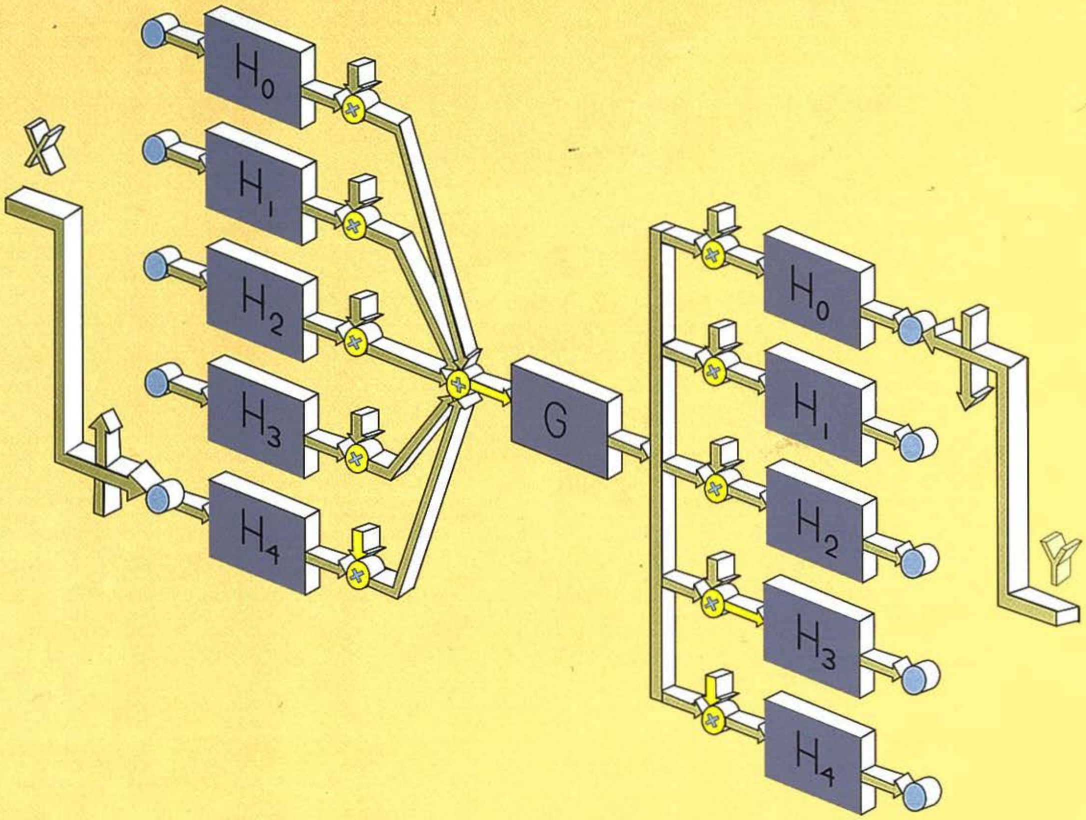
MULTIRATE SIGNAL PROCESSING FOR COMMUNICATION SYSTEMS



fredric j harris

Multirate Signal Processing for Communication Systems

Companion Software Website
http://authors.phptr.com/harris/

fredric j harris

CUBIC Signal Processing Chair

San Diego State University

Prentice Hall PTR
Upper Saddle River, New Jersey 07458
http://www.phptr.com

BIÇAKLAR KİTABEVİ
Basım Yayın ve Dağıtım Ltd. Şti.
Karanfil Sokak No: 27 Kızılay - ANKARA
Tel: (0.312) 417 51 70 Pbx Fax: (0.312) 417 81 46
Çankaya V.D. 168 006 3351 e-mail:info@bicaklar.com.tr

ODTÜ [illegible] ESİ
M. E. [illegible] ARY

Library of Congress Cataloging-in-Publication Data
harris, fred (fredric j.)
Multirate signal processing for communication systems, /frederic j. harris
Includes bibliographical references and index.
ISBN 0-13-146511-2 (alk. paper)
1. Signal processing—Digital techniques.
TK5103.7.H38 2004
621.382'2--dc22 2004044419

TK5103.7
H38
2004

Companion Software Website http://authors.phptr.com/harris/

Editorial/Production Supervision: *William Mara*
Cover Design Director: *Jerry Votta*
Cover Design: *Mary Jo DeFranco*
Cover Illustration: *fredric harris*
Art Director: *Gail Cocker-Bogusz*
Manufacturing Manager: *Carol Melville*
Acquisitions Editor: *Bernard Goodwin*
Editorial Assistant: *Michelle Vincenti*
Marketing Manager: *Dan Depasquale*
Copy Editor: *Elizabeth Martin*

METU LIBRARY
0050348939

PRENTICE HALL PTR
2004 Pearson Education, Inc.
Publishing as Prentice Hall Professional Technical Reference
Upper Saddle River, New Jersey 07458

Prentice Hall PTR offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales. For more information, please contact: U.S. Corporate and Government Sales, 1-800-382-3419, corpsales@pearsontechgroup.com. For sales outside of the U.S., please contact: International Sales, 1-317-581-3793, international@pearsontechgroup.com.
Company and product names mentioned herein are the trademarks or registered trademarks of their respective owners.

All rights reserved. No part of this book may be reproduced, in any form or by any means, without permission in writing from the publisher.
Prentice Hall PTR, One Lake Street, Upper Saddle River, NJ 07458.

Printed in the United States of America
First Printing: May 2004
ISBN 0-13-146511-2

Pearson Education Ltd.
Pearson Education Australia Pty., Limited
Pearson Education South Asia Pte. Ltd.
Pearson Education Asia Ltd.
Pearson Education Canada, Ltd.
Pearson Educacion de Mexico, S.A. de C.V.
Pearson Education—Japan
Pearson Malaysia S.D.N. B.H.D.

To the memory of my parents,
Edith and Seymour Harris,
My wife, Penelope,
Our Children, Danielle and Robyn,
Our Grandson, Justin,
And to all my students and colleagues
with whom I share the joy of learning.

"A little learning is a dangerous thing;
drink deep, or taste not the Pierian spring:
there shallow draughts intoxicate the brain,
and drinking largely sobers us again."

–Alexander Pope (1688-1744)
An Essay on Criticism

Contents

Preface

Digital signal processing (DSP) has become a core body of material in undergraduate electrical engineering programs. Several threads branch from this core to enable related disciplines, such as communication systems, source coding, multimedia entertainment, radar, sonar, medical and laboratory instruments, and others. *Multirate signal processing* is one of these major threads. Multirate signal processing is the body of material that deals with concepts, algorithms, and architectures that embed sample rate changes at one or more sites in the signal flow path.

There are two reasons to include multirate signal processing in the solution of a particular signal-processing task. The first is reduction in cost of the implementation. The second is enhanced performance of the implementation. We might also include a third, personal incentive, which is, that it is fun to apply clever concepts to solve problems. We can hardly complete a multirate DSP design without a smile and the accompanying thought, "Boy, this is neat!"

Traditional concepts developed in the DSP world are the same as those developed in the analog-processing world. In both domains we learn and use concepts such as convolution, Fourier transforms, transfer functions, poles and zeros, and others. When required to distinguish the two approaches we use the qualifier "discrete" when discussing the DSP version of these fundamental concepts. The reason the two approaches are so similar is that they both emphasize *linear time invariant* (LTI) systems for which the tools of analysis and synthesis are well developed.

Multirate signal processing brings to the designer an important tool not available to the traditional DSP designer, who to the first order applies DSP techniques to emulate analog systems. We note that the interface between the two versions of the world, continuous and discrete, is the sampling process. In the traditional DSP perspective, the sample rate is selected to satisfy the Nyquist criterion but is otherwise incidental to the problem. In multirate signal processing, selection and modification of the sample rate are primary considerations and options in the signal processing chain. The option to change the sample rate is the additional tool offered to the DSP designer. Discrete systems with embedded sample rate changes are characterized as *linear time varying* (LTV) or as *periodically time varying.* (PTV) Most of us have very little experience in the continuous world with LTV filters, thus their unique properties come as a pleasant surprise as we learn how to design and use them.

The ability to change sample rate within the processing stream presents a remarkable list of processing tricks and performance enhancements. A consistent theme in this book is the presentation of perspectives that access these processing tricks. The first perspective we present is that a processing task should always be performed at the lowest rate commensurate with the signal bandwidth. This is the Nyquist rate of the signal component of interest.

We note that a common processing task is to reduce the bandwidth of a signal by filtering and then reduce the sample rate to match the reduced bandwidth. Our first processing trick interchanges the order of filtering and sample rate change so that the processing proceeds at the reduced output sample rate rather than at the high input sample rate. The condition under which this interchange is permitted is known as the *noble identity*. Reducing the sample rate prior to reducing the bandwidth causes aliasing of the input spectrum. Multirate signal processing permits, and in fact, supports this intentional aliasing, which can be unwrapped by subsequent processing. In fact, most of the tricks and enhancements associated with multirate signal processing are related to spectral aliasing due to the sample rate change. It seems counterintuitive to use intentional aliasing as part of the signal processing scheme particularly when we have been told over and over not to alias the signal in the data collection process. It also seems a bit suspicious to claim that aliasing can be reversed. But in fact it can be when the aliasing occurs with a specified structure that we ensure in the multirate processing scheme. We can also use the change in sample rate to intentionally alias a signal at one center frequency to another. This option includes aliasing a signal from an intermediate center frequency to baseband by reducing the sample rate, as well as aliasing a signal from baseband to an intermediate center frequency by increasing the sample rate.

PURPOSE OF BOOK

The purpose of this book is to present a clear and intuitive description of the unique capabilities of multirate signal processing. This is accomplished by presenting the core material along with numerous examples supported by the liberal use of figures to illustrate the time and frequency representations of the multirate processing options. We also present a number of useful perspectives to facilitate development of insight of multirate systems. One such insight is that when describing a multirate system, since the sample rate is changing, we don't use the sample rate as our reference as is done in conventional DSP. Rather, it is useful to use the signal symbol rate or signal bandwidth as our reference since that is the single parameter that remains fixed in the process. The book includes many practical applications of multirate systems to help the reader see novel ways they can be applied to solve real problems. Commentary of traditional design techniques and alternate improved options are sprinkled throughout the text. Some of the material presented in the book, by necessity, must mimic similar expositions found in other texts on multirate processing, while other segments of the material reflect my unique perspective and experience. There are specific segments of material that are covered quite extensively here that are only lightly covered in other texts. In particular, the chapter on recursive all-pass filters is a wealth of material deserving greater exposure than traditionally allocated to this topic.

Much of the material presented in this textbook has been used in my graduate course, "Multirate Signal Processing." Significant segments of this material have found their way into my undergraduate course, "Modem Design," as well as a series of short courses and presentations dealing with synchronization techniques in modern communication systems. A light sprinkling of multirate filters can even be found in my undergraduate course in "Digital

Signal Processing" as well as our undergraduate course in "Real-Time Digital Signal Processing." This book can be used in an undergraduate course in advanced DSP concepts or in a graduate course in multirate signal processing. Segments can also be used in support of various courses in communication system design and modem design, and can be used as a source of real-world applications in a DSP programming lab.

One of the pleasures of being a faculty member proficient in DSP skills as well as knowledgeable in modern communication systems is the ability to roam the halls of knowledge as well as the commercial centers of excellence that feed the economic engines of our society. I have had the good fortune of participating in the development of many systems that require the capabilities of high-performance, cost-effective DSP solutions. These systems include laboratory instruments, cable modems, satellite modems, sonar systems, radar systems, wireless systems, and consumer entertainment products. With a foot in each camp, one in academia and one in commercial, I am exposed to a rich and varied set of questions of interest from residents of the two areas. Questions posed by commercial folks addressing focused problems are very different from those posed by those in academics.

Some of my most creative work has been spawned by questions posed, and challenges offered, by perceptive folks in the industrial arena. The academic environment provides me access to promising and talented students with whom I can share the pleasure of learning and understanding an established knowledge base in multirate digital processing while developing and expanding that same base. This text reflects much of that knowledge base tempered by the insight gathered from problem-solving tasks in the commercial sector and nurtured by scholarly interaction with curious, motivated students.

ORGANIZATION OF BOOK

This book is divided into 13 chapters. Chapter 1 is an introduction to multirate signal processing. The 1-to-4 up-sampler of the common consumer CD player is shown as an example of a ubiquitous application. Chapter 2 describes the process of sampling and resampling in time and frequency domains. Chapter 3 presents the relationship between the specifications of a FIR filter and the number of taps, or the length, of the filter. We also compare the window design and equal-ripple or Remez design techniques. The effects of in-band ripple and of constant level stop band side-lobes are examined and various modifications to the design process are introduced to control stop band side-lobe levels. Chapter 4 presents special filters such as square-root Nyquist filters and half-band filters. Discussions on standard and improved design techniques appropriate for these specific filters are also presented.

Chapter 5 presents examples of systems that use multirate filters and illustrates applications and demonstrates the wide range of applications. Chapter 6 presents resampling low pass and band-pass FIR filters for which the noble identity has been applied to interchange the operations of resampling and filtering. Up-sampling, down-sampling, and cascade up-down sampling filters are examined. Chapter 7 describes polyphase interpolators and filters that perform arbitrary sample rate change. We also examine Farrow filters as well as filters that interpolate while performing alias-based translation. Chapter 8 covers quadrature mirror filters and dyadic half-band FIR filters. Chapter 9 covers M-path modulators and demodulator channel banks. Also discussed are simultaneous interpolation and channel bank formation.

Chapter 10 covers recursive all pass filters implemented as nonuniform phase and equiripple approximations to linear phase filters. A number of structures, including half-band, M-path filter banks, resampling structures and arbitrary bandwidth non-resampling structures, are presented and illustrated. Chapter 11 presents the CIC filter and its resampling version, the Hogenauer filter. Chapter 12 describes cascades of low-order zero-packed up-sampled filters that exhibit periodic spectra with narrow transition band. Chapter 13 presents areas of applications in which multirate filters have significant presence.

At the end of each chapter are a number of problems designed to highlight key concepts presented in the chapter. These problems also serve to test the reader's knowledge and understanding of the material. Following each problem set is a list of references to guide the reader to related areas for further study.

ACKNOWLEDGEMENTS

We are able to see further than those who came before us because we stand on their shoulders. In addition, each day presents a fresh horizon because of the work we have performed as well as that performed by those around us. Interaction with our peers, our students, and our patrons enriches us by moving boundaries and lifting veils through the exchange of ideas. I cherish a group of people with whom I have had the great pleasure of sharing work and developing knowledge and with whom I have established long-standing friendships. I would like to cite some of them with the understanding that their ordering is a random process involving fleeting pleasant remembrances.

Tony Constantinides of Imperial College in London has had a special influence on my work. I met Tony at a conference in Brisbane where we started a lifelong friendship and working relationship. It was through Tony that I gained an appreciation of the recursive polyphase half-band filter. I have yet to put down this valuable filter structure. It was Markku Renfors of Tempre University in Finland who first touched me through a classic paper on M-path recursive polyphase filters. That paper brought forth a torrent of ideas, papers, student projects, and commercial products. Many years later, meeting Markku at a conference in Santorini I quickly warmed to him as a fellow traveler and I continue to enjoy our too infrequent academic and personal encounters. In a similar vein, Ron Crochiere through the classic book that he and Larry Rabiner authored, 'Multirate Digital Signal Processing' helped form many of the ideas I nurtured into my concepts and perspectives on multirate filters. I thank Ron for reviewing the manuscript of this book and I'm relieved that it passed muster. I am always pleased to see and spend time with Ron. From the other side of the playing field I take great pleasure in my nearly daily interaction with Bernie Sklar formerly with Aerospace Corporation and now with Communications Engineering Services. I hone my applied DSP skills on the communication system questions and problems that

Bernie brings to me. Interacting with Bernie is like dancing with a skilled partner with both participants exhilarated and charmed by the encounter. Another fellow dancer who sways to the music of applied multirate DSP is Chis Dick of Xilinx. Chis and I met at a conference in Adelaide while he was at LaTrobe University. Our common interest and passion have been the core of a special friendship and our wives tell us that we are boring company when we climb into our common DSP shell to do our DSP magic. Michael Rice of BYU is another special person with whom I have been sharpening my applied DSP skills.

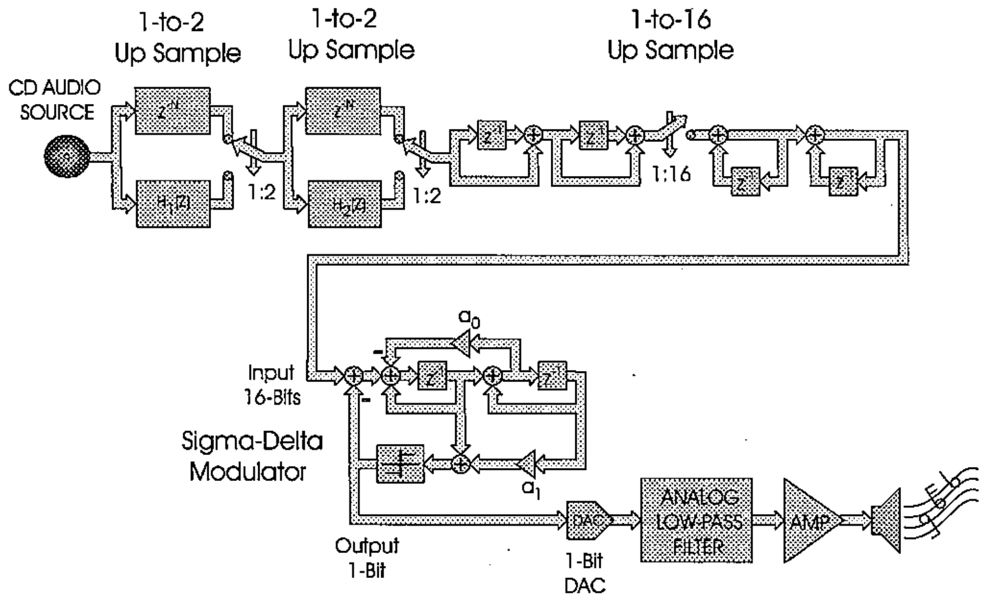
The list of people with whom I initially interacted through professional contacts that have warmed to friendships mingled with professional respect and admiration hold another special place for me. These include Don Steinbrecher, a very creative individual with a piece of his mind firmly planted in our future. Itzhak Gurantz, formerly of Comstream and Rockwell and now with Entropic, is a very smart person who many times has posed fascinating questions that guided me to places I would never have thought to tread. Ralph Hudson with whom I shared an office at Hughes Aircraft while I was on a leave of absence from SDSU, and David Lynch, who enticed me to visit Hughes, are clever individuals with enormous vision and ability to see beyond the view of most folks. I have not seen either of these fellows for many years but they flooded me with great problems from which I was able to develop insight I could not have had without them. Others with whom I have worked and from whom I have learned are Pranish Sinha, Ragavan Sreen, Anton Monk, and Ron Porat.

The third group of people I feel compelled to acknowledge are those with whom I initially interacted through a student-mentor relationship and have continued to interact as friend and colleague. I will limit this list to current and former students who have worked with me in the area of multirate signal processing. Maximilien d'Oreye de Lantramange and Michael Orchard are the best of the best. Dragan Vuletic and Wade Lowdermilk with whom I interact daily on applied problems are rising stars following other exceptional former students such as Bob Bernardi, and Christian Bettwieser and the line continues with Benjamin Egg, a new graduate student at its head. In thirty-five years of teaching I have touched the minds of many students whom I hope have enjoyed the encounters as much as I have. I know that many names are missing from my acknowledgment list and if you feel offended by this, don't be. Please attribute it to a state of absentmindedness rather than to one of intentional omission.

fredric j harris
San Diego, California

CHAPTER 1

Why Multirate Filters?



Why would we want to change the sample rate in a filter? There are two reasons. The first is performance. The second is cost. Multirate systems often perform a processing task with improved performance characteristics while simultaneously offering that performance at significantly lower cost than traditional approaches. *Multirate filters* are digital filters that operate with one more sample rate change embedded in the signal processing architecture. Occasionally, the use of a sample rate change in a filtering is the natural consequence of the signal processing chain. In other cases, the sample rate change is imposed to access the cost advantages related to multirate processing. We will develop examples of both scenarios throughout this book but will identify a few examples here.

1.1 Compact Disc 4-to-1 Oversample

A wonderful example of a multirate filtering application is the signal conditioning performed by a compact disc (CD) player. The CD player converts the digital representation of the music stored on the CD to analog audio for the listening pleasure of the CD user. Figure 1.1 presents the standard signal conditioning operations required when converting a digital signal to its equivalent analog representation. It entails a succession of three operators, a digital-to-analog converter (DAC), a sample and hold, and an analog smoothing filter. The DAC converts the succession of digital sample values to a succession of corresponding analog amplitudes. The sample and hold suppresses glitch transients in the analog amplitudes due to bit race conditions in the multibit conversion, while the smoothing filter suppresses out-of-band spectral components.

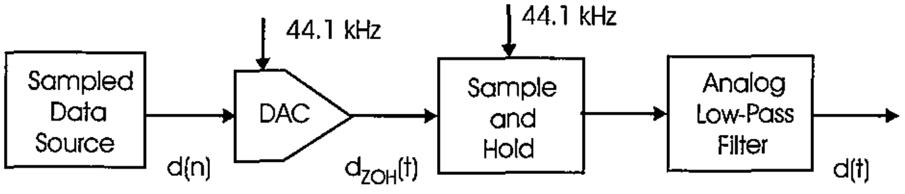


Figure 1.1 Signal Conditioning to Convert a Digital Signal to its Analog Representation

Figure 1.2 presents the time domain representation of the signal at successive points in the signal-conditioning path. Figure 1.3 presents the spectra corresponding to the time signals of Figure 1.2. The two-sided bandwidth of the input signal is 40 kHz. The CD sample rate is 44.1 kHz, which results in the spectral replicates, due to the sampling process being located at multiples of 44.1 kHz. The DAC replaces each sample value by a proportional DC term valid for the interval between sample values. The process of replacing a sample value with a data scaled rectangle is described as a *zero-order hold* (ZOH). The spectral response of the ZOH is a sin(x)/x or sinc(π f/fs) function with zero crossings located at multiples of the sample rate. As seen in the spectral plot, the zeros of the sinc suppress the center of the spectral replicates, the side-lobes of the sinc attenuate the remaining spectral mass, and the sinc main lobe response distorts the desired baseband spectral region.

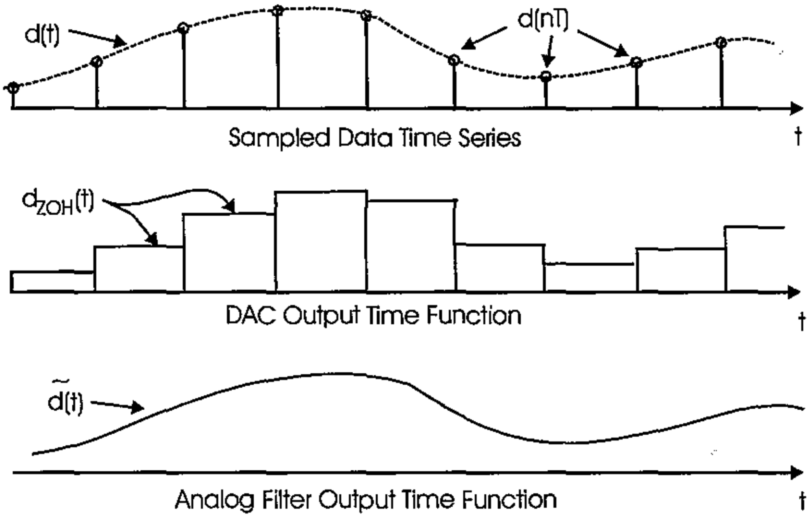


Figure 1.2 Input and Samples of Input, DAC Output, and Filtered Output Signals

The analog-smoothing filter must satisfy a number of signal conditioning requirements. The first is to finish the incomplete filtering task started by the DAC, the suppression of the residual spectral replicates. The filter required to perform this task is of high-order (N≅10) and consequently relatively expensive. The high order filter is required to obtain a narrow transition bandwidth, starting at 20 kHz and achieving the required 80-dB attenuation beyond 24 kHz for a composite DAC and filter attenuation of 96-dB. The second requirement is the correction of the in-band sin(x)/x distortion, which is accomplished by having a pass band response matching the inverse of the sinc response over the signal bandwidth. The third requirement is that the filter should not introduce severe group delay distortion in the vicinity of its band edge. A desired constraint is that the filter be one of a pair with matching gain and phase for the stereo audio signals. And finally, the last requirement is that the pair of filters costs an absurdly low amount, say less than $0.50. If you are still chuckling over this list of requirements you realize that they are not realistic specifications for an analog filter.

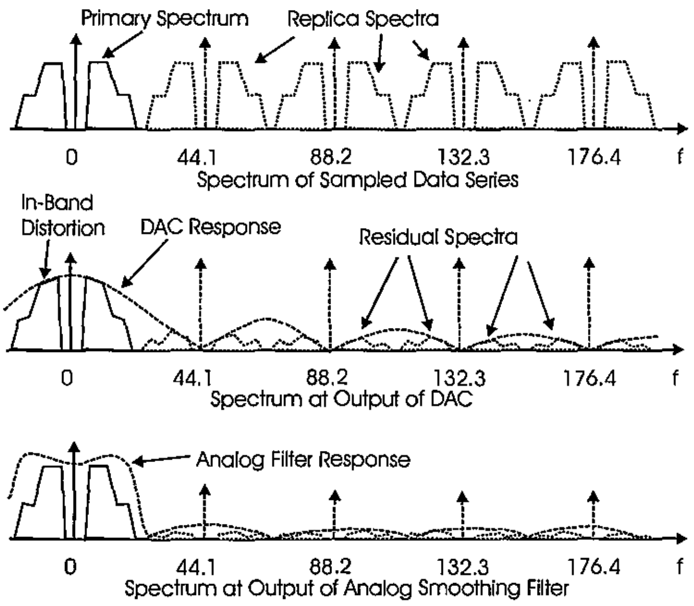


Figure 1.3 Spectra of Sampled Input, DAC Output, and Filtered Output

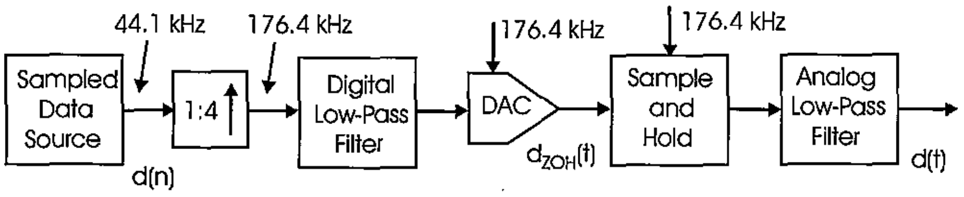


Figure 1.4 Modified Signal Conditioning to Convert Digital Signal to Analog Representation

When faced with a problem we can't solve, we invoke a trick that *Star Trek* enthusiasts will recognize as the *Kobyashi Maru Scenario* (*Wrath of Khan*). When faced with an unsolvable problem, change it into one you can solve, and solve that one instead. Figure 1.4 presents the modified signal conditioning tasks that employ this trick and inexpensively enable the conversion of a sampled data representation of a signal to its analog representation.

Figure 1.5 presents the time domain representation of the signal at various points in the modified signal-conditioning path, while Figure 1.6 presents the spectral description of the corresponding signals shown in Figure 1.5. In the modified processing we digitally raise the sample rate of the input sequence by a factor of 4 from 44.1 kHz to 176.4 kHz. This permits 4-spectral copies of the spectra to be presented to the digital filter with three of them to be suppressed in the sampled data domain. While we can't build an analog filter with the desired specifications listed in the previous paragraph, we have no difficulty meeting these specifications in the digital filter. The output of the digital filter contains interpolated sample points at four times the original input sample rate. The spectra of the oversampled sequence are now separated by 176.4 kHz rather than the original 44.1 kHz. The spectral response of the DAC operating at the new output sample rate has a much easier task of suppressing the spectral replicates with fractional bandwidth one-eighth of the wider sinc main lobe width as opposed to the original one-half of the original sinc main lobe width. The transition bandwidth of the analog filter required to finish the incomplete spectral suppression is now 176.4 – 44.1 or 132.3 kHz, which is nearly 4 octaves, as opposed to the original transition width of

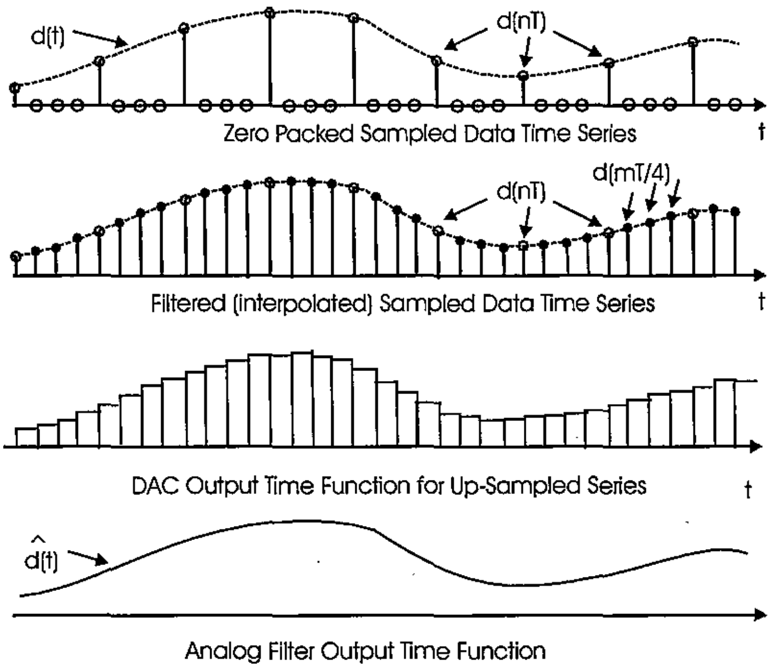

d(t)
d(nT)
Zero Packed Sampled Data Time Series
t
d(nT)
d(mT/4)
Filtered (interpolated) Sampled Data Time Series
t
DAC Output Time Function for Up-Sampled Series
t
$\hat{d}(t)$
Analog Filter Output Time Function

Figure 1.5 Input Samples Zero Packed 1-to-4, Interpolated 1-to-4 Samples of Input, DAC Output 1-to-4 Interpolated Samples, and Filtered Output

4.1 kHz which is 1/5 of an octave. With a significantly larger transition bandwidth the analog filter is of lower order and can be purchased at significantly reduced cost. The analog filter is no longer required to correct the DAC sinc distortion since the corrective predistortion is embedded in the digital up-sampling filter. In the process just described, the conversion from input digital to output analog representation is performed in two stages, a digital up sampling by a low-cost digital filter and a low-cost analog-smoothing filter. The CD players that invoke this option are identified as being 4-to-1 oversampled. We will see shortly that 4-to-1 oversampling is a common option in most DSP-based systems at the interface between the digital and analog versions of the signal. For instance, it is very common for digital modems to use 4-samples per symbol, a 4-times oversampled version of the data, to reduce the distortion due to the analog filtering as well as the cost of the analog filter following the DAC.

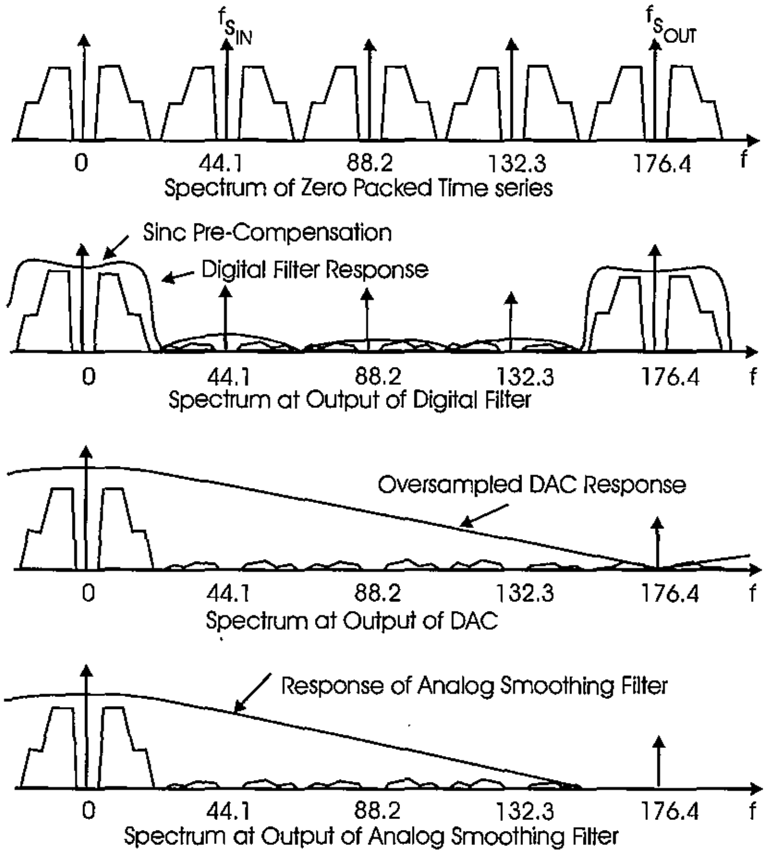

$f_{S_{IN}}$
$f_{S_{OUT}}$
0
44.1
88.2
132.3
176.4
f
Spectrum of Zero Packed Time series
Sinc Pre-Compensation
Digital Filter Response
Spectrum at Output of Digital Filter
Oversampled DAC Response
Spectrum at Output of DAC
Response of Analog Smoothing Filter
Spectrum at Output of Analog Smoothing Filter

Figure 1.6 Spectra: Sampled Input, 1-to-4 Up Sampled Output, DAC Output, and Filtered Output

1.2 ANTI-ALIAS FILTERING

This example is similar to the previous example in the sense we oversample to reduce the complexity and cost of an analog filter. This differs from the previous example by being its dual. The standard task we address here is the signal conditioning required to convert an analog signal to a digital signal. Figure 1.7 presents a block diagram of the primary components similar to that shown in Figure 1.1 for the digital-to-analog signal conditioning. Note the addition of an *automatic gain control* (AGC) block missing from Figure 1.1. Here we see the analog low pass filter that performs the anti-alias function, the AGC block that adjusts the input signal level to match the dynamic range of the sample and hold (S&H), and the analog-to-digital converter (ADC). The sample rate of the S&H and ADC must satisfy the Nyquist criterion of the input signal. The Nyquist rate of a high-quality data collection or recording system is the signal's two-sided bandwidth plus the transition bandwidth of the anti-alias filter. The two-sided signal bandwidth for high-end audio is 40 kHz, and standard sample rates are 44.1 kHz for the CD and 48 kHz for Digital Audio Tape (DAT). The transition bandwidths of the analog filters for CD and DAT are 4.1 kHz and 8 kHz respectively.

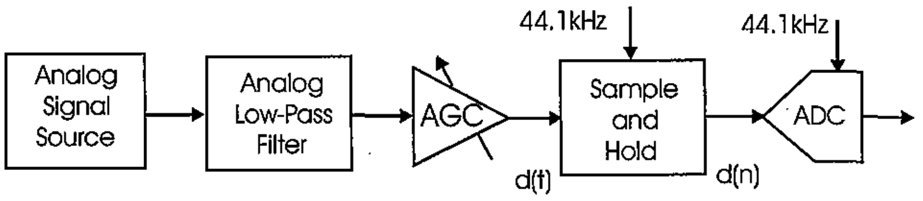


Figure 1.7 Signal Conditioning to Convert an Analog Signal to its Digital Representation

Figure 1.8 presents the frequency domain representation of the signal at successive points in the signal-conditioning path. The two-sided bandwidth of the input signal is 40 kHz. The CD sample rate is 44.1 kHz. The analog anti-alias filter has a pass band that extends to 20 kHz and a stop band that must attenuate adjacent spectra by 96-dB starting at 24.1 kHz. This filter must meet the performance requirements listed for the smoothing filter in the previous section. Here again, we find the requirements to be unrealistic. In particular, the narrow transition bandwidth of the filter requires it to be a high-order filter. High-order filters are characterized by severe group delay distortion near the band edge. High-order filters generally do not meet consumer price requirements and it is unlikely that the cost of the filter pair will meet the desired $0.50 target price.

Figure 1.9 presents the modified signal-conditioning task that uses multirate signal processing to enable a low-cost analog filter to perform the anti-aliasing task. Here we use a low-cost analog filter with wide transition bandwidth. The sample rate of the signal must now be increased to account for the wider transition bandwidth and for this example is 176.4 kHz, which is 4-times oversampled relative to the desired sample rate. A digital filter that meets the desired filter specifications filters the 4-times oversampled sequence. After filtering, the filtered sequence is oversampled and the sample rate is reduced by a factor of 4 from 176.4 kHz to 44.1 kHz. Reducing the sample rate by the factor of 4 reduces the distance between the spectral copies from the input sample rate separation of 176.4 kHz to the output sample rate separation of 44.1 kHz.

Figure 1.10 presents the frequency description of the signal at various positions in the modified signal-processing path. The analog anti-aliasing filter in the modified form has a

132 kHz transition bandwidth as opposed to the 4.1 kHz transition bandwidth of the non-oversampled version shown in Figure 1.5. Consequently, the degree of the analog filter can be reduced from a 10th order to a 4th order. The lower degree filter will exhibit reduced group delay distortion. If the delay distortion has to be eliminated, the digital filter can be designed to absorb the phase response required to equalize the composite analog and digital filter to obtain linear phase.

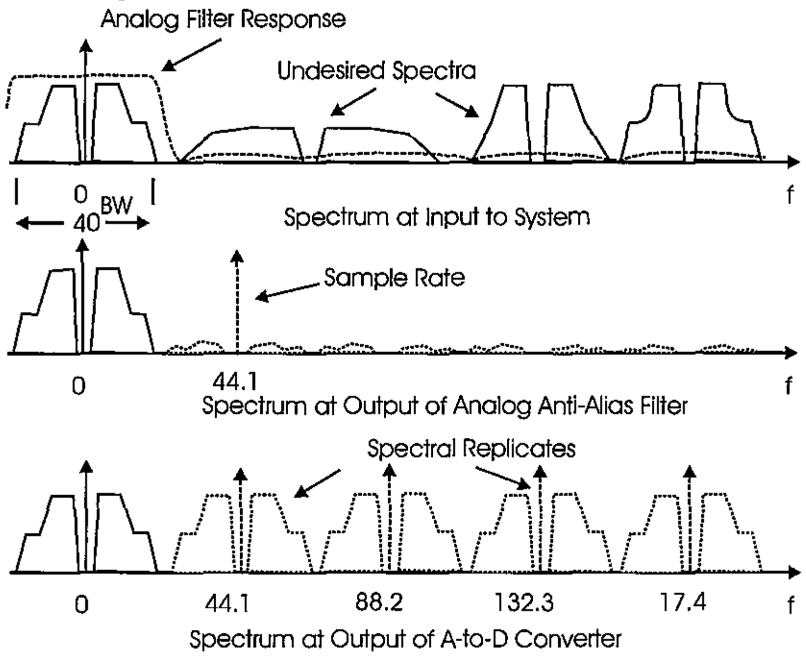

Analog Filter Response
Undesired Spectra
0
BW
40
f
Spectrum at Input to System
Sample Rate
0
44.1
f
Spectrum at Output of Analog Anti-Alias Filter
Spectral Replicates
0
44.1
88.2
132.3
17.4
f
Spectrum at Output of A-to-D Converter

Figure 1.8 Spectra of Input Signal, Analog Filtered Analog Signal, and Sampled Input Signal

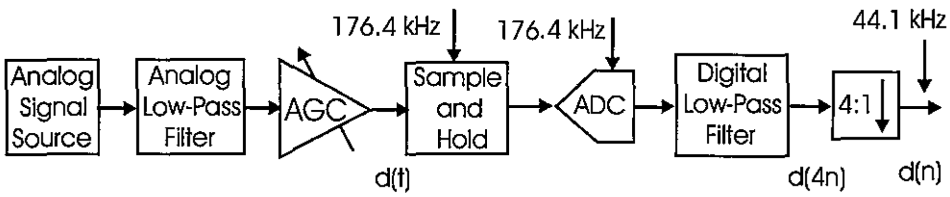


Figure 1.9. Modified Signal Conditioning to Convert an Analog Signal to its Digital Representation

Many DSP-based systems collect data with an oversampled ADC. This is often done to reduce the cost of the analog anti-aliasing filter while preserving the quality of the signal being processed by the filter. A high-quality digital anti-alias filter suppresses the spectral content in the excess spectral span of the extra-wide transition bandwidth. The sample rate of the processed data is then reduced to the desired output sample rate. In some systems the excess data rate is maintained at the 4-times oversampled rate to support additional processing related to interpolation from one sample rate to an arbitrary sample rate. As we will see in later sections, the processing task of an interpolator is reduced significantly when the input data is initially oversampled by a factor of 4.

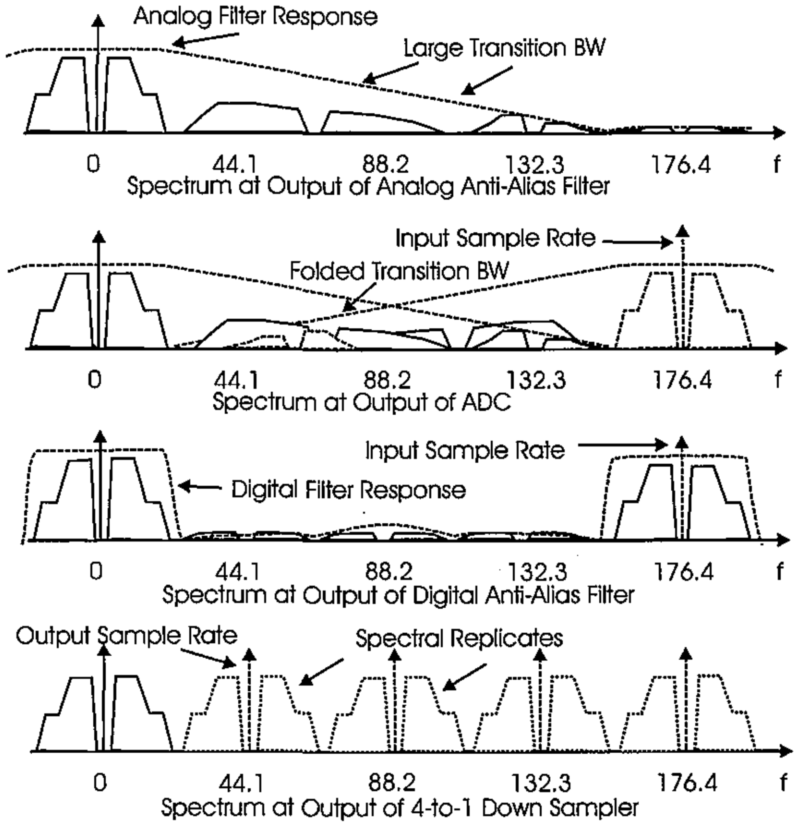


Figure 1.10 Spectra: Filtered Input, 4-Times Oversampled Sampled Input, Digitally Filtered Output, and 4-to-1 Down Sampled Output

References

Candy, James, and Gabor Temes. *Oversampled Delta-Sigma Data Converters*, Piscataway, NJ: IEEE Press, 1992.

Nguyen, Khiem, Robert Adams, and Karl Sweetland, "A 113-dB SNR Oversampled Sigma-Delta DAC for CD/DVD Application," *IEEE Transactions on Consumer Electronics*, Volume 44, Issue 3, Aug. 1998, pp. 1019-1023.

Norsworthy, Steven, Richard Schreier, and Gabor C. Temes, *Delta Sigma Data Converters: Theory, Design and Simulation,* Piscataway, NJ. IEEE Press, 1997.

Pohlman, Ken, *Principles of Digital Audio*, Indianapolis, Howard Sams & Co., 1985

Watkinson, John, *The Art of Digital Audio*, London & Boston, Focal Press, 1989.

Problems

1.1 Determine the order of an analog Butterworth filter that can be used as the anti-alias filter for a CD quality signal. This filter must have its 3-dB pass band edge at 20 kHz and its 96-dB stop band edge at 24.1 kHz. If each component, capacitor, or inductor in this filter costs $0.05, estimate the cost for a pair of anti-alias filters. A digital filter, oversampled by 10 and satisfying the same performance requirements, can be used as an estimate of the required order analog filter.

1.2 Determine the order of an analog elliptic filter that can be used as the anti-alias filter for a CD quality signal. This filter must have its 0.1-dB pass band edge at 20 kHz and its 96-dB stop band edge at 24.1 kHz. If each component, capacitor, or inductor in this filter costs $0.05, estimate the cost for a pair of anti-alias filters. A digital filter, oversampled by 10 and satisfying the same performance requirements, can be used as an estimate of the required order analog filter.

1.3 Determine the order of an analog elliptic filter that can be used as an anti-alias filter for a 4-times oversampled CD quality signal. This filter must have its 0.1-dB pass band edge at 20 kHz, and its 96-dB stop band edge at 156.4 kHz. If each component, capacitor, or inductor of this filter costs $0.05, estimate the cost for a pair of anti-alias filters. A digital filter, oversampled by 10 and satisfying the same performance requirements, can be used as an estimate of the required order analog filter.

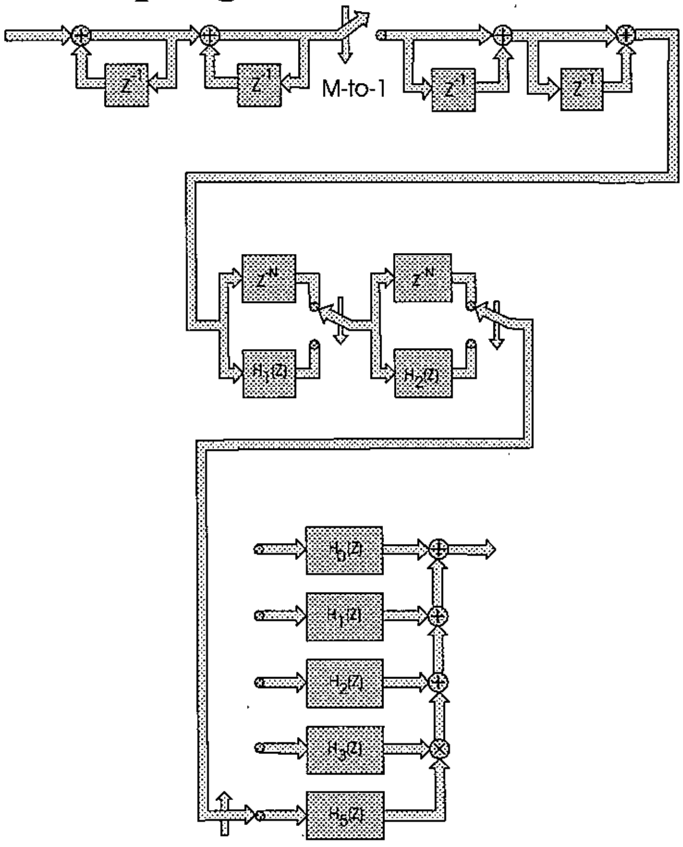
1.4 Determine the order of an analog elliptic filter that can be used as an anti-alias filter for an 8-times oversampled CD quality signal. This filter must have its 0.1-dB pass band edge at 20 kHz, and you have to determine the new sample rate and the spectral location of its 96-dB stop band edge. If each component of this filter, capacitor, or inductor costs $0.05, estimate the cost for a pair of anti-alias filters. A digital filter, oversampled by 10 and satisfying the same performance requirements, can be used as an estimate of the required order analog filter.

1.5 A signal uniformly occupying a full bandwidth of ±20 kHz is sampled at 48 kHz. The signal samples are presented to a D-to-A converter, which performs the task of a ZOH with a frequency response equal to $\sin(\pi f/fs)/(\pi f/fs)$. To better appreciate the distortion and incomplete spectral suppression offered by the ZOH, generate an annotated and properly scaled figure that shows the in band spectral droop due to the main lobe response and the residual spectra of the first spectral copies centered at 48 kHz.

1.6 A signal uniformly occupying a full bandwidth of ±20 kHz originally sampled at 48 KHz is digitally up sampled 4-times to 196 kHz. The signal samples are presented to a D-to-A converter, which performs the task of a ZOH with a frequency response equal to $\sin(\pi f/fs)/(\pi f/fs)$. To better appreciate the improved levels of distortion and incomplete spectral suppression offered by the ZOH, generate an annotated and properly scaled figure that shows the in band spectral droop due to the main lobe response and the residual spectra of the first spectral copies centered at 196 kHz.

CHAPTER 2

The Resampling Process



Central to multirate filters is the concept of sample rate change. In preparation for the study of filters that participate in this process we first address the process of resampling a sampled signal as opposed to the process of sampling a continuous time signal. When a continuous time signal is sampled there are no restrictions on the sample rate or the phase of the sample clock relative to the time base of the continuous time signal. On the other hand, when we resample an already sampled signal, the output sample locations are intimately related to the input sample positions. The resampling operation offers a convenient visualization of a precursor process to the desired process of changing the sample rate of a time series. A resampled time series contains samples of the original input time series separated by a set of zero valued samples. The zero valued time samples can be the result of setting a subset of input sample values to zero or as the result of inserting zeros between existing input sample values. Both options are shown in Figure 2.1. In the first example shown here, the input sequence is resampled 4-to-1, keeping every fourth input sample starting at sample index 0 while replacing the interim samples with zero valued samples. In the second example, the input sequence is resampled 1-to-2, keeping every input sample but inserting a zero valued sample between each input sample. These two processes are sometimes called down sampling and up sampling respectively.

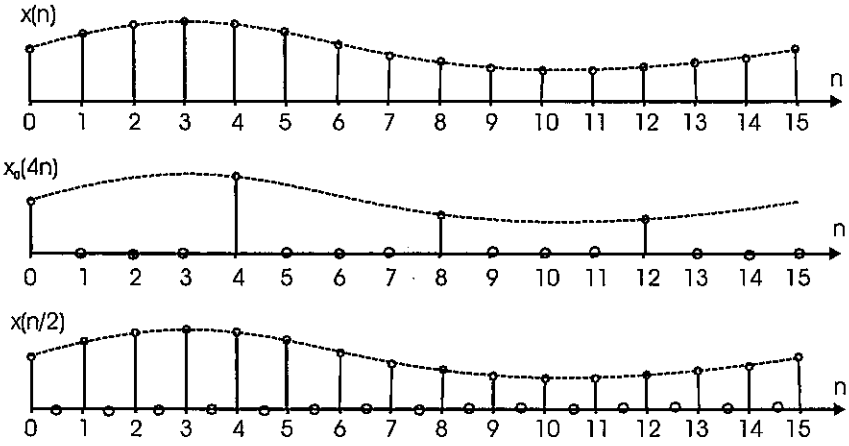


Figure 2.1 Resampling by Zeroing Sample Values and by Inserting Zero Valued Samples

Two examples of 4-to-1 resampling of a time series are shown in Figure 2.2. The first resampling algorithm keeps every 4th sample starting at index 0 while setting the interim samples to zero. The second resampling algorithm keeps every 4th sample starting at index 1 while setting the interim samples to zero. In general there are Q initial starting locations for a Q-to-1 down sampler. The spectra of the Q different down sampled versions of the input signal have different phase profiles related to the time offset of the initial starting point relative to the time origin. The different phase profiles play a central role in multirate signal processing.

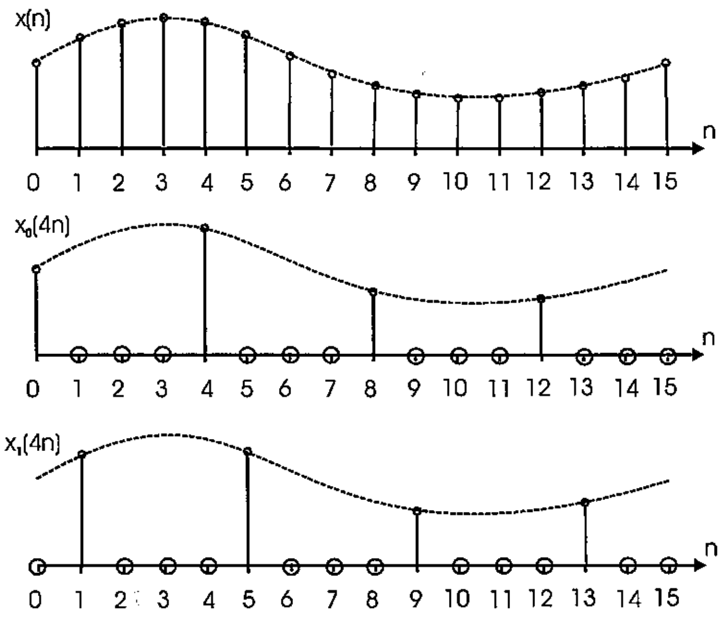


Figure 2.2 Two Examples of 4-to-1 Down Sampling of Input Series

2.1 The Sampling Sequence

The process of performing sampling in the continuous time domain is often described with the aid of the generalized sampling function, a sequence of delayed impulses, as shown in (2.1).

$$s_T(t) = \sum_n \delta(t - nT) \tag{2.1}$$

In a similar manner, the resampling operation can be described with the aid of the discrete time sampling sequence formed by the inverse Discrete Fourier Transform (DFT) shown in (2.2).

$$S_M(n) = \frac{1}{M} \sum_{m=0}^{M-1} \exp(j\frac{2\pi}{M}mn) \tag{2.2}$$

The sequence $S_M(n)$ is seen to be the sum of M complex sinusoidal sequences of amplitude 1/M with frequencies equally spaced around the unit circle at multiples of $(2\pi/M)$. The sum formed by the relationship shown in (2.2) is the sequence shown in (2.3).

$$S_M(n) = \begin{cases} 1 & \text{For } n = \upsilon M,\ \upsilon \text{ an integer} \\ 0 & \text{Otherwise} \end{cases} \tag{2.3}$$

The DFT relating the time and spectral description of the sampling sequence for $S_5(n)$ is shown in Figure 2.3.

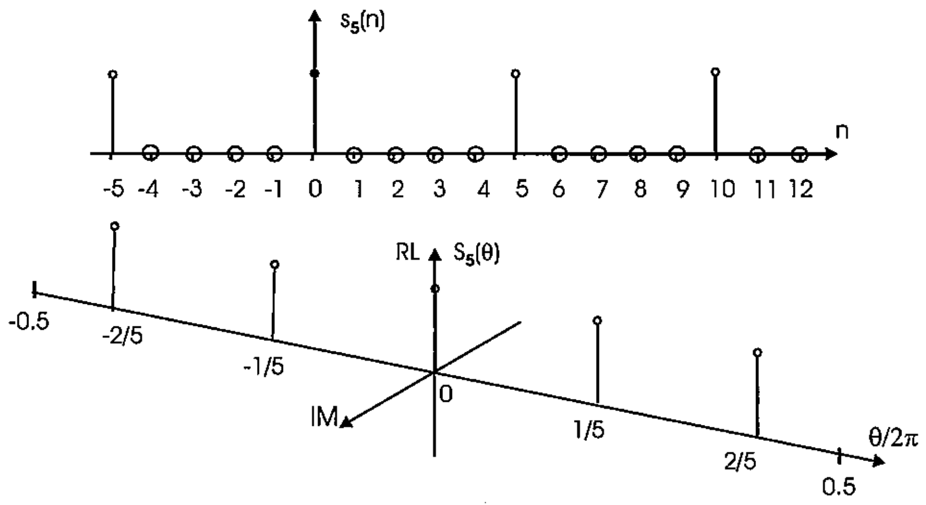


Figure 2.3 Sampling Sequence $S_5(n)$ in Time and Frequency Domain

The sampling sequence can be offset from index 0 to index r as shown in (2.4).

$$S_M(n-r) = \frac{1}{M}\sum_{m=0}^{M-1} \exp\left(j\frac{2\pi}{M}m(n-r)\right) \tag{2.4}$$

The sequence $S_M(n-r)$ is seen to be the sum of M complex sinusoidal sequences with amplitude (1/M), with phase angles $\exp(-j\, r\, m2\pi/M)$ at frequencies equally spaced around the unit circle at multiples of $(2\pi/M)$. The sequence satisfies the relationship shown in (2.5).

$$S_M(n-r) = \begin{cases} 1 & \text{For } (n-r) = \upsilon M, \text{ or } n = r+\upsilon M,\ \upsilon \text{ an integer} \\ 0 & \text{Otherwise} \end{cases} \tag{2.5}$$

The DFT relating the time and spectral description of the sampling sequence for $S_5(n-1)$ is shown in Figure 2.4. Note that the time offset in the sampling sequence has resulted in a phase rotation of the spectral components forming the offset sequence.

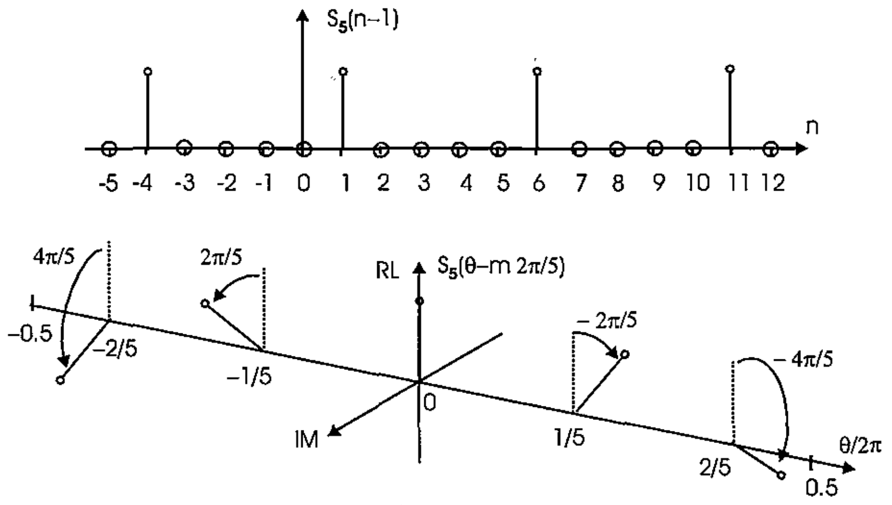


Figure 2.4 Sampling Sequence $S_5(n-1)$ in Time and Frequency Domain

The resampling operation can be visualized as the product of the input sequence x(n) and the sampling sequence $S_M(n)$ to obtain the output sequence $x_0(Mn)$. The time domain product causes a frequency domain convolution. Hence the spectrum of the resampled signal contains M offset replicates of the input spectrum. The spectrum of the resampled sequence is shown in (2.6).

$$X_0\{\exp(j\theta)\} = \frac{1}{M}\sum_{k=0}^{M-1} X\{\exp[j(\theta - \frac{2\pi}{M}k)]\} \tag{2.6}$$

The time and frequency representation of an oversampled time series is shown in Figure 2.5. The time and frequency version of this series resampled by the resample sequence $S_5(n)$ is shown in Figure 2.6. The fivefold replication of the input spectrum is seen in the spectral representation of the resampled sequence. The time and frequency version of the input series resampled by the resample sequence $S_5(n-1)$ is shown in Figure 2.7. Here too we see the fivefold replication of the input spectrum is seen in the spectral representation of the resampled sequence. The difference in the spectral replicates of Figure 2.6 and Figure 2.7 is the phase shifts of the spectral regions inherited from the spectral terms phase shifted by the one-sample offset of the resampled time series. Note there has not been a sample rate change as we converted the input time series to either of the resampled time series. Sample rate changes are yet to come.

