the zero-phase frequency response given by

\[
H(\omega) = \begin{cases} 
  h\left(\frac{N}{2}\right) + 2 \sum_{n=0}^{(N-2)/2} h(n)\left[\cos\left(\frac{N}{2} - n\right)\right], & \text{symmetric, N even (Type I),} \\
  2 \sum_{n=0}^{(N-1)/2-1} h(n)\left[\cos\left(\frac{N-1}{2} - n\right)\right], & \text{symmetric, N odd (Type II),} \\
  2 \sum_{n=0}^{(N-2)/2} h(n)\left[\sin\left(\frac{N}{2} - n\right)\right], & \text{anti-symmetric, N even (Type III),} \\
  2 \sum_{n=0}^{(N-1)/2-1} h(n)\left[\sin\left(\frac{N-1}{2} - n\right)\right], & \text{anti-symmetric, N odd (Type IV).}
\end{cases} \quad (2.5)
\]
The minimum order of an optimum linear-phase FIR filter $H(z)$ meeting the low-pass filter specifications

$$1 - \delta_p \leq |H(e^{j\omega})| \leq 1 + \delta_p \quad \text{for} \quad \omega \in [0, \omega_p],$$

(2.6)

$$|H(e^{j\omega})| \leq \delta_s \quad \text{for} \quad \omega \in [\omega_s, \pi],$$

(2.7)

can be estimated as [1]

$$N \approx \frac{-20 \log_{10}\left(\sqrt{\delta_p \delta_s}\right) - 13}{14.6(\omega_s - \omega_p)/2\pi}.$$  

(2.8)

More accurate formulas are given in [2], [3]. From the above estimate, it is seen that as the transition bandwidth $(\omega_s - \omega_p)$ is made smaller, the filter order increases inversely proportional to it. Since the direct-form implementation of a symmetric (or an anti-symmetric) filter requires approximately $N$ adders, $N$ delays and $N/2$ multipliers (these last being the most expensive and power-consuming elements), this implementation becomes very costly in applications where the transition bandwidth is small.

Many methods have been derived over the past four decades to simplify the realization of FIR digital filters. Two main classes of methods are identified in this dissertation: subfilter-based approaches and multiplierless approaches. The following sections provide a review of the most popular techniques included in these two classes. Section 2.4 introduces more details about some particular methods that are used as starting point to develop the research presented in this dissertation.

### 2.2 Subfilter-based methods

It has been observed that by letting the filter order increase slightly from the minimum, there can be significant savings in the number of required
multipliers. In practical frequency selective filters there is a relatively strong correlation between neighboring impulse response values. By developing filter structures that exploit this correlation, the number of multipliers required in the implementation can be substantially reduced. There are basically two widely known approaches, namely, identical subfilters and periodical subfilters, but there exist also other useful structures less explored. Let us briefly describe these methods.

### 2.2.1 Identical subfilters approach

This approach, studied in [4]-[16], consists basically on transforming the zero-phase frequency response of the subfilter to a new zero-phase frequency response via a polynomial approximation. This polynomial is called Amplitude Change Function (ACF). The overall FIR filter is synthesized by interconnecting a number of identical simple subfilters which roughly approximate a given desired specification, and that interconnection is aided with a few additional coefficients and adders.

The method of identical subfilters was originally introduced by Kaiser and Hamming in [4], and it is commonly known as sharpening. In this method an ideal piece-wise constant ACF was used. In [5], the sharpening method was improved to provide more flexibility to the design by using an ideal piece-wise linear ACF. An explicit formula to obtain the coefficients of the ACFs used in methods [4] and [5] was introduced in [6].

Method [4] was also extended by Nakamura and Mitra in [7]. A different technique where the ACF was modeled as a low-pass filter, sometimes called Frequency Transformation (FT), was presented by Saramäki in [8]. This
technique was improved to provide powers-of-two coefficients in [9], and a similar method was presented in [10]. Other methods are [11] and [12], where the ACFs are based on window functions. In [13]-[14], the FT method was modified for half-band filters and in [15] for Hilbert transformers. A recent work extending the proposal in [15] was introduced in [16]. An important advantage of these methods relies on the fact that the simple subfilter can be used only once through multiplexing [8]. An efficient technique to apply multiplexing to sharpening-based filters was introduced in [17]. The resulting designs can have a reduced area at the cost of increased power consumption [18].

2.2.2 Periodical subfilters approach

This approach has been studied in [19]-[43], among others. Generally speaking, this method is based on the use of distinct subfilters as basic building blocks, which have different powers of $z^{-1}$ as basic delays. A filter having any delay element replaced by $M$ delays is called an expanded by $M$ filter. Its impulse response is sparse because it has few non-zero coefficients. Its frequency response is periodical with period given as $2\pi/M$, where $M$ is the expansion factor (also known as stretching factor or interpolation factor). Some of the subfilters are used to mask the undesired frequency response images (introduced because of the periodicity) of the expanded subfilters. Therefore, the resulting filter can meet a stringent specification by using moderate transition band subfilters, which have a lower complexity. The cost is an increased number of required delay elements.
Method [19] is considered pioneering of this approach. In this method an expanded filter, called model filter, generates the shape of the desired frequency response. A masking filter, called interpolator filter, is cascaded to the model filter to remove the unwanted images. This method is widely known as Interpolated Finite Impulse Response (IFIR). A multi-stage version of IFIR was proposed in [20]. That proposal consists in successive applications of the IFIR technique, where the interpolator filter is kept simple whereas the IFIR method is applied to the model filter. The result is a cascade of a model filter expanded by $M$ with different interpolators, each of them expanded by different factors of $M$. In that method the coefficients of all the involved subfilters are found through applying interlaced error functions to the Parks-McClellan algorithm, and the use of the simple Recursive Running Sum (RRS) filters [44] as interpolators was also studied. The use of Cyclotomic Polynomial Filters (CPFs) [45] in the IFIR architecture was proposed in [21]. A different structure proposed in [22] uses several model filters that differ from each other only in their expansion factors and thus it can take advantage of the use of the same filter by applying the folding technique introduced in [46]. Nevertheless, the basic model filter may still require a large number of coefficients when the desired specifications are strict. In [23] the authors presented a generalized IFIR technique that consists in applying successively the IFIR technique to the interpolator. The resulting filter has several model filters expanded by different values cascaded with an interpolator. Another approach for the design of IFIR filters was presented in [24], where a RRS filter sharpened with method [4] was proposed as the
interpolator filter. The use of rotated-sinc filters [47] as interpolators was introduced in [25]. All these methods are suitable for narrowband cases.

For the case of filters with arbitrary bandwidth, in method [26] was proposed a cascaded interconnection of expanded filters with different factors, aided with some coefficients and adders in a similar structure to [8]. However, [27] is perhaps the most representative method, which generalizes the concept of IFIR by including the complementary version of the model filter, along with its corresponding masking filter. This method is widely known as Frequency-Response Masking (FRM). Since the introduction of the FRM technique, much effort has been made to improve the filter structure. The introduction of various modified FRM structures has significantly enhanced the computational efficiency of the FRM technique by simplifying the subfilters involved in the design [28]-[40]. The usual way of connecting the masking filters in most methods remains the same as proposed in original FRM, i.e., masking filters connected in parallel. In [38] was proposed the use of serial masking schemes, and this scheme was further studied in [39]. The FRM method was modified to design half-band filters in [41], and this method was the basis to develop FRM-based Hilbert transformers in [42]-[43].

2.2.3 Other approaches

Prefilter-equalizer structures are efficient in the design of low-complexity FIR filters. In these schemes a simple multiplierless prefilter provides the required attenuation and a cascaded equalizer improves the passband. This method was introduced in [48] and was continued in [49]-[57]. CPFs have been a useful option as prefilters because of their simplicity. A recent
approach proposed in [58]-[59] consists on splitting a filter into subfilters, based on its de-convolved impulse response instead of focusing on its magnitude-response characteristic.

Other approach consists on implementing sub-structures that create segments of the desired impulse response. In [60], an extrapolated impulse response method was proposed based on the fact that the side-lobes of an impulse response can be considered scaled versions of each other. This method was recently improved in [61]. The design of FIR filters that use piecewise-polynomial approximations for narrow-band filters was introduced in [62], relying on the early works [63]-[65], and this approach was extended in [66] for wide-band filters. The implementation of these structures in Field Programmable Gate Arrays (FPGA) platforms was studied in [67].

A different approach which recently has received considerable attention consists on the design of sparse non-periodic filters [68]-[70]. In this case, the filter is designed with an order higher than the minimum but several of its coefficients are zero-valued. The result is a filter that can meet the desired specification with fewer coefficients that a direct design. Sparse non-periodic filters have been used as masking filters in [71]-[72].

2.3 Multiplierless methods

Since the complexity of FIR filters is mainly dominated by coefficients multiplication operation, one widely adopted design methodology for reducing the complexity is to replace the costly multipliers by simpler addition, subtraction and shift operations. The filter coefficients are expressed
as sums of Signed Power-of-Two (SPT) terms [73]-[101]. Additions and subtractions have nearly the same complexity in hardware, and therefore they are usually referred without distinction as additions. The shifts can be realized by using hardwired shifters, and hence the complexity of multiplierless filters is usually measured by the number of adders.

In direct designs, the transposed form is the preferred structure to implement FIR digital filters because it can have a lesser critical path, resulting in a higher speed, in comparison to the direct form. Moreover, the set of coefficients can be implemented as a multiplier block which multiplies a variable input $x(n)$ by several constants. This operation is commonly known as Multiple Constant Multiplication (MCM). Figure 2.1 shows the block diagram of a FIR filter in transposed form.

![Figure 2.1: FIR filter in transposed form.](image)

For the shift-adds implementation of constant multiplications, a straightforward method initially defines the constants in binary. Then, for each 1 in the binary representation of the constant, according to its bit position, it shifts the variable and adds up the shifted variables to obtain the
result. As a simple example, consider the constant multiplications $29x$ and $43x$. Their decompositions in binary are listed as follows:

$29x = (11101)_{\text{bin}}x = 2^4x + 2^3x + 2^2x + x = x << 4 + x << 3 + x << 2 + x,$

$43x = (101011)_{\text{bin}}x = 2^5x + 2^3x + 2x + x = x << 5 + x << 3 + x << 1 + x,$

where $<<$ indicates a left shifting in hardware. The multiplier block requires six addition operations as illustrated in Figure 2.2.

![Figure 2.2: Multiplierless implementation of products $29x$ and $43x$.](image)

The number of adders to implement the FIR filter can be further reduced by extracting common partial products from the coefficients. This extraction is commonly referred as Common Subexpression Elimination (CSE). For example, if the common subexpressions 101 and 11 are extracted and implemented first, only an additional adder is required for each coefficient, ending up with four adders in total, as shown in Figure 2.3. The problem of reducing the computational complexity in an MCM block is known as MCM problem, and it is conjectured to be NP-complete [99].
Existing methods for the design of FIR filters without multipliers can be classified into two categories, namely, multiplierless methods over discrete spaces and MCM-based multiplierless methods. In the following we review the most important methods in both categories.

![Diagram of multiplierless implementation of products](image)

**Figure 2.3**: Multiplierless implementation of products $29x$ and $43x$ with partial product sharing.

### 2.3.1 Multiplierless methods over discrete spaces

In this class of algorithms, the filter coefficients are optimized in a discrete space, such as the finite word-length space [73], SPT space [74]-[78] or other discrete spaces [79]-[85] where the common subexpression sharing is taken into consideration when filter coefficients are optimized. A local search algorithm over some common patterns was proposed in [79]. In [80], the dynamic range of each coefficient is determined, and thus, the permissible discrete values for each coefficient are obtained. An exhaustive search by applying a simple branch and bound algorithm is then used to evaluate the filter performance and adder cost for each possible combination. In this technique, the search range becomes impractically huge when the filter order
is high and/or the coefficient word-length is large. An improved version of that algorithm was presented in [81].

In [82], a binary integer programming was used to minimize the number of adders subject to a given filter specification. The algorithm produces global optimum solutions in terms of number of adders but it is computationally very demanding, and thus is only suitable for relatively short filters. A branch-and-bound Mixed-Integer Linear Programming (MILP) was proposed in [83] to optimize FIR filters in subexpression spaces. In that algorithm, subexpression spaces were constructed from a given set of subexpression bases. Refined methods based on the search of coefficients in sub-expression spaces were presented in [84]-[85].

2.3.2 MCM-based multiplierless methods

In these algorithms the filters are designed in two stages. First, the FIR filter is designed in a discrete space such as a finite word-length space or SPT space to meet a given specification. This can be performed with methods [73]-[78]. In the second stage, the number of adders required in the multiplier MCM block is reduced with an algorithm specialized in solving the MCM problem, such as [86]-[101].

The algorithms to solve the MCM problem can be classified in two main classes, namely, Common Subexpression Elimination (CSE) algorithms [86]-[92] and Graph-based algorithms [93]-[101]. In the CSE algorithms the basic idea is to find common patterns in the representations of the constants after the constants are converted to a convenient number representation, such as
Canonic Signed Digid (CSD). In this case, the solution of the algorithms depends on the number representation.

On the other hand, graph-based algorithms are bottom-up methods that iteratively construct a graph that represents the MCM block. The graph construction is guided by a heuristic that determines the next graph vertex to add to the graph. Graph-based algorithms offer more degrees of freedom by not being restricted to a particular representation of coefficients and typically produce solutions with the lowest number of operations. The recent efforts have been directed to obtain as few adders as possible but keeping the adder depth, defined as the number of adders that the input signal goes through before reaching a delay element, in a value as small as possible [91]-[92], [99]-[101].

2.4 Methods employed in this dissertation

In this dissertation we employ the following methods as the starting point to develop our research based in subfilters (presented in Chapter 3):

1) The sharpening method from [4]-[6], which is employed in subsection 3.1.2.
2) The Frequency Transformation (FT) method from [8], which inspired our proposal in subsection 3.1.3.
3) The Frequency Transformation method for Hilbert transformers from [15], used in our proposal of subsection 3.2.1.
4) The Pipelining-Interleaving (PI) structure from [17], used to develop the structures in subsection 3.2.1.
5) The Frequency Response Masking (FRM) methods from [42], which are the basis to develop the proposal presented in subsection 3.2.1.

In the following, brief explanations of these methods are presented.

### 2.4.1 Sharpening method

The sharpening technique proposed in [4] permits simultaneous improvements of both passband and stopband characteristics of linear-phase FIR filters. The technique is based on an Amplitude Change Function (ACF) which is a polynomial $P_{m,n}(x)$ that maps the amplitude $x$ into an improved amplitude $y = P_{m,n}(x)$. The improvement in the amplitude near to the passband depends on $m$, the order of tangency of the ACF at the point $(x, y) = (1, 1)$ to a line with slope equal to zero. Similarly, the improvement in amplitudes near the stopband depends on $n$, the order of tangency of the ACF at the point $(x, y) = (0, 0)$ to a line with slope equal to zero.

The sharpening technique proposed in [4] has a limited control of the improvement in both passband and stopband characteristics of the filter. This is because the desired ACF is piecewise constant. The improved sharpening technique was proposed in [5], where the desired ACF is piecewise linear. This offers more direct control to change amplitudes in the passband and/or stopband. Besides of the tangencies $m$ and $n$, the polynomial approximation to the desired ACF is controlled by other two parameters, namely, $\sigma$, the slope of a line that passes over the point $(x, y) = (1, 1)$ and $\delta$, the slope of another line that passes over the point $(x, y) = (0, 0)$. The constrains on the approximating polynomial $y = P_{\sigma, \delta, m,n}(x)$ are:
1. The \( n \)th-order tangency at \((x, y) = (0, 0)\) to the line of slope \( \delta \), i.e.,
\[ P_{\sigma,\delta,m,n}(x) = \delta x, \text{ for } x = 0. \]

2. The \( m \)th-order tangency at \((x, y) = (1, 1)\) to the line of slope \( \sigma \), i.e.,
\[ P_{\sigma,\delta,m,n}(x) = \sigma (x - 1) + 1, \text{ for } x = 1. \]

The desired piecewise linear ACF is illustrated in Figure 2.4, where \( x_{pl} \) and \( x_{pu} \) are, respectively, the minimum and maximum amplitude in the passband of the original filter, and \( x_{sl} \) and \( x_{su} \) are the minimum and maximum amplitude in the stopband of the same filter, respectively. In the same way, \( y_{pl}, y_{pu}, y_{sl}, \) and \( y_{su} \) are the minimum and maximum amplitudes in the passband and the minimum and maximum amplitudes in the stopband of the sharpened filter, respectively.

![Figure 2.4](image)

**Figure 2.4**: Piecewise linear ACF and its polynomial approximation.

In [6] a general formula was deduced to obtain directly the desired amplitude change function from the design parameters. The formula is given by
\[
P_{\sigma,\delta,m,n}(x) = \delta x + \sum_{j=n+1}^{R} (\alpha_{j,0} - \sigma \alpha_{j,1} - \delta \alpha_{j,2}) x^j, \tag{2.9}
\]
where \( R = n + m + 1 \) and
\[
\alpha_{j,0} = \sum_{i=n+1}^{j} (-1)^{-i} \binom{R}{j-i} j, \quad (2.10)
\]

\[
\alpha_{j,1} = \sum_{i=n+1}^{j} (-1)^{-i} \binom{R}{j-i} i \left(1 - \frac{i}{R}\right), \quad (2.11)
\]

\[
\alpha_{j,2} = \sum_{i=n+1}^{j} (-1)^{-i} \binom{R}{j-i} i \left(1 - \frac{i}{R}\right), \quad (2.12)
\]

The two more common sharpening polynomials are:

\[
P_{0,0,1,0}(x) = 2x - x^2, \quad (2.13)
\]

\[
P_{0,0,1,1}(x) = 3x^2 - 2x^3. \quad (2.14)
\]

For a simple subfilter with transfer function \( F(z) \), order \( N_f \) (which must be even) and length \( L_f \), the transfer function of the resulting sharpened filter is

\[
H(z) = \delta z^{-\left(R-1\right)N_f/2} F(z) + \sum_{j=n+1}^{R} (\alpha_{j,0} - \sigma \alpha_{j,1} - \delta \alpha_{j,2}) z^{-\left(R-j\right)N_f/2} F(z). \quad (2.15)
\]

### 2.4.2 Frequency Transformation (FT) method

Similar to the sharpening technique presented in the previous subsection, the Frequency Transformation (FT) method introduced in [8] is based on the repetitive use of an identical simple subfilter \( F(z) \). Let us consider \( F(\omega) \) as the zero-phase frequency response of the subfilter and an Amplitude Change Function \( Q(x) \) given as

\[
Q(x) = \sum_{k=0}^{M} q_k x^k. \quad (2.16)
\]

The function \( Q(x) \) allows changing the values \( x = F(\omega) \) to new values \( y = Q(x) \). Basically, the new amplitude values \( y = Q(x) \) must approximate the desired values \( d = D(x) \) for \( x \in X_p \cup X_s \), where \( X_p \) is the range of values \([x_p,l, x_p,u]\) and \( X_s \) is the range of values \([x_s,l, x_s,u]\), such that the zero-phase frequency
response of the overall filter $H(z)$ achieves the desired values $d$ with a maximum absolute pass-band deviation $\delta_p$ over the pass-band region $\Omega_p$, as well as a maximum absolute stop-band deviation $\delta_s$ over the stop-band region $\Omega_s$. This characteristic is reached if the following conditions are simultaneously met,

$$D(x) - \delta_p \leq Q(x) \leq D(x) + \delta_p, \quad \text{for} \quad x_{p,l} \leq x \leq x_{p,u},$$  \hspace{1cm} (2.17) \\
$$D(x) - \delta_s \leq Q(x) \leq D(x) + \delta_s, \quad \text{for} \quad x_{s,l} \leq x \leq x_{s,u},$$  \hspace{1cm} (2.18) \\
x_{p,l} \leq F(\omega) \leq x_{p,u}, \quad \text{for} \quad \omega \in \Omega_p,$$  \hspace{1cm} (2.19) \\
x_{s,l} \leq F(\omega) \leq x_{s,u}, \quad \text{for} \quad \omega \in \Omega_s. \hspace{1cm} (2.20)

Usually, $D(x) = 1$ for $x \in X_p$ and $D(x) = 0$ for $x \in X_s$.

In the Frequency Transformation approach of [8], the polynomial $Q(x)$ is related to the zero-phase frequency response of a Type-I (i.e., with symmetric impulse response and even order) low-pass FIR prototype filter, $G(\phi)$, as follows:

$$G(\phi) = P(a_1 \cos \phi + a_2),$$  \hspace{1cm} (2.21) \\
a_1 = (x_{p,u} - x_{s,l})/2, \hspace{1cm} (2.22) \\
a_2 = (x_{p,u} + x_{s,l})/2, \hspace{1cm} (2.23)

where $\phi$ is the relative frequency domain of the prototype filter. Thus, solving for the conditions (2.17) and (2.18) is equivalent to designing a lowpass prototype filter satisfying:

$$1 - \delta_p \leq G(\phi) \leq 1 + \delta_p, \quad \text{for} \quad 0 \leq \phi \leq \phi_p,$$ \hspace{1cm} (2.24) \\
$$-\delta_s \leq G(\phi) \leq \delta_s, \quad \text{for} \quad \phi_s \leq \phi \leq \pi,$$ \hspace{1cm} (2.25) \\
$$\phi_p = \cos^{-1}\left(\frac{2x_{p,l} - x_{p,u} - x_{s,l}}{x_{p,u} - x_{s,l}}\right), \hspace{1cm} (2.26)$$
\[ \phi_s = \cos^{-1}\left( \frac{2x_{s,u} - x_{p,u} - x_{s,l}}{x_{p,u} - x_{s,l}} \right). \] (2.27)

Note that the way in which \( G(\phi) \) approximates the desired values makes the difference between methods [4]-[6] and [8].

Basically, two cases can be solved from this approach:

**Case 1:** Given \( M \), the number of subfilters, find the optimal coefficients of \( Q(x) \) and the optimal coefficients of \( F(z) \) to meet the conditions (2.17) to (2.20) with the minimum order \( N_f \) (which must be even).

**Case 2:** Given the subfilter \( F(z) \), find the optimal coefficients of \( Q(x) \) to meet the conditions (2.17) to (2.20) with the minimum value \( M \).

The overall filter formed with the FT technique is given as

\[ H(z) = \sum_{k=0}^{M} q_k z^{-(M-k)N_f/2} [F(z)]^k. \] (2.28)

### 2.4.3 Frequency Transformation for Hilbert transformers

A Hilbert transformer is a special class of filter that introduces a \( \pi/2 \) phase shift in the input signal and whose ideal frequency response is [102]

\[ H(e^{j\omega}) = H(\omega)e^{j\theta(\omega)}, \] (2.29)

\[ H(\omega) = 1, \quad \theta(\omega) = \begin{cases} -\pi/2 & \text{for } 0 < \omega < \pi \\ \pi/2 & \text{for } -\pi < \omega < 0. \end{cases} \] (2.30)

Hilbert transformers are usually used in telecommunications or vibration analysis, among others [103], [104]. FIR Hilbert transformers with constant group delay have anti-symmetric impulse response \( h(n) \). The \( \pi/2 \) phase shift is realized exactly, with an additional linear phase component required for a causal FIR system. Since the phase requirement in FIR Hilbert transformers is accomplished, the design of a FIR Hilbert transformer consists on finding the
impulse response $h(n)$, for $n = 0$ to $L-1$, with $L$ being the length of the Hilbert transformer, which satisfies the following magnitude response specification,

$$(1 - \delta) \leq \left| H(e^{j\omega}) \right| \leq (1 - \delta) \quad \text{for } \omega_L \leq \omega \leq \omega_H.$$  \hspace{1cm} (2.31)

In (2.31), $\delta$ is the allowed pass-band ripple, $\omega_L$ is the lower pass-band edge and $\omega_H$ is given as $\omega_H = \pi - \omega_L$ if the desired Hilbert transformer is a Type III filter or $\omega_H = \pi$ if it is Type IV. The values $\omega_L$ and $\omega_H$ can be made to approach 0 and $\pi$, respectively, as closely as desired by increasing the length $L$ of the filter. For Hilbert transformers, the value $\omega_L/2\pi$ is considered the transition band. The number of multipliers $m$ required in a direct design of a Hilbert transformer is estimated in terms of the filter length $L$ as

$$m \approx C \cdot L,$$  \hspace{1cm} (2.32)

where $C = 0.25$ if the filter is Type III or $C = 0.5$ if it is Type IV [102].

The FT method developed in [15] to design FIR Hilbert transformers allows designing FIR Hilbert transformers using a tapped cascaded interconnection of repeated simple basic building blocks constituted by two identical subfilters. To this end, two simple filters are required, namely, a prototype filter and a subfilter. The number of times that the subfilter is used, as well as the coefficients used between each cascaded subfilter, depends on the prototype filter. Both, the prototype filter and the subfilter are Hilbert transformers. The former is always a Type IV filter whereas the latter can be a Type III or Type IV filter according to the type of the desired Hilbert transformer.

Since the prototype filter must be a Type IV FIR filter, it has an even length given as $L_P = 2N$, with $N$ integer, and anti-symmetric impulse response of the form $p(2N - 1 - n) = -p(n)$. Its frequency response is expressed as
\[ P(e^{j\Omega}) = e^{-j((2N-1)\Omega/2 - \pi/2)}P(\Omega), \quad (2.33) \]

where \( P(\Omega) \), the zero-phase term, is given by

\[ P(\Omega) = j \cdot \sin\left(\frac{\Omega}{2}\right) \sum_{n=0}^{N-1} \tilde{d}(n) \cos(\Omega n), \quad (2.34) \]

and \( \Omega \) denotes the frequency domain of the prototype filter. The coefficients \( \tilde{d}(n) \) can be obtained directly from the impulse response \( p(n) [102] \).

Using the equivalence \( \cos(\Omega n) = T_n[\cos(\Omega)] \) [15], where \( T_n[x] \) is the \( n \)-th degree Chebyshev polynomial defined with the following recursive formulas,

\[ T_0[x] = 1, \quad T_1[x] = x \quad \text{and} \quad T_n[x] = 2xT_{n-1}[x] - T_{n-2}[x], \quad (2.35) \]

the zero-phase term can be rewritten as

\[ P(\Omega) = j \cdot \sin\left(\frac{\Omega}{2}\right) \sum_{n=0}^{N-1} \alpha(n) \left[\cos(\Omega)\right]^n, \quad (2.36) \]

where \( \alpha(n) \) are obtained from \( \tilde{d}(n) \) using the coefficients of the Chebyshev polynomials. Based on the equivalence given as

\[ \cos(2x) = 1 - 2\sin^2(x) = 1 + 2\left(j \cdot \sin(x)\right)^2, \quad (2.37) \]

the zero-phase term can be expressed by

\[ P(\Omega) = j \cdot \sin\left(\frac{\Omega}{2}\right) \sum_{n=0}^{N-1} \alpha(n) \left[1 + 2\left(j \cdot \sin\left(\frac{\Omega}{2}\right)\right)^2\right]^n. \quad (2.38) \]

Consider the case of a Type III subfilter with odd length given as \( L_G = 2M + 1, \) \( M \) integer, and anti-symmetric impulse response of the form \( g(2M - n) = -g(n) \). Its frequency response is expressed as

\[ G(e^{j\omega}) = e^{-j(2M\omega/2)}G(\omega), \quad (2.39) \]

where \( G(\omega) \) is the zero-phase term, given by

\[ G(\omega) = j \cdot \sum_{n=1}^{M} c(n) \sin(\omega n). \quad (2.40) \]
The coefficients \( c(n) \) can be obtained directly from \( g(n) \) [102]. Note that the term \( G(\omega) \) can be put in (2.38) by using the following expression,

\[
j \cdot \sin\left(\frac{\Omega}{2}\right) = j \cdot \sum_{n=1}^{M} c(n) \sin(\omega n),
\]

resulting in

\[
H(\omega) = j \cdot \sum_{n=1}^{M} c(n) \sin(\omega n) \sum_{n=0}^{N-1} \alpha(n) \left[ 1 + 2 \left( j \cdot \sum_{n=1}^{M} c(n) \sin(\omega n) \right)^2 \right]^{\nu},
\]

where \( H(\omega) \) is the zero-phase term of the overall filter. Therefore, the frequency transformation is obtained from (2.38) and is given by

\[
\Omega = 2 \sin^{-1}\left[ \sum_{n=1}^{M} c(n) \sin(\omega n) \right].
\]

Equation (2.43) implies that the magnitude response of the prototype filter is preserved, but its frequency domain is changed by the subfilter. It is worth highlighting that similar result can be obtained by considering that the subfilter is a Type IV Hilbert transformer.

The transfer function of the overall Hilbert transformer is given as

\[
H(z) = G(z) \sum_{n=0}^{N-1} z^{-2M(N-1-n)} \alpha(n) \left[ F(z) \right]^{\nu}, \quad F(z) = z^{-2M} + 2G^2(z),
\]

with \( G(z) \) being the transfer function of the subfilter. For a desired Hilbert transformer specification expressed as in (2.31), the magnitude response \( |P(\Omega)| \) of the prototype filter must satisfy the following condition,

\[
(1-\delta) \leq |P(\Omega)| \leq (1+\delta), \quad \text{for } \Omega_L \leq \Omega \leq \pi,
\]

with \( \Omega_{low} \) being the lower band-edge frequency of the prototype filter. The magnitude response of the subfilter, \( |G(\omega)| \), must fulfill simultaneously

\[
v_d - \delta_G \leq |G_0(\omega)| \leq 1, \quad \text{for } \omega_L \leq \omega \leq \pi - \omega_L,
\]

\[
v_d = \frac{1}{2} + \frac{1}{2} \sin\left(\frac{\Omega_L}{2}\right), \quad \delta_G = \frac{1}{2} - \frac{1}{2} \sin\left(\frac{\Omega_L}{2}\right).
\]
The design procedure proposed in [15] starts with an arbitrary prototype filter, and then the subfilter is designed accordingly.

2.4.4 Pipelining-Interleaving (PI) structure

The Pipelining-Interleaving (PI) technique developed in [17] provides efficient structures of FIR digital filters to avoid the repetitive use of an identical filter. Suppose that we have two sequences of independent signals, \( x_1(n) \) and \( x_2(n) \), that are filtered by two identical filters \( H(z) \). Thus, two corresponding sequences of independent outputs, \( y_1(n) \) and \( y_2(n) \), are obtained. An alternative form for this purpose is the multirate implementation using \( H(z^2) \) as shown in Figure 2.5. This structure uses a single filter to implement two identical filters. The clock rate for this implementation must be twice the data rate [17]. If only one sequence of input signal is filtered, it is possible to connect the first output sequence \( y_1(n) \) to the second input \( x_2(n) \). In this way, \( H(z^2) \) is used to implement \( H^2(z) \).

![Figure 2.5: Filtering of two independent sequences using a single filter.](image)

The PI structure of Figure 2.5 can be extended to implement the filtering of \( K \) different signals, each one filtered by an identical filter \( H(z) \), with \( K \) being an arbitrary positive integer. From this, it is possible to implement the filtering of one signal with \( K \) identical filters in cascade. Figure 2.6a presents the general structure to filter a signal using one filter \( H(z^K) \). Figure 2.6b shows
the equivalent structure, which consists of the filtering of a signal by a cascade of $K$ identical filters $H(z)$ [17].

In the structure shown in Figure 2.6a, the clock rate of $H(z^K)$ must be $K$ times the data rate. Clearly, for high data rate applications, $K$ must be chosen as a relatively small integer, otherwise a very high clock rate will be required.

![Figure 2.6: Filtering of a sequence with $K$ identical cascaded filters, (a) PI architecture with a single filter, (b) equivalent single-rate structure.](image)

**2.4.5 FRM method for Hilbert transformers**

The Frequency Response Masking (FRM) method for Hilbert transformers consists of designing three simple FIR subfilters, $H_a(z)$, $H_b(z^M)$ and $H_m(z)$, where $H_a(z)$ is a low-order Hilbert transformer, $H_b(z^M)$ is a band-edge shaping Hilbert transformer and $H_m(z)$ is a masking filter [43]. The overall transfer function is given as

$$H(z) = H_b(z^M)H_m(z) + H_a(z).$$  \hspace{1cm} (2.48)

In [43], the overall design is performed by a joint simultaneous optimization. However, this method has the disadvantage of being highly sensitive to the rounding of coefficients. On the other hand, method [42] is
straightforward, based on the usual design of two low-pass filters, a half-band filter \( H_{hb}(z) \) and an ordinary filter \( H_{ma}(z) \), and it is less sensitive to rounding since every subfilter is designed separately. Because of these advantages, let us review method [42]. For this method we have,

\[
H_p(z^M) = 2[H_{hb}(j^Mz^M) - \frac{1}{2}(jz)^{-M(L_{hb}-1)/2}],
\]

(2.49)

\[
H_s(z) = 2(jz)^{-M(L_{ma}-1)/2}B(jz),
\]

(2.50)

\[
H_m(z) = 2C(jz) - (jz)^{-(L_{ma}-1)/2}.
\]

(2.51)

In (2.49) and (2.50), \( M \) is an odd integer, so-called interpolation factor, and \( L_{hb} \) is the length of \( H_{hb}(z) \) given as \( L_{hb} = 4k-1 \) with \( k \) integer. In (2.51), \( L_{ma} \) is the length of \( H_{ma}(z) \), given as \( L_{ma} = 4k+1 \) with \( k \) integer.

If \( M \) is expressed as \( M = 4k+1 \) with \( k \) integer, the pass-band and stop-band edge frequencies of \( H_{hb}(z) \), \( \omega_p \) and \( \omega_s \), as well as the pass-band and stop-band edge frequencies of \( H_{ma}(z) \), \( \theta_p \) and \( \theta_s \), are

\[
\omega_p = [(\pi/2) - \omega_L]M - 2\pi[(\pi/2) - \omega_L]M/2\pi,
\]

(2.52)

\[
\omega_s = \pi - \omega_p,
\]

(2.53)

\[
\theta_p = (\pi/2) - \omega_L,
\]

(2.54)

\[
\theta_s = (\pi/M) + (\pi/2) - \omega_L.
\]

(2.55)

If \( M \) is expressed as \( M = 4k-1 \) with \( k \) integer, we have

\[
\omega_p = 2\pi\left[\frac{(\pi/2) + \omega_L}{M/2\pi}\right] - [(\pi/2) + \omega_L]M,
\]

(2.56)

\[
\omega_s = \pi - \omega_p,
\]

(2.57)

\[
\theta_p = (\pi/2) + \omega_L - (\pi/M),
\]

(2.58)

\[
\theta_s = (\pi/2) + \omega_L.
\]

(2.59)

Finally, in (2.50) and (2.51) we have the filters \( B(z) \) and \( C(z) \), which are related to \( H_{ma}(z) \) as follows,
\[ B(z) = h_{ma} (1 + (L_{ma} - 1)/2) \left[ z^{-(L_{ma} - 1)/2} + z^{1-(L_{ma} - 1)/2} \right] + \]
\[ h_{ma} (3 + (L_{ma} - 1)/2) \left[ z^{3-(L_{ma} - 1)/2} + z^{-1-(L_{ma} - 1)/2} \right] + \ldots \tag{2.60} \]
\[ C(z) = h_{ma} ((L_{ma} - 1)/2) + \]
\[ h_{ma} (2 + (L_{ma} - 1)/2) \left[ z^{2-(L_{ma} - 1)/2} + z^{-2-(L_{ma} - 1)/2} \right] + \]
\[ h_{ma} (4 + (L_{ma} - 1)/2) \left[ z^{4-(L_{ma} - 1)/2} + z^{-4-(L_{ma} - 1)/2} \right] + \ldots \tag{2.61} \]

where \( h_{ma}(n) \) is the impulse response of \( H_{ma}(z) \).

### 2.5 References


[63] G. F. Boudreaux and T. W. Parks, “Thinning digital filters: A piecewise-

[64] S. Chu and S. Burrus, “Efficient recursive realizations of FIR filters, part I:
The filter structures,” *Circuits, Syst. Signal Process.*, vol. 3, no. 1, pp. 2–20,
1984.

[65] S. Chu and S. Burrus, “Efficient recursive realizations of FIR filters, part II:

phase FIR filters with a piecewise-polynomial-sinusoidal impulse

[67] R. Lehto, T. Tauren and O. Vainio, “Recursive FIR filter structures on


[70] A. Jiang, H. K. Kwan and Y. Zhu “Peak-Error-Constrained Sparse FIR
Filter Design Using Iterative SOCP”, *IEEE Trans. Signal Processing*, vol. 60,

[71] Z. U. Sheikh and O. Gustafsson, “Design of sparse non-periodic narrow-
band and wide-band FRM-like FIR filters,” *in proc IEEE 1st Int. Conf. on


This chapter presents research contributions on the design of digital FIR filters based on subfilters. The chapter is started with the development of four proposals that improve the magnitude response of the Recursive Running Sum (RRS) filter, a useful filter characterized by its low complexity. The proposed schemes, especially suitable for low-pass narrowband cases applied in sampling rate conversion, provide an efficient balance between magnitude response improvement and added complexity to the RRS filter in comparison with previous schemes developed in literature. We then present the efficient design of Hilbert transformers, which are a special class of wideband filters. Proper combinations of identical-subfilter-based and periodical-subfilter-based schemes are proposed and it is shown that these combinations provide low-complexity filtering solutions.
3.1 Contributions on narrowband filters

Narrowband filtering is required in several parts of any communication system. A traditional example, due to its practical applications, is the required filtering in decimation or interpolation processes [1]-[34]. Narrowband filters are also required as a part of filter banks, where it is necessary to extract the information contained in several narrowband channels from a wideband input signal [35]-[41]. Additionally, these filters can be used as subsystems in the schemes of efficient design of FIR filters with arbitrary passband bandwidths [42]-[54].

In this section we introduce novel design schemes based on the Recursive Running Sum (RRS) filter. The proposed schemes are focused on the improvement of the magnitude response of RRS filters. The resulting filters can be used to design efficient narrowband FIR filters, whose main applications are in sampling rate conversion. In first place we present the definition and usefulness of RRS filters, along with the simple compensation scheme from [5], useful to decrease the passband droop in these filters. A simple and efficient algorithm to design RRS-based filters, which combines these revised filters with sharpening polynomials, is developed in sub-section 3.1.2. Then, in sub-section 3.1.3 we formulate an improved optimization-based sharpening approach to attain given specifications on the magnitude response characteristics of RRS filters. The paradigm of magnitude improvement in RRS filters with sharpening polynomials is changed by introducing the corrector filter in sub-section 3.1.4. The corrector filter is introduced as a low-order linear-phase filter that improves only the worst-case magnitude values, i.e., decreases the passband droop and increases the
attenuation around the first stopband. Finally, the design of low-complexity compensators for RRS-based filters using simple cosine-squared subfilters is introduced in Section 3.1.5. An efficient combination of the Chebyshev sharpening, recently introduced in [27], with these compensators is derived to improve the worst-case magnitude characteristics of traditional RRS filters.

### 3.1.1 Introduction to Recursive Running Sum filters

The simplest low-pass Finite Impulse Response (FIR) filter is the $M$-point moving-average filter, whose impulse response is given as [26]

$$h(n) = \begin{cases} \frac{1}{M}, & \text{for } 0 \leq n \leq M - 1; \\ 0, & \text{otherwise}, \end{cases} \quad (3.1)$$

where $M$ is an integer. Its transfer function is given by

$$H(z) = \frac{1}{M} \sum_{k=0}^{M-1} z^{-k}. \quad (3.2)$$

The scaling factor $1/M$ is needed to provide a gain of 0 dB at the zero frequency. This filter is also known as boxcar filter.

A very convenient form of the above transfer function for realization purpose is given by

$$H(z) = \frac{1}{M} \frac{1 - z^{-M}}{1 - z^{-1}}. \quad (3.3)$$

A filter that implements this transfer function is known as a Recursive Running Sum (RRS) filter. Its frequency response is given by

$$H(e^{j\omega}) = H(\omega)e^{-j\omega(M-1)/2}, \quad (3.4)$$

where

$$H(\omega) = \begin{cases} \frac{1}{M} \frac{\sin(\frac{\omega M}{2})}{\sin(\frac{\omega}{2})}, & \text{for } 0 < \omega < \pi. \end{cases} \quad (3.5)$$
is its zero-phase frequency response. Usually, $K$ RRS filters are cascaded to improve the magnitude characteristic in the stopband region, even though this degrades the magnitude in passband.

RRS filters were introduced in [1] as efficient filtering structures where their main application was as anti-aliasing and anti-imaging suppressors in the processes of decimation by $M$ and interpolation by $M$, respectively. By applying multirate noble identities, the comb section can be moved to the low sampling rate side. Thus, the structure proposed in [1] consists of $K$ cascaded integrator sections with transfer function $1/(1-z^{-1})$ working at high sampling rate and $K$ cascaded comb sections with transfer function $(1-z^{-1})$ working at a low sampling rate. This structure is commonly known as Cascaded-Integrator-Comb (CIC).

The main advantages of an RRS filter are the following:

- The RRS filter is a symmetric FIR recursive system with perfect pole-zero cancellation [55], so it has linear-phase and guaranteed stability.

- The RRS filter is multiplier-free and it only requires two adders. Hence, it can be used to design low-complexity multiplierless filters.

- In the frequency range from $\omega=0$ to $\omega=\pi$, the zeros of an RRS filter are placed over the frequencies $\omega_k = 2\pi k/M$, for $k = 1, 2, \ldots, \lfloor M/2 \rfloor$ ($\lfloor x \rfloor$ denotes the integer part of $x$). These zeros provide stopband regions with natural rejection to aliasing introduced by the decimation process, imaging introduced by the interpolation process or imaging introduced by an expanded-by-$M$ filter in the Interpolated-FIR (IFIR) scheme.
Nevertheless, the magnitude response of an RRS filter is, in general, poor. RRS filters have the following disadvantages:

- The passband exhibits a droop that increases with $K$, the number of cascaded RRS filters.
- The stopband regions are very narrow. These regions can be widened by increasing $K$.

As a consequence, RRS filters are commonly used as multi-band filters, with one passband and several narrow stopband regions. This magnitude characteristic makes the RRS filters especially useful either in multi-stage decimation and interpolation processes or as anti-imaging filters in IFIR filtering schemes. Considering the aforementioned applications of an RRS filter, its passband region $\omega_p$ and its stopband regions $\omega_{s,k}$ can be defined as

$$\omega_p = \left[0, \omega_p\right],$$

$$\omega_{s,k} = \left[\frac{2\pi k}{M} - \omega_p, \frac{2\pi k}{M} + \omega_p\right], \quad k = 1, 2, ..., \left\lfloor M/2 \right\rfloor,$$

where

$$\omega_p \leq \pi/(2M)$$

is an arbitrary passband edge frequency dependent on the application. Figure 3.1 shows the magnitude responses of RRS filters with $M = 8$ and $K = 1, 2, 3$ and 4, with $\omega_p = \pi/(2M)$. From Figure 3.1 it is possible to note the passband droop and also that, beginning with the first stopband, the attenuations in all the stopbands are increasingly higher. Therefore, the worst-case magnitude response value in the passband occurs at the right-most edge of the passband, i.e., at the frequency value $\omega_p$. The worst-case in the stopband occurs in the leftmost edge of the first stopband, i.e., at the frequency value $(2\pi/M) - \omega_p$ [1], [28].
Owing to their reduced computational complexity, research on RRS filters to date has been focused on

1) improving the magnitude characteristic of RRS filters,
2) preserving linearity of phase and
3) having the least possible increase of computational complexity [2]-[31].

The research has been mainly motivated by the application of RRS filters in decimation processes. From the representative work on this subject, we can classify 3 main approaches, detailed in the following.

1. **Passband improvement [2]-[9]**: In this case, the attenuation in the stopband regions is arbitrarily adjusted by varying $K$, the number of cascaded filters. Thus, the passband droop of the RRS filters is considered the main problem. The solutions are based on the design of simple expanded-by-$M$ compensation filters. For RRS decimation filters, these compensators work at the lower sample rate section, along with the comb filtering. The compensation filters should be easily adaptable to distinct passband compensation requirements, such that the remaining filters used for the residual decimation factor (in the
subsequent stages) do not need to be programmable for droop compensation.

2. **Stopbands Improvement** [10]-[17]: In this case, the attenuation in the stopband regions of the RRS filters is considered the main problem. The solutions improve the attenuation in the stopbands by shifting collocated zeros apart from each other. These proposals include zero-rotation techniques and modified-structure techniques.

2.1. **Zero-rotation techniques** [10]-[12]: In these techniques, the collocated zeros can be explicitly moved apart. Every pair consisting of 2 cascaded RRS filters is replaced by a more general recursive filter with two coefficients. The resulting filter is often referred as Rotated-Sinc filter.

2.2. **Modified-structure techniques** [13]-[17]: In the modified-structure techniques, the cascaded RRS filters are aided with either additional coefficients or with additional filters. These techniques are especially useful for sampling rate conversion applications. In [13]-[14], the CIC structure is aided with embedded filters. The structure proposed in [15] is derived from the structure of sigma-delta modulators. However, this method has the drawback of requiring multipliers at the high-rate side. In [16] the RRS filters are cascaded in a similar way to the tapped cascaded interconnection of identical subfilters [43]. For sampling rate conversion cases, the structure can be rearranged to exploit the CIC principle of performing some computations at the lower output sample rate. Embedding an expanded filter in such structure was proposed in
[17], but the resulting scheme, requiring fractional coefficients, has a high hardware complexity even though mainly all the arithmetic operations are performed at lower rate.

3. **Passband-Stopband Improvement [18]-[31]:** In this case, the passband droop and the stopband attenuation in the stopband regions are considered in the same way as main problems. Particularly, some of these methods are focused on improving mainly the magnitude characteristics over the passband and the first stopband, where the worst-case attenuation occurs. Among the existing methods to simultaneously improve the magnitude characteristics of RRS filters in both, passband and stopbands, it is possible to list the following.

3.1 *Methods based on separated improvements [18]-[23]:* In methods [18]-[19], [21] the improvement in the stopbands is realized using the zero-rotation technique, whereas the passband characteristic is improved with compensators. The recursive rotated-sinc filter of [19] allows getting multiplierless coefficients and guarantees perfect pole-zero cancelation. However, the resulting filter coefficients require huge word-lengths. In [21] a two-stage non-recursive structure is proposed instead, with the rotated-sinc filter working at lower rate. The method [20] uses compensators to improve the passband characteristic and applies an additional zero near to the lower edge of the first stopband, but this provokes a high gain over the other stopbands that is decreased only with additional filtering at high rate. In [22]-[23], a two-stage scheme is introduced where the first stopbands are improved by increasing
the order of the second-stage RRS filter and the passband droop is decreased with a compensator.

3.2 Methods based on sharpening structure [24]-[27]: In methods [24]-[26] the magnitude improvement is obtained by using the well-known sharpening technique of Kaiser and Hamming [42]. The recent Chebyshev sharpening of Coleman [27] introduces equiripple stopbands with a Chebyshev polynomial and passband improvement with either a pre-sharpening polynomial or a Chebyshev-based polynomial that has either a 1st- or 2nd-degree tangency at the point (1,1) to the horizontal line.

3.3 Mixed methods [28]-[31]: In [28] the Kaiser-Hamming sharpening technique was combined with rotated-sinc filters, such that the resulting overall filter achieves improved magnitude in stopband regions due to both, the sharpening and the zero-rotation, whereas the magnitude in pasband is improved due to sharpening. Methods [29]-[31] apply the simplest Kaiser-Hamming sharpening polynomials to previously compensated RRS filters, thus achieving an improved magnitude in passband due to both, the sharpening and the compensation, and an improved magnitude in stopband due to the increase of the number of cascaded RRS filters, $K$.

3.1.1.1 Compensated RRS filters

The simplest compensation filter to improve the passband characteristic of RRS filters is the one proposed in [5]. This filter is the expanded-by-$M$ version of a second order filter whose transfer function is given by
\[ C(z) = -2^{-(b+2)} \left[ 1 - (2^{b+2} + 2)z^{-1} + z^{-2} \right]. \] (3.9)

Note that this filter requires only three adders. The parameter \( b \) can be chosen in relation to the number of cascaded RRS filters as given in Table 3.1.

<table>
<thead>
<tr>
<th>Parameter ( K )</th>
<th>Parameter ( b ) (narrowband case)</th>
<th>Parameter ( b ) (wideband case)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>6</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

The compensated RRS filter has a transfer function given by

\[ H_c(z) = H^K(z)C(z^M) = \frac{z^{-2b}}{M^r} \left( \frac{1-z^{-M}}{1-z^{-1}} \right)^K \left[ 1 - (2^{b+2} + 2)z^{-M} + z^{-2M} \right] \] (3.10)

and its frequency response is given by

\[ H_c(e^{j\omega}) = H_c(\omega) = e^{-j\omega[M-1]K/M} \] (3.11)

where

\[ H_c(\omega) = \left\{ \frac{1}{M} \sin \left( \frac{\omega M}{2} \right) \right\}^K \left[ 1 + 2^{1-b} \sin^2 \left( \frac{\omega M}{2} \right) \right] \] (3.12)

is the zero-phase frequency response.

It is worth highlighting that it can be possible to form other compensated RRS filters because there are several compensators in literature. However, the pair RRS-compensator that constitutes \( H_c(z) \) in (3.10), referred here as
compensated RRS filter, is the simplest RRS-based filter with an improved passband characteristic.

3.1.2 Efficient sharpening-based scheme of compensated RRS filters

From the representative sample of works that improve the magnitude characteristics of RRS filters, the rotated-sinc-based schemes [10], [11], [18] have the disadvantage of being susceptible to imperfect pole-zero cancelation. An effective way to prevent this problem consists in designing non-recursive filters [12], [21] with filtering implemented in polyphase form for ensuring power savings. However, this can result in a higher demand of chip area. On the other hand, improving the passband with low-order compensators and stopbands directly with the number of cascaded RRS filters provides low-complexity solutions, but the passband improvement can not be completely controlled, unless that the order of the compensators is increased. Therefore, these methods are convenient when the desired magnitude characteristics are not too stringent, and when the bandwidth of interest is narrow.

In this subsection we introduce a simple and efficient technique to improve the magnitude characteristic of compensated RRS filters. The proposal consists on applying any of the two simple sharpening polynomials $P_{0,0,1,0}(x)$ or $P_{0,0,1,1}(x)$ (see (2.13) and (2.14)) to the compensated RRS filter. The motivation that supports this proposal relies in the following observations:

- The use of the sharpening polynomial $P_{0,0,1,0}(x)$ has been recently proposed in [29]. From that work it can be seen that sharpening RRS-
based filters offers significant advantages mainly for sampling rate conversion processes, perhaps the most common application of RRS filters. In these cases, we can take advantage of the recursive form of RRS filters to obtain a CIC-like architecture that places a great part of the filtering at lower rate. Additionally, this structure has all the sharpening coefficients at lower rate and a simple overall structure is obtained, which does not suffer of finite-precision effects as rotated-sinc-based methods.

- The compensated RRS filter preserves its simplicity since only three extra adders are required, regardless of the number of cascaded RRS filters.
- The compensated RRS filter has an improved passband magnitude response in comparison to the original RRS filter, which yields a better sharpened response over the passband region. Besides, the stopband characteristic can be arbitrarily improved by increasing the number of cascaded RRS filters.
- The sharpening polynomials with parameters \((\sigma, \delta, m, n) = (0, 0, 1, 0)\) and \((\sigma, \delta, m, n) = (0, 0, 1, 1)\) are the simplest polynomials that can improve passband and stopband regions of the original compensated RRS filter, thus there is not excessive increase of the complexity of the resulting filter.

For any of the two polynomials, \(P_{0,0,1,0}(x)\) and \(P_{0,0,1,1}(x)\), the tangency \(m = 1\) allows attaining an improved passband characteristic. This improvement depends on the passband deviation of the original compensated RRS filter, which is controlled by the parameter \(b\). On the other hand, the improvement
over the stopband regions can be controlled with the number of cascaded RRS filters. Additionally, the sharpening polynomial with tangency \( n = 1 \) improves also the stopband regions and thus in this case a smaller value \( K \) can be used. Since the values for \( b \) corresponding to a given \( K \) have been established in [5], we just need to find the proper parameter \( K \) and the sharpening polynomial to be used.

It is worth highlighting that any arbitrary magnitude constraint at the stopband regions can be attained with a proper increase of \( K \). Thus, the appropriate polynomial and the corresponding value for \( K \) should be those that allow accomplishing the given magnitude stopband constraint with the minimum complexity in terms of the number of adders required by the resulting sharpened filter.

For simplicity, let us identify the two sharpening polynomials \( P_{0,0,1,n}(x) \), with \( n = 0 \) or \( n = 1 \), as \( P_n(x) \). Thus, \( P_0(x) = P_{0,0,1,0}(x) \) and \( P_1(x) = P_{0,0,1,1}(x) \). With these two values \( n \) we use the following notation:

- \( H_n(z) \) is the resulting transfer function after sharpening. Clearly, we have either \( H_0(z) \) or \( H_1(z) \) as proposed filters.
- \( K_n \) is the number of cascaded RRS filters in \( H_n(z) \), i.e., the value that \( K \) takes when \( H_n(z) \) is sharpened by \( P_n(x) \).
- \( A_n \) is the number of adders in the filter \( H_n(z) \).

The transfer functions of the proposed filters are given by

\[
H_0(z) = H_1(z) \left[ 2z^{-1} - H_n(z) \right], \tag{3.13}
\]

\[
H_1(z) = H_1^2(z) \left[ 3z^{-1} - 2H_n(z) \right], \tag{3.14}
\]

\[
\tau = (M-1)K / 2 + M. \tag{3.15}
\]
$H_c(z)$ is given in (3.10) and $\tau$ is the delay of $H_c(z)$, introduced to keep the linear phase in the resulting sharpening-based structure. Note that $K$ has to be even for an even $M$ to avoid a half-sample delay. The respective frequency responses are given by

$$H_0(e^{j\omega}) = [2H_c(\omega) - H_c^2(\omega)]e^{-j\omega(M-1)K+2M}, \quad (3.16)$$

$$H_1(e^{j\omega}) = [3H_c^2(\omega) - 2H_c^3(\omega)]e^{-j\omega3(M-1)K/2+M}, \quad (3.17)$$

where $H_c(\omega)$ is given in (3.12). The proposed structures are presented in Figure 3.2.

![Diagram](image)

**Figure 3.2:** Proposed structures to improve the magnitude characteristics of RRS filters.

The numbers of adders of these filters are

$$A_0 = 4K_0 + 7, \quad (3.18)$$

$$A_1 = 6K_1 + 11. \quad (3.19)$$

Observe that a relation between $A_0$ and $A_1$ can be expressed in terms of the values $K_0$ and $K_1$ and it may result in either of the following cases:

- For certain values $K_0$ and $K_1$ we have $A_0 = A_1$.  

For certain values $K_0$ different from the ones mentioned in the previous observation, there are values $K_1$ that yield either the relation $A_0 < A_1$ or the relation $A_0 > A_1$. Based on these cases, we can introduce a simple systematic procedure to find the suitable value $K_0$ and the corresponding polynomial $P_n(x)$.

If the numbers of adders are equal, from (3.18) and (3.19) we have

$$4K_0 + 7 = 6K_1 + 11,$$  \hspace{1cm} (3.20)

$$K_0 = \frac{3}{2}K_1 + 1.$$  \hspace{1cm} (3.21)

Since $K_0$ and $K_1$ have to be integers, (3.21) is satisfied if $K_1$ accomplishes the condition

$$K_1 = 2k, \ k \text{ integer}.$$  \hspace{1cm} (3.22)

Substituting (3.22) in (3.21) we have

$$K_0 = 3k + 1, \ k \text{ integer}.$$  \hspace{1cm} (3.23)

When the numbers of adders are different, we have either

$$4K_0 + 7 < 6K_1 + 11,$$  \hspace{1cm} (3.24)

or

$$4K_0 + 7 > 6K_1 + 11.$$  \hspace{1cm} (3.25)

In these cases $K_1$ must satisfy the condition

$$K_1 = 2k + 1, \ k \text{ integer}.$$  \hspace{1cm} (3.26)

Substituting (3.26) in (3.24) we have

$$K_0 < 3k + \frac{10}{3},$$  \hspace{1cm} (3.27)

and we obtain

$$K_0 = 3k + \lfloor \frac{10}{3} \rfloor = 3k + 2, \ k \text{ integer}.$$  \hspace{1cm} (3.28)

On the other hand, substituting (3.26) in (3.25) we have
\[ K_0 > 3k + \frac{10}{4}, \quad (3.29) \]

and we obtain

\[ K_0 = 3k + \left\lceil \frac{10}{4} \right\rceil = 3k + 3, \quad k \text{ integer}. \quad (3.30) \]

Equations (3.22), (3.23), (3.26), (3.28) and (3.30) are functions of the parameter \( k \), and they give the three possible values of \( K_0 \) and the two possible values of \( K_1 \) for a given integer \( k \). By progressively substituting each of the equations (3.23), (3.28), and (3.30) in (3.18) and each of the equations (3.22) and (3.26) in (3.19) we find the following relations for a given \( k \),

\[
(6K_1 + 11)_{K_0 = 3k + 1} < (4K_0 + 7)_{K_0 = 3k + 2} < (6K_1 + 11)_{K_0 = 3k + 3}. \quad (3.31)
\]

By using every expression for \( K_0 \) in (3.31) from left to right we get a manner to systematically increase either \( K_0 \) or \( K_1 \) such that, for every integer \( k \), the stopband is improved with a minimum increase of the number of adders.

As an example, consider \( k = 0 \). Since \( K = K_1 = 2k = 0 \), we first use \( K = K_0 = 3k + 1 = 1 \) in \( H_c(z) \) and we sharpen \( H_c(z) \) with the polynomial \( P_0(x) \) to obtain \( H_0(z) \). We go ahead with \( K = K_0 = 3k + 2 = 2 \), then with \( K = K_1 = 2k + 1 = 1 \) (in this case we sharpen \( H_c(z) \) with polynomial \( P_1(x) \) to obtain \( H_1(z) \)) and finally with \( K = K_0 = 3k + 3 = 3 \). If in some of these substitutions of \( K \) the desired stopband is met, we have found the solution. Otherwise, we repeat these steps with the next integer \( k \), i.e., \( k = 1 \) in this simple example.

The aforementioned method to obtain \( K \) by means of the increase of \( k \) can be formalized in the following way. Consider

\[ k = a k + b, \quad (3.32) \]

where
\[ \mathbf{a} = \begin{bmatrix} 2 & 3 & 3 & 2 & 3 \end{bmatrix}^T, \quad (3.33) \]
\[ \mathbf{b} = \begin{bmatrix} 0 & 1 & 2 & 1 & 3 \end{bmatrix}^T. \quad (3.34) \]

For every non-negative integer \( k \), starting with \( k = 0 \), the vector \( \mathbf{k} \) in (3.32) contains five possible values for \( K \). The stopband improvement with the minimum increase in the number of adders is obtained by evaluating \( \mathbf{k} \) row-wise and assigning the obtained value (either \( K_0 \) or \( K_1 \)) to \( K \). This can be expressed as \( K = [\mathbf{a}]_e \cdot k + [\mathbf{b}]_e \), where \([\mathbf{a}]_e \) and \([\mathbf{b}]_e \) mean the \( e \)-th element of \( \mathbf{a} \) and the \( e \)-th element of \( \mathbf{b} \), respectively. When \([\mathbf{a}]_e = 3 \), we obtain \( K = K_0 \) and thus the sharpening polynomial \( P_0(x) \) must be used to obtain \( H_0(z) \). When \([\mathbf{a}]_e = 2 \), we obtain \( K = K_1 \) and the sharpening polynomial \( P_1(x) \) must be used to obtain \( H_1(z) \).

The proposed procedure, shown in the flowchart of Figure 3.3, is described in the following steps:

1. If \( M \) is odd, set \( \mathbf{a} = [2 \ 3 \ 3 \ 2 \ 3]^T \), \( \mathbf{b} = [0 \ 1 \ 2 \ 1 \ 3]^T \) and \( r = 5 \), where \( r \) is the number of rows of \( \mathbf{a} \) and \( \mathbf{b} \). If \( M \) is even, set \( \mathbf{a} = [2 \ 3 \ 3 \ 3]^T \), \( \mathbf{b} = [0 \ 1 \ 2 \ 3]^T \) and \( r = 4 \).
2. Set \( k = 0 \) and \( e = 1 \), where \( e \) is an integer used as pointer and \( k \) is the parameter that yields the better choice of \( K \).
3. Set \( K = [\mathbf{a}]_e \cdot k + [\mathbf{b}]_e \). If \( K = 0 \), set \( e = e + 1 \) and obtain \( K \) once more.
4. If \([\mathbf{a}]_e = 2 \) set \( n = 1 \). If \([\mathbf{a}]_e = 3 \) then check \( M \) and \( K \). If \( M \) is odd or \( K \) is even, set \( n = 0 \). Otherwise go to step 7.
5. Apply sharpening to the \( K \) cascaded compensated RRS filters using the polynomial \( P_n(x) \) with the values \( K \) and \( n \) obtained in previous steps.
6. If the stopband filter specification is not reached go to step 7. Otherwise, stop.
7. If \( e = r \), then set \( e = 1, k = k + 1 \) and go to step 3; otherwise, set \( e = e + 1 \) and go to step 3.

![Flowchart of the algorithm to find the proper \( K \) and \( n \) to form \( H_n(z) \).](image)

**Figure 3.3:** Flowchart of the algorithm to find the proper \( K \) and \( n \) to form \( H_n(z) \).

The proposed filters \( H_0(z) \) and \( H_1(z) \) can be used instead of cascaded RRS filters as aliasing suppressors when a decimation by \( M \) is involved. In such case, efficient CIC-like structures can be derived from the proposed filters. Figure 3.4 shows the decimation structures derived from \( H_0(z) \) and \( H_1(z) \).
which will be referred to as Structure-0 and Structure-1, respectively. The following notation is used in that figure:

- \( H_1(z) = 1/(1 - z^{-1})^K \) is the integrator section of \( H_c(z) \).
- \( H_c(z) = (1 - z^{-1})^K \) is the comb section of \( H_c(z) \).
- \( C(z) \) is the compensation filter (see (4.9)).

It is worth highlighting that Structure-0 has been recently proposed in [29].

![Proposed structures for decimation by M.](image)

**Figure 3.4:** Proposed structures for decimation by \( M \).

Besides of the number of adders, the complexities of Structure-0 and Structure-1 can be given in terms of the number of Additions Per Output Sample (APOS) and the amount of memory elements required. For Structure-0 we have

\[
APOS_0 = 2(M + 1)K_0 + 7, \quad \text{(Additions Per Output Sample)} \quad (3.35)
\]

\[
ME_0 = 5K_0 + 5, \quad \text{(Memory Elements)} \quad (3.36)
\]

and for Structure-1,

\[
APOS_1 = 3(M + 1)K_1 + 11, \quad \text{(Additions Per Output Sample)} \quad (3.37)
\]

\[
ME_1 = 7K_1 + 7, \quad \text{(Memory Elements)} \quad (3.38)
\]
Note that either $K_0$ or $K_1$ must be even to avoid fractional delay.

### 3.1.2.1 Design examples and discussion of results

One of the most important applications of the proposed structures in communication systems is for anti-aliasing filtering in the first stage of a multi-stage decimation chain. In the following paragraphs we present three design examples, focused on these applications, to show the effectiveness of the proposed sharpening-based approach in comparison to other works introduced in literature.

**Example 1:** Consider a two-stage decimation process with decimation factor $D = M
u$, $M = 11$ and $\nu = 2$. Design the decimation filter that will be used in the first decimation stage with the following specification [29],

- passband, $\bar{\omega}_p = [0, 0.0273\pi]$, 
- stopbands, $\bar{\omega}_k = [0.182\pi k-0.0273\pi, 0.182\pi k+0.0273\pi], k = 1, 2, ..., 5,$
- $50 \text{ dB} \text{ attenuation in stopbands}$. 

With the proposed procedure we obtain $K = 2$ and $n = 1$. Thus, the magnitude requirement is accomplished with the filter $H_1(z)$. The number of adders is $A_1 = 23$. The solution provided in the recent proposal [29] is with the filter $H_0(z)$, using $K = 4$, and requires $A_0 = 23$ adders. The magnitude responses of both filters are presented in Figure 3.5. Note that the magnitude responses $|H_0(e^{j\omega})|$ and $|H_1(e^{j\omega})|$ are very similar and both filters require the same number of adders. The proposed efficient architecture for decimation is Structure-1. In [29], the architecture for decimation is the Structure-0. Observe
that, even though both solutions use the same number of adders, the number of memory elements in the proposed solution is slightly lower. We have $ME_1 = 21$ and $ME_0 = 25$.

![Passband detail](image1.png)

**Figure 3.5**: Magnitude responses $|H_0(e^{j\omega})|$ (solution given in [29]) and $|H_1(e^{j\omega})|$ in Example 1.

A more important comparison is made in terms of APOS. For the proposed solution we have $APOS_1 = 83$ and for the solution from [29] we have $APOS_0 = 103$. Therefore, the proposed solution achieves a lower computational complexity even though the number of adders in both structures is the same. This observation deserves special attention since, at first glance, Structure-0 seems to be less complex. Thus, let us analyze Structure-1 and Structure-0 in more detail.

First, we should recall that the value $K$ must be even to avoid fractional delay. On the other hand, from (3.20) and (3.21) we have that, for every even value $K = K_1$ used in Structure-1, we have a Structure-0 with the same number of adders required in Structure-1 if we use $K = K_0 = 3K_1/2 + 1$. If $K_1$ is identified by $K_1 = 2k$, we have, from (3.35) and (3.37),
\[ APOS_0 = (M+1)(6k) + 2(M+1) + 7, \quad (3.39) \]
\[ APOS_1 = (M+1)(6k) + 11. \quad (3.40) \]

Note that \( APOS_0 \) is greater than \( APOS_1 \) by \( 2(M+1) - 4 \). This value is always a positive integer, except when \( M = 1 \) (which is the trivial single rate case). Therefore, for \( M \geq 2 \), \( APOS_0 \) will be always greater than \( APOS_1 \) provided that the numbers of adders, \( A_0 \) and \( A_1 \), are equivalent.

Regarding to the number of memory elements, we can repeat the previous assumptions on \( K_0 \) and \( K_1 \) and, from (3.36) and (3.38) we obtain
\[ ME_0 = 15k + 10, \quad (3.41) \]
\[ ME_1 = 14k + 7. \quad (3.42) \]

Clearly, \( ME_0 \) is always greater than \( ME_1 \) by \( k + 3 \), provided that the numbers of adders, \( A_0 \) and \( A_1 \), are equivalent. The importance of developing the previous comparison relies on the fact that the proposed method chooses among the two filters, \( H_0(z) \) and \( H_1(z) \), by comparing their numbers of adders. However, when this metric is equal, the method gives priority on using first the filter \( H_1(z) \), which in fact provides an improved magnitude similar to the one of its counterpart \( H_0(z) \) but with a lower complexity in the resulting decimation structure. Finally, Table 3.2 summarizes the comparison of results with method [29].

<table>
<thead>
<tr>
<th>Method</th>
<th>Number of adders</th>
<th>Number of memory elements</th>
<th>APOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>[29]</td>
<td>23</td>
<td>25</td>
<td>103</td>
</tr>
<tr>
<td>Proposed</td>
<td>23</td>
<td>21</td>
<td>83</td>
</tr>
</tbody>
</table>

Table 3.2: Comparison of results in Example 1
**Example 2:** Design a decimation filter $H(z)$ for decimation by $D = 10$ with the following specification [35]:

- passband, $\tilde{\omega}_p = [0, 0.05\pi]$,
- stopband, $\tilde{\omega}_s = [0.1\pi, \pi]$,
- passband ripple, $R_p = 0.1737$ dB,
- stopband attenuation, $A_s = 60$ dB.

Let us factorize the decimation factor as $D = M\nu$, with $M = 5$ and $\nu = 2$. From the IFIR design method, the decimation filter can be constituted by two subfilters, $H_a(z)$ and $H_b(z^M)$, such that the proposed decimation filter is: $H_p(z) = H_a(z)H_b(z^M)$.

We design the filter $H_a(z)$ with a passband $\tilde{\omega}_{p,a} = [0, 0.05\pi]$ and with stopbands $\tilde{\omega}_{1,a} = [0.4\pi-0.05\pi, 0.4\pi+0.05\pi]$ and $\tilde{\omega}_{2,a} = [0.8\pi-0.05\pi, 0.8\pi+0.05\pi]$. The filter must have an attenuation of 60 dB in its stopbands. On the other hand, we design the filter $H_b(z)$ with passband $\tilde{\omega}_{p,b} = [0, 0.25\pi]$ and with stopband $\tilde{\omega}_{s,b} = [0.5\pi, \pi]$. The passband ripple is $R_{p,b} = 0.17$ dB and its stopband attenuation $A_{s,b} = 60$ dB.

The filter $H_a(z)$ is designed with the proposed procedure. We obtain $K= 6, n=0$. This means $H_a(z) = H_0(z)$. This filter uses 31 adders. The filter $H_b(z)$ is an ordinary low-pass filter with order $N = 21$. This filter uses 21 adders and 11 multipliers. The magnitude response of the resulting filter $H_p(z) = H_a(z)H_b(z^M)$ is presented in Figure 3.6.

The efficient architecture for this decimation process consists on using Structure-0 in the first decimation stage. By multirate identity, $H_b(z^M)$ can be moved to a lower rate after the decimation by $M$. The overall system
performs 200 APOS and 22 Multiplications Per Output Sample (MPOS) and it requires 11 multipliers, 52 adders and 56 memory elements.

For the same example, in [35] is proposed a solution in the form $H(z) = H_e(z)H_f(z^D)$, with $H_e(z)$ having an order $N_e = 38$ and $H_f(z)$ having an order $N_f = 13$. An efficient architecture for this decimation process consists on moving the filter $H_f(z^D)$ after the decimation by $D$. The overall system performs 393 APOS and 207 Multiplications Per Output Sample (MPOS) and it requires 27 multipliers, 51 adders and 51 memory elements.

Table 3.3 summarizes the aforementioned results.

<table>
<thead>
<tr>
<th>Method</th>
<th>Number of multipliers</th>
<th>Number of adders</th>
<th>Number of memory elements</th>
<th>MPOS</th>
<th>APOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>[35]</td>
<td>27</td>
<td>51</td>
<td>51</td>
<td>207</td>
<td>393</td>
</tr>
<tr>
<td>Proposed</td>
<td>11</td>
<td>52</td>
<td>56</td>
<td>22</td>
<td>200</td>
</tr>
</tbody>
</table>

Figure 3.6: Magnitude response of the proposed filter $H_p(z)$ in Example 2.
**Example 3:** Design a decimation filter \( H(z) \) for decimation by \( D = 16 \) with the following specification \([39]\):

- passband, \( \tilde{\omega}_p = [0, 0.05\pi] \),
- stopband, \( \tilde{\omega}_s = [0.0625\pi, \pi] \),
- passband ripple, \( R_p = 0.5 \text{ dB} \),
- stopband attenuation, \( A_s = 55 \text{ dB} \).

For this filter we use again the IFIR design method. The proposed decimation filter is:

\[
H_p(z) = H_a(z) H_b(z^M),
\]

where \( M = 8 \), \( \nu = 2 \) and \( D = M\nu \).

The filter \( H_a(z) \) has a passband given by \( \tilde{\omega}_{p,a} = [0, 0.05\pi] \) and stopbands \( \tilde{\omega}_{k,a} = [0.25\pi k - 0.05\pi, 0.25\pi k + 0.05\pi] \) for \( k = 1, 2, 3 \), and \( \tilde{\omega}_{4,a} = [\pi - 0.05\pi, \pi] \). This filter must have an attenuation of 55 dB in its stopbands. The filter \( H_b(z) \) has a passband given by \( \tilde{\omega}_{p,b} = [0, 0.4\pi] \) and a stopband given by \( \tilde{\omega}_{s,b} = [0.5\pi, \pi] \). The passband ripple is \( R_{p,b} = 0.5 \text{ dB} \) and its stopband attenuation is \( A_{s,b} = 55 \text{ dB} \).

We use the proposed procedure to design \( H_a(z) \). We obtain \( K = 6 \) and \( n = 0 \). Thus the filter is designed as the proposed \( H_a(z) \) and it requires 31 adders. For the filter \( H_b(z) \) we obtain an order \( N = 44 \). This filter uses 44 adders and 23 multipliers. The magnitude response of the resulting filter \( H_p(z) = H_a(z) H_b(z^M) \) is presented in Figure 3.7.

The proposed architecture for decimation consists on using Structure-0 in the first decimation stage. The filter \( H_b(z^M) \) can be moved to a lower rate after the decimation by \( M \). The overall system performs 318 APOS, 46 MPOS and it requires 23 multipliers, 75 adders and 79 memory elements.

For the same example, in \([39]\) is proposed a solution in the form \( H(z) = H_d(z) H_c(z^{M_1}) H_f(z^{M_1 M_2}) \), with \( M_1 = 4 \), \( M_2 = 2 \), \( \nu = 2 \) and \( D = M_1 M_2 \nu \). The filter \( H_d(z) \)
consists of 4 cascaded RRS filters, $H_c(z)$ is a half-band filter with order $N_c = 18$ and $H_f(z)$ is a traditional low-pass filter with order $N_f = 47$. The decimation architecture used in [39] consists on a CIC structure in the first decimation stage, followed by the half-band filter operating at a sampling rate reduced by $M_1$ and finally with the filter $H_f(z)$ operating at a sampling rate reduced by $M_1 M_2$. The overall system performs 216 APOS and 68 MPOS and it requires 30 multipliers, 75 adders and 75 memory elements. Table 3.4 summarizes the aforementioned results.

<table>
<thead>
<tr>
<th>Method</th>
<th>Number of multipliers</th>
<th>Number of adders</th>
<th>Number of memory elements</th>
<th>MPOS</th>
<th>APOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>[39]</td>
<td>30</td>
<td>75</td>
<td>75</td>
<td>68</td>
<td>216</td>
</tr>
<tr>
<td>Proposed</td>
<td>23</td>
<td>75</td>
<td>79</td>
<td>46</td>
<td>318</td>
</tr>
</tbody>
</table>

**Figure 3.7:** Magnitude response of the proposed filter $H_p(z)$ in Example 3.
3.1.3 Optimal sharpening approach for RRS-based filters

From the preceding sub-section we can see that the proposed sharpened compensated RRS filters, in comparison to the original RRS filter, achieve highly improved magnitude characteristics with an acceptable increase in the complexity of the resulting filter. Moreover, efficient CIC-like architectures can be derived for applications in sampling rate conversion. Thus, it is mainly in these applications where the sharpened RRS-based filters become very useful.

Nevertheless, in the sharpening-based schemes proposed in the previous subsection only the attenuation in stopbands can be arbitrarily improved. The passband deviation, even though controlled by the parameter \( b \) of the compensation filter (see subsection 3.1.1.1), is not guaranteed to remain below a predefined specification. In the general case of the sharpening method, there is no optimal control to achieve given passband and stopband magnitude constraints with the minimum increase of complexity. Of course a higher improvement in both, passband and stopband, can be achieved by increasing the respective tangencies \( m \) and \( n \) and, eventually, the sharpened filter will meet the specific magnitude requirements. However, neither the coefficients nor the degree of the resulting sharpening polynomial are guaranteed to be optimal for the given specification.

The previous observations constitute a strong motivation to develop an optimization-based sharpening scheme especially suitable to improve the magnitude characteristics of RRS-based filters. In the following we develop this approach, targeted to decimation applications, where the coefficients of the obtained sharpening polynomials are optimized over a discrete space to
satisfy a priori given specifications with multiplierless coefficients. Moreover, we extend the sharpening of compensated RRS filters from the previous sub-section. We introduce a low-complexity structure for decimation, where the simple multiplierless compensator can be embedded into the cascaded chain of comb filters working at lower rate. We show that, for similar magnitude characteristics, sharpening compensated RRS filters results in a lower complexity than sharpening only RRS filters without compensation, especially when stringent specifications must be met.

Let us refer to Section 2.4.2 and specifically to equations (2.19)-(2.20), where $F(\omega)$ is substituted by the zero-phase frequency response of $K$ cascaded RRS filters, $H^K(\omega)$, with $H(\omega)$ given in (3.5). The values $x_{p,l}$, $x_{p,u}$, $x_{s,l}$, and $x_{s,u}$ are

$$x_{p,l} = \min\{H^K(\omega)\}_{\omega \in [0, \omega_p]}$$

(3.43)

$$x_{p,u} = \max\{H^K(\omega)\}_{\omega \in [0, \omega_p]}$$

(3.44)

$$x_{s,l} = \min\{H^K(\omega)\}_{\omega \in [(2\pi/M)-\omega_p, (2\pi/M)+\omega_p]}$$

(3.45)

$$x_{s,u} = \max\{H^K(\omega)\}_{\omega \in [(2\pi/M)-\omega_p, (2\pi/M)+\omega_p]}$$

(3.46)

The sharpening polynomial $P(x)$ must meet the following simultaneous conditions,

$$D(x) - \delta_p \leq P(x) \leq D(x) + \delta_p, \quad \text{for} \quad x \in X_p = [x_{p,l}, x_{p,u}],$$

(3.47)

$$D(x) - \delta_s \leq P(x) \leq D(x) + \delta_s, \quad \text{for} \quad x \in X_s = [x_{s,l}, x_{s,u}].$$

(3.48)

Thus, the zero-phase frequency response of the sharpened filter achieves the desired values $D(x)$ with a maximum absolute passband deviation $\delta_p$ over the range of $\omega$ where $H^K(\omega) \in X_p$, and with a maximum absolute stopband deviation $\delta_s$ over the range of $\omega$ where $H^K(\omega) \in X_s$. 

82
We can take advantage of the Frequency Transformation approach to estimate the minimum degree of the sharpening polynomial \( P(x) \), \( N \), (which in turn is a half of the order of the prototype filter) with any of the formulas proposed in [56]-[58] that are expressible as \( N(\delta_p, \delta_s, \phi_p, \phi_s) \), where \( \phi_p \) and \( \phi_s \) are respectively given in (2.26) and (2.27). Obviously, this is a preliminary estimation that depends on the accuracy of the formula used. Substituting the values from (3.43)-(3.46) in (2.26)-(2.27), we can estimate \( N \) as,

\[
N \approx \mathcal{N}(\delta_p, \delta_s, \phi_p, \phi_s) / 2 = \frac{-20\log_{10}(\sqrt{\delta_p \delta_s}) - 13 \cdot \pi}{14.6(\phi_s - \phi_p)}.
\] (3.49)

The right side of (3.49) has been obtained using the formula given in [57].

Now we introduce the optimization framework to obtain the discrete coefficients \( p_k \) of \( P(x) \), with \( k = 0, 1, \ldots, N \), where the maximum deviation of \( P(x) \) with respect to \( D(x) \), denoted by \( \delta \), is minimized. Note that this polynomial will attain the desired passband and stopband deviations with a proper polynomial degree. To find the sharpening polynomial coefficients, we evaluate the conditions (3.47) and (3.48) over a dense grid of points \( x \) covering the ranges \( X_p \) and \( X_s \).

Let us consider the following notation in order to formalize the optimization problem:

- \( \delta_p \) and \( \delta_s \) are the desired passband and stopband deviations after sharpening.
- \( m = 10N \) is the number of points partitioning the frequency sets \( X_p \) and \( X_s \), i.e., the overall number of points in the region \( X_p \cup X_s \) is \( 20N \).
- \( \tilde{x}_{i,p} \) and \( \tilde{x}_{i,s} \) are the \( i \)-th points belonging to the sets \( X_p \) and \( X_s \), respectively. To find these points, we divide the range of frequencies
\( \tilde{\omega}_p = [0, \omega_p] \) into \( m \) equally spaced points \( \tilde{\omega}_{i,p} \) and the range of frequencies \( \tilde{\omega}_{i,s} = [(2\pi/M-\omega_p, (2\pi/M)+\omega_p] \) into \( m \) equally spaced points \( \tilde{\omega}_{i,s} \), with \( i = 1, 2, \ldots, m \). Then we set \( \tilde{x}_{i,p} = H^k(\tilde{\omega}_{i,p}) \) and \( \tilde{x}_{i,s} = H^k(\tilde{\omega}_{i,s}) \).

- \( d_{i,p} \) and \( d_{i,s} \) are the desired amplitudes of the polynomial \( P(x) \) at the points \( x = \tilde{x}_{i,p} \) and \( x = \tilde{x}_{i,s} \), respectively. Usually, \( d_{i,p} = 1 \) and \( d_{i,s} = 0 \) for all \( i \).
- \([M]_{i,j}\) denotes the entry in the \( i\)-th row and \( j\)-th column of the underlined matrix \( M \).
- \([v]_i\) denotes the \( i\)-th element of the underlined vector \( v \).
- \( B \) is an arbitrary word-length for the fractional part in a fixed-point representation of the sharpening coefficients. In other words, every sharpening coefficient \( p_k \) is an integer scaled by \( 2^{-B} \),

\[
p_k = 2^{-B} \tilde{p}_k, \quad \tilde{p}_k \in \{\text{integers}\}.
\]

By this setup, the optimization problem can be written as:

\[
\min_{\mathbf{s}} \mathbf{f}^T \mathbf{s} \quad \text{subject to} \quad A\mathbf{s} \leq \mathbf{b}, \quad [s]_i \in \{\text{integer space}\} \quad 2 \leq i \leq N + 2,
\]

where \( \mathbf{f} \) and \( \mathbf{s} \) are vectors of size \((N+2) \times 1\), \( A \) is a matrix of size \( 4m \times (N+2) \) and \( \mathbf{b} \) is a vector of size \( 4m \times 1 \). In addition, we have

\[
[s]_i = \begin{cases} 
\delta; & i = 1, \\
\tilde{p}_{i-2}; & i = 2, 3, \ldots, N + 2,
\end{cases}
\]

\[
[f]_i = \begin{cases} 
1; & i = 1, \\
0; & i = 2, 3, \ldots, N + 2,
\end{cases}
\]

\[
[A]_{i,j} = \begin{cases} 
-1; & 1 \leq i \leq 2m \text{ and } j = 1, \\
-\frac{\delta_{i,j}}{\delta_p}; & 2m < i \leq 4m \text{ and } j = 1,
\end{cases}
\]
\[ [\mathbf{A}]_{i,j} = \begin{cases} 
2^{-B}(\tilde{x}_{i,p})^{j-2}; & 1 \leq i \leq m \text{ and } 2 \leq j \leq N+2, \\
-2^{-B}(\tilde{x}_{i-m,p})^{j-2}; & m < i \leq 2m \text{ and } 2 \leq j \leq N+2, \\
2^{-B}(\tilde{x}_{i-2m,s})^{j-2}; & 2m < i \leq 3m \text{ and } 2 \leq j \leq N+2, \\
-2^{-B}(\tilde{x}_{i-3m,s})^{j-2}; & 3m < i \leq 4m \text{ and } 2 \leq j \leq N+2, 
\end{cases} \quad (3.55) \]

\[ [\mathbf{b}] = \begin{cases} 
d_{i,p}; & 1 \leq i \leq m, \\
-d_{i-m,p}; & m < i \leq 2m, \\
d_{i-2m,s}; & 2m < i \leq 3m, \\
-d_{i-3m,s}; & 3m < i \leq 4m. 
\end{cases} \quad (3.56) \]

The optimization problem in (3.51) is a constrained Mixed Integer Linear Programming (MILP) problem that is, in general, small. Thus, the simple MATLAB code available online in [59] can be used straightforwardly. Such routine is based on the \texttt{linprog} function from the MATLAB Optimization Toolbox. Once the vector \( \mathbf{s} \) has been obtained, the sharpening coefficients can be found as follows,

\[ p_k = 2^{-B}[\mathbf{s}]_{k+2} \quad 0 \leq k \leq N. \quad (3.57) \]

The transfer function of the sharpened filter is given by

\[ H_{sh}(z) = \sum_{i=0}^{N} p_i \cdot H^{iK}(z) \cdot z^{-(N-i)(M-1)K/2}, \quad (3.58) \]

with \( H(z) \) given in (3.3). The corresponding frequency response is given by

\[ H_{sh}(e^{j\omega}) = \left[ \sum_{i=0}^{N} p_i \cdot H^{iK}(\omega) \right] e^{-j\omega N[(M-1)K/2]}, \quad (3.59) \]

where \( H(\omega) \) is given in (3.5).

The resulting CIC-like structure for sharpened RRS filters with discrete sharpening coefficients is straightforwardly derived from the one proposed in [16] and it is illustrated in Figure 3.8. Note that \( K \) must be an even value to avoid fractional delays.
The computational complexity of this structure, quantified in APOS, and the number of memory elements, are respectively

\[ A = NK(M+1)+Q-1+\sum_{i=1}^{Q} S(p_i), \]  

(3.60)

\[ ME = N(N+1)K/2+2NK, \]  

(3.61)

where \( S(p_i) \) indicates the number of adders required to implement the sharpening coefficient \( p_i \) and \( Q \) is the number of non-zero sharpening coefficients.

Given the desired passband and stopband deviations \( \delta_p \) and \( \delta_s \), the design steps are the following:

1. If the degree, \( N \), of the sharpening polynomial is a priori given, go to step 2. Otherwise, obtain \( x_{p,l} \), \( x_{p,u} \), \( x_{s,l} \) and \( x_{s,u} \) using (3.43)-(3.46). Then find the values \( \phi_p \) and \( \phi_s \) with (2.26)-(2.27) and estimate the degree, \( N \), of the sharpening polynomial using (3.49).

2. Obtain \( m = 10N \) equally spaced points \( \tilde{\omega}_{i,p} \) and \( \tilde{\omega}_{i,s} \), \( i = 1, 2, \ldots, m \), over the regions \( \tilde{\omega}_p = [0, \omega_p] \) and \( \tilde{\omega}_{s,1} = [(2\pi/M) - \omega_p, (2\pi/M) + \omega_p] \) respectively.
assigning \( \tilde{\omega}_{1,p} = 0 \), \( \tilde{\omega}_{m,p} = \omega_p \), \( \tilde{\omega}_{1,s} = (2\pi/M) - \omega_p \) and \( \tilde{\omega}_{m,s} = (2\pi/M) + \omega_p \). Then set \( \tilde{x}_{i,p} = H^K(\tilde{\omega}_{i,p}) \) and \( \tilde{x}_{i,s} = H^K(\tilde{\omega}_{i,s}) \), with \( H(\omega) \) given in (3.5).

3. Choose the desired word-length \( B \). Additionally, set \( d_{i,p} = 1 \) and \( d_{i,s} = 0 \) for \( i = 1, 2, \ldots, m \).

4. Create \( f, A \) and \( b \) using (3.53)-(3.56). Then solve the problem (3.51) for \( s \). A straightforward way is using the MATLAB routine available online in [59].

5. Obtain the sharpening coefficients \( p_k \) using (3.57).

### 3.1.3.1 Optimized sharpening of compensated RRS filters

The proposal introduced in sub-section 3.1.2 has shown that sharpening a previously compensated RRS filter results in significant improvement of the passband characteristic. Additionally, the magnitude in the stopband regions can be arbitrarily improved with the number of cascaded RRS filters, \( K \). Thus, sharpening compensated RRS filters with the optimization-based scheme deserves special attention. For this case, the transfer function of the sharpened compensated RRS filter is given by

\[
H_{c,Sh}(z) = \sum_{i=0}^{N} p_i \cdot H_i^c(z) \cdot z^{-(N-i)(M-1)(K/2+M)},
\]

with \( H_c(z) \) given in (3.10). The corresponding frequency response is given by

\[
H_{c,Sh}(e^{j\omega}) = \left[ \sum_{i=0}^{N} p_i \cdot H_i^c(\omega) \right] e^{-j\omega[M(1/(M-1)(K/2+M),
\]

where \( H_c(\omega) \) is given in (3.12).

We propose the general structure presented in Figure 3.9, which is derived from the combination of both, the structure from [16] and the structures introduced in Figure 3.4.
The computational complexity and the number of memory elements of this structure are respectively given as

\[ A_c = N[K(M+1)+3]+Q-1+\sum_{i=1}^{Q} S(p_i), \quad (3.64) \]

\[ ME_c = 3N(N+1)K/4+2N(K+1). \quad (3.65) \]

The design procedure to sharpen compensated RRS filters is basically the same one introduced in the previous subsection. It is only necessary to substitute \( H_c(\omega) \) instead of \( H^K(\omega) \) in every step, with \( H_c(\omega) \) given in (3.12).

In order to explain clearly the advantages of sharpening compensated RRS filters, let us consider the following example.

**Example 4** (see Example 2 in [29]): Consider \( M = 16, \nu = 2 \) and \( \omega_p = 0.6\pi/(M\nu) \). Let us contrast the following solutions:

a) \( K = 4 \) cascaded RRS filters, sharpened by the polynomial \( P_{0,0,2,0}(x) = 3x - 3x^2 + x^3 \). Let us identify the transfer function of this filter as \( H_{sh,1}(z) \) and its frequency response as \( H_{sh,1}(e^{j\omega}) \).
b) $K = 4$ cascaded RRS filters, compensated with $C(z)$ (see 3.9) using the parameter $b = 0$ and sharpened with the polynomial $P_{0,0,0}(x) = 2x - x^2$ (solution given in [29]). Let us call this filter $H_{c,Sk,2}(z)$ and its frequency response $H_{c,Sk,2}(e^{j\omega})$.

c) $H_{Sk,3}(z)$, a filter composed by $K = 2$ cascaded RRS filters, sharpened by the polynomial $P_{0,0,1}(x) = 6x^2 - 8x^3 + 3x^4$. Its frequency response is $H_{Sk,3}(e^{j\omega})$.

d) $H_{c,Sk,4}(z)$, a filter composed by $K = 2$ cascaded RRS filters, compensated using $b = 1$ and sharpened with the polynomial $P_{0,0,1}(x) = 3x^2 - 2x^3$. Its frequency response is $H_{c,Sk,4}(e^{j\omega})$.

Figure 3.10 shows the magnitude response characteristics of these filters with emphasis on the passband and the first stop-band. Note that, in general terms, all of them are similar over the regions of interest. It is also clear that the sharpened-compensated RRS filters $H_{c,Sk,2}(z)$ and $H_{c,Sk,4}(z)$ have a wider passband, which extends beyond of the region of interest. Nonetheless, their computational complexities are quite different from each other. When these filters are implemented in CIC-like structures, using (3.60) and (3.64) we obtain the following APOS:

$A_1 = 208$, corresponding to $H_{Sk,1}(z)$,

$A_{c,2} = 143$, corresponding to $H_{c,Sk,2}(z)$,

$A_3 = 141$, corresponding to $H_{Sk,3}(z)$,

$A_{c,4} = 113$, corresponding to $H_{c,Sk,4}(z)$.

Some observations are in order.

a) Upon comparing separately the sharpened RRS filters, $H_{Sk,1}(z)$ and $H_{Sk,3}(z)$, and the sharpened-compensated RRS filters, $H_{c,Sk,2}(z)$ and
$H_{c,Sh,4}(z)$, we get a lower complexity structure if $K = 2$, even though in these cases the polynomial degree $N$ is higher. This is because both, $K$ and $N$, have the same impact on the APOS metric. However, $K$ can only take even values, whereas $N$ can also be odd. Therefore, preserving a simple sharpening polynomial and improving the stopbands by increasing $K$, as suggested in [29], does not guarantee a lower complexity result.

b) Upon comparing the sharpened RRS filter $H_{Sh,3}(z)$ and the sharpened compensated RRS filter $H_{c,Sh,4}(z)$ (both of them using $K = 2$) we observe that $H_{Sh,3}(z)$ requires a polynomial with higher degree. As a consequence, its complexity is higher, despite the use of compensators in $H_{c,Sh,4}(z)$. The reason is that the increased complexity in the sharpened-compensated RRS structures amounts to only 3 extra additions per polynomial degree, and these additions work at lower rate.

![Magnitude responses of the filters](image)

**Figure 3.10:** Magnitude responses of the filters $H_{Sh,1}(z)$, $H_{c,Sh,2}(z)$, $H_{Sh,3}(z)$ and $H_{c,Sh,4}(z)$, presented in Example 4 with $M=16$, $\nu=2$ and $\omega_p = 0.67\pi/(M\nu)$.  

Now, let us formalize the aforementioned observations. In summary, we can see that the highest impact on the APOS complexity metric depends on the product $NK(M+1)$ when we compare the computational complexity of the CIC-like structures for sharpened RRS filters and sharpened-compensated RRS filters using (3.60) and (3.64). Hence, $K$ and $N$ should be chosen as small values as possible. Since decreasing $K$ is more convenient than reducing $N$, we can set $K = 2$ in advance.

Let us discard the computational complexity introduced by the sharpening coefficients in both (3.60) and (3.64), and denote with $N_a$ the degree of the sharpening polynomial applied to 2 cascaded RRS filters, and with $N_b$ the degree of the sharpening polynomial applied to 2 cascaded and compensated RRS filters (it is being assumed $K = 2$ for the reason discussed above). We will compare the terms $NK(M+1)$ in (3.60) and $N[K(M+1)+3]$ in (3.64), assuming $N_a = N_b + c$, with $c$ being a non-negative integer. With this setup, we have

$$A \approx \tilde{A} = NK(M+1)|_{K=2, N=N_b+c} = 2N_b(M+1)+2c(M+1), \quad (3.66)$$

$$A_c \approx \tilde{A}_c = N[K(M+1)+3]|_{K=2, N=N_b} = 2N_b(M+1)+3N_b. \quad (3.67)$$

Note that $\tilde{A}$ and $\tilde{A}_c$ differ in the second term, i.e., $2c(M+1)$ versus $3N_b$. Clearly, the structure to sharpen compensated RRS filters has higher computational complexity when $c = 0$, i.e., when the degrees of the polynomials used to sharpen both, the RRS filter and the compensated RRS filter, are the same. However, if $c > 0$, sharpening compensated RRS filters becomes convenient when the following inequality is satisfied,

$$M > (3N_b/2c) - 1. \quad (3.68)$$
Upon noticing that the shape of the magnitude response of RRS filters changes very little with $M [60]$, we estimate the degree of $P(x)$ for both cases, sharpened RRS filters and sharpened compensated RRS filters, using the Kaiser formula [57] for $M = 16$; $v = 2, 4, 6$ and $8$; $\delta_p = 0.001 (\approx 0.017$ dB); and $\delta_s = 0.001 (-60$ dB), $0.0001 (-80$ dB) and $0.00001 (-100$ dB). These cases are shown in Figure 3.11. Note that, for these specifications, sharpening compensated RRS filters is convenient when the residual decimation factor $v$ is equal to 2 or 4, i.e., small values. However, generally speaking, sharpened compensated RRS filters become effective for more stringent specifications.

![Figure 3.11: Estimated degree of the sharpening polynomial to sharpen RRS filters (dashed line) and compensated RRS filters (solid line), with $\delta_p = 0.001$ and $\delta_s = 0.001$ (left), $\delta_s = 0.0001$ (middle) and $\delta_s = 0.00001$ (right).](image)

3.1.3.2 Sharpening the second-stage filter in a two-stage RRS-based structure

Consider the case where $M$ can be factorized as the product of two positive integers,

$$M = M_1 \cdot M_2.$$  (3.69)
In the most general case, $M_1$ and $M_2$ can be composite (non-prime) numbers. The transfer function of a two-stage RRS filter can be expressed in terms of the transfer functions of two separated RRS subfilters as [26]

$$H_{TS}(z) = H_1^K(z)H_2^K(z^{M_1}), \quad (3.70)$$

$$H_1(z) = \frac{1}{M_1} \cdot \frac{1-z^{-M_1}}{1-z^{-1}}, \quad (3.71)$$

$$H_2(z) = \frac{1}{M_2} \cdot \frac{1-z^{-M_2}}{1-z^{-1}}. \quad (3.72)$$

When $K_1 = K_2 = K$, the transfer function in (3.70) equals to $H^K(z)$, with $H(z)$ given in (3.3). The frequency responses of the subfilters $H_1(z)$ and $H_2(z)$ are respectively given by

$$H_1(e^{j\omega}) = H_1(\omega)e^{-j\omega(M_1-1)/2}, \quad (3.73)$$

$$H_2(e^{j\omega}) = H_2(\omega)e^{-j\omega(M_2-1)/2}, \quad (3.74)$$

where

$$H_1(\omega) = \frac{1}{M_1} \frac{\sin\left(\frac{\omega M_1}{2}\right)}{\sin\left(\frac{\omega}{2}\right)}, \quad (3.75)$$

$$H_2(\omega) = \frac{1}{M_2} \frac{\sin\left(\frac{\omega M_2}{2}\right)}{\sin\left(\frac{\omega}{2}\right)}. \quad (3.76)$$

The frequency response of the two-stage RRS filter can be expressed in terms of the frequency responses of the two subfilters as

$$H_{TS}(e^{j\omega}) = H_{TS}(\omega)e^{j\omega[(M_1-1)K_1+M_1(M_2-1)K_2]/2}, \quad (3.77)$$

$$H_{TS}(\omega) = H_1^K(\omega)H_2^K(M_1\omega). \quad (3.78)$$

When $K_1 = K_2 = K$, the frequency response in (3.77) equals to $H^K(e^{j\omega})$, with $H(e^{j\omega})$ given in (3.4).

From the analysis developed in [26], the following observations can be
derived.

- In the low-pass region, the zeros of $H_{TS}(z)$ are given by the zeros of $H_{2}^{K_2}(z^{M_2})$. Therefore, the first $M_2 - 1$ stopbands are essentially attenuated by the subfilter $H_{2}^{K_2}(z^{M_2})$. Additionally, the passband droop of $H_{TS}(z)$ is mainly determined by the passband droop of $H_{2}^{K_2}(z^{M_2})$.

- The attenuation in every $M_2$ stopbands depends on $H_{1}^{K_1}(z)$. Moreover, the filter $H_{1}^{K_1}(z)$ also contributes to the attenuation improvement in the stopbands beyond the first one because of the natural decreasing of its magnitude response.

Figure 3.12 shows these observations for the sample case $M_1 = M_2 = 5$ and $K_1 = K_2 = 4$.

![Figure 3.12: Magnitude responses of filters $H_{1}^{K_1}(z)$, $H_{2}^{K_2}(z^{M_2})$ and $H_{TS}(z)$ for the case $M_1 = M_2 = 5$, with $K_1 = K_2 = 4$.](image)

The advantages of the two-stage decomposition of the RRS filter for decimation are the following:
The down-sampling block $M$ can be divided into two separated down-sampling blocks, $M_1$ and $M_2$. By using the non-recursive form of the filter $H_1^K(z)$, this filter can work at a lower rate after the down-sampling by $M_1$ using polyphase decomposition. This results in lower power consumption in comparison to the traditional CIC structure of an RRS filter where the integrators work at high rate [32].

The filter $H_2^K(z^{M_2})$ can be moved after the down-sampling by $M_1$, resulting in low power consumption because $H_2^K(z)$ works at a lower rate. The polyphase decomposition of $H_1^K(z)$ achieves lower power consumption at expenses of an increase of area usage. However, since the second-stage filtering operates at lower rate as well, it can take advantage of the CIC architecture, when $M_2 > 2$, for area reduction. By doing so, the overall RRS-based decimation scheme achieves a trade-off between power and area. For the special case where $M_2 = 2$ we have, from (3.72), $H_2^K(z) = (1/M_2)^K(1+z^{-1})^K$. In this case, the CIC implementation is not necessary, since simple comb filters instead of integrators are placed before the downsampling by $M_2$.

Since the passband droop and the attenuation in the first folding band (the worst case attenuation) are essentially determined by the filter $H_2^K(z^{M_2})$, it is only required to apply sharpening to this filter. As a result we get better passband and stopband characteristics with lower complexity than applying sharpening to the original single stage RRS filter [26].

Owing to the aforementioned advantages, two-stage RRS-based decimation schemes have gained great popularity. This approach has been
applied to traditional RRS filters [33]-[34] and to magnitude-improved RRS filters [11]-[12], [18], [21]-[23], [25]-[26], [30]-[31]. Figure 3.13 illustrates the usual architecture of a two-stage RRS-based filter, where $P_0(z^{M_1})$ to $P_{M-1}(z^{M_1})$ represent the polyphase components that are obtained by using the non-recursive expression for the first-stage RRS filter $H_1^{K_1}(z)$. The choice of $M_1$ is a matter of compromise between two factors: having less complex polyphase components in the first stage and making the filter in the second stage working as much as possible at a lower rate. In [22]-[23] is recommended to use the factors $M_1$ and $M_2$ close in values as much as possible to each other, with $M_1 \leq M_2$.

![Figure 3.13](image)

**Figure 3.13:** Computationally efficient structure for a two-stage RRS-based decimation filter.

When the two-stage RRS-based structure is chosen, the second-stage RRS filter must be carefully designed since this is the filter where the worst-case magnitude characteristic of the overall cascade does occur. Moreover, the first-stage RRS filter introduces a passband droop that should be corrected as well (see the passband detail in Figure 3.12). It is interesting to note that, with the proposed sharpening approach, we can obtain an overall magnitude
response attaining desired passband and stopband deviations by improving only the second-stage filter. However, we must have a monotonic magnitude characteristic over the passband region of the filter to be sharpened.

For a two-stage RRS-based scheme where the sharpening is applied to the second-stage filter, the transfer function is the following,

\[
H_{TS,Sh}(z) = H_1^{K_1}(z) \sum_{i=0}^{N} p_i \cdot \tilde{H}_2^i(z^{M_1}) \cdot z^{-(N-i)M_1\tau},
\]  

with

\[
\tilde{H}_2(z) = \begin{cases} 
H_2^{K_2}(z) & \text{(without compensator)}, \\
H_2^{K_2}(z) \cdot C(z^{M_2}) & \text{(with compensator)},
\end{cases}
\]

\[
\tau = \begin{cases} 
((M_2 - 1)K_2 / 2 & \text{(without compensator)}, \\
[(M_2 - 1)K_2 / 2 + M_2] & \text{(with compensator)},
\end{cases}
\]

where \(C(z), H_1(z)\) and \(H_2(z)\) are respectively given in (3.9), (3.71) and (3.72).

The frequency response is

\[
H_{TS,Sh}(e^{j\omega}) = \left[ H_1(\omega) \sum_{i=0}^{N} p_i \cdot \tilde{H}_2^i(M_1\omega) \right] e^{-j\omega((M_1-1)K_2/2+M_2N\tau)}.
\]

\[
\tilde{H}_2(\omega) = \begin{cases} 
H_2^{K_2}(\omega) & \text{(without compensator)}, \\
H_2^{K_2}(\omega)[1 + 2^{-b} \sin^2 \left( \frac{\omega M_2}{2} \right)] & \text{(with compensator)},
\end{cases}
\]

where \(H_1(\omega)\) and \(H_2(\omega)\) are respectively given in (3.75) and (3.76), and \(b\) is the compensation parameter. From the observations preceding Example 4, we have \(K_2 = 2\). This choice leads us to use \(b = 1\) according to Table 3.1.

The efficient structure for decimation is obtained by substituting the second-stage CIC structure in Figure 3.13 (marked with curly brackets) with any of the CIC-like sharpening structures from either Figure 3.8 or Figure 3.9, depending on whether or not the compensation filter is used. In either case, \(K\) must be replaced by \(K_2 = 2\) and \(M\) by \(M_2\). Recall that if \(M_2 = 2\) the integrators
are not necessary and only comb filters can be implemented before the downsampling by $M_2$. In this case, the second-stage structures with and without compensation are given in Figure 3.14.

![Figure 3.14: Second-stage sharpened decimation structures for $M_2 = 2$ with compensation (upper figure) and without compensation (lower figure).](image)

Let us identify the term $\sum_{i=0}^{N} p_i \cdot \tilde{H}_2^i(M_1\omega)$ in (3.82) as $P(\tilde{H}_2(M_1\omega))$. To correct the passband droop of the first-stage RRS filter, whose zero-phase frequency response is $H_1^K(\omega)$, the zero-phase frequency response of the second-stage sharpened filter, $P(\tilde{H}_2(M_1\omega))$, must be designed to follow an amplitude given by $1/ H_1^K(\omega)$ over the frequency interval $\bar{\omega}_p = [0, \omega_p]$. Therefore, the desired values $d_{i,p}$ with $i = 1, 2, \ldots, m$, must be chosen as

$$d_{i,p} = 1/ H_1^K(\bar{\omega}_{i,p}),$$

(3.84)
where \( \tilde{\omega}_{i,p} \) is the \( i \)-th point of the equally spaced partition of \( \tilde{\omega}_p \) (see Section 3.1.3). Since \( H_1^k(\omega) \) is monotonically decreasing over \( \tilde{\omega}_p \), all the desired values \( d_{i,p} \) are different of each other. Thus, \( P(\tilde{x}_{i,p}) \) can approximate every value \( d_{i,p} \) with \( \tilde{x}_{i,p} = \tilde{H}_2(M \tilde{\omega}_{i,p}) \), if \( \tilde{x}_{i,p} \neq \tilde{x}_{j,p} \) for all \( i \neq j, i, j = 1, 2, \ldots, m \). To meet this condition, \( \tilde{H}_2(M \omega) \) must be monotonic over \( \tilde{\omega}_p \).

Finally, the entries \([A]_{i,1}\) and \([A]_{i+m,1}\) in (3.54) must be multiplied by \( d_{i,p} \). Similarly, the entries \([A]_{i+2m,1}\) and \([A]_{i+3m,1}\) must be multiplied by \( 1/H_1^k(\tilde{\omega}_{i,p}) \), where \( \tilde{\omega}_{i,s} \) is the \( i \)-th point of the equally spaced partition of \( \tilde{\omega}_s \) (see Section 3.1.3). This is done in order to achieve an equiripple passband deviation in the overall filter \( H_{TS,Sh}(z) \).

### 3.1.3.3 Design examples and discussion of results

The following examples are discussed to show the improvement of magnitude characteristics of RRS filters achieved with the proposed optimization-based sharpening method in comparison to other sharpening-based schemes recently introduced in literature.

**Example 5** (see the example in Section IV of [25]): Consider \( M = 16, v = 4 \) and \( \omega_p = 0.907\pi/(Mv) \). The goal is to attain at least a \(-100\) dB gain in the stopbands, with an additional passband improvement without any specific constraint. Let us consider the following solutions:

a) An RRS filter \((K = 1)\) pre-sharpened by the polynomial \( R(x) = 2x^2 - x^4 \) to obtain a passband improvement and then sharpened with the \( 5^{th} \) degree first kind Chebyshev polynomial \( P(x) = 5x - 20x^3 + 16x^5 \)
(solution using the pre-sharpening approach introduced in Section VI.A of [27]). This filter is identified by \( G_1(z) \) and its frequency response is \( G_1(e^{j\omega}) \).

b) \( K = 6 \) cascaded RRS filters compensated using \( b = -1 \) and sharpened with the polynomial \( P(x) = 2x - x^2 \) (solution using the method of [29]). Let us call this filter \( G_2(z) \) and its frequency response is \( G_2(e^{j\omega}) \).

c) \( K = 2 \) cascaded RRS filters compensated using \( b = 1 \) and sharpened with the polynomial \( P(x) = 2^{-5}(-3x^2 + 131x^3 - 96x^4) \) (solution using the proposed optimization-based scheme, with \( \delta_p = 0.0006, \delta_s = 0.000032 \)). This filter is \( G_3(z) \) and its frequency response is \( G_3(e^{j\omega}) \).