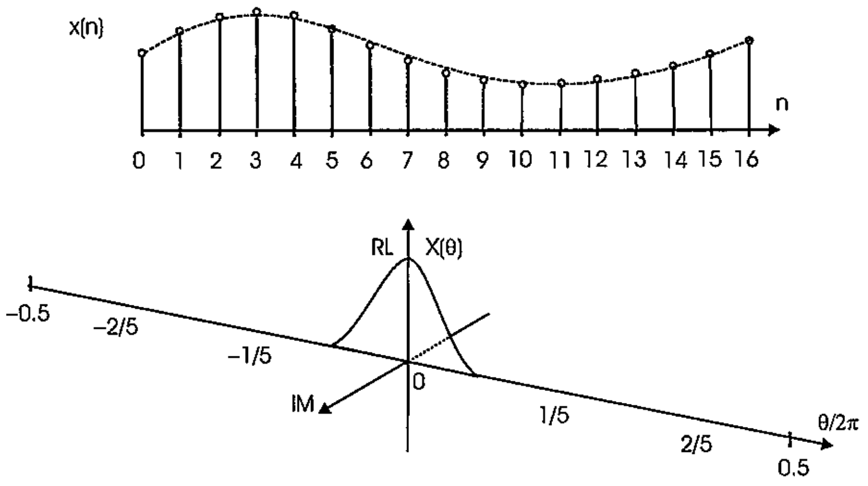


Figure 2.5 Time and Frequency Domain Representation of Sampled Sequence x(n)

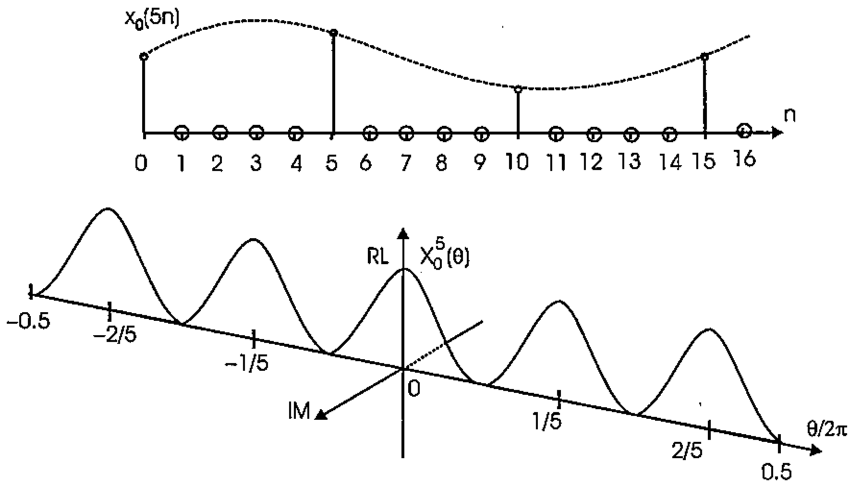


Figure 2.6 Time and Frequency Domain Representation of Sampled Sequence $x_0(5n)$

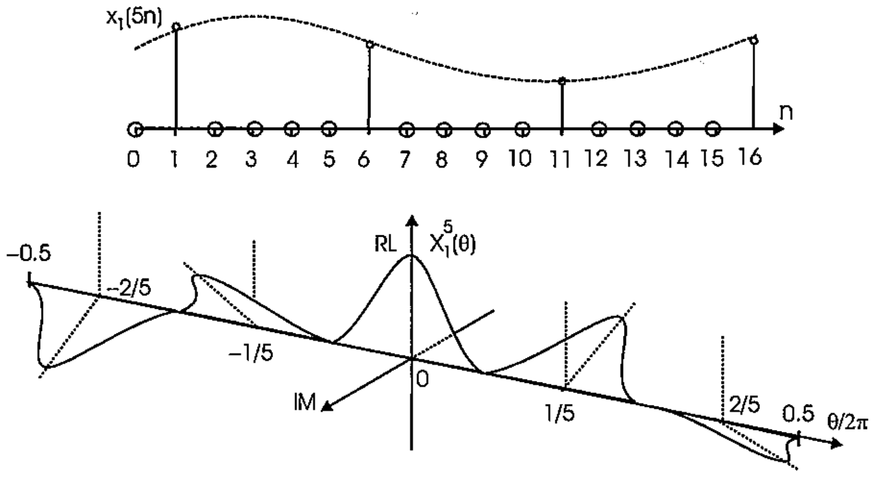


Figure 2.7 Time and Frequency Domain Representation of Sampled Sequence $x_1(5n)$

2.1.1 Modulation Description of Resampled Sequence

A different process can be used to describe the periodic replication of the baseband spectrum caused by the resampling process, illustrated in Figure 2.6. The time and frequency representation of a time series of the form illustrated in Figure 2.5 is shown in (2.7).

$$h(n) \Leftrightarrow H(\theta) \tag{2.7}$$

The spectral replicates indicated in Figure 2.6 are related to heterodyned versions of the input time series as indicated in (2.8).

$$h(n)\exp(jm\frac{2\pi}{M}) \Leftrightarrow H(\theta - m\frac{2\pi}{M}) \tag{2.8}$$

The replicated, translated, and phase shifted version of the resampled time series can be represented as a sum of heterodyned versions of the original baseband signal. This form is shown in (2.9).

$$\begin{aligned} h_r(nM) &= \frac{1}{M}\sum_{m=0}^{M-1} h(n)\exp(jmr\frac{2\pi}{M}) \\ H_r(\theta) &= \frac{1}{M}\sum_{m=0}^{M-1} H(\theta-m\frac{2\pi}{M})\exp(jmr\frac{2\pi}{M}) \end{aligned} \tag{2.9}$$

The importance of the relationship described in (2.9) is that the resampling process appears to be equivalent to the spectral translation of the baseband spectrum. This connection suggests that resampling can be used to affect translation of spectral bands, up and down conversion, without the use of sample data heterodynes. In fact we often embed the spectral translation of narrowband signals in resampling filters and describe the process as *aliasing*. Derivations that use this relationship will be presented in later sections. A final comment about the resampling process is that it can be applied to a time series or to the impulse response of a filter, which, of course, is simply another time series. When the resampling process is applied to a filter, the architecture of the filter changes considerably. The altered form of the filter is called a multirate filter.

2.2 What Is a Multirate Filter?

Multirate filters are digital filters that contain a mechanism to increase or decrease the sample rate while processing input sampled signals. The simplest such filter performs integer up sampling of 1-to-P or integer down sampling of Q-to-1. By extension, a multirate filter can employ both up sampling and down sampling in the same process to affect a rational ratio sample rate change of P-to-Q. More sophisticated techniques exist to perform arbitrary and perhaps slowly time varying sample rate changes. The integers P and Q may be selected to be the same so that there is no sample rate change between input and output but rather an arbitrary time shift or phase offset between input and output sample positions of the complex envelope. The sample rate change can occur at a single location in the processing chain or can be distributed over several subsections.

A number of symbols have been used to represent the sample rate change element in a block diagram. Three of the most common symbols are shown in Figure 2.8 for the down sampling element and for the up sampling element. The two elements are duals and the systems that employ them for sample rate changes will also be seen to be duals.

Conceptually, the process of down sampling can be visualized as a two-step progression indicated in Figure 2.9. There are three distinct signals associated with this procedure. The process starts with an input series x(n) that is processed by a filter h(n) to obtain the output sequence y(n) with reduced bandwidth. The sample rate of the output sequence is then reduced Q-to-1 to a rate commensurate with the reduced signal bandwidth. In reality the processes of bandwidth reduction and sample rate reduction are merged in a single process called a multirate filter. The bandwidth reduction performed by the digital filter can be a low pass process or a band-pass process.

The time and spectral descriptions of the three signal points in a 3-to-1 down sampling version of Figure 2.9 are shown in Figure 2.10. Here the filter limits the bandwidth to the band of interest and initially computes an output sample for each input sample. Reduction of output bandwidth to a third of the input bandwidth permits a corresponding reduction in output sample rate. This is accomplished by having the resample switch output selected samples at the reduced rate and replacing discarded samples with zero-valued output samples. Since these zero-valued samples carry no information about the signal or its bandwidth we are free to discard the zero-valued replacement samples. In reality we do not insert the zero-valued samples since they are immediately discarded.

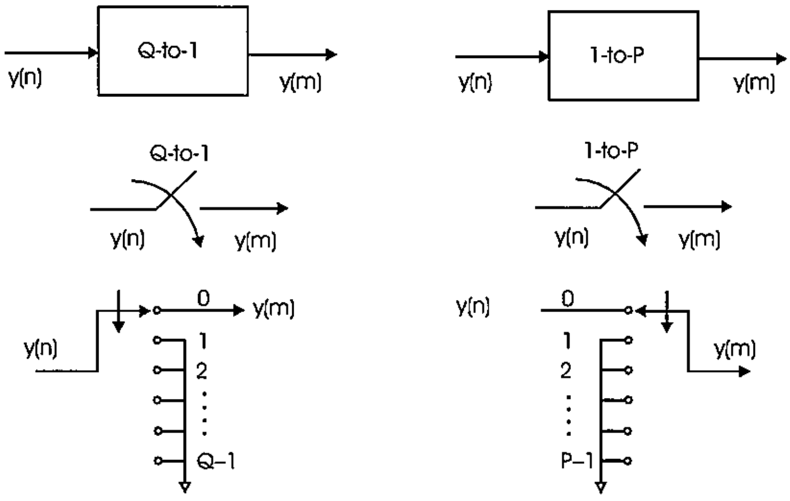


Figure 2.8 Symbols Representing Down Sampling and Up Sampling Elements

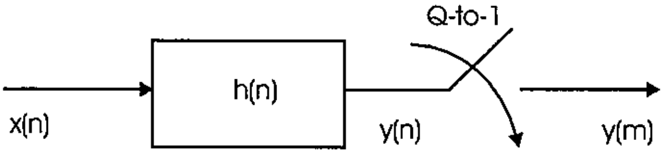


Figure 2.9 Down Sampling Process Filtering and Sample Rate Reduction

In a similar fashion, the process of up sampling can be visualized as a two-step process indicated in Figure 2.11. Here too there are three distinct time series. The process starts by increasing the sample rate of an input series x(n) by resampling 1-to-P. The zero-packed time series with P-fold replication of the input spectrum is processed by a filter h(n) to reject the spectral replicates and output sequence y(m) with the same spectrum as the input sequence but sampled at the P-times higher sample rate. In reality the processes of sample rate increase and selected bandwidth rejection are merged in a single process called a multirate filter. The bandwidth rejection performed by the digital filter can be a low pass or a bandpass process.

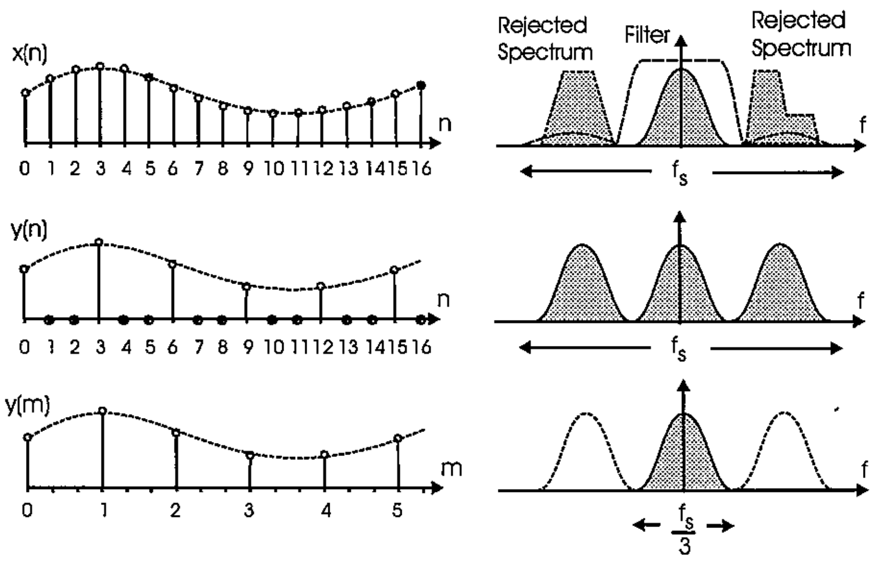


Figure 2.10 Time Series and Spectra for Signal Points in 3-to-1 Down Sample Process

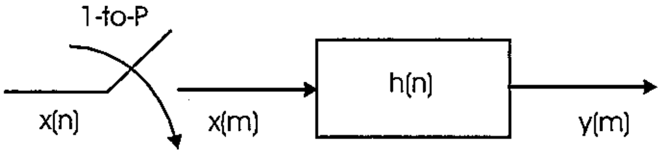


Figure 2.11 Up Sampling Process Filtering and Sample Rate Reduction

The time and spectral descriptions of the three signal points in a 1-to-5 up sampling process of Figure 2.11 are shown in Figure 2.12. Here the resampler increases the sample rate by a factor of 5 by inserting four zero-valued samples between input samples. The filter limits the bandwidth to the band of interest and computes output samples at an increased rate (5/1) relative to input rate, replacing the zero-valued samples with interpolated values. Since these zero-valued samples do not contribute to the filter output they are usually not inserted in the input data sequence. Their presence here is to give us perspective and addressing guidance when forming the multirate filter.

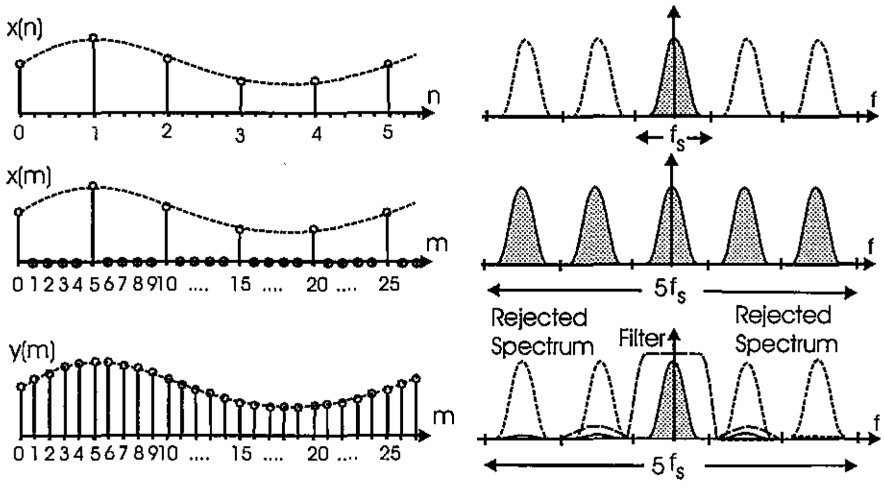


Figure 2.12 Time Series and Spectra for Signal Points in 1-to-5 Up Sample Process

2.2.1 Properties of Resamplers

In coming sections we will be manipulating and rearranging the processes of resampling and filtering. A useful visualization tool that we have developed through exposure to linear systems is the block diagram and signal flow representations of digital filters. The block diagram is, of course, an equivalent (right brain) description of sets of difference equations connecting the variables in a set of difference equations defining the filter. The building blocks we use in a linear time invariant filter structure are delay lines, multipliers, and adders. We now add to this list the resampler, and identify relationships satisfied by the resamplers as they interact with traditional block diagram entities. We will present the relationships in dual pairs, the first for the up sampler, and the second for the down sampler.

Since the addition of scaled sequences occurs without concern for their sample rate, the two operations of scaling and summing can commute. Figure 2.13 demonstrates the reordering of a scaled sum and a down sampler. Similarly, since the application of a scale factor to a sequence occurs without regard to the sample rate, the two operations of up sampling and scaling can also commute. Figure 2.14 demonstrates the reordering of up sampling and scaling. Note that the relationships shown in Figures 2.13 and 2.14 are dual signal flow graphs. Dual graphs are formed by replacing nodes with summations, and summations with nodes, reversing the direction of the signal flow, and then interchanging input and output ports.

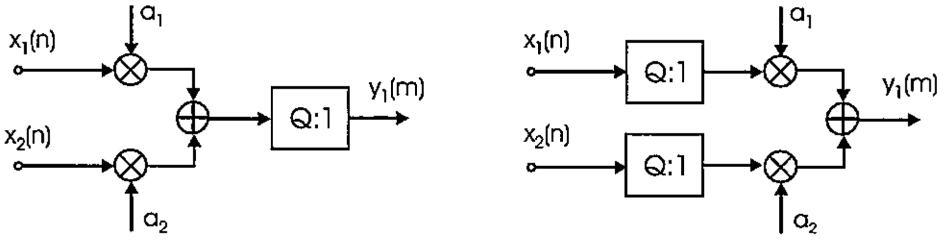


Figure 2.13 Down Sampled Sum of Scaled Sequences Equivalent to Sum of Scaled Down Sampled Sequences

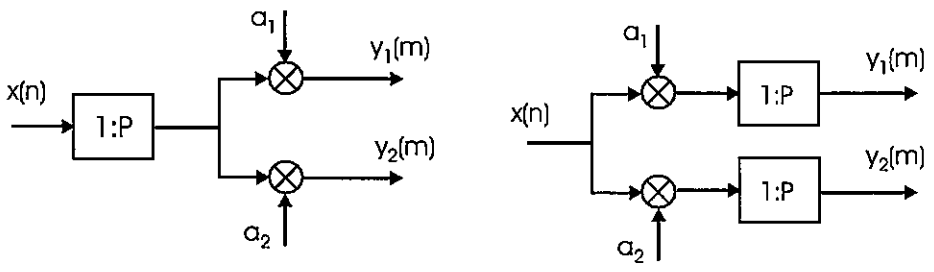


Figure 2.14 Scaled Up Sampled Sequences Equivalent to Up Sampled Scaled Sequences

Figure 2.15 illustrates that Q-units of delay followed by a Q-to-1 down sampler is the same as 1-unit of delay following a Q-to-1 down sampler. This is true because Q clock cycles at the input to the resampler span the same time interval as one clock cycle at the output of the resampler. The resampler changes the number of samples in an interval but does not change the length of the interval. Figure 2.16 illustrates the dual relationship that P-units of delay following a 1-to-P resampler is the same as a 1-to-P resampler following a 1-unit delay. This is true since P clock cycles at the output of the up sampler span the same time interval as one clock cycle at the input to the up sampler.

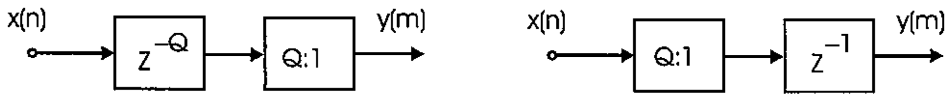


Figure 2.15 Q-units of Delay and Q-to-1 Down Sampler Is Equivalent to Q-to-1 Down Sampler and 1-unit of Delay.

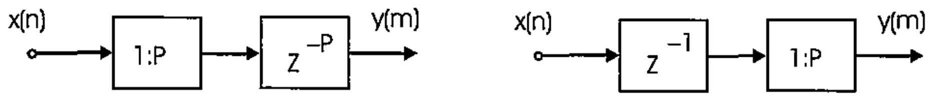


Figure 2.16 1-to-P Up Sampler Followed by P-Units of Delay Is Equivalent to 1-Unit of Delay and P-to-1 Up Sampler.

Figure 2.17 illustrates that a filter defined by polynomials in Z^Q followed by a Q-to-1 down sampler is equivalent to a filter defined by polynomials in Z following a Q-to-1 down sampler. We are essentially pulling the resampler through the filter and using the previous property to replace Q-delays at the input rate with 1-delay at the output rate. In a similar manner Figure 2.18 illustrates that a filter defined by a polynomial in Z^P following a 1-to-P up sampler is equivalent to a filter defined by a polynomial in Z preceding a 1-to-P up sampler. In both cases we can reverse the order of filtering and resampling so that the filtering is performed at the lower of the two rates. The equivalency of this interchange is known as the *noble identity*.

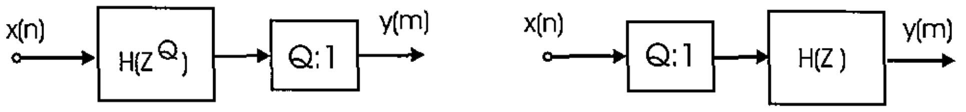


Figure 2.17 Exchanging Order of Filtering and Down Sampling

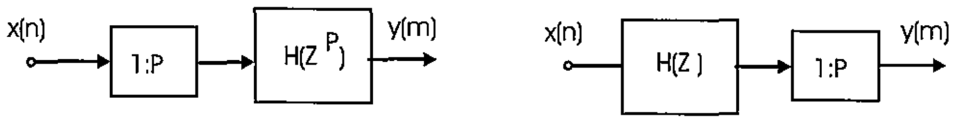


Figure 2.18 Exchanging Order of Up Sampling and Filtering

Figure 2.19 illustrates the interconnection of a 1-to-P up sampler and a Q-to-1 down sampler, cascaded to obtain a Q-to-P resampler. Reversing the order of the resamplers as shown is permitted if the integers P and Q are relatively prime and is not permitted if they share a common factor.

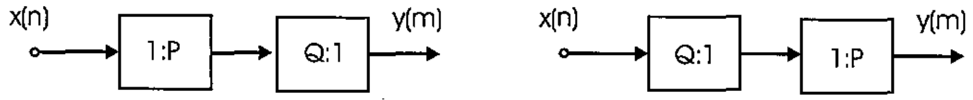


Figure 2.19 Reordering Cascade Up Samplers and Down Samplers

The interchange of order is useful in the following application where a multirate filter is to be designed that performs the resampling function of up P and down Q as shown in Figure 2.20. If P and Q are relatively prime we can pull the 1-to-P up sampler through the filter to its output port, then interchange the up sampler and down sampler and proceed to pull the down sampler through the filter to its input port. The desired effect obtained by interchanging the input up sampler with the output down sampler is that the filter operates at the minimum processing rate. We will illustrate this exchange in a later example.

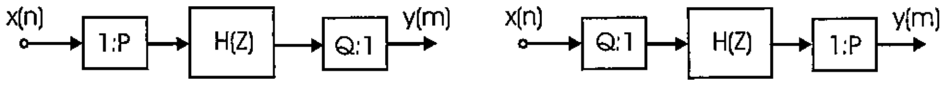


Figure 2.20 Interchange of Relative Prime Input and Output Resamplers

2.2.2 Examples of Resampling Filters

There is a very large class of multirate filters. Since this section has introduced what they are, we thought this would be an appropriate place to show important examples of multirate filters. Figure 2.21 presents three types of resampling filters used in down-sampling applications. The top subfigure shows an example of an M-stage polyphase down-sampling filter. The center subfigure is an example of a cascade of multiple half-band filters in which each stage performs a 1-to-2 down-sample operation. The bottom subfigure is a cascade of multiple digital integrators, a down sampler, and multiple digital differentiators configured in a structure known as the Hogenauer filter. When the down sampler resides at the output to the cascade chain the resulting filter is known as a Cascade Integrator Comb (CIC) filter.

Figure 2.22 presents the three types of resampling filters used in up-sampling applications. These structures are dual versions of their counterparts described in the down-sampling application. To form a dual filter, we replace summing junctions with nodes, replace nodes with summing junctions, and reverse the arrows in the signal flow as well as reverse the input and output terminals of the filter. In a linear time invariant filter structure, a filter and its dual perform the same function and are indistinguishable at their input and output terminals. In the periodically time varying filter structure, a filter and its dual perform opposite functions. If the filter performs down sampling, its dual performs up sampling.

The top subfigure of Figure 2.22 shows an example of an M-stage up-sampling polyphase filter. The center subfigure is an example of a cascade of half-band filters in which each stage performs a 2-to-1 down-sample operation. The bottom subfigure is a cascade of multiple digital differentiators, an up sampler, and multiple digital integrators configured in the Hogenauer filter structure. Here too, when the up sampler resides at the input to the cascade chain the resulting filter is known as a CIC filter.

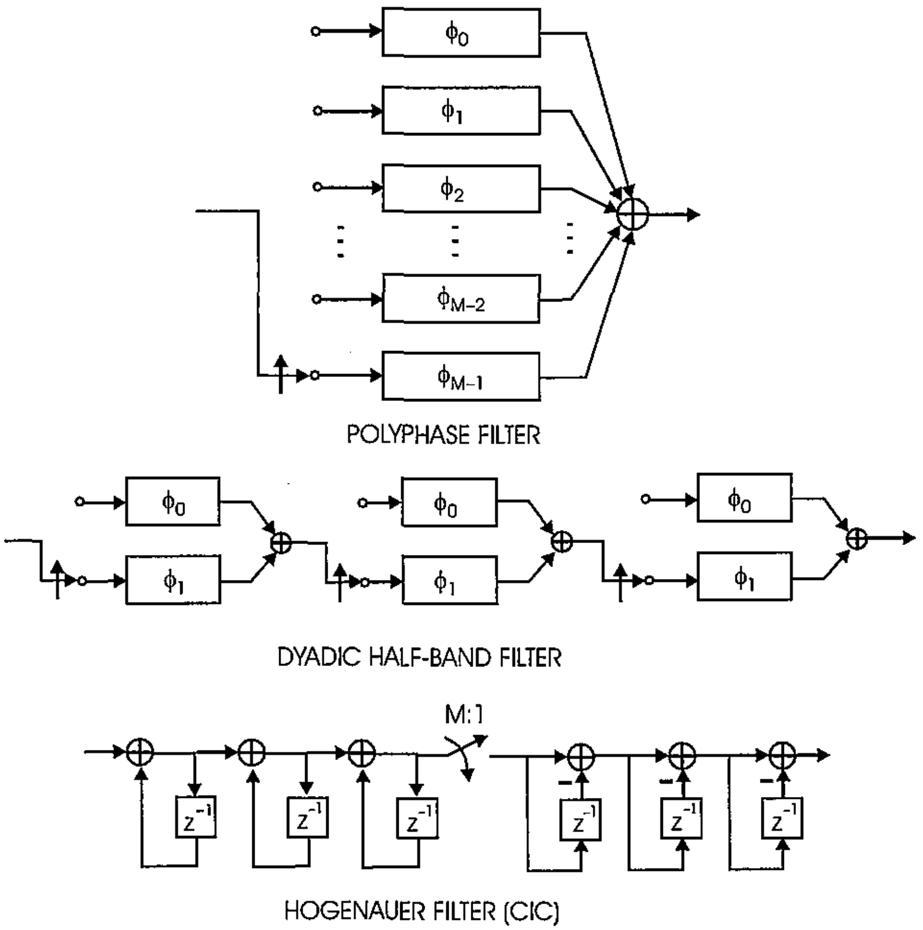
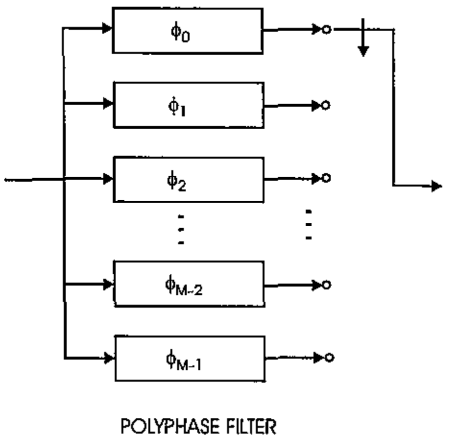
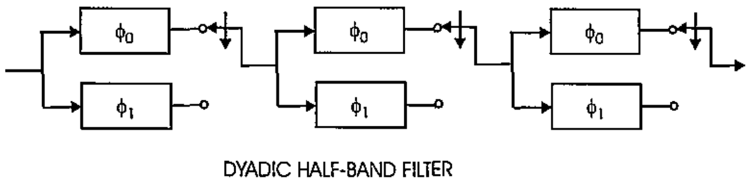


Figure 2.21 Standard Down-Sampling Filter Architectures: M-to-1 Polyphase Filter, 8-to-1 Dyadic Half-band Filter Chain, and M-to-1 Hogenauer (CIC with Embedded Resampler) Filter





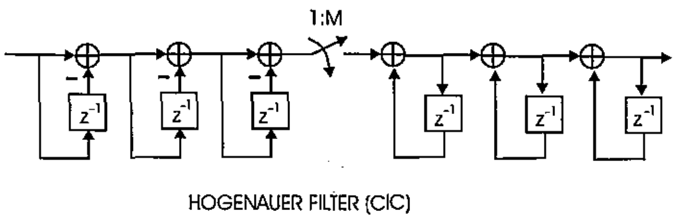


Figure 2.22 Standard Up-Sampling Filter Architectures: 1-to-M Polyphase Filter, 1-to-8 Dyadic Half-band Filter Chain, and 1-to-M Hogenauer (CIC with Embedded Resampler) Filter

2.3 Useful Perspectives for Multirate Filters

We have seen that when the resampling switch modifies a time series, the new time series exhibits spectral replicates at equally spaced spectral intervals. This means that samples of a sinusoid located at a particular center frequency observed at the input to a resampler results in a new series with replicates of the sinusoid located at other frequency locations. Here we have output frequencies not equal to the input frequencies. This behavior would not be possible if the system is a linear time invariant (LTI) process. Thus the multirate filter is not

LTI but in fact is a linear time varying (LTV) process. If we examine the filter structure of the polyphase filter shown in Figure 2.22, we see that the impulse response of this system depends on which subfilter is connected to the output port when the input impulse is presented to the filter. Since the output periodically revisits each commutator port, we say that the multirate filter is a periodically time varying (PTV) process.

The sample rate change that accompanies the multirate filter leads to an interesting quandary. We normally use the sample rate of a process as a reference interval when we discuss signal or process bandwidth. When we change the sample rate, we change the reference making it awkward to measure relative bandwidth with a flexible ruler. For a specific example consider Figure 2.23 which presents a simple time series, that of a low pass filter, sampled at 4-times the bandwidth, and at 5-times the bandwidth. Glancing at the spectra of the two versions of the filter, it appears at first that raising the sample rate reduced the bandwidth of the filter. In fact the bandwidth didn't change, the sample rate was increased and the image had to be rescaled to permit the larger spectral interval to fit into the same width display interval. We find it useful to avoid the use of the sample rate as a reference when discussing a multirate process but rather use the unchanged bandwidth as the reference. This advantage of this perspective will be obvious in applications presented in later sections.

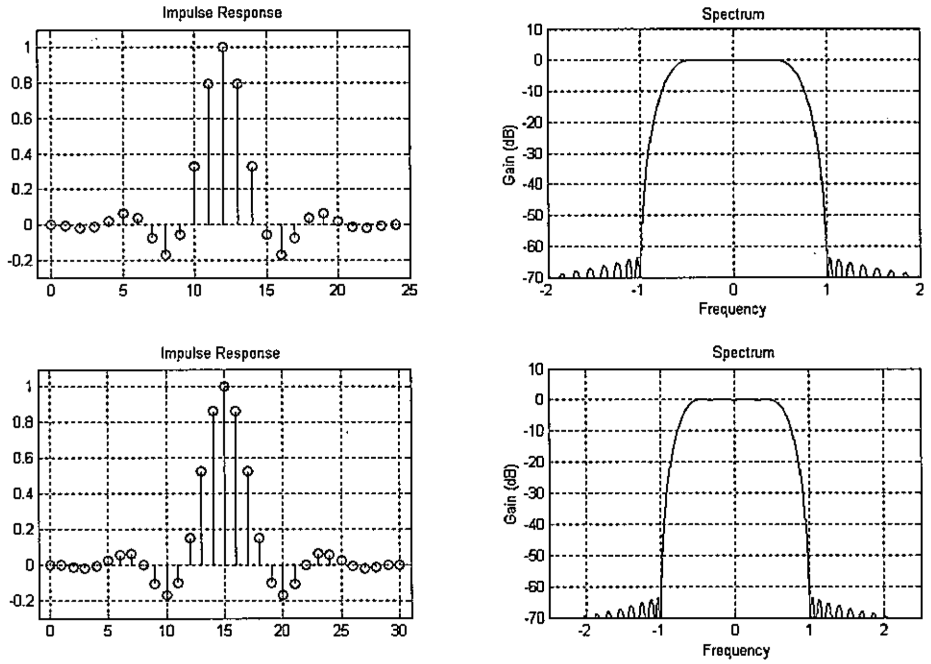


Figure 2.23 Filter Impulse Response and Filter Spectrum Sampled at 4-Times Bandwidth and at 5 -Times Signal Bandwidth

A similar relationship can be illustrated in the case of a single sine wave sampled at two different rates. Figure 2.24 presents four pairs of time and frequency descriptions of a sinusoid. The first pair presents the waveform and spectrum of a continuous sine wave of center frequency f_0. The second pair presents the sampled waveform and spectrum with the folding frequency 0.5 fs_1 indicated on the same frequency axis. A normalized frequency axis, f/fs_1 is also shown for the sampled data spectrum. The third pair presents the sampled waveform and spectrum with the folding frequency 0.5 fs_2 indicated on the same frequency axis. Here too a normalized frequency axis, f/fs_2, is shown for the sampled data spectrum. The second and third figure pairs present their time series on the same axis but their spectra on a different scaled axis with the analog frequency common to both axes. The fourth pair is a scaled version of the third with the scaled axis aligned to the same image width. This scaled spectrum with aligned axis gives the appearance that the sine wave center frequency has been reduced. In fact the frequency in cycles per interval is the same in the two images, but the digital frequency in radians/sample has changed because the sample-to-sample interval has changed.

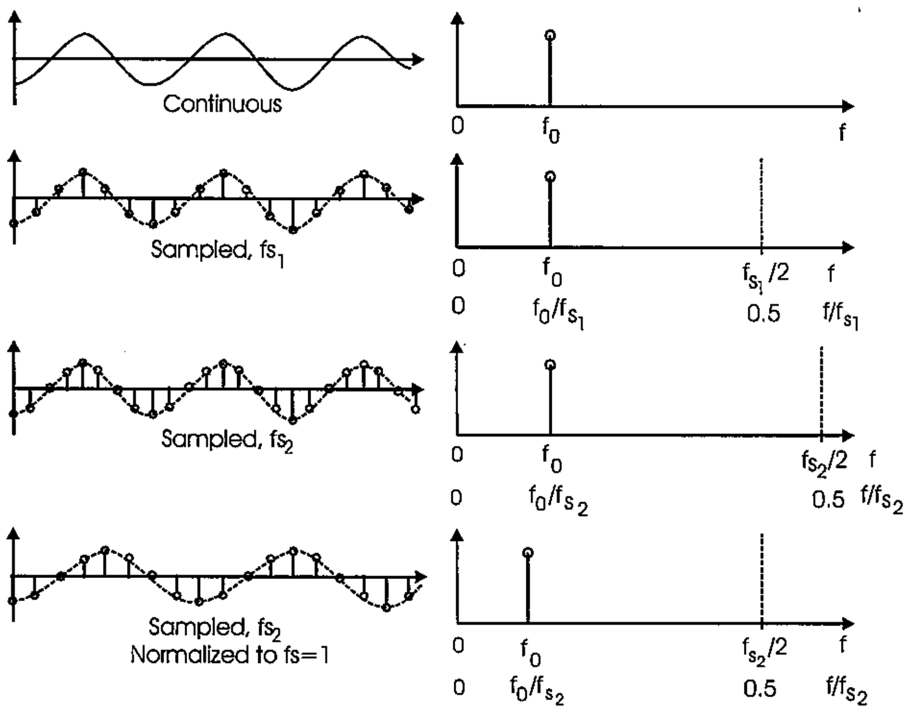


Figure 2.24 Time and Spectral Presentations of a Continuous and Sampled Data Sine Wave for Two Different Sample Rates and Two Different Scaling Options

While on the topic of scaling, let us describe an experiment in which we hold the time interval fixed and then sample a complex sinusoid with different ratios of signal center frequency to sample frequency. In the first experiment we accomplish this by holding the interval fixed, extending over 80 samples, at a fixed sample rate of unity and selecting different normalized center frequencies values of 0.1, 0.2, 0.4, and 0.8. In each case, the data set contains 80 samples of the sinusoid so that the spectrum of the time series contains the same energy. In the last case, the frequency of the sinusoid exceeds the half sample rate and aliases to the negative frequency. We see this scenario in Figure 2.25.

In the second experiment, we alter the data collection process by holding the interval fixed, extending over 80 samples at the maximum rate of unity and then selecting a sequence of sample rates of 1, 1/2, 1/4, and 1/8. As we collect data over the same interval at successively lower sample rates, we accumulate a successively smaller number of samples so that the sequence contains proportionally less energy. As in the previous experiment, the sample rate of the last sinusoid violates the Nyquist criterion so that the signal aliases to the negative frequencies. This scenario is illustrated in Figure 2.26.

The point of this demonstration is that when we change the sample rate for data collected over a fixed interval the energy or spectral amplitude of the sequence is changed and it may be necessary to apply a compensating scaling factor to the resampled data.

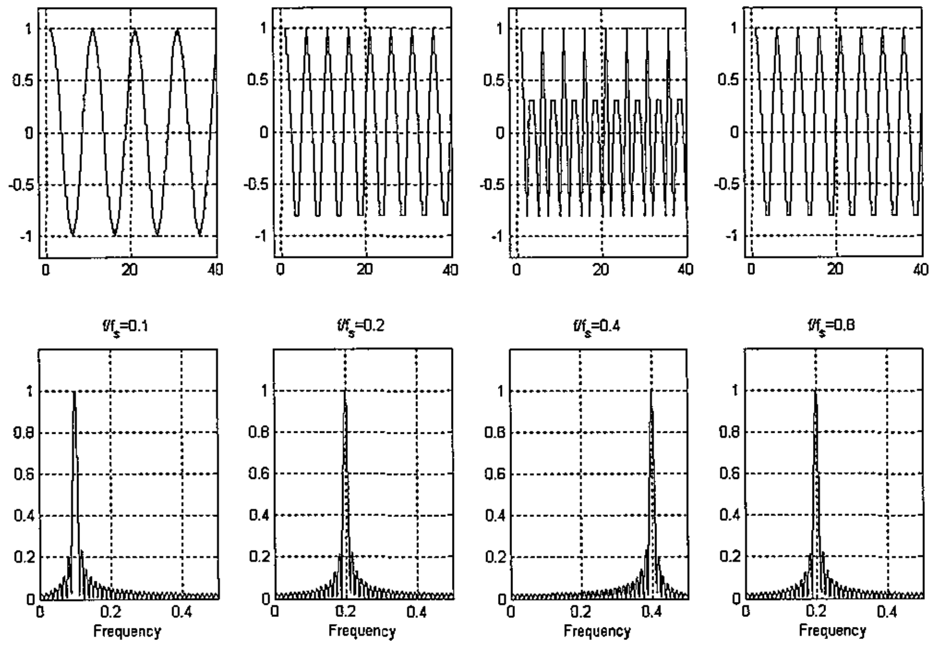


Figure 2.25 Time and Frequency Description of Four, Fixed Time Interval, Fixed Sample Rate = 1, with Successively Higher Center Frequencies f_0 = 0.1, 0.2, 0.4, and 0.8

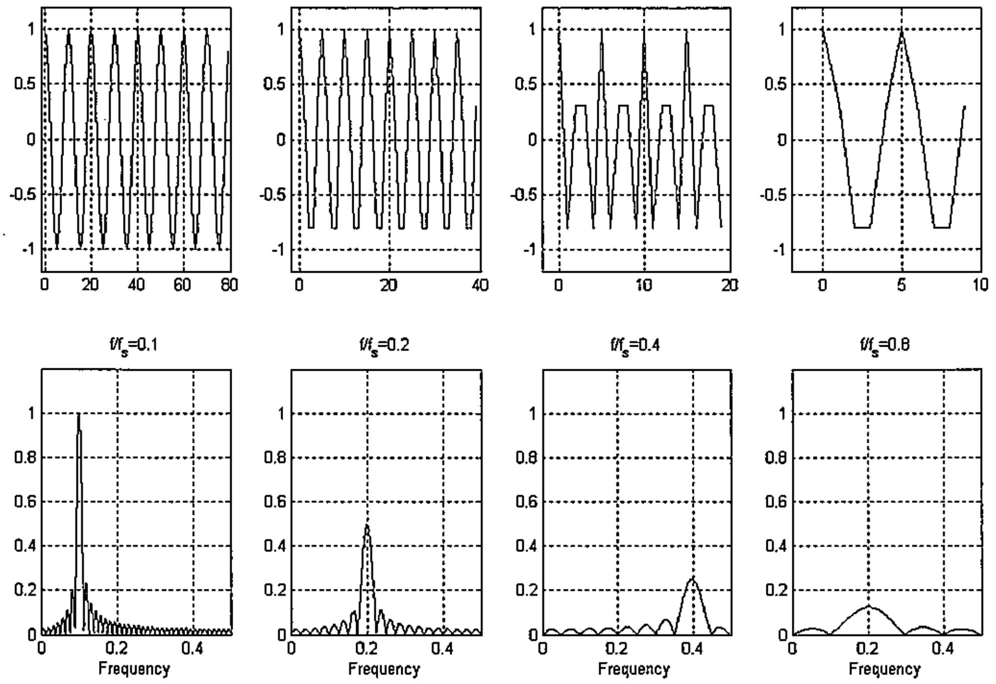


Figure 2.26 Time and Frequency Description of Four, Fixed Time Interval, Fixed Input Frequency $f_0 = 0.1$, with Successively Lower Sample Rates fs = 1, 1/2, 1/4, and 1/8

2.4 Nyquist and the Sampling Process

Much of the signal processing discussed here deals with the process of changing the sample rate of a time sequence. When describing the sample process we have as a primary concern, or at least as a background concern, the task of reconstructing the continuous waveform from the sampled data sequence. We understand the conditions under which this task is possible, and recognize the condition is a consequence of the sampling theorem. The sampling theorem states that a band-limited signal having no frequency components above f_{MAX} Hz can be determined uniquely by values sampled at uniform intervals of T_S, satisfying the relationship shown in (2.10).

$$T_S \le \frac{1}{2f_{MAX}} \text{ (sec)} \tag{2.10}$$

This relationship is known as the uniform sampling theorem, which is perhaps best known in terms of the sample frequency restriction shown in (2.11).

$$f_S \geq 2 f_{MAX} \ (Hz) \tag{2.11}$$

The sampling rate restriction is known as the *Nyquist criterion* with the minimum sample rate from (2.11) $f_S = 2f_{MAX}$, known as the *Nyquist rate*. The intuitive description of the sampling theorem is related to our understanding that when a signal is uniformly sampled, its spectrum is replicated at all multiples of the sample rate f_S. If the signal is strictly band limited with a two-sided spectral support or bandwidth of 2 BW we can prevent overlap of the spectral copies by separating them by more than their width. This statement of the sampling criterion is shown in (2.12).

$$f_s > \text{Two Sided BW} \tag{2.12}$$

The engineer's response to (2.12) is, "By how much should the sample rate exceed the two-sided bandwidth?" We have to ask this because the assumption embedded in any statement of the sampling theorem is that the spectrum is isolated and free standing. In fact, most data collection schemes require an anti-alias filter to isolate the spectrum residing in a selected spectral span from other spectra in adjacent spectral spans. We require a filter to perform the spectral separation, and the spectral response of this filter affects the sample rate. Figure 2.27 presents a spectrum comprising three adjacent spectral spans and the effect of the signal conditioning on the signal with this spectrum. We see that the filter rejects the adjacent channels down to a level matching the dynamic range of the data collection process. Typical values of dynamic range are 60, 72, and 96-dB for an ADC with spurious free dynamic range of 10, 12, and 16 bits respectively. When the filtered signal is sampled, the spectral copies must be separated sufficiently so that the levels of the folded remnants that fall back into the signal bandwidth are below the dynamic range of the ADC. This is assured when the sample rate satisfies (2.13). In words, the engineer's version of the sampling theorem is "The sample rate must equal the two-sided bandwidth plus the transition bandwidth of the anti-aliasing filter". This is the design criterion to which we will adhere as we develop our applications of multirate filters.

$$\begin{aligned} f_S &= \text{Two-Sided BW} + \text{Filter Transition BW} \\ f_S &= 2\,BW \ + \Delta f \end{aligned} \tag{2.13}$$

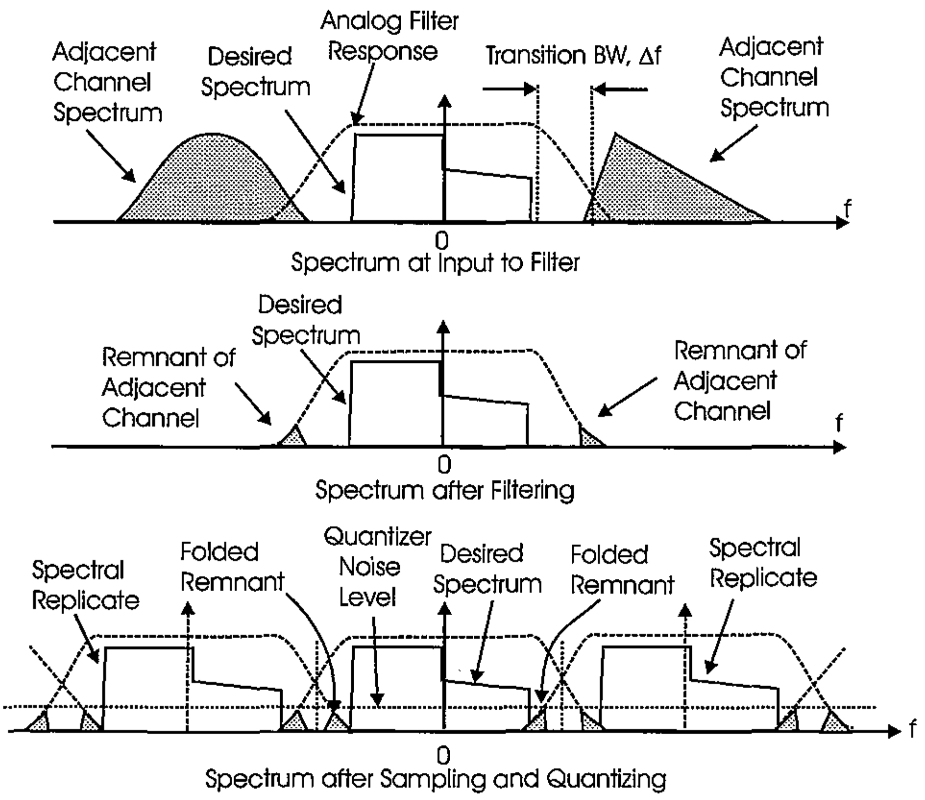


Figure 2.27 Spectrum of Input Signal and of Filter, of Filtered Input Signal, and of Sampled Input Signal

References

Crochiere, Ronald, and Lawrence Rabiner, *Multirate Signal Processing*, Englewood Cliffs, NJ: Prentice-Hall, Inc., 1983.

Fliege, Norbert, *Multirate Digital Signal Processing: Multirate Systems, Filter Banks, Wavelets*, West Sussex, John Wiley & Sons, Ltd., 1994.

Jovanovic-Dolecek, Gordana, *Multirate Systems: Design and Applications*, London, Idea Group, 2002.

Mitra, Sanjit, *Digital Signal Processing: A Computer-Based Approach*, 2nd ed., New York, McGraw-Hill, 2001.

Mitra, Sanjit, and James Kaiser, *Handbook for Digital Signal Processing*, New York, John Wiley & Sons, 1993.

Vaidyanathan, P. P., *Multirate Systems and Filter Banks*, Englewood Cliffs, NJ: Prentice-Hall, Inc., 1993.

Problems

2.1 Determine the Z-Transform and Discrete-Time Fourier series expansion of

a) $\delta(n)$

b) $\delta(n-10)$

c) $\delta(n-20)$

d) $s_0(n) = \sum_{k=0}^{99} \delta(n-10k)$

2.2 Determine the Z-Transform and Discrete-Time Fourier series expansion of

a) $\delta(n-1)$

b) $\delta(n-11)$

c) $\delta(n-21)$

d) $s_1(n) = \sum_{k=0}^{99} \delta(n-10k-1)$

2.3 Determine the Z-Transform and Discrete-Time Fourier series expansion of

a) $\delta(n-r)$

b) $\delta(n-10-r)$

c) $\delta(n-20-r)$

d) $s_r(n) = \sum_{k=0}^{99} \delta(n-10k-r)$

2.4 Determine the Z-Transform and sampled data time sequence corresponding to

a) $H(\theta) = 1$

b) $H(\theta) = \exp(j\,20\,\theta)$

c) $H(\theta) = \exp(j\,40\,\theta)$

d) $H(\theta) = \sum_{k=0}^{49} \exp(j10k\theta)$

2.5 A complex time series $h(t)=\exp(j\,2\,\pi\,500\,t)$ is sampled at a 4.0 kHz rate and 200 samples collected for processing. Determine

a) $h(n)$, (sample time values)

b) $H(k)$, (DFT frequency samples)

c) The number of cycles of the input time series contained in the span of collected data

d) Index k of the DFT containing the non-zero component of the DFT

e) The amplitude of the non-zero DFT sample

2.6 A complex time series $h(t) = \exp(j\ 2\ \pi\ 1000\ t)$ is sampled at a 4.0 kHz rate and 200 samples collected for processing. Determine

a) h(n), (sample time values)

b) H(k), (DFT frequency samples)

c) The number of cycles of the input time series contained in the span of collected data

d) Index k of the DFT containing the non-zero component of the DFT

e) The amplitude of the non-zero DFT sample

2.7 A complex time series $h(t) = \exp(j\ 2\ \pi\ 2000\ t)$ is sampled at a 4.0 kHz rate and 200 samples collected for processing. Determine

a) h(n), (sample time values)

b) H(k), (DFT frequency samples)

c) The number of cycles of the input time series contained in the span of collected data

d) Index k of the DFT containing the non-zero component of the DFT

e) The amplitude of the non-zero DFT sample

2.8 A complex time series $h(t) = \exp(j\ 2\ \pi\ 4000\ t)$ is sampled at a 4.0 kHz rate and 200 samples collected for processing. Determine

a) h(n), (sample time values)

b) H(k), (DFT frequency samples)

c) The number of cycles of the input time series contained in the span of collected data

d) Index k of the DFT containing the non-zero component of the DFT

e) The amplitude of the non-zero DFT sample

2.9 A complex time series $h(t) = \exp(j\ 2\ \pi\ 500\ t)$ is sampled at a 2.0 kHz rate and 200 samples collected for processing. Determine

a) h(n), (sample time values)

b) H(k), (DFT frequency samples)

c) The number of cycles of the input time series contained in the span of collected data

d) Index k of the DFT containing the non-zero component of the DFT

e) The amplitude of the non-zero DFT sample

2.10 A complex time series $h(t) = \exp(j\ 2\ \pi\ 500\ t)$ is sampled at a 1.0 kHz rate and 200 samples collected for processing. Determine

a) h(n), (sample time values)

b) H(k), (DFT frequency samples)

c) The number of cycles of the input time series contained in the span of collected data

d) Index k of the DFT containing the non-zero component of the DFT

e) The amplitude of the non-zero DFT sample

2.11 A complex time series h(t) = exp(j 2 π 500 t) is sampled at a 0.50 kHz rate and 200 samples collected for processing. Determine

a) h(n), (sample time values)

b) H(k), (DFT frequency samples)

c) The number of input time series contained in the span of collected data

d) Index k of the DFT containing the non-zero component of the DFT

e) The amplitude of the non-zero DFT sample.

2.12 Use a windowed sin(x)/x to generate the impulse response of a sampled low pass filter with the following specifications: pass band bandwidth, 0-to-40 Hz, stop band bandwidth, 60-to-200 Hz, sample rate, 400 Hz, and stop band attenuation greater than 60-dB. The following MATLAB script will accomplish this:

```
h1 = sinc(−10.0:0.25:10.0.*kaiser(81,6)';
h1 = h1/sum(h1);
```

Zero pack the impulse response h1 to form a new sequence h2. The following MATLAB script will accomplish this:

```
h2 = reshape([h1;zeros(1,81)],1,162);
```

Plot in two figures the time series and the log-magnitude spectrum of h1 and of h2. The following MATLAB script will accomplish this for h1. The time and frequency axis must be changed for h2 to reflect the double sample rate.

```
subplot(2, 1, 1)
plot(0:0.25:20+0.25, (h1))
grid on;
axis([−2 22 −0.1 0.30])
subplot(2,1,2)
freq = (−0.5:1/1024:0.5−1/1024)*400
plot(freq, fftshift(20*log10(fft(h1,1024))))
grid on
axis([−200 200 −80 10])
```

Examine the two spectra and comment on the number and the locations of the filter bandwidths in the two figures.

2.13 Repeat Problem 2.12 except zero pack 1-to-3 instead of 1-to-2. The following MATLAB script accomplishes the zero packing:

```
h3 = reshape([h1;zeros(2,82)],1,243);
```

2.14 Use a windowed sin(x)/x to generate the impulse response of a sampled low pass filter with the following specifications: pass band bandwidth, 0-to-100 Hz, stop band bandwidth, 150-to-1000 Hz, sample rate, 2 KHz, and stop band attenuation greater than 60-dB. The following MATLAB script will accomplish this:

```
h1 = sinc(−8:01:8)*kaiser(161,8)';
h1 = h1/sum(h1);
```

Also form h2 and h3, down sampled time series by the following MATLAB script:

```
h2 = zeros(1,161);
h2(1:5:161) = h1(1:5:161);
h3 = zeros(1,161);
h3(2:5:161) = h1(2:5:161);
```

Plot in three subplots the spectrum log magnitude of the sequence h1, h2, and h3 and comment on the bandwidths and locations of the observed pass band(s).

2.15 Use a windowed sin(x)/x to generate the impulse response of a sampled low pass filter with the following specifications: pass band bandwidth, 0-to-100 Hz, stop band bandwidth, 150-to-1000 Hz, sample rate, 2 kHz, and stop band attenuation greater than 60-dB. The following MATLAB script will accomplish this:

```
h1 = sinc(−8:01:8)*kaiser(161,8)';
h1 = h1/sum(h1);
```

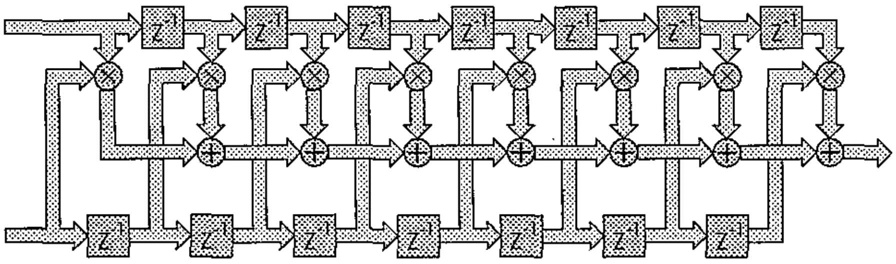
Also form h2 and h3, down sampled time series by the following MATLAB script:

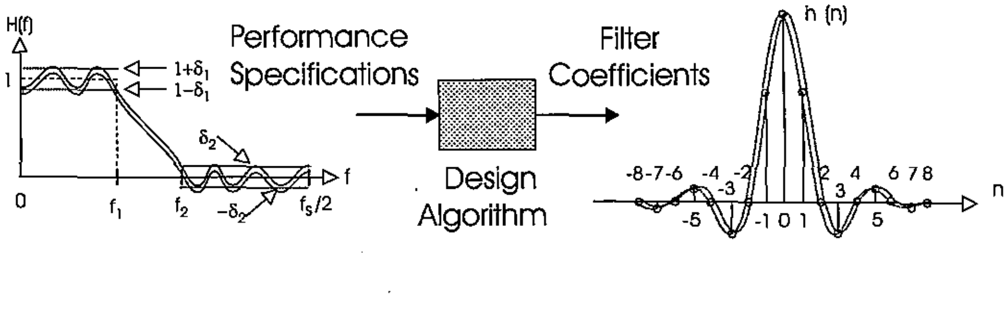
```
h2 = h1(1:5:161);
h3 = h1(2:5:161);
```

Plot in three subplots the log magnitude spectrum of the sequence h1, h2, and h3 and comment on the bandwidths and locations of the observed pass band(s).

CHAPTER 3

Digital Filters





The intent of this chapter is to review the properties and performance constraints of digital filters so that we are better able to embed them in a multirate system. There is no intent here to present a comprehensive overview and detailed description of the many ways to design and implement the various digital filters. What we will do is identify the important system considerations that a designer should consider in the design process and suggest various options as well as present guidelines to help select structures and performance trade-offs required to finalize a digital filter design.

Digital filters can be classified in many ways, including allusion to their general characteristics such as low pass, band-pass, band-stop, and others and secondary characteristics such as uniform and nonuniform group delay. An important classification is the filter's architectural structure with a primary consideration being that of finite (duration) impulse response (FIR) and infinite (duration) impulse response (IIR). Except for special cases, involving pole-zero cancellation, FIR and IIR filters are implemented by nonrecursive and recursive structures. Further subclassifications, such as canonic forms, cascade forms, lattice forms, and the like are primarily driven by consideration of sensitivity to finite arithmetic, memory requirements, ability to pipeline arithmetic, and hardware constraints.

The choice to perform a given filtering task with a recursive or a nonrecursive filter is driven by a number of system considerations, including processing resources, clock speed, and various filter specifications. Performance specifications, which include operating sample rate, pass band and stop band edges, pass band ripple, and out-of-band attenuation, all interact to determine the complexity required of the digital filter. We first examine how the performance parameters of the FIR filter interact to determine the filter length and hence the FIR filter's complexity. We then examine the similar relationship between the parameters of IIR filters that we eventually cast into multirate architectures.

3.1 Filter Specifications

The relationships between filter length and filter specifications are valid for any FIR filter. Since the most common filtering task is that of a low pass filter we examine the prototype low pass filter to understand the coupling between the filter parameters. These interactions remain valid for other filter types. The frequency response of a prototype low pass filter is shown in Figure 3.1. The pass band is seen to be an ideal rectangle that has unity gain between frequencies $\pm f_1$ Hz with zero gain elsewhere. The filter is designed to operate at a sample rate of f_S Hz. For the convenience of dealing with a specific example, we chose the single-sided band edge to be 10 kHz and the sample rate to be 100 kHz. The attraction of the ideal low pass filter H(f) as a prototype is that we have an exact expression for its impulse response h(t) from its closed form inverse Fourier transform, the ubiquitous sin(x)/x as shown in (3.1).

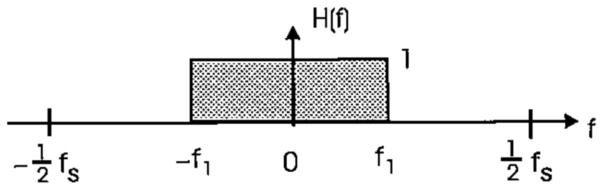


Figure 3.1 Frequency Response Prototype Low pass Filter

$$h(t) = 2f_1 \frac{\sin(2\pi\frac{2f_1}{2}t)}{(2\pi\frac{2f_1}{2}t)} \tag{3.1}$$

In words, the argument of the sin(x)/x function is always the product of 2π, half the spectral support $(2f_1)/2$, and the independent variable t. The numerator is periodic and becomes zero when the argument is a multiple of π, in general at ± kπ. The location of the first zero occurs as shown in (3.2) at:

$$\text{First zero: } 2\pi\frac{2f_1}{2}t = \pi, \quad t_{ZERO} = \frac{1}{2f_1} \tag{3.2}$$

The nonrealizable impulse response of the prototype filter is shown in Figure 3.2. For the two-sided bandwidth of 20 kHz, the first zero occurs at 50 μsec.