Figure 3.15 shows the magnitude response characteristics of these filters, along with detail in passband and the first stopband. Observe that the three filters accomplish the \(-100 \) dB requirement in the stopbands. In the passband, the behaviors of filters [27] and [29] are similar. The proposed filter, on the other hand, achieves a better passband droop correction, which meets the \( 0.01 \) dB ripple (\( \delta_p = 0.0006 \)) specification.

When it comes to the complexities in terms of APOS, the proposed solution achieves better results too. The filter \( G_1(z) \), implemented with a CIC-like structure, requires 20 integrators working at high rate, due to its double-sharpening scheme. Therefore its APOS metric would be higher than \( 320 = 20 \times 16 \). On the other hand, the APOS of \( G_2(z) \) is 211 (see [29] for calculating the APOS in such structure). In the proposed method, we substitute \( M = 16, K = 2, N = 4, Q = 3, S(p_2=3) = 1, S(p_3=131) = 2 \) and \( S(p_4=96) = 1 \) in (3.64) obtaining an APOS of 154. These results are summarized in Table 3.5.
**Example 6:** Consider a two-stage decimation filter with $M_1 = 8$, $M_2 = 17$, $v = 2$ and $\omega_p = 0.9\pi/(M_1 M_2 v)$. The goal is to attain at least $-60$ dB gain in the stopbands, with an additional passband improvement (without any given constraint). Assuming $K_1 = 4$ cascaded RRS filters in the first decimation stage, let us consider the following solutions for the second-stage filter:

a) $K_2 = 6$ cascaded RRS filters, compensated using $b = -1$ and sharpened with the polynomial $P(x) = 2x - x^2$ (solution using method [29]). This filter is denoted as $G_1(z)$, and its frequency response as $G_1(e^{j\omega})$.

b) $K_2 = 2$ cascaded RRS filters, compensated using $b = 1$ and sharpened with the polynomial $P(x) = 2^{-5}(-x + 5x^2 + 116x^3 - 88x^4)$ (solution using the proposed scheme, with $\delta_p = 0.0006, \delta_s = 0.001$). This filter is denoted as $G_2(z)$, and its frequency response as $G_2(e^{j\omega})$.

Figure 3.16 shows the magnitude response characteristics of these filters along with passband and first stopband details. Clearly, the filter designed with the proposed method presents both improvements: 1) better magnitude
characteristic and 2) lower complexity, as summarized in Table 3.5. For this example, the APOS in Table 3.5 corresponds to the second-stage filter (the first-stage filtering is the same in both filters and therefore it is omitted).

![Figure 3.16: Magnitude responses of the filters $G_1(z)$ (method [29]) and $G_2(z)$ (proposed), presented in Example 6 with $M = M_1 M_2 = 8 \times 17 = 136$, $\nu = 2$ and $\omega_p = 0.9 \pi/(M \nu)$.](image)

<table>
<thead>
<tr>
<th>Method</th>
<th>Example 5</th>
<th>Example 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>[27]</td>
<td>APOS &gt; 360</td>
<td>–</td>
</tr>
<tr>
<td>[29]</td>
<td>APOS = 211</td>
<td>APOS = 223</td>
</tr>
<tr>
<td>Proposed</td>
<td>APOS = 154</td>
<td>APOS = 164</td>
</tr>
</tbody>
</table>

### Table 3.5: Comparison in terms of APOS in Examples 5 and 6

3.1.4 Changing the paradigm in the magnitude improvement of RRS filters: the introduction of a corrector filter

Even though the use of extra filters to improve the passband and stopbands of RRS filters is known [2]-[9], [13]-[14], [17], [20], [22]-[23], to our knowledge there have not been simultaneous improvement of the worst-case
passband and stopband characteristics with a single, simple linear-phase filter. However, the idea of using such filter becomes promising if we consider the following:

- The worst-case passband and stopband magnitude characteristics in a RRS filter are in the rightmost edge of the passband and in the leftmost edge of the first stopband (see Figure 3.1). Thus, it is very important to improve the magnitude characteristics of the RRS filter in the passband and in the first stopband.

- In a two-stage RRS-based decimation filtering scheme where the decimation factor $M$ is expressed as the product of $M_1$ and $M_2$ and the transfer function of the two-stage RRS filter is $H_{TS}(z) = H_1^{K_i}(z) H_2^{K_i}(z^{M_1})$, with $H_1(z)$ and $H_2(z)$ respectively given in (3.71) and (3.72), the first $M_2-1$ stopbands are essentially attenuated by the subfilter $H_2^{K_i}(z^{M_1})$ (see subsection 3.1.3.2). Therefore, it is possible to design a single lowpass filter $G(z)$, such that, when it is expanded by $M_1$ and cascaded with $H_{TS}(z)$, its magnitude characteristic compensates the passband droop of the two-stage RRS filter whereas simultaneously gives better attenuation than that of $H_2^{K_i}(z^{M_1})$, thus providing a higher attenuation in the first $M_2-1$ stopbands.

- The filter expanded by $M_1$, $G(z^{M_1})$, can be moved to work at lower rate after the downsampling by $M_1$, in the same way as the filter $H_2^{K_i}(z^{M_1})$ (see subsection 3.1.3.2), resulting in lower complexity.

The previous observations were a motivation to introduce the corrector filter, defined as a linear phase FIR filter that is cascaded with the RRS filter and whose magnitude characteristic compensates the passband droop and
simultaneously improves the attenuation in the first stopband of the RRS filter. In other words, the corrector filter corrects the worst-case passband and stopband magnitude characteristics of the RRS filter. As a result, in the corrected RRS filter the passband droop is decreased and the worst-case attenuation value is increased. With regard to the classification of methods introduced in subsection 3.1.1, these improvements on the magnitude of the RRS filter belong to the class 3. Note that this approach does not fall in any of the three subclasses of methods above mentioned for the class 3.

To introduce the efficient design of a corrector filter, let us start with the two-stage RRS filter $H_{Ts}(z)$, with $H_{Ts}(z)$ given in (3.70). Upon using (3.7), the left-side and right-side edges of the first stopband of the RRS filter, respectively denoted as $\omega_x$ and $\omega_y$, can be defined as

$$\omega_x = \frac{2\pi}{M_1 \cdot M_2} - \omega_p,$$  \hspace{1cm} (3.85)

$$\omega_y = \frac{2\pi}{M_1 \cdot M_2} + \omega_p,$$  \hspace{1cm} (3.86)

where $\omega_p$ is given in (3.8). Let us consider three different corrector filters, $F_1(z)$, $F_2(z)$ and $F_3(z)$, for the case where $M_1 = M_2 = 5$ and $\omega_p = \pi/(4M_1M_2) = 0.01\pi$. The three resulting corrected filters are $H_{C1}(z) = F_1(z)H_{Ts}(z)$, $H_{C2}(z) = F_2(z)H_{Ts}(z)$ and $H_{C3}(z) = F_3(z)H_{Ts}(z)$. The corrector filters are described as follows:

- $F_1(z)$ is a low-pass FIR filter with order $N_1 = 60$. The magnitude responses of the corrector filter, the two-stage RRS filter and the corrected filter, respectively identified as $|F_1(e^{j\omega})|$, $|H_{Ts}(e^{j\omega})|$ and $|H_{C1}(e^{j\omega})|$, are illustrated in Figure 3.17.
• $F_2(z)$ is a multi-band filter that results from expanding by a factor $M_1$ a low-pass FIR filter $G_2(z)$ with order $N_2 = 12$, i.e., $F_2(z) = G_2(z^{M_1})$. The magnitude responses of this corrector filter, the two-stage RRS filter and the corrected filter, respectively identified as $|F_2(e^{j\omega})|$, $|H_{TS}(e^{j\omega})|$ and $|H_{C2}(e^{j\omega})|$, are illustrated in Figure 3.18. The filter $F_2(z)$ has its first stopband region centered in the frequency $\omega_a = \pi/M_1 = 0.2\pi$. The original lowpass magnitude response characteristic of $G_2(z)$ is compressed in the frequency interval $[0, \omega_a]$.

• $F_3(z)$ is a multi-band filter that results from expanding by a factor $M_1L$, with $L = 2$, a low-pass FIR filter $G_3(z)$ with order $N_3 = 6$, i.e., $F_3(z) = G_3(z^{M_1L})$. The magnitude responses of this corrector filter, the two-stage RRS filter and the corrected filter, respectively identified as $|F_3(e^{j\omega})|$, $|H_{TS}(e^{j\omega})|$ and $|H_{C3}(e^{j\omega})|$, are illustrated in Figure 3.19. The filter $F_3(z)$ has its first stopband region centered in the frequency $\omega_b = \pi/M_1L = 0.1\pi$ and the original lowpass magnitude response characteristic of $G_3(z)$ is compressed in the frequency interval $[0, \omega_b]$.

![Figure 3.17: Magnitude responses of the filters $F_1(z)$, $H_{TS}(z)$ and $H_{C1}(z)$.](image)
Figures 3.17 to 3.19 show that the three corrected filters, \( H_{C1}(z) \), \( H_{C2}(z) \) and \( H_{C3}(z) \), have a droop-compensated passband characteristic. These filters have nearly the same worst-case passband and stopband magnitude values, as shown in Figure 3.20.

The first corrected RRS filter, \( H_{C1}(z) \), has a higher attenuation than the original two-stage RRS filter in all of its stopbands. However, its
corresponding corrector filter, \( F_1(z) \), requires a high order, namely, \( N_1 = 60 \). Thus, it is not convenient to use this corrector filter because it adds a considerable complexity to the original two-stage RRS filter.

![Magnitude responses of the filters \( H_{C1}(z), H_{C2}(z) \) and \( H_{C3}(z) \) in the passband and the first stopband (where the worst-case attenuation occurs).](image)

**Figure 3.20:** Magnitude responses of the filters \( H_{C}(z) \), \( H_{C2}(z) \) and \( H_{C3}(z) \) in the passband and the first stopband (where the worst-case attenuation occurs).

On the other hand, the attenuation over the 5th and 10th stopbands of the corrected filter \( H_{C2}(z) \) has not been improved in comparison to the original two-stage RRS filter, as it can be seen in Fig. 3.18. The reason is that the corresponding corrector filter, \( F_2(z) \), is a multiband filter that does not provide additional attenuation in the 5th and 10th stopbands due to a passband characteristic over these frequency regions. Nevertheless, the magnitude response of the two-stage RRS filter has sufficient attenuation in these frequencies and, most importantly, the requirement of correcting the worst-case passband and stopband characteristics has been accomplished. It is worth highlighting that, since we are based in a two-stage decimation scheme, the corrector filter \( F_2(z)=G_2(z^{M_1}) \) can be moved after the
downsampling by $M_1$, resulting in a lowpass filter $G_2(z)$ that works at lower rate and whose order is $N_2 = 12$.

Finally, the third corrected filter, $H_C(z)$, has a corrected passband and a better attenuation in its first stopband in comparison to the two-stage RRS filter. The magnitude response of the two-stage RRS filter has sufficient attenuation beyond of its first stopband region that serves to attenuate the passband characteristics of the corrector filter $F_3(z)$ in these frequency regions. Thus, the corrector filter has improved only the magnitude characteristic in the passband and the first stopband, correcting the original worst-case passband and stopband values. The corresponding corrector filter, $F_3(z) = G_3(z^{M_1})$, can be moved after the downsampling by $M_1$, resulting in another multiband filter, $G_3(z^L)$, which is the version expanded by $L$ of a lowpass filter $G_3(z)$ whose order is $N_3 = 6$.

Note that, comparing the three corrected filters, the third one has the lowest additional computational complexity because it uses a corrector filter that is based on a low-order lowpass filter, which works at a lower rate after the downsampling by $M_1$. Therefore, the design of the corrector filter $F(z)$ starts with a simple $N$-th order, linear phase low-pass FIR filter whose transfer function $G(z)$ is

$$G(z) = \begin{cases} 
  g(N/2)z^{-N/2} + \sum_{n=0}^{(N/2)-1} g(n)(z^{-n} + z^{-(N-n)}) & \text{if } N \text{ is even,} \\
  \sum_{n=0}^{(N-1)/2} g(n)(z^{-n} + z^{-(N-n)}) & \text{if } N \text{ is odd},
\end{cases} \quad (3.87)$$

with $g(n)$ being the coefficients of the symmetric impulse response, and whose frequency response is
This filter will be referred as basic low-pass filter. If such FIR filter is expanded by a factor $LM_i$, with $L \geq 1$, the resulting expanded-by-$LM_i$ filter (i.e., the corrector filter) can be simplified by maximizing its transition band while it provides simultaneously an improvement in both, the passband and the first stopband of the RRS filter, where the worst-case passband and stopband characteristics occur. Thus, the design of the corrector filter consists of two consecutive steps:

1. Calculation of the optimal value $L$.
2. Design of the basic low-pass filter $G(z)$.

The transfer function of the corrector filter is
\[
F(z) = G(z^{LM_i})
\]  
and its frequency response is
\[
F(e^{j\omega}) = F(\omega) \cdot e^{j\omega LM_i N/2},
\]  
with
\[
F(\omega) = G(LM_i \omega).
\]

The transfer function of the proposed corrected RRS filter is given by
\[
H_p(z) = H_{rs}(z) \cdot F(z),
\]
where $H_{rs}(z)$ and $F(z)$ are respectively given in (3.70) and (3.90). The frequency response is
\[
H_p(e^{j\omega}) = H_p(\omega) \cdot e^{j\omega (LM_i-1)K_1 + M_1(M_2-1)K_2 + LM_i N/2},
\]  
with
\[
G(e^{j\omega}) = G(\omega) e^{-j\omega N/2},
\]  
\[
G(\omega) = \begin{cases} 
  g(N/2) + \sum_{n=0}^{N/2-1} 2g(n) \cos(\omega(N/2-n)), & \text{if } N \text{ is even}; \\
  \sum_{n=0}^{(N-1)/2} 2g(n) \cos(\omega(N/2-n)), & \text{if } N \text{ is odd}.
\end{cases}
\]
\[
H_p(\omega) = H_{TS}(\omega) \cdot F(\omega),
\]  
(3.95)

where \(H_{TS}(\omega)\) and \(F(\omega)\) are respectively given in (3.78) and (3.92).

Let us discuss both parts of the design of a corrector filter in the following subsections.

### 3.1.4.1 Calculation of the optimal value \(L\)

As we mentioned earlier, a low-pass filter expanded by a factor \((LM_1)\) becomes a multi-band filter with the original lowpass magnitude response compressed in the frequency interval \([0, \pi/(LM_1)]\) and with the first stopband region centered in the frequency value \(\pi/(LM_1)\). Therefore, the value \(L\) can be chosen in such a way that the first stopband of the corrector filter completely includes the first stopband of the RRS filter whereas its first transition band gets maximized to decrease as much as possible the amount of non-zero filter coefficients. To highlight the aforementioned idea, Figure 3.21 shows a pictorial representation where the original compressed low-pass magnitude response of the basic low-pass filter is depicted along with its first image. Note that the left-side edge of the first stopband of the corrector filter \(G(z^{LM_1})\) is given by the frequency value \(\omega_x\) given in (3.85). The right-side edge of this stopband, \(\omega_z\), is given as

\[
\omega_z = 2\pi/(M_1 \cdot L) - \omega_x.
\]  
(3.96)

Upon replacing \(\omega_x\) from (3.85) in (3.96), we obtain

\[
\omega_z = \frac{2\pi}{M_1 \cdot M_2} \left( \frac{1}{L} - \frac{1}{M_2} \right) + \omega_p.
\]  
(3.97)

These frequency points are depicted in Figure 3.21 as well.
For the overall first stopband of the RRS filter to be included within the first stopband of the corrector filter, the frequency point $\omega_y$ must be less than or equal to $\omega_z$, i.e.,

$$\omega_y \leq \omega_z. \quad (3.98)$$

Substituting (3.86) and (3.97) in (3.98) we get

$$L \leq M_2 / 2. \quad (3.99)$$

Now let us recall the following relations between the basic low-pass filter and the corrector filter.

- Since the corrector filter is the version expanded by $M_1L$ of the basic low-pass filter, in the magnitude characteristic of the corrector filter the low-pass characteristic of the basic low-pass filter is compressed in the frequency interval $[0, \pi/M_1L]$.

- The first passband characteristic of the corrector filter has to compensate for the passband droop over the frequency interval $\bar{\omega}_p = [0, \omega_y]$ and its first stopband characteristic has to increase the attenuation
over the first stopband of the RRS filter, i.e., the interval \( \bar{\omega}_{k,1} = [\omega_x, \omega_y] \) (see (3.7)), with \( \omega_p, \omega_x \) and \( \omega_y \) respectively given in (3.8), (3.85) and (3.86).

From the two previous observations we have that the transition band of the basic low-pass filter is given by

\[
\Delta = M_1 \cdot L \cdot (\omega_x - \omega_p),
\]

(3.100)

where \( (\omega_x - \omega_p) \) is the transition band of the compressed version of the original magnitude characteristic of the basic lowpass filter. After substituting the expression of \( \omega_x \) from (3.85) in (3.100), we have

\[
\Delta = L \left( \frac{2\pi}{M_2} - 2M_1\omega_p \right).
\]

(3.101)

With this setup, the value \( L \) must be the integer maximizing (3.101) under the constraint (3.99). Since \( M_1, M_2 \) and \( \omega_p \) are constants depending on the problem at hand, the maximization of \( \Delta \) implies that \( L \) must be an as large integer as possible subject to the upper bound (3.99).

Nevertheless, an additional consideration should be taken into account if \( M_2 \) is a composite number, such that it can be expressed with the form \( M_2 = (M_2/k) \times k \) where \( k \) is its smallest prime factor. This consideration is important for the realization of an efficient structure, as will be detailed in sub-section 3.1.4.4. In this case the downsampler by \( M_2 \) can be split into two downsamplers, one of them by \( M_2/k \) and the other one by \( k \). For now, it is sufficient to say that, in such case, it is convenient that \( L \) includes the value \( (M_2/k) \) as at least one of its factors. In this way, it will be possible to move the basic low-pass filter after the downsampling by \( (M_2/k) \). From the upper bound (3.99) and considering the aforementioned requirement for \( L \), i.e., \( L = \)
(M2/k)·x, we can express (M2/k)·x ≤ M2/2. Clearly, the greatest value for x that satisfies the previous condition is x = [k/2]. Therefore the value L can be obtained as

\[
L = \begin{cases} 
[M_2 / 2]; & \text{if } M_2 \text{ is prime,} \\
(M_2 / k) \lfloor k / 2 \rfloor; & \text{if } M_2 \text{ is composite, with } k \text{ being its smallest prime factor.}
\end{cases}
\] (3.102)

### 3.1.4.2 Design of the basic low-pass filter \( G(z) \)

Let us define the passband and stopband of the basic low-pass filter, \( \Omega_p \) and \( \Omega_s \) respectively, as follows

\[
\Omega_p = \left[0, \omega_q\right], \tag{3.103}
\]

\[
\Omega_s = \left[\omega_{s,1}, \omega_{s,2}\right]. \tag{3.104}
\]

From the two observations preceding equation (3.100), we have that the passband edge frequencies of the basic low-pass filter are obtained as the band-edge frequencies of the interval \( \tilde{\omega}_p \) multiplied by \( M_iL \). Similarly, the stopband edge frequencies of the basic low-pass filter are obtained as the band-edge frequencies of the interval \( \tilde{\omega}_{s,1} \) multiplied by \( M_iL \). It is clear that, if \( \omega_y \) (the right-most edge of the interval \( \tilde{\omega}_{s,1} \)) is greater than \( \pi/M_iL \), the multiplication \( (M_iL) \times \omega_y \) will result in a value higher than \( \pi \). In such case, the right-most edge of the interval \( \Omega_s \) (i.e., the frequency value \( \omega_{s,2} \)) must be equal to \( \pi \). Therefore, the band-edge frequencies \( \omega_y, \omega_{s,1} \) and \( \omega_{s,2} \) in (3.103) and (3.104) are respectively given by

\[
\omega_q = M_iL\omega_p, \tag{3.105}
\]

\[
\omega_{s,1} = M_iL\omega_s, \tag{3.106}
\]
\[ \omega_{1,2} = \begin{cases} M_1 L \omega_y, & \text{if } M_1 L \omega_y \leq \pi, \\ \pi, & \text{if } M_1 L \omega_y > \pi. \end{cases} \]  

(3.107)

Let us denote by \( G_{des}(\omega) \) the desired amplitude values that the amplitude response of the basic low-pass filter must approximate. It is clear that, over the stopband region \( \Omega_s \), these desired amplitude values are equal to zero. Now, let us consider the desired amplitude values over the passband region \( \Omega_p \). In order to compensate for the passband droop of the RRS filter, the desired values that the amplitude characteristic of the corrector filter must approximate in the frequency interval \( \tilde{\omega}_p = [0, \omega_p] \) are given by the inverse of the amplitude values of the two-stage RRS filter in such frequency interval. By denoting with \( F_{des}(\omega) \) these desired values, and using (3.78), we have

\[ F_{des}(\omega) = \frac{1}{H_1^{K_1}(\omega) H_2^{K_2}(M_1 \omega)}; \quad \omega \in [0, \omega_p]. \]  

(3.108)

Using the relation between the amplitude responses of the corrector filter and the basic low-pass filter given in (3.92), we have that \( F(\omega/M_1 L) = G(\omega) \). Therefore, we can find the desired passband amplitude values for the basic low-pass filter by applying this relation to \( F_{des}(\omega) \) and \( G_{des}(\omega) \), i.e., \( F_{des}(\omega/M_1 L) = G_{des}(\omega) \). Therefore, using (3.108) we have

\[ G_{des}(\omega) = \begin{cases} \frac{1}{H_1^{K_1}(\omega/M_1 L) H_2^{K_2}(\omega)}; & \omega \in \Omega_p, \\ 0; & \omega \in \Omega_s. \end{cases} \]  

(3.109)

From the setup above, the design of the basic low-pass filter can be accomplished by the following minimization problem

\[
\min_{\{g(n)\} \in \Omega_1 \cup \Omega_2} \{ \max_{\omega \in \Omega_1 \cup \Omega_2} |E(\omega)| \}, \quad g(n) \in \text{discrete space},
\]

(3.110)

\[ E(\omega) = W(\omega)[G(\omega) - G_{des}(\omega)], \]  

(3.111)
where $W(\omega)$ is a weighting function accounting for a different approximation in the various frequency bands [61]. In order to have $E(\omega) = 0$ at least in two frequency points (in the passband and stopband regions), the minimum value of $N$ should be 4. This stems from the alternation theorem that characterizes the optimal solution in the minimax sense [62]. Of course, higher values of $N$ can be used to improve the response at the cost of higher computational complexity.

### 3.1.4.3 Design steps of the corrector filter $F(z)$

Given the values $M = M_1M_2$ and $\omega_p$, the steps to design the corrector filter are the following:

1. Obtain $L$ using (3.102).
2. Design the basic low-pass filter $G(z)$ as follows:
   2.1. Obtain the frequency edges of the first folding band, $\omega_x$ and $\omega_y$, using (3.85) and (3.86).
   2.2. Obtain the passband and stopband regions of the filter, $\Omega_p$ and $\Omega_s$, using (3.103) to (3.107).
   2.3. Define the desired values over the ranges $\Omega_p$ and $\Omega_s$ using (3.109), as well as a given filter order $N > 3$.
   2.4. Obtain the coefficients of the basic low-pass filter by solving the problem (3.110). The filter coefficients must be discrete to avoid multipliers.
3. Design the corrector filter $F(z)$ in terms of the basic low-pass filter using (3.90).
3.1.4.4 Proposed structure

Let us address the structure of a corrected RRS filter for decimation applications. To this end, we start with the two-stage RRS-based structure, i.e., assuming that the decimation factor \( M \) is expressed as the product of \( M_1 \) and \( M_2 \), with the transfer function of the filter formed by replacing (3.70) and (3.92) in (3.93). Figure 3.22 shows the initial decimation structure. By moving the filters \( H_2^K(z^{M_1}) \) and \( G(z^{M_2}) \) after the downsampling by \( M_1 \) and using the polyphase decomposition for \( H_1^K(z) \) and the CIC structure for the second-stage filtering, we arrive at the structure presented in Figure 3.23.

**Figure 3.22**: Initial decimation structure where a two-stage RRS filter is cascaded with the proposed corrector filter.

**Figure 3.23**: Computationally efficient structure stemming from multirate identities.
Note that the structure in Figure 3.23 is based on the two-stage RRS-based scheme presented in Figure 3.13, with the additional correction filtering working at a lower rate after the downsampling by $M_1$.

Now, let us focus our attention on the second stage of the structure of Figure 3.23 (marked with curly brackets), namely, the correction filtering and the second-stage RRS filter implemented using the CIC structure. We can have two cases for the value $M_2$, namely, either $M_2$ is prime or $M_2$ is composite. Let us consider each one of these cases.

a) $M_2$ is prime. Note that, since the basic low-pass filter must be expanded by $L$, which is a factor directly proportional to $M_2$ as we see in (3.102), it requires $N\lfloor M_2/2 \rfloor$ memory elements. However, if this filter is implemented in polyphase form, with $M_2$ polyphase components denoted as $G_0(z^{M_2})$ to $G_{M_2-1}(z^{M_2})$, the number of required delay elements can be reduced to be less or equal to $M_2-1$ if $N < M_2$. Moreover, in this case the arithmetic operations of the correction filtering are performed at a lower rate. Therefore, it is convenient to implement the filter in polyphase decomposition. The resulting structure is presented in Figure 3.24.

b) $M_2$ is composite. As we mentioned earlier in the paragraph preceding equation (3.102), when $M_2$ is a composite number the downsampler by $M_2$ can be split into two downsamplers, one of them by $M_2/k$ and the other one by $k$, where $k$ is its smallest prime factor. This is shown in Figure 3.25(a). Note that the value for $L$, obtained from (3.102), contains a factor equal to $M_2/k$. Thus, the basic low-pass filter expanded by $L$ can be moved after the downsampler by $M_2/k$ resulting
in the basic low-pass filter expanded by \([k/2]\), namely, \(G(z^{[k/2]})\). By doing so, and moving the comb part of \(H_2(z)\) after the downsampler by \(k\), we obtain the structure shown in Fig. 3.25(b). Finally, after a polyphase decomposition of \(G(z^{[k/2]})\), we get the structure presented in Figure 3.25(c), where \(G_0(z^k)\) to \(G_{k-1}(z^k)\) represent the polyphase components of \(G(z^{[k/2]})\). Clearly, the additional arithmetic operations introduced by the correction filtering are performed at a lower rate and, since \(k\) is a relatively small number, the number of extra delay elements is generally low. Thus, it is in this two-stage scheme, with a recursive second-stage filter and a composite factor \(M_2\), where the use of the corrector filter becomes more convenient.

Finally, it is worth noticing that in the previous subsections we have considered the case of a two-stage RRS filter, i.e., \(H_{TS}(z) = H_1^{K_1}(z) H_2^{K_2}(z^{M_2})\), with the factor \(M\) being expressed as the product \(M = M_1M_2\). However, the case of a filter \(H^K(z)\) (\(K\) cascaded RRS filters) preserved as a single-stage filter

![Figure 3.24](image)

**Figure 3.24:** Second-stage CIC-like structure with polyphase decomposition of the basic low-pass filter when \(M_2 > 2\) is a prime number (upper figure) and when \(M_2 = 2\) (lower figure).
Figure 3.25: Second-stage structure with embedded corrector filtering, where $M_2$ is a composite number; (a) Initial structure; (b) CIC-like structure obtained after applying noble identities to both, the expanded basic low-pass filter and the comb part of $H_2(z)$; (c) Resulting CIC-like second-stage structure, with the additional computational complexity of the corrector filtering working at lower rate. Note that the structure of Fig. 3.25(c) can be used as a CIC-like overall structure, and not only in the second-stage filtering.

can be considered simply by assuming that $M = M_2$, i.e., $M_1 = 1$. In this case, the design of the corrector filter is derived in the same way as was done in previous subsections, only considering $M_1 = 1$ where is needed. Since $M_1 = 1$, the corresponding structure is found by removing the filtering $H_i(z)$ and its corresponding downsampling by $M_1$ in Figures 3.22 and 3.23, which results only in the second-stage structure. Therefore, the resulting single-stage CIC-based structure is the one presented in Figure 3.24 for $M$ prime or the one presented in Figure 3.25(c) for $M$ being a composite number, where $M = M_2$. 
We can summarize the characteristics of the corrector filter as follows:

a) The corrector filter is a symmetric FIR filter. Therefore, it preserves stability and linear-phase, and it does not suffer of the drawbacks inherent to zero-rotation schemes.

b) The corrector filter requires in general a low number of non-zero coefficients and, therefore, introduces a low complexity. The reason is that, since this filter is focused on the improvement of only the passband and the first folding band (where the worst-case attenuation occurs), it is based on a low-order basic low-pass filter that can be expanded to use only few non-zero coefficients.

c) The corrector filter can be easily designed as a multiplierless filter. Thus, it does not add too much complexity to the traditional RRS filter.

d) The corrector filter can be implemented in polyphase form to work at the lower rate. Therefore, the entire additional computational workload to the classical CIC-like structure is modest.

3.1.4.5 Design examples and discussion of results

The following examples show the improvement of magnitude characteristics of RRS filters achieved with the corrector filter approach, along with the complexity measure in terms of the APOS metric.

Example 7: Consider a two-stage decimation filter with $M_1 = 7$, $M_2 = 2$, $\nu = 4$ and $\omega_p = \pi/(M_1M_2\nu)$. Assume $K_1 = K_2 = 5$. Let us design a corrector filter for this case, using a filter order $N = 4$.

From (3.102) we get $L = 1$ and using $M_1$, $M_2$ and $\nu$ in (3.85) and (3.86) we
obtain the frequency edges of the first folding band, \( \omega_x = 0.125 \pi \) and \( \omega_y = 0.16072 \pi \). The passband and stopband edge frequencies of the basic low-pass filter, \( \omega_p, \omega_{s,1} \) and \( \omega_{s,2} \), are obtained using (3.105) to (3.107). We have \( \omega_p = 0.125 \pi, \omega_{s,1} = 0.875 \pi, \) and \( \omega_{s,2} = \pi \). Therefore, the passband region of the corrector filter is \( \Omega_p = [0, 0.125 \pi] \) while the stopband region is \( \Omega_s = [0.875 \pi, \pi] \).

For the optimization in (3.110) we ran a Mixed Integer Linear Programming (MILP) algorithm over a space of discrete coefficients having 6 digits for the fractional part. The obtained finite precision basic low-pass filter is \( G(z) = 2^{-6} \times [-19(1 + z^{-4}) + 17(z^{-1} + z^{-3}) + 68z^{-2}] \). From (3.90), we obtain the corrector filter as \( F(z) = G(z^7) \).

Figure 3.26 presents the magnitude response of the basic low-pass filter, \( |G(e^{j\omega})| \). Figure 3.27 shows the magnitude responses (in dB) of the corrector filter, \( |F(e^{j\omega})| = |G(e^{7j\omega})| \), along with the magnitude responses of the first- and second-stage RRS filters, \( |H_1^{K_1}(e^{j\omega})| \) and \( |H_2^{K_2}(e^{j\omega})| \). We observe that the odd folding bands are essentially attenuated by both, the corrector filter and the second-stage RRS filter. On the other hand, the even folding bands are attenuated by the zeros of the first-stage RRS filter. Fig. 3.28 presents the magnitude response (dB) of the overall corrected RRS filter, as well as the one of the \( K = 5 \) cascaded RRS filters.

It is worth highlighting that, since the shape of the magnitude characteristic of RRS filters remains almost unchanged regardless of the factor \( M \) [59], we can use the corrector filter presented in this example for all decimation factors \( M = M_1M_2 \) when \( M_2 = 2, K = 5 \) and \( \nu = 4 \). The following example illustrates this case.
Figure 3.26: Magnitude response of the basic low-pass filter $G(z)$ along with the desired passband and stopband characteristics.

Figure 3.27: Magnitude responses of filters $H_1^K(z)$, $H_2^K(z^{M_1})$, and $F(z) = G(z^{L_M})$, with $K_1 = K_2 = 5$, $M_1 = 7$ and $L = 1$, presented in Example 7.

**Example 8** (see the example in Section IV of [27]): Consider $M = 16$, $\nu = 4$ and $\omega_p = 0.907\pi/(M\nu)$. The goal is to attain at least a $-100$ dB gain in stopbands, with an additional passband improvement without any specific constraint. Let us consider the following solutions:
a) An RRS filter \((K = 1)\) sharpened with the 5th degree first kind Chebyshev polynomial \(P(x) = 5x - 20x^3 + 16x^5\) (solution of [27], without passband improvement). We identify this filter as \(H_a(z)\), and its magnitude response as \(|H_a(e^{i\omega})|\).

b) \(K = 5\) cascaded RRS filters corrected using the filter \(F(z) = G(z^8)\), with 
\[G(z) = 2^{5}[−19(1 + z^{−4}) + 17(z^{−1} + z^{−3}) + 68z^{−2}]\]. We identify this corrected filter as \(H_b(z)\), and its magnitude response as \(|H_b(e^{i\omega})|\).

Figure 3.29 shows the magnitude response characteristics of these filters, along with detail in passband and the first folding band. Both filters accomplish the \(-100\) dB requirement in the folding bands, but the proposed filter has a much better passband characteristic, which in fact achieves a 0.04 dB ripple.

Now let us refer to the APOS metric, considering CIC-like structures for both filters. The structure presented in [27] for the filter \(H_a(z)\) requires 5M APOS due to the 5 integrators working at high rate, plus 5 APOS due to the comb stages and 4 extra APOS due to the sharpening coefficients and the
addition of these branches to the path of cascaded combs. This results in an overall APOS metric equal to 89. In the proposed method we use the CIC-like structure of Fig. 3.25(c) with \( M_2 = 8 \) and \( k = 2 \). The high-rate filtering requires \( 5M \) APOS due to the 5 integrators. In the low-rate section, we have two polyphase branches for the corrector filter, namely, \( G_0(z) = -19+68z^{-1}-19z^{-2} \) and \( G_1(z) = 17+17z^{-1} \), preceding 5 cascaded comb filters. The polyphase branch \( G_0(z) \) can be implemented with 4 additions, whereas the polyphase branch \( G_1(z) \) can be implemented with 2 additions. Thus, the additional computational burden of the corrector filtering is 7 (one addition is required to add the two polyphase branches). The overall APOS metric is 92. Therefore, the proposed scheme achieves a significant magnitude improvement in the passband by performing only 3 extra additions per sample when compared with method [27]. These results are summarized in Table 3.6.

![Figure 3.29](image_url)

**Figure 3.29:** Magnitude responses of the filters \( H_a(z) \) (method [27]) and \( H_b(z) \) (proposed method) presented in Example 8.
Table 3.6: Comparison in terms of APOS in Example 8

<table>
<thead>
<tr>
<th>Method</th>
<th>Computational complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>[27] (Whithout passband improvement)</td>
<td>APOS = 89</td>
</tr>
<tr>
<td>Proposed</td>
<td>APOS = 92</td>
</tr>
</tbody>
</table>

**Example 9:** Consider a two-stage RRS-based decimator with \( M = M_1 M_2 = 70 \), where \( M_1 = 7 \) and \( M_2 = 10 \). The goal is to attain at least a \(-50 \) dB gain in the stopbands when the residual decimation factor is \( v = 2 \) and at least a \(-100 \) dB gain in the stopbands when the residual decimation factor is \( v = 4 \), with an additional passband improvement without any specific constraint. The passband edge frequency is \( \omega_p = \pi/(Mv) \). Let us consider the following solutions:

a) For \( v = 2 \), a two-stage RRS filter with \( K_1 = 4 \) and \( K_2 = 6 \), cascaded with a compensation filter \( C_1(z) = \{-2^{-4}[1 + z^{-2}] - 18z^{-1}\}^5 \) (solution proposed in [23]). We identify this filter as \( H_{a,1}(z) \), and its magnitude response as \( |H_{a,1}(e^{j\omega})| \). For \( v = 4 \), a two-stage RRS filter with \( K_1 = 4 \) and \( K_2 = 6 \), cascaded with a compensation filter \( C_2(z) = \{-2^{-2}[1 + z^{-2}] - 35z^{-1}\}^4 \) (solution proposed in [22]). We identify this filter as \( H_{a,2}(z) \), and its magnitude response as \( |H_{a,2}(e^{j\omega})| \).

b) For \( v = 2 \), a two-stage RRS filter with \( K_1 = 4 \) and \( K_2 = 4 \), cascaded with a corrector filter \( F_1(z) = G_1(z^{35}) \), with \( G_1(z) = 2^{-4}[24(1 + z^{-4}) + 25(z^{-1} + z^{-3}) + 60z^{-2}] \). We identify this filter as \( H_{b,1}(z) \), and its magnitude response as \( |H_{b,1}(e^{j\omega})| \). For \( v = 4 \), a two-stage RRS filter with \( K_1 = 4 \) and \( K_2 = 5 \), cascaded with a corrector filter \( F_2(z) = G_2(z^{35}) \), with \( G_2(z) = 2^{-6}[16(1 + z^{-4}) + 25(z^{-1} + z^{-3})] \).
\[ + 17(z^{-1} + z^{-3}) + 62z^{-2} \]. We identify this filter as \( H_{b,2}(z) \), and its magnitude response as \( |H_{b,2}(e^{j\omega})| \).

Figures 3.30 and 3.31 show the magnitude response characteristics of the corresponding filters for the cases when \( \nu = 2 \) and \( \nu = 4 \), respectively. For both cases, the proposed filters accomplish the attenuation requirement in the folding bands and they have a better passband characteristic in comparison with methods [22] (for \( \nu = 4 \)) and [23] (for \( \nu = 2 \)). With regard to the APOS metric, the proposed filters are less complex as shown in Table 3.7. Since the first-stage filtering is the same in all the filters (\( K_1 \) cascaded RRS filters, with \( K_1 = 4 \), decomposed in 7 polyphase branches working after the decimation by \( M_1 = 7 \)), only the APOS corresponding to the second-stage filtering is presented in Table 3.7.

![Figure 3.30](image)

**Figure 3.30:** Magnitude responses of the filters \( H_{a,1}(z) \) (method [23]) and \( H_{b,1}(z) \) (proposed method) presented in Example 9.
Figure 3.31: Magnitude responses of the filters $H_{a,2}(z)$ (method [22]) and $H_{b,2}(z)$ (proposed method) presented in Example 9.

<table>
<thead>
<tr>
<th>Method</th>
<th>$\nu = 4$ APOS</th>
<th>$\nu = 2$ APOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>[22]</td>
<td>APOS = 63</td>
<td>–</td>
</tr>
<tr>
<td>[23]</td>
<td>–</td>
<td>APOS = 75</td>
</tr>
<tr>
<td>Proposed</td>
<td>APOS = 56</td>
<td>APOS = 49</td>
</tr>
</tbody>
</table>

3.1.5 Separated magnitude improvement of RRS filters using compensation filtering and Chebyshev sharpening

It has been shown in subsections 3.1.3.3 and 3.1.4.5 that the proposed methods, namely, the optimization-based sharpening approach and the correction filtering approach, outperform the Chebyshev sharpening recently introduced in [27] when it comes to improvement of the worst-case magnitude characteristics of RRS filters and complexity in terms of APOS.
However, an important advantage of the method [27] is the ability to decouple the stopband improvement. Consider the following observations:

- Method [27] introduces equiripple stopbands in RRS filters by sharpening them with Chebyshev polynomials. If the passband improvement is discarded, the method improves the worst-case magnitude characteristic in the stopband with a low additional complexity, as shown in Example 8 (subsection 3.1.4.5).

- The passband improvement of the RRS filters sharpened with Chebyshev polynomials can be performed with compensation filtering. The compensator filter can work at the low rate section, thus introducing low computational complexity.

In method [27] the author follows a simultaneous passband-stopband improvement either using a double sharpening or employing a single sharpening with modified Chebyshev polynomials. Nevertheless, the aforementioned observations are the two main reasons that motivate the proposal of separating the improvements, taking advantage of method [27] only to enhance the magnitude characteristic in the stopband. The separated improvement approach has been also used in methods [18]-[23], as we detailed in subsection 3.1.1. The following subsection describes the Chebyshev RRS (CRRS) filter, directly derived from method [27].

### 3.1.5.1. Design and characteristics of Chebyshev RRS filters

An $N$-th order Chebyshev RRS (CRRS) filter is a RRS filter sharpened by a Chebyshev polynomial of first kind whose degree is $N$ [27]. Its transfer function is given by
\[ H_{C,N}(z) = \sum_{i=0}^{N} z^{-(N-i)M/2} \cdot c_i \cdot [M'y^{-1}H(z)]', \quad (3.112) \]

with \( H(z) \) given in (3.3) and where \( c_i \) is the \( i \)-th coefficient of the polynomial, i.e., we are using the notation \( c_0 + c_1x + \ldots + c_{N-1}x^{N-1} + c_Nx^N \) for a Chebyshev polynomial of degree \( N \). Its frequency response is given as
\[
H_{C,N}(e^{j\omega}) = H_{C,N}(\omega)e^{-j\omega NM/2},
\]

\[
H_{C,N}(\omega) = \sum_{i=0}^{N} c_i \cdot [M \cdot H(\omega)]',
\]

where \( H_{C,N}(\omega) \) is its zero-phase frequency response and with \( H(\omega) \) given in (3.5).

The factor \( \gamma \),
\[
\gamma \leq \gamma_{\text{max}} = \frac{1}{|M \cdot H(\omega_s)'|},
\]

where \( \omega_s \) given in (3.85), must be expressible in Signed Powers-of-Two (SPT).

The following are important characteristics of an \( N \)-th order CRRS filter in comparison to \( N \) cascaded RRS filters [27]:

1) The filters \( H_{C,N}(z) \) and \( H^N(z) \) have similar passband droops, but the worst-case attenuation of \( H_{C,N}(z) \) is improved (i.e., it is greater) by approximately \( 6(N-1) \) dB in comparison to the corresponding worst-case attenuation of \( H^N(z) \).

2) Unlike \( H^N(z) \), where the worst-case attenuation appears in the left-hand side of the first stopband, in \( H_{C,N}(z) \) this attenuation value appears in every stopband (this occurs also in method [16]).

3) The CIC-like structure introduced in [27] can be used for decimation applications (as proposed in [16]). The structure is multiplierless, it
does not suffer of pole-zero cancelation issues and it has modest extra computational complexity compared with the CIC structure.

The next subsection provides the proposed simple guidelines to decide when to use an $N$-th order CRRS filter instead of $N$ cascaded RRS filters.

### 3.1.5.2. When to use Chebyshev RRS filters

In this section we address the problem to decide whether or not using an $N$-th order CRRS filter instead of $N+1$ cascaded RRS filters. It must be clear that it can be possible to replace $N+1$ cascaded RRS filters by an $N$-th order CRRS filter if all the stopbands of the RRS filter are allowed to have the same worst-case attenuation. In such case we can proceed in trying a possible substitution. Now, let us continue with Figures 3.32 and 3.33 that show the worst-case attenuation values (in dB) for $N$-th order CRRS filters and $K$ cascaded RRS filters against $\nu$, the residual decimation factor, when $M = 16$ and $\omega_p = \pi/(\nu M)$. These figures can be used as a reference regardless of $M$, since the shape of the magnitude response of these RRS-based filters change very little with $M$ [60].

We observe that, for a given $\nu$, the worst-case attenuation of an $N$-th order CRRS filter is equal or greater (greater implies better) than the worst-case attenuation of $N+1$ cascaded RRS filters if the value $N$ is equal or higher than a lower bound $N_{\text{min}}$. It can be seen also that $N_{\text{min}}$ decreases as $\nu$ decreases. For example, if $\nu = 9$, we have that the worst-case attenuation of both, $H_{C,N}(z)$ and $H_{N+1}^{N+1}(z)$, is 150 dB when $N = N_{\text{min}} = 5$ (Fig. 3.33). On the other hand, if $\nu = 2$, the worst-case attenuation of both filters is about 40 dB when $N = N_{\text{min}} = 3$ (Fig. 3.32). Clearly, it is convenient to use an $N$-th order CRRS filter instead of
(N+1) cascaded RRS filters if the worst-case attenuation is high enough to demand $N \geq N_{\text{min}}$ because in such case the CRRS filter requires fewer cascaded RRS filters.

![Figure 3.32: Worst-case attenuation values of $H_{C>(z)}$ and $H^k(z)$ vs $v$ for $1 \leq N \leq 4$ and $1 \leq K \leq 5$.](image)

![Figure 3.33: Worst-case attenuation values of $H_{C>(z)}$ and $H^k(z)$ vs $v$; $5 \leq N \leq 10$ and $6 \leq K \leq 10$.](image)

We formalized the aforementioned discussion with the following formulation. Given a prescribed worst-case attenuation value in dB, denoted
as $A_{\text{min}}$, we propose to obtain the number of cascaded RRS filters necessary to achieve a worst-case attenuation equal to $A_{\text{min}}$ as

$$K_c = \left\lfloor -\frac{|A_{\text{min}}|}{20\log_{10}|H(\omega_s)|} \right\rfloor.$$  \hfill (3.116)

Since an $N$-th order CRRS filter may replace $K_c$ cascaded RRS filters to achieve the same worst-case attenuation, we express the order of the CRRS filter as $N=(K_c-l)$. For the highest possible integer $l$, which we call $l_{\text{max}}$, $N$ becomes equal to $N_{\text{min}}$. We propose to calculate $l_{\text{max}}$ as

$$l_{\text{max}} = \left\lfloor \frac{6K_c}{6-20\log_{10}|H(\omega_s)|} \right\rfloor.$$  \hfill (3.117)

If $l_{\text{max}} \geq 1$, it is convenient to use the $N_{\text{min}}$-th order CRRS instead of $K_c$ cascaded RRS filters.

Now, let us refer to the two-stage RRS-based decimation filter $H_{\text{TS}}(z)$ given (3.70). The two filters that compose $H_{\text{TS}}(z)$ are $H_1^{K_1}(z)$ and $H_2^{K_2}(z^{M_1})$, with $H_1(z)$ and $H_2(z)$ respectively given in (3.71) and (3.72). It is natural to consider if it can be possible to replace the $K_1$ and $K_2$ cascaded RRS filters respectively by $N_1$- and $N_2$-th order CRRS filters. For this case, consider the following observations.

1) In the first-stage decimation block, i.e., a filter $H_1^{K_1}(z)$ (the transfer function is referred to the input sampling rate) followed by a downsampler by $M_1$, the residual decimation factor is $\nu_1 = \nu M_2$. Replacing $M$ by $M_1$ and $\nu$ by $\nu_1$ in (3.116) and (3.117), we have that $l_{\text{max}}$ is, for most typical values of $M_1$, $\nu_1$ and $A_{\text{min}}$, always less than 1.

2) In the second-stage decimation block, i.e., a filter $H_2^{K_2}(z)$ (the transfer function is referred to the downsampled-by-$M_1$ sampling rate)
followed by a downsampler by $M_2$, the residual decimation factor is $v_2 = v$. Substituting $M$ by $M_2$ and $v$ by $v_2$ in (3.116) and (3.117), we have that $l_{\text{max}}$ is, for most typical values of $M_2$, $v_2$ and $A_{\text{min}}$, always equal or greater than 1.

From the previous observations, for the two-stage RRS-based case we propose to use an $N_{\text{min}}$-th order CRRS filter as second-stage filter, whereas the first-stage filter is preserved as a $K_1$ cascaded traditional RRS filters. This leads to a two-stage RRS-based filter whose transfer function is

$$H_{T_{S,C}}(z) = H_{1}^{K_i}(z) \cdot H_{C,N_{\text{min}},2}(z),$$  \hspace{1cm} (3.118)

with $H_1(z)$ given in (3.71) and where $H_{C,N_{\text{min}},2}(z)$ is obtained from (3.112), upon replacing $N$ by $N_{\text{min}}$, $M$ by $M_2$ and $H(z)$ by $H_2(z)$, with $H_2(z)$ given in (3.72). The frequency response is

$$H_{T_{S,C}}(e^{j\omega}) = H_{T_{S,C}}(\omega) \cdot e^{j\omega[H_{1}(M_2^{-1})+M_2N_{\text{min}}]}/2,$$  \hspace{1cm} (3.119)

$$H_{T_{S,C}}(\omega) = H_{1}^{K_i}(\omega) \cdot H_{C,N_{\text{min}},2}(M_1\omega),$$  \hspace{1cm} (3.120)

with $H_1(\omega)$ given in (3.75) and where $H_{C,N_{\text{min}},2}(\omega)$ is obtained from (3.114), upon replacing $N$ by $N_{\text{min}}$, $M$ by $M_2$ and $H(\omega)$ by $H_2(\omega)$, with $H_2(\omega)$ given in (3.76).

The next subsection introduces the design of the low-complexity compensators for RRS-based filters.

### 3.1.5.3. Compensation filter design

The passband droop of an $N_{\text{min}}$-th order CRRS filter is similar to that of $N_{\text{min}}$ cascaded RRS filters. Additionally, in the two-stage RRS-based scheme the passband droop is mainly dominated by the passband droop of the second-
stage filter [22], [23]. Since the proposed second-stage filter, $H_{C,N_{\min}^2}(z)$, is an $N_{\min}$-th order CRRS filter, at first glance we can use any compensator designed for $N_{\min}$ cascaded RRS filters. However, for a more proper compensation we can use an $L$-th order Type-I (i.e., symmetrical) FIR filter with transfer function and frequency response respectively given in (3.121) and (3.122),

$$F(z) = f(L/2)z^{-L/2} + \sum_{n=0}^{(L/2)-1} f(n)(z^{-n} + z^{-(L-n)}), \quad (3.121)$$

$$F(e^{j\omega}) = F(\omega)e^{-j\omega L/2}, \quad (3.122)$$

where

$$F(\omega) = f(L/2) + \sum_{n=0}^{L/2-1} 2f(n)\cos(\omega \times (L/2-n)) \quad (3.123)$$

is its zero-phase frequency response and with $f(n)$ being the coefficients of the symmetric impulse response. The design of the compensator consists on finding the optimal coefficients $f(n)$ that solve the minimization problem

$$\min \left\{ \max \{|E(\omega)|\}, \ f(n) \in \{\text{discrete space}\} \right\}, \quad (3.124)$$

$$E(\omega) = 1 - S \cdot F(\omega)H_0(\omega M^{-1}) \quad (3.125)$$

where $S$ is the corresponding scaling to make $H_0(\omega) = 1$ when $\omega = 0$ and with $H_0(\omega)$ given as

$$H_Q(\omega) = \begin{cases} H_{TS}(\omega) & \text{two-stage case, } l_{\text{max}} < 1, \\ H_{TS,C}(\omega) & \text{two-stage case, } l_{\text{max}} \geq 1, \end{cases} \quad (3.126)$$

with $H_{TS}(\omega)$ and $H_{TS,C}(\omega)$ respectively given in (3.78) and (3.120).

The design of multiplierless compensation filters under the minimax criterion has been presented in [9]. The authors in that reference use Interval
Analysis to solve the optimization problem (3.124). Here we employ the Mixed Integer Linear Programming (MILP) using the MATLAB routine available in [59].

Now let us introduce a different approach to design compensation filters. Recall that the simplest compensator available in literature for RRS filters is the one proposed in [5], whose transfer function is given in (3.9). In [6] is proposed to cascade this filter several times (with the parameter $b$ in (3.9) set equal to 2) to obtain an improved compensation for wide-band cases, i.e., when the passband of the RRS filter is $\omega_p = \pi/2M$ (see (3.8)). The simplicity of the compensator [5] and the improvement on the magnitude response obtained in [6] by cascading simple compensators are the two main motivations to propose an amplitude-transformation-based compensation scheme, which formally introduces the optimal design of compensators using cascaded simple subfilters. This scheme is developed in the following paragraphs.