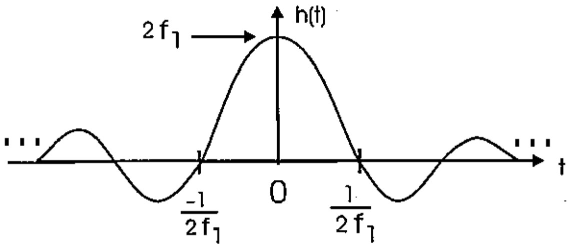


Figure 3.2 Impulse Response of Prototype Low pass Filter

The sin(x)/x filter shown in Figure 3.2 is a continuous function which we have to sample to obtain the prototype sampled data impulse response. To preserve the filter gain during the sampling process we scale the sampled function by the sample rate as shown in (3.3). The sampled impulse response of the prototype low pass is shown in (3.4) while Figure 3.3 presents a visualization of the sampling process.

$$h(n) = \frac{1}{f_S} h(t)\Big|_{t=n\frac{1}{f_S}} \tag{3.3}$$

$$\begin{aligned} h(n) &= \frac{2f_l}{f_S} \frac{\sin(2\pi\frac{f_l}{f_S}n)}{(2\pi\frac{f_l}{f_S}n)} \\ &= \frac{2f_l}{f_S}\frac{\sin(n\theta_l)}{(n\theta_l)}, \quad \text{where } \theta_l = 2\pi\frac{f_l}{f_S} \end{aligned} \tag{3.4}$$

An important observation is that when used in a fixed-point arithmetic processor the composite scale factor $2f_l/f_s$ shown in (3.4) is removed from the impulse response weights and is reinserted as a scaling factor when clearing the accumulator after the accumulation process. Without the scaling factor the filter exhibits processing gain proportional to the ratio of sample rate to bandwidth. Accumulators are designed with extra bit width to accommodate this expected bit growth due to processing gain. While the scaling factor is required to cancel the processing gain of the filter weights, it should not be applied to the coefficient set since it reduces the precision with which the coefficients are represented, which leads to an increase in arithmetic noise of the filter process. This is a common source of error in filter design routines, one that is easily corrected by scaling the filter coefficients by the maximum weight so that the maximum weight is unity rather than $2f_l/f_s$. For the specific example we are using, this scale factor is 20/100 or 0.2, which represents a loss in coefficient precision of more than 2 bits. This loss in coefficient precision is significant when the filter bandwidth is a small fraction of the sample rate. This scaling factor will be seen to be an important concern when the filter is used in a resampling configuration.

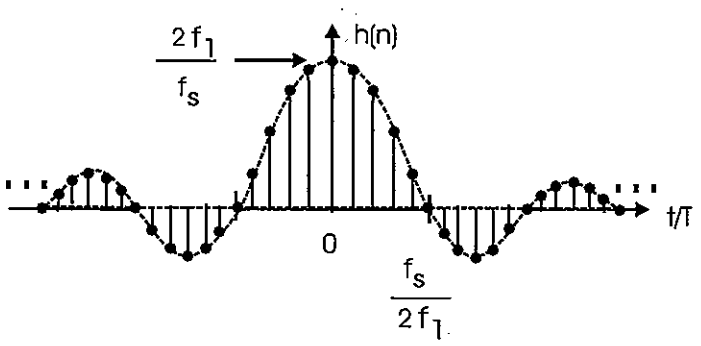


Figure 3.3 Sampled Data Impulse Response of Prototype Filter

An insightful observation is to be had by examining Figure 3.3 and asking, "How many samples are there between the peak and first zero crossing of the prototype impulse response?" The number of samples is seen to be $f_S/(2f_l)$, the ratio of sample rate to two-sided bandwidth, which for our specific example is 5. We thus have an interesting measure of the filter bandwidth: if we examine a filter impulse response and note that there are 50 samples from peak to first zero crossing, we can conclude that the two-sided bandwidth is 1/50th of the sample rate. If we can count, we can estimate the fractional bandwidth of a FIR filter!

The sampling process, of course, causes the spectra to be periodically extended with spectral replicates at all multiples of the sample rate. The expression for the spectrum of the sampled data impulse response is shown in (3.5) where the sampled data frequency variable ωT_s is denoted by θ with units of radians/sample. In this coordinate system, the spectrum is periodic in 2π.

$$H(\theta) = \sum_{n=-\infty}^{+\infty} h(n)e^{-jn\theta} \tag{3.5}$$

3.2 Windowing

The problem with the sample set of the prototype filter is that the number of samples is unbounded and the filter is noncausal. If we had a finite number of samples, we could delay the response to make it causal. Our first task then is to form a finite list of filter coefficients from the unbounded set. The process of pruning an infinite sequence to a finite sequence is called *windowing*. In this process, a new sequence is formed as the product of the finite sequence w(n) and the infinite sequence as shown in (3.6) where the .* operator is the standard MATLAB point-by-point multiply.

$$h_w(n) = w(n)\ .*\ h(n) \tag{3.6}$$

The expression for the spectrum of the windowed impulse response is shown in (3.7) as the transform of the product h(n) and w(n) and once again in (3.8) as the circular convolution of their spectra $H(\theta)$ and $W(\theta)$. We will examine the effect of the convolution shortly.

$$\begin{aligned} H_w(\theta) &= \sum_{n=-\infty}^{+\infty} h(n)w(n)e^{-jn\theta} \\ &= \sum_{n=-N/2}^{+N/2} h(n)w(n)e^{-jn\theta} \end{aligned} \tag{3.7}$$

$$H_w(\theta) = \frac{1}{2\pi}\int_{-\pi}^{+\pi} H(\varphi)W(\varphi-\theta)\,d\varphi \tag{3.8}$$

Our first contender for a window is the symmetric rectangle, sometimes called the default window. This weighting function abruptly turns off the coefficient set at its boundaries. The sampled rectangle weighting function has a spectrum described by the Dirichlet kernel as shown in (3.9). The Dirichlet kernel is seen to be the periodic extension of the $\sin(2\pi f T_{support}/2)/(2\pi f T_{support}/2)$ spectrum, the transform of a continuous time rectangle function.

$$W_{RECT}(\theta) = \sum_{n=-N/2}^{+N/2} 1\, e^{jn\theta} = \frac{\sin(N\frac{\theta}{2})}{\sin(\frac{\theta}{2})} \tag{3.9}$$

Figure 3.4 presents the spectrum of a 100-tap rectangle window as well as a zoom to the neighborhood of its main lobe. The frequency axis here is normalized frequency f/f_S so that the first spectral zero occurs at frequency $1/N = 1/100 = 0.01$. Note that the first side-lobe has an amplitude of approximately -22 which, relative to the peak of amplitude 100, represents a power ratio of -13.2-dB, a handy relationship to remember.

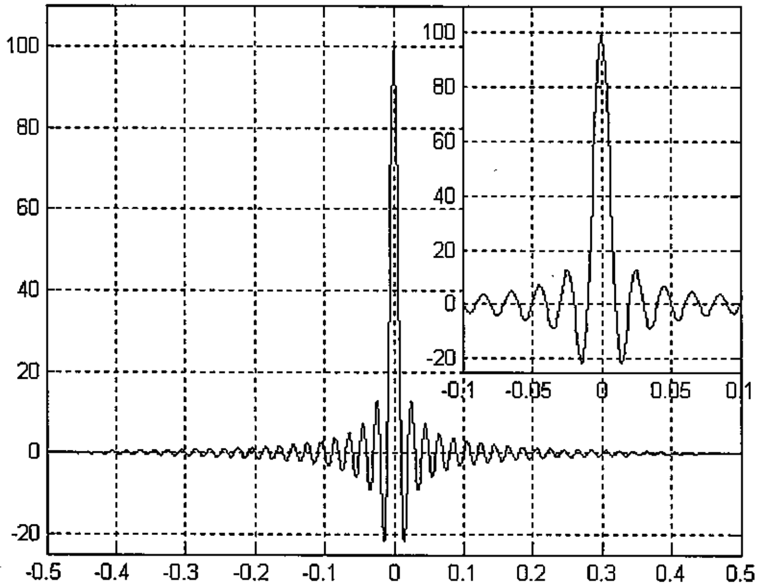


Figure 3.4 Spectrum of 100-point Rectangle Window with Zoom to Main Lobe

The convolution between the spectra of the prototype filter with the Dirichlet kernel forms the spectrum of the rectangle windowed filter coefficient set. This convolution is shown in Figure 3.5. The convolution shows the main lobe of the Dirichlet kernel in three distinct spectral regions: out-of-band, straddling the band edge, and in-band. The contribution to the corresponding output spectrum is seen to be stop band ripple, transition bandwidth, and pass band ripple. The pass band and stop band ripple are due to the side-lobes of the Dirichlet kernel moving through the pass band of the prototype filter while the transition bandwidth is due to the main lobe of the kernel moving from the stop band to the pass band of the prototype. Note that the transition bandwidth is the same as the main lobe width of the kernel, approximately 1/Nth of the sample rate for a filter of length N.

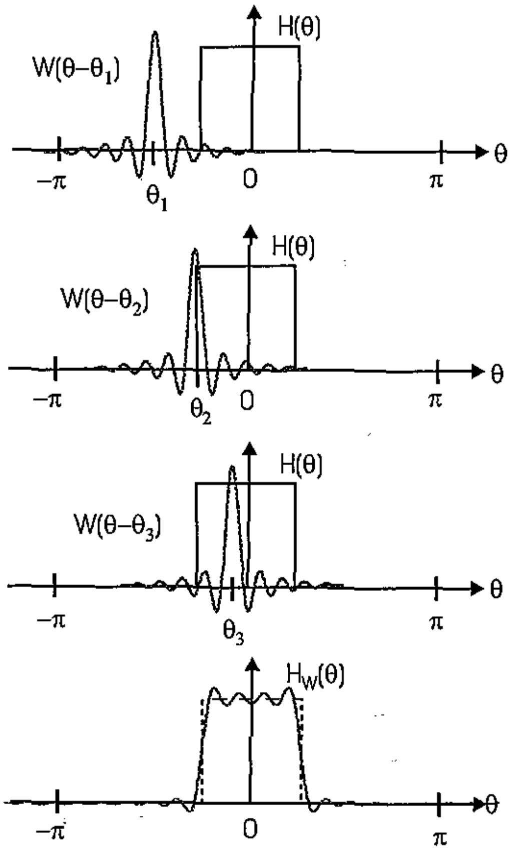


Figure 3.5 Spectrum of Rectangle Windowed Prototype Filter Obtained as Convolution Between Spectrum of Prototype Filter and Spectrum of Rectangle Window

A property of the Fourier series is that a truncated version of the series forms a new series exhibiting the minimum mean-square approximation to the original function. We thus note that the coefficient set obtained by a rectangle window exhibits the minimum mean square (MMS) approximation to the prototype frequency response. The problem with MMS approximations in numerical analysis is that there is no mechanism to control the location or value of the error maxima. The local maximum errors are attributed to the *Gibbs phenomena*, the failure of the series to converge in the neighborhood of a discontinuity. These errors can be objectionably large. Figure 3.6 presents a log-magnitude display of the spectrum formed by the rectangle windowed coefficient set. We see here that the stop band side-lobes only present 22-dB attenuation near the filter band edge.

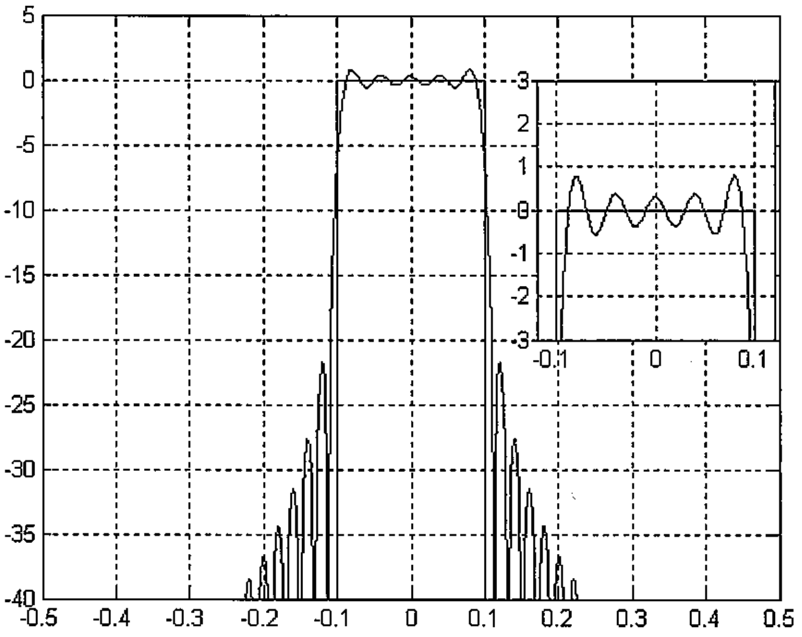


Figure 3.6 Log Display of Spectrum to Emphasize High Levels of Out-of-band Side-lobe Response of Rectangle Windowed Prototype Filter

A process must now be invoked to control the objectionably high side-lobe levels. We have two ways to approach the problem. First we can redefine the frequency response of the prototype filter so that the amplitude discontinuities are replaced with specified transition bandwidth tapers. In this process, we exchange transition bandwidth for side-lobe control. How to make that change effectively is the next question. Equivalently, knowing the objectionable stop band side-lobes are caused by the side-lobes in the spectrum of the window, we can replace the rectangle window with other even symmetric functions with reduced amplitude side-lobe levels. Here the question to be addressed is how do we select weighting functions with low spectral side-lobes. As we will now show, the two techniques, side-lobe control and transition-bandwidth control are tightly coupled. The easiest way to visualize control of the side-lobes is by destructive cancellation between the spectral side-lobes of the Dirichlet kernel associated with the rectangle and the spectral side-lobes of translated and scaled versions of the same kernel. Figure 3.7 presents the window formed by the addition of a single cycle of a cosine to the rectangle window as well as the transforms of the time domain components. Note that the single cycle of cosine is the lowest frequency sinusoid orthogonal to the rectangle. This orthogonality is observed in the frequency domain as the placement of the cosine's spectral components at the first spectral zeros of the rectangle's spectrum. We note that the side-lobes of the DC-centered kernel and the side-lobes of the translated kernels have opposite polarities in the spectral region outside the main lobe spectral support. By judicious choice of weighting terms, the side-lobe amplitudes are significantly reduced.

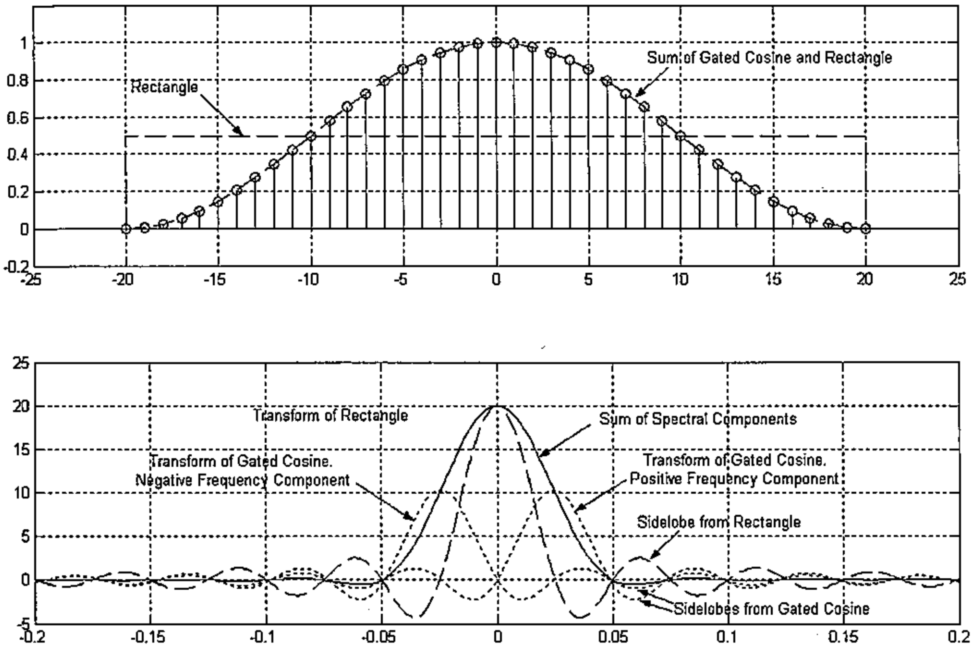


Figure 3.7 Raised Cosine Window and Transform Illustrating Sum of Translated and Scaled Dirichlet Kernels

In Figure 3.7 we easily see that adding the translated kernels to the original spectrum has doubled the distance from peak to first spectral null. Thus the cost we incur to obtain reduced side-lobe levels is an increase in main lobe bandwidth. Table 3-1 presents a list of window functions formed as the sum of translated kernels with their peak side-lobe levels along with their main lobe widths.

Scanning Table 3-1 we can estimate the rate at which we can trade side-lobe levels for main lobe width. This rate is approximately –22-dB/spectral bin so that in order to obtain –60-dB side-lobes, we have to increase the main lobe bandwidth to 2.7 f_S/N. Remembering that the window's two-sided main lobe width is an upper bound to the filter's transition bandwidth, we can estimate the transition bandwidth of a filter required to obtain a specified side-lobe level. Equation (3.10) presents an empirically derived approximate relationship valid for window-based design while (3.11) rearranges (3.10) to obtain an estimate of the filter length required to meet a set of filter specifications.

Table 3-1 Windows Formed as Weighted Sum of Cosines

Window Name	Weights	Maximum Side-lobe	Main lobe Width Peak-to-First Zero
Rectangle	$a_0 = 1.0$	−13.5-dB	1
Hann	$a_0 = 0.5$ $a_1 = -0.5$	−32-dB	2
Hamming	$a_0 = 0.54$ $a_1 = -0.46$	−43-dB	2
Blackman (Approximation)	$a_0 = 0.42$ $a_1 = -0.50$ $a_2 = 0.08$	−58-dB	3
Blackman (Exact)	$a_0 = 0.426\,59$ $a_1 = -0.496\,56$ $a_2 = 0.076\,85$	−68-dB	3
Blackman-harris (3-Term)	$a_0 = 0.423\,23$ $a_1 = -0.497\,55$ $a_2 = 0.079\,22$	−72-dB	3
Blackman-harris (4-Term)	$a_0 = 0.358\,75$ $a_1 = -0.488\,29$ $a_2 = 0.141\,28$ $a_3 = -0.011\,68$	−92-dB	4

$$\Delta f_{\text{MINIMUM}} = \frac{f_S}{N}$$
$$\Delta f = \frac{f_S}{N} K(Atten) \cong \frac{f_S}{N} \frac{Atten(dB)-8}{14} \tag{3.10}$$

$$N \cong \frac{f_S}{\Delta f} \frac{Atten(dB)-8}{14} \tag{3.11}$$

The primary reason we examined windows and their spectral description as weighted Dirichlet kernels was to develop a sense of how we trade window main lobe width for window side-lobe levels and in turn filter transition bandwidth and side-lobe levels. Some windows perform this exchange of bandwidth for side-lobe level very efficiently while others do not. The Kaiser-Bessel window is very effective while the triangle (or Fejer) window is not. The Kaiser-Bessel window is in fact a family of windows parameterized over β, the time-bandwidth product of the window. The main lobe width increases with β while the peak side-lobe level decreases with β. The Kaiser-Bessel window is a standard option in filter design packages such as MATLAB and QED-2000. For completeness we describe it here in (3.12) where I_0 is the zero-order modified Bessel function of the first kind. The series shown converges quite rapidly due to the k! term in the denominator of the expansion. Typical range of the parameter β is 3 to 10 to obtain filter side-lobe levels in the range 40-to-100-dB. We note that the window defaults to a rectangle for β =0.

$$w(n) = \frac{I_0\left[\pi\beta\sqrt{1.0-\left(\frac{n}{N/2}\right)^2}\right]}{I_0[\pi\beta]} : -N/2 \le n < N/2 \qquad (3.12)$$

$$where\ I_0(x) = \sum_{k=0}^{\infty}\left[\frac{(x/2)^k}{k!}\right]^2$$

The transform of the Kaiser-Bessel window is approximately that shown in (3.13).

$$W(\theta) = \frac{N}{I_0(\pi\beta)}\frac{\sinh\left(\sqrt{(\pi\beta)^2-(N\theta/2)^2}\right)}{\sqrt{(\pi\beta)^2-(N\theta/2)^2}} \qquad (3.13)$$

We still have to relate the window side-lobe levels, which are integrated in the convolution process, to form the filter side-lobe levels. As an example the first side-lobe of the Dirichlet kernel is –13.5-dB relative to the spectral peak while the first side-lobe in the resulting low pass filter is –22-dB relative to pass band gain. We will only concern ourselves with this relationship for the Kaiser-Bessel class of windows. Figure 3.8 presents a curve showing the spectral side-lobe levels realized by windowing a prototype impulse response with the Kaiser-Bessel window of specified parameter β along with a second curve showing the spectral side-lobe levels of the corresponding window. Note that at β = 0, the two levels correspond to the rectangle window.

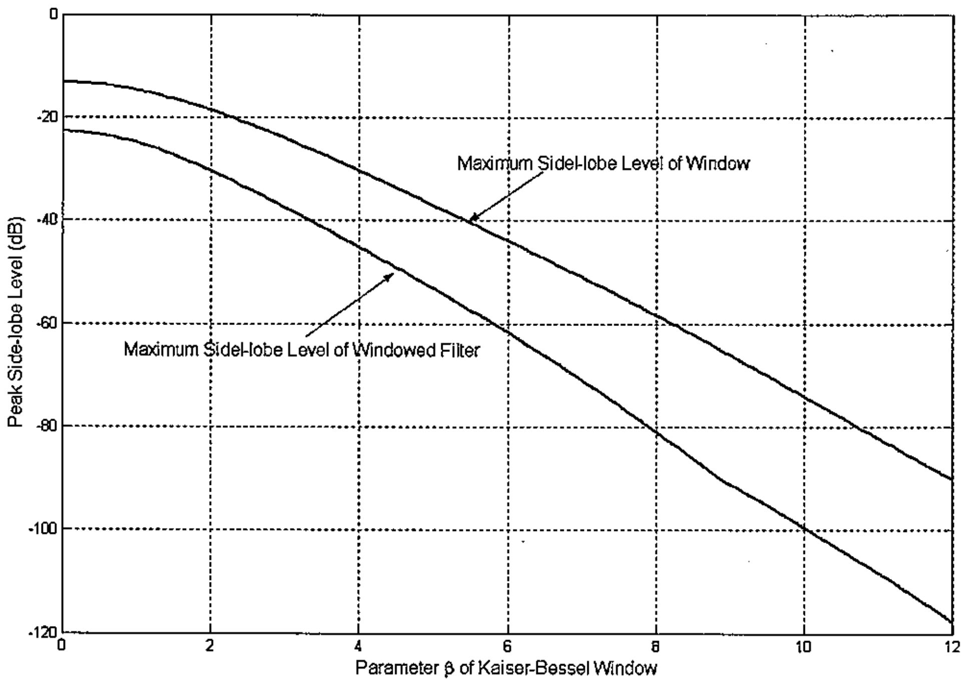


Figure 3.8 Side-lobe Levels of Kaiser-Bessel Windowed FIR Filter and of the Kaiser-Bessel Window as a Function of Window Parameter β

Example 3.1 Window Design of Low pass FIR Filter

Design a FIR filter with the following specifications:

Sample Rate	100 kHz
Pass band Band Edge	±10 kHz
Stop band Band Edge	±15 kHz
Minimum Attenuation	60-dB

From the filter specifications and from (3.11) we estimate the filter length N:

$$N \cong (fs/\Delta f) * (\text{Atten}(\text{-dB})-8)/14$$

$$= (100/5) * (60-8)/14 = 75 \text{ taps}$$

Using the sinc function in MATLAB to form samples of the prototype impulse response and the Kaiser-Bessel window to control the spectral side-lobe levels we estimate from Figure 3.8, or as the result of a few trials, that 60-dB side-lobe levels are obtained with parameter $\beta = 5.7$. Two versions of the filter were designed. The first was designed for the pass band parameter, which resulted in the band edges being centered about the 6-dB frequency of the filter, this being a consequence of the Fourier transform converging to the midpoint of a discontinuity. The second design shifted the pass band parameter to the midpoint of the transition band. Both designs are described compactly in the following two sets of MATLAB script. The time and frequency responses of the two designs are shown in Figure 3.9.

```
hh1=sinc((2*f1)/fs)*(-0.5*(N-1):0.5*(N-1));
hh1=hh1.*kaiser(N,β)';
hh1=sinc(0.2*(-37:1:37)).*kaiser(75,5.7)';
hh2=sinc(((2f1+Δf/2)/fs)*(-0.5*(N-1):0.5*(N-1));
hh2=hh2.*kaiser(N,β)';
hh2 = sinc(0.25*(-37:1:37)).*kaiser(75,5.7)';
```

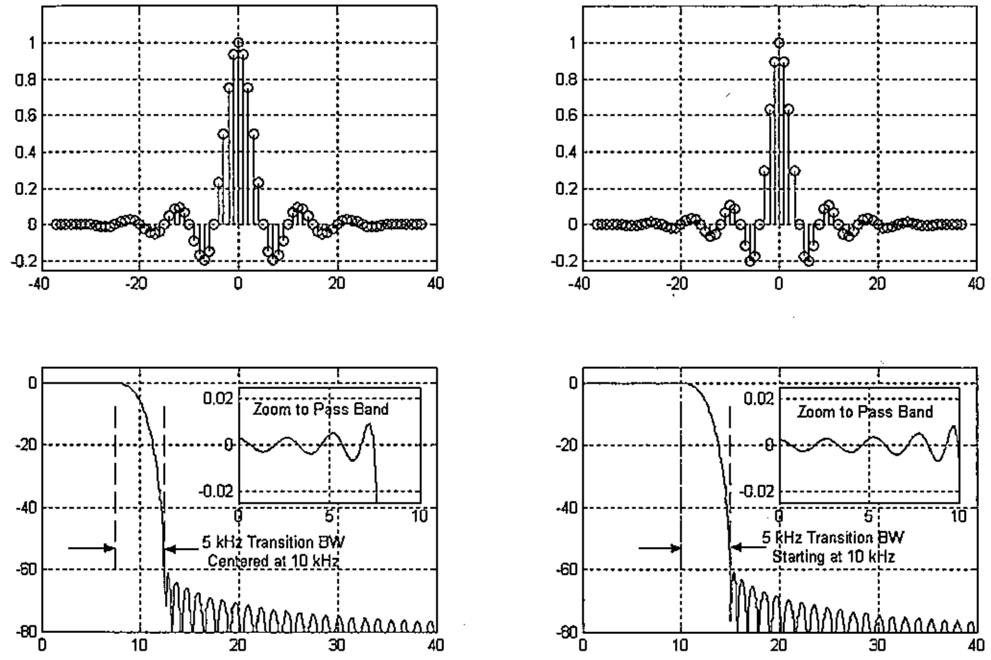


Figure 3.9 Time and Frequency Response of FIR Filter Windowed with Kaiser-Bessel: First Designed for 6-dB Band Edge, Second for Pass band and Stop band Edges

Note in both filter designs, the in-band ripple has the same structure as the out-of-band ripple with peak values on the order of 1-part-1000. For the first design the ripple pass band extends from 0-to-7.5 kHz while in the second design the ripple pass band extends from 0-to-10 kHz.

ODTÜ KÜTÜPHANESİ
M. E. T. U. LIBRARY

3.3 The Remez Algorithm

In the previous section we learned that FIR filters always exhibit ripple in the pass band and in the stop band as well as bandwidth to transition between the pass band and the stop band. Thus filters must be specified in accord with the parameters indicated in Figure 3.10 and identified in the following parameter list.

Filter Specification Parameters

fs: Sample Rate

f_1: Frequency at End of Pass Band

f_2: Frequency at Start of Stop Band

δ_1: Maximum Pass band Ripple

δ_2: Maximum Stop band Ripple

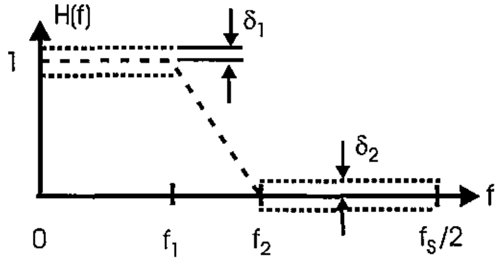


Figure 3.10 Parameters Required to Specify Sampled Data Low pass Filter

$$N = function(f_S, f_1, f_2, \delta_1, \delta_2) \tag{3.14}$$

As indicated in (3.14) N, the number of coefficient taps required of the FIR filter to meet the specifications, is a function of five parameters. How we select most of the parameters is self-evident. The sample rate must satisfy the Nyquist criterion, the pass band and stop band frequencies must satisfy the filtering requirements, and the stop band ripple must satisfy the out-of-band attenuation requirement. The pass band ripple requirement is related to a signal distortion criterion modeled as signal echoes caused by the pass band ripple. We discuss the pass band ripple criterion in terms of system performance in the next section. Maximum pass band ripple values for many system designs are on the order of 1-part in 100 to 5-parts in 100 (i.e., 1% to 5%). These levels are significantly larger than the stop band ripple values that are on the order of 1-part in 1000 to 1-part in 10000 (i.e., 60 to 80-dB). FIR filters designed by the windowing technique exhibit equal pass band and stop band ripple levels. We now seek a design process that permits different levels of pass band and stop band ripple. Filters with relaxed pass band ripple requirements will require fewer coefficients, hence require fewer resources to implement.

The window design of a FIR filter occurs in the time domain as the point-by-point product of a prototype impulse response with the smooth window sequence. The quality of the resultant design is verified by examining the transform of the windowed impulse response. By contrast, the equiripple design is performed entirely in the frequency domain by an iterative adjustment of the location of sampled spectral values to obtain a Tchebyschev approximation to a desired spectrum. The desired, or target, spectrum has accompanying tolerance bands that define acceptable deviations from the target spectrum in distinct spectral regions. The alternation theorem assures us that there exists a Tchebyschev approximation to the target spectrum and that this solution exhibits equiripple errors with local extrema of alternating signs meeting the tolerance boundaries. This approximation and error function are shown for a desired set of tolerance bands in Figure 3.11.

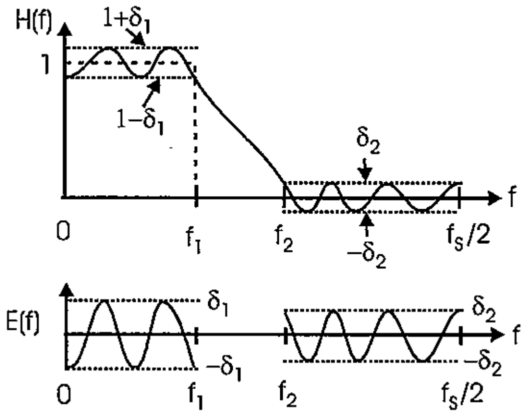


Figure 3.11 Frequency Response of Equiripple Filter and Error Frequency Profile

When the FIR filter has 2M+1 even symmetric coefficients, its spectrum can be expanded as a trigonometric polynomial in θ as shown in (3.15).

$$H(\theta) = \sum_{n=0}^{M} a(n)\cos(n\theta) \tag{3.15}$$

Defining a positive valued weighting function W(θ) and the target function T(θ), we can define the weighted error function E(θ) as shown in (3.16).

$$E(\theta) = W(\theta)\,[H(\theta) - T(\theta)] \tag{3.16}$$

The *M*th order polynomial H(θ) is defined by M+1 coefficients a(n) or equivalently by the M-1 local extrema $H(\theta_k)$ and the 2-boundary values at $H(\theta_{pass})$ and $H(\theta_{stop})$. The problem is that we don't know the locations of the local extrema. The Remez multiple exchange algorithm rapidly locates these extremal positions by iterating from an initial guess of their positions to their actual positions. A ubiquitous design algorithm written by McClellan, Parks, and Rabiner expanded on the original Parks and McClellan design and has become the standard implementation of the Remez algorithm. It is embedded in most FIR filter design routines. The process proceeds as follows: An initial estimate of the extremal frequencies θ_k is assigned to the target function $T(\theta_k)$ with alternating sign offsets of the form shown in (3.17) and in Figure 3.12. A polynomial is generated that passes through these initial points using the Lagrange interpolator as shown in (3.18).

$$\hat{H}(\theta_k)=T(\theta_k)+(-1)^k \varepsilon / W(\theta_k) \tag{3.17}$$

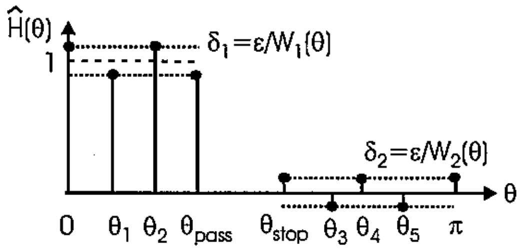


Figure 3.12 Distribution of Initial Estimates of Extrema for Multiple Exchange Algorithm

$$H(\theta)=\sum_{k=0}^{M}\hat{H}(\theta_k)P_k(\theta)$$
$$\text{where } P_k(\theta)=\frac{\prod_{m=0,m\neq k}^{M}(\theta-\theta_m)}{\prod_{m=0,m\neq k}^{M}(\theta_k-\theta_m)} \tag{3.18}$$

The polynomial $\hat{H}(\theta)$ passes through the initial sample points, but generally these points do not correspond to the local extrema points. The polynomial is sampled at a dense grid, on the order of 16 times the number of extremal points, and the samples are searched to locate the true extremal points as indicated in Figure 3.13. The locations of the previous estimate are replaced with the locations of the extremal positions of the polynomial formed from the previous estimate. This exchange is illustrated in Figure 3.14. The process of exchanging previous estimates with improved estimates continues until the amplitudes of the local extrema are within the specified error exit criteria. The algorithm converges quite rapidly, typically on the order of 4-to-6 iterations. After convergence, the polynomial is sampled at M+1 equally spaced positions and inverse transformed to determine its impulse response.

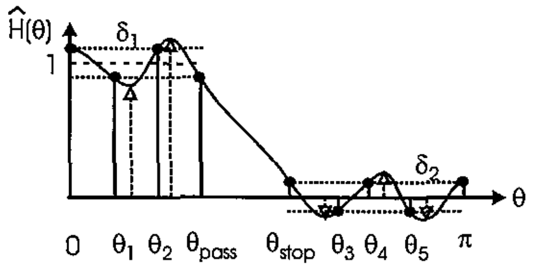


Figure 3.13 Polynomial Passed Through Initial Estimate Sample Points

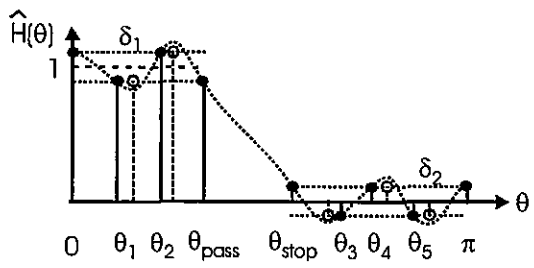


Figure 3.14 Exchanging Sample Points for Iteration k+1 with Sample Points Located at Extremal Values Obtained from Estimate at Iteration k

The example we cited to describe the manner by which the Remez algorithm iteratively converges to the Tchebyschev solution is an even symmetric filter with an odd number of coefficients. A slight variation of the design process is required when the filter is even symmetric with an even number of taps, or when the filter is odd symmetric with an even or an odd number of coefficients. The variation is transparent to the user of standard design tools so we will not discuss the details here. The interested reader should read the material presented in *Handbook for Digital Signal Processing*, edited by Mitra and Kaiser.

The Remez algorithm is also known by other names. These include the Parks-McClellen or P-M, the McClellen, Parks, and Rabiner or MPR, the Equiripple, and the Multiple Exchange. As mentioned earlier, the MPR version of the algorithm permeates the community. It is very versatile, capable of designing FIR filters with various frequency responses including multiple pass bands and stop bands and with independent control of ripple levels in the multiple bands. We limit our discussion to low pass filters.

The first question we address is what is the functional description of (3.14) which relates the filter length and the filter parameters. A number of empirically derived approximations have been published that provide an estimate of filter lengths designed by the Remez algorithm from the filter specifications. A simple estimate based on the relationship between transition bandwidth stop band side-lobe levels presented in Table 3-1, published by harris [sic], is shown in (3.19) and one published by Herrmann is shown in (3.20). The Herrmann approximation is used in the MATLAB function *remezord* with the standard caveat that the estimate often underestimates the filter order, and the user should verify performance and increment the filter length if necessary and repeat the design and verify process.

$$N = \frac{f_S}{\Delta f} K(\delta_2) \cong \frac{f_S}{\Delta f} \frac{Atten(dB)}{22} \tag{3.19}$$

$$\begin{aligned}
&N = \frac{f_S}{\Delta f} K(\delta_1, \delta_2, \Delta f, f_S) \\
&K(\delta_1,\delta_2,\Delta f,f_S) = c_1(\delta_1)\log(\delta_2) + c_2(\delta_1) + c_3(\delta_1,\delta_2)(\frac{\Delta f}{f_S})^2 \\
&c_1(\delta_1) = (0.0729*\log(\delta_1))^2 + 0.07114*\log(\delta_1) - 0.4761 \\
&c_2(\delta_1) = (0.0518*\log(\delta_1))^2 + 0.59410*\log(\delta_1) - 0.4278 \\
&c_3(\delta_1, \delta_2) = 11.01217 + 0.541244*(\log(\delta_1) - \log(\delta_2))
\end{aligned} \tag{3.20}$$

Figure 3.15 is a parameterized set of plots showing how the Herrmann estimate (3.20) of the multiplier factor $K(\delta_1,\delta_2)$ varies with Pass band Ripple δ_1, Stop band Ripple δ_2, and Transition Bandwidth $\Delta f/fs$. The curves correspond to values of in-band ripple equal to 10%, 1%, 0.1%, and 0.01% and further for three values of transition bandwidth equal to 1%, 5%, and 10% of the sample rate. We note that the filter length increases when the filter requires reduced levels of either in-band or out-band ripple as well as reductions in transitional bandwidth. The lesson here is that we should not over satisfy filter specifications, since doing so results in additional processing load for the filter. Plotted on the same figure is the harris [sic] estimate (3.19), which supplies estimates in the center of the solution space from which interaction with design routines can be used to refine the estimate.

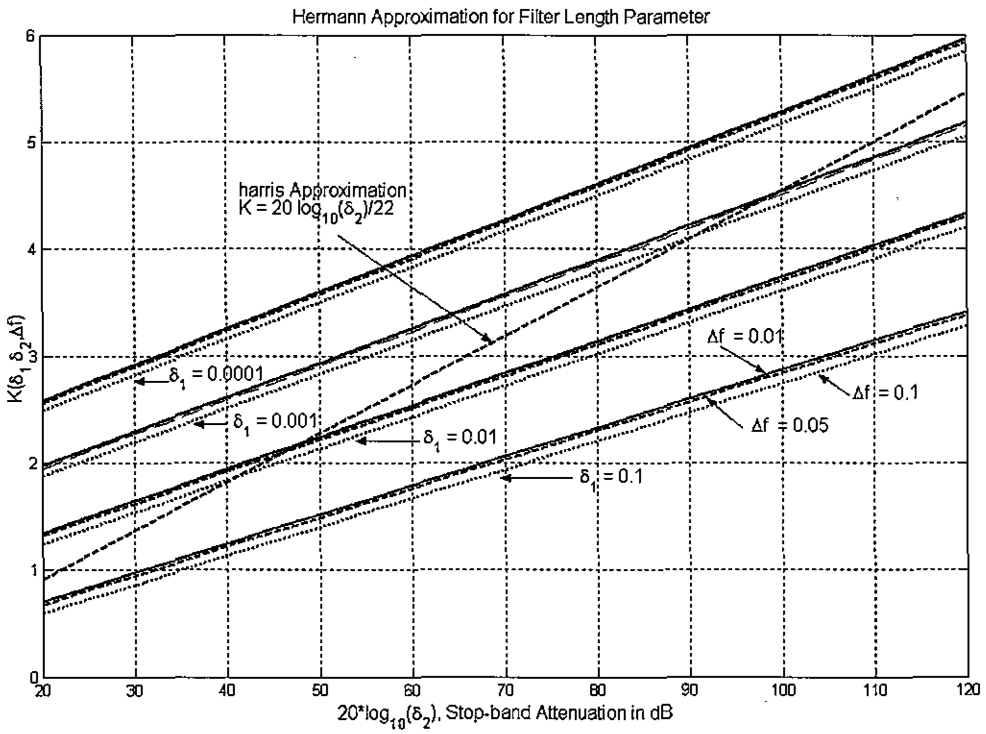


Figure 3.15 Multiplier Parameter $K(\delta_1,\delta_2,\Delta f)$ as Function of Pass band Ripple δ_1, Stop band Ripple δ_2, and Transition Bandwidth $\Delta f/fs$

The weighting function $W(\theta)$ shown in (3.16) is embedded in the MPR version of the Remez algorithm as the penalty function $P(\theta)$ and the weight vector W in the MATLAB implementation of the same algorithm. The MATLAB call to the Remez algorithm uses a frequency vector and gain vector to form a connect-the-dot target function and a weight vector specifying a weight value per interval. MATLAB has two curious conventions of which the user should be aware. The first is that the N in the call to the Remez algorithm is the polynomial order rather than the number of filter coefficients. An *N*th order polynomial is defined by N+1 coefficients, thus if we want a 57 tap filter we use 56 in the function call. MATLAB also uses a frequency axis normalized to the half sample rate rather than normalized to the sample rate: i.e., $f_{norm} = f/(fs/2)$. A MATLAB call to design a low pass filter would have this form:

```
hh=remez(N-1, [0 f1 f2 fs/2]/(fs/2), [1 1 0 0], [w1 w2]);
hh=remez(N-1, [0 f1 f2 f3]/(f3), [1 1 0 0], [w1 w2]);
```

The *remez* function call builds the initial arrays T(f) and W(f) introduced in (3.17) and shown in Figure 3.16. The algorithm then builds the function H(f) with ripple levels δ_1 and δ_2 that satisfy (3.21).

$$\delta_1 W_1 = \delta_2 W_2 \tag{3.21}$$

To obtain a design with out-of-band ripple δ_2 equal to 1/10th of in-band ripple δ_1 we set W_2 to be 10 W_1, or the weight vector W=[1 10].

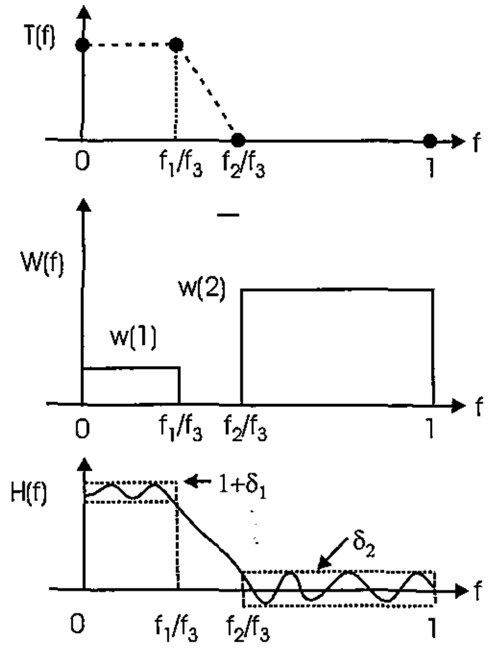


Figure 3.16 Input and Output Arrays in the Remez Algorithm

Example 3.2 Remez Design of Low pass FIR Filter

Design a low pass FIR filter with the Remez algorithm that meets the same specifications as the window design Example 3.1.
The expanded filter specifications are:

Sample Rate	100 kHz
Pass Band	±10 kHz

Stop Band	±15 kHz
Min. Atten.	60-dB
Pass band Ripple	0.1-dB (1.2%)

From the filter specifications we estimate the filter length N from Figure 3.15 to be:

$$N \cong (fs/\Delta f)^* K(\delta_1,\delta_2)$$
$$= (100/5)^*2.5 = 50 \text{ taps}$$

We can also obtain a comparable estimate from the Herrmann estimate in MATLAB by using the call to remezord as shown:

NN = remezord([f_1 f_2],[a_1 a_2],[δ_1 δ_2],fs)

NN = remezord ([10 15], [1 0], [0.01 0.001], 100)

The response to this call is: NN = 51

Note this filter requires a smaller number of coefficients, 51 as opposed to 75, for the same filter designed by the Kaiser-Bessel window design. The filter length is smaller because the in-band ripple for the Remez design is larger than the in-band ripple of the window design. We now use the MATLAB Remez function to design the filter. The estimated filter length of 51 taps did not meet the 60-dB specifications. The filter length had to be increased to 55 to satisfy the attenuation requirement. The call is of the form shown next and the time and frequency response of the filter is shown in Figure 3.17. Note the equal ripple spectral response in both pass band and stop band. By design, the pass band ripple is 1-part-100 (0.1-dB) while the stop band ripple is 1-part-1000 (60-dB).

```
h3=remez(NN-1,[0 f1 f2 fs/2]/(fs/2),[1 1 0 0],[w1 w2]);
h3=remez(54, [0 10 15 50]/50, [1 1 0 0], [1 10]);
```

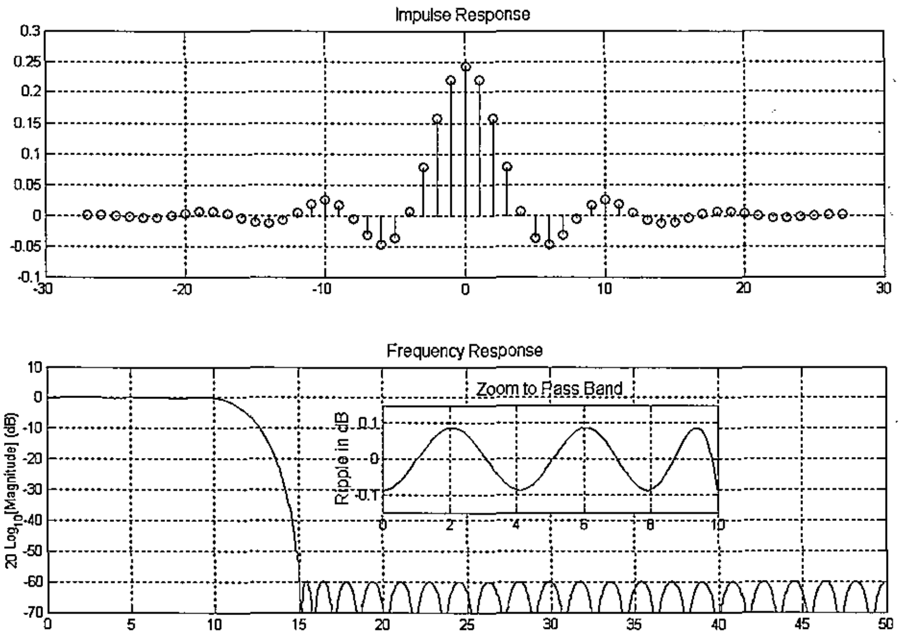


Figure 3.17 Time and Frequency Response of FIR Filter Designed with the Remez Algorithm

3.3.1 Equiripple vs. 1/f Ripple Designs

We note that the filter designed by the equiripple design routine exhibits equal ripple in both pass band and stop band. We would be surprised had it not since we designed it for that property, the property being optimum in the weighted Tchebyschev sense. The stop band spectrum has a rate of attenuation of 0-dB per octave. We recall that a spectrum has a rate of decay related to the order of the discontinuity of its time domain signal. For instance, a time signal with discontinuous amplitude (such as a rectangle) has a spectrum that decays as 1/f or –6-dB per octave. Similarly a time signal that has a discontinuous first derivative (such as a triangle) has a spectrum that decays as $1/f^2$ or –12-dB per octave. The rate of decay for the envelope of a spectrum is shown in (3.22) where k is the order of the time derivative in which a discontinuity appears. Thus if the discontinuity resides in the zeroth derivative, the signal itself is discontinuous, and the spectrum decays as 1/f.

$$\text{Asymptotic Decay Rate} = \frac{1}{f^{(K+1)}} \tag{3.22}$$

This leads to an interesting observation! If the rate of decay is zero, then the discontinuity resides in the –1 derivative, in fact in the first integral of the signal. But if the integral is discontinuous, then the function must contain an impulse. Stated more directly, a FIR filter that exhibits constant level side-lobes has impulses in its time series. We may recall that the Dolph-Tchebyshev window, the window with minimum main lobe width for a given side-lobe level, is characterized by constant level side-lobes. It exhibits a pair of end-point impulses that prevented its use as a shading function in beam-forming applications. The Taylor window was devised to suppress the boundary value impulses and is a common shading function in the radar community.

If we pay particular attention to the end points of the filter designed by the Remez algorithm we often find what appear to be end-point outliers but are in fact the impulses responsible for the constant level spectral side-lobes. The size of the impulse is on the order of the size of the spectral side-lobes and might be overlooked on the scale of the filter coefficient set.

Figure 3.18 shows the impulse response and frequency response of a filter designed with the Remez algorithm. A close-up detail of the end segment of the impulse response clearly shows the outlier. When this sample is clipped to match the amplitude of its neighbor, the filter loses its constant side-lobe characteristic and exhibits a 1/f rate of spectral decay. The 1/f asymptotic spectral decay is usually accompanied by a 6-dB increase of near-in side-lobes. If we use the clipping to obtain the 1/f side-lobe we can compensate for the spectral rise by designing the Remez filter with 6-dB additional attenuation for which the spectrum after clipping the end point can rise by the allotted margin. While most filters exhibit this outlier occasionally it is not apparent, and for those cases, rather than clip the end point, we can attenuate the boundary samples to modify the side-lobe behavior.

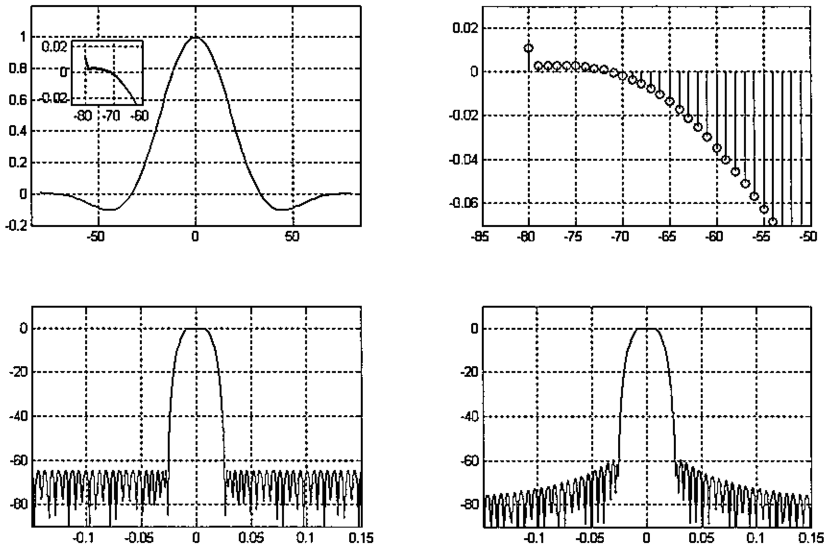


Figure 3.18 Remez Impulse Response, Showing Detail of End Point, Close-up of End Point, and Spectra of Original Filter and of Filter with Clipped End Point

Why would we want to have the spectrum have a 1/f decay rate rather than exhibit equiripple? There are two reasons, both related to system performance. The first is integrated side-lobes levels. We often build systems, as shown in Figure 3.19, comprising a digital filter and a resampling switch. Here the digital filter reduces the bandwidth and is followed by a resampling switch that reduces the output sample commensurate with the reduced output bandwidth. When the filter output is resampled, the low-level energy residing in the out-of-band spectral region aliases back into the filter pass band. When the reduction in sample rate is large, there are multiple spectral regions that alias or fold into the pass band. For instance, in a 16-to-1 reduction in sample rate, there are 15 spectral regions that fold into the pass band. The energy in these bands is additive and if the spectral density in each band is equal, as it is in an equiripple design, the folded energy level is increased by a factor of sqrt(15). To prevent the piling-up of the aliased energy we redesign the filter so that it exhibits 1/f side-lobe attenuation.

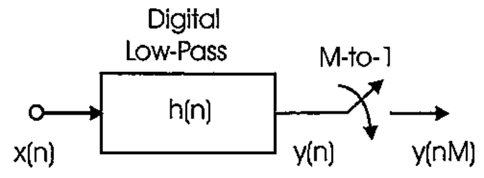


Figure 3.19 Resampling Low pass Filter

For a specific example, the filter presented in Figure 3.20 designed for 60-dB side-lobe levels is used in a 32-to-1 down sampling application. If the side-lobes are equiripple at 60-dB the integrated side-lobe level is –36.1-dB which, when distributed over the remaining bandwidth of 1/32 (–15.1-dB) of input sample rate, results in an effective alias side-lobe suppression of –51.2-dB, equivalent to a 9-dB loss. The filter was redesigned for –67.5-dB equiripple, and the numbers obtained for this design are an integrated side-lobe level of –43.7-dB and an effective alias side-lobe level of –58.8-dB, which matched the expected 7.5-dB improvement. After clipping the end point of the redesigned filter, the close-in side-lobe rose to –62-dB as the side-lobes acquired the 1/f attenuation rate. The numbers for this variant are impressive, the filter exhibiting –51.1-dB integrated side-lobes and an effective alias side-lobe level of –66.2-dB. This represents a 14-dB improvement in aliased spectral levels relative to the uniform side-lobe filter operating in the same resampling mode.

The second reason we may prefer FIR filters with 1/f side-lobe attenuation as opposed to uniform side-lobes is finite arithmetic. A filter is defined by its coefficient set and an approximation to this filter is realized by a set of quantized coefficients. Given two filter sets h(n) and g(n), the first with equiripple side-lobes, the second with 1/f side-lobes, we form two new sets, $h_Q(n)$ and $g_Q(n)$, by quantizing their coefficients. The quantization process is performed in two steps: first we rescale the filters by dividing by the peak coefficient. Second, we represent the coefficients with a fixed number of bits to obtain the quantized approximations. These operations are shown in (3.23).

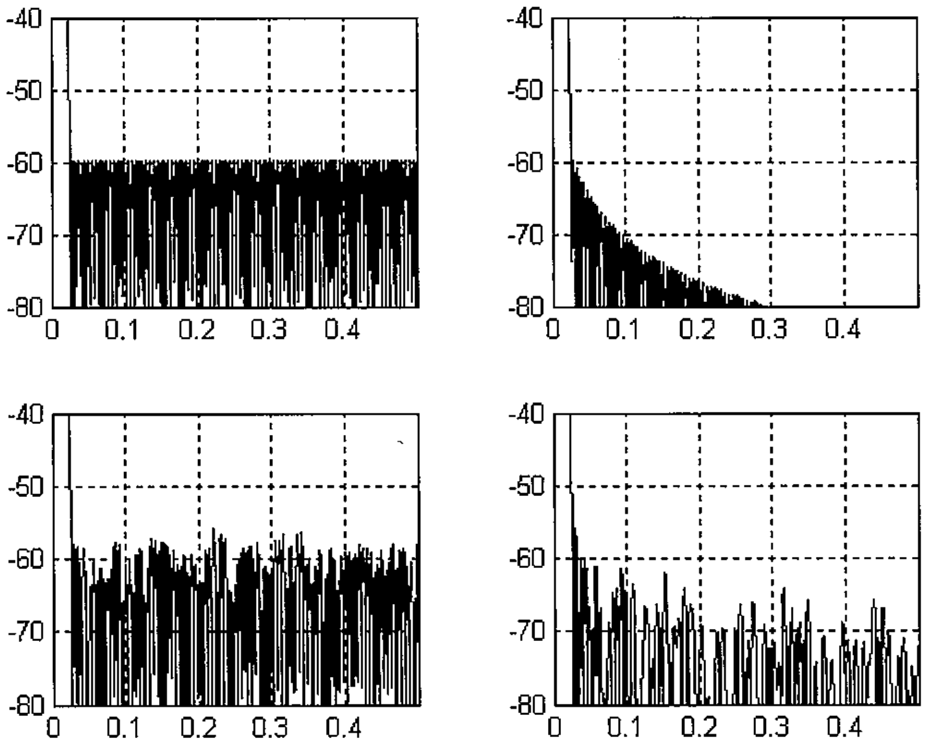


Figure 3.20 Frequency Response of Reference Filters with Equiripple and 1/f Side-lobes and of 10-bit Quantized Version of Same Filters

$$\begin{aligned} h_{SCALED} &= h / \max(h) \\ h_{QUANT} &= round(h_{SCALED} * 2^{(bits-1)}) / 2^{(bits-1)} \end{aligned} \tag{3.23}$$

The zeros of a FIR filter residing on the unit circle perform the task of holding down the frequency response in the stop band. The interval between the zeros contains the spectral side-lobes. When the interval between adjacent zeros is reduced, the amplitude of the side-lobe between them is reduced and when the interval between adjacent zeros is increased the amplitude of the side-lobe between them is increased. The zeros of the filters are the roots of the polynomials H(Z) and G(Z). The roots of the polynomials formed by the quantized set of coefficients $H_{quant}(Z)$ and G_{quant} (Z) differ from the roots of the unquantized polynomials. For small changes in coefficient size, the roots exhibit small displacements along the unit circle from their nominal position. The amplitude of some of the side-lobes must increase due to this root shift. In the equiripple design, the initial side-lobes exactly meet the design side-lobe level with no margin for side-lobe increases due to root shift caused by coefficient quantization. On the other hand, the filter with 1/f side-lobe levels has plenty of margin for side-lobe increases due to root shift caused by coefficient quantization. Figure 3.20 presents the frequency response of two reference filters, one with equiripple side-lobes and the other with 1/f side-lobes. We see their spectra for unquantized and for their 10-bit quantized versions. As expected the side-lobes of the quantized version of the equiripple filter exceed the 60-dB attenuation level while the side-lobes of the quantized 1/f side-lobes continue to meet the required 60-dB attenuation level. For reference, the integrated side-lobes for the four cases are listed in Table 3-2.

Table 3-2 Integrated Side-lobe Levels for Equiripple and 1/f Side-lobe FIR Filters, Unquantized and 10-bit Quantized Versions

	Equiripple Side-lobes	1/f Side-lobes
Unquantized	–36.1-dB	–49.3-dB
Quantized (10-bits)	–35.4-dB	–45.1-dB

The final question we address is how do we design FIR filters with 1/f side-lobes? We need a design technique to replace the trick we illustrated earlier that obtained the desired 1/f side-lobes by clipping the end-point outliers because the process also affects the in-band ripple and may not work for a particular filter design. The tool we apply to side-lobe control is the weighting function described in (3.17) and in Figure 3.16. If we have access to the weight function array we can modify the weight function in the stop band so that it increases linearly with frequency as shown in Figure 3.21. When the weight function increases with frequency, the resultant side-lobe levels vary inversely with frequency. The frequency dependent weighting function is used in the QED-2000 design package. If the weight function array is not directly accessible, we can use a stair-step weighting function in adjacent frequency intervals. This option is also shown in Figure 3.21. A MATLAB call to the Remez algorithm that uses the staircase weight function is shown here.

```
ff=[0 0.6 3.4 5.0 5.1 7.5 7.6 10.0 10.1 15.0. 60.1 64.0]
ff=ff/64.0;
aa=[1  1   0   0   0   0   0    0    0    0    0    0 ];
ww=[ 1      1.5      3      4.5        6          21    ];
hh=remez(154, ff, aa, ww);
```

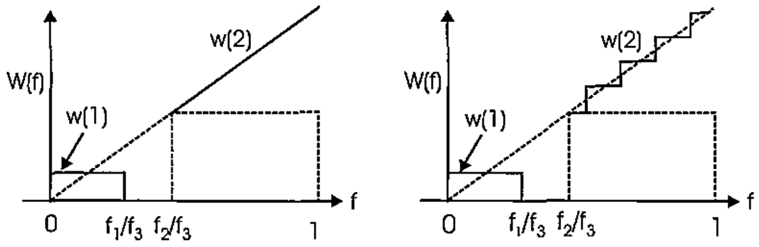


Figure 3.21 Modified Weight Functions for Remez Algorithm to Obtain Staircase Approximation to 1/f Side-lobes

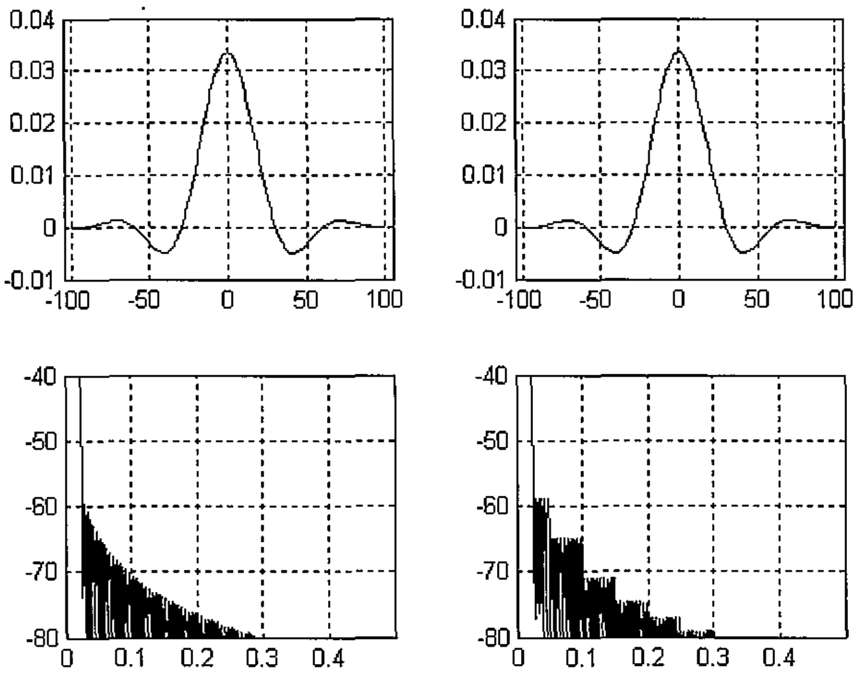


Figure 3.22 Remez Filter Time and Frequency Response Designed with Linear Frequency and with Staircase Frequency Weight Function

Figure 3.22 shows the time and frequency response of FIR filters designed with the two types of frequency-dependent weight functions. We can see the staircase change in side-lobes in the spectrum shown in the lower-right subplot. The two filters exhibit comparable integrated side-lobe levels of –47.5 and –48.5-dB.

The MATLAB script file that sets the penalty function for the Remez algorithm is *remezfrf.m* (here frf means frequency response function). A modified version called *myfrf.m*, available from the book's companion website listed on the title page, forms a 1/f stop band ripple. The MATLAB call for a standard filter and for a modified filter design that uses the script file is shown here.

```
h1=remez(154,[0 0.6 3.4 64]/64,[1 1 0 0],[1 10]);
h2=remez(154,[0 0.6 3.4 64]/64,{'myfrf',[1 1 0 0]},[1 10]);
```

Figure 3.23 shows the frequency response of FIR filters designed with the default *remezfrf.m* and the *myfrf.m* frequency-dependent penalty functions. Also shown is a zoom to the pass band ripple to illustrate that there is only minor effect in the pass band when tilting the stop band ripple.

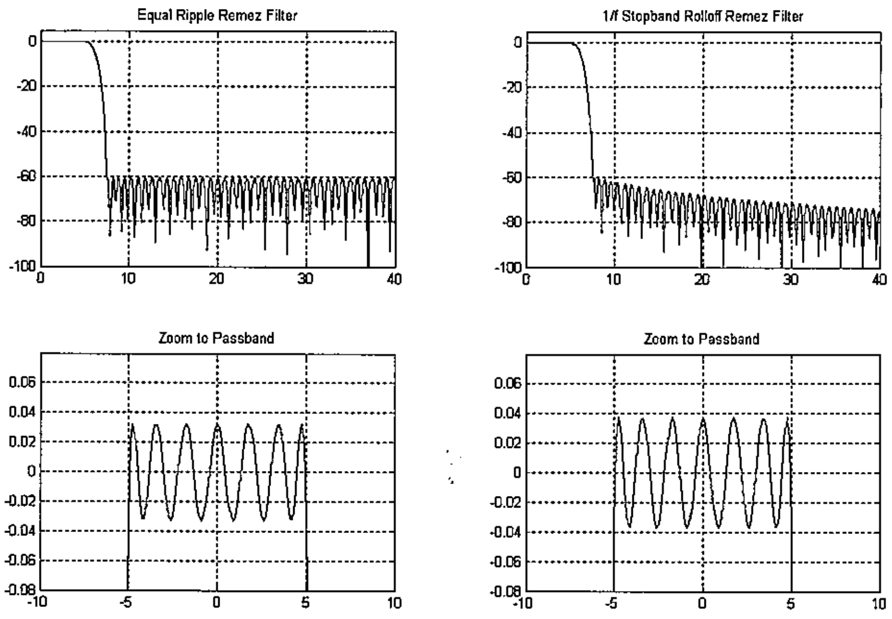


Figure 3.23 Spectral Response of Filters Designed by the Default *remezfrf* and by the *myfrf* Routines in Remez Call

3.3.2 Acceptable In-band Ripple Levels

This section addresses the matter of selecting the in-band ripple specifications for a FIR filter. The classic problem is that we know how to specify the in-band ripple in a filter but have little guidance of how to determine that desired level. To assist in the task of selecting an acceptable level of in-band ripple it is useful to understand the effect of in-band ripple on signals moving through the filter. We start with the requirements for a distortionless channel, one for which the output signal is at most a delayed and scaled version of the input signal. Figure 3.24 identifies the input and output parameters of a linear filter.

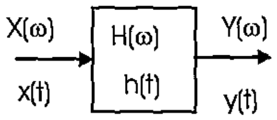


Figure 3.24 Input and Output of a Linear Filter

In order for a wave shape to pass through a filter without distortion, often referred to as distortionless transmission, we require the relationship of (3.24) to be valid for all x(t) with bandwidth less than the filter's bandwidth.

$$y(t) = A\,x(t-\tau) \tag{3.24}$$

Equation (3.25) is the Fourier transform of the two sides of (3.24).

$$Y(\omega) = A\,X(\omega)\,e^{-j\omega\tau} = X(\omega)\,A\,e^{-j\omega\tau} \tag{3.25}$$

Since the output transform Y(ω) is the product of the input transform X(ω) and the filter transform H(ω), we conclude that H(ω), the distortionless filter, satisfies (3.26).

$$H(\omega) = A\,e^{-j\omega\tau} \tag{3.26}$$

We recognize that the distortionless filter must exhibit a constant, but otherwise arbitrary, nonzero amplitude gain A, and a phase shift proportional to frequency, often identified as linear phase shift, with the proportionality factor being the time delay. We usually emphasize the requirement for linear phase, because linear phase is not an attribute of recursive analog filters, and is purchased by the use of additional filters known as phase equalizer filters. We know that linear phase shift, a property equivalent to pure time delay, can never be achieved exactly with lumped linear circuit components but can be achieved with distributed components which form transmission lines that respond with solutions to the wave equation. Analog phase equalizers in the analog domain are used to obtain equiripple approximations to linear phase slope.

The attraction, and an often-cited advantage, of nonrecursive filters is the ease with which they can achieve linear phase shift. To achieve linear phase in a FIR filter, its impulse response must exhibit symmetry with respect to its center point. We thus find that linear phase shift, a difficult attribute to achieve in the analog domain, is essentially free in the sampled data domain. For reasons that escape us, additional discussion of distortion effects seems to stop here as if access to linear phase has solved the problem. This is a bit premature since we still have to address the effect of equiripple deviation from constant amplitude gain as well as the effect of the equiripple deviation from uniform phase shift of the phase equalized recursive filter.

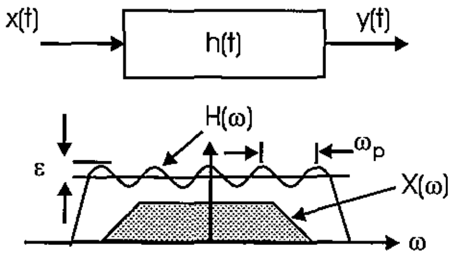


Figure 3.25 Frequency Response of Equiripple FIR Filter and Spectrum of Input Signal

Figure 3.25 presents the frequency response of an equiripple FIR filter and the spectrum of an input signal with bandwidth completely contained within the filter bandwidth. Here the amplitude response is modeled as a nominal gain of unity with a cosine ripple of amplitude ε and of period $\omega_P = 2\pi/T_P$. The filter also has a uniform group delay of T_D seconds to reflect the causality of its impulse response. The spectral response of the filter is described in (3.27). Note that the output spectrum is composed of two components, one due to the nominal unity gain with its linear phase shift and one due to the cosine ripple in addition to its linear phase shift. We partition the cosine modulation into a pair of complex exponential terms and combine these terms with the linear group delay phase term to obtain a total of three spectral components observed at the filter output.

$$\begin{aligned} Y(\omega) &= X(\omega)H(\omega) \\ &= X(\omega)\,[1+\varepsilon\cos(\omega T_P)]\,e^{-j\omega T_D} \\ &= X(\omega)e^{-j\omega T_D} + \varepsilon X(\omega)\cos(\omega T_P)e^{-j\omega T_D} \\ &= X(\omega)e^{-j\omega T_D} + \frac{\varepsilon}{2}X(\omega)e^{-j\omega(T_D+T_P)} + \frac{\varepsilon}{2}X(\omega)e^{-j\omega(T_D-T_P)} \end{aligned} \tag{3.27}$$

$$y(t) = x(t-T_D) + \frac{\varepsilon}{2}x(t-(T_D+T_P)) + \frac{\varepsilon}{2}x(t-(T_D-T_P)) \tag{3.28}$$

When we interpret the time domain response from the spectral description of the output we find three distinct time response contributions. The major component of the output signal is the time-delayed version of the input signal denoted by $x(t-T_D)$. The remaining two components are a pair of scaled and translated versions of the input signal. These components are called *paired echoes*. A pre-echo and a post-echo, each of amplitude ε/2, form the paired echoes residing on each side of the primary response time and translated by the reciprocal period of the filter ripple frequency. The structure of the paired echoes is shown in (3.28) and in Figure 3.26. Higher frequency (i.e., shorter period) spectral ripple causes larger amounts of echo time offset. Similar paired echo responses can be derived for phase ripple, except that phase ripple exhibits odd symmetry from which we establish that the echoes are also odd symmetric about the main response.

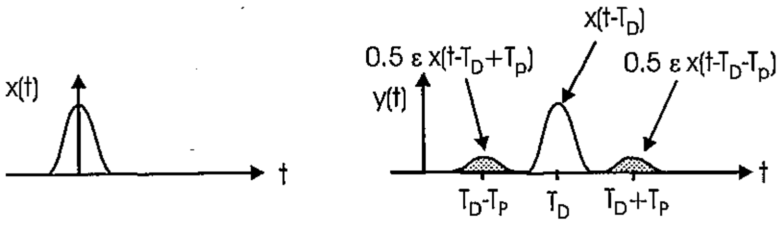


Figure 3.26 Time Signals at Input and Output of Filter with Equiripple Spectral Response

Note the dual relationship: When we multiply a time function by a time domain cosine wave, its transform, a spectrum, is scaled and translated in the frequency domain by the reciprocal period of the temporal cosine. Similarly when we multiply a frequency function by a frequency domain cosine, its transform, a time signal, is scaled and translated in the time domain by the reciprocal period of the spectral cosine.

While on the topic of dual relationships, we recognize that time domain echoes cause a periodic ripple in the frequency response of a channel often modeled as multipath related frequency selective fading. Hence, in retrospect, it is not surprising that filters exhibiting periodic frequency domain ripple are characterized by time domain echo structures. We may recall from our first course in transmission lines that the time domain interaction between the incident and reflected sinusoid due to a reflection causes periodic frequency dependent constructive and destructive cancellation. We also note that time domain reflectometers (TDRs) use this coupling between echo time-response and spectral ripple to extract time position and amplitude information from frequency domain measurement of a transmission line.

We now know why the amplitude of the filter pass band ripple is of concern to us. When used in a communication system, the ripple is a source of distortion called intersymbol interference (ISI). We use the next figure to demonstrate how a filter in a receiver signal path generates ISI. Figure 3.27 shows the impulse response and frequency response of a FIR filter that processes a selected input signal. The spectral response of the input signal is overlaid on the filter response, and as we see, the bandwidth of the input signal is fully contained in the filter bandwidth.

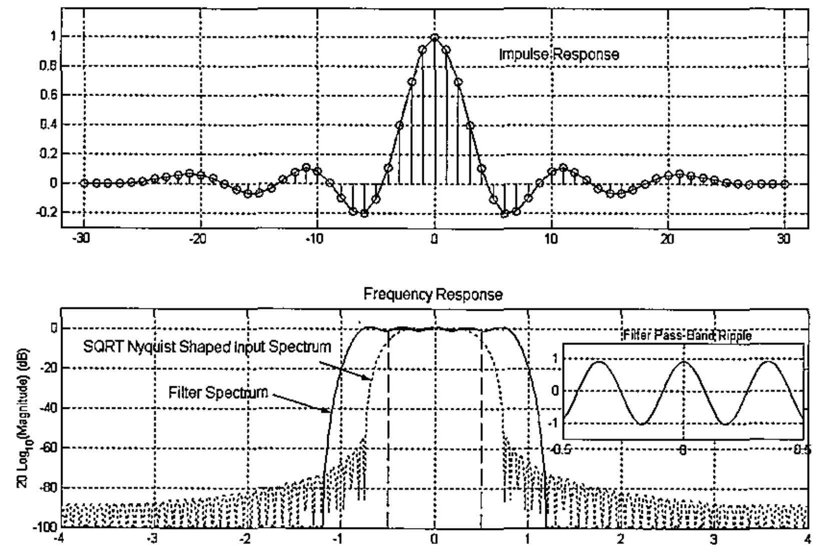


Figure 3.27 Impulse and Frequency Response of a FIR Filter. Detail of Pass Band Shows ±1-dB In-band Ripple. Spectrum of Input Signal Is Fully Contained in Filter Pass Band.

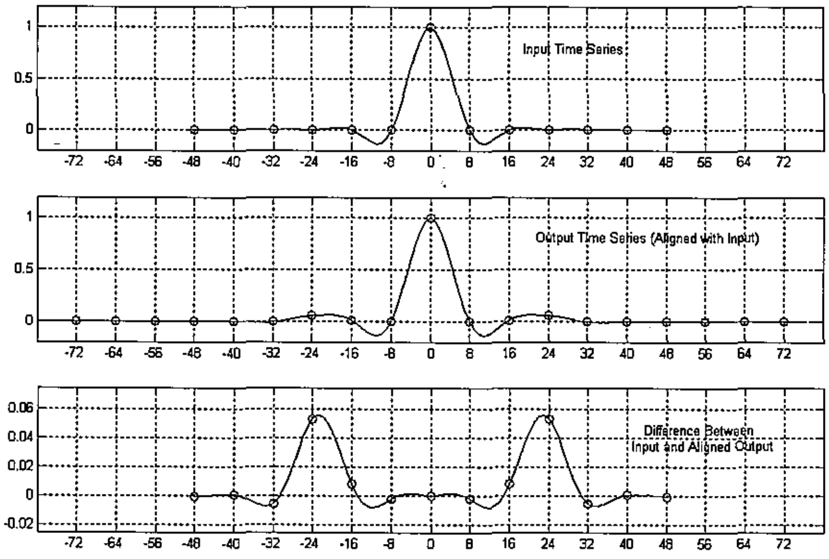


Figure 3.28 Time-aligned Input and Output Signal Waveforms and Difference of Two Waveforms to Show Paired Echoes

We also see a detail of the filter pass band region where we see that the filter exhibits 1-dB ripple. One-dB ripple is approximately 12% that presents amplitude of 0.12-ripple to the input spectrum. We also see the filter contains nearly three cycles of ripple in the bandwidth of the input signal. The number of ripple cycles in the bandwidth of the input signal is important, and we will explain why in a moment. Figure 3.28 shows the input signal from which the input spectrum of the previous figure was formed, and the time-aligned output obtained from the filter output. Also shown is the difference between the input and the time-aligned output. This difference clearly shows the pre- and postechoes due to the filter in-band ripple. These echoes are seen to be located three sample intervals either side of the main response with three cycles per signal bandwidth being the frequency of the filter ripple.

Incidentally, pre- and postechoes are formed by ripple in the phase response as well as by ripple in the magnitude response. The subtle difference is that magnitude response ripple results in an even symmetric echo pair while phase response ripple results in an odd symmetric echo pair. This is, of course, related to the even and odd symmetry of the magnitude and phase characteristics of a filter. Recursive filters, with their nonuniform phase response, are the contributors of the odd symmetric echoes. To illustrate this we examine Figure 3.29, which in the left-hand column presents the impulse response, frequency response, and the in-band group delay of an 8th order elliptic filter. The right-hand column presents the same curves for the elliptic filter in cascade with an 8th order phase equalizer. Notice the symmetric impulse response and the equal-ripple group delay response of the equalized filter. Figure 3.30 presents the input and time-aligned output time series from the elliptic filter. Also seen is the difference between the two series in which we see the odd symmetric echoes caused by the filter's group delay. Figure 3.31 presents the input and time-aligned series from the phase equalized elliptic filter. The final subplot of this figure shows the difference between the input and time-aligned output and the odd symmetric echo pair is clearly seen. We note that the IIR echo components are not exactly odd symmetric and we attribute this to even symmetric echo components due to the amplitude ripple response of the elliptic filter which also contributes echoes to the composite response.

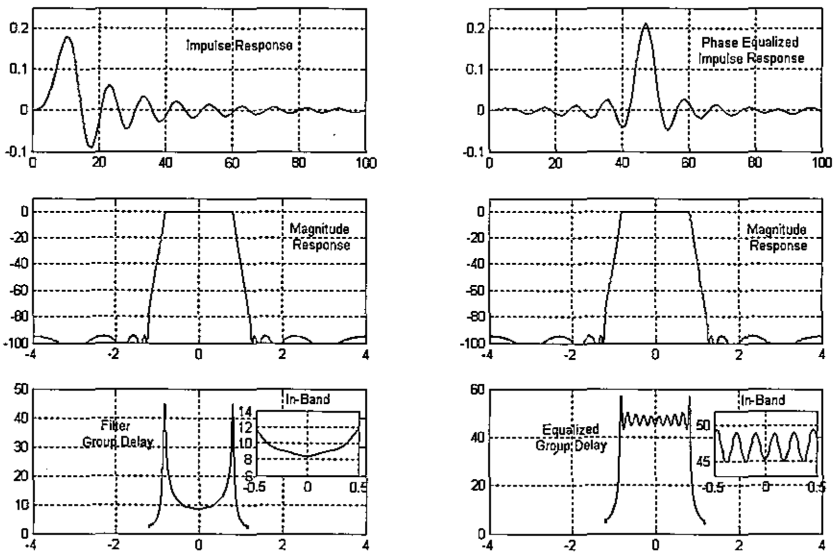


Figure 3.29 Impulse, Magnitude Response, and Group Delay Response of an IIR Filter and of a Phase Equalized Version of Same Filter. Detail of Equalized Phase Shows ±2.2-Degree In-band Ripple.

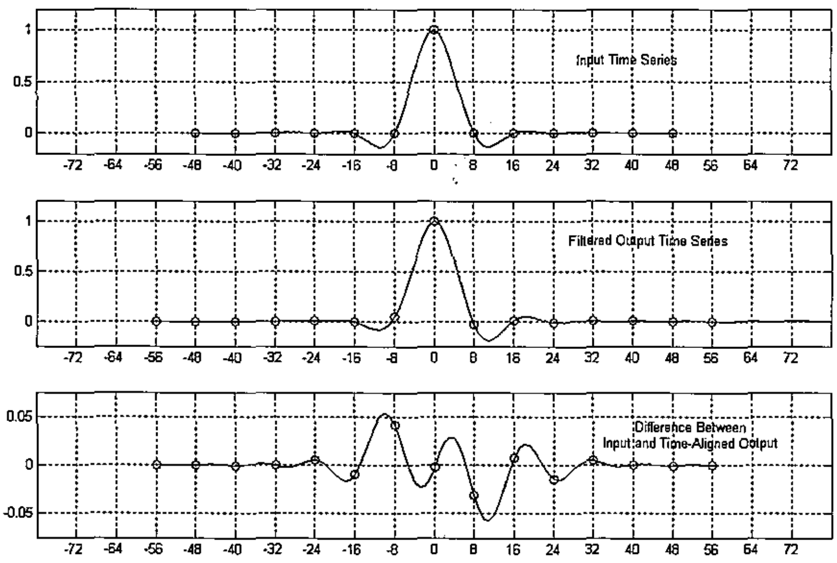


Figure 3.30 Time-aligned Input and Output Waveforms from Nonuniform Phase IIR Filter with the Difference of Two Waveforms to Show Paired Echoes

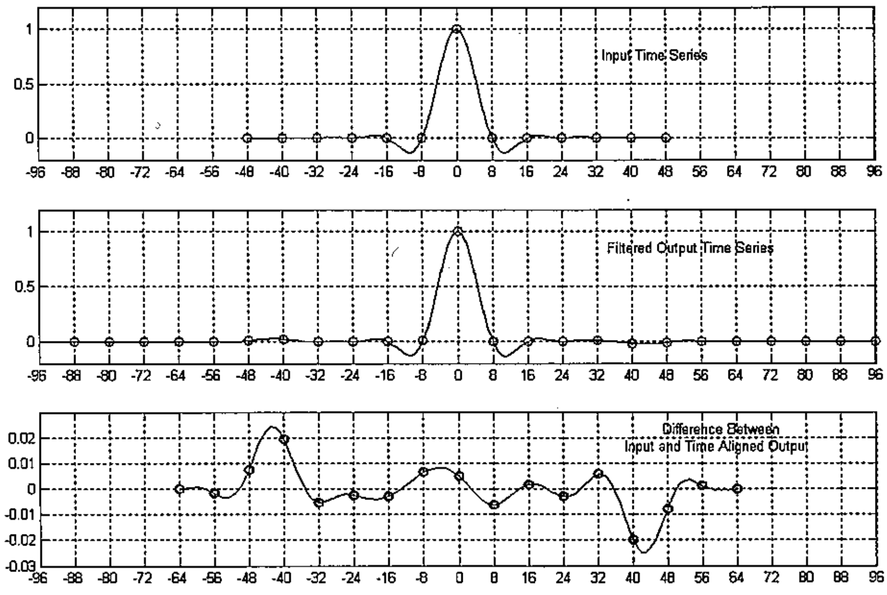


Figure 3.31 Time-Aligned Input and Output Waveforms from Phase Equalized IIR Filter with the Difference of Two Waveforms to Show Paired Echoes

We can now compare the filter responses we examined to illustrate the pre- and postecho distortion. As mentioned earlier, the input signal was a Nyquist pulse, oversampled by a factor of 8, so that the pulse has exactly 8-samples per symbol. Every 8th sample of this input pulse coincides with an expected zero crossing. The distance between the pulse peak and the zero crossing is the symbol time for this pulse. Symbols or waveforms separated by exactly this interval are orthogonal and do not interact. When signaling with the Nyquist pulse, the receiver can collect and measure each waveform independently and is thus able to communicate through band-limited channels without ISI. When we illustrated the echoes with the FIR filter, the input and output were time aligned to extract the added echoes. The filter ripple parameters of 0.12 amplitude ripple with frequency of 3 cycles per input bandwidth told us to expect a pair of echoes of amplitude 0.06 separated by three symbol durations from the primary time response. To verify this, the bottom graph in Figure 3.29 was formed as the difference between the time-aligned input and output waveforms. Here we can verify that the amplitude and location of the paired echoes closely match the values predicted from the parameters of the in band ripple. The offsets tell us the period of the frequency domain ripple was slightly less than 3 cycles per signal bandwidth. Similarly, the ripple parameters of the IIR filter, the 2.2 $*2\pi/360$ peak phase ripple with frequency of 5 cycles per input bandwidth, told us that we should expect a pair of echoes of amplitude 0.019 separated by five symbol durations from the primary time response. This echo pair is seen in subplot three of Figure 3.31.

A receiver attributes the ISI caused by the filters in the signal flow path to channel distortion, and if the receiver contains an equalizer it will attempt to remove the distortion by forming the opposing pass band gain and phase ripple. There are receivers that may not include an equalizer for which the filter-induced ISI would likely cause performance degradation. Examples include DSP-based processing of video signals as part of source coding, signal reconstruction, and simple signal processing of NTSC (National Television Standards Committee) or PAL (Phase Alternating Line) composite video signals. Radar signal processing for imaging radars suffers the same degradation due to inadequate attention to paired echoes from equiripple digital FIR filters as well as equiripple analog Tchebyschev filters in the signal-processing path.

A properly designed system would have part of its implementation loss budget assigned to the pass band ripple and then the specifications would have to convert budget into acceptable ripple level. We commented earlier that 1-dB is 12% ripple from which it follows that 0.1-dB is 1.2%, and if a system has budgeted 0.1-dB to filter losses, the composite filter chain must exhibit less than 1.2% ripple. We often design systems with 0.1-dB in-band ripple, which, except for the most stringent specifications, satisfies most system requirements. Reasonable specifications for a FIR filter might have in-band ripple of 1-part-in-100 and out-of-band ripple of 1-part-in-1000 or 1-part-in-10000. This is the justification for the use of the Remez algorithm and its variants that modify the out-of-band ripple slope.

Figure 3.32 is a block diagram of an end-to-end modulator and demodulator that will be used to illustrate the effect of in-band ripple. Here the output of the shaping filter is fed directly to the input of the matched filter without the intermediate channel that would add noise and channel distortion. The output of the matched filter is passed to the equalizer that would normally remove the distortion caused by the channel. The equalizer delivers its processed signal to a detector that forms an estimate of the data. This estimate is compared with the input to the detector and differences between the detector input and output are attributed to noise and channel distortion. The error, along with the data, directs the adaptive algorithm to adjust the equalizer weights in the direction that results in a reduction in the average detector error.

The upper-left subplot of Figure 3.33 presents the 16-QAM constellation set observed at the matched filter output. The constellation points correspond to the range of amplitudes, four for each of the in-phase and quadrature-phase components of the received and processed waveform. In the absence of noise and distortion these amplitudes are $\pm$ 1/3 and $\pm$ 1.0. We see a small variance cloud centered on the constellation points. This cloud is the observed effect of the ISI caused by in-band ripple of the shaping filter and matched filter. The upper-right subplot presents the log-magnitude of the energy in the error sequence as the adaptive equalizer whitens the error sequence by acquiring the inverse of the distortion process. The subplot in the lower-left shows the constellation diagram obtained at the equalizer output after convergence. Note the reduction in the variance cloud due to the ISI cancellation by the equalizer. Finally, the lower-right subplot presents a close up of the spectral ripple exhibited by the cascade of the shaping filter and matching filter along with the frequency response of the adaptive equalizer. Note that the spectral gain of the equalizer is the inverse of the spectral gain of the filter pair.

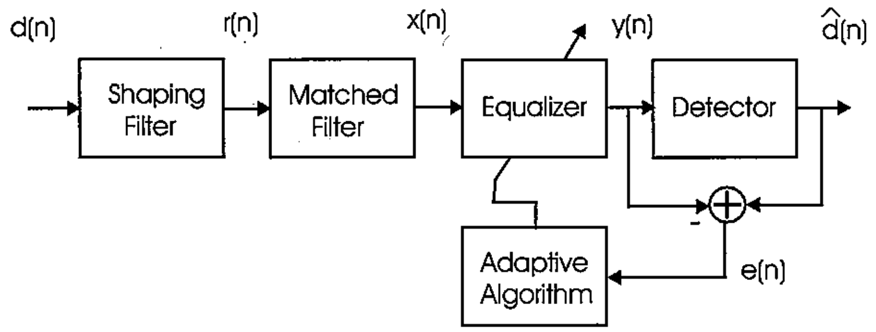


Figure 3.32 Block Diagram of 16-QAM Modulator, Demodulator Chain Used to Illustrate ISI Due to In-band Ripple of Shaping Filter and Matched Filter

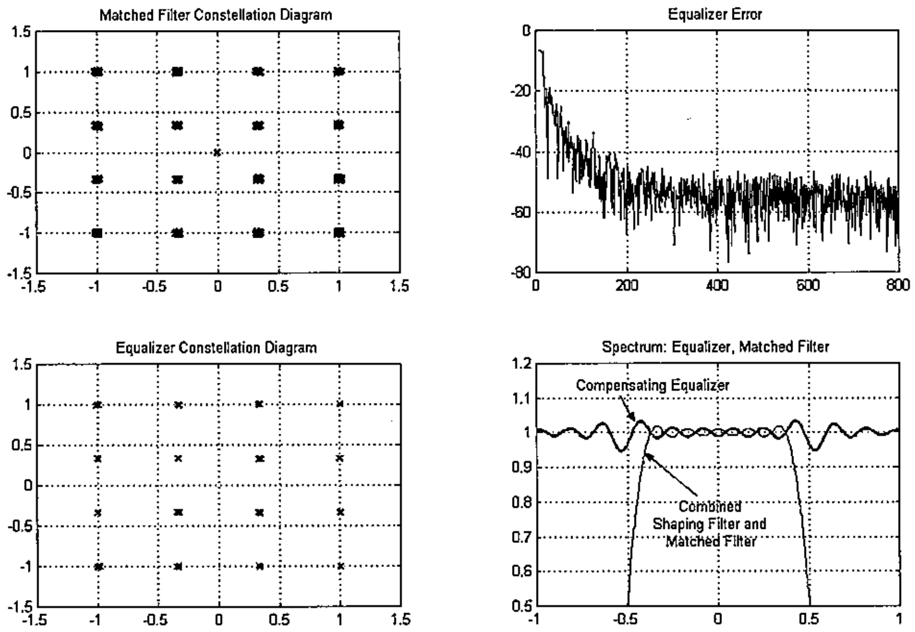


Figure 3.33 Constellation at Output of Matched Filter, Error Profile of Converging Equalizer, Constellation at Output of Equalizer, and Spectra of Shaping Filter and Compensating Equalizing

References

harris, fred, "DATA WINDOWS: Finite Aperture Effects and Applications in Signal Processing," Encyclopedia of Electrical Engineering, John Wiley & Sons, Editor, John Webster, 1999.

harris, fred, "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform," *Proceedings of the IEEE*, Vol. 88, No. 1, January 1978, pp. 51-83.

Herrmann, O., L.R. Rabiner, and D.S.K. Chan, "Practical Design Rules for Optimum Finite Impulse Response Low pass Digital Filters," *Bell Syst. Tech. J.*, 52, pp. 769-799, 1973.

Kaiser, Jim, "Nonrecursive Digital Filter Design Using I_0-sinh Window Function," *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'74)*, pp. 20 - 23, April 1974.

McClellan, J. H., T.W. Parks, and L.R. Rabiner, "FIR Linear Phase Filter Design Program," Programs for Digital Signal Processing, Chapter 5.1, pp. 5.1-1 - 5.1-13, IEEE Press. 1979.

Mitra, Sanjit, and James Kaiser, *Handbook for Digital Signal Processing*, New York, John Wiley & Sons, 1993.

Mitra, Sanjit, *Digital Signal Processing: A Computer-Based Approach*, 2nd ed., New York, McGraw-Hill, 2001.

Problems

3.1 First examine the MATLAB sinc function by typing help in the command window. Now examine the call sinc(–4:0.1:4) and describe how the sinc call responds to the arguments. Plot the subplots for the stem of the sequence and the log magnitude spectrum. The following MATLAB script accomplishes this:

```
subplot(2,1,1);
dat=sinc(-4:0.1:4);
stem(-4:0.1:4,dat);
grid
subplot(2,1,2);
ff=(-0.5:1/1024:0.5-1/1024)*10;
f_dat=fft(dat/sum(dat),1024);
plot(ff, fftshift(20*log10(abs(f_dat))))
grid
axis([-5 5 –60 10])
```

Examine the time series and the spectrum and then describe what will change if the *dat* line is changed to sinc(–8:0.1:8), and then again if it changed to sinc(–4:0.02:4). How should the scale factor on the *ff* line change for the two new sinc options? Why in the *f_dat* line is *dat* divided by *sum(dat)*? Replace *f_dat* with *f_dat=fft(dat,1024)* and examine its spectral plot.

3.2 The DTF series expansion of an N point sampled rectangle is shown next.

$$H(\theta)=\exp(-j\frac{N-1}{2}\theta)\frac{\sin(\frac{N}{2}\theta)}{\sin(\frac{1}{2}\theta)}$$

Determine the locations of this function's zero crossings.

Determine the amplitude of this function at $\theta = 0$.

Determine the value of this function at $\theta = \pi$.

Determine the location and amplitude of this function's first side-lobe.

Is the ratio of the peak value to the first side-lobe level of this function dependent on N? Show this!

3.3 Form and plot the magnitude of the 200-point FFT for a 20-point rectangle sequence (ones (1,20)) and on the same plot, plot the magnitude of the 200-point FFT of the 20-point sequence exp(j*2*pi*(0:19)/20). What can you say about the location of each transform and the zeros of the two transforms?

3.4 Form and plot the magnitude of the 200-point FFT for a 20-point sequence exp(j*2*pi*(0:19)/20) and on the same plot, plot the magnitude of the 200-point FFT of the 20-point sequence exp(-j*2*pi*(0:19)/20). What can you say about the location of each transform and the zeros of the two transforms?

3.5 Form and plot the magnitude of the 200-point FFT for a 20-point rectangle sequence (ones (1,20)) and on the same plot, plot the magnitude of the 200-point FFT of the 20-point sequence exp(j*2*pi*(0:19)/20). What can you say about the location of each transform and the zeros of the two transforms?

3.6 Using the log magnitude spectrum of the 512-point FFT of 51-point sequences, determine the main lobe widths and maximum side-lobe levels of the following traditional window functions: Rectangle, Hann, Hamming, Blackman-harris 3-term, Blackman-harris 4-term, and Kaiser with parameter 10.

3.7 Form a plot showing the progression of main lobe width and side-lobe levels (in-dB) for the Kaiser window over a range of the parameter β equal to the integer values 1-through-10.

3.8 The expression defining the Fourier transform of the Kaiser window is listed in Equation (3.13). Determine and form a plot of the first 10 zeros of this spectrum for a range of parameter β equal to the integers 0-through-10.

3.9 Form plots of log magnitude spectrum of 512-point FTT of 51-point rectangle windowed by Kaiser window for range of parameter values 0-through-10

3.10 Use the Remez algorithm to design a low pass FIR filter satisfying the following specifications:

Fs:	20 kHz		
Pass Band:	0-to-3 kHz	In-band Ripple:	0.1-dB
Stop Band:	5-to-10 kHz	Stop Band:	60-dB

Estimate the length of the filter using Figure 3.15, then verify this estimate with MATLAB *remezord.*

Verify the results of the Remez design by examining in-band Ripple and out-of-band attenuation levels. Change filter length and weighting vector as appropriate to achieve filter design specifications.

3.11 Design a low pass filter with the Remez algorithm to meet the following three specifications:

	Filter 1	Filter 2	Filter 3
Fs:	20 kHz	20 kHz	20 kHz
Pass Band:	0-to-3 kHz	0-to-4 kHz	0-to-5 kHz
Stop Band:	5-to-10 kHz	7-to-10 kHz	8-to-10 kHz
Pass band Ripple:	0.1-dB	0.1-dB	0.1-dB
Stop Band:	60-dB	60-dB	60-dB

The three filters have the same specifications, including transition bandwidth, but different pass band band edges: Using 6 subplots, plot the impulse response and frequency response of the three filters.

Comment on how transition bandwidth affects filter length and on how pass band bandwidth affects the length and shape of filter impulse response.

3.12 Repeat Problem 3.11 but change stop band attenuation from 60-dB to 80-dB.

Comment on how increased out-of-band attenuation affects length and shape of filter impulse response.

3.13 Repeat Problem 3.11 but change in-band ripple from 0.1 to 0.01.

Comment on how decreased in-band ripple affects length and shape of filter impulse response.

3.14 A low pass FIR filter is to be designed with the following specifications:

Fs:	100 kHz		
Pass Band:	0-to-20 kHz,	In-band Ripple:	0.1-dB
Stop Band:	25-to-50 kHz;	Stop Band:	60-dB

a) Use the standard MATLAB *remez* algorithm with weights appropriate to obtain the 0.1-dB in-band ripple and the 60-dB out-of-band attenuation.

b) Use the MATLAB *remez* algorithm with the modified *myfrf* script file to obtain sloping side-lobes.

c) Plot and compare the pass band ripple, and the in-band ripple of the two designs.

3.15 A low pass FIR filter is to be designed with the following specifications:

Fs:	100 kHz		
Pass Band:	0-to-25 kHz	In-band Ripple:	0.1-dB
Stop Band:	35-to-50 kHz	Stop Band:	60-dB

a) Use the MATLAB *remez* algorithm with the modified *myfrf* script file to obtain a design with sloping side-lobes.

b) Use a Kaiser window for a windowed design to meet the same filter specifications.

c) Plot and compare the impulse response and the log magnitude spectrum of the pass band ripple, and the in-band ripple of the two designs.

3.16 Use the Remez algorithm to design an equiripple low pass filter with the following specifications:

Fs:	100 kHz		
Pass Band:	0-to-10 kHz	In-band Ripple:	0.1-dB
Stop Band:	15-to-50 kHz	Stop Band:	60-dB

Use the following MATLAB script to simulate quantizing coefficients to b-bits:

```
hh=remez(N,[0 10 15 50]/50,[1 1 0 0],[1 10]);
hh_q=floor(2^(b-1)*hh)/2^(b-1);
```

Plot a sequence of spectra corresponding to a range of quantization bit levels of 8, 10, 12, and 14 bits. How many bits are required to maintain 60-dB side-lobes?

3.17 Use the *remez* algorithm and the *myfrf* script file to design a –6-dB/octave stop band low pass filter with the following specifications:

Fs:	100 kHz		
Pass Band:	0-to-10 kHz	In-band Ripple:	0.1-dB
Stop Band:	15-to-50 kHz	Stop Band:	60-dB

Use the following MATLAB script to simulate quantizing coefficients to b-bits:

```
hh=remez(N,[0 10 15 50]/50,{'myfrf',[1 1 0 0]},[1 10])
hh_q=floor(2^(b-1)*hh)/2^(b-1)
```

Plot a sequence of spectra corresponding to a range of quantization bit levels of 8, 10, 12, and 14 bits. How many bits are required to maintain 60-dB side-lobes?

3.18 Repeat Problem 3.16 but scale the impulse response by its maximum value prior to quantization.

The following MATLAB script can accomplish this:

```
hh=remez(N,[0 10 15 50]/50,[1 1 0 0],[1 10]);
hh_scl=hh/max(hh);
hh_scl_q=floor(2^(b-1)*hh_scl)/2^(b-1);
```

3.19 Repeat Problem 3.17 but scale the impulse response by its maximum value prior to quantization.

The following MATLAB script can accomplish this:

```
hh=remez(N,[0 10 15 50]/50,{'myfrf',[1 1 0 0]},[1 10]);
hh_scl=hh/max(hh);
```

```
hh_scl_q=floor(2^(b-1)*hh_scl)/2^(b-1);
```

3.20 Use the Remez algorithm to implement the impulse response of the following two filters:

```
hh1=remez(55,[0 10 11.5 40]/40,[1 1 0 0], [1 100]);
hh2=remez(15,[0 1 10 40]/40,[1 1 0 0]);
```

Use subplots to show the impulse response and the frequency response of the two filters. Note and comment on the in-band ripple of the hh1 filter.

Now pass the time series hh2 through the filter hh1 and plot the response.

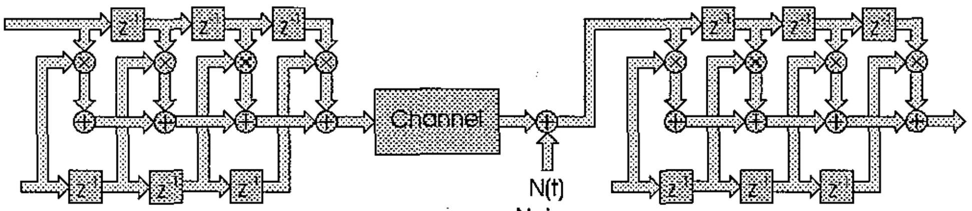
Relate the position, amplitude, and sign of the pre- and postechoes to the frequency, amplitude, and sign of the in-band ripple of filter hh1.

3.21 Repeat Problem 3.20 except change the two filters to match the following:

```
hh1=remez(65,[0 10 11.5 40]/40,[1 1 0 0], [1 100]);
hh2=remez(15,[0 1 10 40]/40,[1 1 0 0]);
```

CHAPTER 4

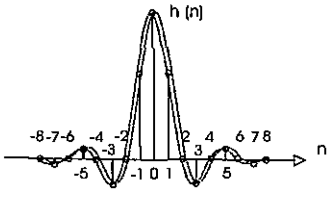
Useful Classes of Filters



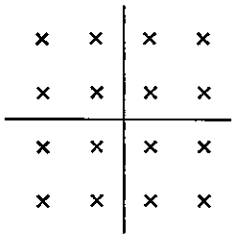
Shaping Filter

Noise

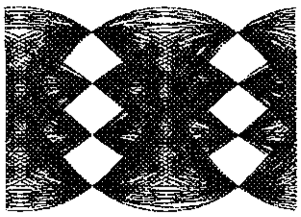
Matched Filter



Nyquist Response



16 QAM Constellation



Eye Diagram

Filters come in all flavors and sizes. We generally describe them with broad stroke coverage such as low pass, high pass, band pass, and the like. We then apply secondary qualifiers such as recursive and nonrecursive or nonlinear phase and linear phase. Certain subclasses of digital filters appear so often in systems that we hardly apply the qualifiers because we know the filter structure by where it resides in the system. One example of such a filter is the ubiquitous filter that shapes the spectrum in modems designed to operate over band-limited channels without intersymbol interference (ISI. This shaping filter is the cosine-tapered square root Nyquist filter compactly described by the term *SQRT-Nyquist filter* or simply the *SQRT* filter. Every book in communication theory describes the properties of the continuous versions of this family of filters while a number of DSP books describe the coefficients of the sampled data version of this same filter. Every designer who has worked on a cable modem or satellite modem has brushed against the same SQRT-Nyquist filter. In this chapter we look very carefully at the digital versions of the SQRT- Nyquist filter from the viewpoint of a filter and then as a system component. Another filter that we see very often in systems is the half-band filter. The half-band appears in many variants, in particular as the Quadrature Mirror filter and the Hilbert transform filter. Both of these are common building blocks in multirate systems and warrant our careful attention and understanding. Another common workhorse, the Cascade Integrator Comb (CIC) appears in its own chapter.

4.1 NYQUIST FILTER AND SQUARE-ROOT NYQUIST FILTER

The Nyquist pulse is the waveshape required to communicate over band-limited channels with no ISI. In many communication systems, the waveform delivered to the receiver's detector is the sum of scaled and offset waveforms as shown in (4.1).

$$s(t) = \sum_{n} d(n)\, h(t-nT) \tag{4.1}$$

The scaling terms d(n) are selected from a small finite alphabet such as $\{-1, +1\}$, or $\{-1, -1/3, +1/3, +1\}$ in accord with a specified mapping scheme between input bits and output levels. The signal s(t) is sampled at equally spaced time increments identified by a timing recovery process in the receiver to obtain output samples as shown in (4.2).

$$s(mT) = \sum_{n} d(n)\, h(mT - nT) \tag{4.2}$$

We can partition this sum as shown in (4.3), to emphasize the desired and the undesired components of the measurement. Here the desired component is d(m) and the undesired component is the remainder of the sum which, if non-zero, is the ISI.

$$s(mT) = d(m)h(0) + \sum_{n \neq m} d(n)h[(m-n)T] \tag{4.3}$$

In order to have zero ISI, the wave shape h(t) must satisfy the specifications identifying sample values at the equally spaced sample increments h(nT) shown in (4.4). This relationship is known as the Nyquist pulse criterion for zero ISI.

$$h(nT) = \begin{cases} 0: n \neq 0 \\ 1: n = 0 \end{cases} \tag{4.4}$$

There are an infinite number of functions that satisfy this set of restrictions. Examples of wave shapes that exhibit equally spaced zeros are shown in Figure 4.1. Note here that one wave shape has duration of exactly one symbol, two of the wave shapes have durations of exactly two symbols, and one wave shape has a duration of four symbols. We restrict the range of possible wave shapes by considering spectral characteristics as well as time domain characteristics. We start by identifying the wave shape with minimum bandwidth. This wave shape is the ubiquitous sin(x)/x in (4.5).

$$h(t) = \frac{\sin(2\pi f \frac{T}{2})}{(2\pi f \frac{T}{2})} \tag{4.5}$$

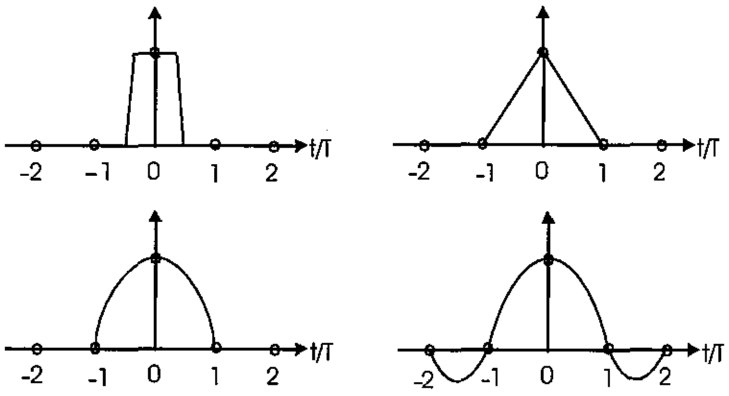


Figure 4.1 Various Wave Shapes Satisfying Condition for Zero ISI

The sin(x)/x wave shape is zero at every sample time, t = nT except for t = 0, and is non-zero elsewhere. This pulse is variously known as the cardinal pulse when used for band limited interpolation and the Nyquist pulse when used in pulse shaping. The transform of this wave shape is the unit area rectangle with spectral support 1/T Hz. A segment of the sin(x)/x waveform and its Fourier transform is shown in Figure 4.2.

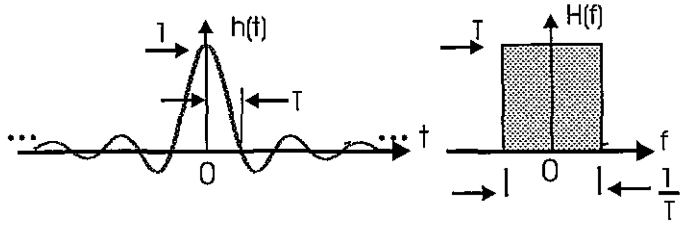


Figure 4.2 Sin(x)/x Waveform and Uniform Pass band Spectrum

The problem with the sin(x)/x waveform is that it is noncausal and further resides on an infinite support. If the pulse resided on a finite support we could delay the response sufficiently for the response to be causal. We have to form finite support approximations to the Nyquist pulse. Our first approximation to this pulse is obtained by convolving the rectangular spectrum H(f) with an even symmetric, continuous spectrum W(f) with finite support α/T. The convolution between H(f) and W(f) in the frequency domain is equivalent to a product in the time domain between the h(t) and w(t), where w(t) is the inverse transform of W(f). The spectral convolution and time product is shown in Figure 4.3 where we see the effect of the spectral convolution is to increase the two-sided bandwidth from 1/T to $(1+\alpha)/T$. The excess bandwidth α/T is the cost we incur to form filters on finite support. The term α is called the roll-off factor and is typically on the order of 0.5 to 0.1 with many systems using values of $\alpha = 0.2$. The transition bandwidth caused by the convolution is seen to exhibit odd symmetry about the half amplitude point of the original rectangular spectrum. This is a desired consequence of requiring even symmetry for the convolving spectral mass function. When the windowed signal is sampled at the symbol rate 1/T Hz, the spectral component residing beyond the 1/T bandwidth folds about the frequency $\pm 1/2T$ into the original bandwidth. This folded spectral component supplies the additional amplitude required to bring the spectrum to the constant amplitude of H(f).

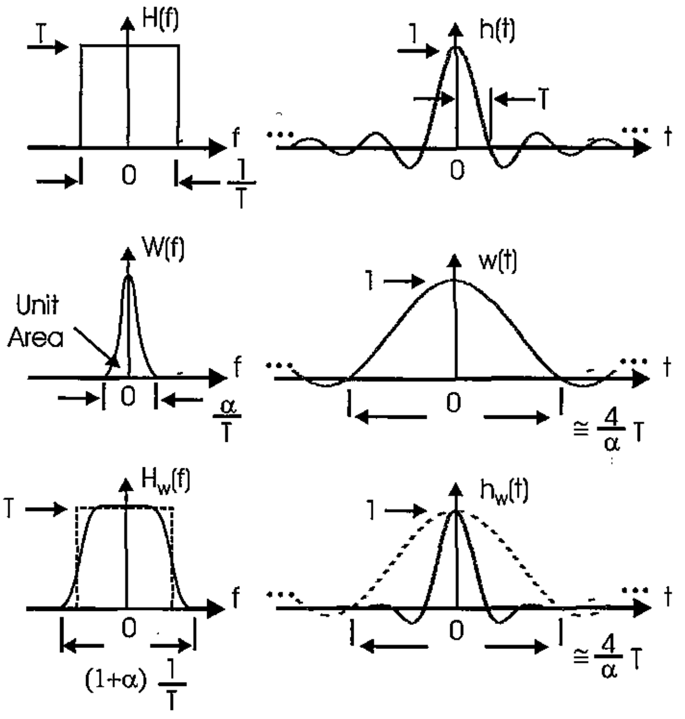


Figure 4.3 Spectral Convolution of Prototype Nyquist Spectrum with Even Symmetric Spectral Mass and Equivalent Time Domain Product of Nyquist Pulse with Window

We also note that the significant amplitude of the windowed wave shape is confined to an interval of approximate width 4T/α so that a filter with α = 0.2 spans approximately 20T, or 20 symbol durations. We can elect to simply truncate the windowed impulse response to obtain a finite support filter, and often choose the truncation points at ± 2T/α. A second window, a rectangle, performs this truncation. The result of this second windowing operation is a second spectral convolution with its transform. This second convolution induces pass band ripple and out-of-band side-lobes in the spectrum of the finite support Nyquist filter. Before performing this truncation, we first address one additional aspect in the design of the Nyquist pulse, which is how the pulse is actually used in a transmitter-receiver pair.

4.2 THE COMMUNICATION PATH

A communication system can be modeled most simply by the signal flow shown in Figure 4.4. Here d(n) represents the sequence of symbol amplitudes presented at symbol rate to the shaping filter $h_1(t)$.

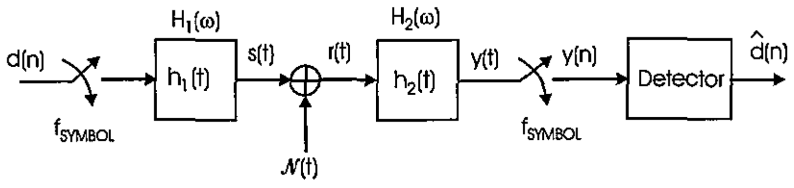


Figure 4.4 Simple Model of Signal Flow in Communication

The superposition of scaled and translated versions of $h_1(t)$ formed by the shaping filter combine to form the transmitter signal s(t) described in (4.6). The channel adds noise to the transmitted signal to form the received signal r(t) as shown in (4.7). The received signal r(t) is processed in the receiver filter $h_2(t)$ to reduce the contribution of the channel noise at y(t), the output of the filter. The receiver filter output is shown in (4.8) which we see is a sum of scaled and translated versions of g(t), the combined impulse response of the transmitter filter and receiver filter plus filtered noise $N_2(t)$. The filter output y(t) is sampled at the symbol rate and time offset to obtain y(n) as shown in (4.9). The sampled output contains three terms, the first proportional to the desired input sample d(m), the second being a sample of filtered noise, and the third containing a weighted sum of earlier and later input samples d(m–n). This last term is the combined ISI due to the memory of the shaping filter $h_1(t)$ and the receiver filter $h_2(t)$.

$$s(t) = \sum_m d(m) h_1(t-mT) \tag{4.6}$$

$$\begin{aligned} r(t) &= s(t) + N_1(t) \\ &= \sum_m d(m)\, h_1(t-mT) + N_1(t) \end{aligned} \tag{4.7}$$

$$\begin{aligned} y(t) &= r(t) * h_2(t) \\ &= \int r(t-\tau) h_2(\tau) d\tau \\ &= \int \sum_m d(m)\, h_1(t - mT - \tau)\, h_2(\tau) d\tau + \int N_1(t-\tau)\, h_2(\tau) d\tau \\ &= \sum_m \mathrm{d(m)\; g(t\text{-}mT)} + \mathrm{N}_2(t) \end{aligned} \tag{4.8}$$

where $\mathrm{g(t)} = \int \mathrm{h}_1(t-\tau)\, h_2(\tau)\, d\tau$, and $N_2(t) = \int N_1(t-\tau)\, h_2(\tau) d\tau$

$$\begin{aligned} y(nT) &= \sum_m d(m) g[(n-m)T] + N_2(nT) \\ &= d(n)g(0) + N_2(nT) + \sum_{m \neq n} d(m) g[(n-m)T] \end{aligned} \tag{4.9}$$

We can obtain from the terms in (4.9) an unbiased estimate of d(n) if g(0) is 1, and zero ISI if g([n-m]T) is zero for all m not equal to n. This is the same requirement presented in (4.4) as the requirement for a zero ISI filter. Thus the convolution of the shaping filter at the transmitter and the noise control filter at the receiver filter must form the Nyquist filter as shown in (4.10).

$$\begin{aligned} h_1(t) * h_2(t) &= h_{NYQ}(t - T_D) \\ H_1(\omega) H_2(\omega) &= H_{NYQ}(\omega)\, e^{-j\omega T_D} \end{aligned} \tag{4.10}$$

To maximize the signal-to-noise (SNR) in (4.10), the receiver filter must be matched to the transmitter-shaping filter. The matched filter is a time-reversed and delayed version of the shaping filter, which is described in the frequency domain as shown in (4.11).

$$H_2(\omega) = H_1^*(\omega)\, e^{-j\omega T_D} \tag{4.11}$$

Combining the requirements in (4.10) and (4.11) we obtain the result in (4.12) from which we determine the relationship between the shaping filter and the desired Nyquist filter response shown in (4.13). The shaping filter is called a SQRT-Nyquist filter.

$$H_1(\omega) H_1^*(\omega)\, e^{-j\omega T_D} = H_{NYQ}(\omega)\, e^{-j\omega T_D} \tag{4.12}$$

$$\begin{aligned} |\, H_1(\omega) \,|^2 &= H_{NYQ}(\omega) \\ H_1(\omega) &= \sqrt{H_{NYQ}(\omega)} = SQRT[H_{NYQ}(\omega)] \end{aligned} \tag{4.13}$$

Reviewing the result of this last derivation we expect the shaping filter and the receiver filter to accomplish two tasks. The two filters in cascade form a Nyquist filter and further interact to maximize SNR. We say that the SQRT-Nyquist filter performs half the spectral shaping at the transmitter and half the spectral shaping at the receiver.

We now examine the frequency response and the time domain response of the SQRT-Nyquist filter. The square-root operation applied to the magnitude spectrum of the Nyquist filter does not affect the zero-valued nor the unit-valued segments of the filter response. Thus the square root affects only the spectrum in the transition bandwidth. The Nyquist filter has a gain of 0.5 or –6.0-dB at the nominal band edge while the square-root filter has a gain of 0.707 or –3.0-dB at the same frequency. The shaping filter and the matched filter each applies 3.0-dB attenuation at the band edge to obtain the desired band-edge attenuation of 6.0-dB. Hence the –6.0-dB bandwidth of the SQRT filter is wider than the –6.0-dB bandwidth of the Nyquist filter and the wider bandwidth square-root Nyquist filter must have a narrower main lobe impulse response than the Nyquist filter. These relationships are illustrated in Figure 4.5.

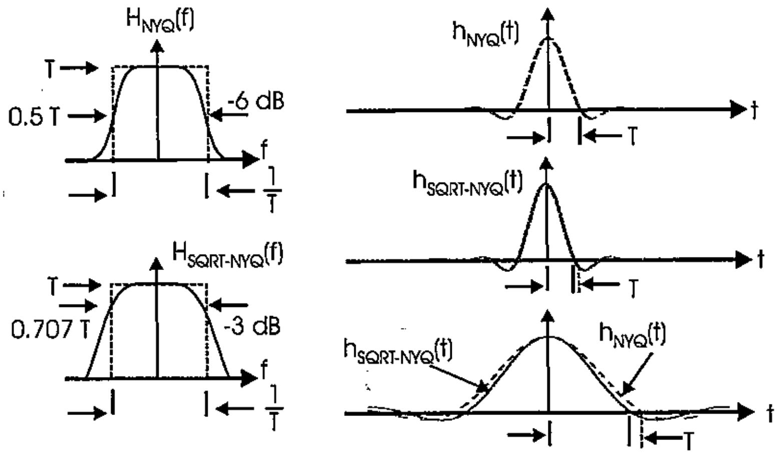


Figure 4.5 Spectrum and Time Response of the Nyquist Filter and SQRT-Nyquist Filter

We now address the taper in the excess bandwidth Nyquist pulse. As mentioned earlier, any even symmetric spectral mass can be used to perform the spectral convolution that forms the transition bandwidth of the tapered Nyquist spectrum. The most common spectral mass selected for communication systems is the half cosine of width $\alpha \cdot f_{SYM}$. The half cosine convolved with the spectral rectangle forms the spectrum known as the cosine-tapered Nyquist pulse with rolloff α. The description of this band-limited spectrum normalized to unity pass band gain is presented in (4.14).

$$H_{NYQ}(\omega)=\begin{cases} 1 & : \text{for}\dfrac{|\omega|}{\omega_{SYM}}\le(1-\alpha) \\ 0.5*\left\{1+\cos\{\dfrac{\pi}{2\alpha}[\dfrac{\omega}{\omega_{SYM}}-(1-\alpha)]\}\right\} & : \text{for } (1-\alpha)\le\dfrac{|\omega|}{\omega_{SYM}}\le(1+\alpha) \\ 0 & : \text{for}\dfrac{|\omega|}{\omega_{SYM}}\ge(1+\alpha) \end{cases} \tag{4.14}$$

The continuous time domain expression for the cosine-tapered Nyquist filter is shown in (4.15). Here we see the windowing operation of the Nyquist pulse as a product with the window that is the transform of the half-cosine spectrum.

$$h_{NYQ}(t)=f_{SYM}\frac{\sin(\pi f_{SYM}\,t)}{(\pi f_{SYM}\,t)}\frac{\cos(\pi\,\alpha f_{SYM}\,t)}{[1-(2\alpha f_{SYM}\,t)^2]} \tag{4.15}$$

4.3 The Sampled Cosine Taper

Since the Nyquist filter is band limited, we can form the samples of a digital filter by sampling the impulse response of the continuous filter. Normally this involves two operations. The first is a scaling factor applied to the impulse response by dividing by the sample rate, and the second is the sampling process in which we replace t with $n{\cdot}T_{SMPL}$ or n/f_{SMPL}. The sample rate must exceed the two-sided bandwidth of the filter that, due to the excess bandwidth, is wider than the symbol rate. It is standard to select the sample rate f_{SMPL} to be an integer multiple of the symbol rate f_{SYM} so that the filter operates at M-samples per symbol. It is common to operate the filter at 4 or 8 samples per symbol but for generality we select $f_{SMPL} = M\ f_{SYM}$ so that f_{SYM} t is replaced by $f_{SYM}\ n/(M{\cdot}f_{SYM})$ or n/M. After applying these operations to (4.15) we obtain the results shown in (4.16).

$$h_{NYQ}(n)=\frac{1}{M}\frac{\sin(\pi\,n/M)}{(\pi\,n/M)}\frac{\cos(\alpha\pi\,n/M)}{[1-(2\alpha n/M)^2]} \tag{4.16}$$

The filter described in (4.16) has a two-sided bandwidth that is approximately 1/Mth of the sample rate. A digital filter exhibits a processing gain proportional to the ratio of input sample rate to output bandwidth, in this case a factor of M. The 1/M scale factor in (4.16) cancels this processing gain to obtain unity gain. When the filter is used for shaping and up sampling, as it is at the transmitter, we remove the 1/M scale factor since we want the impulse response to have unity peak value rather than unity processing gain.