Let us denote the transfer function and frequency response of the $L$-th order linear phase FIR subfilter respectively as $\tilde{B}(z)$ and $\tilde{B}(e^{j\omega}) = \tilde{B}(\omega) e^{j\omega L/2}$. The resulting transfer function is $\tilde{G}(z)$ and the resulting frequency response is $\tilde{G}(e^{j\omega}) = \tilde{G}(\omega) e^{-j\omega P L/2}$, where

$$
\tilde{G}(z) = \sum_{i=0}^{P} \sum_{j=0}^{L} z^{-((P-i)L/2)} p_i \tilde{B}_j(z),
$$

(3.127)

$$
\tilde{G}(\omega) = \sum_{i=0}^{P} p_i \tilde{B}_i(\omega).
$$

(3.128)

The value $P$ is the degree of the transformation polynomial $P(x)$,

$$
P(x) = \sum_{i=0}^{P} p_i x^i.
$$

(3.129)
The amplitude response of the transformed filter, $\tilde{G}(\omega)$, should be ideally equal to $1/H_Q(\omega M^{-1})$ for $0 \leq \omega \leq M\omega_p$ in order to compensate for the passband droop of the two-stage RRS-based filter proposed in the previous section. Moreover, $\tilde{G}(\omega)$ must be monotonically increasing because $H_Q(\omega M^{-1})$ is monotonically decreasing in that frequency range. Hence, $\tilde{B}(\omega)$ must be monotonic. For a cosine-squared filter, we have

$$\tilde{B}(z) = 2^{-2} \cdot [1 + 2z^{-1} + z^{-2}], \quad (3.130)$$

$$\tilde{B}(\omega) = \cos^2(\omega / 2). \quad (3.131)$$

The cosine-squared filter is a good candidate as subfilter because of its simplicity and the fact that $\tilde{B}(\omega)$ is a monotonic function in $0 \leq \omega \leq M\omega_p$. Thus, the proposed compensator is $\tilde{G}(z^M)$, with $\tilde{G}(z)$ given in (3.127). The general form to design the compensation filter is finding the optimal coefficients $p_i$ that solve the minimization problem

$$\min \left\{ \max_{|p_i| \leq M\omega_p} |\varepsilon(\omega)| \right\}, \quad (3.132)$$

where

$$\varepsilon(\omega) = 1 - S \cdot \tilde{G}(\omega)H_Q(\omega / M) = 1 - H_Q(\omega / M)\sum_{i=0}^p p_i \tilde{B}^i(\omega). \quad (3.133)$$

Recall that the compensation filters should be easily adaptable to distinct passband compensation requirements, such that the remaining filters used for the residual decimation factor (in the subsequent stages) do not need to be programmable for droop compensation. Thus, instead of directly solving the problem (3.132), let us look for a practical viewpoint of our transformation-based approach by exploring the first-degree polynomial, i.e., $P=1$ in (3.129). Fig. 3.34 shows the amplitude transformation from $\tilde{B}(\omega)$ to $\tilde{G}(\omega)$ through the line $P(x) = p_0 + p_1 x$. This line (upper-left plot in Fig. 3.34) has an arbitrary value $y_0$.
for $x=1$, thus $p_0 = y_0 - p_1$. By substituting (3.130) in (3.127) using $p_0 = y_0 - p_1$ and developing the sum we get

$$
\tilde{G}(z) = 2^{-2} [p_1 + 2(y_0 - p_1)z^{-1} + p_1 z^{-2}].
$$

(3.134)

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3.34.png}
\caption{Linear transformation of a cosine-squared filter. To see how the compensation characteristic arises, proceed counter-clockwise starting on the upper right. Follow the dashed arrows as a reference.}
\end{figure}

Consider the following characteristics of this second-order filter:

1. To transform the droop of the cosine-squared filter ($\tilde{B}(\omega)$ in the upper-right plot of Fig. 3.34) into a compensating characteristic ($\tilde{G}(\omega)$ in the lower-right plot of Fig. 3.34) the slope $p_1$ of the line $P(x)$ with respect to $x$ must be negative, i.e., $p_1 = -|p_1|$ (see $P(x)$ in the upper-left plot of Fig. 3.34, being aware that the values $x$ are presented in the vertical axis.)
The higher the droop of the RRS-based filter, the greater $|p_1|$. We can implement the filter from (3.134) with the structure of Fig. 3.35.

2. The value $y_0$ is the amplitude response of the compensator at $\omega = 0$. For a $L_\infty$-minimized error, $y_0$ must be less than 1 [9]. However, by setting $y_0=1$ the filter becomes simpler and the maximum peak error deviation is just slightly increased. In this way, the compensation characteristic only depends on $|p_1|$.

![Figure 3.35: Second-order transformation-based compensator. Usually, $y_0 = 1$.](image)

Invariably, any compensator must change its magnitude characteristic in accordance with the droop of the filter to be compensated and in terms of the band of interest to provide a proper compensation. For $K$ cascaded RRS filters used in the first stage of a decimation chain with $v$ as residual decimation factor, the magnitude characteristic of the compensator must change as a function of $K$ and $v$. In the proposed scheme, this change is controlled by the coefficient $|p_1|$. Figure 3.36 shows the plot of optimal values that the coefficient $|p_1|$ takes with respect to $K$, for $v = 2, 3, 4$ and $5$, along with the function $p(K, v)$ that approximates these values. This function is given by

$$p(K, v) = q_2 K^2 + q_1 K + q_0,$$

(3.135)

$$q_2 = 0.0736 v^{-2.578},$$

(3.136)
To compensate for the droop of a $K$-th order CRRS filter, we use the same approximation in (3.135), just changing the coefficient $q_2$ to take the value $q_2 = 0.088\nu^{-2.578}$ in (3.136).

Note that the function $p(K, \nu)$ closely follows the optimal values and thus it is adequate to easily design the compensation filter for a given $K$ and $\nu$. To obtain a simple compensator, $|p_1|$ is made multiplierless by simple rounding as follows,

$$|p_1| = 2^{-r} \text{round}([p(K, \nu)]/2^{-r}),$$

where $r$ is an arbitrary word-length for the fractional part of the coefficient, in this proposal chosen as $2 \leq r \leq 6$.

Observe that, for a given $\nu$, the coefficient $|p_1|$ grows in proportion to $K$. Thus, one can set in advance the number of additions that the multiplierless
coefficient $|p_1|$ can use and then find the closest approximation to $p(K, \nu)$ for the common values $K$ and $\nu$ in terms of simple shift parameters that control the increase of $|p_1|$. For example, the simplest expression of the coefficient $|p_1|$ is a power of two, since no additions are required. By setting $|p_1| = 2^{-b}$, $b$ is the shift parameter that decreases as $K$ increases for a given $\nu$. Substituting $p_1 = 2^{-b}$ in (3.134) with $y_0 = 1$ it can be shown that the result is equivalent to the transfer function given in (3.9). Thus, the parameter $b$ can be selected according to Table 3.1. A more exact approximation with low complexity consists on allowing $|p_1|$ to use one addition/subtraction operation, i.e., setting $|p_1| = 2^a[2^a + (-1)^\nu]$ and varying the shift parameters $a_1$ and $a_2$ and the sign parameter $a_3$ to choose either an addition or subtraction. For the case of $K$ cascaded RRS filters, Table 3.8 shows the values for $a_1$, $a_2$ and $a_3$ for $K$ ranging from 2 to 7, with $\nu = 2$ and $\nu = 4$. Note that the proposed compensator can have a configurable coefficient that changes in terms of $K$ and $\nu$ by using variable shifts as proposed in [63] for reconfigurable filters.

<table>
<thead>
<tr>
<th>Table 3.8: Typical values of the amplitude transformation-based compensation filter parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter $K$</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
</tbody>
</table>
In the following two examples we compare the proposed simple compensator with methods [5], [6] and [9].

**Example 10**: Consider $M = 25$, $v = 2$ (wideband case) and $K = 5$ cascaded RRS filters, which ensures an attenuation of at least 45 dB in the stopbands. Let us contrast the following filters:

a) $H_a(z) = H^K(z) \tilde{G}(z^M)$, i.e., 5 cascaded RRS filters ($H(z)$ is given in (3.3)) compensated using the proposed compensator (see (3.134)), with $|p_1| = 1 + 2^{-2}$. The frequency response is $H_a(e^{j\omega})$.

b) $H_b(z) = H^K(z) \tilde{C}(z^M)$, i.e., 5 cascaded RRS filters compensated using the compensator from [6], given as $\tilde{C}(z) = [-2^{-4} \times (1-2^4+2)z^{-1} + z^{-2}]^4$. The frequency response is $H_b(e^{j\omega})$.

c) $H_c(z) = H^K(z)D(z^M)$, i.e., 4 cascaded RRS filters compensated using the compensator from [9], given as $D(z) = [(1+2^{-4}-2^{-6})(1+z^{-2}) + (1+2^{-1}-2^{-3} + 2^{-7})z^{-1}]$. The frequency response is $H_c(e^{j\omega})$.

Fig. 3.37 shows the magnitude responses $|H^K(e^{j\omega})|$, $|H_a(e^{j\omega})|$, $|H_b(e^{j\omega})|$ and $|H_c(e^{j\omega})|$ over the band of interest ($H(e^{j\omega M})$ is given in (3.4)). We can observe that the magnitude response of the filter [6] is slightly better. However, this filter requires the implementation of 4 basic three-addition subfilters, which results in 12 additions; three times the number of additions required in the proposed filter. On the other hand, the deviation of the proposed filter in comparison to that in [9] is slightly higher, but that filter requires 3 more additions and approximately twice the word-length with respect to the proposed filter. Besides, despite of the number of additions, the proposed
method has the advantage of an easy design procedure with near-optimal solution, whereas method [9] requires a specialized optimization.

**Figure 3.37:** Passband detail of magnitude responses of $K = 5$ cascaded RRS filters without compensation ($|H_K(e^{j\omega})|$), compensated with the proposed filter ($|H_a(e^{j\omega})|$), compensated with method [6] ($|H_b(e^{j\omega})|$) and compensated with method [9] ($|H_c(e^{j\omega})|$) in Example 10.

**Example 11:** Consider $M = 16$, $\nu = 4$ (narrowband case) and $K = 4$ cascaded RRS filters, which ensures an attenuation of at least 65 dB in the stopbands. Let us contrast the following filters:

**d)** $H_d(z) = H^K(z)\tilde{G}(z^M)$, i.e., 4 cascaded RRS filters compensated using the proposed compensator (see (4.134)), with $|p_1|=1-2^{-2}$. The frequency response is $H_d(e^{j\omega})$.

**e)** $H_e(z) = H^K(z)C(z^M)$, i.e., 4 cascaded RRS filters compensated using the compensator from [5], given as $C(z) = -2^{-b+2} \times [1- (2^{(b+2)} + 2)z^{-1} + z^{-2}]$, with $b = 1$. The frequency response is $H_e(e^{j\omega})$. 
f) $H_c(z) = H^K(z)D(z^M)$, i.e., 4 cascaded RRS filters compensated using the compensator from [9], given as $D(z) = [(-2^{-3} - 2^{-4} + 2^{-13})(1+z^{-2}) + (1+2^{-1} - 2^{-3} - 2^{-9})z^{-1}]$. The frequency response is $H_c(e^{j\omega})$.

Fig. 3.38 shows the magnitude responses $|H^K(e^{j\omega})|$, $|H_a(e^{j\omega})|$, $|H_b(e^{j\omega})|$ and $|H_c(e^{j\omega})|$ over the band of interest. The filter from [5] is the simplest compensator, and along with the proposed filter, it also can be designed with a straightforward method. However, its magnitude characteristic is not near optimal. By using only one extra addition, the proposed compensator achieves a much better magnitude characteristic.

![Figure 3.38: Passband detail of magnitude responses of $K = 4$ cascaded RRS filters without compensation ($|H^K(e^{j\omega})|$), compensated with the proposed filter ($|H_a(e^{j\omega})|$), compensated with method [5] ($|H_b(e^{j\omega})|$) and compensated with method [9] ($|H_c(e^{j\omega})|$) in Example 11.](image)

From the previous examples we can see that the simple second-order compensation filters that result from a linear amplitude transformation have good compensation characteristics, a low complexity and they can be used for narrowband as well as wideband compensation.
3.1.5.4. Proposed structure

The transfer function of the proposed two-stage RRS-based decimation filter is

\[ H_p(z) = H_{TS,C}(z) \cdot G(z^{M_1 M_2}), \quad (3.140) \]

where \( H_{TS,C}(z) \) is given in (3.118) and \( G(z) \) denotes the transfer function of the compensation filter (\( G(z) \) is referred to the downsampling-by-\( M_1 M_2 \) sampling rate). Figure 3.39 shows the resulting structure after expressing the first-stage filter, \( H_{1K}(z) \) (see (3.118)), in non-recursive form and decomposing it in polyphase components, which are denoted as \( P_i(z) \), with \( i = 1, 2, ..., M_1 \). The second-stage filter, \( H_{C,N_{\text{min}},2}(z) \) (see (3.118)), is implemented using the CIC-like structure proposed in [16]. However, the slight modification with regard to the distribution of coefficients based on the Horner-algorithm form, as proposed in [27], can also be used. In Fig. 3.39, we have \( A(z) = z^{-1}/(1-z^{-1}) \) and \( B(z) = 1-z^{-1} \). The coefficients \( d_i, i = 0, 1, ..., D \), with \( D = (N_{\text{min}}-r)/2 \), are obtained as \( d_i = c_{2i+r} \) with \( r = 1 \) if \( N_{\text{min}} \) is odd or \( r = 0 \) if \( N_{\text{min}} \) is even. The dotted blocks are bypassed if \( N_{\text{min}} \) is even.

Figure 3.39: Proposed structure. The dotted blocks are included if \( N_{\text{min}} \) is odd.
3.1.5.5 Design steps

Given the values $M = M_1M_2$, $\omega_p$ and a desired minimum attenuation value $A_{\text{min}}$ to be accomplished in all the folding bands, the steps to design the proposed two-stage RRS-based filter are the following:

1. Estimate $K_1$ using (3.116), replacing $M$ by $M_1$ and $\nu$ by $\nu_1 = \nu M_2$.
2. Estimate $N_{\text{min}}$ as $K_2-l_{\text{max}}$. To this end, $K_2$ is calculated with (3.116), upon replacing $M$ by $M_2$. The value $l_{\text{max}}$ is calculated using (3.117), upon replacing $M$ by $M_2$ and $K_e$ by $K_2$.
3. Obtain $\gamma$ as $\gamma = 2^6 \left\lfloor \gamma_{\text{max}} / 2^6 \right\rfloor$, where $\gamma_{\text{max}}$ is obtained from (3.115), upon replacing $M$ by $M_2$.
4. Design the compensation filter $G(z)$ with any of the methods described in subsection 3.1.5.3.

It is worth highlighting that, since the proposed scheme includes compensation, the estimated values $K_1$ and $N_{\text{min}}$ may not be enough to achieve the attenuation $A_{\text{min}}$ and, in that case, they must be increased accordingly. For the case where $M$ is a prime number, the resulting filter is a single-stage case with recursive implementation. The design procedure follows the same aforementioned steps, just substituting $M_1 = 1$ whenever is required. In this case, the resulting design consists in either $K$ cascaded RRS filters or an $N_{\text{min}}$-th order CRRS filter (depending on the resulting value $l_{\text{max}}$) with compensation filtering.

3.1.5.6 Design examples and discussion of results

In the following we use a couple of examples to compare the proposed two-stage RRS-based method with references [22] (when $\nu < 2$) and [23]
when \( \nu = 2 \), which present the best trade-off in terms of computational complexity (quantified in Additions Per Output Sample, APOS) and magnitude-response improvement. In both examples we assume, in accordance with [22] and [23], that the same attenuation is desired in all the stopbands.

**Example 12:** Consider a decimation factor \( M = 21 \), with \( M_1 = 3, M_2 = 7 \), a residual decimation factor \( \nu = 2 \) and a desired minimum attenuation \( A_{\text{min}} = 96 \text{dB} \). Let us compare the following solutions:

a) \( H_a(z) = H_{1K_1}H_{C,N_{\text{min}},2}(z^{M_2})G_1(z^{M_1}) \), with \( K_1 = 4, N_{\text{min}} = 7 \) and \( \gamma = 2^{-6} \times 29 \) \((\gamma^2 \approx 2^{-12} \times 841)\). The values \( K_1, N_{\text{min}} \) and \( \gamma \) are calculated as explained in subsection 3.1.5.5. The compensator is a 4th-order filter designed by solving the optimization problem (3.124) and it has the transfer function \( G_1(z) = 2^{-5} \times 5 \times (1 - z^{-4}) - 2^{-5} \times 28 \times (z^{-1} - z^{-3}) + 2^{-3} \times 78 \times z^{-2} \). The coefficients from the Chebyshev polynomial are \( d_0 = -7, d_1 = 56, d_2 = -112, d_3 = 64 \).

b) \( H_b(z) = H_{1K_1}H_{C,N_{\text{min}},2}(z^{M_2})G_2(z^{M_1}) \) is a filter similar to \( H_a(z) \), described in the previous paragraph, but with a different compensator. The compensation filter has the transfer function given in (3.134), with \( p_1 = -|p_1| \) and \( |p_1| \) obtained from (3.139) using \( r = 2 \), which results in \( G_2(z) = 2^{-2}[-2 + 8z^{-1} - 2z^{-2}] \).

c) \( H_c(z) = H_{1K_1}H_{2K_2}(z^{M_1})G_3(z^{M_2}) \), with \( K_1 = 4, K_2 = 11 \) and compensation filter having the transfer function \( G_3(z) = [-2^{-4}[1 - (2^4 + 2)z^{-1} + z^{-2}]]^{10} \). This is the solution given in [23].
Figure 3.40 shows the magnitude responses $|H_a(e^{j\omega})|$, $|H_b(e^{j\omega})|$ (both corresponding to the proposed two-stage RRS-based scheme) and $|H_c(e^{j\omega})|$ (corresponding to method [23]). Observe that all the filters satisfy the required attenuation, but the proposed filters have a better magnitude response in the passband. Among the two filters based on the proposed method, namely, $H_d(z)$ and $H_b(z)$, the filter $H_d(z)$ has the best passband characteristic because it uses a 4th-order optimized compensator instead of a simple 2nd-order sub-optimal compensator. With regard to the APOS metric, the proposed filters are less complex as shown in Table 3.9. Since the first-stage filtering is the same in all the filters ($K_1$ cascaded RRS filters, with $K_1 = 4$, decomposed in 3 polyphase branches working after the decimation by $M_1 = 3$), only the APOS corresponding to the second-stage filtering is presented in Table 3.9.

![Figure 3.40: Magnitude responses of the filters $H_a(z)$ (proposed method using a 4th-order optimized compensator), $H_b(z)$ (proposed method using a 2nd-order amplitude-transformation-based compensator) and $H_c(z)$ (method [23]), presented in Example 12.](image-url)
Example 13: Consider $M = 85$ and $\nu = 4$, with $M_1 = 5$, $M_2 = 17$, and a desired minimum attenuation $A_{\text{min}} = 120$ dB. Let us compare the following solutions:

a) $H_a(z) = H_1 K_1(z) H_{C,N_{\text{min}},2}(z^M) G_1(z^M)$, with $K_1 = 3$, $N_{\text{min}} = 6$ and $\gamma = 2^{-5} \times 13$ ($\gamma^2 = 2^{-10} \times 169$). The transfer function of the $4^{\text{th}}$-order optimized compensation filter is $G_1(z) = 2^{-4} \times (1 - z^{-1}) - 2^{-4} \times 8 \times (z^{-1} - z^{-3}) + 2^{-4} \times 30 \times z^{-2}$. Additionally we have $d_0 = -1$, $d_1 = 18$, $d_2 = -48$, $d_3 = 32$.

b) $H_b(z) = H_1 K_1(z) H_{C,N_{\text{min}},2}(z^M) G_2(z^M)$ is a filter similar to $H_a(z)$, described in the previous paragraph, but with a different compensator. The $2^{\text{nd}}$-order amplitude-transformation-based compensation filter has the transfer function $G_2(z) = 2^{-5} [-9 + 50z^{-1} - 9z^{-2}]$.

c) $H_c(z) = H_1 K_1(z) H_2 K_1(z^M) G_3(z^M)$, with $K_1 = 3$, $K_2 = 8$ and compensation filter having the transfer function $G_3(z) = -2^{-4} [1 - 2^2z^{-1} + z^{-2}]$. This is the solution given in [22].

Figure 3.41 shows the magnitude responses $|H_a(e^{j\omega})|$, $|H_b(e^{j\omega})|$ and $|H_c(e^{j\omega})|$ (the last one corresponds to method [22]). The proposed filters have a better magnitude response in the passband and they are also less complex, as shown in Table 3.9. In the same way as Example 12, only the APOS corresponding to the second-stage filtering is presented in Table 3.9.

<table>
<thead>
<tr>
<th>Method</th>
<th>Example 12</th>
<th>Example 13</th>
</tr>
</thead>
<tbody>
<tr>
<td>[23]</td>
<td>APOS = 118</td>
<td>–</td>
</tr>
<tr>
<td>[22]</td>
<td>–</td>
<td>APOS = 146</td>
</tr>
<tr>
<td>Proposed (4th-order compensator)</td>
<td>APOS = 84</td>
<td>APOS = 127</td>
</tr>
<tr>
<td>Proposed (2nd-order compensator)</td>
<td>APOS = 78</td>
<td>APOS = 126</td>
</tr>
</tbody>
</table>

Table 3.9: Comparison in terms of APOS in Examples 12 and 13
Figure 3.41: Magnitude responses of the filters $H_a(z)$ (proposed method using a 4th-order optimized compensator), $H_b(z)$ (proposed method using a 2nd-order amplitude-transformation-based compensator) and $H_c(z)$ (method [22]), presented in Example 13.

### 3.2 Contributions on wideband filters

Hilbert transformers are a special class of wideband filters whose characteristic is to introduce a $\pi/2$ radians phase shift of the input signal. They are used in a wide number of Digital Signal Processing (DSP) applications besides of digital communication systems, such as radar systems, medical imaging and mechanical vibration analysis, among others [64]-[68]. Subsection 3.2.1 is dedicated to the proposal of design techniques specialized for Hilbert transformers, taking as a basis the Frequency Transformation (FT) method [69]. An important insight proposed in this subsection is avoiding the arbitrary selection of the prototype filter as in [69]. We introduce the optimized search of the adequate prototype filter, such that the complexity of the overall filter, estimated from the weighted contributions of multiplier coefficients, adders and delay elements, is minimized. The
extension to a general multi-level FT technique is introduced under the proposed optimization framework. We then present a strategic combination of the FT method and the Frequency-Response Masking (FRM) technique [70]-[71], which takes advantage of the characteristics of both techniques to obtain an overall low-complexity design. Additionally, efficient structures based on the Pipelining Interleaving (PI) technique [72] (which employs time-shared subfilters) are introduced for FT-based HTs to avoid the repeated use of identical subfilters.

3.2.1 Efficient design of FIR Hilbert transformers using Frequency Transformation (FT)

In this section we introduce design techniques specialized for Hilbert transformers, taking as a basis the Frequency Transformation (FT) method. The FT method for Hilbert transformers, presented in subsection 2.4.3, was introduced in [69] and allows designing a Hilbert transformer by using several simple identical subfilters $G(z)$. Consider the following important observations on the FT method:

- The complexity of the subfilter $G(z)$ depends in a great extent on its transition bandwidth, since its ripple specification can be considerably relaxed. This characteristic makes the subfilter suitable to be implemented as a simple multiplierless system with rounded coefficients.
- The prototype filter is a low-complexity filter because its transition bandwidth is relaxed, even though its ripple specification might be strict.
The repeated use of identical subfilters can be avoided by taking advantage of the Pipelining-Interleaving (PI) technique [72], presented in Section 2.4.4. Thus, a time-multiplexed design with lower area can be obtained.

The aforementioned observations constitute the motivation to develop new design schemes based on the FT method. To this end, in the following subsection we present how to obtain PI-based structures to efficiently implement the FT-based Hilbert transformers. Then we introduce in subsection 3.2.1.2 an optimization-based method to design Hilbert transformers using FT.

### 3.2.1.1 Filtering structure for FT-based Hilbert transformers using the Pipelining-Interleaving (PI) method

Let us start with Figure 3.42, which presents the FT-based single-rate structure directly derived from (2.44). Note that the structure consists of \( L_p/2 - 1 \) cascaded blocks \( F(z) \) (grey blocks in Figure 3.42), preceded by a cascaded subfilter \( G(z) \), where \( L_p \) is the length of the prototype filter and \( F(z) \) is given in (2.44). The chain of cascaded identical subfilters \( G(z) \) is enclosed within a dashed box. Additionally, Figure 3.42 shows the input and output nodes of the subfilters as \( n_{1,a}, n_{1,b}, n_{2}, n_{3,a}, n_{3,b}, \) and so on. Note that the output of the first subfilter is not directly connected to the second subfilter. Instead, from that output three branches are derived: the first one contains the coefficient \( a(0) \), the second one has a delay block and the last one is connected to the input of the second subfilter through a coefficient with value 2. The same occurs with the output of every \( k \)-th subfilter, with \( k = 2l + 1 \) and \( l \) being a non-negative
integer. Every odd pair of nodes, namely, \((n_{a}, n_{b})\), \((n_{3,a}, n_{3,b})\) and in general \((n_{k,a}, n_{k,b})\), mark these cases.

Figure 3.42: Single-rate structure for FT-based Hilbert transformer using a cascaded line of identical subfilters.

Now, consider a chain of cascaded identical subfilters \(G(z)\). Figure 3.43 presents the equivalent structure of cascaded filters obtained when the Pipelining-Interleaving (PI) technique is used instead. Note that each subfilter is followed by an additional delay. Therefore, the overall delay in the cascaded filter line is increased [72]. The input-output unions between the subfilters are denoted as \(m, n_{2}, n_{3}, n_{4}\) and so on.

Figure 3.43: Equivalent single-rate structure when the PI technique is used for a cascaded line of subfilters.

The chain of cascaded filters in Figure 3.43 can be used to replace the cascaded identical subfilters of Figure 3.42. However, every odd node,
namely, \( n_1, n_3 \) and in general \( n_k \) (with \( k \) being a positive odd number) of Figure 3.43 must be adequately separated in a pair of nodes \((n_{k,a}, n_{k,b})\) to obtain the overall structure that replaces structure of Figure 3.42. The resulting structure is presented in Figure 3.44. The transfer function for this structure is given as

\[
H(z) = G(z) \sum_{n=0}^{L_p/2-1} z^{-(l_{pc}+1) + 2(n)} \alpha(n) \left[ z^{-l_{pc}+1} + 2[z^{-l} G(z)]^2 \right]^n .
\]  (3.141)

The obtained structure has a line of cascaded subfilters that is the equivalent single-rate of the chain of cascaded filters derived from the PI technique. Therefore, the PI technique can be applied in a straightforward manner to avoid the repetitive use of the subfilter. Identifying the equivalent nodes \( n_{1,a}, n_{1,b}, n_2, n_{3,a}, n_{3,b} \) etc. in both, the line of cascaded filters of Figure 3.44 and its PI-based equivalent, we arrive to the proposed PI-based architecture presented in Figure 3.45.

The proposed PI-based structure has the following advantages:

- It requires the implementation of only one subfilter \( G(z) \).
• The coefficients of the subfilter, as well as the coefficients obtained from the prototype filter by using Chebyshev polynomials, are implemented only once.

• The overall number of multiplier coefficients in the proposed architecture is the same as the number of distinct coefficients required in the FT method.

![Diagram of Proposed PI-based structure](image)

**Figure 3.45:** Proposed PI-based structure.

### 3.2.1.2 Optimal design of Hilbert transformers based on FT method

The important insight proposed in this subsection is avoiding the arbitrary selection of the subfilter as in [69]. From (2.45)-(2.47), it can be observed that the prototype filter and the subfilter can be designed if the frequency \( \Omega_l \) is known. Thus, instead of choosing an arbitrary prototype filter
as in [69], an optimization-based search can be employed to find a proper value $\Omega_\ell$ such that an arbitrary cost measure is minimized.

Let us introduce the optimized search of the adequate prototype filter, such that the complexity of the overall filter, estimated from the weighted contributions of multiplier coefficients, adders and delay elements, is minimized. To this end, consider a function $\phi(x, y)$ that estimates with an acceptable exactitude the length of a Hilbert transformer in terms of its ripple $x$ and its low passband edge frequency $y$. We have,

$$L_G \approx L_G = \phi(\delta_G / v_d, \omega_L), \quad (3.142)$$

$$L_p \approx L_p = \phi(\delta, \Omega_L), \quad (3.143)$$

where $L_G$ and $L_p$ are the respective approximations to $L_G$ and $L_p$, whereas $v_d$ and $\delta_G$ are given in (2.47). Clearly, $L_G$ and $L_p$ are given as functions of the ripple and the low passband edge frequency of the subfilter and the prototype filter, respectively. Note that the ripple argument for $L_G$ in (3.142) is normalized to $v_d$, the desired amplitude value of the subfilter, because it is assumed that the function $\phi(x, y)$ is for cases with an amplitude equal to 1 (0 dB).

For the single-rate structure presented in Figure 3.42, the time-multiplexed structure of Figure 3.44 or, in general, any of the structures introduced in [43], it is always possible to express the overall numbers of multipliers, $m$, adders, $a$, and delays, $d$, in terms of the lengths of the prototype filter and the subfilter. This can be expressed as

$$m = f_m(L_G, L_p) \approx f_m(L_G, L_p), \quad (3.144)$$

$$a = f_a(L_G, L_p) \approx f_a(L_G, L_p), \quad (3.145)$$

$$d = f_d(L_G, L_p) \approx f_d(L_G, L_p). \quad (3.146)$$
Note that these functions depend on the structure that will be used. Therefore they are just represented with an \( f(x, y) \) form in this and the next subsections. In subsection 3.2.1.5 we will present the corresponding function according to the example being considered.

Substituting \( v_d \) and \( \delta_G \) from (2.47) in (3.142), and using the approximations (3.142) and (3.143) respectively in (3.144) to (3.146), we can express a cost \( c \) in terms of the approximated values \( m, a \) and \( d \) with a function \( f_c(m, a, d) \) as follows,

\[
c(\delta, \omega_L, \Omega_L) = f_c \left( f_m(L_G, L_p), f_a(L_G, L_p), f_d(L_G, L_p) \right) = f \left( L_G, L_p \right) = \\
f \left( \phi \left( \frac{1 - \sin(\Omega_L/2)}{1 + \sin(\Omega_L/2)}, \omega_L \right), \phi(\delta, \Omega_L) \right).
\]

(3.147)

The function \( f_c(m, a, d) \) can be, for example, with the form \( f_c(m, a, d) = \gamma_1 m + \gamma_2 a + \gamma_3 d \), where \( \{\gamma_1, \gamma_2, \gamma_3\} \in [0, 1] \) are factors that balance the relative complexity between multipliers, adders and delays. It is clear that this weighted cost function is not completely exact, but similar approaches with adequate complexity estimation using this type of weighting have been developed, for example, in [74]-[76]. Usually, the main concern has been on minimizing the number of multipliers, since they are the most demanding filtering elements in terms of power and area. Hence, \( \gamma_1 = 1 \) and \( \{\gamma_2, \gamma_3\} = 0 \) would be a practical choice.

Note that, even though \( c(\delta, \omega_L, \Omega_L) \) is a function of \( \delta, \omega_L \), and \( \Omega_L \), the values \( \delta \) and \( \omega_L \) are known a priori because they are given by the problem at hand (see (2.31)). Therefore, since the unique unknown is \( \Omega_L \), the approach consists in finding the optimum value \( \Omega_L^* \), with \( 0 < \Omega_L^* < \pi \), such that \( c(\delta, \omega_L, \Omega_L) \) is minimized. This optimization problem is given as

\[
\min_{\Omega_L \in \mathbb{R}} c(\delta, \omega_L, \Omega_L) \quad \text{such that} \quad \Omega_{\text{low}} \leq \Omega_L < \pi.
\]

(3.148)
Obviously, with this optimization framework the subfilter is not arbitrarily chosen, as in [69]. The general objective function in (3.147) estimates the cost (for example, the number of multipliers) for any given structure with the function \( f(L_G, L_p) \). However, such estimation depends on the exactitude of the function \( \phi(x, y) \).

Note that \( \Omega_{\text{low}} \) is the lower bound for \( \Omega_L \) in (3.148). This value is obtained as follows. First, consider that one of the most important characteristic of the FT-method relies in the use of several identical subfilters, which gives the possibility to couple the PI technique with FT-based Hilbert transformers to obtain a time-multiplexed implementation, as we presented in the previous subsection. However, the clock rate to operate the pipelined-interleaved subfilter grows \( K \) times, with \( K \) integer and \( K > 1 \), in a direct proportion with the length of the prototype filter as follows,

\[
K = L_p - 1. \tag{3.149}
\]

In second place, note that, since \( L_p \) is always an even number (see section 2.4.3), \( K \) is always odd for the structure proposed in Figure 3.45. Thus, the minimum increase in the clock rate, \( K_{\text{min}} \), is

\[
K_{\text{min}} = \min_{L_p} \{L_p - 1\}, \text{ such that } (L_p - 1) > 1 \text{ and } L_p \text{ is even}, \tag{3.150}
\]

which yields, \( L_p^{(\text{min})} = 4 \) and \( K_{\text{min}} = 3 \), respectively.

Let us consider that \( K \) is constrained to be less or equal to an arbitrary upper bound \( K_{\text{max}} \). Additionally, recall that \( L_p \) decreases if \( \Omega_L \) increases because the transition bandwidth \( \Omega_L/2\pi \) is nearly inversely proportional to the length of the Hilbert transformer. Therefore, to keep a value \( K \leq K_{\text{max}} \), which implies a relatively low value \( L_p \) in accordance with (3.149), \( \Omega_L \) should
be relatively high. Using (3.143) in (3.149), it is possible to obtain the lower bound $\Omega_{\text{low}}$ for a given $K_{\text{max}}$ by solving (3.151) for $\Omega_L$,

$$\phi(\delta, \Omega_L) - 1 = K_{\text{max}}.$$  \hfill (3.151)

Now, let us focus on the formula $\phi(x, y)$. The available function in literature at the time of developing this method was

$$\phi(x, y) = \left[ 0.002655 \left( \log_{10}(x) \right)^3 + 0.031843 \left( \log_{10}(x) \right)^2 ight.$$  

$$-0.554993 \log_{10}(x) - 0.049788 \pi y + 1, \hfill (3.152)$$

derived in [71] from the one introduced by O. Hermann et al. in [56]. With the use of the normalized ripple argument of $L_G$ in (3.142) the objective function (3.147) has the problem of estimating the number of required multiplier coefficients as zero or negative values. As an example, for a Type III Hilbert transformer with the following specification, $\omega_L = 0.0001\pi$ and $\delta = 0.0001$, we get an estimated overall number of multipliers, $m$, about $-220$, which is senseless. Using (3.142) and (3.143), the estimated lengths for the prototype filter and the subfilter are, respectively, $L_G \approx -719$ and $L_P \approx 276$. While the estimated length $L_P$ can be considered proper, the value for $L_G$ is completely pointless. Therefore, the inexactitude of the function $\phi(x, y)$ in (3.152) may lead to non-optimal solutions.

To solve the aforementioned problem, we have proposed the derivation of a new function to estimate the length of FIR Hilbert transformers. To our knowledge, a most exact formula to estimate the length of a FIR filter optimized under the minimax criterion is given by I. Koichi et al. [58]. We use the following straightforward approach to adjust this formula for the case of Hilbert transformers:
1. Consider that $L_{III}$ and $L_{IV}$ are the lengths of a Type III and a Type IV Hilbert transformer, respectively. From the relation between a half-band filter and a Type III Hilbert transformer we have the passband and stopband edge frequencies of the former, $\omega_p$ and $\omega_s$, as well as its passband and stopband ripples, $\delta_p$ and $\delta_s$, given in terms of the low passband edge frequency $y$ and the ripple $x$ of the latter as follows:

$$\omega_p = \pi/2 - y, \quad \omega_s = \pi/2 + y, \quad \delta_p = \delta_s = x/2.$$  \hspace{1cm} (3.153)

Thus, we substitute the half-band filter specification in the general formula given by I. Koichi et al. using the expressions for $\omega_p$, $\omega_s$, $\delta_p$ and $\delta_s$ in terms of $x$ and $y$ and with this we can estimate the length $L_{III}$.

2. A Type III Hilbert transformer is derived from a Type IV Hilbert transformer by substituting each delay with two delays. With this, the lengths are related as $L_{III} = 2L_{IV} - 1$ and the respective low passband edge frequencies of the Type III Hilbert transformer and the Type IV Hilbert transformer are $y < \pi/2$ and $\bar{y} = 2y$. Let us express $L_{IV} = (L_{III} + 1)/2$.

Substituting $L_{III}$ from step 1, $L_{IV}$ becomes an estimation of the length of a Type IV Hilbert transformer with ripple $x$ and low passband edge $2y$. In order to be able to use $y$ in the range $0 < y < \pi$, the transition band considered in the estimation of $L_{III}$ of step 1 must be divided by 2.

Following the two aforementioned steps, we get

$$\phi_r(x, y) = \frac{1}{2} + \left[ \frac{1.101 \left[- \log_{10}(x) \right]^{1.1}}{(y/\pi)} \right] + 1 \left[ \frac{2}{3\pi} \arctan \left[ \frac{2.325 \times \left[ 0.30103 - \log_{10}(x) \right]^{0.445} \left( \frac{y}{2\pi} \right)^{-1.39} \times \left[ \frac{1}{0.5 - \left( \frac{y}{2\pi} \right)} \right] + \frac{1}{6} \right] \right].$$ \hspace{1cm} (3.154)

Let us compare the formula given in (3.152) with the proposed formula. To this end, consider again (2.45)-(2.47). Note that the prototype filter and the
subfilter have, respectively, the ripple specification and the transition band specification of the desired filter. According to the proposed optimization approach, the optimum value \( \Omega^*_L \) balances the narrowing of the transition band of the prototype filter and the shrinking of the ripple of the subfilter in such a way that the corresponding cost function required in a given structure is minimized, as is illustrated in Figure 3.46. Therefore, it can be assumed that the prototype filter will have a non-stringent transition band specification, whereas the subfilter will have a non-stringent ripple specification.

![Figure 3.46](image)

**Figure 3.46**: Transition band of the prototype filter and ripple of the subfilter as functions of \( \Omega^*_L \). As \( \Omega^*_L \) increases, the Prototype filter becomes simpler because its transition band increases. On the contrary, the subfilter becomes more complex because its ripple decreases.

With the aforementioned considerations in mind, let us analyze (3.152) and (3.154) for two cases. The first one corresponds to prototype filters, i.e., the design of Type IV Hilbert transformers using 20 different low passband edge values \( \omega_L \) equally spaced in the range from \( 0.1\pi \) to \( 0.9\pi \), considering the ripple values \( \delta = 0.0001 \) and \( \delta = 0.005 \). The second one corresponds to
subfilters, i.e., the design of Type III Hilbert transformers using 20 different ripple values \( \delta \) equally spaced in the range from 0.1 to 0.9, considering the low passband edge values \( \omega_L = 0.0001\pi \) and \( \omega_L = 0.005\pi \). In this second case the choice of Type III Hilbert transformers is arbitrary and the results for Type IV subfilters are quite similar. Figure 3.47 shows the percentage of absolute error for the estimation of the length corresponding to prototype filters. Figure 3.48 shows the percentage of absolute error for the estimation of the length of the subfilters.

![Graph showing the percentage of absolute error for the estimation of lengths of Hilbert transformers](image)

**Figure 3.47:** Percentage of absolute error in the estimation of lengths of Hilbert transformers using equations (3.152) and (3.154) for cases with ripples \( \delta = 0.0001 \) and \( \delta = 0.005 \), with \( \Omega_L \) in the range from \( 0.1\pi \) to \( 0.9\pi \).

From Figure 3.47 we have that (3.152) and (3.154) exhibit almost the same percentage of error for the length estimation of prototype filters. For any low passband edge less than \( \Omega_L = 0.6\pi \), the maximum absolute error is no more than 13% for the cases with ripple \( \delta = 0.005 \) and no more than 17% for the
cases with ripple $\delta = 0.0001$. For the other passband edge values some absolute errors are between 20% and 30% and for the specification $\Omega_L = 0.9\pi$ and $\delta = 0.0001$ the absolute error is about 65%. However, note that in these cases the filter length is small because the transition band is considerably wide. Therefore, even though the percentage of error is high, the difference between the estimated and the real length is small. For the highest percentage of error (65%), the estimated and the real lengths are 6.6 and 4, respectively. In such case the estimated number of multipliers is 3.3, whereas the real is 2.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3_48.png}
\caption{Percentage of absolute error in the estimation of lengths of Hilbert transformers using equations (3.152) and (3.154) for cases with relative frequencies $\omega_L = 0.0001\pi$ and $\omega_L = 0.005\pi$, with $\delta$ in the range from 0.1 to 0.9.}
\end{figure}

Fig. 3.48 shows that (3.154) has a much less percentage of error than (3.152) for the length estimation of subfilters. Using (3.152), for any ripple less than $\delta = 0.3$ the maximum absolute error is less than 3% in both cases, with low passband edge $\omega_L = 0.0001\pi$ and with $\omega_L = 0.005\pi$. Using (3.154), the maximum absolute error is less than 5% in both cases, with low passband
edges $\omega_L = 0.0001\pi$ and $\omega_L = 0.005\pi$. As the ripple value $\delta$ increases, equation (3.152) shows a growing error which is close to or more than 100% when $\delta \geq 0.8$. On the other hand, equation (3.154) exhibits a good length estimation over all the cases proved, having a worst absolute error less than 24%. It is obvious that (3.154) does provide reliable results for cases with wide ripple and it can be used in the proposed optimization-based FT method.

Finally, it is worth highlighting that the function (3.152) provides a good estimation of the length of a Hilbert transformer whenever the ripple deviation is less than 0.3, which is commonly the case for prototype filters in the FT method. Therefore, we can take advantage of this characteristic to obtain the value $\Omega_{\text{low}}$ by replacing (3.152) in (3.151). We obtain

$$\Omega_{\text{low}} = 2\pi \left[0.002655(\log_{10}(\delta))^3 + 0.031843(\log_{10}(\delta))^2 - 0.554993\log_{10}(\delta) - 0.049788\right] / K_{\text{max}}.$$  

(3.155)

### 3.2.1.3 Multi-level FT method for Hilbert transformers

The proposed optimization approach to design FT-based Hilbert transformers paves the way to study the extension of such method to a generalized multi-level scheme. The multi-level concept was recently introduced in [77], independently of the research pursued here. However, the subsequent analysis follows a more complete study based on the proposed optimization-based framework.

The multi-level approach is explained as follows. Consider a case where the desired filter is designed using the FT method and the generated subfilter is designed once more using FT. We call first level to the inner FT structure, the external FT structure is called second level and the complete design is
based on a two-level scheme. Generalizing this concept, the $M$-level approach involves the use of the FT method $M$ times in the overall desired filter, designing each one of the first $M – 1$ external subfilters with FT and preserving a direct design of the innermost subfilter. Observe that, since the first $M – 1$ external subfilters are designed using FT, we finally have only one direct subfilter, i.e., the innermost one. We consider it the first subfilter of the overall design and, if there is no place to confusion, we will refer to it simply as the subfilter. Therefore, given a desired specification for the magnitude response $|H(\omega)|$ of a FIR Hilbert transformer as in (2.31), the $M$-level FT method requires the design of $M + 1$ simple filters, namely, $M$ prototype filters and the subfilter.

From (2.44) we derive the transfer function of a FT-based Hilbert transformer placed in the $i$-th level, with $i$ ranged from 1 to $M$, as follows

$$
H_i(z) = \begin{cases}
G(z) \sum_{n=0}^{\lfloor L_{P,i}/2 \rfloor - 1} \left\{ z^{-(L_{P,i} - 1)(\lfloor L_{P,i}/2 \rfloor - n)} \alpha_i(n) \times 
\left[ z^{-(L_{P,i} - 1)} + 2G^2(z) \right]^n \right\}; & \text{for } i = 1 \\
H_{i-1}(z) \sum_{n=0}^{\lfloor L_{P,i}/2 \rfloor - 1} \left\{ z^{-(L_{P,i} - 1)(\lfloor L_{P,i}/2 \rfloor - n)} \alpha_i(n) \times 
\left[ z^{-(L_{P,i} - 1)} + 2H_{i-1}^2(z) \right]^n \right\}; & \text{for } i = 2, 3, ..., M
\end{cases}
$$

(3.156)

$$
L_{H,i}(z) = \begin{cases}
(L_{P,i} - 1)L_G - L_{P,i} + 2; & \text{for } i = 1, \\
(L_{P,i} - 1)L_{H,i-1} - L_{P,i} + 2; & \text{for } i = 2, 3, ..., M.
\end{cases}
$$

(3.157)

The subscript $i$ in $H_i(z)$ indicates the overall transfer function in the $i$-th level, thus the overall $M$-level transfer function is $H_M(z)$. $L_{H,i}$ and $L_{P,i}$ are respectively the overall length in the $i$-th level and the length of the $i$-th prototype filter.
(the one placed in the $i$-th level), whereas $L_G$ is the subfilter length. Additionally, the coefficients $\alpha_i(n)$ with $n$ ranged from 0 to $L_{P,i}/2 - 1$ can be derived from the coefficients of the $i$-th prototype filter through the Chebyshev polynomials [69]. Note that the transfer function in (3.156) corresponds to an overall single-rate structure.

To accomplish the desired specification for $|H(\omega)|$, the magnitude response $|P_i(\Omega)|$ of the $i$-th prototype filter must satisfy the following condition

$$ (b_i - \delta_i) \leq |P_i(\Omega_i)| \leq (b_i + \delta_i), \quad \text{for } \Omega_{L,i} \leq \Omega_i \leq \pi, \quad (3.158) $$

where $\Omega_i$ denotes the frequency domain of the $i$-th prototype filter and $\Omega_{L,i}$ is its unknown low passband edge frequency. The values $b_i$ and $\delta_i$ are respectively the amplitude and the ripple of the subfilter placed in the nearest external level, i.e, the $(i+1)$-th subfilter. Note that $b_i = 1$ and $\delta_i = \delta$ for the case $i = M$, i.e., the $(M+1)$-th subfilter becomes the overall desired Hilbert transformer. Therefore, we have

$$ b_i = \begin{cases} 
1; & \text{for } i = M, \\
\frac{1}{2} + \frac{1}{2} \sin \left( \frac{\Omega_{L,i}}{2} \right); & \text{for } i = 1, 2, ..., M-1,
\end{cases} \quad (3.159) $$

$$ \delta_i = \begin{cases} 
\delta; & \text{for } i = M, \\
\frac{1}{2} - \frac{1}{2} \sin \left( \frac{\Omega_{L,i}}{2} \right); & \text{for } i = 1, 2, ..., M-1.
\end{cases} \quad (3.160) $$

The magnitude response of the first subfilter, $|G(\omega)|$, must fulfill simultaneously the specification given in (2.46)-(2.47), only substituting $\Omega_{L,1}$ instead of $\Omega_L$. 

165
Generalizing (3.144) to (3.166), the numbers of multipliers, adders and delays in the $i$-th level are

$$
m^{(i)} \approx \begin{cases} 
  f_m^{(i)}(L_G, L_{p,i}); & \text{for } i = 1, \\
  f_m^{(i)}(m^{(i-1)}, L_{p,i}); & \text{for } i = 2, 3, ..., M.
\end{cases}
$$

(3.161)

$$
d^{(i)} \approx \begin{cases} 
  f_a^{(i)}(L_G, L_{p,i}); & \text{for } i = 1, \\
  f_a^{(i)}(d^{(i-1)}, L_{p,i}); & \text{for } i = 2, 3, ..., M.
\end{cases}
$$

(3.162)

$$
d^{(i)} \approx \begin{cases} 
  f_d^{(i)}(L_G, L_{p,i}); & \text{for } i = 1, \\
  f_d^{(i)}(d^{(i-1)}, L_{p,i}); & \text{for } i = 2, 3, ..., M.
\end{cases}
$$

(3.163)

Observe that $m^{(i)}$ and $f_m^{(i)}(x, y)$ respectively indicate the number of multipliers in the $i$-th level and the corresponding function to obtain the number of multipliers for a given structure in such level. The same holds for $a^{(i)}$ and $f_a^{(i)}(x, y)$ with regard to the number of adders and $d^{(i)}$ and $f_d^{(i)}(x, y)$ with regard to the number of delays. Note that the arguments in these functions are the estimated lengths of the subfilter and the lengths of all the prototype filters, from the first prototype filter to the $i$-th prototype filter. These lengths can be approximated using (3.142) and (3.143) with the following substitutions:

1) Use the function $\phi_p(x, y)$ of (3.154) in (3.142) and (3.143).

2) Replace $\delta$ with $\delta_i/b_i$ and $\Omega_l$ with $\Omega_{L,i}$ in (3.143) to obtain $L_{p,i}$, with $b_i$ and $\delta_i$ respectively given in (3.159) and (3.160).

3) Replace $\Omega_l$ with $\Omega_{L,1}$ in (2.47) to obtain the values $v_d$ and $\delta_G$ that are used in (3.142).

With the previous substitutions we have that $m^{(i)}$, $a^{(i)}$ and $d^{(i)}$ in (3.161) to (3.163) are given in terms of $\omega_L$ (the desired low passband edge specification), 
\[ c(\delta, \omega_L, \Omega_{L,1}, \ldots, \Omega_{L,M}) = f_c(f_m(m^{(M-1)}, L_{p,M}), f_a(a^{(M-1)}, L_{p,M}), f_d(d^{(M-1)}, L_{p,M})) = \\
= f\left(C, \Omega_{L,1}, \ldots, \Omega_{L,M}\right) = \\
= f\left(\Phi_p(1-\sin(\Omega_{L,1}/2), \omega_L), \Phi_p(1-\sin(\Omega_{L,2}/2), \Omega_{L,1}), \ldots, \Phi_p(\delta, \Omega_{L,M})\right). \]

(3.164)

As we did in the previous subsection for the single-level case, the function \( f_c(m^{(M)}, a^{(M)}, d^{(M)}) \) can be, for example, with the form \( f_c(m^{(M)}, a^{(M)}, d^{(M)}) = \gamma_1 m^{(M)} + \gamma_2 a^{(M)} + \gamma_3 d^{(M)} \). Note that the values \( \delta \) and \( \omega_L \) are known a priori. Therefore, the approach consists in finding the optimal point \( \Omega^* = [\Omega_{L,1}, \Omega_{L,2}, \ldots, \Omega_{L,M}]^T \) in an \( M \)-dimensional space, such that \( c(\delta, \omega_L, \Omega_L) \) is minimized. This optimization problem is given as follows,

\[
\min_{\Omega_L \in \mathbb{R}^M} c(\delta, \omega_L, \Omega_L) \quad \text{such that} \quad 0 < \Omega_{L,i} < \pi, \forall i \in 1, 2, \ldots, M. \quad (3.165)
\]

### 3.2.1.4 Design of Hilbert transformers using a combined FRM-FT approach

In the following we present a strategic combination of the FT and the Frequency-Response Masking (FRM) methods which takes advantage of the characteristics of both techniques to obtain an overall low-complexity design. The proposal relies on designing the subfilter \( G(z) \) of the FT-based structure with the FRM technique. The motivation rises from the following observations.

- In the FT method the prototype filter is a simple filter because it has a non-stringent transition band specification. Similarly, the subfilter can
be considered simple because it has a non-stringent ripple specification. However, since the transition bandwidth specification of the subfilter is stringent (the same as the desired Hilbert transformer), the subfilter still may have a high length. Its ripple can not be arbitrarily aggrandized, since it compromises the transition band of the prototype filter.

- The FRM approach is convenient to obtain low-complexity Hilbert transformers whose transition band specification is stringent [70], [71]. If the ripple specification of a FRM-based Hilbert transformer is not too small, the resulting filter can have a much lower complexity compared to a direct design, because the main impact in the filter complexity designed with a direct method relies on the transition band.

From the above observations, is clear that a FRM-based subfilter will provide a number of advantages, as follows.

- The ripple of the subfilter will not need to be extremely wide, and therefore the length of the prototype filter will be preserved small.
- The overall architecture is based on FT. Therefore, one can take advantage of the repetitive use of identical subfilters with, for example, the PI-based structure presented in Figure 3.45.
- The subfilter will require a short wordlength, due to the size of its ripple. Therefore, rounding can be applied to get a simple multiplierless subfilter.
- A complexity reduction will be obtained in the overall design, better than using a subfilter designed with the direct method.
We have seen in subsection 3.2.1.2 (see Figure 3.46) that the narrowing of the transition band of the prototype filter and the shrinking of the ripple of the subfilter should be balanced such that a given cost (usually the overall number of multiplier coefficients) is minimized. For the proposed scheme, such balance is found with the optimal value of the lower band-edge frequency of the prototype filter, $\Omega_L$, and the optimal interpolation factor of the subfilter, $M$. The steps of design for the proposed scheme are detailed as follows.

**Step 1: Calculation of the optimal value $M$ for the FRM-based subfilter design**

Since the FRM-based design of a Hilbert transformer is straightforwardly derived from the design of a FRM-based half-band filter, the optimal factor $M$ is estimated as,

$$M \approx (1/2)\sqrt{\pi/\omega_L}.$$  

This formula is directly derived from the one proposed in [78]. The obtained value must be rounded to the nearest odd integer.

**Step 2: Calculation of the optimal value $\Omega_L$ for the design of the prototype filter**

As we mentioned in subsection 3.2.1.2, the optimal value $\Omega_L$ can be obtained by minimizing a cost function that is expressed in terms of the lengths of the prototype filter and the subfilter. In this proposal, the subfilter is composed by the filters $H_{lb}(z)$ and $H_{ma}(z)$ (see subsection 2.4.5). Therefore,
we express the overall numbers of multipliers, $m$, adders, $a$, and delays, $d$ as follows,

$$m = f_m(L_{hb}, L_{ma}, L_P) \approx f_m(L_{hb}, L_{ma}, L_P),$$  \hspace{1cm} (3.167)

$$a = f_a(L_{hb}, L_{ma}, L_P) \approx f_a(L_{hb}, L_{ma}, L_P),$$  \hspace{1cm} (3.168)

$$d = f_d(L_{hb}, L_{ma}, L_P) \approx f_d(L_{hb}, L_{ma}, L_P),$$  \hspace{1cm} (3.169)

where $L_{hb}$ and $L_{ma}$ are the lengths of the filters $H_{hb}(z)$ and $H_{ma}(z)$, respectively. The respective estimated lengths, $L_{hb}$ and $L_{ma}$, are obtained as follows,

$$L_{hb} \approx \psi(0.425 \delta_c / v_d, \omega_p, \omega_s),$$  \hspace{1cm} (3.170)

$$L_{ma} \approx \psi(0.425 \delta_c / v_d, \theta_p, \theta_s),$$  \hspace{1cm} (3.171)

where $\delta$ is the desired ripple, $\delta_c$ and $v_d$ are given in (2.47) and $\omega_p$, $\omega_s$, $\theta_p$ and $\theta_s$ are respectively given in (2.52) to (2.55) if $(M-1)/4$ is an integer or respectively given in (2.56) to (2.59) if $(M+1)/4$ is an integer. The function $\psi(w, x, y)$, obtained from [58], is given as

$$\psi(w, x, y) \approx \left( \frac{1.101(-\log_{10}(2w))^{1.1}}{(y-x)/2\pi} + 1 \right).$$

$$\left[ \frac{2}{\pi} \arctan \left\{ \left( 2.325 (-\log_{10} w)^{-0.445} \cdot ((y-x)/2\pi)^{-1.39} \cdot \left( \frac{1}{x/2\pi} - \frac{1}{0.5 - (y-x)/2\pi} \right) \right) \right\} + \frac{2}{\pi} \arctan \left\{ \left( 2.325 (-\log_{10} w)^{-0.445} \cdot ((y-x)/2\pi)^{-1.39} \cdot \left( \frac{1}{x/2\pi} - \frac{1}{0.5 - (y-x)/2\pi} \right) \right) \right\} \right] + 1 \right] \frac{1}{3}. \hspace{1cm} (3.172)$$

The estimated length $L_P$ is obtained using (3.143), with the function $\phi(x, y)$ being replaced by the function $\phi_p(x, y)$ given in (3.154).