The square root of the cosine-tapered Nyquist filter results in a quarter cycle cosine-tapered filter. This description is normally contracted to square-root raised cosine or root raised cosine Nyquist filter. The description of this band-limited spectrum normalized to unity pass band gain is shown in (4.17).

$$H_{SQRT-NYQ}(\omega) = \begin{cases} 1 & : for \frac{|\omega|}{\omega_{SYM}} \leq (1-\alpha) \\ \cos\left\{\frac{\pi}{4\alpha}[\frac{\omega}{\omega_{SYM}} - (1-\alpha)]\right\} & : for (1-\alpha) \leq \frac{|\omega|}{\omega_{SYM}} \leq (1+\alpha) \\ 0 & : for \frac{|\omega|}{\omega_{SYM}} \geq (1+\alpha) \end{cases} \tag{4.17}$$

The continuous time domain expression for the square-root raised cosine Nyquist filter is shown in (4.18).

$$h_{SQRT-NYQ}(t) = f_{SYM} \frac{(4\alpha f_{SYM}\ t)\cos[\pi(1+\alpha) f_{SYM}\ t] + \sin[\pi(1+\alpha) f_{SYM}\ t]}{[1-(4\alpha f_{SYM}\ t)^2](\pi f_{SYM}\ t)} \tag{4.18}$$

We perform the same scaling and sampling operation we performed in (4.18) to obtain the sampled data version of the square-root raised cosine Nyquist pulse shown in (4.19).

$$h_{SQRT-NYQ}(n) = \frac{1}{M}\left[\frac{4\alpha \frac{n}{M} n \cos[\frac{\pi\, n\, (1+\alpha)}{M}] + \sin[\frac{\pi\, n\, (1+\alpha)}{M}]}{[1-(4\alpha \frac{n}{M})^2]\ \pi \frac{n}{M}}\right] \tag{4.19}$$

When the impulse response is used as an up-sampler and shaping filter at the transmitter, (4.19) must be rescaled. For this application we want unity peak impulse response rather than unity processing gain. Multiplying by the term shown in (4.20) scales the coefficients of (4.19).

$$\text{Transmitter Scale Factor:} \quad \frac{\text{M}}{1+(\frac{4}{\pi}-1)\alpha} \tag{4.20}$$

When the impulse response is used in a matched filter at the receiver, (4.19) must be rescaled again to account for the scaling applied at the transmitter. Multiplying by the term shown in (4.21) scales the coefficients of (4.19).

$$\text{Receiver Scale Factor:} \quad 1+(\frac{4}{\pi}-1)\alpha \tag{4.21}$$

4.3.1 Root-raised Cosine Side-lobe Levels

We commented earlier that when we implement the SQRT-Nyquist filter, we actually apply two windows; the first window is a smooth continuous function used to control the transition bandwidth and the second is a rectangle used to limit the impulse response to a finite duration. This second window forces side-lobes in the spectrum of the SQRT-Nyquist filter. These side-lobes are quite high, on the order of 24 to 46-dB below the pass-band gain depending on roll-off factor and the length of the filter in number of symbols. The reason for the poor side-lobe response is the discontinuous first derivative at the boundary between the half-cosine transition edge and the start of the stop band. Consequently the envelope of the time function falls off, as seen in (4.18), as $1/t^2$ enabling a significant time discontinuity when the rectangle window is applied to the filter impulse response. In retrospect, the cosine-tapered Nyquist pulse was a poor choice for the shaping and matched filter in communication systems.

Figure 4.6 presents measured levels of side-lobe levels for a range of roll-off factors as a function of filter length in number of symbol. We see that side-lobe levels fall very slowly with increased filter length and increase with reduced transition bandwidth. These levels of attenuation will not meet realistic spectral mask requirements for out-of-band attenuation that are typically on the order of 60 to 80-dB. Some mechanism must be invoked to control the filter out-of-band side-lobe levels related to the rectangle window. Whatever process is invoked should preserve the ISI levels obtained by convolving the fixed-length SQRT-Nyquist filter with itself.

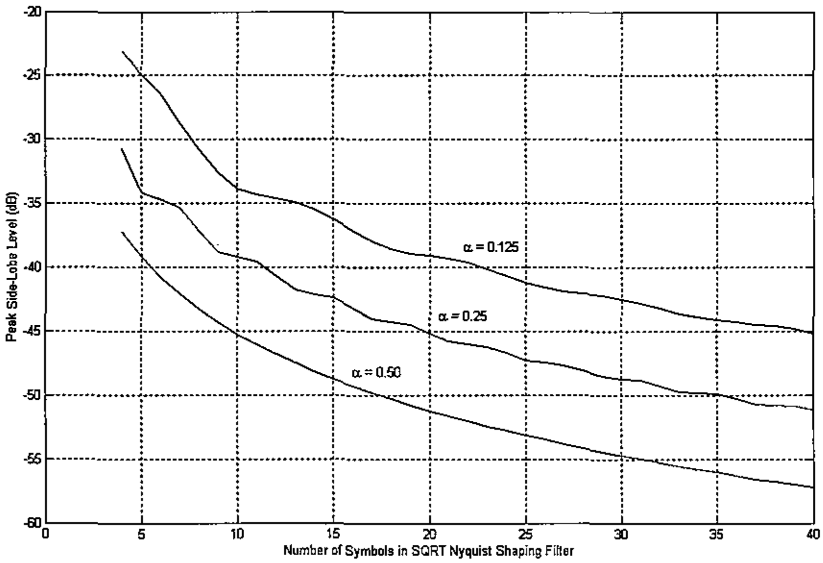


Figure 4.6 Highest Out-of-band Side-lobe Levels for SQRT Raised Cosine Nyquist Filter for a Range of Roll-off Bandwidths, and Filter Lengths

Attempting to control the spectral side-lobes by applying a second window, to the already windowed, prototype filter leads to significant increases in the receiver ISI levels. This is shown in Figure 4.7, which illustrates the effect on spectral side-lobes and ISI levels as a result of applying windows to the prototype impulse response. The increase in ISI is traced to the shift of the filter's 3-dB point away from the nominal band edge. The requirement for zero ISI at the output of the matched filter requires that the shaping and matched filters each exhibit 3-dB attenuation at the filter band edge, half the symbol rate. A design technique must control side-lobe levels while maintaining the 3-dB frequency at the symbol band edge.

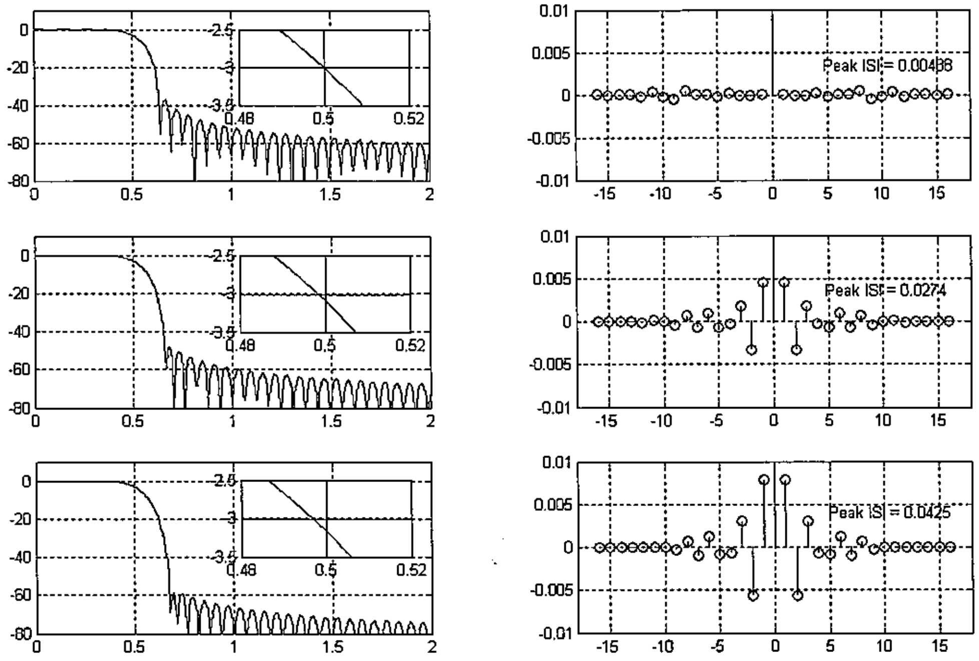


Figure 4.7 Spectrum of Windowed SQRT-Raised Cosine Nyquist Filter, Kaiser(N,0) –37-dB), Kaiser(N,2) (–50-dB), and Kaiser(N,3) (–60-dB), Inserted Details of Spectrum near 3-dB Bandwidth and Detail of ISI in Matched Filter Outputs

4.3.2 Improving the Stop band Attenuation

The important attributes of the SQRT-Nyquist spectrum are the transition bandwidth or roll-off defined by α and the 3-dB attenuation at the band edge. A very simple iterative algorithm based on the Remez algorithm will transform an initial low pass filter to a SQRT-Nyquist spectrum with the specified roll-off while preserving the ability to independently control pass band ripple and stop band ripple. Figure 4.8 illustrates the form of the algorithm by starting the Remez algorithm with pass band and stop band edges matched to the roll-off boundaries of the Nyquist spectrum. The resulting filter will cross the band edge (f/f_{SYM} = 0.5) with more attenuation than the desired -3-dB level. We can raise the attenuation level toward the desired -3-dB level by increasing the frequency of the pass band edge. The algorithm performs the successive shifts to the right of the pass band edge (frequency f_1) until the error between the desired 0.707 and the $H_1(0.5)$ level is reduced to zero by using the gradient descent method shown in (4.22).

$$\begin{aligned} error(n) &= \sqrt{2}/2 - abs(H_1(0.5)\,|_n) \\ f_1(n+1) &= f_1(n)\cdot(1+\mu\cdot error(n)) \end{aligned} \qquad (4.22)$$

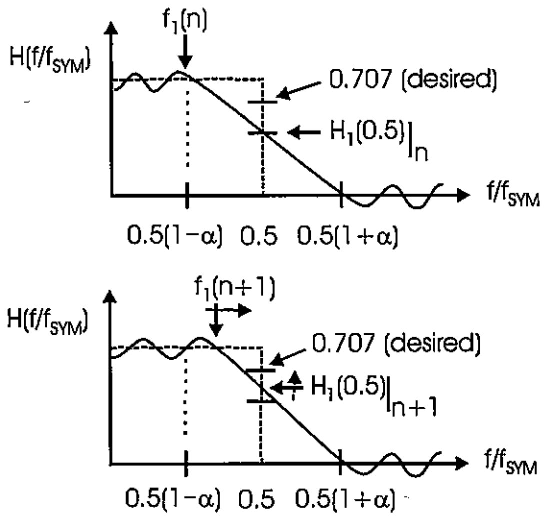


Figure 4.8 Spectrum of Remez Algorithm Output at Iteration n Matching Band Edges of Nyquist Pulse and then Increase of Frequency f_1 for Iteration n+1 to Raise $h_1(n)$ to $h_1(n+1)$ and Eventually to Desired -3-dB Level

A MATLAB script file named *nyq2*, available from the book's companion website listed on the title page, implements the algorithm just described. Figure 4.9 presents the spectra of the filter response at its first and final iteration as the initial –6-dB gain is raised to –3-dB gain at the normalized frequency band edge of 0.5. Also seen here is the detail of the filter –3-dB gain at frequency 0.5. Figure 4.10 presents a comparison of the frequency response of a SQRT-Nyquist filter designed by the MATLAB *rcosine* script file and designed by the *nyq2* script file. The *nyq2* design is designated as the *3-dB harris filter*. In both cases, the filter sample rate is 5 times the symbol rate with roll-off factor α equal to 0.25. The filters are both of length 20 symbols, the length suggested by *nyq2* in response to the input specifications. The first item of interest in the *nyq2* design is the side-lobe level obtained by the design. The highest side-lobe level is approximately –72-dB as opposed to the –40-dB of the *rcosine* design. The second feature of note is the reduced level of in-band ripple of the *nyq2* design relative to the *rcosine* design. Note that by permitting the transition bandwidth to differ from the cosine taper of the standard SQRT-Nyquist pulse we obtain significant improvement in filter characteristics. The resulting pulse is still a SQRT-Nyquist pulse but the transition is no longer cosine tapered.

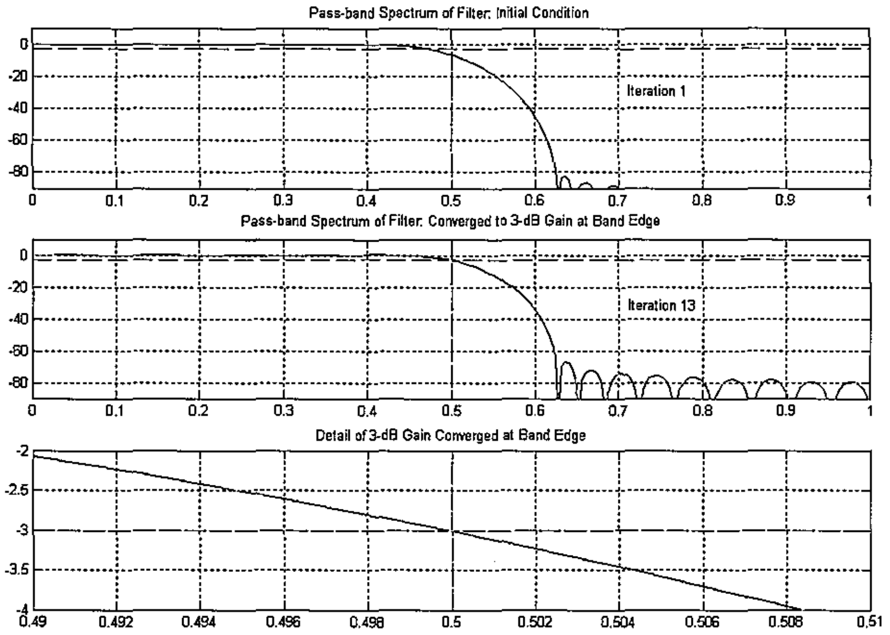


Figure 4.9 Spectra of SQRT-Nyquist Filter at First and Last Iteration Plus Detail of Spectra Near –3-dB Gain

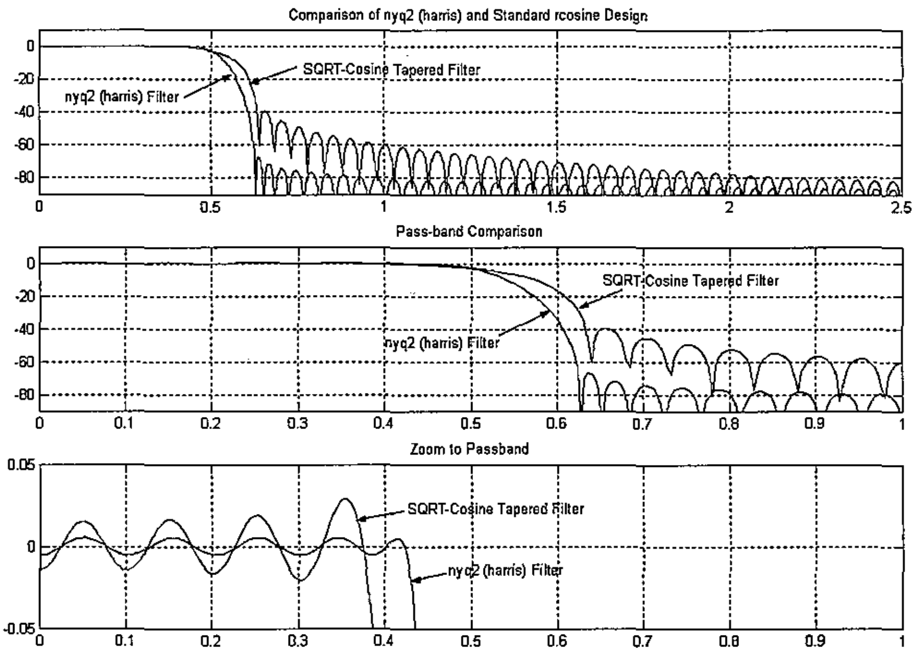


Figure 4.10 Spectra of Converged and of SQRT-Cosine Filters, Detail of Same Spectra, and Finer Detail of Same Spectra

Figure 4.11 presents the details of the matched filter output time series after resampling to symbol rate. The three subfigures show the result of convolving the *nyq2* filter with itself, the result of convolving the *rcosine* filter with itself, and the result of convolving the *nyq2* with the *rcosine* filter. Each subplot indicates the maximum ISI level as well as the RMS ISI level. The values are repeated in Table 4-1.

Table 4-1. Peak and RMS ISI Levels for Three Operating Conditions for Shaping and Matched Filters: Remez and Remez, SQRT and SQRT, and Remez and SQRT

Operating Condition	PEAK ISI	RMS ISI
nyq2 Filter * *nyq2* Filter	0.00589	0.0012
rcosine Filter * *rcosine* Filter	0.0136	0.0048
nyq2 Filter * *rcosine* Filter	0.0293	0.0074

Note that the filter designed by the *nyq2* algorithm exhibits smaller ISI levels than the *rcosine* filter of the same length. Thus we pay no ISI penalty for using the *nyq2* Remez-based design that was initially attractive for its low out-of-band side-lobe levels. We also addressed the possibility of designing the shaping filter with the *nyq2* algorithm but using an *rcosine* filter for a matched filter to determine the effect of the mismatch. This is the third operating condition in the table and in Figure 4.11.

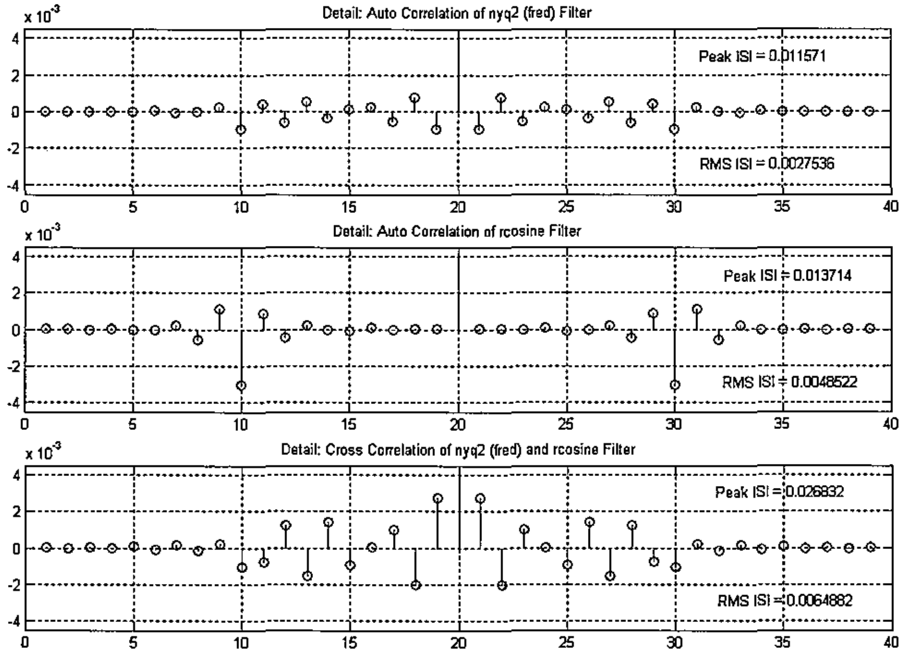


Figure 4.11 Details of Matched Filter Outputs. *nyq2* Filter Convolved with *nyq2* Filter, *rcosine* Filter Convolved with *rcosine* Filter, and *nyq2* Filter Convolved with *rcosine* Filter

The mismatch between the two filters increases the Peak ISI by a factor of 5 from 0.6% to 2.9% and the RMS ISI by a factor of 6 from 0.12% to 0.74%. As a reference 1% ISI is approximately a 0.12-dB implementation loss if uncorrected. Most digital receiver systems include an equalizer that would strip the ISI from the matched filter output prior to the detection process. Thus the *nyq2* filter can be used for a shaping filter to access the improved out-of-band attenuation without penalty when received by a matched filter or with a mismatched approximate matched filter and a channel equalizer.

Figure 4.12 presents the linear spectra obtained from the impulse responses of the two filters we have been examining. As expected, both filters pass through the 0.707 gain point at their band edge. It appears that the *nyq2* spectrum has a steeper slope and perhaps, a reduced roll-off factor. In fact the two filters have the same roll off of 0.25 but the *nyq2* has a smoother transition, which is the standard practice required to achieve side-lobes with deeper out-of-band attenuation. The square of two spectra is the composite spectra that should be the Nyquist spectra with odd symmetric transition through the gain of 0.5 at the filter band edge. This appears to be what we see. The third subfigure is the frequency derivative of the two transition bandwidths. These should be unit-area, even symmetric spectral masses that when convolved with the rectangle spectrum forms the transition bandwidth of the Nyquist filter. For the ideal Nyquist pulse, this spectral mass should be a half cosine.

Surprisingly, the spectral mass obtained from the *rcosine* filter is not symmetric. This is due to the high side-lobes added to the spectrum by truncating the impulse response with the rectangle window. The spectral mass of the *nyq2* filter does not suffer this asymmetry. We would expect some of the ISI formed in the *rcosine* filter to be related to this asymmetry and poor approximation to an odd-symmetric transition.

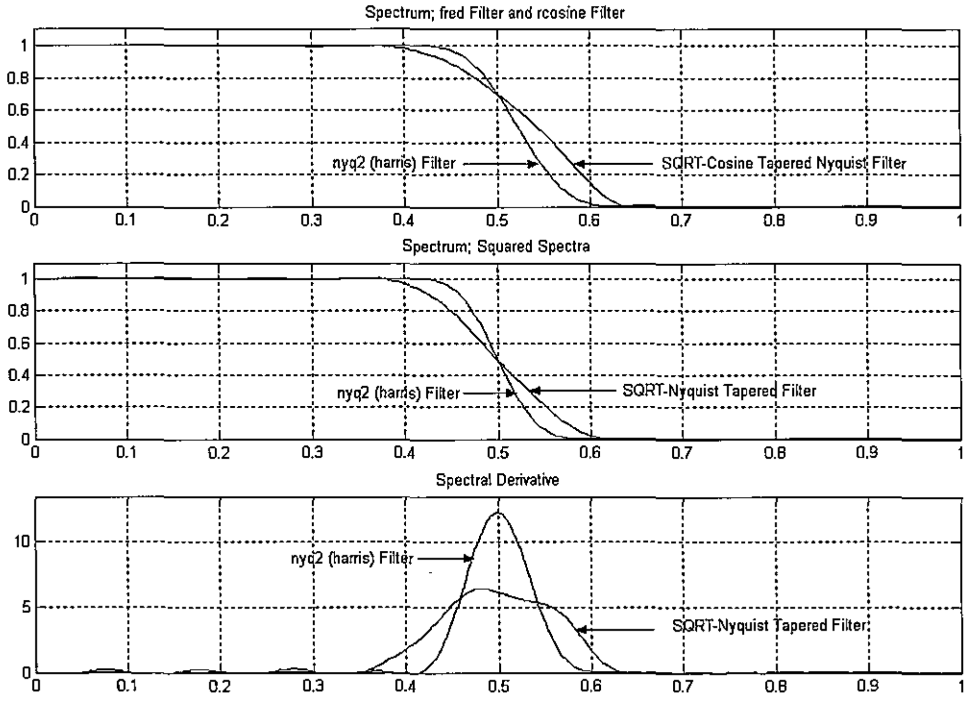


Figure 4.12 Spectra of *nyq2* and *rcosine* SQRT-Cosine Filters, Squared Spectra of Both Filters, and Derivative of Squared Spectra

4.4 HALF-BAND FILTERS

A half-band filter has a particularly attractive property that makes it uniquely desirable for use in multirate filters. We now identify that attribute. The frequency response and the impulse response of a zero-phase, hence nonrealizable, half-band filter is shown in Figure 4.13.

We note in a normalized frequency axis, the pass band is the interval between the plus and minus quarter sample rate. The equation describing the impulse response is shown in (4.23).

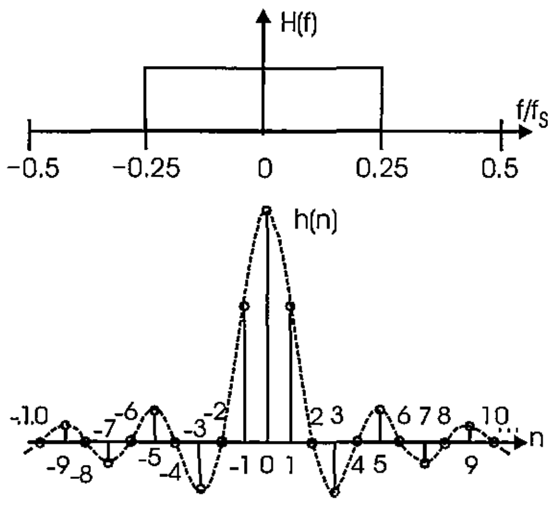


Figure 4.13. Impulse Response and Spectrum of Ideal Low pass Half-band Filter

$$h(n) = \frac{1}{2}\frac{\sin(n\pi/2)}{(n\pi/2)} \tag{4.23}$$

The first thing we note about the impulse response is that, except for the sample value at the origin, all the even indexed sample values are zero. It is this property that makes the half-band filter interesting to us. The impulse response of the filter shown in (4.23) extends over all integers and our first modification to this filter is to apply a window as shown in (4.24), to make it finite duration. The window, of course causes, a ripple in the pass band and in the stop band of the filter as well as induces an odd symmetric transition bandwidth that passes through the quarter sample rate with a gain of 0.5, the midpoint of the gain discontinuity. To control the amplitude of the in-band and out-of-band ripple we apply a smooth window such as a Hann or Kaiser window. The time-limited version of the impulse response and its related finite transition bandwidth spectrum is shown in Figure 4.14.

$$h_{LOW}(n) = h(n)\cdot w(n) \tag{4.24}$$

The windowed impulse response continues to exhibit zero amplitude at all even indexed data samples as well as amplitude 0.5 for the data sample at index zero. The window only affects the odd indexed filter coefficients.

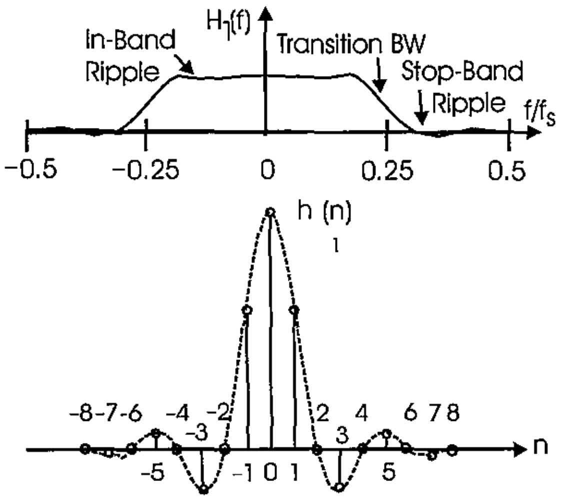


Figure 4.14 Impulse Response and Spectrum of Finite Duration Low pass Half-band Filter

The half-band low pass can be converted to a half-band high pass filter by using the modulation theorem to translate the spectral center from DC, $\theta = 0$, to the half sample rate, $\theta = \pi$. This is shown in (4.25). The impulse response and the frequency response of the translated filter are shown in Figure 4.15.

$$h_{HIGH}(n) = h_{LOW}(n) \cdot \cos(\pi n) \tag{4.25}$$

We note that the heterodyning cosine is alternately +1 and –1 and that the +1 occurs on the even indices and the –1 occurs on the odd indices. We already noted that except for the zero index, all the non-zero coefficients of the low pass impulse response occur on the odd indices. Thus the product formed in (4.25) simply changes the sign of all the coefficients except the coefficient at index zero. As a result of sign changes just described, we can relate the even-indexed and odd-indexed coefficients of the two filters $h_{LO}(n)$ and $h_{HI}(n)$ as shown in (4.26).

$$\begin{aligned} h_{HIGH}(2n) &= h_{LOW}(2n) \\ h_{HIGH}(2n+1) &= -h_{LOW}(2n+1) \end{aligned} \tag{4.26}$$

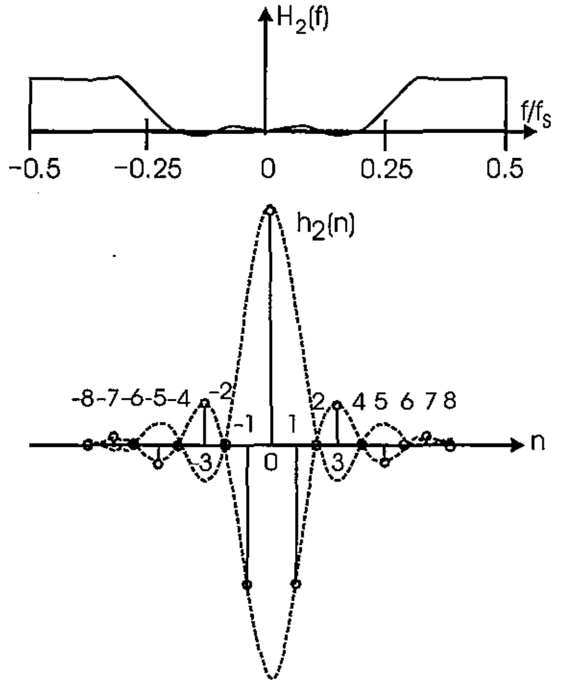


Figure 4.15 Impulse Response and Spectrum of FIR High pass Half-band Filter

It is convenient to identify the Z-transform of the even-indexed coefficients and the Z-transform of the odd-indexed coefficients of the low pass filter. This is shown in (4.27). The two units of delay between successive samples in the two sequences are accounted for in their Z-transforms by the Z^2 in the argument of the transform.

$$
\begin{aligned}
H_0(Z^2) &= \sum_n h_{LOW}(2n)\, Z^{-2n} \\
Z^{-1}H_1(Z^2) &= \sum_n h_{LOW}(2n+1)\, Z^{-(2n+1)}
\end{aligned}
\tag{4.27}
$$

Using the relationships identified in (4.27) and the sign reversals described in (4.26) we can form the transform of the low pass and of the high pass filters as a weighted sum of the sub transforms as shown in (4.28).

$$
\begin{aligned}
H_{LOW}(Z) &= H_0(Z^2) + Z^{-1}H_1(Z^2) \\
H_{HIGH}(Z) &= H_0(Z^2) - Z^{-1}H_1(Z^2)
\end{aligned}
\tag{4.28}
$$

The block diagram of the filter structure suggested in (4.28) is shown in Figure 4.16. This form of filter is known as a *polyphase partition* of the prototype filter, and since it offers both the low pass and the high pass version of the half-band filter, it is also known as a *quadrature mirror filter*.

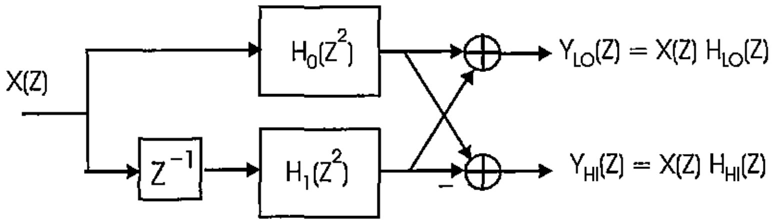


Figure 4.16 Polyphase Partition of Half-band Filter Pair

If we form the sum of the two transforms identified in (4.28) we obtain twice the transform of the even-indexed coefficients. But the even-indexed coefficients have only a single non-zero term, which is the index zero term. The resultant transform can be seen in (4.29) to be zero everywhere except at index zero.

$$\begin{aligned} H_{LOW}(Z) + H_{HIGH}(Z) &= 2H_0(Z) \\ &= 2h(0) = 1 \end{aligned} \tag{4.29}$$

Equation (4.29) can be rearranged to form (4.30), which demonstrates that $H_{LOW}(Z)$, the low pass filter, and $H_{HIGH}(Z)$, the high pass filter are complementary.

$$H_{LOW}(Z) = 1 - H_{HIGH}(Z) \tag{4.30}$$

An alternate representation of the quadrature mirror filter pair presented in Figure 4.16 is shown in Figure 4.17 as a complementary filter pair.

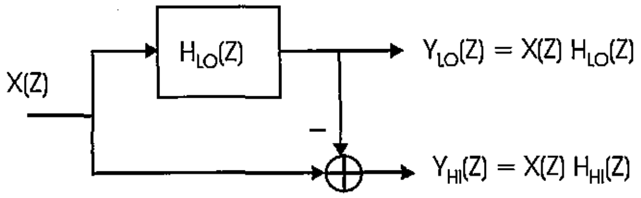


Figure 4.17 Complementary Partition of Nonrealizable Half-band Filter Pair

The filter presented in Figures 4.14 and 4.15 are noncausal. To make them causal a delay of length (N-1)/2 must be inserted in the time response so that the impulse response values for negative time indices are exactly zero. The redefinition of the origin for the impulse response of the half-band filter is shown in Figure 4.18. When the delay is inserted in the filter path to make it realizable, a matching delay has to be inserted in the second path containing the direct input-to-output connection. This change is shown in Figure 4.19.

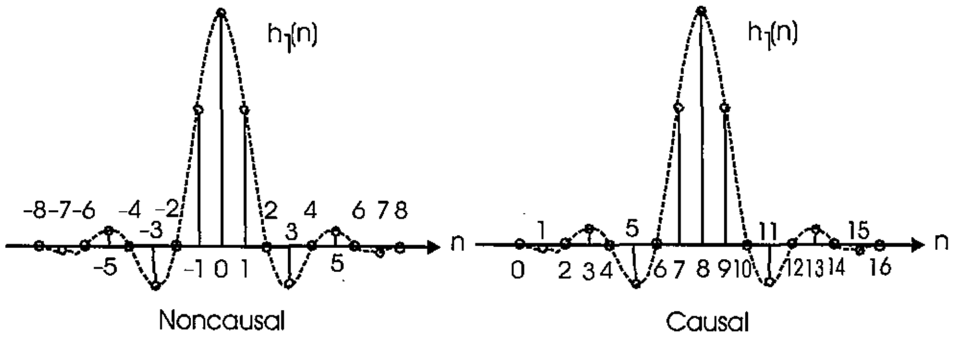


Figure 4.18 Noncausal and Causal Forms of Half-band Low pass Filter

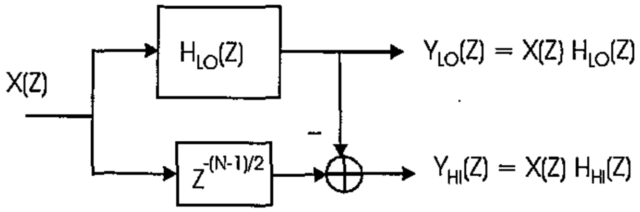


Figure 4.19 Complementary Partitions of Realizable Half-band Filter Pair

References

Anderson, John, *Digital Transmission Engineering*, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1999.

Crochiere, Ronald and Lawrence Rabiner. *Multirate Signal Processing*, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1983.

Fliege, Norbert, Multirate *Digital Signal Processing: Multirate Systems, Filter Banks, Wavelets*, West Sussex, John Wiley & Sons, Ltd., 1994.

Harada, Hiroshi and Ramjee, Prasad, *Simulation and Software Radio for Mobile Communications*, Norwood, MA, Artech House, 2002.

Hentschel, Tim, *Sample Rate Conversion in Software Configurable Radios*, Norwood, MA, Artech House, 2002.

Jovanovic-Dolecek, Gordana, *Multirate Systems: Design and Applications*, London, Idea Group, 2002.

Mitra, Sanjit and Kaiser, James, *Handbook for Digital Signal Processing*, New York, John Wiley & Sons, 1993.

Proakis, John, *Digital Communications*, 4th ed., New York, McGraw-Hill, 2001.

Problems

4.1 The following sequences all are time limited to an interval that permits modulation with independent peak samples every 10 output samples.

```
h1 = rectpuls([–0.5:0.1:0.5]);
h2 = tripuls([–0.5:0.1:0.5]);
h3 = cos(pi*(–0.5:0.1:0.5));
h4 = cos(pi*(–0.5:0.1:0.5)).^2;
h5 = exp(-20*(–0.5:0.1:0.5).^2);
```

On a set of subplots plot the time response and the frequency response of each wave shape.

Comment on their respective bandwidths, side-lobe levels, and relative smoothness of the time series.

4.2 The following sequences all are time limited to an interval that permits modulation with overlapped but independent peak samples every twenty output samples.

```
h1 = tripuls([–1.0:0.1:1.0]);
h2 = cos(pi*(–1.0:0.1:1.0));
h3 = cos(pi*(–1.0:0.1:1.0)).^2;
h4 = exp(-5*(–1.0:0.1:1.0).^2);
h5 = sinc(–1.0:0.1:0.1);
```

On a set of subplots plot the time response and the frequency response of each wave shape.

Comment on their respective bandwidths, side-lobe levels, and relative smoothness of the time series.

4.3 Use the MATLAB script file *rcosine* to generate samples of the root-raised cosine Nyquist filter. The following MATLAB script will accomplish this;

```
hh1 = rcosine(1,8,'sqrt',0.5,6);
```

Convolve the sequence hh1 with itself and plot the two sequences hh1/max(hh1) and conv(hh1,hh1) and their spectra. Comment on the locations of the zero crossings of the two time series hh1/max(hh1) and conv(hh1,hh1). Comment on the amplitude response of the two spectra, particularly at the band edges, ±1/16 of the sample rate.

4.4 Use the MATLAB script file *rcosine* to generate samples of the root-raised cosine Nyquist filter and then generate a time series or random modulated data and pass it through the receiver matched filter. Finally form eye diagrams for the input and output of the matched filter. The following MATLAB script will accomplish this;

```
hh1 = rcosine(1,8,'sqrt',0.25);
dat1 = (floor(4*rand(1,1000))–1.5)/1.5;
dat2 = reshape([dat1; zeros(7,1000)],1,8000)
dat3 = conv(dat2,hh1)/max(hh1);
dat4 = conv(dat3,hh1)
plot(0,0);
hold on
for nn=1:16:8000-16
plot(–1:1/8:1,dat3(nn:nn+16))
end
hold off
grid
% Repeat for dat4
```

Examine and comment on the two eye diagrams paying particular attention to the width and height of the eye opening.

4.5 Repeat Problem 4.4 except change the excess bandwidth parameter from 0.25 to 0.125. The following MATLAB script will accomplish this:

```
hh1 = rcosine(1,8,'sqrt',0.125);
```

4.6 Repeat Problem 4.4 except change the excess bandwidth parameter from 0.25 to 0.50. The following MATLAB script will accomplish this:

```
hh1 = rcosine(1,8,'sqrt',0.50);
```

4.7 Design a half-band low pass FIR by windowing a sinc series filter with transition bandwidth 10% of the sample rate and with in-band and out-of-band ripple less than 0.001. Plot the impulse response and the frequency response. Plot the frequency response with linear and with log magnitude coordinates.

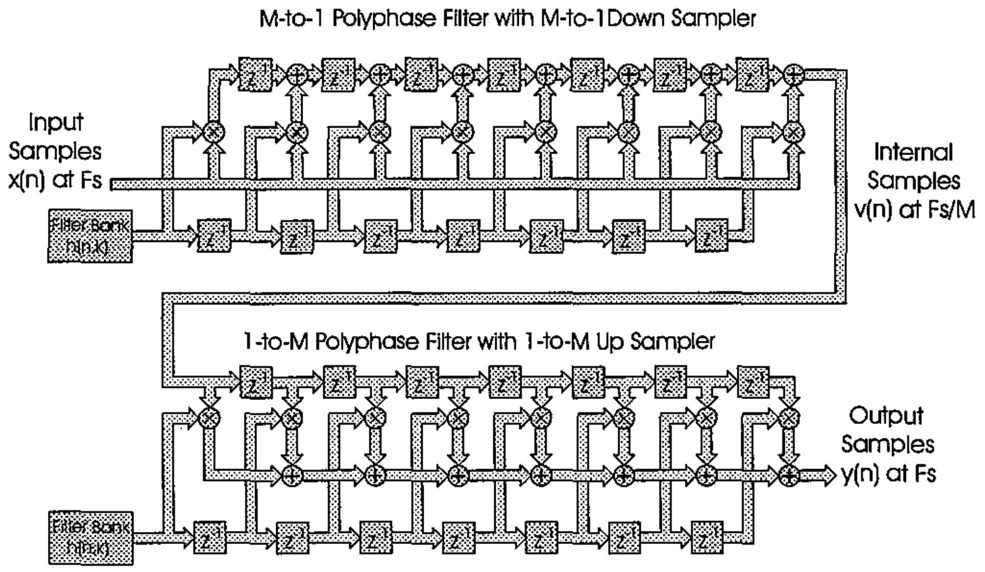
4.8 Design a half-band high pass FIR by windowing a sinc series filter with transition bandwidth 10% of the sample rate and with in-band and out-of-band ripple less than 0.001. The high pass is formed from the low pass as a heterodyned filter or as a complementary filter. Try both! Plot the impulse response and the frequency response. Plot the frequency response with linear and with log magnitude coordinates.

4.9 Design a half-band low pass FIR with the Remez algorithm. The filter has a transition bandwidth 10% of the sample rate and with in-band and out-of-band ripple less than 0.001. Plot the impulse response and the frequency response. Plot the frequency response with linear and with log magnitude coordinates.

4.10 Design a half-band high pass FIR with the Remez algorithm. The filter has a transition bandwidth 10% of the sample rate and with in-band and out-of-band ripple less than 0.001. The high pass is formed from the low pass as a heterodyned filter, as a complementary filter, or by changing the gain vector in the Remez algorithm. Try all three. Are they the same? Plot the impulse response and the frequency response. Plot the frequency response with linear and with log magnitude coordinates.

CHAPTER 5

Systems That Use Resampling Filters


M-to-1 Polyphase Filter with M-to-1 Down Sampler
Input Samples x(n) at Fs
Filter Bank h(n,k)
Internal Samples v(n) at Fs/M
1-to-M Polyphase Filter with 1-to-M Up Sampler
Output Samples y(n) at Fs
Filter Bank h(n,k)

There are many applications for multirate filters that, when invoked, lead to reduced cost to implement the desired processing task. A common theme in many of these applications is that the filtering should occur at a rate matching the Nyquist rate. The next two examples illustrate this concept.

5.1 Filtering with Large Ratio of Sample Rate to Bandwidth

A common application for multirate signal processing is the task of filtering to reduce the signal bandwidth without changing the sample rate. This might occur when the output of the filter is to be presented to a DAC operating at a fixed output rate matching the input rate. Let us examine the specifications of a filter with a small bandwidth relative to sample rate. The specifications required for the filter are listed in Table 5-1 and are illustrated in Figure 5.1.

Table 5-1 Filter Specifications

Parameter	Specification
Sample Rate	20 kHz
Pass band Frequency	100 Hz
Stop band Frequency	300 Hz
Pass band Ripple	0.1-dB
Stop band Ripple	80-dB
Rate of Side-lobe Attenuation	6-dB/Octave

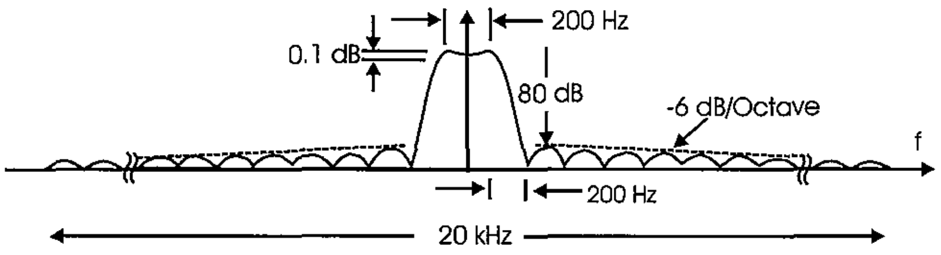


Figure 5.1 Specifications of Low pass Filter

We determine that a 360-tap FIR filter is required to meet these specifications. The processing task implemented directly with this FIR filter is indicated in Figure 5.2. The computational workload required of this implementation is 360 ops per output or, since there is no sample rate change, 360 ops per input.

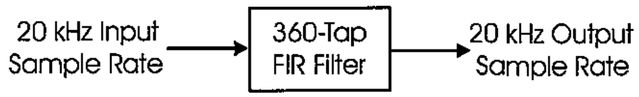


Figure 5.2 Initial Implementation of a Filtering Task

The Nyquist rate for this filter, equal to the filter two-sided bandwidth plus the transition bandwidth, is seen to be 400 Hz. This rate is 1/50th of the input sample rate. If we were to permit sample rate changes, we could reduce the sample rate by 50 and operate the filter at an output rate of 400 Hz. Let us examine this option. A 50-stage polyphase partition of the 360-tap filter is shown in Figure 5.3.

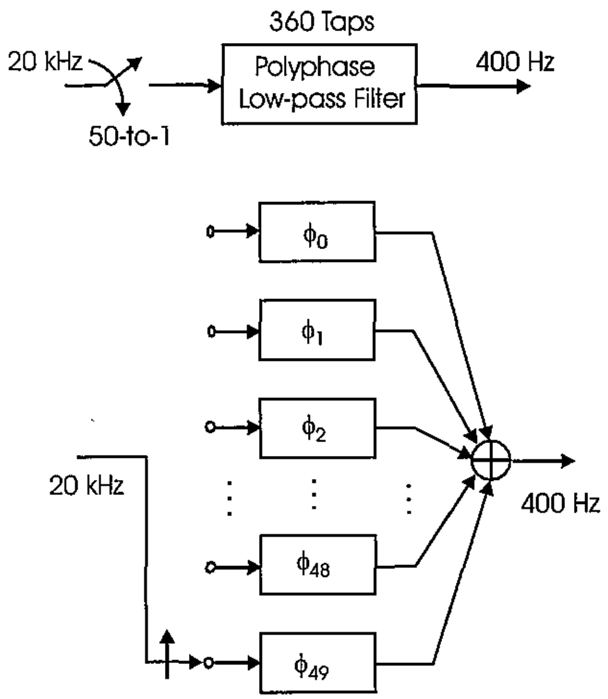


Figure 5.3 50-to-1 Polyphase Partition and Down Sampling of Low pass Filter

By simple division, we determine the average length of each polyphase filter stage is 360/50 or 7.2 taps. In actuality, when loading the filter coefficients by successive columns of length 50, we would find that the first 10 subfilters contain 8 taps and the remaining 40 subfilters contain 7 taps. There is no problem with the subfilters being different lengths. The hardware or software implementation of the filter would, in the interest of regularity, simply zero extend all the subfilters to length 8. Another option would alter the filter length to be a multiple of 50, or relax a design specification to design the filter with 350 taps, or oversatisfying the specification and design the filter with 400 taps. With the 350-tap version, the polyphase partition would require 350 ops per output but would only require 7 ops per input. This is significantly less than that required by the direct implementation suggested by Figure 5.2. Further, while we require all 350 coefficients to implement the filter, we only require 7 storage registers for data storage as opposed to the 350 registers required in the direct form. Our only problem at this point is we have not satisfied one of the filter specifications, namely that the output sample rate be the same as the input sample rate.

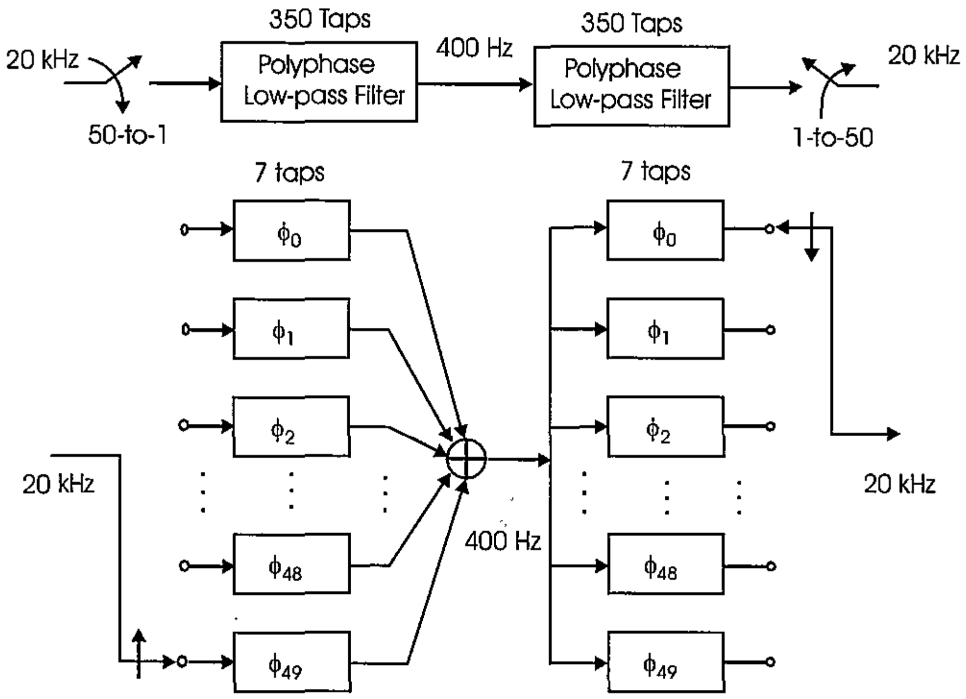


Figure 5.4 Cascade 50-to-1 and 1-to-50 Polyphase Maintains Constant Sample Rate

We respond to this last objection by following the down-sample filter by an up-sample filter. This cascade filter form is shown in Figure 5.4. In this form, we output one sample from the input process for every 50 input samples and then output 50 output samples for every intermediate input sample. This process generates one output sample for each input sample. We note that when the filter is implemented in this form there are 7 ops per input and 7 ops per output for a total workload of only 14 ops per input-output pair. This compares with 350 ops per input-output pair in the direct implementation. Similarly, since only one arm of the polyphase filter is engaged at any one time, we only require 7 data registers for the input and 7 data registers for the output, a total of 14 registers which is certainly a reduction from the 350 data registers for the direct implementation. Figure 5.5 presents the same filter but reflects the reduced resources required for this filter. Compare Figure 5.5 to Figure 5.2. The lesson we have learned in this example is that a filter should be operated at its Nyquist rate. If the desired output sample rate is different from the Nyquist rate, consider that a problem to be addressed in a second filter.

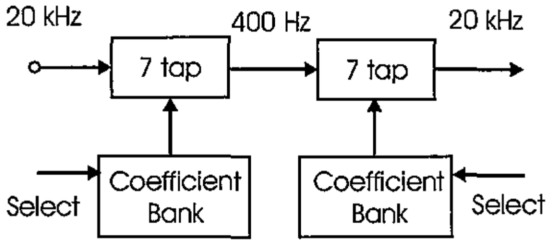


Figure 5.5 Minimum Resource Cascade 50-to-1 and 1-to-50 Polyphase Filter

Examining Figure 5.4, in particular the up sampling half of the cascade, we can clearly see that a single stage 7-tap FIR filter can perform the sequential processing tasks of the entire polyphase filter. This is obvious since all the stages of the filter store and process the same input series obtained from the previous stage. With this awareness, when we examine the input down-sampling structure we face a quandary. The separate stages store and process different input data streams and it would appear that the resource sharing of the output stage is not applicable to the input stage. The quandary is that if we can share resources in the output filter we must be able to do the same in the input filter since the two processes perform dual functions. The statement of the problem contains the solution to the problem. The two filter banks must be dual structures; they cannot both be tapped delay line implementations of the filter stages. We now examine the dual form or alternate form of the FIR filter.

5.1.1 Partial Sum Accumulator: The Dual Form

We start with the up-sampling filter to first establish a sequence of transformations that we then apply to the down-sampling filter. Figure 5.6 presents the polyphase partition of a three-stage up-sampling filter. The separate arms of the filter are implemented as the standard tapped delay line filter structure that supports the conventional inner product form or multiplier-accumulator (MAC) form of the FIR filtering process. We note that the data registers of the three filters all contain the same input samples. In Figure 5.7 we implement the filter with a single register set that is accessed by the three sets of MACs that feed the output commutator. We note that even though there are three sets of MACs we use them sequentially, one at a time. Rather than use three MACs, we can time share one MAC and commutate filter weights to synthesize successive filter sections. This structure is shown in Figure 5.8. Note here that one polyphase filter segment and a commutator pointing to filter weights can form a filter with any number of polyphase arms. The process of presenting successive weight sets to the single stage can be likened to the operation of a Gatling gun in which sets of weights are successively rolled into the filtering task.

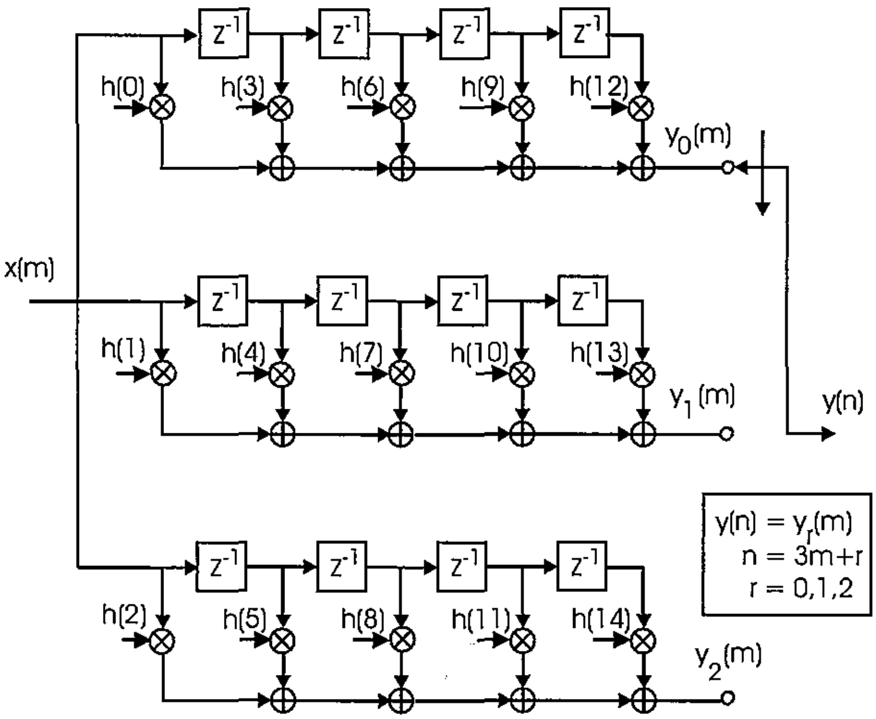


Figure 5.6 Standard Three-stage, Three-MAC, Polyphase Up-sampling Filter

The sequence of transformations just applied to the up-sampling filter is now applied to the dual down-sampling filter. Figure 5.9 presents the polyphase partition of a 1-to-3 down-sampling filter. The separate stages are implemented in the conventional MAC structure.

Note that the three stages are combined as a dual of the three stages in the corresponding up-sampling process of Figure 5.6. Comparing the two figures, we see that the input node of Figure 5.6 has become the output-summing junction of Figure 5.9, and that the arrows indicating signal flow to and from the three stages and the commutator direction have been reversed. Since the tasks of up sampling and down sampling are dual processes it is not surprising that the stages are connected in dual forms. What we now realize is that dual structures should also be applied to the stage implementations. This dual form with nodes replacing summing junctions, summing junctions replacing nodes, and reversed arrow directions and coefficient ordering is shown in Figure 5.10.

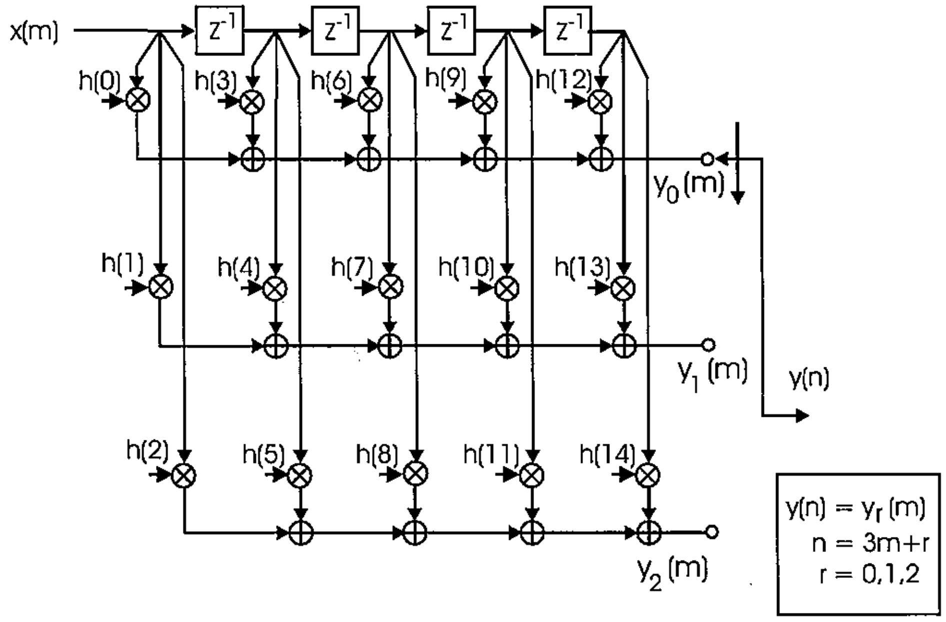

x(m)
z⁻¹
h(0) h(3) h(6) h(9) h(12)
h(1) h(4) h(7) h(10) h(13)
h(2) h(5) h(8) h(11) h(14)
y0(m)
y1(m)
y2(m)
y(n)
y(n) = yr(m)
n = 3m+r
r = 0,1,2

Figure 5.7 Shared Registers of Three-stage, Three-MAC, Polyphase Up-sampling Filter

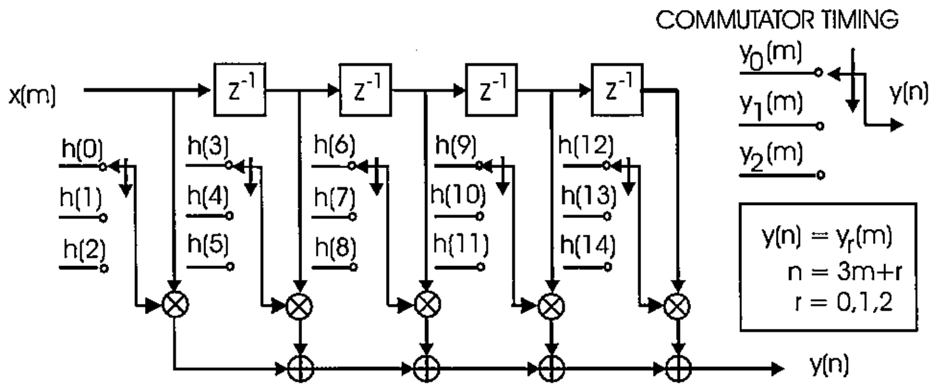


Figure 5.8 Minimum Resource Version of Three-stage Polyphase Up-sampling Filter

This structure represents an alternate implementation of a FIR filter structure and is known as the partial sum accumulator form. Interestingly, the tapped delay line model performs an inner product and is the mental image of a FIR filter conjured by an imbedded system programmer. The partial sum accumulator model performs simultaneous parallel processing and is the mental image of a FIR filter conjured by an application-specific integrated circuit (ASIC) designer. Note that in the tapped delay line model, the registers contain input data samples. These samples are stored and accessed by shifting to apply the successive filter weights to form the sum for successive output samples. In the partial sum accumulator, the registers contain the sum of products. The products are formed between each input sample and all filter coefficients. The products are accumulated over successive inputs and then shifted towards the filter output port to form the final sum for successive output samples.

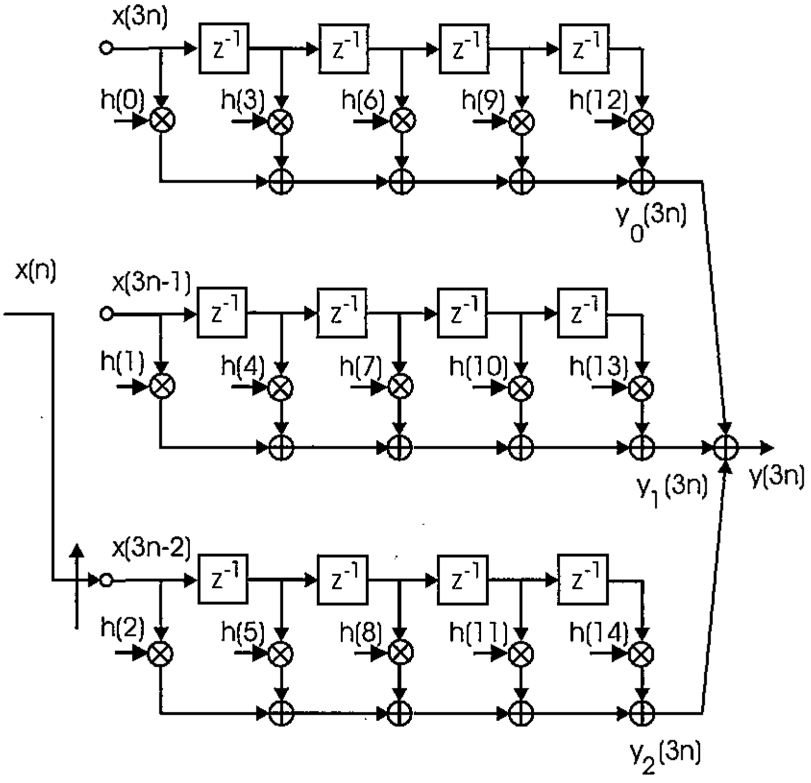


Figure 5.9 Standard Three-stage, Three-MAC, Polyphase Down-sampling Filter

In Figure 5.10 we note that the sum formed at the output of the three filters and at the output of the combined filter is in fact performed in the same accumulator. We also note that the outputs of the earlier accumulators eventually arrive at the output accumulator where they are combined to form the composite filter output. We can save memory by combining the outputs of the separate accumulators as a single accumulator. This combining is shown in Figure 5.11, which is seen to be the dual form of Figure 5.7.

We note here that even though there are three sets of multipliers to apply coefficients to the data that feed the common accumulators we use them sequentially, one at a time. Rather than use three multiplier sets we can time share one set of multipliers and commutate filter weights to synthesize successive filter sections. This structure is shown in Figure 5.12. Note here again that one polyphase filter segment and a commutator pointing to filter weights can form a filter with any number of polyphase arms. This process of presenting successive weight sets to the single stage can again be likened to the operation of a Gatling gun as was the dual process presented in Figure 5.8. Note in particular the dual structure of the filters shown in Figures 5.8 and 5.11. These two efficient forms of the polyphase filter are used in their appropriate position in the filter structure presented in Figure 5.5.

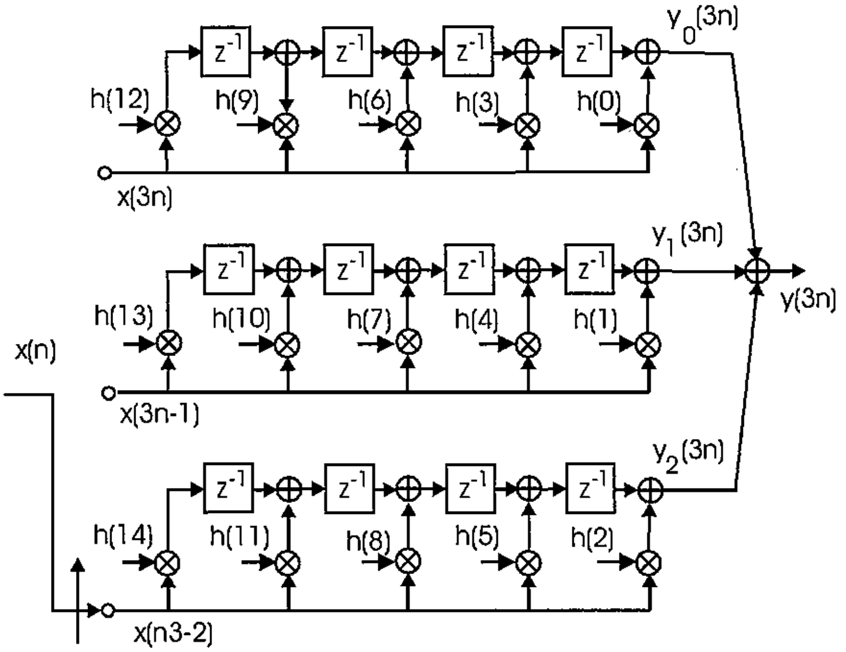


Figure 5.10 Dual Form Three-stage Polyphase Down-sampling Filter

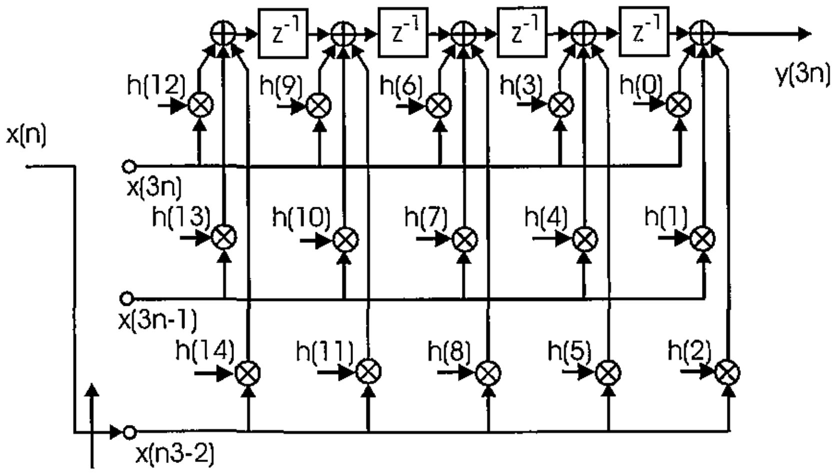


Figure 5.11 Shared Registers of Three-stage, Multiplier-accumulator, Polyphase Down-sampling Filter

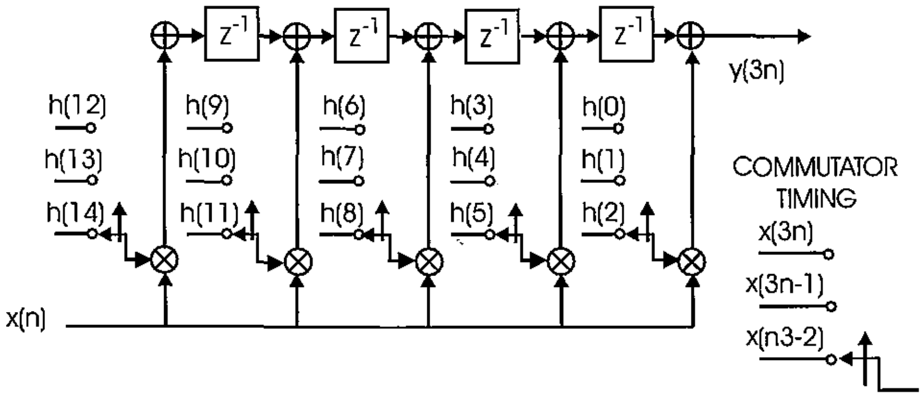


Figure 5.12 Minimum Resource Version of Three-stage Polyphase Down-sampling Filter

5.1.2 Generate Baseband Narrowband Noise

We now address a related signal-processing task: that of forming a baseband time series with a narrow bandwidth at a sample rate that is large compared to the signal bandwidth. The specific example we examine is that of forming a digital narrowband noise source. The spectral characteristics of the filtered noise, by a remarkable coincidence, are the same as those presented in Table 5-1. The narrowband noise is to be sampled at 20 kHz, have a bandwidth of 100 Hz, a transition bandwidth of 200 Hz, and a dynamic range of 80-dB. In a previous section, we learned that the FIR filter that meets these specifications has a length of 350 taps. Figure 5.13 presents a brute-force solution to the processing task. We operate a long period PN noise generator at the desired 20 kHz output rate and deliver the noise sequence to the 350-tap filter. As shown in Figure 5.14, the output spectrum is equal to the product of the input spectrum and the power spectral response of the filter. The output time series of the filter will inherit the spectral properties of the filter and hence meets the spectral requirements placed on the processing task.

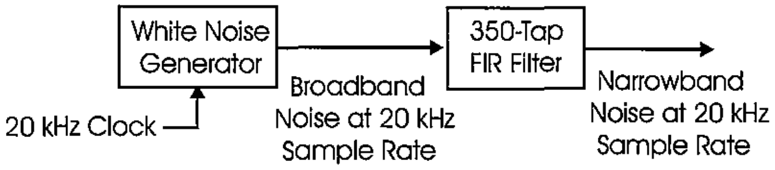

White Noise Generator
20 kHz Clock
Broadband Noise at 20 kHz Sample Rate
350-Tap FIR Filter
Narrowband Noise at 20 kHz Sample Rate

Figure 5.13 Block Diagram of Simple Narrowband Noise Generator

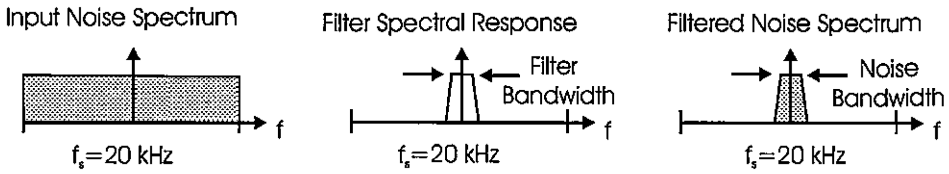


Figure 5.14 Input Spectrum, Filter Spectral Response, and Output Spectrum

The problem with the noise generation process presented in Figure 5.13 is the high workload per output noise sample of 350 ops per output. Following the lesson taught in an earlier section, that the filter should operate at its Nyquist rate, we recast the solution in the following way. We operate the noise generator at the filter's Nyquist rate of 400 Hz and then use the polyphase partition of the 350-tap filter as a 1-to-50 up sampler. This is shown in Figure 5.15. The polyphase partition is implemented as a single 7-tap filter with the 50 coefficient sets applied sequentially to the filter for each input sample to the filter. In this structure, the workload is 7 ops per output point, a rather significant improvement relative to the 350 ops per output point of the simple solution.

Conceptually, the input noise spectrum at 400 Hz sample rate has been up sampled by 1-to-50 zero packing which gives us access to the desired output sample rate of 20 kHz. The 350-tap filter processes the zero-packed data to extract the original baseband spectrum at the higher sample rate. Only the non-zero samples of the zero packed input data contribute to the output samples. Tracking the position of these samples in the filter to determine which coefficients interact with the known data samples is equivalent to the polyphase partition of the filter. The zero packing is introduced to help visualize the process, not to offer implementation instructions. The spectrum of the input signal, of the zero-packed input signal, of the filter, and of the filter output is shown in Figure 5.16.

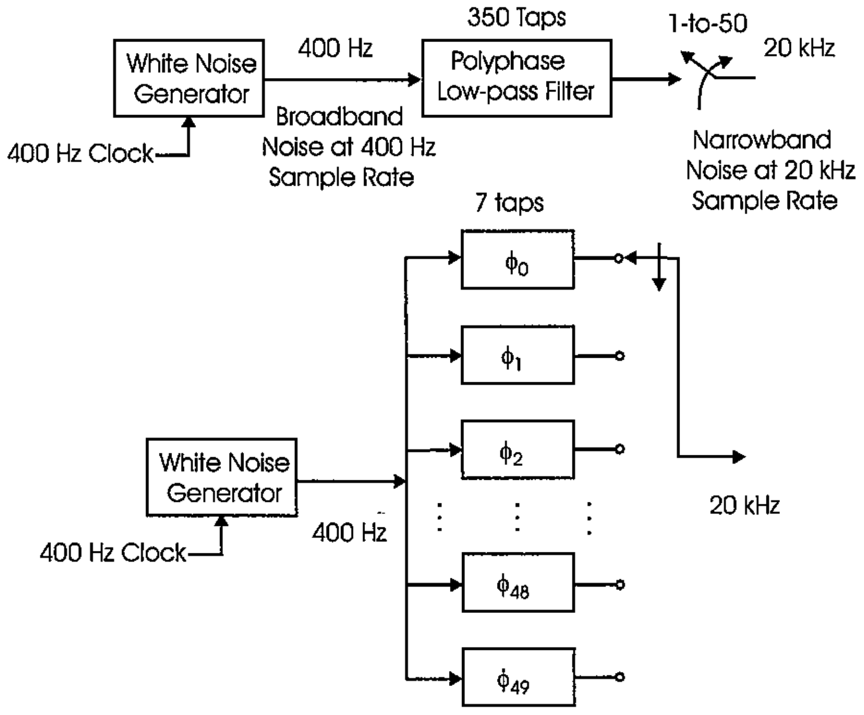


Figure 5.15 Block Diagram of Efficient Narrowband Noise Generator

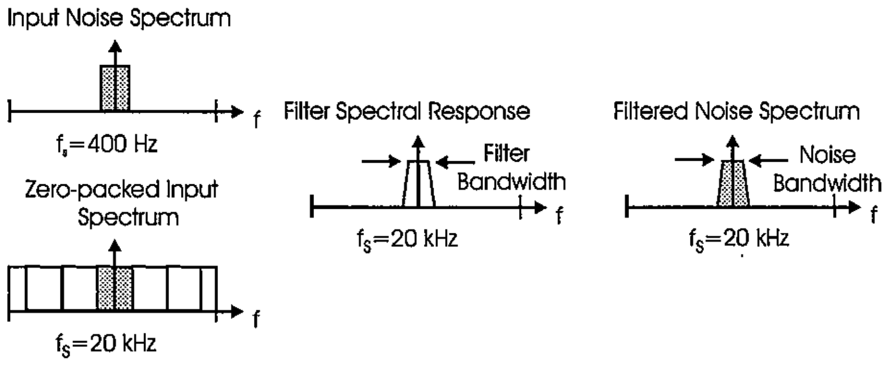


Figure 5.16 400-Hz Input Spectrum, 1-to-50 Zero-packed 20 kHz Input Spectrum, Filter Spectral Response, and Filtered Output Spectrum

5.1.3 Generate Narrowband Noise at a Carrier Frequency

A variant of the previous example is that we want the same narrowband noise sampled at 20 kHz rate but want it centered at 1200 Hz. We might want such a noise signal to add to a narrowband communication signal to test its susceptibility to additive noise. For instance, a V.22 bis telephone modem operates full duplex in frequency division multiplex (FDM) mode, transmitting or receiving at the two center frequencies of 1200 Hz and 2400 Hz. One way to obtain the narrowband noise centered at the 1200 Hz carrier is to up convert the baseband signal generated by the previous multirate up-sampling process. The implementation of this option is shown in Figure 5.17. The spectrum of the baseband signal and of the spectrally shifted signal is shown in Figure 5.18.

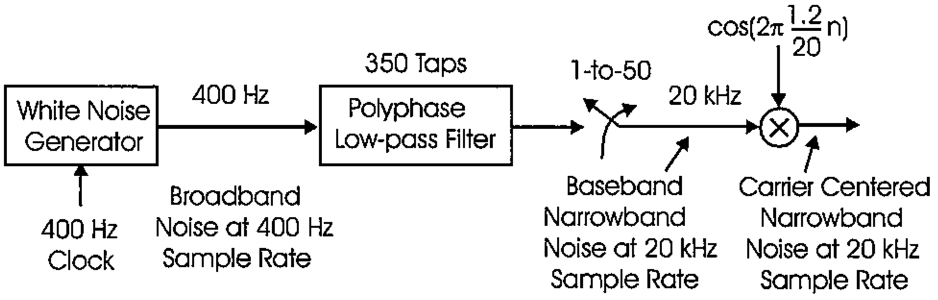

cos(2π 1.2/20 n)
350 Taps
1-to-50
400 Hz
20 kHz
White Noise Generator
Polyphase Low-pass Filter
400 Hz Clock
Broadband Noise at 400 Hz Sample Rate
Baseband Narrowband Noise at 20 kHz Sample Rate
Carrier Centered Narrowband Noise at 20 kHz Sample Rate

Figure 5.17 Translate Baseband Signal Formed by Polyphase Filter

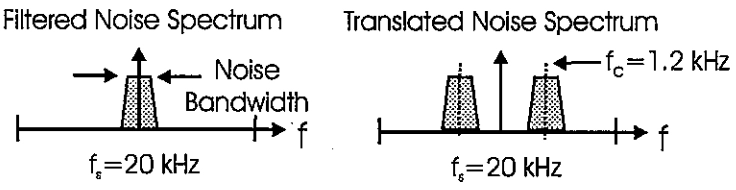


Figure 5.18 Baseband Spectrum and Translated Spectrum at 20 kHz Sample Rate

The center frequency selected for the up conversion process is 1.2 kHz, a frequency that happens to be a multiple of the input sample rate of 400 Hz. In our model of the process, the input signal was zero packed to raise the sample rate to 20 kHz. In the zero-packed model, copies of the spectrum are distributed over the 20 kHz interval at multiples of the 400 Hz input sample rate. What we accomplished was to place a spectrum at baseband, zero pack it to access spectral copies at multiples of 400 Hz, filter out all the copies, and translate the surviving baseband spectrum up to 1200 Hz. In an alternate process, we can translate the spectral response of the polyphase filter and have it extract the spectral copy of the input signal already residing at 1200 Hz.

The effect of translating a baseband prototype up-sampling polyphase filter to a multiple of the input rate can be easily described. Equation (5.1) presents the relation between the baseband prototype and its polyphase partition while (5.2) presents the same relationship for the translated version of the same filter.

$$\begin{array}{ll} \text{Prototype Filter} & h(n) \\ \text{Polyphase Partition} & h_r(n) = h(r + nM) \end{array} \qquad (5.1)$$

$$\begin{array}{ll} \text{Translated Filter} & g(n,k) = h(n)\cos[n\,\theta_k] \\ \text{Polyphase Partition} & g_r(n,k) = h_r(n)\cos[(r + nM)\,\theta_k] \end{array} \qquad (5.2)$$

When we substitute the center frequency θ_k defined in (5.3) into (5.2), we obtain (5.4).

$$\theta_k = \frac{2\pi}{M}k \qquad (5.3)$$

$$\text{Polyphase Partition} \quad g_r(n,k) = h_r(n)\cos\left(\frac{2\pi}{M}r\,k\right) \qquad (5.4)$$

Here we see that when the heterodyne is applied to the polyphase partition and the heterodyne corresponds to a multiple of the input sample rate, the heterodyne defaults to a scalar for each path that depends only on the index r, the polyphase arm, and the index k, the selected frequency offset. The scalar can be distributed through the polyphase coefficients as an offline operation or can be applied as an online scaling term to the input data as it enters each polyphase filter arm. The former makes sense if the filter is to service a single frequency, and the latter makes sense if the filter is to service multiple frequencies.

Applying the heterodyne to the filter coefficients allows the polyphase filter to accomplish three operations simultaneously: sample rate change, bandwidth control, and center frequency translation. This filter requires only 7 ops per input sample to accomplish all three processing tasks. The form of the translated polyphase filter that accomplishes these three tasks is shown in Figure 5.19.

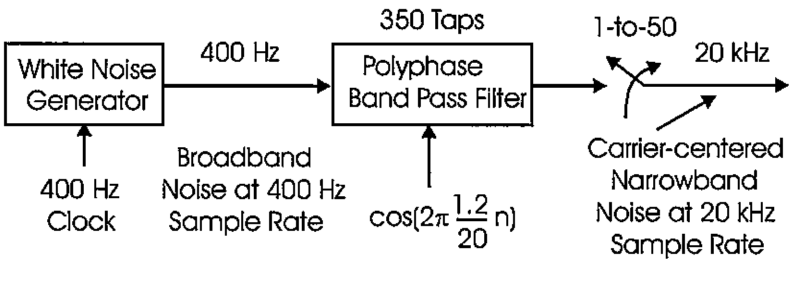

White Noise Generator
400 Hz
350 Taps
Polyphase Band Pass Filter
1-to-50
20 kHz
400 Hz Clock
Broadband Noise at 400 Hz Sample Rate
cos(2π 1.2/20 n)
Carrier-centered Narrowband Noise at 20 kHz Sample Rate

Figure 5.19 Band-pass Signal Formed by Translated Polyphase Filter

5.2 Workload of Multirate Filter

It is common practice to describe a filter's specifications relative to its sample rate. For instance, a filter may have a bandwidth that is one fourth of the sample rate. It is also common to describe specifications relative to the filter's bandwidth. As an example, a Nyquist or square-root Nyquist filter with a roll off of 0.20 has an excess bandwidth equal to 20% of the symbol rate. Describing filter parameters of a multirate filter relative to its sample rate presents an interesting quandary. Which sample rate, input or output? What does one do when the filter supports arbitrary resampling ratios? We now address a perspective that proves to be useful when specifying filter specifications relative to filter sample rate.

Figure 5.20 shows the sample rates associated with a multirate filter while Figure 5.21 shows the spectral description of a multirate filter at the input rate and output sample rates. We use the two versions of the spectral description to examine a connection between filter length N and the resampling ratio M.

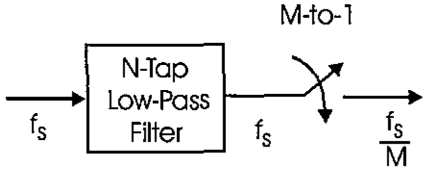


Figure 5.20 Multirate Filter Indicating Input and Output Sample Rates

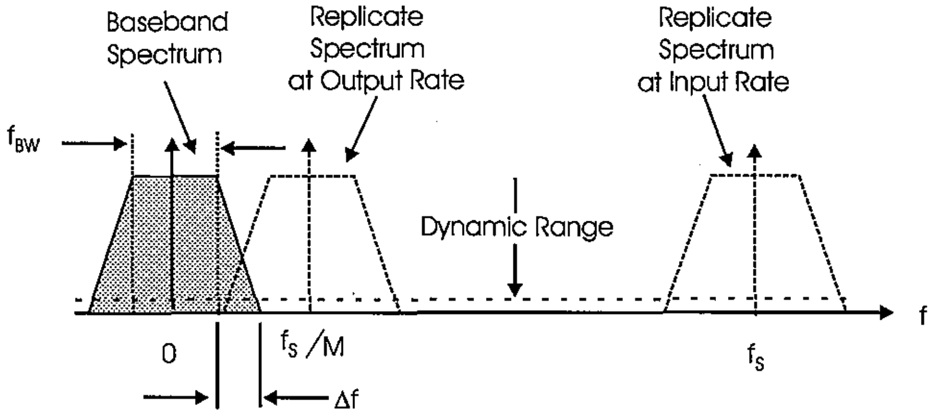


Figure 5.21 Spectra of Low pass Filter at Input and Output Sample Rates

Equation (5.5) is a statement of how the transition bandwidth Δf is related to filter length N and filter pass band and stop band ripple parameters δ_1 and δ_2.

$$\Delta f = K(\delta_1,\delta_2)\frac{f_S}{N} \tag{5.5}$$

The filter bandwidth, described relative to output sample rate, is indicated in (5.6) where the parameter α represents the fractional bandwidth of the pass band relative to the output sample rate.

$$f_{BW} = \alpha\frac{f_S}{M} \tag{5.6}$$

As shown in (5.7), we satisfy the Nyquist criterion when the output sample rate is equal to the sum of the filter's two-sided bandwidth and the transition bandwidth.

$$\frac{f_S}{M} = \alpha\frac{f_S}{M} + \Delta f \tag{5.7}$$

Substituting (5.5) in (5.7) we obtain (5.8), which when rearranged leads to (5.9).

$$\frac{f_S}{M} = \alpha\frac{f_S}{M} + K(\delta_1,\delta_2)\frac{f_S}{N} \tag{5.8}$$

$$N = M\frac{K(\delta_1,\delta_2)}{(1-\alpha)} \tag{5.9}$$

We now interpret the relationship derived in (5.9). We know that the filter length N is proportional to the stop band and pass band ripple controlled parameter $K(\delta_1,\delta_2)$ and the ratio of sample rate to transition bandwidth. This relationship is shown in (5.10) with respect to the input sample rate fs.

$$N = K(\delta_1,\delta_2)\frac{f_S}{\Delta f} \tag{5.10}$$

We find conflicting definitions of the estimate of filter length when comparing (5.10) and (5.9). Equation (5.9), strangely enough, seems to couple the filter length to the resampling ratio that follows the filter, while (5.10) does not. How can this be? We first resolve this apparent inconsistency.

The inconsistency is related to the fact that transition bandwidth is not normally related to filter bandwidth, but is in the multirate filter. Figure 5.22 presents the spectra of two low pass filters with different bandwidths but with a fixed ratio of filter bandwidth to transition bandwidth. Note as the filter bandwidth is reduced, so is the transition bandwidth. This means that the transition bandwidth is proportional to the filter bandwidth and as the filter bandwidth is reduced so is the transition bandwidth. Filters with a fixed ratio between bandwidth and transition bandwidth are said to have a *constant form factor*. Analog filters have this property. Hence, as the filter bandwidth and the transition bandwidth become smaller we apply additional down-sampling or we increase M, the down sampling ratio. We now conclude that a filter with a constant form factor inserted in a multirate filter has a transition bandwidth that varies inversely with the down sample ratio. The relationship that illustrates this is found by rearranging (5.7) to obtain (5.11).

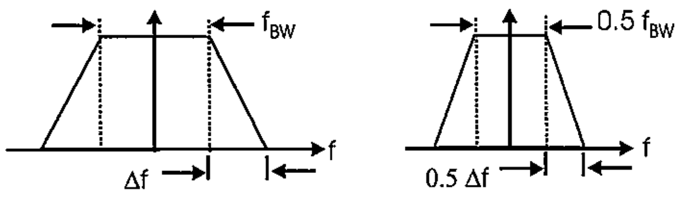


Figure 5.22 Spectra of Two Low pass Filters of Different Bandwidths but with Fixed Ratio of Bandwidth to Transition Bandwidth

$$\Delta f = (1-\alpha)\frac{f_S}{M} \tag{5.11}$$

We now can solve for the ratio of sample rate to transition bandwidth as shown in (5.12).

$$\frac{f_S}{\Delta f} = M\frac{1}{(1-\alpha)} \tag{5.12}$$

Now solving for filter length with the transition bandwidth coupled to filter bandwidth, we have the results shown in (5.13), which matches the result derived in (5.9).

$$N = K(\delta_1,\delta_2)\frac{f_S}{\Delta f} = M\frac{K(\delta_1,\delta_2)}{(1-\alpha)} \tag{5.13}$$

Rearranging (5.13), we have (5.14), a form that offers interesting insight into multirate filters.

$$\frac{N}{M} = \frac{K(\delta_1,\delta_2)}{(1-\alpha)} \tag{5.14}$$

We first examine the units of the ratio N/M. N, the filter length, has units of ops/output, while M, the resampling ratio has units of inputs/output. Thus the units of N/M are seen in (5.15) to be ops/input.

$$\frac{N(ops/output)}{M(input/output)} = \frac{N}{M}(ops/input) \tag{5.15}$$

The quantity N/M comes about naturally as we partition a length N prototype filter into M paths. The length of each path is N/M, which is seen as the amount of work required to perform each path in the polyphase filter partition. From (5.14) we learn that ops/input is a constant dependent upon the filter quality, where quality is defined by pass band and stop band ripples δ_1 and δ_2, respectively, and the fractional bandwidth $(1-\alpha)$ allocated to the transition band. Typical values of $K(\delta_1,\delta_2)$ are on the order of 3-to-4 while typical values of $1/(1-\alpha)$ are on the order of 2-to-5. From these typical values we determine that N/M spans the range of 6-to-20 ops/input. The example demonstrated in the narrowband noise generator required an N/M ratio equal to 7. Of course if we reduce the filter pass band or stop band ripple or if we reduce the filter transition bandwidth, the quality of the filter is increased, and the N/M ratio is increased appropriately.

The important concept presented here is that in any data-resampling task, the workload per data point is defined entirely by the filter quality. If we are performing a 10-to-1 down sampler the workload per input sample N_1/M_1 is a fixed constant, say 8. If we elect to perform a 20-to-1 down sampler, the filter length doubles but so does the number of stages so that the ratio N_2/M_2 is still 8. The workload per input N/M is fixed independently of sample rate change even though the filter length N is proportional to the sample rate change. As the sample rate ratio changes, the filter length changes but the number of stages change proportionally so that the ratio N/M is fixed, fixed to the ratio defined by the filter quality.

References

Crochiere, Ronald and Lawrence Rabiner, *Multirate Signal Processing*, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1983.

Fliege, Norbert, *Multirate Digital Signal Processing: Multirate Systems, Filter Banks, Wavelets*, West Sussex, John Wiley & Sons, Ltd., 1994.

Harada, Hiroshi and Ramjee Prasad, *Simulation and Software Radio for Mobile Communications*, Norwood, MA, Artech House, 2002.

Hentschel, Tim, *Sample Rate Conversion in Software Configurable Radios*, Norwood, MA, Artech House, 2002.

Jovanovic-Dolecek, Gordana, *Multirate Systems: Design and Applications*, London, Idea Group, 2002.

Mitra, Sinjit and James Kaiser, *Handbook for Digital Signal Processing*, New York, John Wiley & Sons, 1993.

Problems

5.1 Design a 5-to-1 5-path polyphase down sampling filter for the following performance requirements:

Input Sample Rate:	100 kHz		
Pass Band:	0-8 kHz	In-band Ripple	0.1-dB
Stop Band:	12-50 kHz	Stop band Atten.	60-dB
Output Sample Rate:	20 kHz		

First verify the design meets the specifications. Then partition the filter into a 5-path filter and verify that it successfully rejects out-of-band signals by having it process 1,000 samples of out-of-band signal, a sinusoid in the first Nyquist zone: specifically the sequence shown here.

x_1 = exp(j*2*pi*(0:999)*21/100);

Plot the real part of the time domain response and the windowed log magnitude FFT response of the polyphase filter output response to the signal x_1.

Generate a signal containing an in-band and an out-of-band component as shown next.

$x_2 = x_1$ + exp(j*2*pi*(0:999)*1.5/100);

Plot the real part of the time domain response and the windowed log magnitude FFT response of the polyphase filter output response to the signal x_2.

5.2 Design a 1-to-4 4-path polyphase up sampling filter for the following performance requirements:

Input Sample Rate:	25 kHz		
Pass Band:	0-10 kHz	In-band Ripple	0.1-dB
Stop Band:	15-50 kHz	Stop band Atten.	60-dB
Output Sample Rate:	100 kHz		

First verify the design meets the specifications. Then partition the filter into a 4-path filter and verify its operation by performing an impulse response test. Plot the time domain response and the log magnitude FFT response of the polyphase filter output response to the impulse.

Now have the filter up sample the following in-band sequence:

x_1 = exp(j*2*pi*(0:199)*3/20);

Plot the real part of the time domain response and the windowed log magnitude FFT response of the polyphase filter output response to the signal x_1.

5.3 Design a 1-to-20 20-path polyphase up-sampling filter for the following performance requirements:

Input Sample Rate:	10 kHz		
Pass Band:	0-4 kHz	In-band Ripple	0.1-dB
Stop Band:	6-100 kHz	Stop band Atten.	60-dB

Output Sample Rate:	200 kHz

First verify the design meets the specifications. Then partition the filter into a 20-path filter and verify its operation by performing an impulse response test. Plot the time domain response and the log magnitude FFT response of the polyphase filter output response to the impulse.

Now have the filter up sample following the in-band sequence:

x_1 = randn(1,200);

Plot the time domain response and the windowed log magnitude FFT response of the polyphase filter output response to the signal x_1.

5.4 Design a 1-to-20 20-path polyphase up sampling filter for the following performance requirements:

Input Sample Rate:	10 kHz		
Pass Band:	0-4 kHz	In-band Ripple	0.1-dB
Stop Band:	6-100 kHz	Stop band Atten.	60-dB
Output Sample Rate:	200 kHz		

Tune the filter by heterodyning the impulse response to the second Nyquist zone. This is shown here:

$h_2 = h_1$.*cos(2*pi*(0:length(h_1)−1)*20/200)

First verify the heterodyned design meets the specifications but offset to 20 kHz, the center of the second Nyquist zone. Then partition the filter into a 20-path filter and verify its operation by performing an impulse response test. Plot the time domain response and the log magnitude FFT response of the polyphase filter output response to the impulse.

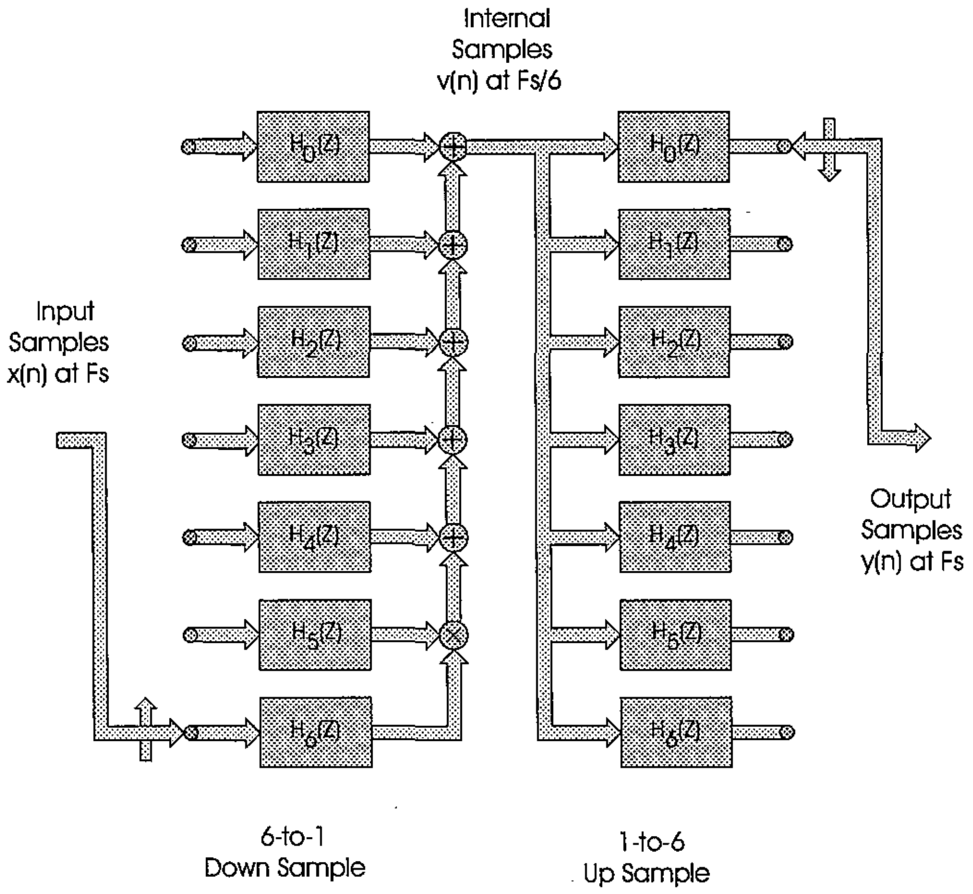
Now have the filter up sample the following in-band sequence noise sequence:

x_1 = randn(1,200);

Plot the time domain response and the windowed log magnitude FFT response of the polyphase filter output response to the signal x_1.

CHAPTER 6

Polyphase FIR Filters



Digital filters are employed in various signal-processing systems. The filter can often perform more than its intended primary filtering task. It can absorb many of the secondary signal processing tasks of a signal processing system. In this light, a filter should always be designed from a system viewpoint rather than as an isolated component in a system. There are so many ways of folding other functions into the filtering process we hardly know where to start. We thus use this section to demonstrate the great versatility to be had by performing multiple functions in a single resampling filter. This approach gives the reader an idea of how the material developed in later sections can be applied to specific applications. This path differs from the traditional approach of first presenting the theoretical tools and then presenting applications. The material presented here is expanded upon in later sections.

6.1. CHANNELIZER

The problem we address here to introduce the fundamental concepts is that of down converting a single frequency band or channel located in a multi channel Frequency Division Multiplexed (FDM) input signal. The spectrum of the input signal is a set of equally spaced, equal bandwidth FDM channels as shown in Figure 6.1. The input signal has been band limited by analog filters and has been sampled at a sufficiently high sample rate to satisfy the Nyquist criterion for the full FDM bandwidth.

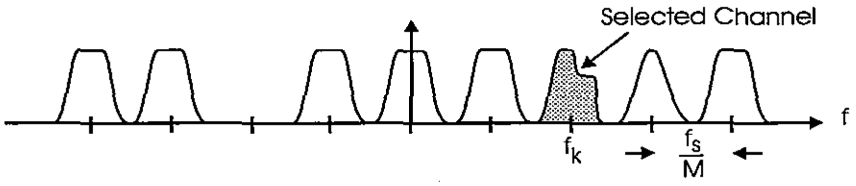


Figure 6.1 Spectrum of Multichannel Input Signal, Processing Task: Extract Complex Envelope of Selected Channel

The standard single processing process to down convert a selected channel is shown in Figure 6.2. This structure performs the standard operations of down conversion of a selected frequency band with a complex heterodyne, low pass filtering to reduce bandwidth to the channel bandwidth, and down sampling to a reduced rate commensurate with the reduced bandwidth. The structure of this processor is seen to be a digital signal processor implementation of a prototype analog I-Q down converter. We mention that the down sampler is commonly referred to as a decimator, a term that means to destroy every tenth one. Since nothing is destroyed and nothing happens in tenths, we prefer, and will continue to use, the more descriptive name, down sampler.

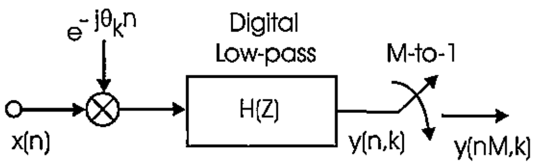


Figure 6.2 Standard Single Channel Down Converter

The spectrum we would observe at various points in the processing chain of Figure 6.2 is shown in Figure 6.3. Here we see the spectra at the input and output of the complex heterodyne, the spectral response of the baseband filter, the now oversampled spectrum at the output of the filter, and finally the spectrum at the output of the resampler

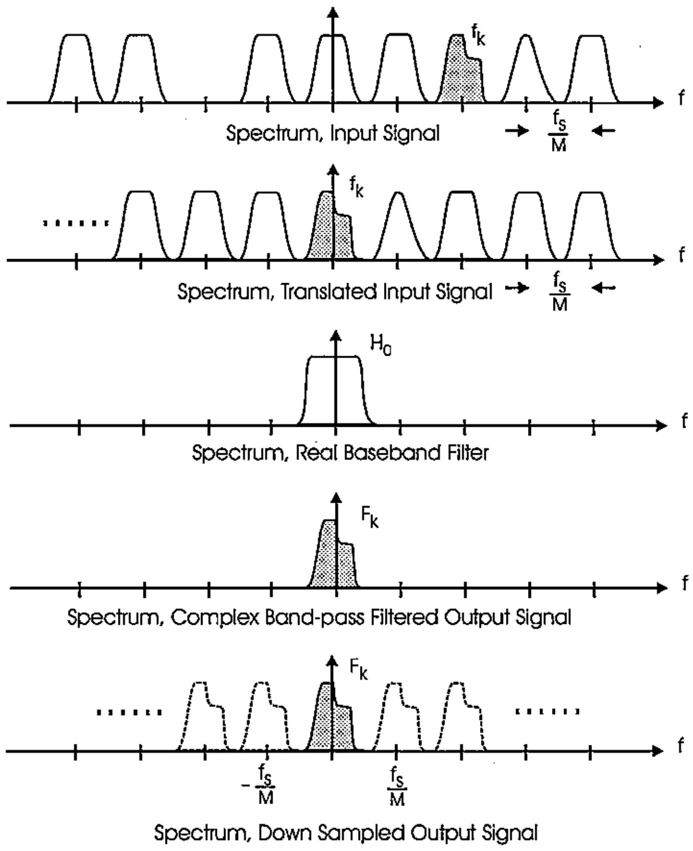

f_k
f
Spectrum, Input Signal
$\frac{f_s}{M}$
f_k
f
Spectrum, Translated Input Signal
$\frac{f_s}{M}$
H_0
f
Spectrum, Real Baseband Filter
F_k
f
Spectrum, Complex Band-pass Filtered Output Signal
F_k
f
$-\frac{f_s}{M}$
$\frac{f_s}{M}$
Spectrum, Down Sampled Output Signal

Figure 6.3 Spectra Observed at Various Points in Processing Chain of Standard Down Converter

The expression for y(n,k), the time series output from the down converted kth channel, prior to resampling, is a simple convolution as shown in (6.1).

$$\begin{aligned} y(n,k) &= [x(n)\ e^{-j\theta_k n}] * h(n) \\ &= \sum_{r=0}^{N-1} x(n-r)\ e^{-j\theta_k (n-r)} h(r) \end{aligned} \tag{6.1}$$

The output data from the complex mixer is complex, hence is represented by two time series, I(n) and Q(n). The filter with real impulse response h(n) is implemented as two identical filters, each processing one of the quadrature time series. The convolution process is often performed by a simple digital filter that performs the multiply and add operations between data samples and filter coefficients extracted from two sets of addressed memory registers. In this form of the filter, one register set contains the data samples while the other contains the coefficients that define the filter impulse response. This structure is shown in Figure 6.4.

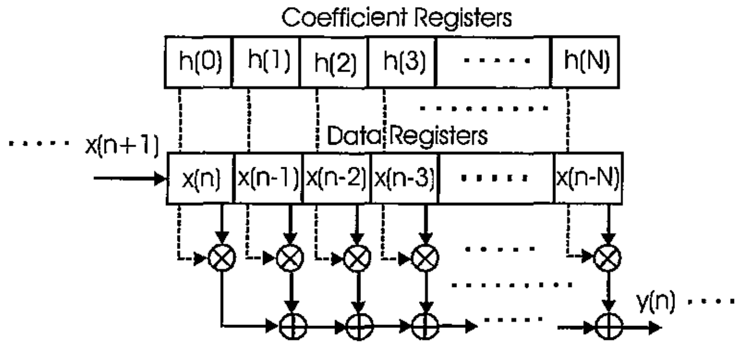


Figure 6.4 Tapped Delay Line Digital Filter: Coefficients and Data Registers, Multipliers, and Adders

We can rearrange the summation of (6.1) to obtain a related summation reflecting the *equivalency theorem.* The equivalency theorem states that the operations of down conversion followed by a low pass filter are totally equivalent to the operations of a band-pass filter followed by a down conversion. The block diagram demonstrating this relationship is shown in Figure 6.5, while the rearranged version of (6.1) is shown in (6.2).

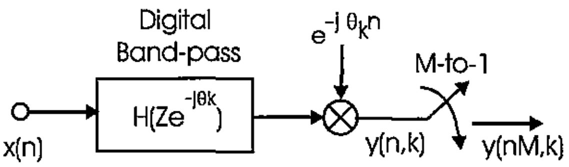


Figure 6.5 Band-pass Filter, Version of Down Converter

Note here that the up-converted filter, h(n) exp(jθ_kn), is complex and as such its spectrum resides only on the positive frequency axis without a negative frequency image. This is not a common structure for an analog prototype because of the difficulty of forming a pair of analog quadrature filters exhibiting a 90-degree phase difference across the filter bandwidth. The closest equivalent structure in the analog world is the filter pair used in image-reject mixers and even there, the phase relationship is maintained by a pair of complex heterodynes.

$$\begin{aligned} y(n,k) &= \sum_{r=0}^{N-1} x(n-r)\, e^{-j\theta_k(n-r)}\, h(r) \\ &= \sum_{r=0}^{N-1} x(n-r)\, e^{-jn\theta_k}\, h(r)\, e^{jr\theta_k} \\ &= e^{-j\,n\,\theta_k} \sum_{r=0}^{N-1} x(n-r)\; h(r) e^{j\,r\,\theta_k} \end{aligned} \tag{6.2}$$

Applying the transformation suggested by the equivalency theorem to an analog prototype system does not make sense since it doubles the required hardware. We would have to replace a complex scalar heterodyne (two mixers) and a pair of low pass filters with a pair of band-pass filters, containing twice the number of reactive components, and a full complex heterodyne (four mixers). If it makes no sense to use this relationship in the analog domain, why does it make sense in the digital world? The answer is found in the fact that we define a digital filter as a set of weights stored in coefficient memory. Thus, in the digital world, we incur no cost in replacing the pair of low-pass filters h(n) required in the first option with the pair of band-pass filters h(n) cos(nθ_k) and h(n) sin(nθ_k) required for the second one. We accomplish this task by a simple download to the coefficient memory. The filter structures corresponding to the two sides of the equivalency theorem are shown in Figure 6.6. Note the input signal interacts with the complex sinusoid as a product at the filter input in the first option or through products in the convolution with the filter weights in the second option.

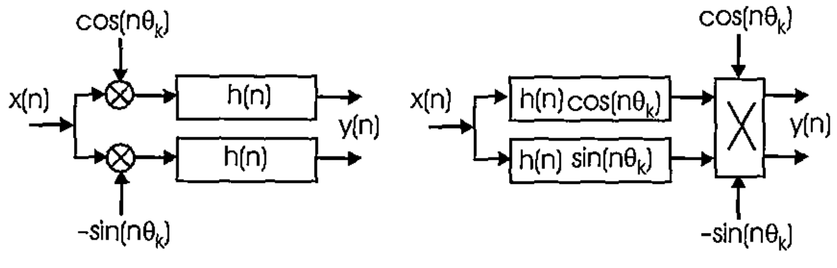


Figure 6.6 Block Diagrams Illustrating Equivalency Between Operations of Heterodyne and Baseband Filter with Band-pass Filter and Heterodyne

An interesting historical perspective is worth noting here. In the early days of wireless, radio tuning was accomplished by sliding a narrow band filter to the center frequency of the desired channel to be extracted from the FDM input signal. These radios were known as tuned radio frequency (TRF) receivers. The numerous knobs on the face of early radios adjusted reactive components of the amplifier chain to accomplish this tuning process. Besides the difficulty in aligning multiple tuned stages, shifting the center frequency of the amplifier chain often initiated undesired oscillation due to the parasitic coupling between the components in the radio. Edwin Howard Armstrong, an early radio pioneer, suggested that rather than move the filter to the selected channel, move the selected channel to the fixed frequency filter. This is known as the *superheterodyne principle,* a process invented by Armstrong in 1918 and quickly adopted by David Sarnoff of the Radio Corporation of America (RCA) in 1924. Acquiring exclusive rights to Armstrong's single-tuning dial radio invention assured the commercial dominance of RCA in radio broadcasting as well the demise of hundreds of manufacturers of TRF radio receivers. It seems we have come full circle. We inherited from Armstrong a directive to move the desired spectrum to the filter and we have readily applied this legacy to DSP-based processing. We are now proposing that, under appropriate conditions, it makes sense to move the filter to the selected spectral region.

We still have to justify the full complex heterodyne required for the down conversion at the filter output rather than at the filter input. Examining Figure 6.5, we note that following the output down conversion, we perform a sample rate reduction in which we retain one sample in every M-samples. Recognizing that there is no need to down convert the samples we discard in the down sample operation, we choose to down sample only the retained samples. This is shown in Figure 6.7. Here we note that when we bring the heterodyne to the low data rate side of the resampler, we are in fact also down-sampling the time series of the complex sinusoid. The rotation rate of the sampled complex sinusoid is θ_k and $M\theta_k$ radians per sample at the input and output respectively of the M-to-1 resampler.

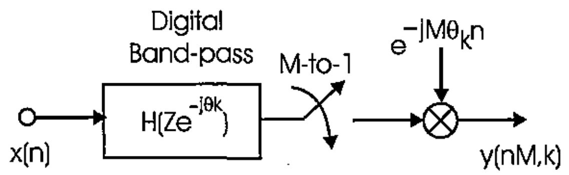


Figure 6.7 Down Sampled Band-pass Down Converter

This change in observed rotation rate is due to aliasing. When aliased, a sinusoid at one frequency or phase slope appears at another phase slope due to the resampling. We now invoke a constraint on the sampled data center frequency of the down-converted channel. We choose center frequencies θ_k, which will alias to DC (zero frequency) as a result of the down sampling to $M\theta_k$. This condition is assured if $M\theta_k$ is congruent to 2π, which occurs when $M\theta_k = k\,2\pi$, or more specifically, when $\theta_k = k\,2\pi/M$. The modification to Figure 6.7 to reflect this provision is seen in Figure 6.8. The constraint, that the center frequencies be limited to integer multiples of the output sample rate, assures aliasing to baseband by the sample rate change. When a channel aliases to baseband by the resampling operation the resampled related heterodyne defaults to a unity-valued scalar, which consequently is removed from the signal-processing path. If the center frequency of the aliased signal is offset by $\Delta\theta$ rad/smpl from a multiple of the output sample rate, the aliased signal will reside at an offset of $\Delta\theta$ rad/smpl from zero frequency at baseband. A complex heterodyne or baseband converter will shift the signal by the residual $\Delta\theta$ offset. This baseband mixer operates at the output sample rate rather than at the input sample rate for a conventional down converter. We can consider this required final mixing operation a post conversion task and allocate it to the next processing block.

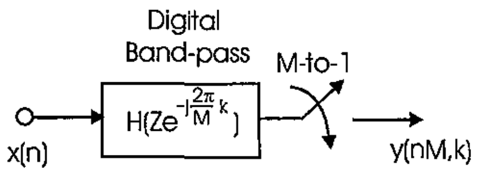


Figure 6.8 Band-pass Down Converter Aliased to Baseband by Down Sampler

The operations invoked by applying the equivalency theorem to the down conversion process guided us to the following sequence of maneuvers:

i slide the input heterodyne through the low-pass filters to their outputs,
ii doing so converts the low-pass filters to a complex band-pass filter,
iii slide the output heterodyne to the downside of the down sampler,
iv doing so aliases the center frequency of the oscillator,
v restrict the center frequency of the band-pass signal to be a multiple of the output sample rate,

vi doing so assures the alias of the selected pass band to base band by the resampling operation,
vii discard the now unnecessary heterodyne.

The spectral effect of these operations is shown in Figure 6.9. The savings realized by this form of the down conversion is due to the fact that we no longer require a quadrature oscillator or the pair of input mixers to effect the frequency translation.

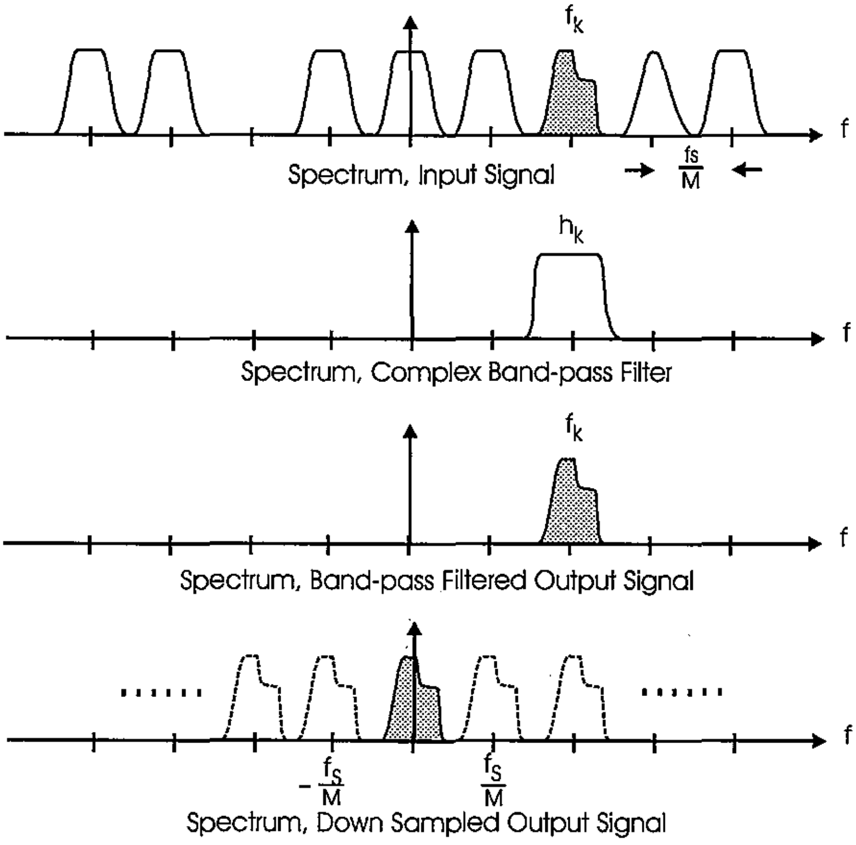

f_k
f
Spectrum, Input Signal
$\frac{fs}{M}$
h_k
f
Spectrum, Complex Band-pass Filter
f_k
f
Spectrum, Band-pass Filtered Output Signal
f
$-\frac{f_S}{M}$
$\frac{f_S}{M}$
Spectrum, Down Sampled Output Signal

Figure 6.9 Spectral Description of Down Conversion Realized by a Complex Band-pass Filter at a Multiple of Output Sample Rate, Aliased to Baseband by Output Resampling

6.1.1. Transforming the Band-pass Filter

Examining Figure 6.8, we note that the current configuration of the single channel down converter involves a band-pass filtering operation followed by a down sampling of the filtered data to alias the output spectrum to baseband. Following the idea developed in the previous section that led us to down convert only those samples retained by the down sampler, we similarly conclude that there is no need to compute the output samples from the pass-band filter that will be discarded by the down sampler. Conceptually we accomplish this in the following manner: After computing an output sample from the non-recursive (FIR) filter we shift the data in the filter register M positions and wait until M new inputs are delivered to the filter before computing the next output sample. In a sense, by moving input data through the filter in stride of length M we are performing the resampling of the filter at the input port rather than at the output port. Performing the resampling at the filter input essentially reorders the standard operations of the filter followed by a down sample with the operations of down sample followed by the filter. The formal description of the process that accomplishes this interchange is known as the noble identity, which we now describe in detail.

The noble identity is compactly presented in Figure 6.10, which we describe with similar conciseness as "The output from a filter $H(Z^M)$ followed by an M-to-1 down sampler is identical to an M-to-1 down sampler followed by the filter H(Z)." The Z^M in the filter impulse response tell us that the coefficients in the filter are separated by M-samples rather than the more conventional one sample delay between coefficients in the filter H(Z). We must take care to properly interpret the operation of the M-to-1 down sampler. The interpretation is that the M-to-1 down-sampled time series from a filter processing every Mth input sample presents the same output by first down sampling the input by M-to-1 to discard the samples not used by the filter when computing the retained output samples and then operating the filter on only the retained input samples. The noble identity works because M-samples of delay at the input clock rate is the same interval as one-sample delay at the output clock rate.

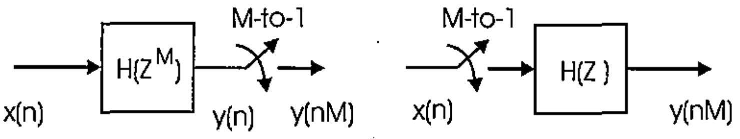


Figure 6.10 Statement of Noble Identity: A Filter Processing Every Math Input Sample Followed by an Output M-to-1 Down Sampler Is the Same as an Input M-to-1 Down Sampler Followed by a Filter Processing Every Input Sample.

We might ask, "Under what condition does a filter manage to operate on every Mth input sample?" We answer by rearranging the description of the filter to establish this condition so that we can invoke the noble identity. This rearrangement starts with an initial partition of the filter into M-parallel filter paths. The Z-transform description of this partition is presented in Equations 6.3 through 6.6, which we interpret in Figures 6.11 through 6.13. For ease of notation, we first examine the base-band version of the noble identity and then trivially extend it to the pass band version.

$$\begin{aligned} H(Z) &= \sum_{n=0}^{N-1} h(n)Z^{-n} \\ &= h(0) + h(1)\, Z^{-1} + h(2)\, Z^{-2} + h(3)\, Z^{-3} + \cdots + h(N-1)\, Z^{-(N-1)} \end{aligned} \tag{6.3}$$

Anticipating the M-to-1 resampling, we partition the sum shown in (6.3) to a sum of sums as shown in (6.4). This partition maps a one-dimensional array of weights (and index markers Z^{-n}) to a two-dimensional array. This mapping is sometimes called *lexicographic*, for natural order, a mapping that occurs in the Cooley-Tukey fast Fourier transform. In this mapping we load an array by columns but process the array by rows. In our example, the partition forms columns of length M containing M successive terms in the original array, and continues to form adjacent M-length columns until we account for all the elements of the original one-dimensional array.

$$\begin{array}{lclclc} H(Z) = h(0) & + & h(M+0)Z^{-M} & + & h(2M+0)Z^{-2M} & + \cdots \\ +h(1)Z^{-1} & + & h(M+1)Z^{-(M+1)} & + & h(2M+1)Z^{-(2M+1)} & + \cdots \\ +h(2)Z^{-2} & + & h(M+2)Z^{-(M+2)} & + & h(2M+2)Z^{-(2M+2)} & + \cdots \\ +h(3)Z^{-3} & + & h(M+3)Z^{-(M+3)} & + & h(2M+3)Z^{-(2M+3)} & + \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ +h(M-1)Z^{-(M-1)} & + & h(2M-1)Z^{-(2M-1)} & + & h(3M-1)Z^{-(3M-1)} & + \cdots \end{array} \tag{6.4}$$

We note that the first row of the two-dimensional array is a polynomial in Z^M, which we will denote $H_0(Z^M)$, a notation to be interpreted as an addressing scheme to start at index 0 and increments in stride of length M. The second row of the same array, while not a polynomial in Z^M, is made into one by factoring the common Z^{-1} term and then identifying this row as $Z^{-1}\, H_1(Z^M)$. It is easy to see that each row of (6.4) can be described as $Z^{-r}\, H_r(Z^M)$ so that (6.4) can be rewritten in a compact form as shown in (6.5).

$$\begin{aligned} H(Z) = H_0(Z^M) + Z^{-1}H_1(Z^M) + Z^{-2}H_2(Z^M) + Z^{-3}H_3(Z^M) + \\ Z^{-4}H_4(Z^M) + \cdots\cdots + Z^{-(M-1)}H_{M-1}(Z^M) \end{aligned} \tag{6.5}$$

We rewrite (6.5) in the traditional summation form as shown in (6.6), which describes the original polynomial as a sum of delayed polynomials in Z^M. The block diagram reflecting this M-path partition of a resampled digital filter is shown in Figure 6.11. The output formed from the M separate filter stages representing the M separate paths is the same as that obtained from the nonpartitioned filter. We have not yet performed the interchange of filter and resampling.

$$H(Z) = \sum_{r=0}^{M-1} Z^{-r} H_r(Z^M)$$
$$= \sum_{r=0}^{M-1} Z^{-r} \sum_{n=0}^{(N/M)-1} h(r+nM)\, Z^{-nM} \tag{6.6}$$

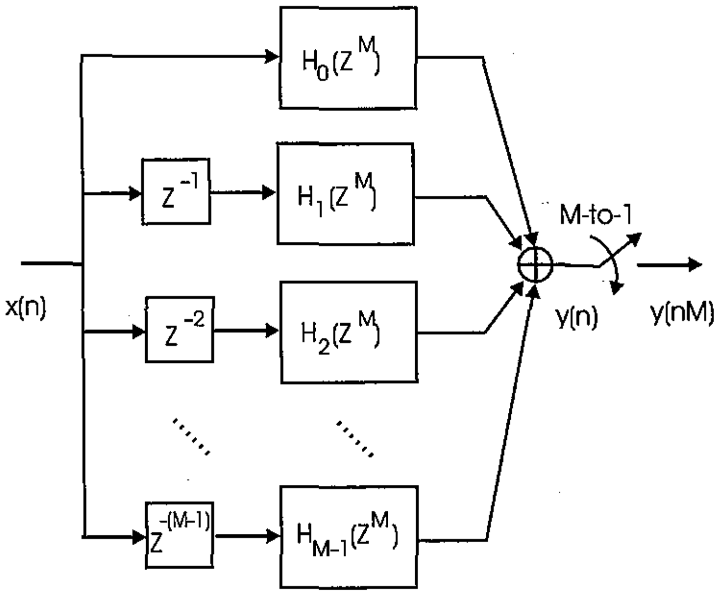


Figure 6.11 M-path Partition of Prototype Low pass Filter with Output Resampler

We first pull the resampler through the output summation element and down sample the separate outputs by performing the output sum only for the retained output sample points. With the resamplers now at the output of each filter, which operates on every Mth input sample, we are prepared to invoke the noble identity and pull the resampler to the input side of each filter stage. This is shown in Figure 6.12.