We can express a cost $c$ in terms of the approximated values $m$, $a$ and $d$ with a function $f_c(m, a, d)$ as follows,
\[ c(\delta, \omega_L, M, \Omega_L) = f_c \left( f_m(L_{hb}, L_{ma}, L_P), f_d(L_{hb}, L_{ma}, L_P), f_d(L_{ma}, L_{d}), (L_{ma}, L_{d}) \right) \]
\[ = f(L_{hb}, L_{ma}, L_P) \]  
(3.173)

With respect to the arguments required to obtain \( L_{hb} \) and \( L_{ma} \) (see (3.170) and (3.171)), it can be noted from (2.47) that \( \delta_G \) and \( v_d \) are functions of \( \Omega_L \). Similarly, from (2.52) to (2.55) or from (2.56) to (2.59) we can see that \( \omega_p, \omega_s, \theta_p \) and \( \theta_s \) are given in terms of \( \omega_L \) and \( M \), which respectively are the desired lower passband edge frequency and the interpolation factor obtained in the previous step. Additionally, from (3.143) we note that \( \varphi \) is given in terms of \( \delta \), the desired ripple specification, and \( \Omega_L \). Because of this, the cost function \( c(\delta, \omega_L, M, \Omega_L) \) in (3.173) explicitly presents these arguments. Since \( \delta, \omega_L \) and \( M \) are a priori known, the objective is finding the optimum value \( \Omega_L^* \), with \( 0 < \Omega_L < \pi \), such that \( c(\delta, \omega_L, \Omega_L) \) is minimized. This optimization problem is given as

\[ \min_{\Omega_L \in \mathbb{R}} c(\delta, \omega_L, M, \Omega_L) \quad \text{such that } \Omega_{low} \leq \Omega_L < \pi. \]  
(3.174)

If the time-multiplexed structure proposed in subsection 3.2.1.1 is used, the value \( \Omega_{low} \) can be found in terms of a given maximum clock increase \( K_{max} \) using (3.155).

**Step 3: Design of the prototype filter and the subfilter**

Once known \( M, \Omega_L \) and the estimated lengths \( L_P, L_{hb} \) and \( L_{ma} \), we design the prototype filter to accomplish the specification (2.45) (see subsection 2.4.3). The subfilter is designed with the Frequency Response Masking (FRM) method introduced in section 2.4.5). To this end, we design the half-band filter \( H_{low}(z) \) with the pass-band and stop-band edge frequencies given in (2.52) and (2.53) if \( (M-1)/4 \) is an integer or in (2.56) and (2.57) if \( (M+1)/4 \) is an
integer. Its ripples are $\delta_p = \delta_s = 0.425 \delta_G/v_d$. The low-pass filter $H_{ma}(z)$ must be designed with the pass-band and stop-band edge frequencies given in (2.54) and (2.55) if $(M-1)/4$ is an integer or in (2.58) and (2.59) if $(M+1)/4$ is an integer, and its ripples are also $\delta_p = \delta_s = 0.425 \delta_G/v_d$. Finally, from the filter $H_{ma}(z)$ we obtain $B(z)$ and $C(z)$ using (2.60) and (2.61).

With $H_{ma}(z)$, $B(z)$ and $C(z)$, the filters $H_{s}(z)$, $H_{s}(z^{M})$ and $H_{m}(z)$ are designed using (2.49) to (2.51). Then, the subfilter $G(z)$ is formed using (2.48). Note that the obtained subfilter must be scaled by $v_d$.

### 3.2.1.5 Design examples and discussion of results

The following couple of examples compare the proposed method in terms of computational complexity with those presented in [71], [77] and [79], which to our knowledge are the most efficient to design FIR Hilbert transformers.

**Example 14:** Design a Type III Hilbert transformer to satisfy (2.31) with $\delta = 0.004$ and $\omega_L = 0.01\pi$ with a minimum number of multipliers. Consider that the filter is restricted to operate at a clock rate not higher than $K_{\text{max}} = 9$ times the data rate.

Since the clock rate is allowed to be higher than the data rate, we use the structure proposed in subsection 3.2.1.1. For such structure, the estimated overall number of multiplier coefficients is $m = f_m(L_G, L_P) = L_G/4 + L_P/2$, with $L_G = \phi_p(\delta_G/v_d, \omega_L)$, $L_P = \phi_p(\delta, \Omega_L)$, $\delta_G$ and $v_d$ given in (2.47) and $\phi_p(x, y)$ introduced in (3.154). The objective to minimize is the number of multipliers, thus, with the proper substitutions we get the cost function $c(\delta, \omega_L, \Omega_L) = c(0.004, 0.01\pi,$
\( \Omega_L = \phi_p \left( \frac{1 - \sin(\Omega_L/2)}{1 + \sin(\Omega_L/2)} \right), \frac{0.01\pi}{4} + \phi_p(0.004, \Omega_L)/2 \) and the optimization consists on finding the optimal value \( \Omega_L \) that minimizes \( c(0.004, 0.01\pi, \Omega_L) \). The lower bound for \( \Omega_L \), which is a consequence of the aforementioned clock rate constraint, is obtained using (3.155), resulting in \( \Omega_{low} = 0.3173\pi \). The function \texttt{fminbnd} from the Optimization Toolbox of MATLAB is employed to solve the optimization problem. After the optimization we obtain \( \Omega_L = 0.3173\pi, L_P = 10 \) and \( L_G = 47 \). The overall number of multipliers, which is also the number of distinct coefficients in the FT method, is equal to 17, i.e., \( (L_G+1)/4 = 12 \) multipliers belonging to the subfilter plus \( L_P/2 = 5 \) structural multipliers obtained from the prototype filter using Chebyshev Polynomials. The coefficients of the subfilter are rounded to allow 8 bits in the fractional part, whereas the coefficients of the prototype filter are rounded to allow 10 bits in the fractional part. The overall number of adders is 32, i.e., \( (L_P-2)/2 = 24 \) adders belonging to the subfilter plus \( (L_P - 2) = 8 \) structural adders. The subfilter is operated at a clock rate \( K \) times higher than the data rate, with \( K = (L_P - 1) = 9 \). Thus, the filter performs \( 12 \times 9 + 5 = 113 \) Multiplications Per Output Sample (MPOS) and \( 24 \times 9 + 8 = 224 \) Additions Per Output Sample (APOS). Tables 3.10 and 3.11 show the coefficients of the subfilter and the structural coefficients, respectively.

The recently proposed method [77], even though being a heuristic, provides near-optimal solutions when the minimum number of distinct coefficients is searched. That method completely relies on assuming that the numbers of multipliers of both, the prototype filter and the subfilter, are nearly equal. With that method we obtain \( L_P = 16 \) and \( L_G = 27 \), which results in a number of distinct multipliers equal to 15. However, method [77] does not
consider clock rate constraints and the proposed PI-based structure cannot be employed because it does not meet the clock rate constraint.

<table>
<thead>
<tr>
<th>Table 3.10: Coefficients of the subfilter $G(z)$ in Example 14</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g(2k) = -g(L_G - 1 - 2k)$, $g(2k+1) = g(L_G - 1 - 2k - 1) = 0$, $k = 0, 1, \ldots, (L_G+1)/4$;</td>
</tr>
<tr>
<td>$g(0) = -0.1328125$</td>
</tr>
<tr>
<td>$g(2) = -0.0234375$</td>
</tr>
<tr>
<td>$g(4) = -0.0234375$</td>
</tr>
<tr>
<td>$g(6) = -0.02734375$</td>
</tr>
<tr>
<td>$g(8) = -0.03125$</td>
</tr>
<tr>
<td>$g(10) = -0.03515625$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 3.11: Structural coefficients in Example 14</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha(0) = -1.41015625$</td>
</tr>
<tr>
<td>$\alpha(1) = -0.6953125$</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Even though method [77] does not mention explicitly a given structure for the designed Hilbert transformers, it suggests a PI-based structure with 3 implemented subfilters, namely, a single subfilter $G(z)$ cascaded with a time-multiplexed block $F(z)$ (one of the grey blocks in Figure 3.42). Such structure is obtained by mapping straightforwardly the chain of cascaded blocks $F(z)$ into its PI equivalent, as was done in [72] for the Kaiser-Hamming sharpening structure. With that structure, the clock rate to operate the multiplexed block is $L_P/2 - 1 = 7$ times the data rate. However, the overall number of multipliers employed in the structure is 29, i.e., $L_P/2 = 8$ structural multipliers plus $3(L_G+1)/4 = 21$, the number of multipliers of 3 implemented subfilters. Similarly, the overall number of adders is 49, i.e., $(L_P/2-1) = 7$ structural
adders plus \(3(L_G+1)/2 = 42\), the number of adders of 3 implemented subfilters. The filter performs \((2 \times 7 \times 7) + 7 + 8 = 113\) MPOS and \((2 \times 14 \times 7) + 14 + 7 = 217\) APOS. These results are summarized in Table 3.12. Figures 3.49, 3.50 and 3.51 respectively show the magnitude response of the subfilter, the prototype filter and the resulting overall Hilbert transformer. Note that the filter has the transition band specifications of the subfilter and the ripple specification of the prototype filter.

![Magnitude response of the subfilter G(z) in Example 14.](image)

**Figure 3.49:** Magnitude response of the subfilter \(G(z)\) in Example 14.

<table>
<thead>
<tr>
<th>Method</th>
<th>Clock rate increase with respect to data rate</th>
<th>No. Multipliers / No. Adders</th>
<th>MPOS / APOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>[77]</td>
<td>(K = 7)</td>
<td>29 / 49</td>
<td>213 / 217</td>
</tr>
<tr>
<td>Proposed</td>
<td>(K = 9)</td>
<td>17 / 32</td>
<td>213 / 224</td>
</tr>
</tbody>
</table>

**Table 3.12:** Comparison of results for the design of a Hilbert transformer in Example 14
Example 15: Design a Type III Hilbert transformer to satisfy (2.31) with δ = 0.0001 and \( \omega_L = 0.00125\pi \) with a minimum number of distinct coefficients.

In this example we employ the FRM-FT method introduced in subsection 3.2.1.4. The design steps are as follows.
**Step 1:** The interpolation factor $M = 15$ is obtained using (3.166) and rounding the result to the nearest odd integer.

**Step 2:** The overall number of distinct coefficients is $m \approx f_m(L_{hb}, L_{ma}, L_P) = (L_{hb} + 1)/4 + (L_{ma} + 1)/2 + L_P/2 + 1$, with $L_{hb} = \psi(0.425\delta c/\nu_d, \omega_p, \omega_s)$, $L_{ma} = \psi(0.425\delta c/\nu_d, \theta_p, \theta_s)$ and $L_P = \phi_p(\delta, \Omega_L)$. The functions $\psi(w, x, y)$ and $\phi_p(x, y)$ are respectively given in (3.172) and (3.154). Since $(M + 1)/4$ is an integer, the values $\omega_p, \omega_s, \theta_p$ and $\theta_s$ are found using (2.56) to (2.59). With these values we obtain the cost function

$$c(0.0001, 0.00125\pi, 15, \Omega_L) = \left[\psi\left(0.425\frac{\sin(\Omega_L/2)}{1+\sin(\Omega_L/2)} , 0.4813\pi, 0.5187\pi\right) + 1\right]/4 + \left[\psi\left(0.425\frac{\sin(\Omega_L/2)}{1+\sin(\Omega_L/2)} , 0.4346\pi, 0.4988\pi\right) + 1\right]/2 + \phi_p(0.0001, \Omega_L)/2 + 1.$$

After the optimization we obtain $\Omega_L^* = 0.2674\pi$.

**Step 3:** The lengths of the filters are $L_{hb} = 35, L_{ma} = 21$ and $L_P = 20$. The subfilter $G(z)$ is formed in terms of the half-band filter $H_{hb}(z)$ and the masking filter $H_{ma}(z)$ using (2.48) as explained in subsection 2.4.5. The coefficients of $H_{hb}(z)$ and $H_{ma}(z)$ are presented in Table 3.13 and Table 3.14, respectively. The structural coefficients obtained from the prototype filter, which are rounded to allow 17 bits in the fractional part, are shown in Table 3.15. Note that the coefficients of the subfilter do not require multipliers. Even though the subfilter must be scaled by the constant $\nu_d = 0.7039$, the rounded scaling $\nu_{d,r} = 2^{-4} \times (2^{3} + 2^{1} + 2^{0})$ yields good results. Thus, the subfilter is a complete multiplierless subsystem. The subfilter uses $(L_{hb} + 1)/2 + L_{ma} + 5 = 44$ adders plus 2 extra adders from the scaling $\nu_{d,r}$, resulting in 46 adders in total. Additionally $L_P/2 - 1 = 9$ structural adders are required. Therefore, the overall number of adders in the proposed structure (Figure 3.45) is $46 + 9 = 55$ adders.
The overall number of multipliers is 10, i.e., $L_P/2 = 10$ structural multipliers obtained from the prototype filter.

<table>
<thead>
<tr>
<th>Table 3.13: Coefficients of $H_{\text{fb}}(z)$ in Example 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_{\text{fb}}(2k) = h_{\text{fb}}(L_{\text{fb}} - 1 - 2k)$, $h_{\text{fb}}(2k+1) = h_{\text{fb}}(L_{\text{fb}} - 1 - 2k - 1) = 0$, $k = 0, 1, \ldots, (L_{\text{fb}}+1)/4$;</td>
</tr>
<tr>
<td>$h_{\text{fb}}(0) = 2^4$</td>
</tr>
<tr>
<td>$h_{\text{fb}}(2) = -2^6$</td>
</tr>
<tr>
<td>$h_{\text{fb}}(4) = 2^6$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 3.14: Coefficients of $H_{\text{ma}}(z)$ in Example 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_{\text{ma}}(n) = h_{\text{ma}}(L_{\text{ma}} - n)$</td>
</tr>
<tr>
<td>$h_{\text{ma}}(0) = 2^{-5}$</td>
</tr>
<tr>
<td>$h_{\text{ma}}(1) = 2^4$</td>
</tr>
<tr>
<td>$h_{\text{ma}}(2) = -2^4$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 3.15: Structural coefficients in Example 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha(0) = -1.414215087890625$</td>
</tr>
<tr>
<td>$\alpha(1) = -0.705902099609375$</td>
</tr>
<tr>
<td>$\alpha(2) = -0.5299072265625$</td>
</tr>
<tr>
<td>$\alpha(3) = -0.4698486328125$</td>
</tr>
<tr>
<td>$\alpha(4) = -0.4078369140625$</td>
</tr>
</tbody>
</table>

Figures 3.52, 3.53 and 3.54 respectively show the magnitude responses of the subfilter $G(z)$, the prototype filter $P(z)$ and the overall Hilbert transformer $H(z)$. Table 3.16 shows the comparison with methods [71] and [79]. Note that the saving in terms of number of multipliers and adders is substantial because methods [71] and [79] do not use time-multiplexed hardware. The computational complexity is also shown in Table 3.16. Since the subfilter is
operated at a clock rate \((L_p - 1)\) times higher than the data rate, the overall number of APOS is \((L_p - 1) \times 46 + 9 = 883\). The MPOS is preserved equal to 10.

**Figure 3.52**: Magnitude response of the subfilter \(G(z)\) in Example 15.

**Figure 3.53**: Magnitude response of the prototype filter \(P(z)\) in Example 15.
Figure 3.54: Magnitude response of the desired Hilbert transformer \( H(z) \) in Example 15.

Table 3.16: Comparison of results for the design of a Hilbert transformer in Example 15

<table>
<thead>
<tr>
<th>Method</th>
<th>Clock rate increase with respect to data rate</th>
<th>No. Multipliers / No. Adders</th>
<th>MPOS / APOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>[71]</td>
<td>–</td>
<td>127 / 211</td>
<td>127 / 211</td>
</tr>
<tr>
<td>[79]</td>
<td>–</td>
<td>48 / (not given)</td>
<td>48 / (not given)</td>
</tr>
<tr>
<td>Proposed</td>
<td>( K = 19 )</td>
<td>10 / 55</td>
<td>10 / 883</td>
</tr>
</tbody>
</table>

3.3 Conclusion

In this chapter we have introduced efficient methods to design both, narrowband and wideband filters using simple subfilters. The design methods for narrowband filtering have been specifically focused on the improvement of the magnitude response of the Recursive Running Sum (RRS) filter. The reason is that this is a very useful filter mainly employed in sampling rate conversion processes, fundamental in digital communications. The simplicity of the RRS filter makes it an attractive and effective subfilter to
form decimation filters, but its magnitude response characteristics are undesirable in many cases. Particularly, the problem of improving the magnitude characteristics of the RRS filter, keeping its computational complexity as low as possible in the context of a decimation process, has been our main objective.

We first presented two sharpened RRS-based filtering structures where the subfilters are compensated RRS filters instead of single RRS filters. The structures were based on the two simplest sharpening polynomials proposed by Kaiser and Hamming and an straightforward algorithm to select the better of the two structures in terms of computational complexity for a given stopband constraint was provided. These structures resulted in a considerable improvement in the passband due to the compensation of the RRS filter. Moreover, two interesting and apparently contradictory characteristics were observed mainly in the cases where it was desired to obtain a prescribed and high passband-stopband improvement. First, the sharpening of compensated RRS subfilters resulted in lower computational complexity than the sharpening of RRS subfilters without compensation. Second, the use of the sharpening polynomial with the higher degree resulted in lower computational complexity. These characteristics were the starting point to develop an optimization-based method to design sharpened RRS filters and sharpened-compensated RRS filters. With this method we obtained better improvement in the magnitude characteristics of RRS filters and simultaneously lower increase of computational complexity in comparison to recent methods proposed in literature.
Another important observation relies in the fact that the RRS filter has its worst-case magnitude characteristics in the low-pass region and thus the magnitude of the RRS filter only needs to be improved in this region. From this important consideration, we introduced the corrector filter as a low complexity filter optimized to improve only the worst-case magnitude characteristics of RRS filters. This proposal was shown to be especially useful for the two-stage RRS-based decimation scheme, which has been the most popular scheme to balance power consumption and area requirement in RRS-based decimation filtering in the recent years. Additionally, we proposed an efficient formulation to properly delimit the efficient use of the recently proposed Chebyshev sharpening to improve the worst-case stopband magnitude response of the RRS filter. For the passband improvement, we introduced a new compensation filter design based on the sharpening principle. We focused on the case of a second-order compensator and we have obtained a convenient trade-off between complexity and magnitude improvement. The complete approach has been shown to be a very efficient way to improve the magnitude characteristics of RRS filters.

With regard to the design of wideband filters, the efficient design of Hilbert transformers, digital filters with wide passband useful to generate analytic signals, has been the main objective. Taking as a basis the Frequency Transformation (FT) technique that allows designing a filter in terms of several identical subfilters and the Pipelining-Interleaving (PI) technique employed to implement a cascade of identical subfilters using a single time-multiplexed subfilter, we have introduced a simple method to obtain an FT-PI-based structure that employs only a single time-multiplexed subfilter. We
proposed an optimization framework to design FT-based Hilbert transformers. With this method, any objective function that includes the weighted costs of computational elements and memory elements can be minimized. The proposed optimization framework allowed us to properly formulate the extension of the FT method to a multi-level scheme where the FT is used in the subfilters. The multi-level scheme was independently developed by other authors and was presented recently in literature. However, the proposed formulation has been shown to be superior since it is based on optimization and is flexible to take into account different costs in the objective function. Finally, with our optimization-based approach and the key observation that in the FT method the subfilter has a strict transition band with wide ripple, we proposed a hybrid FRM-FT-based design method especially suitable for very sharp Hilbert transformers. This proposal has been shown to be effective in saving computational elements, with better results than other existing methods in literature.

3.4 References


Contributions on a special class of subfilters: Cyclotomic Polynomial Filters

Oh, knowledge too wonderful for me! It is so high; I cannot attain to it.

Psalms 139:6

Cyclotomic Polynomial Filters (CPFs) are a special class of subfilters widely used in the design of low-complexity FIR digital filters, particularly for low-pass and high-pass applications where either narrow or wide passbands are required. A detailed description of the CPFs is given in this chapter and an important contribution to design CPF-based filters is presented, namely, the extension of the search space of CPFs beyond of the limits used in literature. This extension converges to the theorem of preservation of unitary coefficients, the main contribution in this chapter, which enlarges the capabilities of CPFs by showing that any CPF can have a transfer function with unitary coefficients and with the lowest computational complexity.
4.1 Introduction to Cyclotomic Polynomial Filters

It has been seen in the previous chapter that one of the most successful approaches to design low-complexity FIR filters consists on using several subfilters to perform the desired filtering task. Prefilter-Equalizer ([1]-[2], [4]-[9]) and Periodic Filter-based ([10]-[18]) structures have been, to date, under continuous investigation because of their powerful advantages. It is mainly in such design schemes where filters based on Cyclotomic Polynomials (CPs) play a very important role. In most of the literature, CP-based filters have found special utility for low-pass and high-pass applications, which are closely related to each other. Therefore, in this chapter we will explore only the case of low-pass CP-based filter designs, since high-pass CP-based filters can be straightforwardly designed from low-pass filters.

In the following we give the definition and usefulness of filters constituted by CPs, namely, Cyclotomic Polynomial Filters (CPFs), along with the problem formulation to design low-pass filters based on CPFs. Section 4.2 presents the importance of using an extended search of CPFs. We present an algorithm to find low-complexity transfer functions of CPFs whose indexes are extended beyond of the upper bound imposed in literature and we detail the advantages of using these CPFs, mainly when recursive structures are used. The results presented in that section are a motivation to develop the theorem of preservation of unitary coefficients in Section 4.3, where an explicit formula, which is guaranteed to have the minimum number of arithmetic operations, is derived to find a transfer function with unitary coefficients for any CPF.
4.1.1 Definition and usefulness

A Cyclotomic Polynomial (CP) in $z^{-1}$, identified by an integer index $p$ and denoted as $C_p(z^{-1})$, is defined as a unique polynomial whose roots satisfy two conditions:

- the roots of $C_p(z^{-1})$ are a subset of the roots of $(1 - z^{-N})$, with $N \geq p$, i.e., they are a subset of the $N$ roots of unity,
- the roots of $C_p(z^{-1})$ are not in the set of the roots of $(1 - z^{-d})$, with $p \geq d$, i.e., they are not in the set of the $d$ roots of unity,

where $p$, $d$, and $N$ are integers greater than zero. For any integer $N$, the $N$ roots of unity are the roots of all the CPs whose indexes $p$ exactly divide $N$. Thus we have

$$1 - z^{-N} = \prod_{p\mid N} C_p(z^{-1}),$$  \hspace{1cm} (4.1)

where $p\mid N$ identifies the positive integers $p$, less than or equal to $N$, which divide $N$ (in other words, the remainder of the division between $N$ and $p$ is zero) [4].

A Cyclotomic Polynomial Filter (CPF) with index $p$ is a digital filter whose transfer function is equal to a CP in $z^{-1}$, i.e., $H_p(z) = C_p(z^{-1})$. The transfer function of any CPF can be obtained as follows [4], [19],

$$H_p(z) = \prod_{k(k, p) = 1}^{p} \left(1 - z^{-1}e^{-j2\pi k \frac{1}{p}}\right),$$  \hspace{1cm} (4.2)

where $k(k, p) = 1$ is used to identify the positive integers $k$ less or equal to $p$ such that $k$ and $p$ are co-prime (in other words, $k$ and $p$ do not contain any common factor) [19]. Notice that, given any integer $p$, (4.2) allows to write the transfer function of any CPF indexed by $p$.  

197
The order of a CPF is given by the totient function \( \phi(p) \), which finds the number of positive integers less than or equal to \( p \) that are co-prime to \( p \), as follows

\[
\phi(p) = \sum_{k\mid p} k \cdot \mu(p/k),
\]

where \( \mu(p/k) \) is the Möbius function defined as

\[
\mu(p/k) = \begin{cases} 
1, & p/k = 1; \\
-1^n, & p/k = p_1 \cdot p_2 \cdot \ldots \cdot p_n, \\
0, & p/k \text{ is divisible by the squares of a prime.}
\end{cases}
\]

The index \( n \) in the second entry stands for the number of distinct prime numbers that decompose the argument \( p/k \).

The transfer function of a CPF with square-free index \( p \) is [19]

\[
H_p(z) = \sum_{k=0}^{\phi(p)} h_{p,k} z^{-(\phi(p)-k)},
\]

where the coefficients \( h_{p,k} \) can be obtained with the recursive relation

\[
h_{p,k} = \frac{\mu(p)}{k} \sum_{i=0}^{k-1} h_{p,i} \cdot \mu(g(p,k-i)) \cdot \phi(g(p,k-i))
\]

using the initial value \( h_{p,0} = 1 \). The function \( g(p, k-i) \) is the greatest common divisor between \( p \) and \( k-i \). Notice that (4.6) represents an effective algorithm for automatically generating the transfer functions of CPFs with square-free indexes \( p \).

The following properties give relations between the transfer function of any CPF with index \( p \) in terms of other transfer function of a CPF with different index [4], [19]. These properties were applied in [19] to establish guidelines on the design of low-complexity CPFs. Particularly, property 5
assures that for indexes \( p > 1 \), the transfer function of a CPF has unity gain in baseband provided that \( p \neq q^k \). Thus, for other cases the transfer functions have to be normalized in order to assure unity gain in baseband. As we will see in subsequent sub-sections, these properties are not only useful to obtain the transfer function of any CPF, but they support the demonstration that any CPF can have a transfer function with unitary coefficients.

**Property 1:** if \( p \) is a prime number, we have

\[
H_p(z) = \sum_{i=0}^{p-1} z^{-i} = \frac{1-z^{-p}}{1-z^{-1}}. \tag{4.7}
\]

**Property 2:** if \( p = mn^k \), with \( m, n \) and \( k \) being integers, we have

\[
H_p(z) = H_{mn}(z) = H_{mn}(z^{n^{k-1}}). \tag{4.8}
\]

**Property 3:** if \( p = 2v \), with \( v \) being an odd integer and \( v \geq 3 \), we have

\[
H_p(z) = H_{2v}(z) = H_v(-z). \tag{4.9}
\]

**Property 4:** if \( p = q \times r \), with \( q \) being a prime number that does not divide \( r \), we have

\[
H_p(z) = H_{qr}(z) = \frac{H_q(z^q)}{H_r(z)}. \tag{4.10}
\]

**Property 5:** for \( z = 1 \), we have

\[
H_p(z) = \begin{cases} 
0, & q = 1 \\
q, & p = q^k, q \text{ prime} \\
1, & \text{otherwise}. 
\end{cases} \tag{4.11}
\]

The preliminary work in using CPs for the transfer functions of digital filters was presented in [1], where the utility of using cascaded CPFs as standalone, multiplierless, linear phase filters was described. Among the most favorable characteristics of CPFs we have:
• CPFs are symmetric Finite Impulse Response (FIR) filters, so they have desirable characteristics such as linear-phase and guaranteed stability.
• CPFs are multiplier-free, thus they can be used to design low-complexity multiplierless filters.
• CPFs have all zeros on the unit circle of the complex z-plane, which can help to attain the desired stopband and transition band specifications.
• The zeros of each CPF are distinct resulting in unique stopband performance for different designs.
• CPFs can be implemented in recursive form with perfect pole-zero cancellation, thus yielding important savings on the number of required additions whereas preserving FIR filter characteristics.

It was noted in [2]-[3] that CPFs could be used for efficient prefiltering in a Prefilter-Equalizer cascade. A trial-and-error visual search over a field of 24 eligible CPFs was given as design method, which yielded acceptable, although suboptimal, prefilters. To design more efficient prefilters, in [4]-[5] was proposed an automated method where the first 104 CPFs, i.e., $1 \leq p \leq 104$, were used as a search space. That research showed that, since CPFs are unique, a great variety of solutions is available for diverse applications, improving the search space previously constrained to only Recursive Running Sum (RRS) filters and 24 CPFs. In [6]-[7], an effective and straightforward constrained linear optimization problem formulation was given to design CPF-based filters, which can be solved with Integer Linear Programming (ILP). Then, the Prefilter-Equalizer approach was improved in [8] by using Interpolated Second Order Polynomials (ISOP) for FIR filters and
Inverse Interpolated First Order Polynomials (IIFOP) for Infinite Impulse Response (IIR) filters.

CPF have been also used for Periodical Filter-based schemes mainly to simplify the masking subfilters, such as in the Interpolated Finite Impulse Response (IFIR) technique [12] and in the Frequency-Response Masking (FRM) technique [13]. Additionally, CPFs have found useful applications in multirate systems. As an example, a Residue Number System (RNS)-based implementation of an eight channel cochlea filter bank was presented in [18], whereas the optimization framework for multiplierless design of FIR filters for decimation based on CPFs was proposed in [19]-[20].

4.1.2 Problem formulation to design CPF-based filters

A CPF-based filter is designed using a cascade of subfilters. Each subfilter is formed by cascading several CPFs with the same index and each index is different of the one of any other CPF used in other subfilter. Thus, the transfer function of the CPF-based filter is given as

$$H(z) = \prod_{i=1}^{\vert S \vert} H_{p_i}^m(z),$$

with

$$p_1, p_2, ..., p_{\vert S \vert} \in S, \text{ and } p_1 < p_2 < ... < p_{\vert S \vert},$$

where $S$ is the set of eligible CPF’s indexes, $\vert S \vert$ is the cardinality of $S$ (i.e., the number of elements in $S$), whereas $p_i$, with $i = 1, 2, ..., \vert S \vert$, is the index of the $p_i$-th CPF, $H_{p_i}^m(z)$. Each integer $m_i$ is the number of times that the $p_i$-th CPF must be cascaded. If any $m_i$ is equal to zero, then the corresponding $p_i$-th CPF is not used in the cascaded interconnection of CPFs. Figure 4.1 depicts a
A general scheme of the architecture of a CPF-based filter whose transfer function is (4.12).

To design CPF-based filters an effective formulation, introduced in [7] and specially adapted for decimation filters in [19] and [20], consists on applying logarithms to the frequency response of the CPFs such that the filter design problem can be expressed as an Integer Linear Programming (ILP) optimization problem. A slightly modified version of the CPF-based filter design problem formulation given in [19] is presented in two parts as follows:

1. Find the values \( p_i \), with \( i = 1, 2, \ldots, |S| \), that must be included in the set \( S \). These values are the indexes of eligible CPFs.

2. Find the optimal values \( m_i \), with \( i = 1, 2, \ldots, |S| \), such that the filter \( H(z) \) has the minimum computational complexity whereas it satisfies specific magnitude constraints in passband and stopband regions.

The first part of the problem is solved by designing all the CPFs using the first \( B \) indexes, with \( B = 104 \) as an upper limit set in literature. Starting with the first CPF, the magnitude response of each CPF is evaluated in passband and in stopband regions. If the CPF does not have zeros in its passband(s) and simultaneously has an acceptable percentage of zeros in its stopband(s), the index of such CPF is included in the set \( S \).
The second part of the problem, specially adapted for low-pass designs, is formally given as follows:

\[
\begin{align*}
\text{minimize } & F(\mathbf{m}) \\
\text{subject to: } & \mathbf{A} \mathbf{m} \leq \mathbf{b} \\
& \mathbf{m}_i \geq 0 \text{ and } \mathbf{m}_i \text{ integer } \quad \forall i, \quad i = 1, 2, ..., |S|,
\end{align*}
\]

with

\[
F(\mathbf{m}) = [c_1 \ c_2 \ ... \ c_{|S|}] \cdot \mathbf{m},
\]

\[
c_i = a_i + \gamma d_i, \quad i = 1, 2, ..., |S|,
\]

\[
\mathbf{m} = [m_1 \ m_2 \ ... \ m_{|S|}]^T,
\]

\[
\mathbf{A} = \begin{bmatrix}
H_{p_1,1} & \cdots & H_{p_1,|S|} \\
\vdots & \ddots & \vdots \\
H_{p_{1+N_{\text{stop}}},1} & \cdots & H_{p_{1+N_{\text{stop}}},|S|}
\end{bmatrix},
\]

\[
H_{p_i,j} = \begin{cases} 
-\min \left\{ 20 \log_{10} \left( |H_{p_i}(\omega_j)| \right) \right\} & \forall i = 1, 2, ..., |S| \text{ with } j = 1, \\
\max \left\{ 20 \log_{10} \left( |H_{p_i}(\omega_j)| \right) \right\} & \forall i = 1, 2, ..., |S| \text{ with } j = 2, ..., 1 + N_{\text{stop}},
\end{cases}
\]

\[
\mathbf{b} = [R_1 \ -A_1 \ -A_2 \ ... \ -A_{N_{\text{stop}}}]^T.
\]

In (4.14), \([\mathbf{m}]_i\) stands for the \(i\)-th element of \(\mathbf{m}\). The value \(c_i\) in (4.15), with \(i = 1, 2, ..., |S|\), is the cost associated to the \(p\)-th CPF. This cost is expressed in (4.16), with \(a_i\) and \(d_i\) being respectively the number of additions and delays required in the \(p\)-th CPF. The constant \(\gamma \in [0, 1]\) depends on the relative complexity of delays with respect to adders and, for cases where the computational complexity is of primary importance, \(\gamma = 0\) is recommended [19].

In (4.18) to (4.20) \(N_{\text{stop}}\) represents the number of stopbands in a low-pass design with multiple stop-band regions. In (4.19) \(|H_{p_i}(\omega)|\) is the magnitude...
response of the $p$-th CPF, whereas $\bar{\omega}_j$ represents either the passband region if $j = 1$ or a stopband region if $j = 2, 3, \ldots, 1+N_{\text{stop}}$. Thus, $H_{p,j}$ is either the worst-case passband droop of the $p$-th CPF in passband if $j = 1$ or the worst-case stopband attenuation of the $p$-th CPF in the $(j-1)$-th stopband if $j = 2, 3, \ldots, 1+N_{\text{stop}}$. Finally, in (4.20) the value $R_1$ is the maximum allowed ripple in passband (in dB) and the value $A_i$, with $i = 1, 2, \ldots, N_{\text{stop}}$, is the minimum allowed attenuation in the $i$-th stopband (in dB).

### 4.2 The advantages of using an extended search

A key characteristic of all the extensive research developed for CPFs is that the authors have considered a search space comprising only the CPFs with indexes $p$ up to 104. The reason is simple: the transfer functions of the first 104 CPFs have unitary coefficients, which considerably simplify the computational complexity of the filter. For the current practical applications, such search space has been shown to be adequate. However, the subtle disadvantage of a constrained search space which includes only the first 104 CPFs still remains.

The recent advances in DSP technology facilitate the trends of future communication systems where the higher rates are a necessity. With more and more users desiring to share communication channels, the importance of clever exploitation of the bandwidth becomes paramount [21]-[22]. One of the implications of such evolution is that filters with narrower passband and a very high selectivity are required. Additionally, the most recent approaches for decreasing the computational complexity in digital filters with arbitrary bandwidth have combined the Prefilter-Equalizer and the Periodic Filter
schemes [17]. This combination can take advantage of low-complexity multiplierless filters with very narrow passbands.

Figure 4.2 shows the order of a CPF plotted against the index $p$ for $1 \leq p \leq 1000$. Since the order tends to increase as $p$ increases, it can be inferred that a CPF with an increasing index $p$ will have a narrower passband because more roots are accumulated in the unit circle. Therefore, CPFs with indexes higher than 104 would be a convenient option to construct narrowband multiplierless filters provided that their low complexity is still available. Thus, the needs highlighted in the preceding paragraph along with the previous observations are a motivation to study CPFs whose indexes go beyond of $p = 104$.

To prove the usefulness of CPFs with indexes higher than $p = 104$ we solve the CPF-based filter design problem over an extended search space with indexes $p \in \{1, 2, \ldots, B\}$, with the upper bound set as $B = 200$. We have chosen designing CPF-based multiband lowpass filters whose main application is on decimation filtering and as masking filter for Periodic Filter-based structures.
Particularly, the problem of decimation filtering has been recently studied in [19] and effective results were obtained with the usual upper bound $B = 104$.

### 4.2.1 Proposed systematic solution procedure

We solved the CPF-based filter design problem by means of the following systematic procedure:

- **Step 1**: obtain the cost of all CPFs in the initial search space.
- **Step 2**: obtain the set $S$ with the indexes of eligible CPFs from the initial search space.
- **Step 3**: obtain the worst-case magnitude characteristics of each eligible CPF in each one of the bands of interest.
- **Step 4**: solve the optimization problem given in (4.14).

Now let us analyze every step in detail.

**Step 1: obtain the cost of all CPFs**

The cost of each CPF can be estimated directly from its transfer function. For the case of computational complexity, the number of additions in the transfer function is the cost measure, provided that the transfer function has unitary coefficients. If not, the complexity of the coefficients must be considered. In the same way as [19] and [20], we consider a cost dominated by the number of additions, i.e., $\gamma = 0$ in (4.16) for every CPF with index $p$, with $p \in \{1, 2, \ldots, 200\}$.

Since the computational complexity is of primary interest, a transfer function with few additions must be chosen for every CPF. It has been considered in [4], [5], [18]-[20] that recursive structures are preferable because
they may have less addition operations than the non-recursive ones. Therefore, special attention must be paid to these structures. The following observations on the transfer functions of CPFs can be made:

- For a CPF with arbitrary index \( p \), either a recursive or a non-recursive transfer function can be obtained from either the recursive or the non-recursive transfer function of other CPF with smaller index by using Properties 2 and 3, presented in sub-section 4.1.1, if they can be applied to the index \( p \).
- If Property 1 can be used for a given \( p \), both recursive and non-recursive transfer functions can be easily created.
- If Property 4 can be used for a given \( p \), a recursive transfer function can be created.
- If the index of the CPF is square-free, expressions (4.5) and (4.6) can be used to obtain the non-recursive transfer function.

Note that these observations allow obtaining transfer functions for a given CPF in terms of previously obtained transfer functions by using Properties 1 to 4 whenever they can be applied. Thus, the direct computation of transfer functions for all the CPFs using (4.2) can be avoided. Moreover, only through these observations it is possible getting recursive transfer functions.

Obtaining the cost of all the eligible CPFs with indexes up to 104 has been a straightforward task in previous works on CPFs in literature because these CPFs have always unitary coefficients in their transfer functions. However, it is a well-known fact that non-recursive CPFs with indexes higher than 104 lose this property [4]-[8], [18]-[20]. Moreover, for many CPFs there may be several options to represent the recursive transfer function. Therefore, a
problem encountered in the calculation of the recursive transfer functions consists on the choice of the transfer function with a low number of additions (a low computational cost).

In order to solve the aforementioned problem an efficient algorithm to select a recursive transfer function with few additions for every CPF with index \( p \) (where \( p \in \{1, 2, ..., 200\} \)) is proposed. It is worth highlighting that the algorithm can be also used to obtain non-recursive transfer functions with less effort. This algorithm is based on the previous observations and it takes advantage on the Properties 1 to 4 to express the desired transfer function in terms of the transfer function of another CPF whose index is a reference index. Starting with an arbitrary index \( p \), the reference index is reduced in every step, so that it is guaranteed that the transfer function of the CPF with this index has been calculated previously. This transfer function is called reference transfer function.

Low-complexity transfer functions for the first 60 CPFs were published in [19] and for the first 104 in the internal report [23]. These transfer functions can be used as an initial database. Thus in every step, if the reference transfer function can be found in the initial database, the algorithm finds directly the desired transfer function in terms of the reference transfer function, assuring a low-complexity expression. The cost (in this case, number of additions) of the desired transfer function is stored in a Look-Up table (LUT) and the algorithm continues with the calculation of the transfer function of the CPF with the next index. As a consequence, the first index to be solved is \( p = 105 \).

The proposed algorithm, depicted in the flowchart of Figure 4.3, consists in the following steps:
1. Load the initial database. Set $i = 1$ and $P_i = 1$. Assign $\tilde{p}_i = p$ (with the initial $p = 105$) and find the prime factors of $\tilde{p}_i$.

2. If $\tilde{p}_i$ is prime, use Property $P_i$ to obtain the recursive and non-recursive transfer functions directly. Include $H_{\hat{p}_i}(z)$ in the database and store the corresponding cost in the LUT. Go to step 9. Otherwise, continue with step 3.

3. If $\tilde{p}_i$ can be factorized as $mn^k$ with $k > 1$, set $r_i = mn$, $P_i = 2$ and continue with step 4. Otherwise, go to step 5.

4. If $r_i \leq 104$, go to step 8. Otherwise, set $i = i+1$. Assign $\tilde{p}_i = r_{i-1}$ and return to step 3.

5. If one of the prime factors of $\tilde{p}_i$ is 2, set $r_i = \tilde{p}_i/2$, $P_i = 3$, and continue with step 6. Otherwise go to step 7.

6. If $r_i \leq 104$, go to step 8. Otherwise, set $i = i+1$. Assign $\tilde{p}_i = r_{i-1}$.

7. To obtain the non-recursive transfer function, use equations (4.5) and (4.6). To obtain the recursive transfer function, if $\tilde{p}_i$ is not prime, set $r_i = \tilde{p}_i/q$ with $q$ being the largest prime factor of $\tilde{p}_i$ and set $P_i = 4$. Otherwise set $P_i = 1$.

8. For $j = i, i-1, i-2, \ldots, 2, 1$, use Property $P_j$ to express $H_{\tilde{p}_j}(z)$ in terms of $H_{\eta_j}(z)$, taking $H_{\eta_j}(z)$ from the database, and include $H_{\tilde{p}_j}(z)$ in the database. Store the corresponding cost in the LUT.

9. If $p < B$ (we arbitrarily use an upper bound $B=200$), set $i = 1$ and $P_i = 1$. Assign $\tilde{p}_i = p+1$, find the prime factors of $\tilde{p}_i$ and return to step 2. Otherwise, stop.
Figure 4.3: Flowchart of the algorithm to find non-recursive and low-complexity recursive transfer functions for CPFs with indexes $p$ up to 200.
The following examples illustrate the previous procedure.

**Example 1:** find the transfer functions for the CPF with index \( p = 105 \) as well as the cost for the recursive transfer function.

*Step 1:* the prime factorization of constants less than \( 2^{32} \) can be carried out with the MATLAB instruction `factor`. By using this function we have \( \tilde{p}_1 = 3 \times 5 \times 7 = 105 \).

*Step 2:* since 105 is not prime we go to step 3.

*Step 3:* the factorization in the form \( mn^k \) is not possible. We go to step 5.

*Step 5:* none of the prime factors is 2. We go to step 7.

*Step 7:* to obtain the non-recursive transfer function, we use equations (4.5) and (4.6). We have

\[
H_{105}(z) = 1 + z^{-1} + z^{-2} + z^{-5} + z^{-6} - 2z^{-7} - z^{-8} - z^{-9} + z^{-12} + z^{-13} + z^{-14} + z^{-15} + z^{-16} + z^{-17} - z^{-20} \\
- z^{-22} - z^{-24} - z^{-26} - z^{-28} + z^{-31} + z^{-32} + z^{-33} + z^{-34} + z^{-35} + z^{-36} - z^{-39} - z^{-40} - 2z^{-41} \\
- z^{-42} + z^{-43} + z^{-46} + z^{-47} + z^{-48} \text{ (non-recursive form)}.
\]

To obtain the recursive transfer function we set \( r_1 = (3 \times 5 \times 7)/7 = 15 \) and \( P_1 = 4 \) because 105 is not prime.

*Step 8:* applying Property \( P_1 = 4 \) to express \( H_{\tilde{p}_1}(z) = H_{105}(z) \) in terms of \( H_{\hat{n}}(z) = H_{15}(z) \) results in \( H_{105}(z) = H_{15}(z^7)/H_{15}(z) \). From the database (see [19]) we have

\[
H_{15}(z) = 1 - z^{-1} + z^{-3} - z^{-4} + z^{-5} - z^{-7} + z^{-8} \text{ (non-recursive form)},
\]

and therefore we obtain

\[
H_{105}(z) = (1 - z^{-7} + z^{-21} - z^{-28} + z^{-35} - z^{-49} + z^{-56})/(1 - z^{-1} + z^{-3} - z^{-4} + z^{-5} - z^{-7} + z^{-8}) \text{ (recursive form)}.
\]

The cost of the recursive transfer function is 12 adders.
Example 2: find the transfer functions for the CPF with index $p = 175$ as well as the cost for the recursive transfer function.

*Step 1:* the prime factorization is $\tilde{p}_1 = 5 \times 5 \times 7 = 175$.

*Step 2:* since 175 is not prime we go to step 3.

*Step 3:* the factorization in the form $mn^k$ is $\tilde{p}_1 = 5^2 \times 7$. We set $r_1 = 5 \times 7 = 35$ and $P_1 = 2$.

*Step 4:* since $r_1 < 104$, we go to step 8.

*Step 8:* applying Property $P_1 = 2$ to express $H_{\tilde{p}_1}(z) = H_{175}(z)$ in terms of $H_{r_1}(z) = H_{35}(z)$ results in $H_{175}(z) = H_{35}(z^5)$. From the database (see [19]) we have

$$H_{35}(z) = 1 - z^{-1} + z^{-5} - z^{-6} + z^{-7} - z^{-8} + z^{-10} + z^{-11} - z^{-12} + z^{-13} + z^{-14} + z^{-16} + z^{-17} - z^{-18} + z^{-19} - z^{-23} + z^{-24} \text{ (non-recursive form)},$$

$$H_{35}(z) = (1 - z^{-1} - z^{-35} + z^{-36}) / (1 - z^{-5} - z^{-7} + z^{-12}) \text{ (recursive form)},$$

and we obtain

$$H_{175}(z) = 1 - z^{-5} + z^{-25} - z^{-30} + z^{-35} - z^{-40} + z^{-50} - z^{-55} + z^{-60} - z^{-65} + z^{-70} - z^{-80} + z^{-85} + z^{-90} + z^{-95} - z^{-115} + z^{-120} \text{ (non-recursive form)},$$

$$H_{35}(z) = (1 - z^{-5} - z^{-175} + z^{-180}) / (1 - z^{-25} - z^{-37} + z^{-60}) \text{ (recursive form)}.$$

The cost of the recursive transfer function is 6 adders.

After running the proposed algorithm, an updated database with the non-recursive and the recursive transfer functions of the CPFs with indexes up to $B = 200$ is easily obtained, as well as a LUT containing the cost of every CPF. Clearly, if a wider search space is wanted, i.e., with a greater upper bound $B$, the algorithm can be applied straightforwardly.

Two important characteristics have been observed from the updated set of transfer functions:
1) Non-recursive transfer functions have unitary coefficients, except for the ones of CPFs with indexes \( p=105, p=165, \) and \( p=195 \) that have coefficients in the set \( \{-2, -1, 0, 1, 2\} \).

2) Recursive transfer functions have unitary coefficients.

Therefore, the cost of every CPF can be directly calculated as the number of addition and subtraction operations in the corresponding transfer function. As it was previously mentioned, recursive transfer functions are preferred because they have less arithmetic operations than their non-recursive counterparts. Hence, the LUT of costs contains the number of additions of the recursive transfer functions.

**Step 2: obtain the set \( S \) with the indexes of eligible CPFs**

An index is eligible and can be included in \( S \) if its corresponding CPF accomplishes the following two conditions:

1) The CPF does not have zeros in the passband region.

2) The CPF has at least \( b\% \) of its zeros inside the stopband regions.

We use \( 5\leq b\leq 20 \) as an arbitrary election to get a high number of eligible indexes.

The two aforementioned conditions can be straightforwardly evaluated for every CPF from the roots of its corresponding transfer function. The positions of the zeros of the frequency response over the relative frequency axis are the angles of the roots. From (4.2), it can be seen that in the frequency range from 0 to \( \pi \) the zeros of a CPF whose index is \( p \) are placed over the frequencies \( \omega_i = 2\pi k_i/p, \) for all \( k_i \) in the set of integers coprime to \( p, \) \( \{k_1, k_2, \ldots, k_{\varphi(p)}\}, \) that are
less or equal to \( p/2 \). Conditions 1 and 2 are easily evaluated by simple comparison of the position of the roots with respect to the bands of interest.

A different way of obtaining the position of the zeros of the magnitude response, instead of obtaining the values \( k_i \), is using the MATLAB instruction \texttt{roots} to easily compute the roots of the non-recursive transfer functions generated with the algorithm introduced in the previous step. The positions of the zeros along the relative frequency axis can be calculated by using the MATLAB instruction \texttt{angle}.

Figure 4.4 shows a simple example of an eligible CPF and a non-eligible CPF, considering \( b = 20 \), to design a lowpass filter whose passband is \( \omega_1 = [0, 0.01 \pi] \) and whose 12 stopbands are \( \omega_k = [0.08(k-1)\pi - 0.01\pi, 0.08(k-1)\pi + 0.01\pi] \) with \( k = 2, 3, \ldots, 13 \). This is the first decimation filter in a two-stage decimation chain with \( M = 25 \) as first decimation factor and \( v = 4 \) as residual decimation factor. The CPF with index \( p = 15 \) (Figure 4.4a) is not eligible because it does not have enough zeros on the stopband zones. On the other hand, the CPF with index \( p = 25 \) (Figure 4.4b) is eligible because all of its zeros are on the stopband zones and it does not have zeros over the passband region.

\textbf{Step 3: obtain the worst-case magnitude characteristics of eligible CPFs}

The worst-case magnitude value \( H_{p_i,j} \) in the \( j \)-th band of interest for the \( i \)-th eligible CPF (whose index is \( p_i \)) is obtained directly from (4.19). Determining \( H_{p_i,j} \) requires knowing the magnitude response of every CPF in every band of interest. The magnitude response can be obtained from the transfer functions calculated in \textit{Step 1}, only substituting \( e^{-j\omega} \) instead of \( z^{-1} \). A
simple approach to evaluate (4.19) over the bands $\tilde{\omega}_j$ consists on splitting these bands in a grid of $K$ frequency values $\omega_{j,k}$, with $k = 1, 2, \ldots, K$, setting the leftmost edge of $\tilde{\omega}_j$ equal to $\omega_{j,1}$ and the rightmost edge of $\tilde{\omega}_j$ equal to $\omega_{j,K}$. Thus, $\tilde{\omega}_j$ becomes a set of values $\tilde{\omega}_j=\{\omega_{j,1}, \omega_{j,2}, \ldots, \omega_{j,K}\}$. The value $K$ does not need to be too high. A moderate value, $K = 50$ for example, is enough.

Figure 4.4: Magnitude responses of two CPFs that are possible candidates to be constituent subfilters of a multiband lowpass CPF-based filter with 12 stopbands; (a) CPF with index $p = 15$ is non-eligible, (b) CPF with index $p = 25$ is eligible.
As a simple example, let us consider again the desired CPF-based decimation filter mentioned in the previous step. For that filter we have $N_{\text{step}} = 12$, and the bands of interest are:

$\bar{\omega}_1 = [0, 0.01\pi]$ (passband),
$\bar{\omega}_2 = [0.07\pi, 0.09\pi]$ (stopband),
$\bar{\omega}_3 = [0.15\pi, 0.17\pi]$ (stopband),
$\bar{\omega}_4 = [0.23\pi, 0.25\pi]$ (stopband),
$\bar{\omega}_5 = [0.31\pi, 0.33\pi]$ (stopband),
$\bar{\omega}_6 = [0.39\pi, 0.41\pi]$ (stopband),
$\bar{\omega}_7 = [0.47\pi, 0.49\pi]$ (stopband),
$\bar{\omega}_8 = [0.55\pi, 0.57\pi]$ (stopband),
$\bar{\omega}_9 = [0.63\pi, 0.65\pi]$ (stopband),
$\bar{\omega}_{10} = [0.71\pi, 0.73\pi]$ (stopband),
$\bar{\omega}_{11} = [0.79\pi, 0.81\pi]$ (stopband),
$\bar{\omega}_{12} = [0.87\pi, 0.89\pi]$ (stopband),
$\bar{\omega}_{13} = [0.95\pi, 0.97\pi]$ (stopband).

Additionally, let us consider that 25 is the 14-th eligible index in the set $S$, i.e., $p_{14} = 25$.

After sampling every band and obtaining a set of 50 equally spaced frequency points for one of them, the following worst-case values have been obtained:

$H_{14,1} = 0.2155$,  $H_{14,2} = -16.6843$,  $H_{14,3} = -21.6349$,
$H_{14,4} = -21.6349$,  $H_{14,5} = -16.6843$,  $H_{14,6} = 0$,
$H_{14,7} = -16.6843$,  $H_{14,8} = -21.6349$,  $H_{14,9} = -21.6349$,
$H_{14,10} = -16.6843$,  $H_{14,11} = 0$,  $H_{14,12} = -16.6843$,  $H_{14,13} = -21.6349$.

These values are depicted in Figure 4.5.

**Step 4: solve the optimization problem**

Note from (4.14) that in order to solve the CPF-based filter design problem, the objective function, the matrix $A$ and the vector $b$ must be defined. Once the previous steps have been accomplished, forming the objective and
constraint functions of the optimization problem (4.14) becomes a direct process. Specifically, in Step 1 the costs were calculated for every CPF and with Step 2 the eligible CPFs were selected. Therefore, the cost function (described in equations (4.15) and (4.16)) can be formed directly by taking the corresponding costs obtained in Step 1 for every of the eligible CPFs. The matrix A is formed in Step 3, using eq. (4.19).