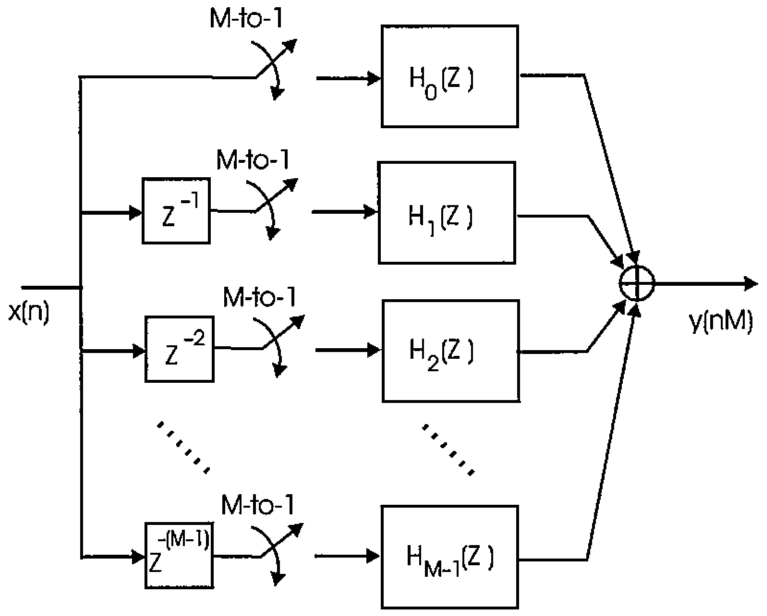


Figure 6.12 M-path Partition of Prototype Low-pass Filter with Input Resamplers

The input resamplers operate synchronously, all closing at the same clock cycle. When the switches close, the signal delivered to the filter on the top path is the current input sample. The signal delivered to the filter one path down is the content of the one sample delay line, which of course is the previous input sample. Similarly, as we traverse the successive paths of the M-path partition, we find upon switch closure that the kth path receives a data sample delivered k-samples ago.

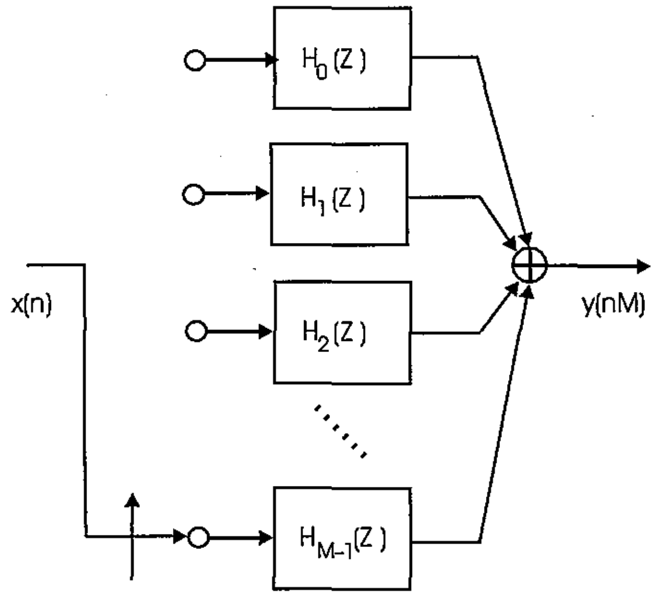


Figure 6.13 M-path Partition of Prototype Low pass Filter with Input Path Delays and M-to-1 Resamplers Replaced by Input Commutator

We conclude that the interaction of the delay lines in each path with the set of synchronous switches can be likened to an input commutator that delivers successive samples to successive legs of the M-path filter. This interpretation is shown in Figure 6.13.

We now complete the final steps of the transformation that changes a standard mixer down converter to a resampling M-path down converter. We note and apply the frequency translation property of the Z-transform. This property is illustrated and stated in (6.7). Interpreting the relationship presented in (6.7), we note that if h(n), the impulse response of a base-band filter, has a Z-transform H(Z), then the sequence $h(n)e^{+j\theta n}$, the impulse response of a pass band filter, has a Z-transform $H(Z\ e^{-j\theta n})$. Simply stated, we can convert a low pass filter to a band-pass filter by associating the complex heterodyne terms of the modulation process of the filter weights with the delay elements storing the filter weights.

$$\begin{aligned}
\text{if } H(Z) &= h(0)+h(1)Z^{-1}+h(2)Z^{-2}+\cdots+h(N-1)Z^{-(N-1)} \\
&= \sum_{n=0}^{N-1} h(n)Z^{-n} \\
\text{and } G(Z) &= h(0)+h(1)e^{j\theta}Z^{-1}+h(2)e^{j2\theta}Z^{-2}+\cdots+h(N-1)e^{j(N-1)\theta}Z^{-(N-1)} \\
&= h(0)+h(1)[e^{-j\theta}Z]^{-1}+h(2)[e^{-j\theta}Z]^{-2}+\cdots+h(N-1)[e^{-j\theta}Z]^{-(N-1)} \\
&= \sum_{n=0}^{N-1} h(n)\,[e^{-j\theta}Z]^{-n} \\
\text{then } G(Z) &= H(Z)\big|_{Z\Rightarrow e^{-j\theta}Z} = H(e^{-j\theta}Z)
\end{aligned} \tag{6.7}$$

We now apply this relationship to (6.2), or equivalently to Figure 6.11, by replacing each Z with $Z\,e^{-j\theta}$, or perhaps more clearly, replacing each Z^{-1} with $Z^{-1}\,e^{j\theta}$, with the phase term θ satisfying the congruency constraint of the previous section, that $\theta = k(2\pi/M)$. Thus Z^{-1} is replaced with $Z^{-1}\,e^{jk(2\pi/M)}$, and Z^{-M} is replaced with $Z^{-M}\,e^{jkM(2\pi/M)}$. By design, the kMth multiple of $2\pi/M$ is a multiple of 2π for which the complex phase rotator term defaults to unity, or in our interpretation, aliases to base band (DC or zero frequency). The default to unity of the complex phase rotator occurs in each path of the M-path filter shown in Figure 6.14. The nondefault complex phase angles are attached to the delay elements on each of the M paths. For these delays, the terms Z^{-r} are replaced by the terms $Z^{-r}\,e^{jkr(2\pi/M)}$. The complex scalar $e^{jkr(2\pi/M)}$ attached to each path of the M-path filter can be placed anywhere along the path, and in anticipation of the next step, we choose to place the complex scalar after the down-sampled path filter segments $H_r(Z)$. This is shown in Figure 6.14.

The modification to the original partitioned Z-transform of (6.6) to reflect the added phase rotators of Figure 6.14 is shown in (6.8).

$$H(Z\,e^{-j\frac{2\pi}{M}k}) = \sum_{r=0}^{M-1} Z^{-r} e^{j\frac{2\pi}{M}rk} H_r(Z) \tag{6.8}$$

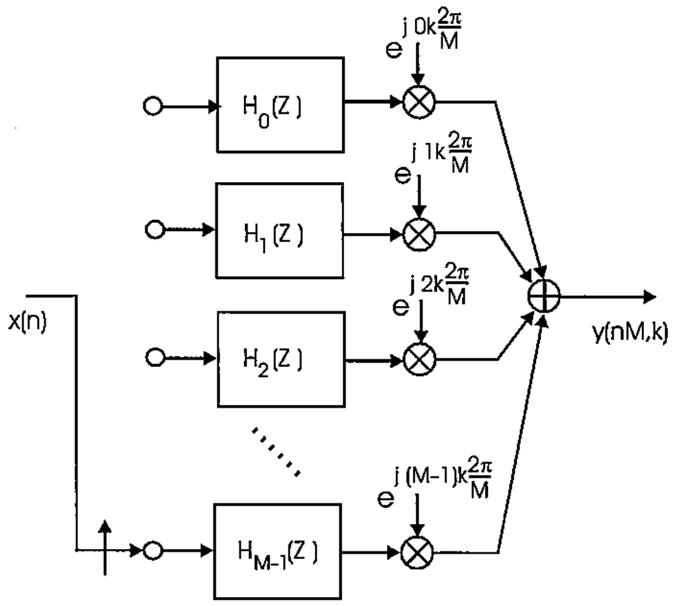


Figure 6.14 Resampling M-path Down Converter

The computation of the time series obtained from the output summation in Figure 6.14 is shown in (6.9). Here the argument nM reflects the down sampling operation which increments through the time index in stride of length M, delivering every Mth sample of the original output series. The variable $y_r(nM)$ is the nMth sample from the filter segment in the rth path, and y(nM,k) is the nMth time sample of the time series from the kth center frequency. Remember that the down-converted center frequencies located at integer multiples of the output sample frequency are the frequencies that alias to zero frequency under the resampling operation. Note the output y(nM,k) is computed as a phase coherent summation of the M output series $y_r(nM)$.

$$y(nM,k)=\sum_{r=0}^{M-1} y_r(nM)e^{j\frac{2\pi}{M}rk} \tag{6.9}$$

The M-path down-converter structure is known as a polyphase filter. We will shortly describe the property of the filter partition that is responsible for the name polyphase. We first take a side trip to examine an amazing property of the filter partition and the application of the noble identity. We have already noted that when we down sample by a factor of M-to-1, every multiple of the output sample rate aliases to baseband. In Figure 6.8 we used the band-centered digital filter to suppress all but one of the frequency bands that would alias to the base band. After successfully cleaning house, as it were, we were free to down sample and let the only surviving band-pass spectra alias to baseband without concern that signal components in other spectral bands would alias on top of our spectral band. During our development of the resampled M-path filter we blithely used the noble identity to interchange the order of filtering and down sampling. By interchanging the order of filtering and resampling to resampling and filtering, we have reduced the sample rate prior to the bandwidth reduction and have caused M-fold aliasing of the input spectrum. It seems that we have started the process by violating the Nyquist criterion. We have indeed violated the Nyquist criterion, and in fact have violated it M times at each of the M-commutator ports. We will show that the M-different realizations of the M-fold aliasing represent M-distinct equations with M-distinct aliasing terms, hence containing sufficient information to separate and remove the aliases. The polyphase filter structure performs the processing required to separate the aliases formed by the input resampling.

6.2 Separating the Aliases

Each of the M-time series observed at the ports of the input commutator represents M-fold aliased data. The aliases are the spectral terms from each multiple of the output sample rate, which, due to the resampling by the factor M-to-1, appear at baseband centered at DC. The alias from any particular center frequency exhibits a unique phase profile across the set of M-aliased time series. Prior to aliasing, the signal component at the kth center frequency (k 2π/M) exhibits an arbitrary phase angle, and at each successive data sample, the kth center frequency experiences a phase rotation of k 2π/M. In the same way a heterodyned signal preserves the phase of an input sinusoid, the spectral terms that alias to baseband when down sampled also preserve their phase. Thus as we progress up the successive ports of the input commutator, the kth alias exhibits successive phase increments of k 2π/M. Starting at the top of the commutator and counting down to the rth port we would find phase angles for the kth center frequency equal to –r k 2π/M. The phase rotators shown in (6.9) and in Figure 6.14 are seen to be the phase shift required to de-rotate and align the phase shifts from the kth center frequency seen at successive ports of the input commutator. When the sum of the phase aligned terms are formed in (6.9), the remaining aliasing terms, residing on the M-roots of unity, sum to zero and are destructively canceled during the summation. The destructive cancellation of the aliased terms is the process that separates the aliases formed by the resampling process that occurs at the input commutator.

This phase coherent sum is in fact, a DFT of the M-path outputs, which can be likened to beamforming the output of the path filters. The beam-forming perspective offers interesting insight to the operation of the resampled down-converter system we have just examined. The reasoning proceeds as follows: The commutator delivering consecutive samples to the M input ports of the M-path filter performs a down-sampling operation. Each port of the M-path filter receives data at one-Mth of the input rate. The down sampling causes the M-to-1 spectral folding, effectively translating the M-multiples of the output sample rate to the baseband. The alias terms in each path of the M-path filter exhibit unique phase profiles due to their distinct center frequencies and the time offsets of the different down sampled time series delivered to each port. These time offsets are in fact the input delays shown in Figure 6.11 and in (6.10). Each of the aliased center frequency experiences a phase shift shown in (6.10), equal to the product of its center frequency and the path time delay.

$$\begin{aligned}\phi(r,k) &= -\omega_k \Delta T_r \\ &= -2\pi \frac{f_S}{M} k\, r\, T_S \\ &= -2\pi \frac{f_S}{M} k\, r \frac{1}{f_S} = -\frac{2\pi}{M} r\, k\end{aligned} \tag{6.10}$$

The phase shifters of the DFT perform phase coherent summation, very much like that performed in narrow-band beamforming, extracting from myriad aliased time series, the alias with the particular matching phase profile. This phase sensitive summation aligns contributions from the desired alias to realize the processing gain of the coherent sum while the remaining alias terms, which exhibit rotation rates corresponding to the M roots of unity, are destructively canceled in the summation.

The inputs to the M-path filter are not narrowband, and phase shift alone is insufficient to effect the destructive cancellation over the full bandwidth of the undesired spectral contributions. Continuing with our beam-forming perspective, to successfully separate wideband signals with unique phase profiles due to the input commutator delays, we must perform the equivalent of time-delay beam forming. The M-path filters that are obtained by M-to-1 down-sampling of the prototype low pass filter, supply the required time delays. The M-path filters are approximations to all-pass filters, exhibiting, over the channel bandwidth, equal ripple approximation to unity gain and the set of linear phase shifts that provide the time delays required for the time-delay beam-forming task. The inputs to the polyphase paths are delivered sequentially with time offsets. The separate time series must be time aligned at the output of the polyphase filters in order to perform the summation required to compute the output samples. Figure 6.15 illustrates the task required of the path filters of the polyphase partition while Figure 6.16 illustrates the time alignment of the various path sequences for a commutated input series.

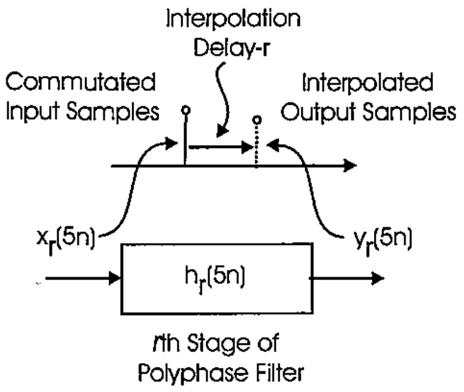


Figure 6.15 Time Delay Function Performed by Path Filter of M-path Polyphase Partition

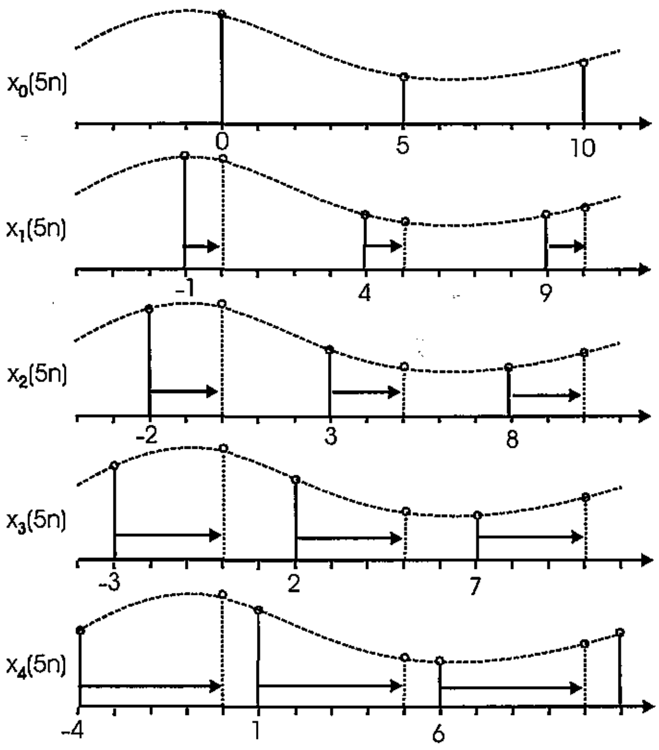


Figure 6.16 Time Alignment of Input Samples Performed by M-path Filter Stages

The filter achieves this time alignment process by virtue of the way we partitioned the low pass prototype. As shown in (6.6), we formed each of the M-path filters, filter $h_r(n)$ for instance, with weights h(r+nM) starting at an initial offset of r samples and then incrementing by stride of M samples. In the mapping of the polyphase partition, the r sample offset is suppressed, as the rth sample becomes the initial weight for the rth filter path. These initial offsets, unique to each path, are the source of the different linear phase shift profiles. It is for this reason, the different linear phase profiles, that the filter partition is known as a *polyphase* filter. The phase shift and group delay profiles for the filter weights of a 10-path polyphase filter are shown in Figures 6.17 and 6.18. Figures 6.19 and 6.20 present the phase profile of the weights h(r+nM) formed at the initial step of the polyphase partition. Here we have not yet dropped the (r–1) leading zeros nor the (M–1) intersample zeros so the weights still share a common time origin and offer the spectral replicates of the zero packing. We clearly see that in the successive Nyquist zones the 10-phase profiles have the same slope but exhibit phase offsets that are multiples of the Nyquist zone number k times $2\pi/10$. The phase rotators after the polyphase rotators access a selected Nyquist zone by canceling the phase offsets of that zone. Figure 6.20 is a set of subplots presenting zooms of the phase profiles showing the phase transition between the zones as well as the fixed phase offsets unique to that zone. These figures are part of the output suite of figures formed by the MATLAB m-file *filter_ten* available from the text's companion software website. This file synthesizes a 10-stage polyphase channelizer and presents input and output time series and spectra of the channelizers as well as various figures demonstrating characteristics of the 10-path filter.

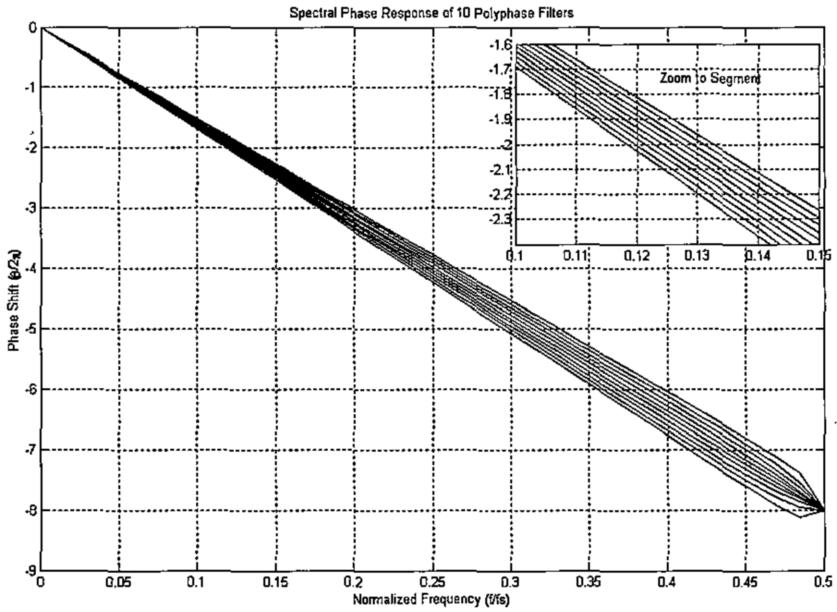


Figure 6.17 Phase Profiles for 10-paths of 10-path Polyphase Partition

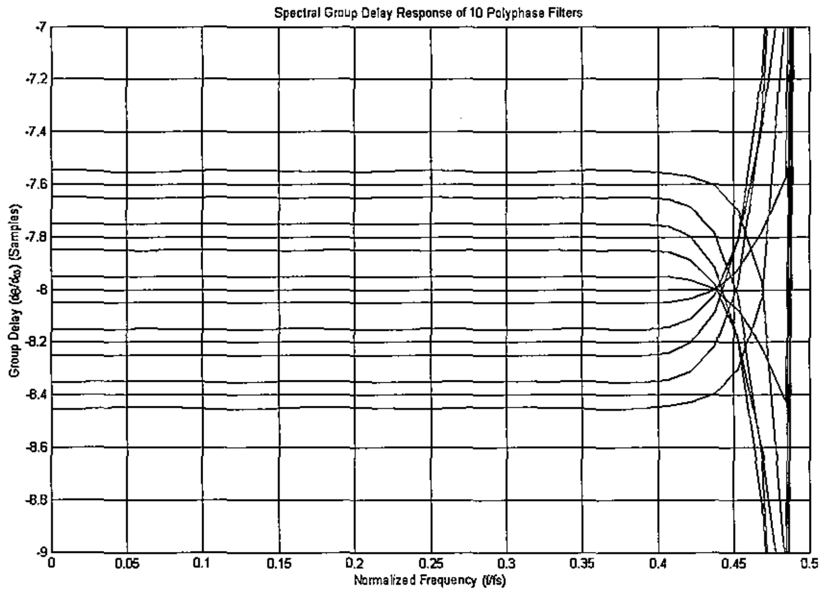


Figure 6.18 Group Delay Profiles for 10-paths of 10-path Polyphase Partition

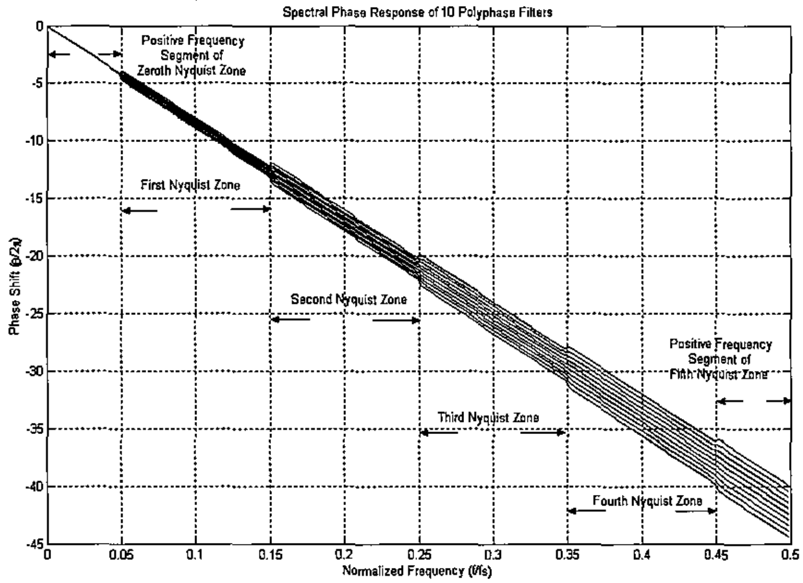


Figure 6.19 Phase Profiles of 10 Separate Paths of 10-path Polyphase Partition

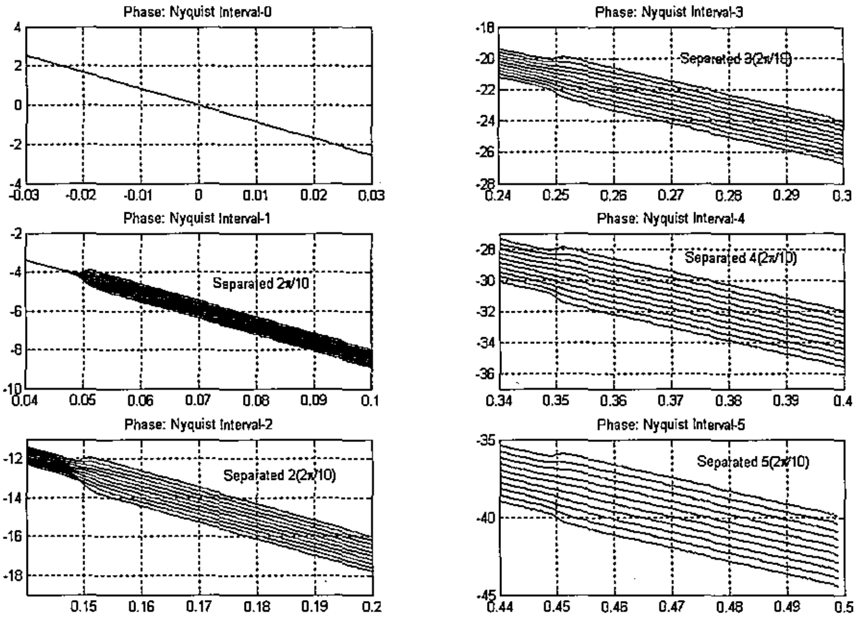


Figure 6.20 Details of Phase Profiles of 10-paths of 10-path Polyphase Partition

A useful perspective is that the phase rotators following the filters perform phase alignment of the band center for each aliased spectral channel or Nyquist band while the polyphase filters perform the required differential phase shift across these same channel bandwidths to compensate for the offset time origins of the commutated data streams delivered to each path. When the polyphase filter is used to down convert and down sample a single channel, the phase rotators are implemented as external complex products following each path filter. When a small number of channels are being down converted and down sampled, appropriate parallel sets of phase rotators can be applied to the filter stage outputs and summed to form each channel output. We take a different approach when the number of channels becomes sufficiently large. Here sufficiently large means on the order of $\log_2(N)$. Since the phase rotators following the polyphase filter stages are the same as the phase rotators of a DFT, we can use the DFT to simultaneously apply the phase shifters for all of the channels we wish to extract from the aliased signal set. This is the structure shown in Figure 6.21. This is reminiscent of phased array beam forming. For computational efficiency, the FFT (actually an IFFT) algorithm implements the DFT. Additional material in this vein will be found in the section describing multichannel receivers.

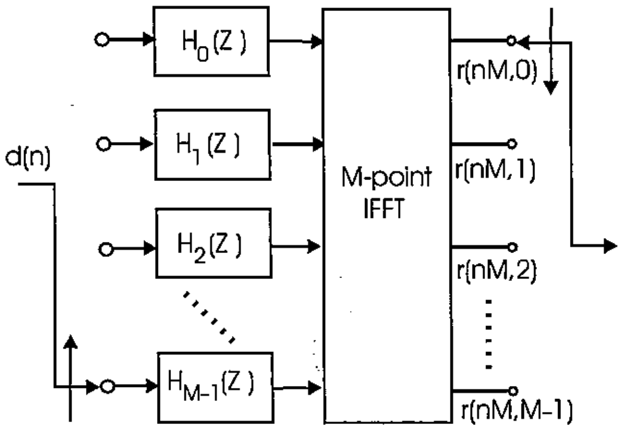


Figure 6.21 Polyphase Filter Bank: M-path Polyphase Filter and M-point IFFT

It is instructive to compare the conventional mixer down converter and the resampled polyphase down converter. The input to either process can be real or complex. In the mixer down-converter model, a separate mixer pair and filter pair must be assigned to each channel of the channelizer and these mixers and filters all operate at the high input data rate, prior to down sampling. By way of contrast, in the resampled polyphase there is only one low pass filter required to service all the channels of the channelizer, and this single filter accommodates all the channels as co occupying alias contributors of the baseband bandwidth. This means that all the processing performed in the resampled polyphase channelizer occurs at the low output sample rate. When the input signal is real, there is another significant difference between the two processes. In the mixer downconverter model the signal is made complex by the input mixers as we enter the filtering process, which means that the low-pass filtering task requires two filters, one for each of the quadrature components, while in the resampling channelizer the signal is made complex by the phase rotators as we leave the process. Consequently we require only one partitioned low pass filter to process the real input signal.

Let us summarize what we have accomplished to this point. The commutator performs an input sample rate reduction by commutating successive input samples to selected paths of the M-path filter. Sample rate reduction occurring prior to any signal processing causes spectral regions residing at multiples of the output sample rate to alias to baseband. This desired result allows us to replace the many down converters of a standard channelizer, implemented with dual mixers, quadrature oscillators, and bandwidth reducing filters, with a collection of trivial aliasing operations performed in a single partitioned and resampled filter.

The partitioned M-path filter performs the task of aligning the time origins of the offset sampled data sequences delivered by the input commutator to a single common output time origin. This is accomplished by the all-pass characteristics of the M-path filter sections that apply the required differential time delay to the individual input time series. The DFT performs the equivalent of a beam-forming operation; the coherent summation of the time-aligned signals at each output port with selected phase profiles. The phase coherent summation of the outputs of the M-path filters separate the various aliases residing in each path by constructively summing the selected aliased frequency components located in each path, while simultaneously destructively canceling the remaining aliased spectral components.

This section introduced the structure of a process that can perform down conversion of one or more channels using a polyphase filter. A similar exposition can be mounted for a process that up converts one or more channels with a polyphase filter. We discuss both options in detail in a later chapter. For now it suffices to state that the up-converter process is the dual process of the down-converter process. The dual process simply reverses all signal flow of the original process. In the dual structure, we enter the channel process at the phase rotators (or DFT) and leave the process by the polyphase commutator. Reversing the signal flow results in a process that up samples and up converts rather than one that down converts and down samples.

References

Crochiere, Ronald, and Lawrence Rabiner, *Multirate Signal Processing*, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1983.

Fliege, Norbert, *Multirate Digital Signal Processing: Multirate Systems, Filter Banks, Wavelets*, West Sussex: John Wiley & Sons, Ltd., 1994.

Harada, Hiroshi and Ramjee Prasad, *Simulation and Software Radio for Mobile Communications*, Norwood, MA, Artech House, 2002.

Hentschel, Tim, *Sample Rate Conversion in Software Configurable Radios*, Norwood, MA, Artech House, 2002.

Jovanovic-Dolecek, Gordana, *Multirate Systems: Design and Applications*, London, Idea Group, 2002.

Lutovac, Miroslav, Dejan Tošić and Brian Evans, *Filter Design for Signal Processing: Using MATLAB and Mathematica*, Upper Saddle River, NJ, Prentice Hall, Inc, 2001.

Mitra, Sanjit and James Kaiser, *Handbook for Digital Signal Processing*, New York, John Wiley & Sons, 1993.

Mitra, Sanjit, *Digital Signal Processing: A Computer-Based Approach*, 2nd ed., New York, McGraw-Hill, 2001.

Paul Nahin, *The Science of Radio*, 2nd ed. New York, Springer-Verlag, 2001.

Vaidyanathan, P. P., *Multirate Systems and Filter Banks*, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1993.

Problems

6.1 Design a low pass FIR filter satisfying the following specifications:

Sample Rate:	100 kHz		
Pass Band:	0-4 kHz	In-band Ripple	0.1-dB
Stop Band:	6-50 kHz	Stop band Atten.	60-dB

Form an input sequence as 2,000 samples of a real sinusoid at frequency 32.0 kHz with random phase. Use a complex heterodyne centered at 30 kHz to down convert the input signal and present the down-converted data to the baseband filter designed previously. Form subplots of the 10-to-1 down-sampled time series from the filter pair and the log magnitude windowed spectrum of the same time series.

Now up convert the baseband low pass filter designed earlier to form a complex band-centered filter centered at 30 kHz. Pass the input signal through the complex filter and down convert the complex output with the 30 kHz heterodyne. Form subplots of the 10-to-1 down-sampled time series from the sequence following the heterodyne and the log magnitude windowed spectrum of the same time series.

Finally, form a 10-to-1 down-sampled time series without the complex heterodyne. Form subplots of this 10-to-1 down-sampled time series and the log magnitude windowed spectrum of the same time series.

Compare the three time series and their corresponding spectra. They should be the same signal and have the same spectra. We are using this exercise to illustrate the equivalency theorem.

6.2 Consider a trivial filter corresponding to the Z-transform shown here:

$$H(Z)=1+Z^{-10}+Z^{-20}+Z^{-30}+Z^{-40}$$

Form subplots of the impulse response and magnitude spectrum of this filter.

Now down sample the impulse response by 10-to-1 and form subplots of the down sampled impulse response series and its magnitude spectrum.

Apply 500 samples of a real sinusoid at normalized frequency 0.11 to the original filter and then plot the time series of the 10-to-1 down-sampled output. Also pass the 10-to-1 down-sampled input series through the 10-to-1 down-sampled filter and plot the output time response. Compare the two time series and comment on the results.

6.3 Design a low pass FIR filter satisfying the following specifications with the filter length increased to the next multiple of 10:

Sample Rate:	100 kHz		
Pass Band:	0–4 kHz	In-band Ripple	0.1-dB
Stop Band:	6–50 kHz	Stop band Atten.	60-dB

Perform a 10-path polyphase partition of this impulse response and form and plot on a single subplot the magnitude frequency response of all 10 paths. Also form and plot on a single subplot the unwrapped phase (angle) response of all 10 paths.

Comment on what is the same and what is different about the ten overlaid magnitude and phase response plots.

6.4 Design a low pass FIR filter satisfying the following specifications with the filter length increased to the next multiple of 10:

Sample Rate:	100 kHz		
Pass Band:	0–4 kHz	In-band Ripple......	0.1-dB
Stop Band:	6–50 kHz	Stop band Atten.	60-dB

Perform a 10-path polyphase partition of this impulse response and convolve each path of the partition with the Gaussian sequence exp(5*(–0.5:0.1:0.5).*(–0.5:0.1:0.5)).

On 10 subplots plot the input and output series from the 10 convolutions. The successive time series represent the time-shifted samples of the input signal. Can you explain the amount of time shift exhibited by each output time series?

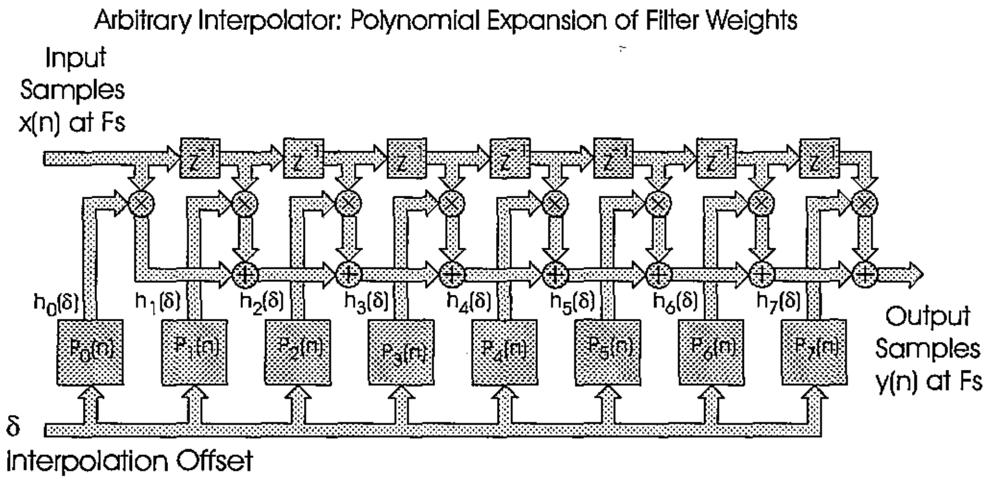
6.5 Design a low pass FIR filter satisfying the following specifications with the filter length increased to the next multiple of 10:

Sample Rate:	100 kHz		
Pass Band:	0–4 kHz	In-band Ripple	0.1-dB
Stop Band:	6–50 kHz	Stop band Atten.	60-dB

Perform a 10-path polyphase partition of this filter and follow the path outputs with a 10-point FFT. Generate a 2000 sample input time series comprising the sum of two complex sinusoids at frequency 10.5 kHz and at –22 kHz. Feed the polyphase filter with 10 input samples per input cycle and compute the 10 separate outputs to form the 10 time series from the 10 polyphase paths. Transform the time series from two paths and identify the spectral aliases. Now perform the FFT on the polyphase vector sequences to form the 10 phase corrected time series that correspond to the channelized filter bank. Plot the real part and the imaginary part of each time series in a set of subplots. Identify the transients associated with the unoccupied channels and the time responses of the two occupied channels. Also form and plot the windowed FFT of the 10 time series. Identify the spectral lines that correspond to the two input spectral lines in the occupied channels as well as the suppressed copies of these lines in the unoccupied channels.

CHAPTER 7

Resampling Filters



In this chapter we discuss variants of the interpolation process and present a number of applications dependent on the process. The simplest such process is a 1-to-M up sampler where M is an arbitrary integer. Many applications require up sampling by a small integer in the range 4-to-10 but the integer M may be in the range of hundred to thousands. When M is a large nonprime integer, the up sampling is performed by a sequence of low-order up-sampling filters. The next most common interpolation process is in interpolation by a rational ratio P/Q where both P and Q can be arbitrary integers but most often are small integers. A common interpolation task is one that requires arbitrary resampling including slowly varying time-varying resampling. The later examples are very common in modern digital receivers. The filters participate in the timing recovery process by interpolating input sample values to offset output time samples. We will present these examples and others as well as present mechanisms for control of the time-varying interpolation process.

7.1 INTERPOLATORS

My Webster's Second Collegiate Dictionary lists, in its third entry, a math definition of *interpolate* as "To estimate a missing functional value by taking a weighted average of known functional values at neighboring points." Not bad, and that certainly describes the processing performed in a multirate filter. Interpolation is an old skill that many of us learned before the advent of calculators and key strokes replaced tables of transcendental functions such as log(x) and the various trigonometry functions. Take for example the NBS Applied Mathematics Series, AMS-55, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, by Abramowitz and Stegan. This publication contains tables listing functional values of sin(θ) for values of θ equal to 40.0, 40.1, 40.2, etcetera. Interpolation is required to determine the value of sin(θ) for the value of θ equal to 40.137. Interpolation was such an important tool in numerical analysis that three pages in the introduction of the handbook are devoted to the interpolation process. Interpolation continues to be an important tool in signal processing, and we now present the DSP description of the interpolation process.

7.1.1 Simple 1-to-M Interpolator

A very common interpolator task is raising the sample rate of a sampled data sequence that is already oversampled by a factor of 2 or by a factor of 4. This happens, for instance, when the original data is the output of a shaping filter in a modulator in which the sample rate is increased relative to the symbol rate to accommodate the excess bandwidth of the shaping filter. The interpolator following the original shaping filter is used to raise the sample rate so that the first spectral translation to an intermediate frequency is performed in the DSP domain rather than in the analog domain. To compensate for sinc response of the output DAC the up converter is followed by a $[\sin(x)/(x)]^{-1}$ predistortion filter. A block diagram of such a system is shown in Figure 7.1.

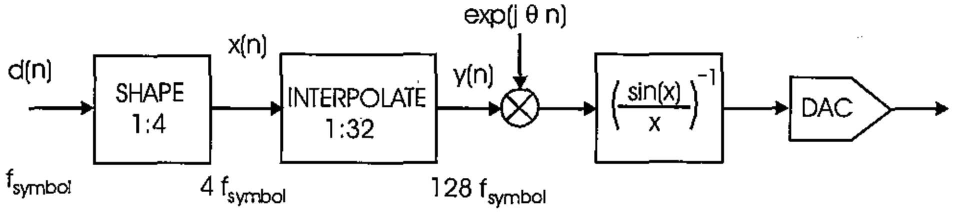


Figure 7.1 Example of Interpolator Following Initial Time Series Generator

Figure 7.2 presents a typical output spectrum at the output of the shaping filter. In this example we can assume a square root Nyquist filter with 50 percent roll off with side-lobe levels held to –60-dB. We will use normalized frequency, with the reference being the symbol rate which we can call "1" or, if we require a typical value, we will use 100 kHz. The two-sided bandwidth in this example is 1.5 or 150 kHz, and the sample rate is 4 or 400 kHz. We now examine the design, properties, and performance of the 1-to-32 up sampler that is to raise the sample rate to 128 or to 12.8 MHz.

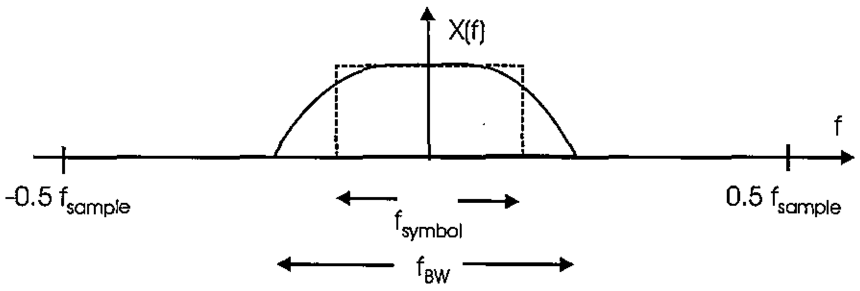


Figure 7.2 Spectrum at Output of Shaping Filter with Embedded 1-to-4 Up Sampler

Figure 7.3 presents the periodic spectrum of the input sampled data signal with an overlay approximating the frequency response of the interpolating filter that is designed to suppress the spectral replicates. The performance specifications required of this filter are listed in Table 7-1.

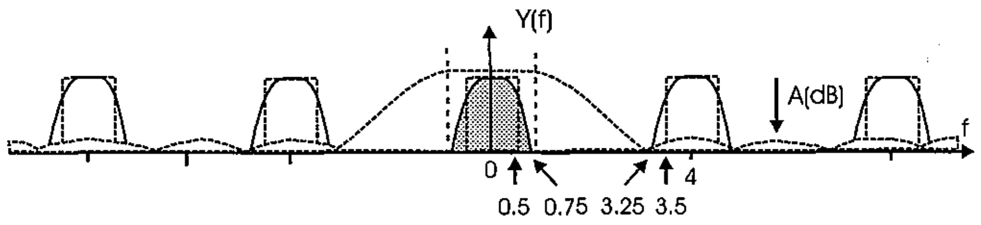


Figure 7.3 Periodic Input Spectrum and Filter Response of 1-to-32 Interpolator Filter

Table 7-1 Parameters Required for Interpolating Filter

Parameters	**Values**
Pass band Ripple	0.1-dB
Stop band Attenuation	60-dB
Pass band Frequencies	0-to-0.75 (0-to-75 kHz)
Stop band Frequencies	3.25-to-64 (325 kHz-to-6.4 MHz)
Sample Frequency	128 (12.8 MHz)

An estimate of the number of filter taps required to implement the 1-to-32 interpolating filter is obtained from standard estimate rules or algorithms. One such rule, shown in (7.1), leads to the estimate shown in (7.2).

$$N \cong \frac{fs}{\Delta f} K(\delta_1, \delta_2) \tag{7.1}$$

$$N \cong \left(\frac{128}{3.25 - 0.75}\right) 2.5 = 128 \tag{7.2}$$

An estimate for the filter length obtained by a call to the MATLAB script file *remezord* is shown next.

```
NN=remezord([0.75 3.25],[1 0],[0.01 0.001],128)
```

The response to this call is:

```
NN = 130
```

The two estimates of filter length are comparable, but the actual filter length must be verified by applying the Remez algorithm to the design task. The filter length required to meet the ripple specifications was found to be 140. We have the option of modifying the pass band and stop band frequency edges to widen the transition band and hopefully meet the requirements with a 128-point filter. We are permitted to do this because the input spectrum at the stop band edge has significantly less energy than at the band center and thus does not require as much attenuation to meet the desired spectral mask. Similarly the filter rolls off very slowly at the pass band edge and since, the input signal has very little spectral energy at this edge, there is essentially no signal degradation if the pass band edge overlaps the input signal's transition band edge.

It would be desirable to have the filter length be 128 taps so that the 32-polyphase stages would all be of length 4 taps. This requirement assures that all stages are the same length. To enable the 128 tap filter to meet the attenuation specifications the band edge specifications of the filter were modified slightly. This entailed allowing overlap of the transition band and the first spectral replicate. By trial and error we found that widening the transition band by moving the stop band edge from 3.3 to 3.5 the 128 point filter would meet the required stop band attenuation. The filter specifications were also modified to obtain a non-equiripple stop band by the use of a frequency-dependent penalty sequence in the penalty vector of the Remez algorithm.

Figure 7.4 presents the spectrum of the 1-to-32 interpolating filter overlaid on the replicated spectrum of the shaping filter over the full bandwidth and a zoomed bandwidth to show stop band detail. Here we plainly see the overlap of the filter's transition band with the adjacent channel. Figure 7.5 presents the result of the interpolation process with this interpolation filter. Here we see that the adjacent spectral copies are suppressed by at least 60-dB and that the spectra residing in the overlapped transition band are also attenuated by 60-dB.

Using an alternate strategy, we can redefine the interpolating filter to have multiple stop bands of width $\pm$ 0.7 bracketing the periodic input spectra centered at multiples of 4. We treat the intervals between the stop bands as don't care regions. These regions are permitted since we know that they contain no significant spectral content by design of the oversampled-by-4 shaping filter. Since there is no need to suppress the spectral interval between the periodic spectra we can release the stop band zeros normally assigned to this region and permit the Remez algorithm to use them in the desired stop band regions. This filter exhibits additional attenuation and smaller in-band ripple than the previous filter design with full stop band attenuation. The spectra of the filter designed in this manner and of the filtered data obtained with this filter are shown in Figures 7.6 and 7.7

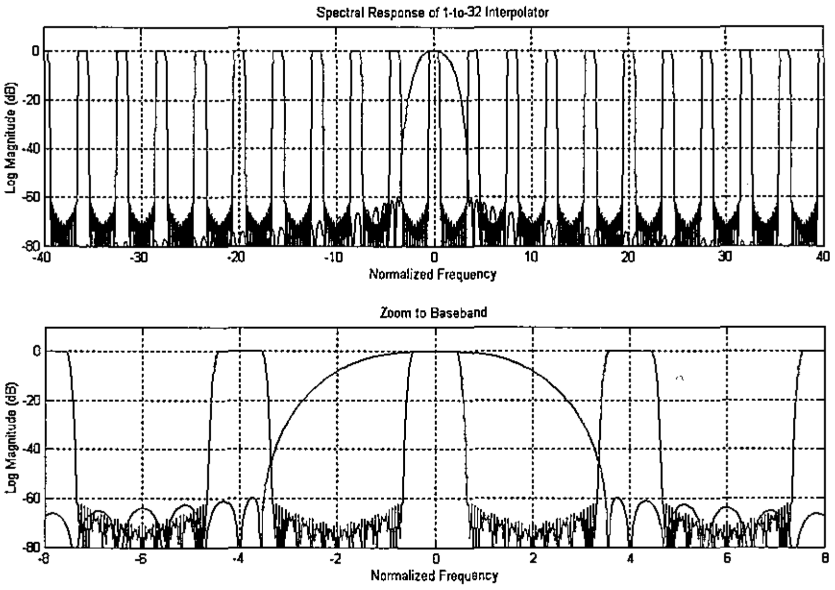


Figure 7.4 Spectrum of Interpolator and Periodic Spectrum of Zero-packed Shaping Filter

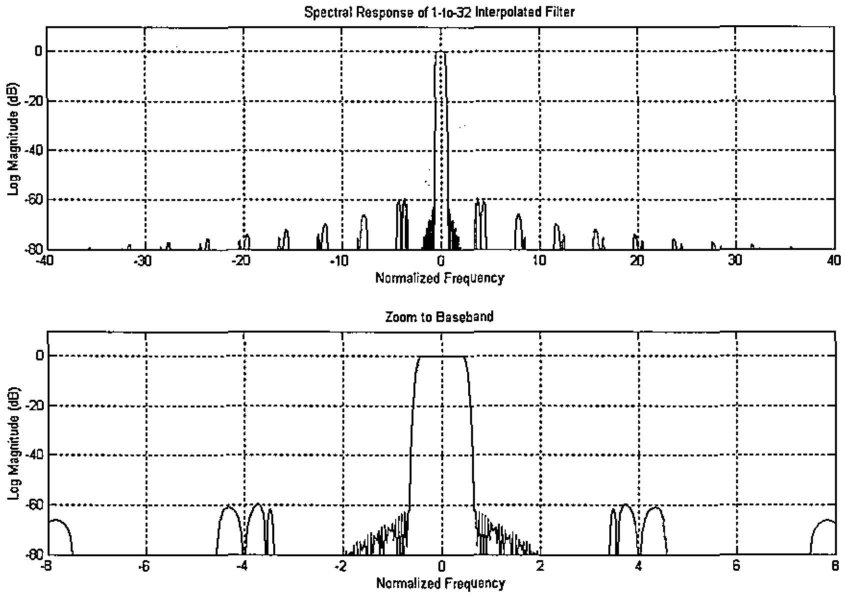


Figure 7.5 Spectrum of 1-to-32 Interpolated Shaping Filter

Figures 7.4 and 7.6 present the frequency responses of the interpolator filter designed with a single stop band and then with multiple stop bands interspersed with don't care bands. As mentioned, use of the don't care bands permitted the design routine to obtain improved suppression in the spectral regions containing the spectral replicates. An unexpected consequence of the design option using the don't care regions is the change in the filter's impulse response. Figure 7.8 presents the impulse response of the two versions of the interpolator filter whose spectra appeared in Figures 7.4 and 7.6. We call the impulse response of the second design the Tibetan hat filter. The outlier samples that appear in this impulse response are reminiscent of the end-point outliers generated by the Remez algorithm designs for filters with equiripple stop band. These outliers present no problem to the implementation of the interpolator. They simply map to the first or last segments of the 32-stage polyphase partition. We only called attention to the impulse response because of its unusual shape.

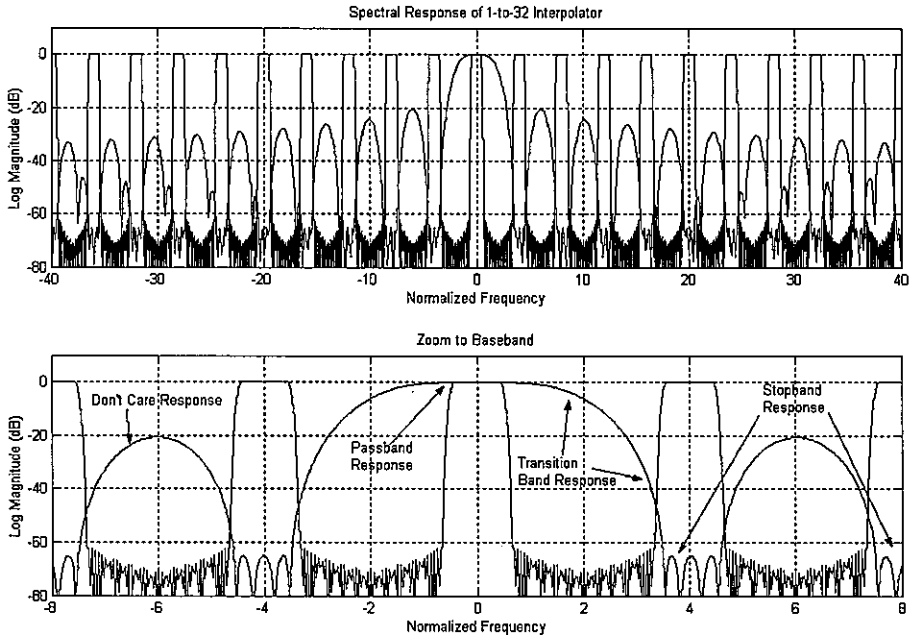


Figure 7.6 Spectrum of Alternate Interpolator and Periodic Spectrum of Zero-packed Shaping Filter

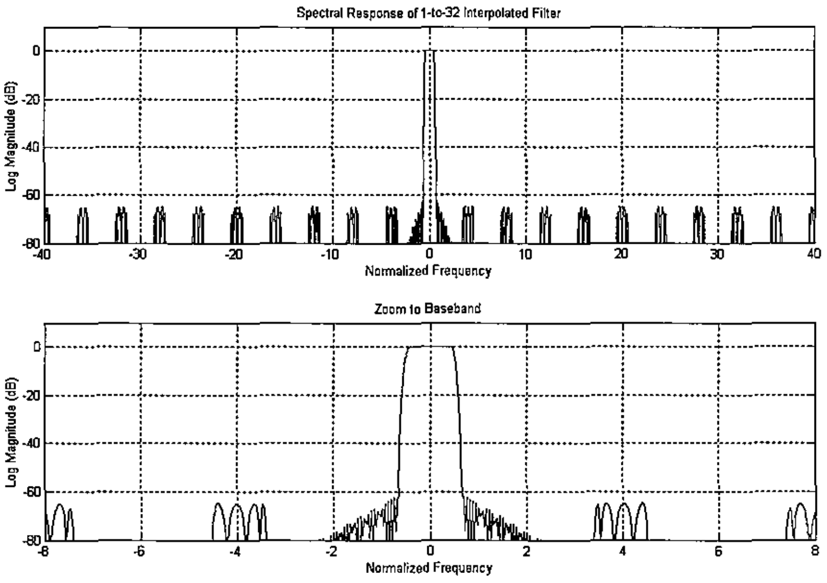


Figure 7.7 Spectrum of Alternate 1-to-32 Interpolated Shaping Filter

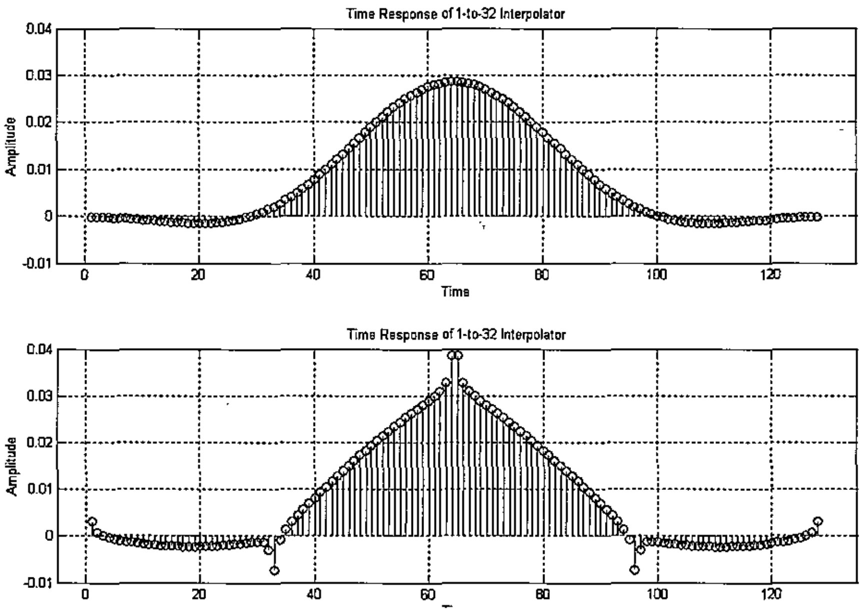


Figure 7.8 Impulse Response of Interpolator Filter with Single Stop Band and with Multiple Stop Bands with Don't Care Bands

7.2 INTERPOLATOR ARCHITECTURE

We now examine the architectural structure of the interpolator derived in the previous section. The initial model of the 1-to-32 interpolator is a 1-to-32 up sampler followed by an appropriate low pass filter. In this configuration, the up sampler converts the Nyquist interval, the observable frequency span, from the input sample rate to a span 32 times as wide, the output sample rate. The 1-to-32 up sampler zero packs the input series, effectively decreasing the distance between input samples without modifying the spectral content of the series. The wider Nyquist interval, spanning 32 input Nyquist intervals, presents 32 spectral copies of the input spectrum to the FIR filter. The amplitude of each of the 32 copies is 1/32 of the amplitude of the input signal spectrum When a FIR low pass filter is scaled so that the peak coefficient is unity, the filter exhibits a processing gain inversely proportional to its fractional bandwidth. Thus, as the filter eliminates the 31 spectral copies, reducing the bandwidth by a factor of 1/32, the filter gain precisely compensates for the attenuation of the input spectra due to the zero packing of the input series. This structure is shown in Figure 7.9.

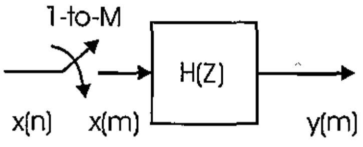


Figure 7.9 Initial Structure of 1-to-32 Interpolator

7.2.1 Polyphase Partition

The zeros of the zero-packed input time series do not contribute to the weighted sums formed at the filter output. Since they do not contribute, there is no need to perform the product sum from the input data registers containing the known zero-valued samples. Since only the non-zero packed samples contribute to the filter output, we can track their location in the filter and perform the weighted sum only from the register locations containing these samples. These locations are separated by 32 sample values, and their position shifts through the filter as each new zero-valued input is presented to the input of the filter. Keeping track of the coefficient stride and the position of each coefficient set is automatically performed by the polyphase partition of the filter, as shown next. Equation (7.3) is the Z-transform of the standard FIR filter structure representing a set of delayed filter coefficients. Equation (7.4) is the polyphase partition of the same filter representing the filter as a sum of successively delayed subfilters with coefficients separated by the stride of M samples. Finally, (7.5) is a compact representation of (7.4) where the rth stage $H_r(Z^M)$ of the polyphase filter is formed by the coefficient set that starts at index r and increments in steps of length M.

$$H(Z) = \sum_{n=0}^{N-1} h(n) Z^{-n} \tag{7.3}$$

$$H(Z) = \sum_{r=0}^{M-1} Z^{-r} \sum_{n=0}^{\frac{N}{M}-1} h(r+nM)\, Z^{-nM} \tag{7.4}$$

$$H(Z) = \sum_{r=0}^{M-1} Z^{-r} H_r(Z^M) \tag{7.5}$$

The form of the filter in the M-path polyphase partition is shown in Figure 7.10. This structure enables the application of the noble identity in which we slide the resampler through the filter and replace the M units of delay at the output clock rate with one unit of delay at the input clock rate. This form of the filter is shown in Figure 7.11. Note that the resampler cannot slide through the delays Z^{-r} following each filter segment H_r.

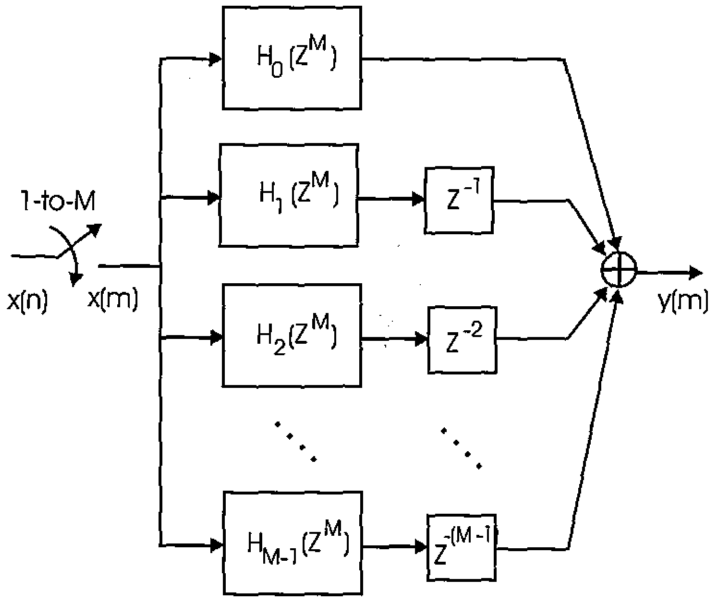


Figure 7.10 Initial Structure of 1-to-M Polyphase Interpolator

The resamplers following the separate filter stages up sample each time series by a factor of M, and the delays in each arm shift each resulting time series a different time increment so that only one non-zero time sample is presented to the summing junction at each output time. Thus, rather than perform the sum with multiple zeros, we can simply point to the arm that sequentially supplies the non-zeros sample. The output commutator in Figure 7.12 performs this selective access.