![Graph](image-url)

**Figure 4.5:** Worst-case magnitude response values in every band of interest of the CPF with index \(p_{14}=25\).

On the other hand, the vector \(b\) depends on the magnitude characteristics of the desired filter, which must be given in dB (see (4.20)). The passband specification is the maximum allowed deviation from 0dB, whereas the stopband specifications are the maximum gain values allowed in every stopband region and they must be negative values in dB. After completing the entries of vector \(b\) with the required specifications, the ILP optimization problem can be solved by using any ILP solver package. In this work the MATLAB routine IP1, based on the MATLAB function `linprog` (which
corresponds to the Optimization Toolbox), has been used. The routine IP1 is available online [24].

4.2.2 Design examples and preliminary observations

In the following paragraphs, two design examples are realized to show both, the effectiveness of the CPFs as low-complexity building blocks and the advantages of extending the search space up to \( B = 200 \) CPFs.

**Example 3:** consider the design of a CPF-based filter with the following specification,

- passband, \( \bar{\omega}_1 = [0, 0.0134\pi] \),
- stopbands, \( \bar{\omega}_k = [0.08(k-1)\pi - 0.0134\pi, 0.08(k-1)\pi + 0.0134\pi] \) with \( k = 2, 3, \ldots, 13 \),
- passband ripple, \( R_1 = 1.5 \) dB,
- minimum attenuation in the \( k \)-th stopband, \( A_k = 50 \) dB.

From the proposed procedure we know that *Step 1* generates a database containing the transfer functions and a LUT with the costs for all the CPFs used as initial search space (i.e., the first \( B = 200 \) CPFs). Therefore, it must be clear that *Step 1* should be run only once. The required costs to form the objective function (4.15) can be taken from the LUT regardless of the problem to be solved. For *Step 2*, we choose \( b\% = 5\% \) to select eligible CPFs. After the selection procedure, 138 eligible indexes are found, as shown in Table 4.1. Finally, matrix \( A \) and \( b \) are formed with equations (4.18) to (4.20) and the ILP optimization problem is solved with the
MATLAB routine IP1, according to Step 3 and Step 4. The obtained results after the optimization are:

\[ m_{16} = 1, m_{22} = 1, m_{24} = 1, m_k = 0 \text{ for other values of } k. \]

Therefore, the transfer function of the resulting CPF-based filter is:

\[
H(z) = H_{p_{16}}^{m_{16}}(z) \cdot H_{p_{22}}^{m_{22}}(z) \cdot H_{p_{24}}^{m_{24}}(z) = H_{23}(z) \cdot H_{29}(z) \cdot H_{31}(z)
\]

\[
= \left[ \frac{1 - z^{-23}}{1 - z^{-1}} \right] \cdot \left[ 1 - z^{-29} \right] \cdot \left[ \frac{1 - z^{-31}}{1 - z^{-1}} \right].
\]

Figure 4.6 presents the magnitude characteristic of \( H(z) \).

Table 4.2 shows the resulting cost of the CPF-based filter designed in Example 3 in comparison to other schemes. Clearly, the CPF-based filter has a higher advantage than [25] because of its multiplierless characteristic, whereas a 57\% of reduction in the number of adders is gained with respect to [26]. It is worth highlighting that a simple Recursive Running Sum (RRS) filter does not accomplish the required characteristics, regardless of the number of stages used.

![Magnitude response of the CPF-based filter designed in Example 3.](image)
Table 4.1: Eligible indexes of CPFs in Example 3

<table>
<thead>
<tr>
<th>Filter design</th>
<th>No. of Additions</th>
<th>No. of Multiplications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method [25]</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>Method [26]</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>CPF-based filter</td>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>

Example 4: consider the design of a CPF-based filter with the following specification,

- passband, $\tilde{\omega}_i = [0, 0.00202\pi],$
- stopbands, $\tilde{\omega}_k = [0.03636(k-1)\pi-0.00202\pi, 0.03636(k-1)\pi+0.00202\pi]$ with $k = 2, 3, \ldots, 28,$
- passband ripple, $R_1 = 0.5$ dB,
- minimum attenuation in the $k$-th stopband, $A_k = 102$ dB.

For Step 2, we choose $b\% = 5\%$ to select eligible CPFs and 156 eligible indexes are found (see Table 4.3). The obtained results after the optimization are:

$m_{27} = 1, m_{33} = 1, m_{73} = 2, m_k = 0$ for other values of $k$.

Therefore, the transfer function of the resulting CPF-based filter is:

$$H(z) = H_{p_{27}}^{m_{27}}(z) \cdot H_{p_{33}}^{m_{33}}(z) \cdot H_{p_{73}}^{m_{73}}(z) = H_{53}(z) \cdot H_{59}(z) \cdot H_{109}(z)$$

$$= \left[ \frac{1-z^{-53}}{1-z^{-1}} \right] \cdot \left[ \frac{1-z^{-59}}{1-z^{-1}} \right] \cdot \left[ \frac{1-z^{-109}}{1-z^{-1}} \right]^2.$$

Figure 4.7 presents the magnitude characteristic of $H(z)$.

Example 4 has been also solved for the traditional initial search space, i.e., considering $B = 104$ (method [19]). In this case, the search space consists of the CPFs with indexes from $p_1$ to $p_{68}$ (see Table 4.3). The resulting values after the
optimization are $m_{22} = 1$, $m_{27} = 4$, $m_{37} = 1$, $m_k = 0$ for other values of $k$, and the resulting transfer function requires 11 adders. Table 4.4 shows the cost for every solution. Note that extending the search space has a reduction of 28% in the arithmetic complexity.

<table>
<thead>
<tr>
<th>Filter design</th>
<th>No. of Additions</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPF-based filter from [19]</td>
<td>11</td>
</tr>
<tr>
<td>CPF-based filter with extended search space</td>
<td>8</td>
</tr>
</tbody>
</table>
It is worth pointing out the following comments:

- As expected, CPFs with indexes from 105 to 200 are useful basic building blocks for cases where the passband of the CPF-based filter has a narrow bandwidth. Example 4 shows a simple case like this.

- For all CPFs with indexes up to 200, non-recursive transfer functions have unitary coefficients, except the ones of CPFs with indexes $p=105$, $p=165$, and $p=195$ that have coefficients in the set $\{-2, -1, 0, 1, 2\}$. On the other hand, all recursive transfer functions have unitary coefficients. Note that there is not additional complexity due to coefficients. Therefore, the simplicity of CPFs is preserved.

- The unitary coefficients in recursive transfer functions is a remarkable advantage since one of the most important characteristic of CPFs is their ability to be represented as recursive structures with less computational complexity.

- As shown in literature for the first 104 CPFs, CPFs with indexes from 105 to 200 extend the variety of solutions in comparison to RRS filters, especially for designing single-rate FIR filters.

- The computational complexity of recursive CPFs is not directly proportional to their corresponding indexes. As an example, applying the algorithm proposed in sub-section 4.2.1 to find the number of additions of CPFs with indexes 105 and 200, we get respectively 12 and 2. Therefore, the limitation of the upper bound $B$ for the search space should not be justifiable by the usual assumption in literature, that the first 104 CPFs have the lowest complexity.
4.3 Theorem of preservation of unitary coefficients

An interesting characteristic of CPFs, observed from the extension of the search space, is that the recursive transfer functions of CPFs with indexes up to 200 have unitary coefficients. Thus, these filters preserve their simplicity and it is possible to take advantage of them to design low-complexity filters. However, a natural question is: What is the lowest index $p$ whose recursive transfer function no longer has unitary coefficients? Moreover, the number of additions of a recursive CPF is not directly proportional to the corresponding index $p$ of such CPF. Does it depend on $p$ in some way?

A closer look to the algorithm proposed in sub-section 4.2.1 to obtain the cost of CPFs, as well as the observations highlighted in the description of Step 1 in the same sub-section, reveals two important characteristics that give the basis to answer the aforementioned questions. These characteristics are:

- The transfer functions of CPFs with prime indexes have always unitary coefficients. This follows directly from Property 1. The recursive transfer functions in these cases deserve special attention because they require only two arithmetic operations.
- The strategy of expressing the transfer function of a CPF in terms of a reference transfer function (a strategy used to simplify the search of recursive transfer functions) implies that the desired transfer function has a complexity strongly related to that of the reference transfer function. This relation can be obtained from Properties 2, 3 and 4.

The questions in the first paragraph of this section are answered with our theorem of preservation of unitary coefficients, whose demonstration connects these two aforementioned characteristics.
4.3.1 Theorem and proof

The theorem of preservation of unitary coefficients states:

For any CPF there exists a transfer function with unitary coefficients, whose number of additions is the minimum and is given by $2^n$. This transfer function is a product of basic recursive transfer functions that must not be expanded, and is given by

$$H_p(z) = \begin{cases} 
1 - z^{-1}; & \text{if } p = 1, \\
1 + z^{2^{k-1}}; & \text{if } p = 2^k; k \in \mathbb{N}, \\
1 + z^{-3^{k-1}} + z^{-2 \times 3^{k-1}}; & \text{if } p = 3^k; k \in \mathbb{N}, \\
\frac{1 - z^{-p}}{1 - z^{-q^{m-1}}}; & \text{if } p = q^k; q \text{ prime and } q > 3; k \in \mathbb{N}, \\
\frac{1 + z^{-2^{k-1} q^l}}{1 + z^{-2^{k-1} q^l}}; & \text{if } p = 2^{l} q^k; q \text{ prime and } q > 3; k, l \in \mathbb{N}, \\
H_{q_n}^{(-1)^{n-1}} \left( (-1)^{1 - \text{mod}(p, 2)} x \right) \prod_{i=1}^{n-1} \prod_{j=1}^{C(n-1,i)} H_{q_n}^{(-1)^{n-1}} \left( ((-1)^{1 - \text{mod}(p, 2)} x) \prod_{k=1}^{M_{n-1}^{(n-1)}(q_n, i, j)} \right) ; & \text{otherwise},
\end{cases}$$

(4.21)

where:

- $\mathbb{N}$ is the set of natural numbers,
- $p$ is the index of a CPF,
- $m$ is the number of distinct prime factors in $p$,
- $n$ is the number of distinct odd prime factors in $p$,
- $q_1, q_2, \ldots, q_m$ are the distinct prime factors of $p$. For $i = 1, 2, \ldots, n-1$, $q_i > q_{i+1}$. Thus $q_n$ is the smallest odd prime in $p$. Note that, if $m = n + 1$, $q_n = 2$.
- $k$, for $i = 1, 2, \ldots, m$, is the number of times that the prime factor $q_i$ is repeated in $p$.
- $M_{k=1}^{n-1} (q_i, i, j)$ stands for the $j$-th product in the set of possible products formed using $i$ factors $q_n$, with $k$ ranging from 1 to $n-1$.
- $C(n-1,i)$ is the binomial coefficient.
- $\text{mod}(p, 2)$ returns 1 if $p$ is an odd number or 0 if $p$ is an even number.
The corresponding proof can be developed with the aid of Properties 1 to 4. Three steps are used to demonstrate the theorem:

**Step 1:** Demonstration of the preservation of unitary coefficients.

**Step 2:** Derivation of an explicit formula to express the transfer function with unitary coefficients of any CPF.

**Step 3:** Demonstration of the minimum number of additions in the explicit formula.

Let us review these three steps in detail.

**Step 1: Demonstration of the preservation of unitary coefficients**

The following statements demonstrate the preservation of unitary coefficients in the transfer function of CPFs.

**Statement 1:** The CPF with index $p=1$ has a transfer function with unitary coefficients.

**Proof:** The transfer function for a CPF with index $p=1$ is $H_1(z)=1-z^{-1}$ [19]. ■

**Statement 2:** Any CPF whose index $p$ is a prime number has a transfer function with unitary coefficients.

**Proof:** From Property 1 in (4.7), it follows that for any prime index $p$, the corresponding CPF has both, recursive and non-recursive transfer functions with coefficients in the set $\{-1, 0, 1\}$. ■

Given this, we only need to show that a CPF with a composite index $p$ has a transfer function with coefficients in the set $\{-1, 0, 1\}$. 

226
**Statement 3:** Any CPF with composite index $p$ has the same absolute-value coefficients as any CPF with index $q \geq 1$, where $q$, when is greater than 1, is an odd number whose prime factors are the odd distinct prime factors of $p$.

**Proof:** Consider a CPF with composite index $p$ expressed as

$$p = \prod_{i=1}^{m} q_i^{k_i},$$

(4.22)

where $q_1, q_2, \ldots, q_m$ are the distinct prime factors of $p$, respectively repeated $k_1, k_2, \ldots, k_m$ times. Let us start with $m = 1$, i.e., with a CPF whose composite index $p$ is expressed as $p = q_1^{k_1}$, where $q_1$ is prime. From Property 2 in (4.8), it follows that the corresponding transfer function can be expressed in terms of the transfer function of another CPF with a smaller index $\tilde{q}_1$ as follows,

$$H_p(z) = H_{\tilde{q}_1}(z^{q_1^{k_1-1}}), \quad \tilde{q}_1 = q_1.$$  

(4.23)

Since the only change is the exponent in the argument $z$, the resulting filter $H_p(z)$ has the same coefficients as $H_{\tilde{q}_1}(z)$.

Now consider $m = 2$, i.e., $p = q_1^{k_1} q_2^{k_2}$. From Property 2 we have

$$H_p(z) = H_{\tilde{q}_1}(z^{q_1^{k_1-1}}), \quad \tilde{q}_1 = q_1 q_2^{k_2},$$  

(4.24)

$$H_{\tilde{q}_1}(z) = H_{\tilde{q}_2}(z^{q_2^{k_2-1}}), \quad \tilde{q}_2 = q_1 q_2,$$  

(4.25)

$$H_p(z) = H_{\tilde{q}_1}(z^{q_1^{k_1-1}}) = H_{\tilde{q}_2}((z^{q_2^{k_2-1}})^{q_1^{k_1-1}}).$$  

(4.26)

Equations (4.23) and (4.26) allow expressing the transfer function $H_p(z)$ in terms of a basic transfer function that corresponds to a CPF with index $\tilde{q}_i$, with $i = 1$ in (4.23) and $i = 2$ in (4.26). Note that (4.23) and (4.26) can be considered base cases for a demonstration by induction.

Using the same derivation as (4.22) and (4.25) to obtain (4.26), let us consider the following inductive hypothesis for up to $m$ prime factors,
\[ H_{q_i}(z) = H_{q_{i+1}}(z^{q_i^{k_{i+1}-1}}), \quad \text{for } i = 1, 2, ..., m, \quad (4.27) \]

\[
\tilde{q}_i = \begin{cases} 
\prod_{j=1}^{i} q_j \cdot \prod_{j=i+1}^{m} q_j^{k_j} & \text{for } i = 1, 2, ..., m-1, \\
\prod_{j=1}^{m} q_j & \text{for } i = m. \end{cases} 
(4.28) \]

\[ H_p(z) = H_{q_m}\left((\ldots(z^{q_m^{k_m^{-1}}q_{m-1}^{k_{m-1}-1}})\ldots)^{q_1^{k_1-1}}\right) = H_{q_m}(z^{\prod_{i=1}^{m} q_i^{k_i-1}}). \quad (4.29) \]

Now let us prove (4.27) to (4.29) by induction on \( m \). For the case \( m+1 \), we have from (4.28)

\[ \tilde{q}_m = q_{m+1}^{k_{m+1}} \prod_{i=1}^{m} q_i. \quad (4.30) \]

By applying Property 2 to the transfer function of a CPF whose index is given by (4.30) we get

\[ H_{\tilde{q}_m}(z) = H_{q_{m+1}}(z^{\tilde{q}_m^{k_{m+1}-1}}). \quad (4.31) \]

The index of the CPF in the right hand of (4.31) can be expressed according to (4.28) as \( \tilde{q}_{m+1} \). Thus, (4.31) proves (4.27) for \( m+1 \). Substituting (4.31) in (4.29) we have

\[ H_p(z) = H_{q_{m+1}}(z^{q_{m+1}^{k_{m+1}-1}}) = H_{q_{m+1}}(z^{\prod_{i=1}^{m+1} q_i^{k_i-1}}), \quad (4.32) \]

which proves (4.29) for \( m+1 \).

Note that Property 2 allows proving (by induction on \( m \)) that the transfer function of any CPF whose index \( p \) is given as (4.22) can be expressed with the same coefficients of a CPF with index \( \tilde{q}_m \) that is composed by the distinct (i.e., non-repeated) prime factors of \( p \).

Note that, if \( \tilde{q}_m \) is odd, this index becomes \( q \). On the other hand, from Property 3 in (4.9), it follows that for any even index \( \tilde{q}_m \) which has only one
prime factor equal to 2, the corresponding transfer function can be expressed in terms of the transfer function of another CPF with a smaller odd index given by \( q = \tilde{q}_m / 2 \) as follows,

\[
H_{\tilde{q}_m}(z) = H_q(-z).
\tag{4.33}
\]

Since the only change is the sign in the argument \( z \), the resulting filter \( H_{\tilde{q}_m}(z) \) has the same coefficients as \( H_q(z) \), only with a sign change in the coefficients that are multiplied by the argument \( z \) with odd exponent. Note that, if \( p = 2^k \), we have \( \tilde{q}_m = \tilde{q}_1 = 2 \) and \( q = 1 \). Therefore, from Properties 2 and 3, it follows that for any composite index \( p \), the CPF has a transfer function with the same absolute-value coefficients of a CPF with index \( q \), where \( q \), when is greater than 1, is an odd number whose prime factors are all distinct of each other. ■

At this point, it is only necessary to show that a CPF with an odd composite index \( q \) having only non-repeated prime factors has a transfer function with coefficients in the set \( \{-1, 0, 1\} \).

**Statement 4:** Any CPF with index \( q \) having only odd distinct prime factors has a transfer function with unitary coefficients.

**Proof:** Consider that \( q \) is an odd number whose \( n \) prime factors, \( q_1, q_2, \ldots, q_n \), are all distinct of each other. The case for \( n=1 \) is trivial and follows directly from Statement 2. Thus, let us start with \( n=2 \). From Property 4 in (4.10), we observe that the prime number \( q_1 \) can be used as exponent of the argument \( z \). Therefore the transfer function of the filter \( H_q(z) \) is expressed as

\[
H_q(z) = H_{q_1q_2}(z) = \frac{H_{q_1}(z^{q_1})}{H_{q_2}(z)} = H_{q_1}^{-1}(z)H_{q_2}(z^{q_1}).
\tag{4.34}
\]
Since \( q^2 \) is a prime number, from Statement 2 we have that the expression in (4.34) preserves unitary coefficients if the product at the right hand is not expanded. Clearly, for any CPF whose index \( q \) has only two different odd prime factors, the corresponding transfer function can be constructed with a product of transfer functions with unitary coefficients.

Now consider \( n = 3 \). From Property 4 we have

\[
H_q(z) = H_{q_2q_3}(z) = \frac{H_{q_2}(z^{q_3})}{H_{q_3}(z)} = H_{q_2}^{-1}(z)H_{q_3}(z^{q_2}), \tag{4.35}
\]

\[
H_{q_2q_3}(z) = \frac{H_{q_2}(z^{q_3})}{H_{q_3}(z)} = H_{q_2}^{-1}(z)H_{q_3}(z^{q_2}). \tag{4.36}
\]

Substituting (4.36) in (4.35) we have

\[
H_q(z) = H_{q_2}(z)H_{q_3}(z^{q_2})H_{q_2}^{-1}(z^{q_3})H_{q_3}^{-1}(z^{q_2})H_{q_3}(z^{q_2}). \tag{4.37}
\]

From Statement 2, we have that \( H_{q_3}(z) \) has unitary coefficients. Thus, a CPF whose index \( q \) has three different odd prime factors has a transfer function with unitary coefficients if the product at the right hand of (4.37) is not expanded.

Equations (4.34) and (4.37) allow expressing the transfer function \( H_q(z) \) as the product of a basic transfer function that corresponds to a CPF with index \( q_i \) with \( i = 2 \) in (4.34) and \( i = 3 \) in (4.37). The index \( q_i \) is, in fact, one of the prime factors of \( q \) and, from Statement 2, it is demonstrated that such basic transfer function have unitary coefficients.

Let us consider the following inductive hypothesis for up to \( n \) prime factors,

\[
H_{\prod_{j=1}^{n} q_j}(z) = \frac{H_{\prod_{j=1}^{n-1} q_j}^{-1}(z)}{H_{\prod_{j=1}^{n-1} q_j}^{-1}(z^{q_n})}H_{\prod_{j=1}^{n-1} q_j}^{-1}(z^{q_n}) \quad \text{for } j = 1, 2, \ldots, n-1. \tag{4.38}
\]
If (4.38) is applied for every value \( j \), starting with \( j = n-1 \), the first result expresses the transfer function

\[
H_{\eta_{n-j}q_j}(z) = H_{\eta_1}^{-1}(z)H_{\eta_{n-1}}(z^{\eta_{n-1}}),
\]

(4.39)

which has unitary coefficients because \( q_n \) is prime (this follows from Statement 2). A substitution of the first result into the second result (i.e., when \( j = n-2 \)), then the second result into the third result and so forth, can follow from Property 4. The reason is that the exponent of the argument \( z \) in the right hand of each one of these results is a prime number and it does not divide the product of the other prime factors which constitute the index of the CPF at the left hand. Note that, if the product at the right hand is not expanded, every result preserves unitary coefficients.

Let us prove (4.38) by induction on \( n \). For the case \( n+1 \) we have from (4.38)

\[
H_{\prod_{i=j}^{n+1} q_i} (z) = H_{\prod_{i=j}^{n+1} q_i}^{-1} (z)H_{\prod_{i=j}^{n+1} q_i} (z^{q_i}) \quad \text{for } j=1, 2, \ldots, n.
\]

(4.40)

Let us define new indexes as follows,

\[
\tilde{q}_{n-j} = q_j, \quad i=1, 2, \ldots, n+1.
\]

(4.41)

Using these indexes in (4.40) results in

\[
H_{\prod_{i=j}^{n+1} \tilde{q}_i} (z) = H_{\prod_{i=j}^{n+1} \tilde{q}_i} (z)H_{\prod_{i=j}^{n+1} \tilde{q}_i} (z^{\tilde{q}_i}) \quad \text{for } j=1, 2, \ldots, n.
\]

(4.42)

Since \( \tilde{q}_i \) are prime numbers distinct of each other, we can use (4.38) in the right hand of (4.42), resulting in

\[
H_{\prod_{i=j}^{n+1} \tilde{q}_i} (z) = H_{\prod_{i=j}^{n+1} \tilde{q}_i} (z) \left[ H_{\prod_{i=j}^{n+1} \tilde{q}_i}^{-1} (z)H_{\prod_{i=j}^{n+1} \tilde{q}_i} (z^{\tilde{q}_i}) \right]^{-1} \times
\]

\[
H_{\prod_{i=j}^{n+1} \tilde{q}_i} (z^{\tilde{q}_i})H_{\prod_{i=j}^{n+1} \tilde{q}_i} (z^{\tilde{q}_i-1})
\]

(4.43)

Using \( j = n-1 \) and recovering indexes from (4.41) we have
\[ H_{\eta_{n-1} \eta_{n+1}}(z) = H_{\eta_{n+1}}(z) H_{\eta_{n}}^{-1}(z^{\eta_{n}}) H_{\eta_{n}}^{-1}(z^{\eta_{n-1}}) H_{\eta_{n+1}}(z^{\eta_{n+1}}) \]  

(4.44)

Clearly, (4.44) is a product of the basic transfer function \( H_{\eta_{n+1}}(z) \) which, according to Statement 2, has unitary coefficients because \( q_{n+1} \) is a prime number. Moreover, (4.44) is equivalent to using (4.40) for two values of \( j \), namely, \( j=n-1 \) and \( j=n \), and then substituting the result for \( j=n \) into the one for \( j=n-1 \). Therefore, it is shown by induction on \( n \) that any CPF whose index \( q \) has only distinct odd prime factors, has a transfer function with unitary coefficients. ■

**Step 2: Derivation of an explicit formula**

From the previous step we have that the transfer function of any CPF with a composite index \( p \), whose \( m \) distinct prime factors \( q_1, q_2, \ldots, q_m \), are respectively repeated \( k_1, k_2, \ldots, k_m \) times, can be expressed in terms of the transfer function of a CPF with index \( q \). From (4.7), (4.28), (4.29) and (4.33), as well as from the first three transfer functions for CPFs in [19], we have

\[
H_p(z) = \begin{cases} 
1 - z^{-1}; & \text{if } p = 1, \\
1 + z^{-1} + z^{-2}; & \text{if } p = 3, \\
\frac{1 - z^{-p}}{1 - z^{-1}}; & \text{if } p \text{ is prime higher than } 3, \\
H_q(-x) \big|_{x = z^{\prod_{i=q_{n+1}}^{k_{n+1}} \forall p}^{k_{n+1}-1}}; & \text{if } p \text{ is even,} \\
H_q(x) \big|_{x = z^{\prod_{i=q_{n+1}}^{k_{n+1}} \forall p}^{k_{n+1}-1}}; & \text{if } p \text{ is composite odd.}
\end{cases}
\]

(4.45)

The index \( q \) is an odd number whose \( n \) prime factors are \( q_1, q_2, \ldots, q_n \), the distinct odd prime factors of \( p \). Note that, if there are no odd prime factors in \( p \) (i.e., \( p = 2^k \) and \( n=0 \)), we have \( q = 1 \). Also note that, for \( p \) prime, the recursive transfer function with unitary coefficients has been selected to express \( H_p(z) \) because of its low computational complexity. However, it is worth
highlighting that an exception occurs for the case $p = 3$, where the non-recursive transfer function has been used instead. In this case, the non-recursive transfer function results more convenient because it has the same computational complexity but a less number of delay elements.

Now, let us obtain an explicit formula for the transfer function of a CPF with index $q$. Clearly, for either $q = 1$ or $q$ prime, $H_q(z)$ can be directly obtained from (4.45). For a composite $q$ with $n \geq 2$, the transfer function can be found using (4.38). It is necessary just an evaluation of (4.38) for a few values $n$ over all values $j$ to observe the implicit regularity. Let us consider two cases, $n = 4$ and $n = 5$, and let us apply (4.38) for all $j$, starting with $j = n - 1$ and going backwards to $j = 1$, substituting each result into the subsequent one. The case $n = 2$ is given in (4.34) and case $n = 3$ is given in (4.37). For cases $n = 4$ and $n = 5$ we obtain (4.46) and (4.47) respectively,

$$H_q(z) = H_{\prod_{i=q_i}^1} (z) = H_{q_1}^{-1}(z)H_{q_2}^{-1}(z)H_{q_3}^{-1}(z)H_{q_4}^{-1}(z)H_{q_4}^{-1}(z) \times$$

$$H_{q_4}(z)H_{q_4}(z)H_{q_4}(z)H_{q_4}(z)H_{q_4}(z)H_{q_4}(z),$$

$$H_q(z) = H_{\prod_{i=q_i}^1} (z) = H_{q_5}^{-1}(z)H_{q_5}^{-1}(z)H_{q_5}^{-1}(z)H_{q_5}^{-1}(z)H_{q_5}^{-1}(z) \times$$

$$H_{q_5}(z)H_{q_5}(z)H_{q_5}(z)H_{q_5}(z)H_{q_5}(z)H_{q_5}(z) \times$$

$$H_{q_5}(z)H_{q_5}(z)H_{q_5}(z)H_{q_5}(z)H_{q_5}(z)H_{q_5}(z) \times$$

$$H_{q_5}(z)H_{q_5}(z)H_{q_5}(z)H_{q_5}(z)H_{q_5}(z)H_{q_5}(z) \times$$

$$H_{q_5}(z)H_{q_5}(z)H_{q_5}(z)H_{q_5}(z)H_{q_5}(z)H_{q_5}(z) \times$$

A careful examination of (4.34), (4.37), (4.46) and (4.47) results in the following observations:

- The resulting product is always expressed in terms of the basic transfer function $H_{q_n}(z)$, and there are $2^{n-1}$ of these factors.
- The exponent of the argument of each basic transfer function $H_{q_n}(z)$ is one of the possible products with $k$ elements, with $k = 0, 1, 2, \ldots, n-1$, \ldots, n-1,
from the set \( \{q_1, q_2, \ldots, q_{n-1}\} \), and every exponent is different of each other.

- All factors \( H_{q_i}(z) \) whose argument have a product with the same number of elements \( q_i \) with \( i = 1, 2, \ldots, n-1 \), have the same exponent.
- The exponent of \( H_{q_i}(z) \) can be either 1 or \(-1\), and it is different to the exponent of factors \( H_{q_i}(z) \) whose argument have a product with one element less or one element more \( q_i \).
- The exponent of \( H_{q_i}(z) \), when the exponent of its argument is 1, is given by \((-1)^{n-1}\).

A compact form that synthesizes the previous observations is

\[
H_q(z) = H_{q_{i_1}}^{(-1)^{n-1}}(z) \prod_{i=1}^{n-1} \prod_{j=1}^{C(n-1,i)} H_{q_{i_j}}^{(-1)^{n-1}}(z^{M_{k,1}^{n-1}(q_{i_1}, i, i)}) ,
\]

where \( C(n-1,i) \) denotes the binomial coefficient. The notation \( M_{k,1}^{n-1}(q_{i_1}, i, i) \) stands for the \( j \)-th product in the set of possible products formed using \( i \) factors \( q_k \), with \( k \) ranging from 1 to \( n-1 \). For example, \( M_{k,1}^{4}(q_{i_1}, 2, 4) \) is short for the fourth product in the set of possible products formed using 2 factors from \( q_1, q_2, q_3, \) and \( q_4 \), i.e., the fourth product of the set \( \{q_1q_2, q_1q_3, q_1q_4, q_2q_3, q_2q_4, q_3q_4\} \).

Thus, we have \( M_{k,1}^{4}(q_{i_1}, 2, 4) = q_3q_4 \).

Let us prove (4.48) by induction on \( n \). Clearly, (4.34), (4.37), (4.46) and (4.47) are base cases with \( n = 2, n = 3, n = 4 \) and \( n = 5 \), respectively. For the case \( n+1 \), we have from (4.38)

\[
H_{\prod_{i=1}^{n+1} q_i}(z) = H_{\prod_{i=1}^{n} q_i}^{-1}(z) \cdot H_{\prod_{i=1}^{n} q_i}(z^{\hat{q}_i}) .
\]

Using the indexes defined in (4.41) we can rewrite (4.49) as,

\[
H_{\prod_{i=1}^{n+1} q_i}(z) = H_{\prod_{i=1}^{n} \hat{q}_i}(z) = H_{\prod_{i=1}^{n} \hat{q}_i}^{-1}(z) \cdot H_{\prod_{i=1}^{n} \hat{q}_i}(z^{\hat{q}_i}) .
\]
Since $q_i$ are prime numbers distinct of each other, we can use (4.48) in the right hand of (4.50), resulting in

$$H_{\prod_{i=1}^{n} q_i} (z) = H_{\prod_{i=1}^{n} q_i} (z) = H^{(-1)^{n-1}}_{\prod_{i=1}^{n} q_i} (z) \prod_{i=1}^{n-1} C(n-1,i) \prod_{j=1}^{n-1} H^{(-1)^{n-2i}}_{\prod_{i=1}^{n} q_i} (z) \times \left[ H^{(-1)^{n-1}}_{\prod_{i=1}^{n} q_i} (z) \prod_{i=1}^{n-1} C(n-1,i) \prod_{j=1}^{n-1} H^{(-1)^{n-2i}}_{\prod_{i=1}^{n} q_i} (z) \right]^{-1} \times \left[ H^{(-1)^{n-1}}_{\prod_{i=1}^{n} q_i} (z) \prod_{i=1}^{n-1} C(n-1,i) \prod_{j=1}^{n-1} H^{(-1)^{n-2i}}_{\prod_{i=1}^{n} q_i} (z) \right].$$

Recovering indexes results in

$$H_{\prod_{i=1}^{n} q_i} (z) = H^{(-1)^{n-1}}_{\prod_{i=1}^{n} q_i} (z) \prod_{i=1}^{n-1} C(n-1,i) \prod_{j=1}^{n-1} H^{(-1)^{n-2i}}_{\prod_{i=1}^{n} q_i} (z) \times \left[ H^{(-1)^{n-1}}_{\prod_{i=1}^{n} q_i} (z) \prod_{i=1}^{n-1} C(n-1,i) \prod_{j=1}^{n-1} H^{(-1)^{n-2i}}_{\prod_{i=1}^{n} q_i} (z) \right]^{-1} \times \left[ H^{(-1)^{n-1}}_{\prod_{i=1}^{n} q_i} (z) \prod_{i=1}^{n-1} C(n-1,i) \prod_{j=1}^{n-1} H^{(-1)^{n-2i}}_{\prod_{i=1}^{n} q_i} (z) \right].$$

It can be shown that (4.52) is equal to

$$H_{\prod_{i=1}^{n} q_i} (z) = H^{(-1)^{n-1}}_{\prod_{i=1}^{n} q_i} (z) \prod_{i=1}^{n-1} C(n-1,i) \prod_{j=1}^{n-1} H^{(-1)^{n-2i}}_{\prod_{i=1}^{n} q_i} (z) \times \left[ H^{(-1)^{n-1}}_{\prod_{i=1}^{n} q_i} (z) \prod_{i=1}^{n-1} C(n-1,i) \prod_{j=1}^{n-1} H^{(-1)^{n-2i}}_{\prod_{i=1}^{n} q_i} (z) \right],$$

which essentially is the same as (4.48) but for $n+1$. This proves (4.48) by induction on $n$.

Finally, the combination of (4.38), (4.45) and (4.48) results in (4.21).

**Step 3: Demonstration of the minimum number of additions**

Note from (4.21) that the CPF with index $p$ requires $2^n$ additions. In the following we show that the transfer function of any CPF can be expressed with no less than $2^n$ additions. Thus, the transfer function with minimum number of additions is the one stemming from (4.21).

The demonstration that $2^n$ is the minimum number of additions for any CPF with index $p$ having $n$ distinct odd prime factors can be made by means of the following statements:
Statement 5: Any transfer function of a CPF whose index is $p=2^l q^k$, where $q$ is a prime number and $l$ and $k$ are non-negative integers, has the minimum number of additions if it is written with (4.21).

Proof: Note that $l=k=0$ correspond to $p=1$. From (4.21) we see that cases with $l$ being a natural number and $k=0$ require the same number of additions as in a CPF with index $p=1$ and these cases require 1 addition. For other values of $l$ and $k$, the same number of additions as in a CPF with index $q$ is used, and this number is equal to 2. From (4.2) we have that at least 1 addition is required to form $H_1(z)$. Therefore, this is the minimum number of additions for all cases that have the same number of additions as a CPF with index $p = 1$. From (4.7) we have that the minimum number of additions is 2 for any CPF whose index is prime. Therefore, the other cases that use 2 additions and require the same number of additions as a CPF with prime index have the minimum number of additions. ■

Statement 5 proves that the transfer functions with 1 and 2 additions in (4.21) have the minimum number of additions. Therefore, now we have to prove that the transfer function corresponding to cases with $n \geq 2$, i.e., the product of basic transfer functions $H_{q^k}(z)$, has also the minimum number of additions.

Statement 6: Any transfer function written as a product of basic transfer functions $H_{q^k}(z)$ is irreducible. In other words, there is no cancellation of terms in the product of the transfer functions.
Proof: From (4.21) we can observe that, for any value \( n \) (with \( n \) at least equal to 2), we have only one basic transfer function \( H_{q_s}(z) \) whose argument has an exponent equal to 1. Additionally, we have \( C(n-1, i) \) basic transfer functions whose respective arguments have the different combinations of \( i \) factors \( q_s \), where the values of \( k \) range from 1 to \( n-1 \) and \( i \) goes from 1 to \( n-1 \). Therefore, all these basic transfer functions have different numerators and denominators since all the exponents in their arguments are different. Taking into account this characteristic, the observation that it is not possible to cancel numerator and denominator pairs that differ from each other easily follows. ■

Statement 7: Expanding the products obtained by (4.21) always results in higher number of additions when it comes to represent the overall transfer function. In other words, the representation with the product of basic transfer functions has the lowest number of additions.

Proof: Consider the following multiplication

\[
\frac{1-z^{-d}}{1-z^{-b}} \cdot \frac{1-z^{-c}}{1-z^{-d}} = \frac{1-z^{-d}-z^{-c}+z^{-(a+c)}}{1-z^{-b}-z^{-d}+z^{-(b+d)}}.
\] (4.54)

Note that upon accomplishing the product of at least two basic transfer functions (this product is represented in the left hand side of (4.54)) we get a transfer function having two more additions (this transfer function is shown in the right hand side of (4.54)). Therefore, the minimum number of additions is obtained preserving the product of basic transfer functions without expanding it. ■
4.3.2 Improvements on forming a suitable search space

The theorem of preservation of unitary coefficients shows that the upper bound $B$ for the index of CPFs in an initial search space is not dependent on the complexity of the filters, as originally was supposed in literature. Therefore, the indexes can have an upper bound as high as necessary and it is possible to take advantage of CPFs with indexes higher than $B=104$ or $B=200$. A natural question is: What should be the value for $B$?

In subsection 4.2.1 (where the second step of the algorithm to obtain the transfer functions of CPFs was described) we mentioned that, in the frequency range from $0$ to $\pi$, the zeros of a CPF whose index is $p$ are placed over the frequencies $\omega_i = \frac{2\pi k_i}{p}$, for all $k_i$ in the set of integers coprime to $p$, $\{k_1, k_2, \ldots k_i, \ldots, k_{\phi(p)}\}$, that are less or equal to $p/2$. Note that, since $1$ is coprime to any integer, the first zero of any CPF is located in the frequency $\omega_1 = \frac{2\pi}{p}$. For the case of lowpass designs the relation $\omega_1 > \omega_p$ must be accomplished, where $\omega_p$ is the passband edge frequency of the desired filter. Therefore, the upper bound $B$ must be the highest integer less than $\frac{2\pi}{\omega_p}$, i.e.,

$$
B = \begin{cases} 
\frac{2\pi}{\omega_p} - 1; & \text{if } \frac{2\pi}{\omega_p} \text{ is integer,} \\
\left\lfloor \frac{2\pi}{\omega_p} \right\rfloor; & \text{otherwise.}
\end{cases} 
$$

(Note that, due to their passband droop, many CPFs with indexes below or equal to $B$ have a passband droop that does not satisfy the passband ripple requirement. Thus, the search space (with indexes ranging from $p=1$ to $p=B$) can be substantially reduced by first evaluating the passband of each CPF and then by eliminating the CPFs that do not accomplish the passband
specification. After performing this selection, we obtain a preliminary set of
eligible CPF indexes, which we call \( \tilde{S} \).

Moreover, the selection of eligible CPFs can be improved by considering
the useful zeros of every CPF along with its corresponding cost. The
proposed way to choose an eligible CPF consists in assigning a profit
coefficient \( \psi_i \) to the CPF with index \( p_i \in \tilde{S} \) as follows,
\[
\psi_i = w_i - c_i, \tag{4.56}
\]
where \( w_i \) is the number of useful zeros, i.e., zeros that lie in the stopband
region, and \( c_i \) is the cost of the CPF (see (4.16)). Once all profit coefficients are
available, the set of all eligible CPFs, \( S \), is constituted by all CPFs that have a
profit coefficient equal or greater to a threshold given by the average profit.
The average profit is defined as
\[
\psi_{average} = \frac{1}{|\tilde{S}|} \sum_{i=1}^{|\tilde{S}|} \psi_i, \tag{4.57}
\]
where \( |\tilde{S}| \) is the number of elements in \( \tilde{S} \).

From the previous discussion, we propose to form the set of eligible
indexes, \( S \), with the following 5 steps:

1. Define \( B \) using (4.55), set a counter index \( i = 1 \) and set \( p = 1 \).
2. Find the magnitude response of the CPF with index \( p \) using \(|H_p(z)|\),
   with \( z = e^{j\omega} \) and \( H_p(z) \) given in (18), and evaluate it in passband and in
   stopband regions.
3. If the CPF accomplishes the desired deviation in the passband, set \( p_i = p \),
   include \( p_i \) in \( \tilde{S} \), find the corresponding profit \( \psi_i \) using (4.56) and then
   set \( i = i+1 \). Otherwise, go directly to step 4.
4. Set \( p = p+1 \) and, if \( p \leq B \), repeat from step 2. Otherwise, go to step 5.
5. For all indexes \( p_i \) in \( \tilde{S} \), with \( i \) going from 1 to \( |\tilde{S}| \), include the index \( p_i \) in \( S \) if \( \psi_i \geq \psi_{\text{average}} \), with \( \psi_{\text{average}} \) given in (4.57).

The aforementioned steps provide a more careful selection of the search space \( S \) and produce a small and more effective set \( S \) in comparison to the original scheme used in methods [19], [20]. As an instance, Table 4.5 shows the set of eligible CPFs to solve the filter design problem given in Example 3. Comparing with Table 4.1, it is clear that the search space is still effective since the solution indexes \( p_2 = 23 \), \( p_5 = 29 \), \( p_6 = 31 \) are included.

<table>
<thead>
<tr>
<th>( p_1 = 19 )</th>
<th>( p_2 = 23 )</th>
<th>( p_3 = 25 )</th>
<th>( p_4 = 27 )</th>
<th>( p_5 = 29 )</th>
<th>( p_6 = 31 )</th>
<th>( p_7 = 32 )</th>
<th>( p_8 = 34 )</th>
<th>( p_9 = 37 )</th>
<th>( p_{10} = 38 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p_{11} = 40 )</td>
<td>( p_{12} = 41 )</td>
<td>( p_{13} = 43 )</td>
<td>( p_{14} = 44 )</td>
<td>( p_{15} = 46 )</td>
<td>( p_{16} = 47 )</td>
<td>( p_{17} = 48 )</td>
<td>( p_{18} = 52 )</td>
<td>( p_{19} = 54 )</td>
<td></td>
</tr>
</tbody>
</table>

It is worth highlighting that reducing the size of the set \( S \) by using (4.56) can make the solution sub-optimal, since the complete search space is not available. Thus, the profit coefficient can be also considered directly as \( \psi_i = w_i \) and the resulting set \( S \) will contain more elements, depending on how low is the threshold to compare the profit coefficient. Note that, even if this boundary is made arbitrarily low (let say, a CPF is considered eligible if it has one or more useful zeros, i.e., if \( w_i \geq 1 \)), the resulting search space will be still smaller than the one obtained with methods [19], [20] because we have previously discarded all CPFs with a passband droop equal or greater than the passband specification.
4.3.3 Design examples and discussion of results

In the following paragraphs, two design examples are realized to show the effectiveness of the theorem of unitary coefficients, which allows using an arbitrarily extended search space where each CPF has the minimum computational complexity.

**Example 5**: consider the design of a CPF-based filter with the following specification,

- passband, \( \bar{\omega}_1 = [0, 0.001653\pi] \),
- stopbands, \( \bar{\omega}_k = [0.03637(k-1)\pi-0.001653\pi, \ 0.03637(k-1)\pi+0.001653\pi] \) with \( k = 2, 3, \ldots, 28 \),
- passband ripple, \( R_1 = 0.1 \text{ dB} \),
- minimum attenuation in the \( k \)-th stopband, \( A_k = 80 \text{ dB} \).

Let us follow a design procedure based on the proposed steps detailed in section 4.2.1. In **Step 1** a database containing the transfer functions and a LUT with the costs for all the CPFs is required. Note that, instead of using either \( B=104 \) or \( B=200 \), the upper bound \( B \) can be obtained with (4.55). Moreover, from the theorem of unitary coefficients it is possible to obtain the recursive transfer functions by means of (4.21) and the costs can be known in advance. Thus, it is not necessary to use the algorithm described by the flowchart in Fig. 4.3. Since a reduction of the search space must be realized, it is not necessary to obtain the costs at this step, but only the transfer functions.
In order to reduce the initial search space it is necessary to know the passband magnitude response characteristic of every CPF with index below \( B \). Thus, Step 3 must be next. Note that we only need to obtain the worst-case passband values. Once the initial search space has been reduced, we obtain the worst-case stopband values. After that, we must use Step 2 to obtain the useful zeros of every CPF. Also the cost of every CPF must be obtained. This cost is calculated with the following MATLAB command: \( 2^\left( \text{length}\left( \text{find}\left( \text{unique}\left( \text{factor}\left( p \right) \right) > 2 \right) \right) \right) \), where \( p \) is the index of the CPF. With the costs and the number of useful zeros of each CPF we use (4.56) to calculate the profit coefficients and the set of eligible CPFs is formed. Finally, we solve the ILP problem as in Step 4.

After the selection procedure, 87 eligible indexes are found, as shown in Table 4.6. The obtained results after the optimization are: \( m_4=1, m_{10}=1, m_{31}=1, m_{44}=1, m_{46}=2, m_i=0 \) for other values of \( k \).

Therefore, the transfer function of the resulting CPF-based filter is:

\[
H(z) = H_{p_4}^{m_4}(z) \cdot H_{p_{10}}^{m_{10}}(z) \cdot H_{p_{31}}^{m_{31}}(z) \cdot H_{p_{44}}^{m_{44}}(z) \cdot H_{p_{46}}^{m_{46}}(z) = \\
H_5(z) \cdot H_{11}(z) \cdot H_{37}(z) \cdot H_{53}(z) \cdot H_{55}^2(z) = \\
\left[ \frac{1-z^{-5}}{1-z^{-1}} \right] \cdot \left[ \frac{1-z^{-11}}{1-z^{-1}} \right] \cdot \left[ \frac{1-z^{-37}}{1-z^{-1}} \right] \cdot \left[ \frac{1-z^{-53}}{1-z^{-1}} \right] \cdot \left[ \frac{1-z^{-11}}{1-z^{-5}} \right] \cdot \left[ \frac{1-z^{-5}}{1-z^{-1}} \right] \cdot \left[ \frac{1-z^{-11}}{1-z^{-5}} \right].
\]

Figure 4.8 presents the magnitude characteristic of \( H(z) \).

Example 5 has been also solved with the procedure proposed in Section 4.2.1. The resulting values after the optimization are \( m_4=1, m_{20}=1, m_{24}=1, m_{26}=1, m_{31}=1, m_{45}=2, m_{46}=1, m_i=0 \) for other values of \( k \). The resulting transfer function requires 20 adders. Table 4.7 shows the cost for every solution.

Note that the filter designed with the method proposed in Section 4.2.1 is a result of an optimization over a set of eligible CPFs formed from transfer
functions previously derived in literature. On the other hand, the filter
designed with CPFs obtained from (4.21) takes advantage of the CPF \( H_{55}(z) \),
using it two times, because the search space has been formed from the
theorem of preservation of unitary coefficients and all the eligible CPFs have
their lowest computational complexity. As a consequence, there is a reduction
of 20% in the arithmetic complexity.

Table 4.6: Eligible indexes of CPFs in Example 5

<table>
<thead>
<tr>
<th>( p_1 = 2 )</th>
<th>( p_2 = 3 )</th>
<th>( p_3 = 4 )</th>
<th>( p_4 = 5 )</th>
<th>( p_5 = 6 )</th>
<th>( p_6 = 7 )</th>
<th>( p_7 = 8 )</th>
<th>( p_8 = 9 )</th>
<th>( p_9 = 10 )</th>
<th>( p_{10} = 11 )</th>
<th>( p_{11} = 12 )</th>
<th>( p_{12} = 13 )</th>
<th>( p_{13} = 14 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p_{14} = 16 )</td>
<td>( p_{15} = 17 )</td>
<td>( p_{16} = 18 )</td>
<td>( p_{17} = 19 )</td>
<td>( p_{18} = 20 )</td>
<td>( p_{19} = 22 )</td>
<td>( p_{20} = 23 )</td>
<td>( p_{21} = 24 )</td>
<td>( p_{22} = 25 )</td>
<td>( p_{23} = 26 )</td>
<td>( p_{24} = 27 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( p_{25} = 28 )</td>
<td>( p_{26} = 29 )</td>
<td>( p_{27} = 31 )</td>
<td>( p_{28} = 32 )</td>
<td>( p_{29} = 34 )</td>
<td>( p_{30} = 36 )</td>
<td>( p_{31} = 37 )</td>
<td>( p_{32} = 38 )</td>
<td>( p_{33} = 40 )</td>
<td>( p_{34} = 41 )</td>
<td>( p_{35} = 43 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( p_{36} = 44 )</td>
<td>( p_{37} = 46 )</td>
<td>( p_{38} = 47 )</td>
<td>( p_{39} = 48 )</td>
<td>( p_{40} = 49 )</td>
<td>( p_{41} = 50 )</td>
<td>( p_{42} = 51 )</td>
<td>( p_{43} = 52 )</td>
<td>( p_{44} = 53 )</td>
<td>( p_{45} = 54 )</td>
<td>( p_{46} = 55 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( p_{47} = 56 )</td>
<td>( p_{48} = 57 )</td>
<td>( p_{49} = 58 )</td>
<td>( p_{50} = 59 )</td>
<td>( p_{51} = 61 )</td>
<td>( p_{52} = 62 )</td>
<td>( p_{53} = 63 )</td>
<td>( p_{54} = 64 )</td>
<td>( p_{55} = 67 )</td>
<td>( p_{56} = 68 )</td>
<td>( p_{57} = 69 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( p_{58} = 71 )</td>
<td>( p_{59} = 72 )</td>
<td>( p_{60} = 73 )</td>
<td>( p_{61} = 74 )</td>
<td>( p_{62} = 76 )</td>
<td>( p_{63} = 79 )</td>
<td>( p_{64} = 80 )</td>
<td>( p_{65} = 81 )</td>
<td>( p_{66} = 82 )</td>
<td>( p_{67} = 83 )</td>
<td>( p_{68} = 86 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( p_{69} = 87 )</td>
<td>( p_{70} = 88 )</td>
<td>( p_{71} = 89 )</td>
<td>( p_{72} = 91 )</td>
<td>( p_{73} = 92 )</td>
<td>( p_{74} = 93 )</td>
<td>( p_{75} = 94 )</td>
<td>( p_{76} = 96 )</td>
<td>( p_{77} = 97 )</td>
<td>( p_{78} = 98 )</td>
<td>( p_{79} = 100 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( p_{80} = 101 )</td>
<td>( p_{81} = 104 )</td>
<td>( p_{82} = 106 )</td>
<td>( p_{83} = 108 )</td>
<td>( p_{84} = 112 )</td>
<td>( p_{85} = 114 )</td>
<td>( p_{86} = 116 )</td>
<td>( p_{87} = 120 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 4.8: Magnitude response of the CPF-based filter designed in Example 5.
Table 4.7: Comparison of results in Example 5

<table>
<thead>
<tr>
<th>Filter design</th>
<th>No. of Additions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method based on transfer functions from [19]</td>
<td>20</td>
</tr>
<tr>
<td>Method based on the theorem of preservation of</td>
<td>16</td>
</tr>
<tr>
<td>unitary coefficients</td>
<td></td>
</tr>
</tbody>
</table>

**Example 6:** consider the design of a CPF-based filter with the following specification,

- passband, $\tilde{\omega}_1 = [0, 0.000397\pi]$,
- stopbands, $\tilde{\omega}_k = [0.00873(k-1)\pi-0.000397\pi, 0.00873(k-1)\pi+0.000397\pi]$ with $k = 2, 3, \ldots, 115$,
- passband ripple, $R_1 = 0.1$ dB,
- minimum attenuation in the $k$-th stopband, $A_k = 80$ dB.

We use the same design procedure as in Example 5, i.e., we take advantage of the theorem of preservation of unitary coefficients to form the set of eligible CPFs. The set of eligible indexes contains 180 elements as shown in Table 4.8. The obtained results after the optimization are: $m_6=1, m_{22}=1, m_{26}=2, m_k=0$ for other values of $k$.

Therefore, the transfer function of the resulting CPF-based filter is:

$$H(z) = H_{P_6}^{m_6}(z) \cdot H_{P_{22}}^{m_{22}}(z) \cdot H_{P_{26}}^{m_{26}}(z) = H_{163}(z) \cdot H_{217}(z) \cdot H_{229}(z) = \left[ \frac{1-z^{-163}}{1-z^{-1}} \right] \cdot \left[ \frac{1-z^{-217}}{1-z^{-31}} \right] \cdot \left[ \frac{1-z^{-1}}{1-z^{-7}} \right] \cdot \left[ \frac{1-z^{-229}}{1-z^{-1}} \right]^2.$$

Figure 4.9 presents the magnitude characteristic of $H(z)$. 

244
Note that the obtained solution takes advantage of CPFs with indexes $p_6=163$, $p_{22}=217$ and $p_{26}=229$. None of them is available in the classical set with the first 104 CPFs used in literature, and neither $p_{22}=217$ nor $p_{26}=229$ are contained in the set with the first 200 CPFs, explored in Section 4.2.1. Therefore, with those limited search spaces the algorithm is just capable to reach sub-optimal solutions. Additionally, the same design problem can not be solved by using only RRS filters. Table 4.9 shows a comparison with
methods [25] and [26]. The CPF-based design achieves about 28.6% of arithmetic complexity reduction against [26] and it does not require multipliers, as [25].

![Figure 4.9: Magnitude response of the CPF-based filter designed in Example 6.](image)

<table>
<thead>
<tr>
<th>Filter design</th>
<th>No. of Additions</th>
<th>No. of Multiplications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method [25]</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>Method [26]</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>CPF-based filter</td>
<td>10</td>
<td>0</td>
</tr>
</tbody>
</table>

Finally, it is worth highlighting the following discussions.

- CPFs with recursive structures can preserve unitary coefficients. The use of the explicit formula derived from the theorem of preservation of unitary coefficient provides an effective way to obtain CPFs whose transfer functions have the lowest number of additions. Thus, it is possible to take advantage of CPFs with indexes as high as necessary.
As we observed in Section 4.2.2, CPFs with higher indexes are useful to design CPF-based filters with a narrow passband.

- When the index of a CPF is a prime number, this CPF is equivalent to a RRS filter. However, for composite indexes, CPFs are different than RRS filters. Therefore, RRS filters should not be considered a special class of CPFs, as stated in literature [4]-[8], because this statement can be interpreted as “RRS filters are a subset of CPFs”.

- The effectiveness of CPFs is not completely seen for decimation applications, where the stopbands are centered around $2\pi k/M$ for $k = 1, 2, \ldots, \lfloor M/2 \rfloor$, with $M$ being the decimation factor. The reason is that the traditional RRS filter has its zeros placed at the center of these stopbands, regardless if $M$ is prime or not, making it a good candidate for decimation filtering. On the other hand, CPFs have zeros in the center of these stopbands only if $M$ is a prime factor. As a consequence, using RRS filters for decimation with composite factors $M$ can be a more effective solution.

- Despite of the previous observation, a combination of CPFs may sometimes give a better solution for decimation filtering, even if the decimation factor is not a prime number. As an instance, a filter with the specifications in Example 3 can be used for decimation by $M = 25$ and residual factor $\nu = 3$. Note that a RRS filter with $M = 25$ is not available in the search space. However, a combination of CPFs with indexes 23, 29 and 31 allows placing the zeros over the first stopband (see Fig. 4.6) in a better manner, without any considerable increase of the passband droop. As a consequence, the resulting filter
accomplishes the passband and stopband requirements with only 6 adders. Note that it is not possible to reach the desired filter specifications only with a cascade of RRS filters for $M = 25$.

- For an $M$ prime, a CPF with index $p = M$ is equivalent to a RRS filter. Even though a RRS-based filter is a potential solution in such case, a combination of CPFs and RRS filters can be more effective. Example 6, where $M = 25$ and $\nu = 11$, shows that it is not possible achieving a solution with only RRS filters (i.e., by using only CPFs with index $p = M$), but the solution is reached by using this RRS filter along with other CPFs.

- CPFs do not have an additional advantage that RRS filters have for decimation applications, namely, the efficient implementation via Hogenauer architecture, also known as Cascaded-Integrator-Comb (CIC) structure (except for cases when the CPF and the RRS filter are the same). However, some operations of a CPF-based decimation filter can be moved to lower rate by applying noble identities, as explained in [19]. Moreover, when it is not possible to use noble identities directly, the numerator of the transfer function of a CPF-based filter can be moved to lower rate as a single non-recursive filter via polyphase decomposition. The resulting architecture does not have an enormous amount of operations at lower rate, which implies a high increase of area (this occurs when a CPF or RRS filter is completely implemented as a polyphase non-recursive structure), and its operations at lower rate imply low power consumption.
The computational complexity of recursive CPFs depends on the number of prime factors of the index of the CPF, discarding 2. Thus, using the first 104 or the first 200 CPFs as an initial search space is no longer justified with regard to the computational complexity. Therefore, the upper bound for the index of a CPF in the search space must be obtained in a different way as was explained in sub-section 4.3.2. On the other hand, the complexity of a CPF provides useful information on the selection of the eligible CPFs, as shown in (4.56). This refined eligibility criterion, even though simple, considerably reduces the size of the set of eligible indexes, thus decreasing the complexity of the optimization algorithm, without significantly compromising the optimality of the solution.

4.4 Conclusion

The use of Cyclotomic Polynomials Filters (CPFs) with indexes up to 104 has been attractive for multiplierless digital filter structures because their transfer functions are known to have unitary coefficients. However, the need for narrow passband filters, either as single filters for high decimation factors or as subfilters in Prefilter-Equalizer and Periodic-Filter-based filtering schemes, along with the possibility of getting narrower passband CPFs as the indexes increase, have set the motivation to explore CPFs with indexes beyond of the traditional upper bound. In this chapter we have developed a fruitful research that enlarges the capabilities of CPFs and extends the theoretical knowledge on these filters, especially for cases where recursive
structures, advisable when the computational complexity is a main concern, are used.

An efficient design procedure for CPF-based filters was developed, and the use of an extended search of CPFs with indexes up to 200 was studied. One of the main observations stemming from the results was that recursive transfer functions for all these CPFs still have coefficients in the set \{-1, 0, 1\}. Moreover, as was expected, this property yields filters with lower computational complexity than the ones designed over the usual search space of 104 CPFs, especially for narrowband filters. A promising matter identified from these results was the feasibility of proving that the recursive-form of CPFs can be computationally effective regardless of the index of the CPF.

After exploring the first 200 CPFs, we proved that any CPF, regardless of the value of its index, can be expressed as a cascade of recursive basic building blocks with unitary coefficients. Moreover, an explicit formula, which is the basis to obtain the transfer function of any CPF expressed as a product of basic building blocks with unitary coefficients, was developed. Additionally, it was proved that this formula provides a recursive transfer function with the minimum number of additions for any CPF. These demonstrations allowed us to develop the theorem of preservation of unitary coefficients, which constitutes the base to form an arbitrarily extended search space for the design of filters with arbitrarily narrow passband.

From the proposed theorem, the extended set of eligible CPFs is guaranteed to have the minimum number of additions if the CPFs are expressed as a cascade of basic building blocks. This theorem also clarifies the compromise between the index of the CPFs and their minimum
computational complexity. Thus, we developed a simple but refined eligibility criterion for CPFs when the eligible search space is under formation. Finally, the limitations of CPFs were also discussed with special emphasis in decimation applications, clarifying the relation between CPFs and Recursive Running Sum (RRS) filters.

4.5 References


This chapter introduces the contributions on the Single Constant Multiplication (SCM) and Multiple Constant Multiplication (MCM) approaches for the design of multiplierless arithmetic blocks, widely employed in FIR filters. We develop an extension to the theoretical lower bounds for the adder cost and adder depth in the SCM problem. With this extension, the hidden theoretical lower bound for the number of adders required to preserve the minimum adder depth is revealed. From this study we introduce a general algorithm to design multiplierless filters with minimum number of adders subject to the theoretical lower bound for the adder depth, which is shown to be proper for Field Programmable Gate Arrays (FPGA) implementations.
5.1 Introduction

Abundant research has been realized since the past two decades to solve Single Constant and Multiple Constant Multiplication problems (SCM and MCM, respectively), where the hardware requirements can be reduced by exploiting the constant coefficient characteristics known a priori [1]-[26]. In these cases, the multiplications are performed without using general multipliers and the only arithmetic operations are additions and subtractions. Besides these arithmetic operations, only powers of two are allowed. These powers of two are implemented using hardwired shifts and therefore are considered with no cost.

Even though MCM and SCM problems are closely related, the latter constitutes the fundamental basis for all the constant multiplication problems and is commonly solved by specialized, fine-tuned algorithms [1], [3]-[4], [6], [12]-[15]. Section 5.2 of this chapter is focused in the fundamental SCM case. We derive the extension of the current theoretical lower bounds, recently introduced in [17], for the number of arithmetic operations needed to implement the constant multiplier and the number of sequentially-connected operations forming a critical path. With this, the preliminary estimation of how many extra arithmetic operations are needed to preserve the lowest critical path in a SCM block becomes possible.

On the other hand, Field Programmable Gate Array (FPGA) platforms have become the dominating Digital Signal Processing (DSP) technology for real-time and high-speed systems mainly due to their flexibility and low non-recurring engineering costs. In FPGA-based designs the constant multiplications involving shifts, additions and subtractions, can be made
fully-pipelined with a low extra cost. Pipelining has a low overhead due to the fact that the logic blocks in FPGAs include memory elements, which are otherwise unused [18]. In Section 5.3 of this chapter, the addition-subtraction-shift approach for constant multipliers is considered along with the fully-pipelined characteristic adequate for FPGA-platforms. In this case we show that the lower bound for the number of pipelined operations is dependent on the critical path of the multiplier. As a result, we introduce an algorithm to design fully-parallel and fully-pipelined FIR filters, where the minimization of the number of arithmetic operations is subject to the theoretical lower bound for the critical path, resulting in low-complexity and high-speed solutions.

5.2 Inclusion of prime factors to calculate the theoretical lower bounds in Single Constant Multiplications (SCM)

The usual metric to minimize in the SCM algorithms has been the number of arithmetic operations needed to implement the constant multiplier. However, it has been reported that the number of sequentially-connected arithmetic operations forming a critical path has the main impact in performance and power consumption [3], [8]-[9]. Therefore, minimizing the number adders subject to a minimum critical path is a more effective goal.

Theoretical lower bounds for these two metrics in SCM, MCM and other related problems have been recently derived in [17] based on a simple number that can be calculated in advance from the involved constants,
namely, their Minimum Number of Signed Digits (MNSD). In [17], the theoretical lower bounds were first obtained for the simplest case of constant multiplications, SCM, and then deduced for cases involving multiple constants. Nevertheless, the compromise between the number of arithmetic operations and the critical path was only mentioned and the theoretical lower bound for such compromise still remained hidden. It has been pointed out in literature that using the lowest critical path often results in higher performance and lower power consumption at expenses of increasing the number of arithmetic operations. Similarly, the minimization of the number of arithmetic operations may result in higher area saving at expenses of an increased critical path [3], [8]-[9].

This section introduces the extension to the theoretical lower bounds given in [17] for the SCM case. Subsection 5.2.1 presents the key theorems from [6], used in [17] as a starting point. Subsection 5.2.2 develops an analysis of the SCM lower bounds from [17]. In subsection 5.2.3 we show that these bounds can be increased in certain cases which have not been exposed by including the number of prime factors of the constants. Additionally, the hidden theoretical lower bound for the number of arithmetic operations required to preserve the minimum critical path is revealed. Subsection 5.2.4 presents the comparison between the proposed lower bounds and the lower bounds from [17].

5.2.1 Key theorems for the current lower bounds

The main operation in SCM blocks, called $A$-operation in [7], is defined as

$$w = A_x(u, v) = 2^{e_1} u + (-1)^{e_2} 2^{e_3} v 2^{-e_5},$$  \hspace{1cm} (5.1)
where $e_1 \geq 0$, $e_2 \geq 0$ are integers denoting left shifts, $e_3 \geq 0$ is an integer indicating right shift, $e_4 \in \{0, 1\}$ chooses the addition or subtraction operation to be performed, $x = \{e_1, e_2, e_3, e_4\}$ is the parameter set or $A$-configuration of $A(x, u, v)$ and $u, v$ and $w$ are positive and odd constants. As additions and subtractions have a similar complexity when it comes to hardware implementation, they are usually referred without distinction as $A$-operations. An SCM block is designed as a network of $A$-operations represented using Direct Acyclic Graphs (DAGs) with the following characteristics [6], [16]-[17]:

- Shifts are assumed to be free. Additionally, the sign of the constants formed in the DAG is assumed to be adjusted at some part of the design. Therefore, only positive and odd integers are considered. These constants are known as fundamentals.
- For a graph with $n$ $A$-operations, there are $n + 1$ vertices and $n$ fundamentals. Every fundamental is obtained as a result of an $A$-operation.
- Each vertex has an in-degree two except for the input vertex which has in-degree zero.
- A vertex with in-degree two corresponds to an $A$-operation.
- Each vertex has out-degree larger than or equal to one except for the output vertex which has out-degree zero.
- The constant resulting from the last $A$-operation is known as Output Fundamental (OF), whereas the constants resulting from previous $A$-operations are Non-Output Fundamentals (NOFs). In a DAG that does not have the minimum number of $A$-operations the constants resulting from the extra $A$-operations are referred as Non-Essential
Fundamentals (NEFs).

The number of $A$-operations, $N_A$, is frequently called *adder cost*. However, this value will be referred here as *$A$-cost* for consistency. The number of cascaded $A$-operations, $N_d$, where the output of an $A$-operation is at least one input of another $A$-operation, is frequently called *logic depth* or *adder depth* and it will be referred here as *$A$-depth*.

The following theorems from [6] relate the number of nonzero digits of a constant with its $A$-depth.

Theorem A: The sum of two coefficients with $k_1$ and $k_2$ nonzero digits respectively, has at most $k_1+k_2$ nonzero digits.

Theorem B: A multiplier graph with $A$-depth equal to ‘d’ can generate coefficients with at most $2^d$ nonzero digits.

On the other hand, a *multiplicative graph* is the graph obtained by cascading two subgraphs such that the resulting OF is a product of the OFs of the two subgraphs (classification of graph structures is detailed in [6] and extended in [16].) An articulation point is the point where the output of the first subgraph is joined with the input of the second subgraph. Since in DAGs the $A$-operations become nodes, the $A$-operation whose output is an articulation point can be also referred as an articulation point.

Theorem C: If a graph has an articulation point, the graph is multiplicative.
Proofs of these theorems are given in [6] and more details can be found in [19].

5.2.2 Analysis of the current lower bounds

Consider a given constant $c$ whose MNSD, i.e., the number of digits obtained by representing $c$ in Canonic Signed Digit (CSD), is $S(c)$. The lower bounds $L_A$ and $L_d$ for the $A$-cost and the $A$-depth of the graph for $c$, respectively, are given in [16] as

$$L_A = L_d = \lceil \log_2 S(c) \rceil,$$  \hspace{1cm} (5.2)

where $\lceil x \rceil$ is the nearest integer greater than or equal to $x$. Taking into account that the MNSD can be expressed as $2^{p-1} < S(c) \leq 2^p$ for all $p \geq 1$ and $p$ integer, we can note that all values $S(c)$ in the range $(2^{p-1}, 2^p]$ have the same $L_A$ and $L_d$ according to (5.2), with

$$p = \lceil \log_2 S(c) \rceil.$$ \hspace{1cm} (5.3)

We will refer the range $(2^{p-1}, 2^p]$ as MNSD-range. Note that the function $S(c)$ is the unique information taken from the constants and used in [17] to derive the current lower bounds for constant multiplication problems.

An important characteristic given in [17] and related to the graph structure presented in Figure 5.1, which will be called here the Completely Multiplicative (CM) graph, is that if $S(c)=2^p$ (the highest MNSD in the MNSD-range), the lower bound $L_A = p$ will be obtained only with a CM graph.

![Figure 5.1: Completely Multiplicative (CM) graph.](image)
It is always possible to find a solution with the lower bound $L_d$ because the coefficient $c$ can be implemented by summing all its non-zero terms using $A$-operations arranged in binary tree [17]. Nevertheless, the lower bounds given in (5.2) do not make any distinction among the constants whose MNSD are into the MNSD-range.

### 5.2.3 Proposed extension of the current lower bounds

Let us start with the *CM-based graph* of Figure 5.2, which consists of the cascade of a subgraph $H$ with $p-lA$-operations and a CM subgraph with $lA$-operations. This graph is exploited in [15] to obtain optimal multiplications by rational constants with periodic binary representations, and it has the following characteristics:

1) At the output of every $A$-operation of the CM subgraph, the MNSD of the resulting constant is at most twice the MNSD of the constant at its inputs, according to Theorem A. If the MNSD at the output of the subgraph $H$, denoted by $n_H$, is the highest possible in $H$, the maximum resulting MNSD at the overall graph is $2^{l} \times n_H$.

2) If $H$ has the minimum $A$-depth, the overall graph has the minimum $A$-depth because the CM subgraph has also the minimum $A$-depth. The same holds for the $A$-cost.

3) If $H$ is non-multiplicative, the CM-based graph has $l$ articulation points formed with the minimum $A$-cost and $A$-depth and clustered in the higher depth levels. Thus, this graph has the highest MNSD in its first articulation point in comparison to any other graph with $l$ articulation.
points. Moreover, it can have the maximum MNSD with respect to other graph with $l$ articulation points.

![Diagram of subgraph H and CM subgraph with $p$ A-operations](image_url)

**Figure 5.2:** CM-based graph with $p$ A-operations.

Due to the their aforementioned characteristics, CM-based graphs will be analyzed in the following to show that the graph of a constant whose MNSD is given in specific intervals into the MNSD-range requires certain multiplicative characteristics to preserve the lower bound $L_A$. It shall be considered henceforth that the MNSD is given in the MNSD-range $(2^{p-1}, 2^p]$ with $p>1$, since $L_A=L_d=1$ for $S(c)=2$ (i.e., when $p=1$).

**Theorem 1:** A constant $c$ whose MNSD is $S(c) > 2^{p-1}+1$ can not be obtained with a non-multiplicative graph with $p$ A-operations.

**Proof:** Let us review the simplest case $p=2$. There are 2 possible graphs, one of them additive and the other one multiplicative (see Appendix A of [16]). The highest MNSD of any NOF in both graphs is equal to 2. The last A-operation of the non-multiplicative graph only can add a constant with one nonzero digit, thus yielding OFs whose highest MNSD is $2^{p-1}+1=3$ according to Theorem A. The highest MNSD of any OF in the multiplicative graph is equal to 4 because both inputs to the last A-operation can have up to 2 nonzero digits each. Thus, the case $p=2$ is a base for the following inductive
hypothesis: the highest MNSD of any OF in a non-multiplicative graph with $k$ $\mathbb{A}$-operations is equal to $2^{k-1}+1$ for all $1<k<k+1$. Assuming it as a true statement, let us construct a non-multiplicative graph with $k+1$ $\mathbb{A}$-operations, with the aim of obtaining an OF whose MNSD is $n=2^{k+2}$, i.e., the lowest MNSD that contradicts the hypothesis.

A non-multiplicative graph can not have the inputs of its last $\mathbb{A}$-operation (the $\mathbb{A}$-operation placed at the $p$-th depth level) coming from the same position because an articulation point is generated and the graph becomes multiplicative (see Theorem C). For the sake of clarity, let us identify the two inputs of the last $\mathbb{A}$-operation as $a_1$ and $a_2$. Let us assume, without loss of generality, that $a_1$ comes from the output of a subgraph $G_i$. This subgraph can be either non-multiplicative or multiplicative. In the following we will review each of these two cases, along with the point where $a_2$ can come from.

If $G_i$ is non-multiplicative, the highest MNSD of any of its OFs is $n_i=2^{k-1}+1$ (this follows from the inductive hypothesis). For the overall graph, the only way to obtain an OF whose MNSD is $n=2^{k+2}$ is having $a_2$ coming from an $\mathbb{A}$-operation that produces constants with an MNSD at least equal to $n-n_i = 2^{k-1}+1$. From Theorem B, we have that this MNSD can be obtained only from the $k$-th $\mathbb{A}$-operation, which means that $a_1$ and $a_2$ would be coming from the same $\mathbb{A}$-operation, generating an articulation point. If $a_2$ comes from the $(k-1)$-th $\mathbb{A}$-operation, the articulation point is avoided and the whole graph can still be non-multiplicative. However, the highest MNSD of a constant generated from the $(k-1)$-th $\mathbb{A}$-operation is equal to $2^{k-1}$ (see Theorem B) and thus the highest MNSD of the OF in the overall graph is equal to $n_1+2^{k-1} = 2^{k+1}+2^{k-1} = 2^{k+1}$. 
If $G_1$ is multiplicative, it can have $l$ articulation points where $1 \leq l \leq k-2$. With $l$ articulation points and $k$ operations, the highest MNSD of any OF produced by $G_1$ is $n_1 = 2^l \times (2^{k-l} + 1)$. This follows from the assumption that $G_1$ is a CM-based graph and from the inductive hypothesis. In order to obtain an OF whose MNSD is $n = 2^k + 2$, $a_2$ must come from an $A$-operation that produces constants with an MNSD at least equal to $n - n_1 = 2^{k-1} - 2^{k-l} + 2$. Moreover, such $A$-operation must be placed at a depth level less than the depth level where the first articulation point is placed; otherwise this articulation point will make the whole graph multiplicative. Since $G_1$ has its first articulation point at a depth level equal to $k-l$, $a_2$ must come from an $A$-operation placed in a depth level that does not exceed $k-l-1$. The highest MNSD of any NOF obtained from an $A$-operation placed in the $(k-l-1)$-th depth level is $2^{k-l-1}$ (from Theorem B). Hence, this value should be equal or greater than $n - n_1 = 2^{k-1} - 2^{k-l} + 2$ in order to obtain an OF in the overall graph whose MNSD is $n = 2^k + 2$. However, it can be shown that $2^{k-l-1} < 2^{k-1} - 2^{k-l} + 2$ holds (see Appendix A.1). Thus, the highest MNSD of any OF in the overall graph is equal to $n_1 + 2^{k-l-1} = 2^l \times (2^{k-l} + 1) + 2^{k-l-1} \leq 2^k + 1$ (see Appendix A.2).

As a result, we have that the inductive hypothesis for a graph with $k$ $A$-operations implies that it holds for $k+1$ $A$-operations. Therefore, considering this implication along with the base case with $p=2$, we prove Theorem 1 by induction on $p$. □

**Theorem 2:** A constant $c$ whose MNSD is $S(c) > 2^{p-1} + 2^{q-1}$ with $q = \{1, 2, \ldots, (p-1)\}$ can be obtained by using a graph with $p$ $A$-operations only if at least $q$ of these operations are articulation points.
**Proof:** Consider that the overall graph is a CM-based graph with \( l \) articulation points, where \( l < p \). The highest MNSD of any OF in the overall graph is \( n = 2^l \times (2^{p-1}+1) = 2^{p-1}+2^l \) (From Theorem 1). Therefore, \( l \) has to be at least equal to \( q \) in order to achieve \( n > 2^{p-1}+2^q \). Note that Theorem 1 corresponds to the case \( q = 1 \). 

**Theorem 3:** A constant \( c \) whose MNSD is \( S(c) > 0.75 \times 2^p+1 \) can not be obtained by using a Non-Multipliciative graph with only \( p+1 \) \( \mathcal{A} \)-operations and \( \mathcal{A} \)-depth preserved equal to \( p \).

**Proof:** For the simplest case \( p=2 \), there is only one graph with \( p+1=3 \) \( \mathcal{A} \)-operations and \( \mathcal{A} \)-depth preserved equal to \( p=2 \) (see Appendix A of [16]). The highest MNSD of the OF in this graph is \( 0.75 \times 2^p+1 = 4 \). Now we review the case \( p \geq 3 \). Since the \( \mathcal{A} \)-depth must be preserved equal to \( p \), only \( p \) of these operations can be sequentially-connected. Consider that a graph \( G_1 \) with \( \mathcal{A} \)-cost and \( \mathcal{A} \)-depth equal to \( p-1 \) is connected to one of the inputs of the last \( \mathcal{A} \)-operation. In order to avoid an articulation point, the other input must come from a different position. Thus, this input is connected to the remaining \( \mathcal{A} \)-operation, which should be placed in the \((p-1)\)-th depth level to obtain the highest possible MNSD (from Theorem B), and whose inputs are identified as \( a_1 \) and \( a_2 \). Consider that \( a_1 \) comes from the \( \mathcal{A} \)-operation placed in the \((p-2)\)-th depth level. Let us review the cases when \( G_1 \) is either non-multiplicative or multiplicative, along with the point where \( a_2 \) can come from.

If \( G_1 \) is non-multiplicative, the highest MNSD of its OFs is \( n_1 = 2^{p-2}+1 \) (from Theorem 1) and \( a_1 \) with \( a_2 \) can come from the \( \mathcal{A} \)-operation placed in the \((p-2)\)-th level. Thus, the highest possible MNSD of the OF in the last \( \mathcal{A} \)-operation is...
\[ n = n_1 + n_{a_1} + n_{a_2} \] where \( n_1 \) and \( n_{a_2} \) are the respective highest MNSD in \( a_1 \) and \( a_2 \).

The highest MNSD of \( G_1 \) is \( n_1 = 2^{p-2} + 1 \) and \( n_{a_1} = n_{a_2} = 2^{p-2} \). Therefore we have \( n = (2^{p-2} + 1) + 2^{p-2} + 2^{p-2} = 2^{p-1} + 2^{p-1} = 0.75 \times 2^{p-1} \).

If \( G_1 \) is multiplicative, consider that it is a CM-based graph with \( l \) articulation points, where \( 1 \leq l \leq p-2 \). The highest MNSD of its OFs is \( n_1 = 2^{l-1} \times (2^{p-l-2} + 1) \). Similarly, the highest MNSD in \( a_1 \) is \( n_{a_1} = 2^{l-1} \times (2^{p-l-2} + 1) \). To avoid an articulation point, \( a_2 \) must come from a depth level that does not exceed \( p-l-2 \), thus \( n_{a_2} = 2^{p-l-2} \). The highest possible MNSD of the OFs in the last \( A \)-operation is \( n = n_1 + n_{a_1} + n_{a_2} = 2^{p-2} + 2^{l-1} + 2^{p-l-2} \), which can be shown to be less or equal to \( 0.75 \times 2^{p-1} \) (see Appendix A.3).

**Theorem 4:** A constant \( c \) whose MNSD is \( S(c) > 0.75 \times 2^{p+2} \) with \( q \in \{1, 2, \ldots, p-2\} \) can be obtained by using a Non-Multiplicative graph whose \( A \)-depth is equal to \( p \) only if at least \( p+q+1 \) \( A \)-operations are used.

**Proof:** Consider \( p+r \) \( A \)-operations, with \( r < p \). Since the \( A \)-depth is equal to \( p \), only \( p \) of these operations can be sequentially-connected. Let us assume that a graph \( G_1 \) with \( A \)-cost and \( A \)-depth equal to \( p-1 \) is connected to one of the inputs of the last \( A \)-operation, whereas the other input comes from a different position to avoid an articulation point. Consider that this input comes from a graph \( G_2 \), formed with \( r-1 \) of the remaining \( A \)-operations. To obtain the highest MNSD in \( G_2 \), we can consider that \( G_2 \) is a CM graph with its last \( A \)-operation placed at the \((p-1)\)-th depth level. Let us assume that the last of the remaining \( A \)-operations is connected to the input of \( G_2 \), with its inputs being \( a_1 \) and \( a_2 \). The highest MNSD of any OF from \( G_2 \) is \( n_2 = 2^{l-1} \times (n_{a_1} + n_{a_2}) \), where \( n_{a_1} \) and \( n_{a_2} \) are the respective highest MNSD in \( a_1 \) and \( a_2 \).
If \( G_1 \) is non-multiplicative, the highest MNSD of its OFs is \( n_1 = 2^{p-2} + 1 \) (from Theorem 1). Both inputs \( a_1 \) and \( a_2 \) can come from the same point because \( G_1 \) is non-multiplicative, and the highest depth level where this point can be placed is \( p-r-1 \), thus \( n_{a_1} = n_{a_2} = 2^{p-r-1} \). Hence, the highest MNSD of any OF in the overall graph is \( n = n_1 + n_2 = (2^{p-2}+1) + 2^{r-1} \times (n_{a_1}+n_{a_2}) = (2^{p-2}+1) + 2^{r-1} \times (2^{p-r}) = (2^{p-2} + 1) + 2^{r-1} = 0.75 \times 2^p + 1 \), which clearly is less or equal than 0.75\( \times 2^{p+2r-1} \) even for the smallest \( q = 1 \).

If \( G_1 \) is multiplicative, consider that it is a CM-based graph with \( l \) articulation points, where \( 1 \leq l \leq p-2 \). The highest MNSD of its OFs is \( n_1 = 2^l \times (2^{p-l-2}+1) \). We can have two cases, depending on whether \( r \) is greater than \( l \) or not. Let us see each one of these cases.

When \( r \leq l \), only either \( a_1 \) or \( a_2 \) can come from the \( \mathcal{A} \)-operation placed in the \((p-l-1)\)-th depth level but not both, because an articulation point would be generated. If \( a_1 \) comes from this \( \mathcal{A} \)-operation, \( a_2 \) can come from the immediate lower depth level. Thus, \( n_{a_1} = (2^{p-l-2}+1) \) and \( n_{a_2} = 2^{p-l-2} \). Hence, the highest MNSD of any OF in the overall graph is \( n = n_1 + n_2 = (2^{p-2}+2^l) + 2^{r-1} \times (n_{a_1}+n_{a_2}) = (2^{p-2} + 2^l) + 2^{r-1} \times [(2^{p-l-2}+1)+2^{p-l-2}] = 2^{p-2}+2^l+2^{r-1} \times (2^{p-l-1}+1) \), which can be shown to be less or equal to \( 0.75 \times 2^p + 2^{r-1} \) (see Appendix A.4). Thus, \( r \) must be at least equal to \( q+1 \) in order to achieve a value \( n > 0.75 \times 2^p + 2^{r-1} \).

When \( r > l \), \( a_1 \) and \( a_2 \) can come from the same point because the subgraph \( H \) in \( G_1 \) is non-multiplicative, thus \( n_{a_1} = n_{a_2} = 2^{p-r-1} \). Hence, the highest MNSD of any OF in the overall graph is \( n = n_1 + n_2 = (2^{p-2}+2^l) + 2^{r-1} \times (n_{a_1}+n_{a_2}) = (2^{p-2}+2^l) + 2^{r-1} \times (2^{p-r}) = 2^{p-2}+2^l+2^{r-1} = 0.75 \times 2^p + 2^l \). Since \( l \) is at most equal to \( r-1 \), we have that the highest value for \( n \) is \( n = 0.75 \times 2^p + 2^{r-1} \). Thus, \( n > 0.75 \times 2^p + 2^{r-1} \) holds only if \( r \) is at least equal to \( q+1 \). ■
Now, let us introduce the number of prime factors of the constant \( c \), denoted as \( \Omega(c) \). Note that, along with \( S(c) \), \( \Omega(c) \) is a function that can be known a priori from the constants. Generally speaking, obtaining the prime factors of \( c \) is a challenging problem for very large constants (over 110 digits) and it is one of the bases for the Rivest-Shamir-Adleman (RSA) encryption scheme [27]. However, for constants up to 32 bits, which cover the DSP problem sizes of the most practical importance [14], the prime factors of \( c \) can be obtained straightforwardly even with the simple sieve approach, such as the one used in the MATLAB function \texttt{factor}.

The following theorems present the relations between the values \( \Omega(c) \) and \( S(c) \). If the required multiplicative characteristics for the corresponding graph of the constant \( c \), highlighted in the previous theorems, can not be accomplished due to the value \( \Omega(c) \), the lower bounds for the \( \mathcal{A} \)-cost and the \( \mathcal{A} \)-depth are affected.

\textbf{Theorem 5:} A constant \( c \) whose MNSD is \( S(c) > 2^{p-1} + 2^{\Omega(c)-1} \) only can be obtained by using a graph with at least \( p+1 \) \( \mathcal{A} \)-operations.

\textbf{Proof:} From Theorem 2 it is known that at least \( \Omega(c) \) articulation points are required if only \( p \) \( \mathcal{A} \)-operations can be used. However, since \( \Omega(c) \) is the number of prime factors, only up to \( \Omega(c)-1 \) articulation points are allowed. Thus, the only solution is using an additional \( \mathcal{A} \)-operation. By adding one more \( \mathcal{A} \)-operation, the overall Non-Multiplicative graph can give an OF whose MNSD is up to \( 2^{p+1} \), which covers the MNSD-range regardless of the value \( \Omega(c) \). Therefore, at least \( p+1 \) \( \mathcal{A} \)-operations are required. \( \blacksquare \)
**Theorem 6:** A constant $c$ whose MNSD is $S(c) > 0.75 \times 2^p + 2^{Q(c)q-2}$ with $q = 1, 2, \ldots, (p - \Omega(c) - 1)$ only can be obtained by using a graph with $\mathcal{A}$-depth at least equal to $p+1$ if up to $p+q$ $\mathcal{A}$-operations are used, or with at least $p+q+1$ $\mathcal{A}$-operations if the minimum $\mathcal{A}$-depth equal to $p$ is preserved.

**Proof:** Let us consider a CM-based graph with $p+r$ $\mathcal{A}$-operations and $\mathcal{A}$-depth equal to $p+s$, where $r < p$ and $s \geq 0$. Since at most $\Omega(c)$-1 articulation points can be used, its CM subgraph has $\Omega(c)$-1 $\mathcal{A}$-operations, whereas its non-multiplicative subgraph $H$ has $p+r-\Omega(c)+1$ $\mathcal{A}$-operations and an $\mathcal{A}$-depth equal to $p+s-\Omega(c)+1$. According to Theorem 4, the highest MNSD of any OF generated in the subgraph $H$ is $0.75 \times 2^p + 2^{r-s-1}$ and the highest MNSD at the output of the overall graph is $n = 2^{\Omega(c)-1}(0.75 \times 2^{p+\Omega(c)-1} + 2^{r-1}) = 0.75 \times 2^{p+s} + 2^{\Omega(c)p+r-s-2}$. If the $\mathcal{A}$-depth is preserved equal to $p$, we have $s=0$. In such case $r$ needs to be at least equal to $q+1$ in order to achieve $n > 0.75 \times 2^p + 2^{Q(c)q-2}$. On the other hand, if $r \leq q$ holds, the value $s$ should be at least equal to 1 in order to achieve $n > 0.75 \times 2^p + 2^{Q(c)q-2}$, implying that the $\mathcal{A}$-depth of the overall graph is at least equal to $p+1$.

From Theorem 5 and substituting $p$ using (5.3), the lower bound $L_A$ can be expressed as

$$L_A = \begin{cases} \lceil \log_2 S(c) \rceil; & \text{if } \Omega(c) \geq \log_2 \left\{ S(c) - 2^{\lceil \log_2 S(c) \rceil - 1} \right\} + 1, \\ \lceil \log_2 S(c) \rceil + 1; & \text{otherwise.} \end{cases} \quad (5.4)$$

Note that if $L_A = \lceil \log_2 S(c) \rceil$ holds, $L_A$ will have the same value as $L_A$ without requiring any additional $\mathcal{A}$-operation (as given by (5.2)). From (5.4), we can see that this only can be accomplished if $\Omega(c) \geq \log_2 \left\{ S(c) - 2^{\lceil \log_2 S(c) \rceil - 1} \right\} + 1$ holds.
The argument \( S(c) - 2^{\lfloor \log_2 S(c) \rfloor - 1} \) in the \( \log_2 \{ \} \) operation is always greater than zero and therefore the condition can always be evaluated.

From Theorem 6 we have that if the \( \mathcal{A} \)-cost is given as \( N_A \geq \lfloor \log_2 S(c) \rfloor + q + 1 \) and if \( \log_2 \{ S(c) - 0.75 \times 2^{\lfloor \log_2 S(c) \rfloor} \} + 2 - \Omega(c) > q \) holds, the \( \mathcal{A} \)-depth can be kept equal to \( \lfloor \log_2 S(c) \rfloor \). By noting that any real number \( a \) is greater than an integer \( b \) if this integer is given as \( b = \lfloor a \rfloor - 1 \) and applying this observation to \( q \), we obtain

\[
q = \lfloor \log_2 \{ S(c) - 0.75 \times 2^{\lfloor \log_2 S(c) \rfloor} \} + 2 - \Omega(c) \rfloor - 1.
\]

In this equality, the \( \log_2 \{ \} \) operation does not yield finite values when \( S(c) \leq 0.75 \times 2^{\lfloor \log_2 S(c) \rfloor} \) holds. However, according to Theorem 3, the \( \mathcal{A} \)-depth can be kept equal to \( \lfloor \log_2 S(c) \rfloor \) by using only one additional \( \mathcal{A} \)-operation if the condition \( S(c) \leq 0.75 \times 2^{\lfloor \log_2 S(c) \rfloor} + 1 \) holds. Therefore,

\[
L_d = \begin{cases} 
\lfloor \log_2 S(c) \rfloor; & \text{if } N_A \geq \lfloor \log_2 S(c) \rfloor + L_{EA}, \\
\lfloor \log_2 S(c) \rfloor + 1; & \text{otherwise}, 
\end{cases}
\]

where \( L_{EA} \), the lower bound for the number of extra \( \mathcal{A} \)-operations required to generate NEFs and preserve \( L_d = \lfloor \log_2 S(c) \rfloor \), is given as

\[
L_{EA} = \begin{cases} 
0; & \text{if } \Omega(c) \geq \log_2 \{ S(c) - 2^{\lfloor \log_2 S(c) \rfloor - 1} \} + 1, \\
1; & \text{if } S(c) \leq 0.75 \times 2^{\lfloor \log_2 S(c) \rfloor} + 1, \\
\lfloor \log_2 \{ S(c) - 0.75 \times 2^{\lfloor \log_2 S(c) \rfloor} \} + 2 - \Omega(c) \rfloor; & \text{otherwise}.
\end{cases}
\]  

It is worth highlighting that, whereas the lower bound \( L_A \) only depends on the relation between \( \Omega(c) \) and \( S(c) \), the lower bound \( L_d \) depends on the \( \mathcal{A} \)-cost \( N_A \). As pointed out in [17], for many problem instances there is a tradeoff between the \( \mathcal{A} \)-cost and the \( \mathcal{A} \)-depth. From (5.5), we have that such tradeoff is the minimum theoretical value \( L_{EA} \), given in (5.6).
5.2.4 Comparison of proposed and current lower bounds

Let us review a simple example to illustrate how the proposed lower bounds given in (5.4) and (5.5) are more precise with respect to the lower bounds from [17], expressed in (5.2). To this end, consider the 14-bit constant 11467, which is known to have an optimal $\mathcal{A}$-cost that does not exceed 4, as mentioned in [18], and whose CSD representation is $10\overline{0}1\overline{0}0\overline{1}0\overline{1}0\overline{1}\overline{1}$. Clearly, $S(11467)=8$ and from (5.2) we have $L_\mathcal{A}=L_d=3$. However, we also have $\Omega(11467)=1$, i.e., 11467 is a prime number. Using $S(11467)$ and $\Omega(11467)$, the new lower bound for the $\mathcal{A}$-cost of this constant is $L_\mathcal{A}=4$, according to (5.4). This means that the solution with $\mathcal{A}$-cost equal to 4, based on the graph No. 10 of Appendix A of [16], is in fact optimal in terms of the $\mathcal{A}$-cost. Moreover, from (5.5) and (5.6) we have that the lower bound $L_d=3$ can be preserved if at least $L_{EA}=2$ extra $\mathcal{A}$-operations are used in addition to the 3 $\mathcal{A}$-operations dictated by the bound $L_d$, i.e., at least 5 $\mathcal{A}$-operations are required to preserve $L_d=3$. It is clear that the proposed lower bounds are more informative in comparison to the lower bounds from [17]. Similar examples arise with constants 11093 and 13003, among others.

Finally, Table 5.1 shows the percentage of constants with improved lower bounds among all the 14-bit odd integer constants and among 10000 randomly generated constants that can be expressed with $B$-bits, where $14 < B \leq 32$.

<table>
<thead>
<tr>
<th>Word-length</th>
<th>Total</th>
<th>$L_{EA} = 1$</th>
<th>$L_{EA} = 2$</th>
<th>$L_{EA} = 3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>14-bits</td>
<td>19.6%</td>
<td>19.4%</td>
<td>0.2%</td>
<td>0%</td>
</tr>
<tr>
<td>(14 &lt; $B$ &lt; 32)-bits</td>
<td>33.36%</td>
<td>32.08%</td>
<td>1.14%</td>
<td>0.14%</td>
</tr>
</tbody>
</table>

Table 5.1: Percentage of constants with improved lower bounds
5.3 Optimization of high-speed pipelining in FPGA-based FIR filters

For FPGAs we need to consider the underlying logic block architecture that is used to implement the digital filter. The elementary Logic Blocks (LBs), sometimes called Logic Elements (LEs) by Altera, Logic Cells (LCs) by Actel or Configurable Logic Blocks (CLBs) by Xilinx, in the majority of devices used today have typically a 4 or 5 inputs Look-Up Table (LUT) and one or two outputs and a flip-flop [18]. In Xilinx devices, usually a couple of CLBs is called a Slice. Figure 5.3 show a simplified example of a LB.

![Figure 5.3: A simplified example of a Logic Block (LB) in an FPGA.](image)

In many constant-coefficient filter designs the task of efficient pipelining arises in order to enhance the speed and decrease the power consumption. From an optimal implementation standpoint, the filter should consume a minimum number of LBs and run at high speed. Distributed Arithmetic (DA) has been the most common implementation strategy in digital filter designs by vendors such as Xilinx or Altera. The main advantages of DA-based systems are that the required resource only depends on the number of
coefficient and bit width; no adder graph is built whose size depends on the coefficient distribution. The DA-based designs are easily pipelined and no coefficient computational optimization is required, just LUT generation is needed, i.e., the core generators for DA-based systems are very compact and fast.

From Figure 5.3 can be concluded that an adder, a register or an adder combined with a register will all incur the same implementation cost, i.e., number of LBs. Based on this observation, a promising approach recently investigated to implement multiplication by constants in FPGA devices has been derived from the algorithms that solve MCM problems, with the difference that pipeline registers are used after each addition/subtraction operation [20]-[26]. As has been shown in [20]-[26], MCM-based filter designs have lower area than DA-based filters and the recent efforts are focused on improving the Area-Time (AT) performance (decreasing the Area×Time product or AT cost) of MCM-based filters by using pipeline principles.

In subsection 5.3.1 we define the pipelined operations as either pipelined additions/subtractions or only pipeline registers and introduce the Fully-Pipelined (FP) SCM and MCM problems. In subsection 5.3.2 is shown by means of key observations that the lower bound for the number of pipelined operations in a FP-SCM block is dependent on the $A$-depth of the graph multiplier. Taking this characteristic as a starting point, we introduce in subsection 5.3.3 an algorithm to reduce the number of logic blocks by minimizing the number of adders subject to the theoretical lower bound for the adder $A$-depth.
5.3.1 Pipelining in SCM and MCM blocks

The difference between a usual MCM design and a pipelined MCM can be best understood by introducing the following example.

**Example 1:** Consider the design of a multiplier by the constant 25.

In a typical MCM algorithm such as \( n \)-dimensional Reduced Adder Graph (RAG-\( n \)) [2] or its variants like Cumulative-benefit Heuristic (Hcub) [7], Modified RAG (MRAG) [5] or RAG-05 [21] it would be preferred a design using the equation \( 25 = 3 \times 8 + 1 \) as shown in Figure 5.4a. Another alternative is shown in Figure 5.4b for the same coefficient. In this case, 25 is implemented via \( 25 = 5 \times 5 \). RAG-\( n \) and its variants would pick the solution shown in Figure 5.4 based on the following two reasons: first, fewer resources are needed to build the NOF 3 in comparison with the NOF 5 and second, a small NOF (or sum of NOFs for more complicated coefficients) is preferred because it has the potential to generate more fundamentals than a larger NOF. However, these rules change if pipelined designs are considered. In order to preserve the I/O behavior all inputs to every adder in the graph need to have the same delay with regard to the input sequence \( x(n) \). For both NOFs (3 and 5) no additional resources are required, since the LB’s Flip-Flop (FF) are now utilized not incurring any additional LBs. However, in order to guaranty the same delay of the output for the solution \( 25 = 3 \times 8 + 1 \), one additional pipeline register is required, which results in an overall higher LB cost than the \( 25 = 5 \times 5 \) design, as corroborated by the synthesis results shown in Table 5.2. The better choice in the pipelined design is now using the NOF 5 and not the NOF 3.
Figure 5.4: Fully Pipelined Reduced Adder Graphs (FP-RAGs) for the implementation of a coefficient multiplier 25: (a) Solution with the form $25=3\times8+1$, (b) Solution with the form $25=5\times5$.

Table 5.2: Synthesis results for the implementation of the constant multiplier 25 on a Xilinx Virtex-4 XC4VSX25-10FF680

<table>
<thead>
<tr>
<th>Type</th>
<th>Pipelining</th>
<th>No. of FFs</th>
<th>No. of 4-input LUTs</th>
<th>No. of Slices</th>
<th>$F_{\text{max}}$ (MHz)</th>
<th>AT cost (Slices / $F_{\text{max}}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$25=3\times8+1$</td>
<td>No</td>
<td>29</td>
<td>26</td>
<td>22</td>
<td>254.45</td>
<td>0.086</td>
</tr>
<tr>
<td>$25=5\times5$</td>
<td>No</td>
<td>29</td>
<td>27</td>
<td>22</td>
<td>250.88</td>
<td>0.088</td>
</tr>
<tr>
<td>$25=3\times8+1$</td>
<td>Yes</td>
<td>52</td>
<td>29</td>
<td>27</td>
<td>411.18</td>
<td>0.066</td>
</tr>
<tr>
<td>$25=5\times5$</td>
<td>Yes</td>
<td>44</td>
<td>28</td>
<td>23</td>
<td>411.86</td>
<td>0.056</td>
</tr>
</tbody>
</table>

Two important observations can be made from the previous example:
- The number of additional pipeline registers is important in minimizing the required LBs; small NOF (or NOF sum) or counting the required full-adders in the design is less useful optimization goal.
• Each adder needs to be pipelined; otherwise a high degradation of the overall performance may occur as shown in the $F_{\text{max}}$ column in Table 5.2.

The aforementioned observations are a motivation to formally describe the Fully-Pipelined Single Constant Multiplication (FP-SCM) block. Thus, consider the following definitions.

**Definition 1: Delayed constants**

Given a constant $c$, its delayed-by-$l$ version is denoted by $c^{(l)}$, with

$$c^{(l)} = c \cdot z^{-l},$$  \hspace{1cm} (5.7)

where $z^{-l}$ is a delay by $l$ samples and $l$ can be any positive integer including 0. As an example, $7^{(4)} = 7z^{-4}$.

**Definition 2: $\Phi$-Operation**

Let $e_1, e_2 \geq 0$ be integers (left shifts), $e_3 \geq 0$ be an integer (right shift), $e_4 \in \{0, 1\}$ (Addition/subtraction selector) and $e_5 \in \{0, 1\}$ (addition-delay/only-delay selector). A $\Phi$-operation is an operation with either one or two integer equally-delayed inputs $u^{(l)}$ and $v^{(l)}$ and one output defined as

$$w^{(l+1)} = \Phi(y)(u^{(l)}, v^{(l)}) = e_5 \cdot (2^{e_1} u^{(l)}) + (-1)^{e_4} (2^{e_2} v^{(l)}) \cdot 2^{-e_3} \times z^{-1},$$  \hspace{1cm} (5.8)

where $z^{-1}$ is one sample-delay and $y = (e_1, e_2, e_3, e_4, e_5)$ is the parameter set or $\Phi$-configuration of $\Phi$. We consider the $\Phi$-operation as an abstraction of either a pipelined addition (or a pipelined subtraction) or only a pipeline register. In other words, any value delayed by one is also a $\Phi$-operation and it is considered to have the same implementation cost as a pipelined addition/subtraction. Figure 5.5 details this operation.
The FP-SCM block can be designed as a network of shifts and $\mathcal{D}$-operations represented also using DAGs with the same characteristics as the ones described in subsection 5.2.1 (considering the $\mathcal{A}$-operations as two-input $\mathcal{D}$-operations). According to the definition of a $\mathcal{D}$-operation, one additional characteristic must be considered, namely, that the $\mathcal{D}$-operations with two inputs must have equally-delayed inputs. This means that $\mathcal{D}$-operations with only one input must be added to balance any unequally-delayed input.

Now, let us find $L_D$, the lower bound for the minimum number of $\mathcal{D}$-operations of a SCM graph, in terms of the following key observations.

**Observation 1:** Consider two subgraphs, $A$ and $B$, with the output of $A$ coming from a 2-input $\mathcal{D}$-operation, $B$ having $r \geq 2$ inputs that must be equally-delayed and one of these inputs coming from the output of $A$. An articulation point is the only way to connect these two subgraphs without additional $\mathcal{D}$-operations.

**Motivation:** Let us call “first input” to the input of the subgraph $B$ that is obtained from the output of the subgraph $A$. Each of the $(r - 1)$ remaining inputs of $B$ can be obtained from two possibilities: $a$) from the output of $A$ or, $b$) from any other operation in $A$ placed before the output $\mathcal{D}$-operation of the
subgraph $A$. If all the remaining inputs of the subgraph $B$ are in case $a$, the connection between $A$ and $B$ becomes an articulation point and the output of the subgraph $A$ can emerge as $r$ equally-delayed inputs to the subgraph $B$. On the other hand, for every of the $(r - 1)$ remaining inputs in case $b$, the articulation point is eliminated but at least an additional $D$-operation is required to balance the delay of the last $D$-operation of $A$ and to make equally-delayed inputs for $B$.

**Observation 2:** The lower bound $L_D$ is equal to $L_A$ in the graph of a constant $c$ provided that the constant is implemented using the Completely Multiplicative (CM) graph (see Figure 5.1).

**Motivation:** Any SCM block with $N_A$ number of $A$-operations requires at least $N_A$ number of $D$-operations. According to Observation 1, no additional $D$-operations in the form of only-delay are required because all the $A$-operations in the CM graph are articulation points. Therefore, the lower bound for the number of $D$-operations is $L_D = L_A$.

**Observation 3:** Any Non-Multiplicative graph with $A$-depth $N_d$ requires at least $N_D = (2N_d - 1)$ $D$-operations.

**Motivation:** According to Observation 2, if a graph with $N_d$ sequentially connected $D$-operations has only $N_d$ $D$-operations, it must be a CM graph. Since the graph must be Non-Multiplicative, the $(N_d - 1)$ articulation points of the CM graph must be eliminated. From Observation 1, it is known that at least one additional $D$-operation must be added for every eliminated articulation point. Therefore, at least $(2N_d - 1)$ $D$-operations are required, i.e.,
the original \( N_d \) minimum number of \( D \)-operations in the form of addition-delay plus the additional \( (N_d - 1) D \)-operations.

Recall from section 5.2.2 that the highest Minimum Number of Signed Digits (MNSD) in the MNSD-range (defined below (5.3)) can be obtained with the minimum number of \( A \)-operations only using a CM graph. From Observation 2, the same holds for the pipelined case. In general, the multiplicative characteristic of the graphs allows the maximum increase of the NMSD because in every articulation point the next \( A \)-operation (or 2-input \( D \)-operation) has its inputs coming from the same point (see Observation B). However, from Theorem 5 (subsection 5.2.3) we have that only \( \Omega(c) D \)-operations (\( \Omega(c) \) is the number of prime factors of \( c \)) can be arranged in CM structure because the number of articulation points of the graph for a given constant can be, at most, \( \Omega(c) - 1 \). Thus, the cases where the CM graph can be used for a given constant \( c \) are when the functions \( \Omega(c) \) and \( S(c) \) accomplish the relation \( \Omega(c) \geq \left\lceil \log_2 S(c) \right\rceil \). In such cases, we have \( L_D = L_A = \left\lceil \log_2 S(c) \right\rceil \).

If the CM graph is not used, \( L_D \) is no longer equal to \( L_A \). In such cases, the lower bound \( L_D \) can be increased. These cases can be identified when the relation \( \Omega(c) < \left\lceil \log_2 S(c) \right\rceil \) is accomplished. Nevertheless, it can be still taken advantage of the multiplicative characteristic as much as possible. Observation 3 gives the lower bound \( L_D \) when \( \Omega(c) = 1 \). On the other hand, if \( \Omega(c) = 2 \), only one \( D \)-operation can be used as articulation point and a Non-Multiplicative subgraph must be connected in its output. Similarly for a general case, at most \( \Omega(c) - 1 \) \( D \)-operations can be arranged as a CM subgraph.
having $\Omega(c) - 2$ articulation points. The other articulation point is formed with the output of the CM subgraph connected to the input of a Non-Multiplicative subgraph. The lower bound $L_D$ can be obtained as the number of $\Phi$-operations arranged in the CM subgraph, $\Omega(c) - 1$, plus the number of $\Phi$-operations in the Non-Multiplicative subgraph, $2[N_d - (\Omega(c) - 1)] - 1$. The value $[N_d - (\Omega(c) - 1)]$ is the $A$-depth of the Non-Multiplicative subgraph and $N_d$ is the $A$-depth of the overall graph. Hence, the lower bound $L_D$ is given as $L_D = 2N_d - \Omega(c)$ when $\Omega(c) < \lceil \log_2 S(c) \rceil$. Thus, we have

$$L_D = \begin{cases} 
2N_d - \Omega(c); & \Omega(c) < \lceil \log_2 S(c) \rceil, \\
\lceil \log_2 S(c) \rceil; & \Omega(c) \geq \lceil \log_2 S(c) \rceil.
\end{cases}$$

(5.9)

Consider the following characteristics of $L_D$ for the case $\Omega(c) < \lceil \log_2 S(c) \rceil$:

- The lower bound $L_D$ is directly proportional to the $A$-depth $N_d$. It achieves its minimum value when $N_d = L_d = \lceil \log_2 S(c) \rceil$, with $L_d$ given in (5.5), regardless of the fact that in that case at least $L_{EA}$ extra $\Phi$-operations must be included, with $L_{EA}$ given in (5.6).

- Even though a graph with minimum $A$-depth given as $L_d = \lceil \log_2 S(c) \rceil + 1$ in (5.5) can avoid the $L_{EA}$ additional 2-input $\Phi$-operations, it requires more 1-input $\Phi$-operations and it has a higher value for $L_D$. In other words, the minimum number of additional 2-input $\Phi$-operations, $L_{EA}$, does not increase the lower bound $L_D$.

For a set of $N$ constants, $T = \{t_1, t_2, \ldots, t_N\}$, a MCM block can be considered as the SCM block of the constant with the highest MNSD, plus several NOFs needed to form the other constants and having several OFs instead of only
one. For the case of a Fully-Pipelined MCM (FP-MCM) block, all of these OFs must have the same delay.

The minimum $\mathcal{A}$-depth of a MCM (or a FP-MCM) block, $L_{d,MCM}$, is the lower bound $L_d$ of the constant with the highest MNSD, as shown in [17]. Using (5.5) we obtain

$$L_{d,MCM} = \begin{cases} \max_i \left\lceil \log_2 S(t_i) \right\rceil; & \text{if } N_{A,i} \geq \left\lceil \log_2 S(t_i) \right\rceil + L_{EA,i}, \\ \max_i \left\lceil \log_2 S(t_i) \right\rceil + 1; & \text{otherwise}, \end{cases} \quad (5.10)$$

where $1 \leq k \leq N$, $N_{A,i}$ being the number of $\mathcal{A}$-operations (or 2-input $\mathcal{D}$-operations in the FP-MCM case) required for the $i$-th constant (the constant with the highest MNSD) and $L_{EA,i}$ obtained from (5.6) upon replacing $c$ by $t_i$. From the previous characteristics for the lower bound $L_D$ of a FP-SCM block we can derive the following observations:

- Preserving $L_{d,MCM} = \max_i \left\lceil \log_2 S(t_i) \right\rceil$ guarantees the minimum value for $L_D$ corresponding to the constant with the highest MNSD according to (5.9), and implies adding at least $L_{EA,i}$ additional 2-input $\mathcal{D}$-operations according to (5.10). These operations can generate more Non-Output Fundamentals (NOFs), which can be useful to form more constants.

- Preserving $L_{d,MCM} = \max_i \left\lceil \log_2 S(t_i) \right\rceil$ allows having all the individual $\mathcal{A}$-depths of the OFs similar to each other (the absolute difference among them is small). Therefore, few pipeline registers can be required to make all the OFs having the same pipeline stages.

Even though the aforementioned discussion does not provide a formal proof, it presents a strong reasoning that justifies the use of algorithms that have the constraint of obtaining a graph subject to the theoretical lower bound for the
A-depth, $L_{d,MCM} = \max_i \left\lceil \log_2 S(t_i) \right\rceil$, to solve the FP-MCM problem.

### 5.3.2 Proposed method

Let us define the FP-MCM problem in the following, taking as a basis the definitions 1 and 2 introduced in the previous subsection.

**Definition 3: Set of delayed constants**

Given a set of $m$ not necessarily different constants $C = \{c_1, c_2, \ldots, c_m\}$, we define a set of delayed constants as

$$\tilde{C} = \{c_1^{(l_1)}, c_2^{(l_2)}, \ldots, c_m^{(l_m)}\},$$

where $l_1, l_2, \ldots, l_m$ can be positive integers including 0, with $l_i \leq l_j \forall i \leq j$. As an example, for the graph of Figure 5.4a we have the sets of constants $C_1 = \{1, 1, 3, 25\}$ and $\tilde{C}_1 = \{1, 1^{(1)}, 3^{(1)}, 25^{(2)}\}$. Similarly, for the graph of Figure 5.4b we have the sets of constants $C_2 = \{1, 5, 25\}$ and $\tilde{C}_2 = \{1, 5^{(1)}, 25^{(2)}\}$.

**Definition 4: The Fully-Pipelined MCM problem**

Given a set of $N$ positive and odd target constants $T = \{t_1, t_2, \ldots, t_N\} \subset \mathbb{N}$, find the smallest set of delayed constants $\tilde{C} = \{c_0^{(l_0)}, c_1^{(l_1)}, \ldots, c_m^{(l_m)}\}$ and its corresponding non-delayed version $C = \{c_0, c_1, \ldots, c_m\}$, with $T \subset C$ such that:

a) $c_0^{(l_0)} = 1$ (i.e., $c_0 = 1$ and $l_0 = 0$);

b) for all $c_i^{(l_i)}$ with $1 \leq k \leq m$ there exist $c_i^{(l_i)}$, $c_j^{(l_j)}$, with $0 \leq i, j < k$, and a $D$-configuration $y_k$, such that $c_k^{(l_k)} = D_{y_k}(c_i^{(l_i)}, c_j^{(l_j)})$, with $l_i = l_j$;

c) $l_i = l_j$ for $(m - N + 1) \leq i, j \leq m$.

The FP-SCM problem is a special case with $N = 1$. 

The delayed set $\tilde{C}$ contains all the NOFs required to form the FP-MCM block, with the explicit representation of the delay of every NOF. The first element in $\tilde{C}$ is always 1 and it incurs no cost, since it represents the input of the DAG. All the other elements have the cost of one $\mathcal{D}$-operation. The last $N$ elements are the equally-delayed OFs of the FP-MCM block. Therefore, the cost of the last $N$ elements in any solution $\tilde{C}$ is always the same. The set $C$ contains the same NOF values in $\tilde{C}$ but without the explicit representation of the delays.

Besides of the trivial fact that in the traditional MCM problem there is no definition for a delayed set, the following are the two main differences with respect to the non-pipelined MCM problem given in [7]:

- The solution set $C$ allows repeated NOFs in the FP-MCM problem because of pipelining, whereas there are no repeated NOFs in the MCM problem.
- Excluding the first element equal to 1 in $C$ which comes free, the solution set $C$ may include more NOF values 1, since they are delayed inputs and have a cost of a one-input $\mathcal{D}$-operation. For example, for the graph of Figure 5.4a we have $C_1 = \{1, 1, 3, 25\}$, where the second 1 is a NOF that represents the delayed input. There are no NOF equal to 1 in the traditional MCM problem.

The solution of the FP-MCM problem can be found by an exhaustive exploration of all possible $\mathcal{D}$-configurations $y_k = (e_1, e_2, e_3, e_4, e_5)$, with $1 \leq k \leq m$ and $e_{j,\min} \leq e_j \leq e_{j,\max}$ for $j = 1, 2, 3, 4$ and 5, where the $e_{j,\min}$ and $e_{j,\max}$ are the respective upper and lower limits for $e_j$, that form the target set $T$ with the
minimum \( m \). However, the following assumptions have been made in literature to reduce the search space:

- All NOFs are odd integers [5], therefore we can set \( e_2 = 0 \) in advance.
- Division by powers of two is prohibited because it requires more FPGA resources [18], therefore we can set \( e_3 = 0 \) in advance.

Note that, even though two parameters have been eliminated, the universe of solutions is still enormous and an exhaustive search is not recommended. In the following we introduce an efficient strategy to solve the FP-MCM problem. To this end, consider the following observations:

- The search of the optimal configurations \( y_k = (e_1, 0, 0, e_4, e_5) \) can be understood as the search of the optimal NOFs that can form the FP-MCM block. Therefore, the universe of solutions will consist of NOFs. The NOF value 1 must be included to consider cases where the input of the graph is being delayed (a 1-input \( \Phi \)-operation is being used).
- The previous subsection recommends that the FP-MCM block should have the minimum \( A \)-depth, which is equal to the lower bound of the \( A \)-depth of its coefficient with the highest MNSD, i.e.,

\[
L_{d,MCM} = \max_i \left\lceil \log_2 S(t_i) \right\rceil.
\]  

(5.12)

Therefore, the NOFs in the universe of solutions will have \( A \)-depths less than \( L_{d,MCM} \).
- We consider that the OFs that are formed with fewer 2-input \( \Phi \)-operations (simple OFs) should be placed at the end of the graph. This allows having more room to insert pure delays previous to these OFs and form the OFs with higher 2-input \( \Phi \)-operations (complex OFs) using less resources. If the simple OFs are formed first, their NOFs
may be not useful to form complex OFs and more 1-input D-operations will be required to skip these NOFs.

With these observations in mind, let us define the following vectors that will be used in the search algorithm:

- The vector \( \mathbf{u} = \begin{bmatrix} 1 & u_1 & u_2 & \ldots & u_{L_{d,MCM}-1} \end{bmatrix} \) (size \( 1 \times K \), with \( K \) being an arbitrary number depending on the problem) contains the universe of solutions, i.e., the NOFs that can be used for a given problem. This vector is composed by a element equal to 1 and \( L_{d,MCM} - 1 \) sub-vectors, namely, \( u_1, u_2, \) etc. (whose respective sizes are \( 1 \times K_1, 1 \times K_2, \) etc.) The elements of the sub-vector \( u_k \) are all the NOFs that can be constructed with \( k \) 2-input D-operations. The last element in every sub-vector is the NOF closest and greater to twice the largest coefficient in the target set \( T \).

- The vector \( \mathbf{s} = \begin{bmatrix} s_1 & s_2 & s_3 & \ldots & s_{L_{d,MCM}-1} \end{bmatrix} \) (row vector) has the same size as \( \mathbf{u} \) (the sub-vector \( s_k \) has the same size as the sub-vector \( u_k \), for \( k = 1, 2, \ldots, L_{d,MCM}-1 \)) and contains only binary entries. This vector identifies what of the NOFs in \( \mathbf{u} \) are used to form the FP-MCM block. Let us denote the \( k \)-th element of \( \mathbf{s} \) and \( \mathbf{u} \) as \( s_k \) and \( u_k \), respectively. We have \( s_k = 1 \) if \( u_k \) is a used NOF or \( s_k = 0 \) if \( u_k \) is not used.

The algorithm consists on finding the optimum vector \( \mathbf{s}^* \) which identifies the NOFs in \( \mathbf{u} \) that form the FP-MCM block and simultaneously result in the lowest cost of this block. Therefore, at this point we must define a cost function that expresses how costly a NOF is. The cost for a NOF \( u_k \) can be estimated via
where $B_{in}$ is the input word-length to the FP-MCM block. The cost for a given vector $s$ is the sum of the costs of all the used NOFs in $u$, identified by the entries equal to 1 in $s$. In the case that these NOFs do not generate all the desired coefficients, we set the cost equal to 1000 (or a sufficiently large value). Thus, we have the cost of a vector $s$ expressed as

$$F(s) = \begin{cases} 
1000 \text{ (or higher)}, & \text{if the NOFs do not} \\
& \text{form the desired constants;} \\
\sum_{k=1}^{K} s_k [B_{in} + \log_2(u_k)], & \text{otherwise.}
\end{cases}$$

(5.14)

Note that, to evaluate the cost of a given $s$, we need to form the FP-MCM block with the NOFs in $u$ identified by $s$. This can be done with the following steps:

1. Assign $j = 1$ and form the set $W_{task} = T$ ($T$ is the target set).
2. Form the set $W_{in} = \{ \text{all non-zero elements of the Hadamard product } \mathbf{s}_{\mathbf{u}} \circ \mathbf{u} \}$.
3. Form the set $W_{out}$ containing all the possible elements of $W_{task}$ that can be constructed using the elements of $W_{in}$ and only a 2-input $\mathcal{D}$-operation for each element. If no elements can be formed, it is not possible to form the FP-MCM block with the elements of $W_{in}$.
4. Remove the elements of $W_{task}$ that are already in $W_{out}$. Assign $j = j+1$ and repeat from step 2. Finish when $W_{task}$ is empty.

The following example is introduced to clarify the previous paragraphs.

**Example 2:** Obtain the vector $u$ corresponding to the target set $T = \{1, 3, 5, 25\}$, along with the costs for all the possible representations of $s$, assuming an
input word-length \( B_m = 12 \).

To obtain \( \mathbf{u} \) we need to know the minimum \( A \)-depth of the FP-MCM block, \( L_{d,MCM} \), which can be obtained from (5.12). Since the coefficient 1 does not require logic operators to be formed, we evaluate the constants 3, 5, 25 and 75 to obtain \( L_{d,MCM} \). For these constants we have

\[
S(3) = 2, \quad \lceil \log_2 S(3) \rceil = 1;
\]
\[
S(5) = 2, \quad \lceil \log_2 S(5) \rceil = 1;
\]
\[
S(25) = 3, \quad \lceil \log_2 S(25) \rceil = 2;
\]
\[
S(75) = 4, \quad \lceil \log_2 S(75) \rceil = 2.
\]

Thus, \( L_{d,MCM} = \max\{1, 1, 2, 2\} = 2 \). The vector of NOFs is \( \mathbf{u} = [1 \ \mathbf{u}_1] \). The vector \( \mathbf{u}_1 \) contains all the NOFs that can be constructed with one 2-input \( D \)-operations, with its last element being the NOF closest and greater to \( 2 \times 25 = 50 \). We obtain
\[
\mathbf{u}_1 = [1 \ 3 \ 5 \ 7 \ 9 \ 15 \ 17 \ 31 \ 33 \ 63].
\]

Since \( \mathbf{u}_1 \) has 10 elements, there are \( 2^{10} = 1024 \) possible combinations to form \( \mathbf{s} \). These combinations are presented in Figure 5.6, along with their corresponding costs. The search space in this case is small, thus we obtain the optimal solution, \( \mathbf{s}^* = [1 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0] \), by exhaustive search. The cost for this solution is found by using (5.14). We have \( F(\mathbf{s}^*) = \{12 + \log_2(1)\} + \{12 + \log_2(3)\} = 25.585 \).

Now, let us have a closer look at the behavior of the cost function of Figure 5.6. The cost function is plotted against the count of all possible combinations of 0s and 1s available to form all the possible versions of vector \( \mathbf{s} \). If these combinations are coded in binary, then transitions of the kind
1111 → 10000 occur. This results in a large change in the used/not used NOFs.
For example, if \( s = [1 0 1 1 1 1 1 1 1] \) is assumed to be a binary code, it would take the value 767. The cost of this vector is 140.27, because many NOFs are used. The next value of \( s \), 768, would correspond to the vector \( s = [1 1 0 0 0 0 0 0 0] \) with a cost 25.585. This high transition in the cost is due to the transition from a vector that represents many NOFs being used to a vector that represents few NOFs being used. In order to avoid this, \( s \) is counted up via a code (the Gray code) that only changes in one bit position at a time. However, note that despite of this count many local minima exist in Figure 5.6. Therefore, the cost function can be expected in general to be multimodal.

![Figure 5.6: Cost function with vector \( s \) counted in Gray code.](image)

Nature inspired algorithms have become popular because of their ability to minimize multimodal cost functions by mimicking nature’s principles [28]-[32]. Some of the best studied algorithms in engineering are the Simulated Annealing (SA), Genetic Algorithm (GA), Particle Swarm Optimization (PSO)...
and Ant Colony Optimization (ACO). To minimize the cost $F(s)$ in (5.14) we use GA, since the binary representation of the algorithm allows an implementation without mapping of floating-point data to binary and vice versa as required by other nature inspired algorithms. To run the GA algorithm we have used the MATLAB Global Optimization Toolbox.

The GA has some basic parameters such as:

- Population Size (PS): the number of random vectors $s$ that are used to explore the search space,
- Number of Variables (NV): the size of the vector $s$,
- Number of Generations (NG): the number of iterations that the GA performs,
- Elite Count (EC): the percentage of best solutions in the population that can be passed to the next iteration,
- Type of Crossover (TC): how a vector $s$ in the population exchange some of its elements with the elements of another vector in the population.
- Crossover Percentage (CP): Percentage of the population that undergoes crossover, and
- Mutation Probability (MP): the probability that an arbitrary element of a vector $s$ in the population will be changed from its original value.

For a bit string type of problem the following are useful parameters: $PS = 2 \times NV; \ NG = 100; \ EC = 10\%; \ CP = 70-80\%, \ TC = \text{Scattered and MP = Gaussian.}$ Secondary parameter such as selection functions, crossover points, etcetera, were less significant in the simulation and the default GA setting worked well.
5.3.3 Design examples and discussion of results

In the following we compare the proposed method with references [21], [25] and [26], which have presented the most outstanding results in solving the FP-MCM problem, as well as with the Distributed Arithmetic approach from the Xilinx Core Generator 5.0. Comparisons with the simple CSD coding of coefficients and with methods [2] and [7], which are focused on the traditional MCM problem, are also given. Xilinx ISE 12.3 was used as synthesis tool. As device the Virtex-4 XC4VSX25-10FF680 was chosen.

**Example 3**: Design the FP-MCM block for the filter FIR3 from [33] with the target set of coefficients \( T=\{105, 621, 815, 831\} \).

The FP-MCM block is designed with methods [2], [26] and proposed. Figure 5.7 shows the resulting FP-MCM block from method [2] (the original RAG-\( n \)). Note that the \( A \)-depth is 5 and the MCM graph only requires 6 \( A \)-operations. However, when pipelining is considered, 9 extra registers are included, resulting in 15 \( D \)-operations.

![Figure 5.7: FP-MCM block from method [2]. It requires 15 \( D \)-operations.](image-url)
Figure 5.8 shows the resulting FP-MCM block from method [26]. That method achieves a graph with $A$-depth equal to 3 and uses 11 $D$-operations. The FP-MCM block generated with the proposed method is presented in Figure 5.9, and it requires 10 $D$-operations.

![Figure 5.8: FP-MCM block from method [26]. It requires 11 $D$-operations.](image)

![Figure 5.9: FP-MCM block from proposed method. It requires 10 $D$-operations.](image)

Table 5.3 presents the synthesis results for the three blocks. The lowest cost measured by the Area×Time product ($\text{Slices}/F_{\text{max}}$) is achieved by the proposed graph.

292
Table 5.3: Synthesis results for the implementation of filter FIR3 on a Xilinx Virtex-4 XC4VSX25-10FF680

<table>
<thead>
<tr>
<th>Method</th>
<th>No. of FFs</th>
<th>No. of 4-input LUTs</th>
<th>No. of Slices</th>
<th>$F_{\text{max}}$ (MHz)</th>
<th>AT cost (Slices / $F_{\text{max}}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method [2]</td>
<td>314</td>
<td>213</td>
<td>162</td>
<td>335.57</td>
<td>0.48</td>
</tr>
<tr>
<td>Method [26]</td>
<td>293</td>
<td>209</td>
<td>150</td>
<td>335.57</td>
<td>0.45</td>
</tr>
<tr>
<td>Proposed</td>
<td>273</td>
<td>219</td>
<td>141</td>
<td>337.84</td>
<td>0.42</td>
</tr>
</tbody>
</table>

**Example 4:** Design the FP-MCM blocks for the half-band filters F5 to F9 given in [34], the filters from Tables II, IV and V of [35] (respectively denoted here as L1, L2 and L3), and the filters from the two examples in [36] (denoted here as S1 and S2).

In this example the FP-MCM blocks designed with the traditional CSD coefficients representation and with method [21] are compared with the designs based on the proposed method. Table 5.4 presents details of the filters, namely, number of coefficients and word-length of coefficients, and the $A$-depth of the resulting blocks. The synthesis results for these blocks are also included, with cost measured by the Area×Time product (Slices/$F_{\text{max}}$).

It can be seen that for half-band filters F5 to F9, which have many zero coefficients, the pipeline retiming approach of [21] gives the best results because retiming can be done with few extra cost. For other filters the proposed method gives the best performance. The results shown in Table 5.4 for more complex designs are more favorable to the proposed method.
Table 5.4: Synthesis results for the implementation of first benchmark filters on a Xilinx Virtex-4 XC4VSX25-10FF680

<table>
<thead>
<tr>
<th>Filter</th>
<th>A-depth</th>
<th>No. of coeff.</th>
<th>Word-length</th>
<th>Method</th>
<th>CSD</th>
<th>[21]</th>
<th>Proposed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>AT cost (Slices / $F_{max}$)</td>
<td>AT cost (Slices / $F_{max}$)</td>
<td>AT cost (Slices / $F_{max}$)</td>
<td></td>
</tr>
<tr>
<td>L3</td>
<td>1</td>
<td>36</td>
<td>11</td>
<td>2.29</td>
<td>1.35</td>
<td>1.35</td>
<td></td>
</tr>
<tr>
<td>F5</td>
<td>2</td>
<td>11</td>
<td>8</td>
<td>0.82</td>
<td>0.33</td>
<td>0.39</td>
<td></td>
</tr>
<tr>
<td>F7</td>
<td>2</td>
<td>11</td>
<td>9</td>
<td>1.00</td>
<td>0.42</td>
<td>0.46</td>
<td></td>
</tr>
<tr>
<td>F8</td>
<td>2</td>
<td>15</td>
<td>10</td>
<td>1.05</td>
<td>0.64</td>
<td>0.67</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>2</td>
<td>25</td>
<td>9</td>
<td>1.65</td>
<td>0.92</td>
<td>0.92</td>
<td></td>
</tr>
<tr>
<td>S2</td>
<td>2</td>
<td>60</td>
<td>14</td>
<td>7.47</td>
<td>3.45</td>
<td>3.46</td>
<td></td>
</tr>
<tr>
<td>L2</td>
<td>2</td>
<td>63</td>
<td>13</td>
<td>6.38</td>
<td>3.17</td>
<td>3.16</td>
<td></td>
</tr>
<tr>
<td>F6</td>
<td>3</td>
<td>11</td>
<td>9</td>
<td>0.85</td>
<td>0.39</td>
<td>0.40</td>
<td></td>
</tr>
<tr>
<td>F9</td>
<td>3</td>
<td>19</td>
<td>13</td>
<td>2.47</td>
<td>0.90</td>
<td>0.89</td>
<td></td>
</tr>
<tr>
<td>L1</td>
<td>3</td>
<td>121</td>
<td>17</td>
<td>20.76</td>
<td>9.38</td>
<td>7.59</td>
<td></td>
</tr>
</tbody>
</table>

**Example 5:** Design the FP-MCM blocks for the nine filters given in [25], which are 16 bit symmetric FIR filters that all have minimum A-depth equal to 3 and range from 6 to 151 in filter length.

In this example we compare the synthesis results of the implementation of the FP-MCM blocks designed with the Parallel Distributed Arithmetic (PDA) of the Xilinx Core Generator 5.0, and with methods [25], [26] and proposed. Table 5.5 shows these synthesis results. Note that the proposed method offers the minimum (best) Area×Time (Slices/$F_{max}$) cost in all the cases.
Table 5.5: Synthesis results for the implementation of second benchmark filters on a Xilinx Virtex-4 XC4VSX25-10FF680

<table>
<thead>
<tr>
<th>Filter length</th>
<th>Method</th>
<th>AT cost (Slices ( / F_{max} ))</th>
<th>AT cost (Slices ( / F_{max} ))</th>
<th>AT cost (Slices ( / F_{max} ))</th>
<th>AT cost (Slices ( / F_{max} ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>PDA (Xilinx)</td>
<td>0.81</td>
<td>1.17</td>
<td>0.55</td>
<td>0.52</td>
</tr>
<tr>
<td>10</td>
<td>[25]</td>
<td>1.60</td>
<td>2.02</td>
<td>1.14</td>
<td>0.81</td>
</tr>
<tr>
<td>13</td>
<td>[26]</td>
<td>1.19</td>
<td>2.29</td>
<td>1.45</td>
<td>1.17</td>
</tr>
<tr>
<td>20</td>
<td>Proposed</td>
<td>0.96</td>
<td>3.89</td>
<td>1.97</td>
<td>1.51</td>
</tr>
<tr>
<td>28</td>
<td></td>
<td>4.70</td>
<td>5.38</td>
<td>2.81</td>
<td>2.45</td>
</tr>
<tr>
<td>41</td>
<td></td>
<td>7.00</td>
<td>8.34</td>
<td>4.43</td>
<td>3.40</td>
</tr>
<tr>
<td>61</td>
<td></td>
<td>11.36</td>
<td>11.64</td>
<td>5.62</td>
<td>5.23</td>
</tr>
<tr>
<td>119</td>
<td></td>
<td>21.50</td>
<td>19.18</td>
<td>12.15</td>
<td>9.81</td>
</tr>
<tr>
<td>151</td>
<td></td>
<td>29.02</td>
<td>29.52</td>
<td>15.03</td>
<td>13.26</td>
</tr>
</tbody>
</table>

5.4 Conclusion

Whenever the multiplication by constant coefficients is needed, the use of costly general multipliers can be avoided. The efficient design of multiplierless blocks with parallel multiplications by constants (known as SCM for Single Constant Multiplication or MCM for Multiple Constant Multiplications) is an active research field due to its wide range of applications, mainly in the area of Digital Signal Processing. One of these applications is in the design of high-speed FIR digital filters.

In this chapter we have introduced important contributions to solve the SCM and MCM problems. It has been presented an extension of the current
theoretical lower bounds for the $A$-cost and the $A$-depth in SCM blocks constructed with shifts and $A$-operations (additions and subtractions). The key in these new lower bounds has been the introduction of the number of prime factors of the constant multiplier as input information. Additionally, a new lower bound has been revealed, namely, the theoretical minimum number of $A$-operations required to preserve the minimum $A$-cost. The developed lower bounds can provide a more realistic information for a given constant multiplier in terms of its a priori estimated complexity when it is implemented as a SCM block.

The inclusion of pipeline registers in SCM and MCM blocks can provide a considerable increase in the throughput of the blocks at expenses of requiring more area to implement the registers. This is an especially attractive approach for FPGA-based implementations, where pipelining have a low cost if the SCM and MCM blocks are properly optimized. To solve the Fully-Pipelined Multiple Constant Multiplication (FP-MCM) problem, mainly focused to design high-speed FIR filters, we have defined the $D$-operation as an abstraction of a pipelined operation, i.e., either an $A$-operation followed by a register or only a simple register. With the basis on our extended lower bounds for the SCM case, we obtained the lower bound for the number of $D$-operations in SCM blocks and it was shown that this lower bound is directly proportional to the $A$-depth of the block. With this we justified the constraint of preserving the theoretical minimum $A$-depth of the multiplier in any SCM or MCM algorithm employed to minimize the number of $D$-operations. By adding the key observation that the simplest constant multipliers in a MCM block should be synthesized at the end (i.e., they should be placed in the last
stage of pipelining), we provided a simple formulation to the FP-MCM problem that was efficiently solved via Genetic Algorithm. The synthesis results on a Xilinx Virtex-IV device have shown that the proposed algorithm results in a lower Area-Time cost with respect to other methods recently proposed in literature.

5.5 References


6 General conclusions

What profit does the worker have in all that he labors?  
Ecclesiastes 3:9

6.1 Conclusion

This dissertation has presented novel methods to design low-complexity digital FIR filters. The design of low-pass narrowband filters applied in decimation processes has been specially covered due to its particular importance, and efficient proposals based on the identical subfilters approach have been introduced. The most used subfilter in these filtering schemes is the Recursive Running Sum (RRS) filter. Thus, the problem of improving the magnitude characteristics of RRS filters with as low as possible additional computational complexity has been addressed. It has been paid special attention to the two-stage RRS-based decimation schemes due to their efficiency. The results have outperformed other recent methods reported in
literature in terms of both, magnitude response improvement and computational complexity measured in Additions Per Output Sample (APOS) metric. An important observation relies in the fact that in two-stage RRS-based decimation filters, the second stage filter can improve the worst-case attenuation and simultaneously operate after the first-stage downsampling, thus resulting in less computational complexity. The passband improvement can be performed with simple compensation filters that work after the second-stage downsampling, and the proposed second-order compensators based on amplitude transformation are efficient to perform this task with a convenient trade-off between magnitude response improvement and filter complexity. As the magnitude specifications become stricter, the approach of preconditioning two cascaded RRS filters with a simple compensator and then applying an optimal sharpening polynomial with the proper order results in an efficient solution.

Additionally, an optimization framework to design Hilbert transformers (a special class of wideband filters particularly useful in communication applications) with simple identical subfilters was proposed. With this method, any objective function that includes the weighted costs of computational elements and memory elements can be minimized. The proposed formulation is flexible to take in account different costs in the objective function, and therefore is superior to other similar schemes recently proposed in literature.

On the other hand, we have developed a fruitful research that enlarges the capabilities of Cyclotomic Polynomial Filters (CPF) and extends the theoretical knowledge on these filters, especially for cases where recursive
structures, advisable when the computational complexity is a main concern, are used. After exploring the first 200 CPFs, it has been proved that any CPF, regardless of the value of its index, can be expressed as a cascade of recursive basic building blocks with unitary coefficients. Moreover, an explicit formula, which is the basis to obtain the transfer function of any CPF expressed as a product of basic building blocks with unitary coefficients, was developed. It was proved that this formula provides a recursive transfer function with the minimum number of additions for any CPF. These demonstrations allowed us to develop the theorem of preservation of unitary coefficients, which constitutes the base to form an extended search space for the design of filters with arbitrarily narrow passband. From the proposed theorem, the extended set of eligible CPFs is guaranteed to have the minimum number of additions if the CPFs are expressed as a cascade of basic building blocks. This theorem also clarifies the compromise between the index of the CPFs and their minimal computational complexity.

Finally, important contributions to solve the Single Constant Multiplication and Multiple Constant Multiplication (respectively SCM and MCM) problems were presented. We introduced an extension of the current theoretical lower bounds for the adder cost and the adder depth in SCM blocks constructed with shifts and $\Lambda$-operations (additions and subtractions) and a new lower bound has been revealed, namely, the theoretical minimum number of adders required to preserve the minimum adder depth. Since reducing the adder cost subject to the minimum adder depth is currently the main objective in multiplierless arithmetic blocks (because the critical path contributes to degrade the speed and increase power consumption in these
blocks), the revelation of this new lower bound represents a useful theoretical result. Thus, the developed lower bounds can provide a more realistic information for a given constant multiplier in terms of its a priori estimated complexity when it is implemented as a SCM block. With the basis on our extended lower bounds for the SCM case, we obtained the lower bound for the number of pipelined addition operations in SCM blocks and it was shown that this lower bound is directly proportional to the adder depth of the block. With this we justified the constraint of preserving the theoretical minimum adder depth of the multiplier in any SCM or MCM algorithm employed to minimize the number of pipelined addition operations. Besides, a key observation in the case of pipelined MCM problems is that the simplest constant multipliers in a MCM block should be synthesized at the end (i.e., they should be placed in the last stage of pipelining). From these observations a simple formulation to the FP-MCM problem was developed and it was efficiently solved via Genetic Algorithm. The synthesis results on a Xilinx Virtex-IV device have shown that the proposed algorithm results in a lower Area-Time cost with respect to other methods recently proposed in literature.

6.2 Future work

The efficient design of digital FIR filters is closely related to the emerging DSP technologies and applications. The hitherto methods are not capable of satisfying the plethora of conflicting design requirements like stringent design specifications, low power consumption, low area requirements, high speed of computations and low time and design effort, just to name a few. Therefore, it is important to pursue research towards more general methods
to design low-complexity FIR filters, where subfilter-based and MCM-based techniques can be properly improved and then applied in a holistic point of view depending on the required application and the implementation platform. However, this task must be properly delimited taking in account the evolving technologies and their capacities, as well as the priorities in the design goals. The boundaries where identical-subfilter-based and periodical-subfilter-based designs can be effectively used, depending on the frequency response specifications, the clock rate available, and the metrics for area, speed and power efficiency in an arbitrary target technology, can not be completely clear because these technologies evolve fast, but they can be better defined.

Keeping separated design methods for distinct applications and gradually migrating to most general methods whenever is possible is a wise decision. In this sense, we can list future works in a direct connection with the research covered in this dissertation. With respect to the design of RRS-based filters for decimation, we can explore further the proposed design of compensation filters based on amplitude transformation, which can provide an intuitive and low-complexity method for compensators with higher order. Moreover, these compensation filters can be combined with the proposed method for CPF-based filters to give rise to efficient narrow-band multiplierless filters. The integration of these filters as a part of subfilter-based schemes to generate filters with arbitrary bandwidth can be also investigated.

Moreover, FPGA platforms have gained a tremendous ground in the field of DSP and therefore this can be the target technology for future research. The four recent Xilinx FPGA families, in comparison to the two previous ones,
include a much wider number of dedicated DSP resources that can be efficiently used for the proposed schemes. The efficient integration of these resources to design low-complexity FIR filters with increasingly strict frequency response specifications is an important research to be carried out.
Additional proofs for theorems in Chapter 5

Now faith is the substantiation of things hoped for, the conviction of things not seen.

Hebrews 11:1

The following proofs are provided to support Theorems 1 to 4, introduced in Chapter 5 to derive the extended theoretical lower bounds for adder cost and adder depth in SCM multiplierless blocks.

A.1 First proof required in Theorem 1

Prove that the statement $S(k,l)$ is true for $k \geq 2$ and $1 \leq l \leq k-1$, with

$$S(k,l) = \{ 2^{k-1} < 2^{k-1} - 2^{l+2} \}.$$

Step 1 (base case): We have $S(k, 1) = \{ 2^{k-2} < 2^{k-1} \}$ and $S(k, k-1) = \{ 1 < 2 \}$, which hold for $k \geq 2$. This proves the cases when $k = 2$ and $k = 3$ and their corresponding values for $l$. It is also proved the case when $k = 4$, but only for $l$
= 1 and \( l = 3 \). In this case, we need to prove separately only \( k = 4 \) and \( l = 2 \). This can be easily proved, since \( S(4, 2) = \{ 2^{+2-1} < 2^{+1-2+2} \} = \{ 2 < 6 \} \).

**Step 2 (induction on \( l \)):** From Step 1, we only need to show that \( S(k, l) \) holds for \( k > 4 \) and \( 2 \leq l \leq k-2 \). Let us assume that \( S(k, l) \) holds for \( k > 4 \) and \( 1 \leq l \leq k-3 \), i.e.,

\[
2^{k-l-1} < 2^{k-1-2}+2 \quad \text{(for \( k > 4 \) and \( 1 \leq l \leq k-3 \)).} \tag{A.1}
\]

By multiplying both sides by \( 2^{-1} \) we get

\[
2^{k-(l+1)-1} < 2^{l-1}[2^{k-1-2}+2]. \tag{A.2}
\]

On the other hand, note that the following inequality,

\[
2^{-1}[2^{k-1-2}+2] \leq 2^{k-1-2(l+1)+2}, \tag{A.3}
\]

is true for \( k > 4 \) and \( 1 \leq l \leq k-3 \) (see the proof in the subsection A.5.1). Therefore, from (A.2) and (A.3) we can write

\[
2^{k-(l+1)-1} < 2^{-1}[2^{k-1-2}+2] \leq 2^{k-1-2(l+1)+2} \quad \text{(for \( k > 4 \) and \( 1 \leq l \leq k-3 \)).} \tag{A.4}
\]

which becomes

\[
S(k, l+1) = \{ 2^{k-(l+1)-1} < 2^{k-1-2(l+1)+2} \} \quad \text{(for \( k > 4 \) and \( 2 \leq (l+1) \leq k-2 \)).} \tag{A.5}
\]

Clearly, assuming that \( S(k, l) \) holds for \( k > 4 \) and \( 1 \leq l \leq k-3 \) implies that \( S(k, l+1) \) holds for \( k > 4 \) and \( 2 \leq (l+1) \leq k-2 \). By joining this implication with the base cases in Step 1, the proof is complete.

### A.2 Second proof required in Theorem 1

Prove that the statement \( S(k, l) \) is true for \( k \geq 2 \) and \( 1 \leq l \leq k-1 \), with

\[
S(k, l) = \{ 2^j(2^{k-l+1}+1) + 2^{k-1} \leq 2^k+1 \}
\]

Let us start by rewriting \( S(k, l) \) as

\[
2^{k-1} + 2^l + 2^{k-1} \leq 2^k+1 \quad (k \geq 2 \text{ and } 1 \leq l \leq k-1). \tag{A.6}
\]
By adding \(-2^{k-1}\) to both sides we get the statement \(P(k,l)\) as
\[
P(k,l) = \{ 2^l + 2^{k-1} \leq 2^k - 2^{k-1} + 1 \} \quad (k \geq 2 \text{ and } 1 \leq l \leq k-1).
\]
(A.7)
It is clear that, by giving the proof for \(P(k,l)\), we also prove \(S(k,l)\). Thus, let us prove \(P(k,l)\).

**Step 1 (base case):** We have
\[
P(k, 1) = \{ 2 + 2^{k-2} \leq 2^k - 2^{k-1} + 1 \} = \{ 1 \leq 2^k - 2^{k-1} - 2^{k-2} \}
\]
and
\[
P(k, k-1) = \{ 2^{k-1} + 2^{k-(k-1)-1} \leq 2^k - 2^{k-1} + 1 \} = \{ 2^{k-1} + 1 \leq 2^k - 2^{k-1} + 1 \} = \{ 2^{k-1} = 2^l + 1 \}
\]
which hold for \(k \geq 2\). This proves the cases when \(k = 2\) and \(k = 3\) and their corresponding values for \(l\). It is also proved the case when \(k = 4\), but only for \(l = 1\) and \(l = 3\). In this case, we need to prove separately only \(k = 4\) and \(l = 2\). This can be easily proved, since
\[
P(4, 2) = \{ 2^2 + 2^{4-2} \leq 2^4 - 2^{4-1} + 1 \} = \{ 6 < 9 \}.
\]

**Step 2 (induction on \(l\)):** From Step 1, we only need to show that \(P(k,l)\) holds for \(k > 4\) and \(2 \leq l \leq k-2\). Let us assume that \(P(k,l)\) holds for \(k > 4\) and \(1 \leq l \leq k-3\), i.e.,
\[
2^l + 2^{k-1} \leq 2^k - 2^{k-1} + 1 \quad (k > 4 \text{ and } 1 \leq l \leq k-3).
\]
(A.8)
By multiplying both sides of (A.8) by \(2^{-1}\) we get
\[
2^{l-1} + 2^{k-1)}, (l-1) \leq 2^{k-1} - 2^{k-2} + 2^{-1},
\]
(A.9)
and adding \(3 \times 2^{l-1}\) to both sides of (A.9) results in
\[
2^{l-1}(1 + 3) + 2^{k-1)}, (l-1) \leq 2^{k-1} - 2^{k-2} + 2^{-1} + 3 \times 2^{l-1},
\]
(A.10)
\[
2^{l-1} + 2^{k-1)}, (l-1) \leq 2^{k-1} - 2^{k-2} + 2^{-1} + 3 \times 2^{l-1}.
\]
(A.11)
On the other hand, note that the following inequality,
\[
2^{k-1} - 2^{k-2} + 2^{-1} + 3 \times 2^{l-1} \leq 2^{k-1} - 2^{k-1} + 1,
\]
(A.12)
is true for \( k > 4 \) and \( 1 \leq l \leq k-3 \) (see the proof in the subsection A.5.2). Therefore, from (A.12) and (A.11) we can write
\[
2^{l+1} + 2^{k-(l+1)} - 1 \leq 2^{l-2} + 2^{l-1} + 3 \times 2^{l-1} \leq 2^{k-2} + 1 \quad \text{(for } k > 4 \text{ and } 1 \leq l \leq k-3 \text{)} \quad (A.13)
\]
which becomes
\[
P(k, l+1) = \{ 2^{l+1} + 2^{k-(l+1)} - 1 \leq 2^{k-2} + 1 \} \quad \text{(for } k > 4 \text{ and } 2 \leq (l+1) \leq k-2 \text{).} \quad (A.14)
\]
Clearly, assuming that \( P(k, l) \) holds for \( k > 4 \) and \( 1 \leq l \leq k-3 \) implies that \( P(k, l+1) \) holds for \( k > 4 \) and \( 2 \leq (l+1) \leq k-2 \). By joining this implication with the base cases in Step 1, the proof is complete for \( P(k,l) \).

A.3 Proof required in Theorem 3

Prove that the statement \( S(k, l) \) is true for \( k \geq 3 \) and \( 1 \leq l \leq k-2 \), with
\[
S(k, l) = \{ 2^{k-2} + 2^{k-3} + 2^{l-2} + 2^{l-1} \leq 0.75 \times 2^k + 1 \}
\]
Let us start by adding \(-2^{k-2} - 2^{k-3}\) to both sides of the inequality in \( S(k, l) \). We get the statement \( P(k, l) \) as
\[
P(k, l) = \{ 2^{l} + 2^{l-1} + 2^{k-l-2} \leq 0.75 \times 2^k + 1 - 2^{k-2} - 2^{k-3} \} \quad (k \geq 3 \text{ and } 1 \leq l \leq k-2). \quad (A.15)
\]
It is clear that, by giving the proof for \( P(k, l) \), we also prove \( S(k, l) \). Thus, let us prove \( P(k, l) \).

Step 1 (base case): We have
\[
P(k, 1) = \{ 2 + 1 + 2^{k-3} \leq 0.75 \times 2^k - 2^{k-2} - 2^{k-3} + 1 \} = \{ 2 \leq 0.75 \times 2^k - 2^{k-1} \} = \{ 2 \leq 0.25 \times 2^k \}
\]
and
\[
P(k, k-2) = \{ 2^{k-2} + 2^{k-3} + 1 \leq 0.75 \times 2^k + 1 - 2^{k-2} - 2^{k-3} \} = \{ 0 \leq 0.75 \times 2^k - 2^{k-1} - 2^{k-2} \} = \{ 0 \leq 0 \}
\]
which hold for \( k \geq 3 \). This proves the cases when \( k = 3 \) and \( k = 4 \) and their corresponding values for \( l \). It is also proved the case when \( k = 5 \), but only for \( l \)
= 1 and \( l = 3 \). In this case, we need to prove separately only \( k = 5 \) and \( l = 2 \). This can be easily proved, since
\[
P(5, 2) = \{ 2^5+2^2+1+2^5-2^2-2^3 \} = \{ 8 < 13 \}.
\]

**Step 2 (induction on \( l \)):** From Step 1, we only need to show that \( P(k,l) \) holds for \( k > 5 \) and \( 2 \leq l \leq k-3 \). Let us assume that \( P(k,l) \) holds for \( k > 5 \) and \( 1 \leq l \leq k-4 \), i.e.,
\[
2^4+2^4+2^{l-2} \leq 0.75 \times 2^k + 1 - 2^{k-2} - 2^{k-3} \quad (k > 5 \text{ and } 1 \leq l \leq k-4).
\]
By multiplying both sides of (A.16) by \( 2^3 \) we get
\[
2^7+2^7+2^{l-2}(1+3) \leq 0.75 \times 2^{k-1} + 2^1 - 2^{k-3} - 2^{k-4},
\]
and adding \( 3 \times 2^{l-1} + 3 \times 2^{l-2} \) to both sides of (A.17) results in
\[
2^{l-1} + 2^{l-2} + 2^{l-1} \leq 0.75 \times 2^{k-1} + 2^1 - 2^{k-3} - 2^{k-4} + 3 \times 2^{l-1} + 3 \times 2^{l-2} \quad (A.18)
\]
\[
2^{l} + 2^{l-1} + 2^{l-1} \leq 0.75 \times 2^{k-1} + 2^1 - 2^{k-3} - 2^{k-4} + 3 \times 2^{l-1} + 3 \times 2^{l-2}. \quad (A.19)
\]
On the other hand, note that the following inequality,
\[
0.75 \times 2^{k-1} + 2^1 - 2^{k-3} - 2^{k-4} + 3 \times 2^{l-1} + 3 \times 2^{l-2} \leq 0.75 \times 2^k + 1 - 2^{k-2} - 2^{k-3}, \quad (A.20)
\]
is true for \( k > 5 \) and \( 1 \leq l \leq k-4 \) (see the proof in the subsection A.5.3). Therefore, from (A.20) and (A.19) we can write
\[
2^{l+1} + 2^{l+1} + 2^{l+1} \leq 0.75 \times 2^{k-1} + 2^1 - 2^{k-3} - 2^{k-4} + 3 \times 2^{l-1} + 3 \times 2^{l-2} \leq 0.75 \times 2^k + 1 - 2^{k-2} - 2^{k-3}
\]
\[\text{ (for } k > 5 \text{ and } 1 \leq l \leq k-4), \quad (A.21)\]
which becomes
\[
P(k, l+1) = \{ 2^{l+1} + 2^{l+1} + 2^{l+1} \leq 0.75 \times 2^k + 1 - 2^{k-2} - 2^{k-3} \}
\]
\[\text{ (for } k > 5 \text{ and } 2 \leq (l+1) \leq k-3). \quad (A.21)\]
Clearly, assuming that \( P(k,l) \) holds for \( k > 5 \) and \( 1 \leq l \leq k-4 \) implies that \( P(k,l+1) \) holds for \( k > 5 \) and \( 2 \leq (l+1) \leq k-3 \). By joining this implication with the base cases in Step 1, the proof is complete for \( P(k,l) \).
A.4 Proof required in Theorem 4

Prove that the statement $S(p, l, r)$ is true for $p \geq 2$, $1 \leq l \leq p-2$ and $1 \leq r \leq l$ with

$$S(p, l, r) = \{ 2^{r-2} + 2^l + 2^{r-1} \times (2^{p-1}+1) \leq 0.75 \times 2^r + 2^{r-1} \}$$

Let us start by adding $-2^{r-1}$ to both sides of the inequality in $S(p, l, r)$. We get the statement $P(p, l, r)$ as

$$P(p, l, r) = \{ 2^{r-2} + 2^l + 2^{r-1} \leq 0.75 \times 2^r \} \quad (p \geq 2, 1 \leq l \leq p-2 \text{ and } 1 \leq r \leq l). \quad (A.22)$$

It is clear that, by giving the proof for $P(p, l, r)$, we also prove $S(p, l, r)$. Additionally, note that the term on the left-hand side of the inequality (A.22), namely, $2^{r-2} + 2^l$, increases as $r$ increases. Thus, if $P(p, l, r)$ holds for the upper limit of $r$, it holds for all the other values of $r$. Substituting $r = l$ in (A.22) we have

$$2^{r-2} + 2^l \leq 0.75 \times 2^r, \quad (A.23)$$

$$2^{r-1} + 2^l \leq 0.75 \times 2^r, \quad (A.24)$$

which clearly holds for all values of $l$ and $p$. Thus, the proof is complete.

A.5 Additional proofs

The following proofs are required in the previous sections.

A.5.1 Proof for (A.3)

Prove that the inequality $2^{-1} [2^{k-1} - 2^l + 2] < 2^{k-1} - 2^{(l+1)+1} + 2$ is true for $k > 4$ and $1 \leq l \leq k-3$.

By making the product in the left-hand term of the inequality we can write
If we add the term $-2^{k-2}+2^{l-1} - 2$ to both sides of the inequality (A.25) we obtain
\begin{align*}
-1 &< 2^{k-1} - 2^{l-1} - 2^{k-2} + 2^{l-1}, \\
-1 &< 2^{k-2}(2-1) + 2^{l-1}(-2^2 + 1), \\
-1 &< 2^{k-2} - 3 \times 2^{l-1}.
\end{align*}
(A.26) 
(A.27) 
(A.28)

Observe from (A.28) that, as $l$ increases, the term on the right hand (namely, $2^{k-2} - 3 \times 2^{l-1}$) decreases for a given $k$, with $k > 4$. Thus, if (A.28) holds for the highest value of $l$, it holds for all its values. By substituting $l = k - 3$ in (A.28) we get
\begin{align*}
-1 &< 2^{k-2} - 3 \times 2^{(k-3)-1}, \\
-1 &< 2^2 \times 2^{k-4} - 3 \times 2^{k-4}, \\
-1 &< 2^{k-4}.
\end{align*}
(A.29) 
(A.30) 
(A.31)

Clearly, (A.31) holds for $k > 4$. ■

A.5.2 Proof for (A.12)

Prove that the inequality $2^{k-1} - 2^{k-2} + 2^{l-1} + 3 \times 2^{l-1} \leq 2^k - 2^{k-1} + 1$ is true for $k > 4$ and $1 \leq l \leq k-3$.

Observe that, as $l$ increases, the term on the left hand (namely, $2^{k-1} - 2^{k-2} + 2^{l-1} + 3 \times 2^{l-1}$) increases for a given $k$, with $k > 4$. Thus, if the statement holds for the highest value of $l$, it holds for all the other values of $l$. By substituting $l = k - 3$ we get
\begin{align*}
2^{k-1} - 2^{k-2} + 2^{l-1} + 3 \times 2^{(k-3)-1} &\leq 2^k - 2^{k-1} + 1, \\
2^k - 2^{k-2} + 2^{k-3} + 2^{l-1} &\leq 2^k + 1, \\
-2^{k-2} + 2^{k-3} + 2^{k-4} &\leq 2^{-1},
\end{align*}
(A.32) 
(A.33) 
(A.34)
which holds for $k > 4$. ■

A.5.3 Proof for (A.20)

Prove that the inequality $0.75 \times 2^{k-1} + 2^1 - 2^{k-3} - 2^{k-4} + 3 \times 2^{k-3} + 3 \times 2^{k-2} \leq 0.75 \times 2^k + 1 - 2^{k-2} - 2^{k-3}$ is true for $k > 5$ and $1 \leq l \leq k - 4$.

Observe that, as $l$ increases, the term on the left hand (namely, $0.75 \times 2^{k-1} + 2^1 - 2^{k-3} - 2^{k-4} + 3 \times 2^{k-3} + 3 \times 2^{k-2}$) increases for a given $k$, with $k > 5$. Thus, if the statement holds for the highest value of $l$, it holds for all the other values of $l$.

By substituting $l = k - 4$ we get

$$2^{k-4} (-4+2+1) \leq 2^{-1}, \quad (A.35)$$

$$2^{k-4} (-1) \leq 2^{-1}, \quad (A.36)$$

$$2^{k-4} \leq 2^{-1}, \quad (A.37)$$

$$-0.75 \times 2^k + 0.75 \times 2^{k-1} + 2^{k-2} - 2^{k-4} + 3 \times 2^{k-5} + 3 \times 2^{k-6} \leq 2^{-1}, \quad (A.38)$$

$$2^{k-6} (-0.75 \times 64 + 0.75 \times 32 + 16 - 4 + 6 + 3) \leq 2^{-1}, \quad (A.39)$$

$$2^{k-6} (-3) \leq 2^{-1}, \quad (A.40)$$

which holds for $k > 5$. ■
Now to the one who works, his wages are not accounted according to grace, but according to what is due.

Romans 4:4

This appendix lists nineteen publications resulting from this dissertation.

B.1 Journals


### B.2 Chapter books


### B.3 International conferences


**B.4 National congresses**

# Glossary

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACF</td>
<td>Amplitude Change Function</td>
</tr>
<tr>
<td>ACO</td>
<td>Ant Colony Optimization</td>
</tr>
<tr>
<td>APOS</td>
<td>Additions Per Output Sample</td>
</tr>
<tr>
<td>ASIC</td>
<td>Application-Specific Integrated Circuit</td>
</tr>
<tr>
<td>AT</td>
<td>Area-Time product</td>
</tr>
<tr>
<td>CIC</td>
<td>Cascaded Integrator-Comb</td>
</tr>
<tr>
<td>CLB</td>
<td>Configurable Logic Block</td>
</tr>
<tr>
<td>CM</td>
<td>Completely Multiplicative</td>
</tr>
<tr>
<td>CP</td>
<td>Cyclotomic Polynomial</td>
</tr>
<tr>
<td>CPF</td>
<td>Cyclotomic Polynomial Filter</td>
</tr>
<tr>
<td>CRRS</td>
<td>Chebyshev Recursive Running Sum</td>
</tr>
<tr>
<td>CSD</td>
<td>Canonic Signed Digit</td>
</tr>
<tr>
<td>CSE</td>
<td>Common Subexpression Elimination</td>
</tr>
<tr>
<td>DA</td>
<td>Distributed Arithmetic</td>
</tr>
<tr>
<td>DAG</td>
<td>Direct Acyclic Graph</td>
</tr>
<tr>
<td>DSP</td>
<td>Digital Signal Processing</td>
</tr>
<tr>
<td>FF</td>
<td>Flip-Flop</td>
</tr>
<tr>
<td>FIR</td>
<td>Finite Impulse Response</td>
</tr>
<tr>
<td>FP</td>
<td>Fully Pipelined</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>FPGA</td>
<td>Field Programmable Gate Array</td>
</tr>
<tr>
<td>FP-MCM</td>
<td>Fully Pipelined Multiple Constant Multiplication</td>
</tr>
<tr>
<td>FP-SCM</td>
<td>Fully Pipelined Single Constant Multiplication</td>
</tr>
<tr>
<td>FRM</td>
<td>Frequency Response Masking</td>
</tr>
<tr>
<td>FT</td>
<td>Frequency Transformation</td>
</tr>
<tr>
<td>GA</td>
<td>Genetic Algorithm</td>
</tr>
<tr>
<td>HT</td>
<td>Hilbert Transformer</td>
</tr>
<tr>
<td>IFIR</td>
<td>Interpolated Finite Impulse Response</td>
</tr>
<tr>
<td>IIFOP</td>
<td>Inverse Interpolated First Order Polynomial</td>
</tr>
<tr>
<td>IIR</td>
<td>Infinite Impulse Response</td>
</tr>
<tr>
<td>ILP</td>
<td>Integer Linear Programming</td>
</tr>
<tr>
<td>ISOP</td>
<td>Interpolated Second Order Polynomial</td>
</tr>
<tr>
<td>LB</td>
<td>Logic Block</td>
</tr>
<tr>
<td>LC</td>
<td>Logic Cell</td>
</tr>
<tr>
<td>LE</td>
<td>Logic Element</td>
</tr>
<tr>
<td>LTI</td>
<td>Linear Time Invariant</td>
</tr>
<tr>
<td>LUT</td>
<td>Look-Up Table</td>
</tr>
<tr>
<td>MCM</td>
<td>Multiple Constant Multiplication</td>
</tr>
<tr>
<td>MILP</td>
<td>Mixed Integer Linear Programming</td>
</tr>
<tr>
<td>MNSD</td>
<td>Minimum Number of Signed Digits</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>MPOS</td>
<td>Multiplications Per Output Sample</td>
</tr>
<tr>
<td>NEF</td>
<td>Non-Essential Fundamental</td>
</tr>
<tr>
<td>NOF</td>
<td>Non-Output Fundamental</td>
</tr>
<tr>
<td>NP</td>
<td>Nondeterministic Polynomial time</td>
</tr>
<tr>
<td>OF</td>
<td>Output Fundamental</td>
</tr>
<tr>
<td>PDA</td>
<td>Parallel Distributed Arithmetic</td>
</tr>
<tr>
<td>PI</td>
<td>Pipelining-Interleaving</td>
</tr>
<tr>
<td>PSO</td>
<td>Particle Swarm Optimization</td>
</tr>
<tr>
<td>RAG</td>
<td>Reduced Adder Graph</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>RNS</td>
<td>Residue Number System</td>
</tr>
<tr>
<td>RRS</td>
<td>Recursive Running Sum</td>
</tr>
<tr>
<td>SA</td>
<td>Simulated Annealing</td>
</tr>
<tr>
<td>SCM</td>
<td>Single Constant Multiplication</td>
</tr>
<tr>
<td>SPT</td>
<td>Signed Power-of-Two</td>
</tr>
<tr>
<td>SRC</td>
<td>Sampling Rate Conversion</td>
</tr>
</tbody>
</table>