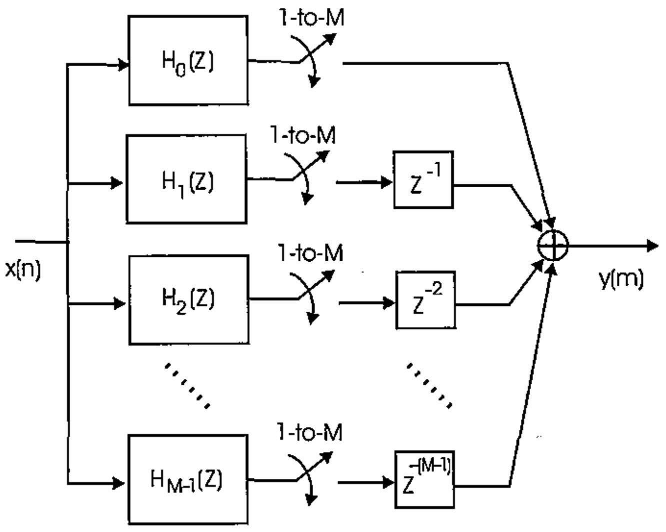


Figure 7.11 Transition Structure of 1-to-M Polyphase Interpolator

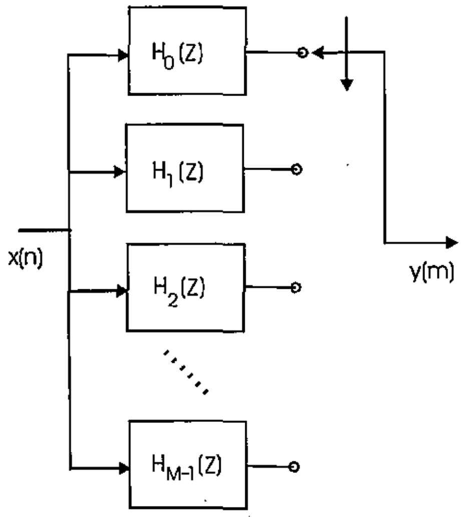


Figure 7.12 Standard Structure of 1-to-M Polyphase Interpolator

We note the polyphase arms contribute their outputs one at a time as the commutator points to successive output ports. We also note that the separate filters all contain the same input data and differ by only their unique coefficient sets. We can replace the M-path version of the polyphase filter with a single stage filter with M-coefficient sets that are sequentially presented to the filter to compute successive outputs. This structure is shown in Figure 7.13. What we have accomplished here is first, move the input commutator that performed the input zero packing to the output of the filter where it selected successive filter outputs and, second, move the commutator to coefficient memory where it selected successive coefficient sets rather than filter outputs.

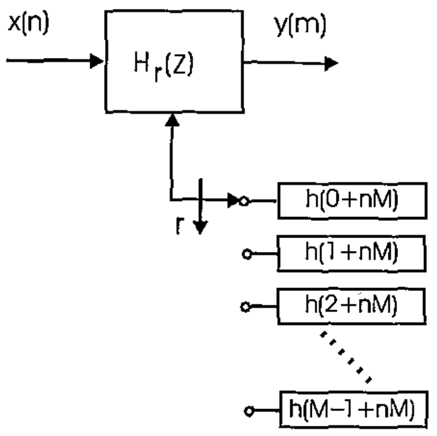


Figure 7.13 Efficient Implementation Structure of 1-to-M Polyphase Interpolator

7.3 Band-Pass Interpolator

Examining Figure 7.1, we recall that the purpose of the interpolation process was to raise the input sample rate to allow the translation of the input spectrum to an intermediate frequency (IF), and then output the digital IF signal via a single DAC. This option reduces system cost by using a single DAC and band-pass filter to replace the standard baseband process requiring matched DACs, matched low pass filters, matched balanced mixers, and a quadrature oscillator to form the first IF frequency band. For completeness the baseband version of the modulator structure comparison is shown in Figure 7.14, and the digital IF version is shown in Figure 7.15.

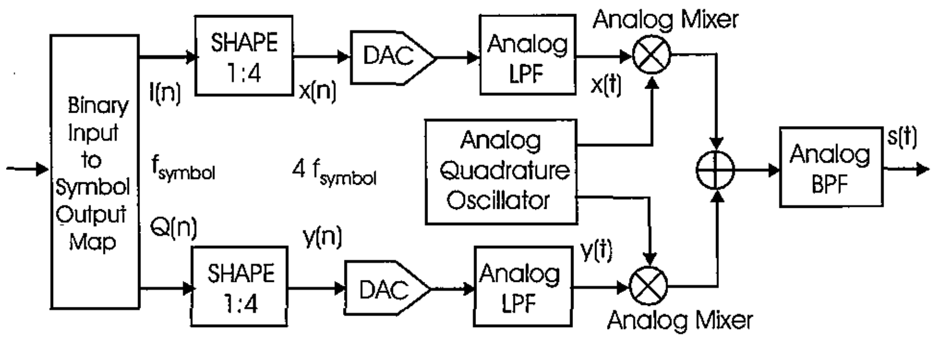


Figure 7.14 Baseband Modulator with Dual Analog Components to Be Replaced by Digital Components in Digital IF Modulator

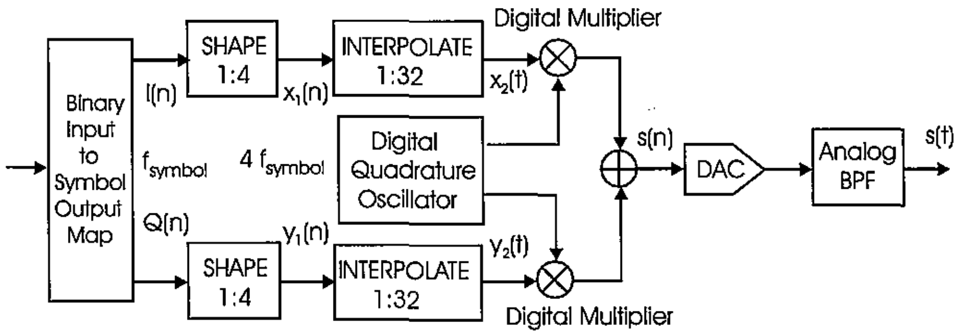


Figure 7.15 Digital Baseband and IF Modulator with Minimum Analog Components

The first model of the interpolating process zero packed the input data to give us access to 32 spectral copies of the input time series. The spectral copies reside at multiples of the input sample rate, at normalized frequencies 0, 4, 8, 12, ..., 60, and 64. The low pass filter rejected the spectral copies, retrieving the baseband copy at frequency 0 that was then passed on to the digital up converter for translation to the desired center frequency. It is possible to perform the spectral translation as part of the interpolation process when the desired center frequency coincides with one of the multiples of the input sample rate. Rather than extracting the spectral copy at baseband from the replicated set of spectra we can directly extract one of the spectral translates by using a band-pass filter as opposed to a low pass filter. The band-pass filter is simply an up converted version of the low pass filter with weights shown in (7.6). Here the center frequency is interpreted as the kth multiple of 1/Mth of the output sample rate.

$$\text{Low Pass:}\quad h(n)$$
$$\text{Band Pass:}\quad g(n,k) = h(n)\exp(j\frac{2\pi}{M}kn) \tag{7.6}$$

The Z-transform of the two filters, the low pass and the band pass, are shown in (7.7).

$$\text{Low Pass:}\quad H(Z) = \sum_{n=0}^{N-1} h(n)Z^{-n}$$
$$\text{Band Pass:}\quad G_k(Z) = \sum_{n=0}^{N-1} h(n)\exp(j\frac{2\pi}{M}nk)Z^{-n} \tag{7.7}$$

The Z-transform of the polyphase versions of the two filters is shown in (7.8). Here we see that the length M stride in coefficient index due to the 1-to-M resampling aliases the phase rotators in the polyphase filter stages to DC and hence have no effect on the polyphase weights. The phase rotator does, however, have a contribution related to the delay associated with each arm of the partition. The output commutator process absorbs the delays while the phase rotators are applied to each arm to obtain the spectral translation as part of the interpolation.

Low Pass:

$$H(Z) = \sum_{r=0}^{M-1} Z^{-r} \sum_{n=0}^{\frac{N}{M}-1} h(r+nM)\, Z^{-nM}$$

Band Pass:

$$G_k(Z) = \sum_{r=0}^{M-1} Z^{-r} \exp(j\frac{2\pi}{M}r\,k) \sum_{n=0}^{\frac{N}{M}-1} h(r+nM)\exp(j\frac{2\pi}{M}n\,M\,k)\, Z^{-nM}$$
$$= \sum_{r=0}^{M-1} Z^{-r} \exp(j\frac{2\pi}{M}r\,k) \sum_{n=0}^{\frac{N}{M}-1} h(r+nM)\, Z^{-nM} \tag{7.8}$$

The filter that accomplishes the task of simultaneous interpolation and up conversion to the kth Nyquist zone is shown in Figure 7.16.

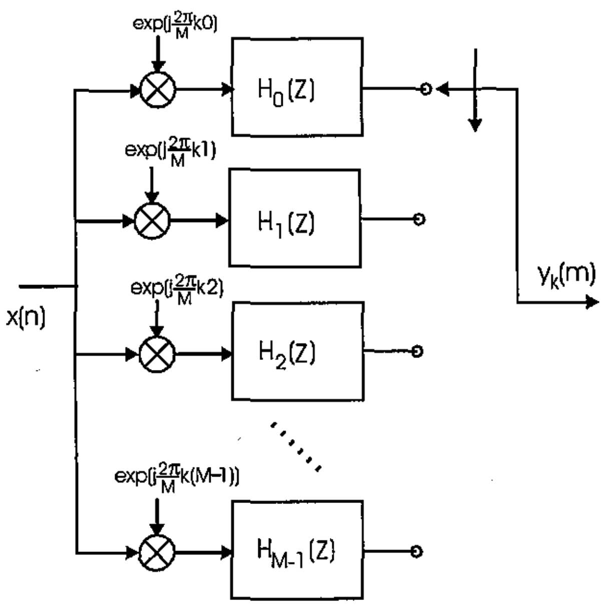

$\exp(j\frac{2\pi}{M}k0)$
$\exp(j\frac{2\pi}{M}k1)$
$\exp(j\frac{2\pi}{M}k2)$
$\exp(j\frac{2\pi}{M}k(M-1))$
$H_0(Z)$
$H_1(Z)$
$H_2(Z)$
$H_{M-1}(Z)$
x(n)
$y_k(m)$

Figure 7.16 Structure of 1-to-M Polyphase Interpolator with Phase Rotators for Selecting Spectrum Centered in Mth Nyquist

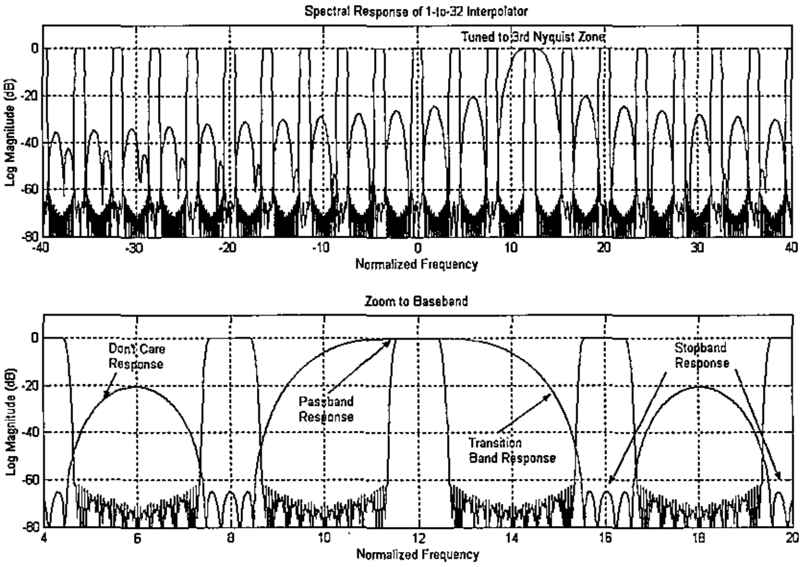


Figure 7.17 Spectrum of Translated Interpolator and Periodic Spectrum of Zero-packed Shaping Filter

Figure 7.17 presents the spectrum of the 1-to-32 interpolating filter with the phase rotators tuned to the third Nyquist zone. This spectrum is overlaid on the replicated spectrum of the shaping filter over the full bandwidth and a zoomed bandwidth to show stop band detail. Here we plainly see the phase rotators have positioned the spectrum of the filter to recover the third Nyquist zone centered at normalized frequency 12. Figure 7.18 presents the result of the interpolation process with this interpolation filter. Here we see that the band passed by the filter is the third Nyquist zone and that the channels adjacent to this zone are suppressed by the filtering action. We have an interest only in the real part of the time series associated with the spectrum shown in Figure 7.18. The spectrum shown in Figure 7.18 represents the most general case of processing complex input data with a complex filter requiring four convolutions as shown in (7.9). The real part is generated with only two convolutions as shown in (7.10), which is also indicated in the block diagram of Figure 7.15.

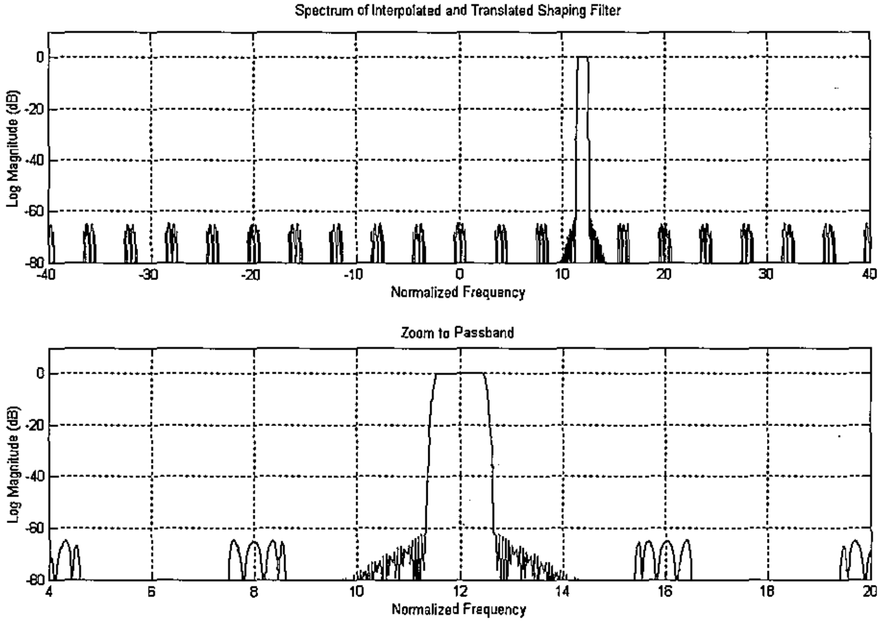


Figure 7.18 Spectrum of 1-to-32 Interpolated and Translated Shaping Filter

$$\begin{aligned} d_{IF}(n) &= [x_{in}(n) + j\,y_{in}(n)] * [h_{RL}(n) + j\,h_{IM}(n)] \\ &= [x_{in}(n) * h_{RL}(n) - y_{in}(n) * h_{IM}(n)] + j\,[x_{in}(n) * h_{IM}(n) - y_{in}(n) * h_{RL}(n)] \end{aligned} \tag{7.9}$$

$$\text{Real}[d_{IF}(n)] = [x_{in}(n) * h_{RL}(n) - y_{in}(n) * h_{IM}(n)] \tag{7.10}$$

7.4 Rational Ratio Resampling

We know how to up sample a time series by any integer. This integer is usually small, say up 4, or up 8, and in the example examined in the previous section, up 32. There are occasions that the desired resampling ratio is a ratio of integers, say up P and down Q for an increase in sample rate of P/Q. While there is no restriction on the values of the two integers P and Q, they are often selected to small integers such as up 5 and down 3. We will use the 5/3 ratio as an example to demonstrate rational ratio resampling. The resampling process can be visualized as a two-step process. In the first step we raise the sample rate by a factor of 5 by a standard 5-path polyphase filter. In the second step we reduce the new sample rate by a factor of 3-to-1 with a resampling switch. In this process for every 3 input samples we extract 5 output samples. This straightforward approach is shown conceptually in Figure 7.19.

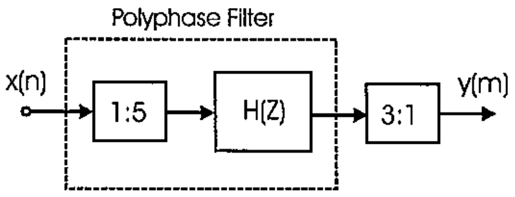


Figure 7.19 Straightforward Approach to 5/3 Interpolator

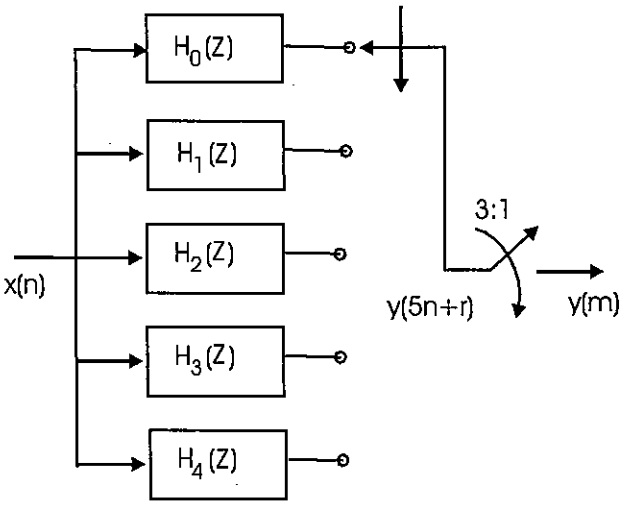


Figure 7.20 Detail Presentation of 5/3 Interpolator

Figure 7.20 presents the polyphase partition of the 1-to-5 interpolator with the output port commutating between successive output samples at the intermediate output rate of 5-times the input sample rate. Since the 3-to-1 resampler following the 1-to-5 interpolator discards two out of three samples, it would be foolish to compute the output points from the interpolator that are destined to be discarded.

Figure 7.21 shows successive indices of the input time series with respect to an arbitrary starting index n as well as the indices of the commutated output series from the 1-to-5 interpolator. The 3-to-1 resampler following the commutator saves one sample in three. We indicate the saved sample location by a check mark and the discarded samples with an X. The successive indices of the output commutator matching the check marked samples are shown as final output samples. The output samples are labeled with successive sample numbers with respect to an arbitrary starting index m. As expected, we observe that when the input index has stepped through 3 input samples, the output index steps through 5 output samples.

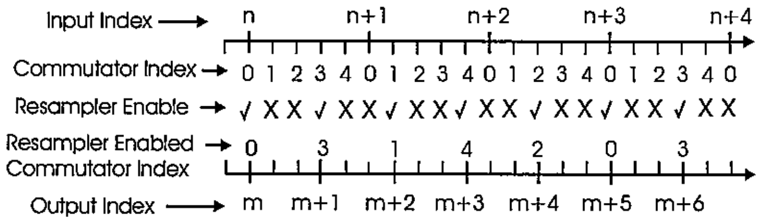


Figure 7.21 Time Indices for Input Series, for Output Commutator Series, Save and Discard Indicator for 3-to-1 Resampler, Saved Commutator Indices, and Output Series

We note that the preserved samples from the resampled commutator indices in stride of 3 with an address overflow implied by a modulo-5 operation on successive increments. An input sample is delivered to the polyphase filter each time the modulo-5 operation is invoked. For instance, starting at input n, we access commutator 0, then 0 + 3 = 3. The next port is 3 + 3 = 6 or 6 mod-5 = 1, so we insert the next sample n + 1 and access port 1 then 1 + 3 = 4. Again the next port is 4 + 3 = 7 or 7 mod-5 = 2, so we insert the next sample and access port 2. The next port is 2 + 3 = 5 or 5 mod-5 = 0, and the pattern continues to repeat. The indexing process is shown in Table 7-2, and a simple state machine can track and control the input-output sequencing process.

Table 7-2 Indices for Input, for Commutator, and for Output Series

Input Index	Commutator Index	Output Index
n	0, 3	m, m + 1
n + 1	1, 4	m + 2, m + 3
n + 2	2	m + 4

7.5 Arbitrary Resampling Ratio

We demonstrated in the previous section that we could use a P-stage polyphase filter to up sample by P and down sample by Q to obtain any rational ratio resampling of the form P/Q. Thus for instance, if we have a 12-stage polyphase filter, we can obtain 12 possible output rates formed by taking an output sample from each output port (12/1), or from every second output port (12/2), or from every third output port (12/3), and so on. A visualization of a possible resampling option is to be seen in Figure 7.22, where an input sequence is resampled by a factor of 12/5. This is accomplished by accessing the output commutator in stride of length 5 with commutator address computed modulo 12.

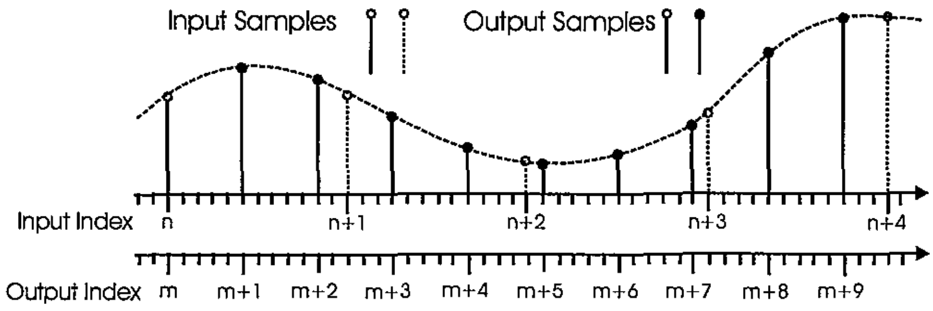


Figure 7.22 Resampling an Input Sequence by Accessing Every 5th Output Port of a 12-Stage Polyphase Filter

In general, the interpolation process computes output samples with fractional spacing between samples of αT with T being the input sample interval. When the parameter α is a rational ratio of the form P/Q, we can obtain that value precisely with a P-stage polyphase filter. This filter partitions the interval between successive input samples into P increments and forms successive output samples from every Qth commutator port. The number of stages, P, in the polyphase filter defines the time granularity of the interpolating process. Increasing the number of available stages in the polyphase filter will satisfy a requirement for smaller granularity. An increased number of stages does not increase the computational complexity of the interpolator but rather increases the number of coefficient sets available to compute output points. The increased number of coefficient sets requires additional memory space that leads to practical limits on the time granularity of this form of the interpolation process.

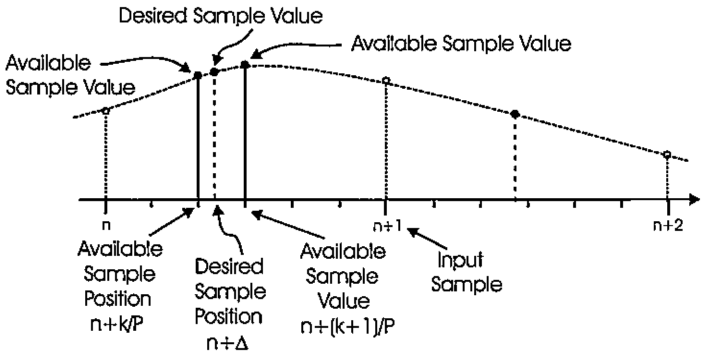

Desired Sample Value
Available Sample Value
Available Sample Value
n
n+1
n+2
Available Sample Position n+k/P
Desired Sample Position n+Δ
Available Sample Value n+(k+1)/P
Input Sample

Figure 7.23 Interpolating to a Position Between Available Output Points in a P-Stage Interpolator

With a sufficiently large number of polyphase partitions, the time granularity may be fine enough to use the outputs from the nearest available sample positions to form an acceptable approximation to the output at any arbitrary time location. The task of forming a sample value at a location that does not correspond to an available output location of a P-stage interpolator is illustrated in Figure 7.23.

7.5.1 Nearest Neighbor Interpolation

One option for arbitrary resampling with a polyphase interpolator is to assign the sample value obtained from the nearest neighbor sample location. This nearest neighbor replacement is shown in Figure 7.24. In practice, the nearest neighbor can be replaced with the neighbor to the immediate left. This is equivalent to truncating the desired offset value rather than rounding the desired offset value.

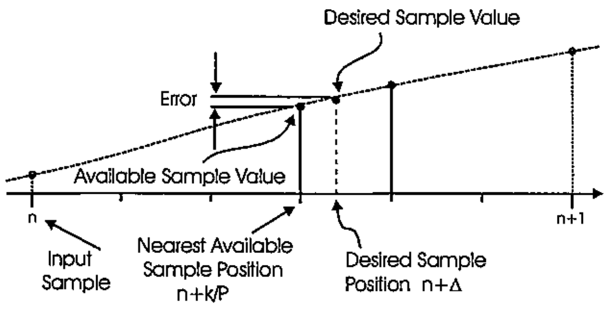

Desired Sample Value
Error
Available Sample Value
n
n+1
Input Sample
Nearest Available Sample Position n+k/P
Desired Sample Position n+Δ

Figure 7.24 Replacing Desired Sample Value with Nearest Neighbor Sample Value

The amplitude error in the interpolation process is related to an error in sample position. The sample value errors formed over successive output samples are modeled as timing jitter errors. If the resulting amplitude errors are smaller than the errors due to amplitude quantization of the sample values, the timing jitter errors do not degrade the quality of the interpolated samples. A simple way to estimate the size of the timing jitter errors is to describe the errors due to nearest neighbor replacement as an equivalent linear process and then estimate the errors in that process. We do this by imagining that we fully interpolate the input series to the maximum output sample rate using every output port, form an analog signal from these samples with a perfect DAC or zero-order hold, then resample this virtual analog waveform at the desired time locations. This model is shown in Figure 7.25.

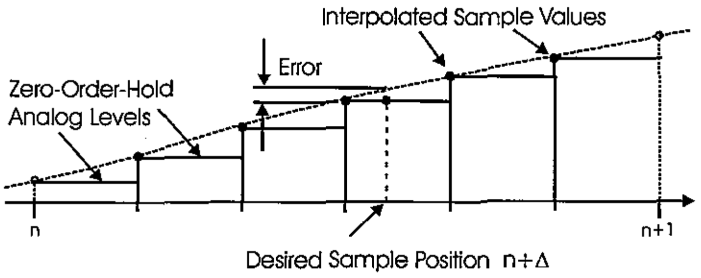


Figure 7.25 Timing Jitter Model: Sampling of Virtual DAC Output for Maximum Interpolated Sample Values

We can upper bound the errors due to the timing jitter by examining the spectrum of the signal obtained by maximally up sampling and then converted to an analog signal by the virtual DAC. We model the original sampled signal as uniformly occupying unity bandwidth, such as a Nyquist shaped spectrum in a communication system. A signal might originally be formed with a sample rate that is 4-times the input bandwidth and then up sampled to a new sample rate N times its Nyquist rate with an N/4 stage interpolator. The important parameter here is the output sample rate, not how it was obtained. The spectrum of the up-sampled signal at the input and output of the DAC processed signal is shown in Figure 7.26. The frequency response of the DAC is the standard sin(x)/x, shown in (7.11), with zeros located at multiples of the output sample rate. These zeros suppress the spectral copies centered at multiples of the sample rate but leave a residual spectrum in the neighborhood of the spectral zero.

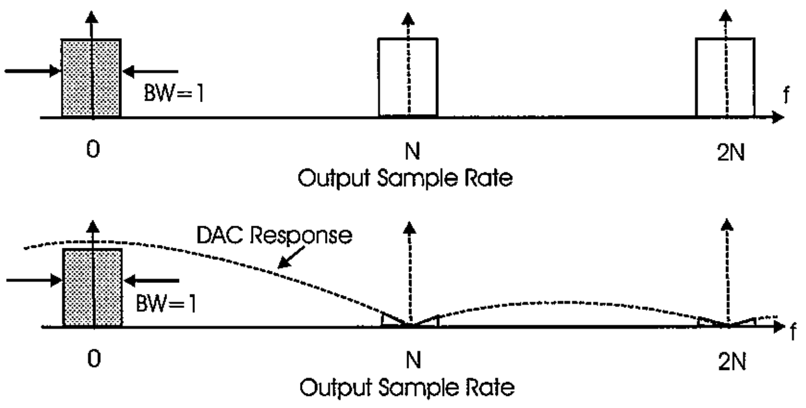


Figure 7.26 Spectrum of Up-Sampled Signal at Input and Output of DAC

The DAC spectral response is shown in (7.11), and its first derivative is shown in (7.12). Evaluating (7.12) at the first zero crossing of the spectrum, at f = 1/T, we obtain the results shown in (7.13).

$$H(f) = \frac{\sin(2\pi f \frac{T}{2})}{(2\pi f \frac{T}{2})} \tag{7.11}$$

$$\frac{d}{df}H(f)=\frac{(2\pi f\frac{T}{2})(2\pi\frac{T}{2})\cos(2\pi f\frac{T}{2})-(2\pi\frac{T}{2})\sin(2\pi f\frac{T}{2})}{(2\pi f\frac{T}{2})^2} \tag{7.12}$$

$$\left.\frac{d}{df}H(f)\right|_{f=\frac{1}{T}} = -T = -\frac{1}{f_S} \tag{7.13}$$

The Taylor series expansion of the DAC's spectral response at the first zero crossing is shown in (7.14).

$$H(\Delta f) = -\frac{1}{f_S}\Delta f \tag{7.14}$$

Substituting the sample rate indicated in the normalized frequency axis presented in Figure 7.26, we obtain the local Taylor series shown in (7.15).

$$H(\Delta f) = -\frac{1}{N}\Delta f \qquad (7.15)$$

A zoom to the spectral response of the DAC in the neighborhood of the first spectral zero is shown in Figure 7.27.

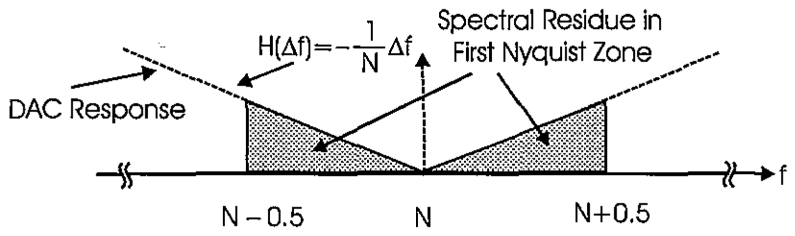

$H(\Delta f) = -\frac{1}{N}\Delta f$
Spectral Residue in First Nyquist Zone
DAC Response
f
N − 0.5
N
N+0.5

Figure 7.27 Frequency Response in Neighborhood of the DAC's First Spectral Null

The maximum amplitude of the residual spectrum centered about the first spectral null, obtained by substituting 1/2 for Δf, is seen in (7.16).

$$|H(\Delta f)|_{MAX} = \frac{1}{N}\Delta f\bigg|_{\Delta f=1/2} = \frac{1}{2N} \qquad (7.16)$$

The smallest resolvable signal level of a b-bit quantizer is 2^{-b}. If the residual spectral levels at the output of the DAC are below this level, the errors attributed to the timing jitter of the nearest neighbor interpolator are below the quantization noise level of quantized signal samples. To assure this condition, the maximum spectral level of the residual spectrum must satisfy the condition shown in (7.17).

$$\frac{1}{2N} < \frac{1}{2^b} \text{ or } N > 2^{(b-1)} \qquad (7.17)$$

The virtual analog signal described by the up sampling condition satisfying (7.17) can now be sampled at any output rate that satisfies the Nyquist criterion for the input bandwidth. The aliasing terms that fold into the primary Nyquist zone are the multiple residual spectra residing at the successive spectral nulls of the DAC interpolator. The amplitudes of the aliasing terms inherit the alternating signs and 1/M gain terms of the DACs sin(x)/x. In general the spectral terms do not alias to the same frequency, so we are not concerned with a build-up of their contributions. If the amplitudes of the successive alias terms do stack up, they form a geometric series with alternating signs or with the same signs. If we up sample by N and down sample by Q the worst case cumulative gain due to the folding is (Q/(Q−1), which results in worst case folded spectral levels of Q/(Q−1) * 1/(2N). In practice the collective aliases of all the residual spectra from the DAC's spectral zero crossings to the Nyquist interval of the new sample rate do not rise above the highest level of 1/(2N).

By way of example, consider interpolating a time series represented by 8-bit samples. For this case, if we operate the interpolator to obtain a maximum output sample rate 128 times the signal bandwidth, the noise spectrum due to nearest neighbor interpolation will be below the noise caused by the signal quantization process. If the input signal is originally oversampled by a factor of 4, the interpolator must make up the additional factor of 128/4 or 32. Thus a 32-stage interpolator can resample an 8-bit input signal with jitter-related noise levels below the –48-dB dynamic range noise level of the 8-bit quantized data. Figure 7.28 presents the time and spectral response of a 45-tap filter response that is initially oversampled by a factor of 4. Note the spectrum is scaled for two-sided 3-dB bandwidth of 1 for which the sample rate of 4 is presented on an axis of ± 2.

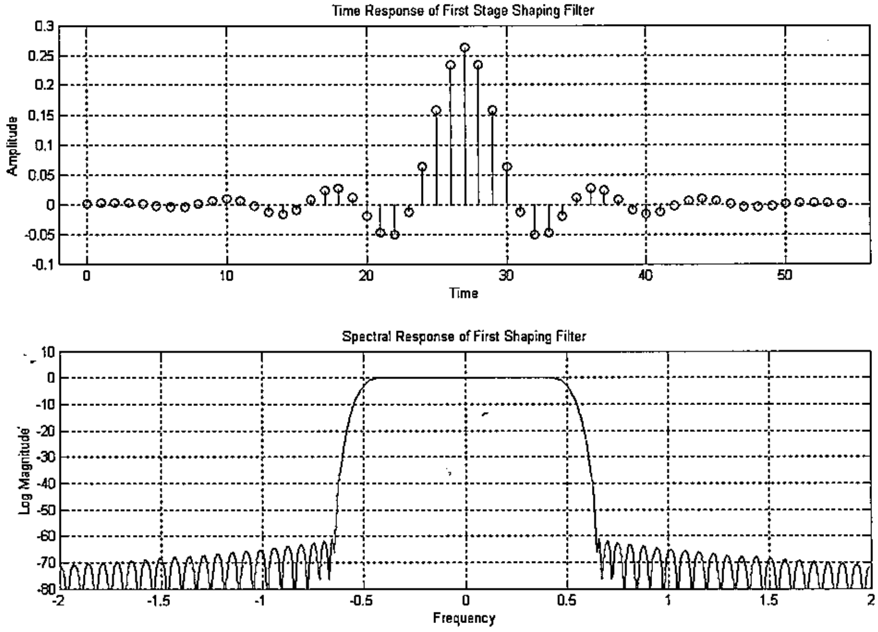


Figure 7.28 Shaping Filter: Time and Frequency Response Four Times Oversampled

Figure 7.29 presents the time and frequency response obtained by applying the interpolation process to resample the time response of Figure 7.28. The sample rate change shown here is 32/6.4, a sample rate change of 5. This sample rate change is obtained by stepping through the 32 output commutator port indices by the integer part of an accumulator that is incremented in steps of 6.4. The number of output samples formed by the interpolation process is seen to be 245 points. The spectrum obtained by the interpolation process clearly shows the 4-spectral regions at multiples of the input rate that have been suppressed by the sin(x)/x response of the virtual DAC employed by the nearest neighbor interpolator.

The level of suppression is 50-dB, 2-dB more than the maximum 48-dB level estimated in (7.14). This apparent excess attenuation is due to the fact that the spectral width of the input signal was less than unity bandwidth due to its transition roll off. The reduced spectral occupancy lowers the Δf in (7.13) and results in a reduced level estimate for the maximum amplitude spectral residue.

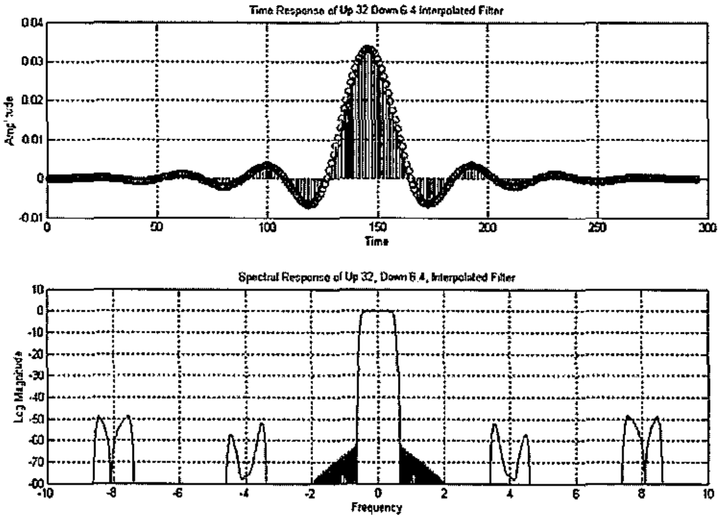


Figure 7.29 Time and Frequency Response of 32/6.4 Nearest Neighbor Interpolator

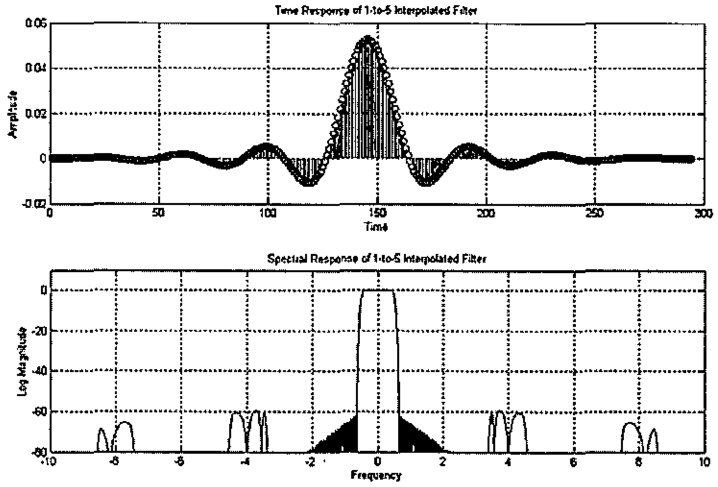


Figure 7.30 Time and Frequency Response of Straight 1-to-5 Polyphase Interpolator

Note that the spectrum shown in Figure 7.29 differs slightly from the spectrum obtained by a straight 1-to-5 interpolator performing the same up-sampling task. A spectrum obtained from a 1-to-5 up sampler is shown in Figure 7.30. Note that here the residual spectra are filtered versions of the spectral replicates as opposed to the folded sin(x)/x suppressed spectral images of Figure 7.29. Both filters were designed to suppress spectral copies by 60-dB and different effects form the spectral residues we observe in Figures 7.29 and 7.30. Both filters satisfy the design requirements with the residual spectra submerged under the quantizing noise of the 8-bit representation of the input signal.

For comparison, Figure 7.31 shows the time and spectral response when the same 32-stage interpolator is incremented in steps of 6.37 to obtain a sample rate change of 32/6.37 or 5.0235. The slightly higher sample rate presents 246 instead of the 245 output samples of the previous example. We note here that the aliases no longer fold to four common frequencies and now appear over the entire span of the new Nyquist interval of width 4* 5.0235 or 20.094.

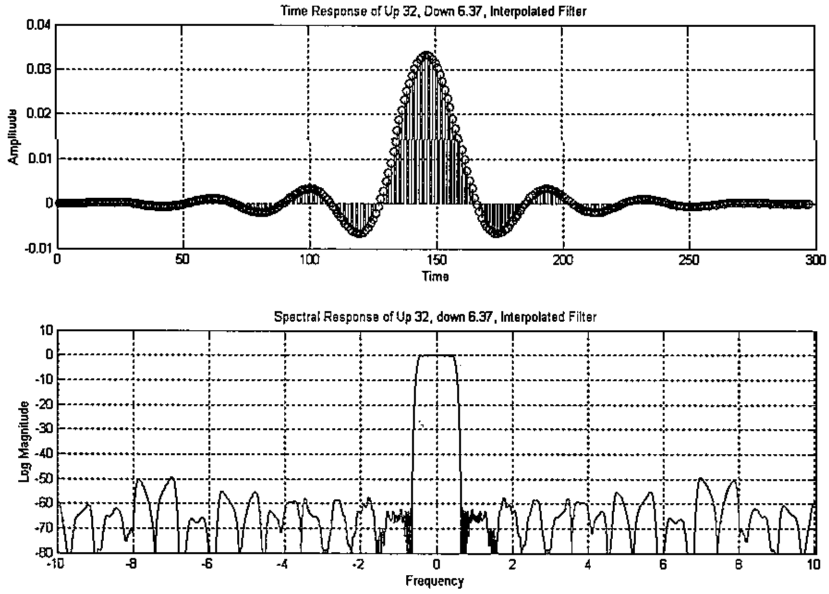


Figure 7.31 Time and Frequency Response of 32/6.37 Nearest Neighbor Interpolator

The nearest neighbor indexing is usually implemented as a truncation-indexing scheme. A block diagram of the input and output index processing is shown in Figure 7.32. The process can be visualized with the aid of a pair of sawtooth waveforms representing the input and output clocks shown in Figure 7.33. The process operates as follows: A new input data sample is shifted into the filter register. The integer part of the accumulator content selects one of the 32 filter weights that is used to compute and output a sample point. The output clock then directs the accumulator process to add the desired increment delta to the modulo-32 accumulator and increment the output index m. This process continues until the accumulator overflows at one of the output clock increments. Upon this overflow the input index is incremented, a new input data point is shifted into the filter, and the process continues as before.

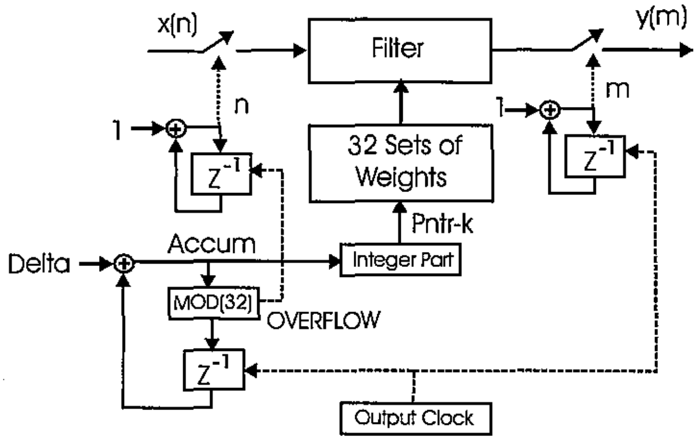


Figure 7.32 Block Diagram of Input, Output, and Filter Weight Set Index Control

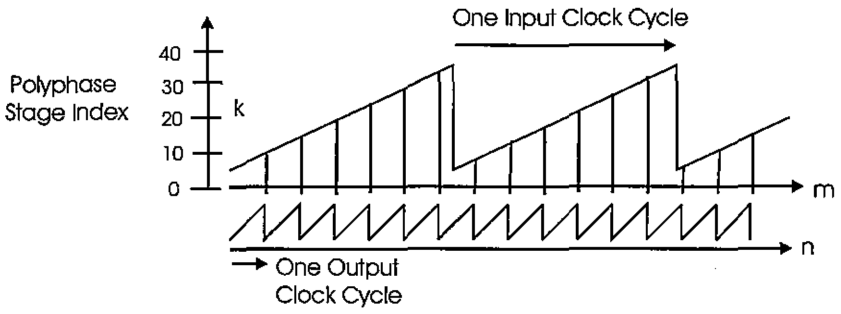


Figure 7.33 Visualization of Relationship Between Input Clock, Output Clock, and Polyphase Filter Index Pointer

A segment of MATLAB code that performs the alignment of output clock index m with filter weight index pntr_k and input clock index n is shown next. The interpolating filter weights are stored in hh, and filtering is performed as the inner product of the 4-tap register named reg and the selected filter weights wts(pntr_k,:). Figure 7.34 displays a segment of the sequence of the pntr_k indices obtained using this MATLAB code with del = 3.14. Note the precession of the start and stop index in successive cycles through the 32-stage index selection.

```
        del=5.3;     %del=32*f_in/f_out;
        accum=0;
        mm=1;
        nn=1;
        reg=zeros(1,4);
        wts=reshape(hh,32,4);
% Append Zeros to flush Filter Register
        xx=[xx zeros(1,4)];

        while nn<=length(xx)
          reg=[xx(nn) reg(1,3)];

             while accum < 32
               pntr_k=floor(accum)+1;
               yy(mm)=reg*wts(pntr_k,:)';
               mm=mm+1;
               accum=accum+del;
             end

          accum = rem(accum,32);
          nn=nn+1;
        end
```

Segment of MATLAB Code to Control Input and Output Indexing

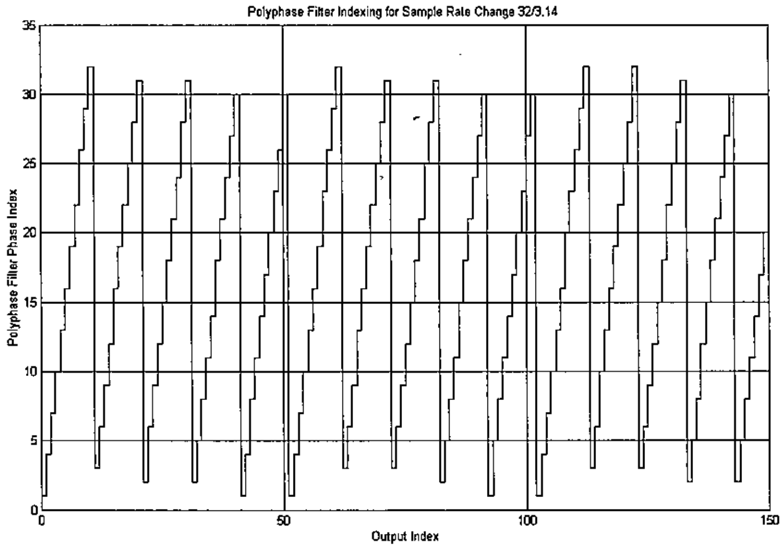
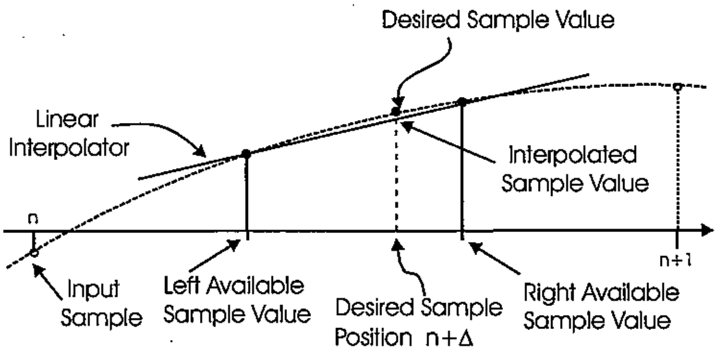


Figure 7.34 Example of Indexing Sequence Formed by Address Control in Resampling Routine

7.5.2 Two Neighbor Interpolation

Equation (7.17) provided an estimate of the amount of oversampling required to keep spectral artifacts in nearest neighbor interpolation below the quantizing noise level of a b-bit representation of the data samples. When the number of bits is large, the required oversample rate becomes excessive. For instance, to match the –96-dB noise level of a 16-bit data set we require an oversample rate of 65,536, which in turn requires a polyphase filter of 16,384 stages. We must improve on the nearest neighbor approximation. One option that leads to a smaller number of stages is linear interpolation between the two available adjacent neighbors that bracket the desired sample position. This option is shown in Figure 7.35.



Desired Sample Value
Linear Interpolator
Interpolated Sample Value
n
n+1
Input Sample
Left Available Sample Value
Desired Sample Position n+Δ
Right Available Sample Value

Figure 7.35 Linear Interpolation Between Available Left and Right Sample Values

The equation for linear interpolation between two samples at k and k+1 is shown in (7.18).

$$x(k+\Delta)=x(k)\cdot(1-\Delta)+x(k+1)\cdot\Delta \tag{7.18}$$

In a manner similar to the DAC process being equivalent to convolving a time sequence with a rectangle pulse, the linear interpolator can be modeled as convolving a time sequence with a triangle pulse. This interpretation is shown in Figure 7.36. The arithmetic required to compute the linearly interpolated sample is shown in (7.19)

$$\begin{aligned} x(k+\Delta)&=x(k)+[x(k+1)-x(k)]\,\Delta \\ &=x(k)+\dot{x}(k)\cdot\Delta \end{aligned} \tag{7.19}$$

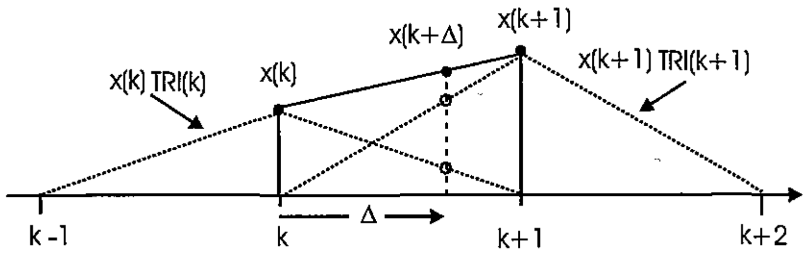


Figure 7.36 Linear Interpolation as Convolution of Sample Points with Triangle Pulse

The triangle pulse can be formed as the convolution of the rectangle with a second identical rectangle. Since convolution in time is equivalent to a product in frequency, the Fourier transform of the triangle is the product of the transform of the rectangle with itself. The transform of the triangle is shown in (7.20).

$$H(f)=\left[\frac{\sin(2\pi f\frac{T}{2})}{(2\pi f\frac{T}{2})}\right]^2 \tag{7.20}$$

The spectral response of the up-sampled data set subjected to the linear interpolator is a $[\sin(x)/x]^2$ weighting applied to periodic spectra of the up-sampled sequence formed by the interpolator. This is shown in Figure 7.37.

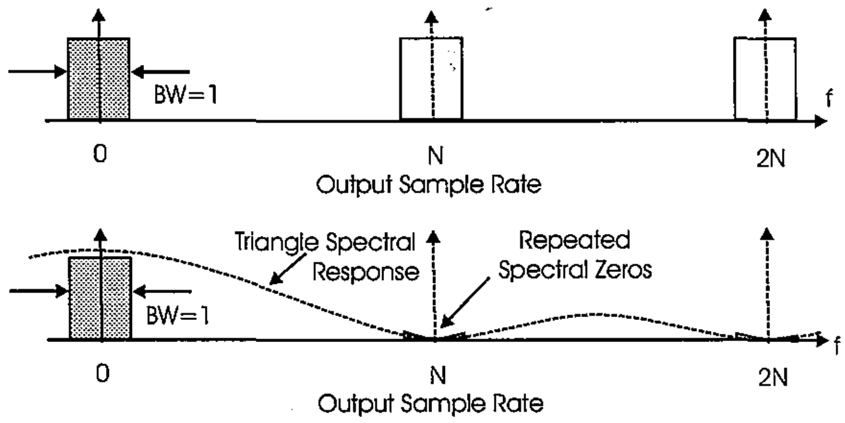


Figure 7.37 Spectrum of Up-sampled Signal at Input and Output of Triangle Interpolator

Note that the repeated zeros of the $[\sin(x)/x]^2$ offer additional suppression of the spectral replicates. We seek the first non-zero term in the Taylor series at the first repeated zero of the $[\sin(x)/x]^2$. This is simply the square of the first non-zero term of the Taylor series of sin(x)/x. This is shown in (7.21) and is illustrated in Figure 7.38.

$$H(\Delta f) = \left[\frac{1}{N}\Delta f\right]^2 \tag{7.21}$$

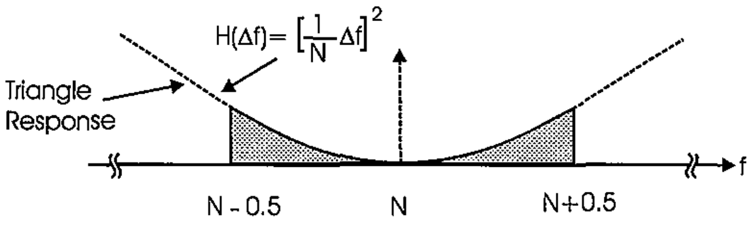


Figure 7.38 Frequency Response in Neighborhood of Triangle First Spectral Null

As in the previous section, the errors due to the linear interpolation will not be observable if the residual spectral level is below the noise level attributed to the quantization noise of the signal. Since the error due to the b-bit quantized signal is 2^{-b}, we require the residual spectral levels to be below this level. Substituting the maximum frequency offset Δf of 1/2 in (7.21) and comparing this level to the quantization noise level, we obtain the condition shown in (7.22).

$$\left[\frac{1}{2N}\right]^2 < \frac{1}{2^b} \quad \text{or} \quad N > 2^{(b-2)/2} \tag{7.22}$$

Thus, for this example, if we have a requirement to interpolate a 16-bit data set, we can keep the spectral artifacts below the quantization noise level if N, the oversample rate, is greater than 128. Continuing with the assumption of the previous example, that the input signal is oversampled by a factor of 4, the number of stages required in the polyphase interpolator is 128/4 or 32. Figure 7.39 shows the time and frequency response of a 4-times oversampled shaping filter with –96-dB side-lobes. Figure 7.40 shows the time and frequency response formed by a 32-stage interpolating filter using linear interpolation between available output samples bracketing the desired time samples. Note that the levels of the spectral errors seen in this figure are at the expected –96-dB level. In order to obtain the desired 96-dB attenuation, the length of the interpolating filter had to be increased from 4 taps per stage, used for the previous example, to 5 taps per stage. There is a slight spectral droop of the in-band spectrum caused by the main lobe curvature of the $[\sin(x)/x]^2$. We can precompensate for this droop in the design of the input shaping filter, or we can correct for the anticipated droop with a compensating filter prior to use of the interpolator.

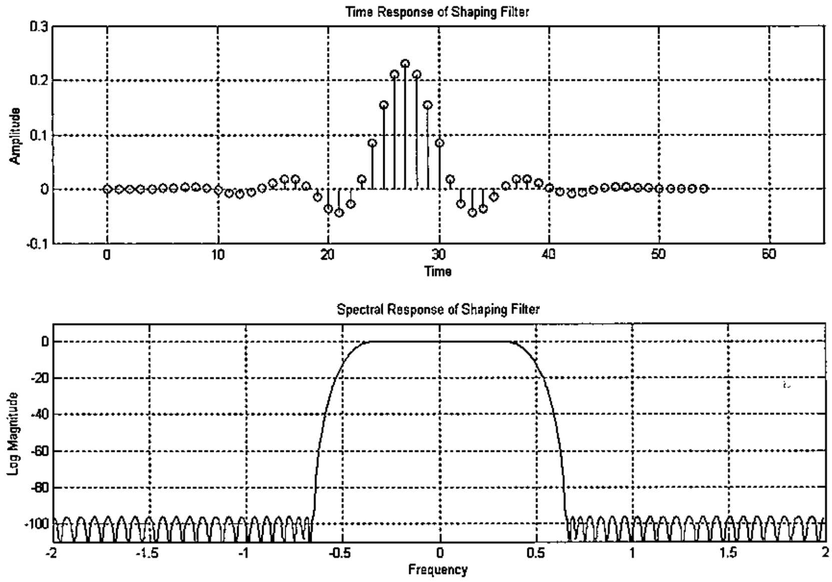


Figure 7.39 Shaping Filter: Time and Frequency Response Four Times Oversampled

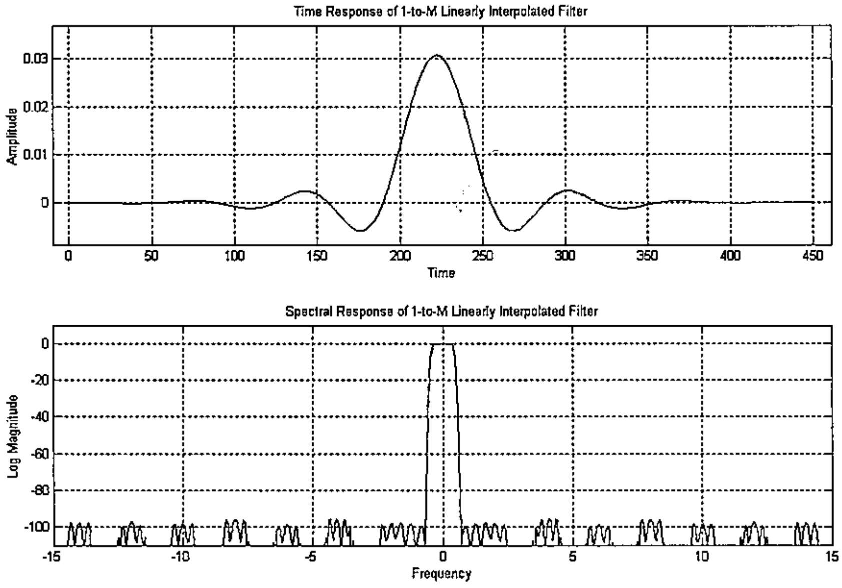


Figure 7.40 Time and Frequency Response of 32/6.4 Linear Interpolator

7.6 Farrow Filter

The Farrow filter offers two alternative approaches to the arbitrary resampling problem. In the first approach, coefficients of the polyphase filter stages are computed on the fly from low order piecewise polynomials. These polynomials form approximations to segments of the impulse response of the original interpolating filter prior to the polyphase partition. In the second alternative, the coefficients of the approximating polynomials are rearranged and are applied directly to the input data to form data-dependent, locally valid, polynomial approximations to the input data as opposed to the filter. The data polynomial is in turn evaluated at the desired sample points.

7.6.1 Classical Interpolator

Arbitrary resampling filters are designed and implemented as polyphase P-path filters. Each path provides a delay equal to an integer multiple of 1/Pth of the input sample interval for an up-sampler and of 1/P of the output sample interval for a down sampler. Down sampling is implemented concurrently with the up sampling by stepping through the commutator output ports in increments of Q to realize a rational ratio sample rate change of P/Q. In efficient implementations, the filter is a single stage with the commutation process performed by a pointer in coefficient (memory) space. Figure 7.41 shows the functional structure of a 1-to-P up sampler with an embedded Q-to-1 down sampler performed by stepping in stride of Q through P commutated stages.

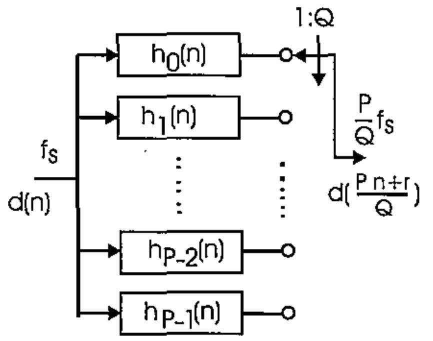


Figure 7.41 A Polyphase Up-Sampling Filter

When used for arbitrary resampling, P, the number of stages in the filter, is selected to be sufficiently large so that phase jitter artifacts due to selecting the nearest neighbor to the desired interpolation point are sufficiently small. This time jitter is shown in Figure 7.42 where the desired output sample point is located between sample points k and k+1. The output port is normally selected from the left stage to avoid the problem of stage index overflow for the P−1 stage of sample n and the 0 stage of sample n+1.

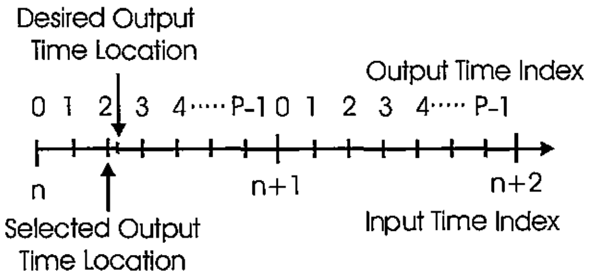

Desired Output Time Location
Output Time Index
0 1 2 3 4·····P-1 0 1 2 3 4····· P-1
n
n+1
n+2
Selected Output Time Location
Input Time Index

Figure 7.42 Input and Output Sample Location for P-stage Resampling Filter

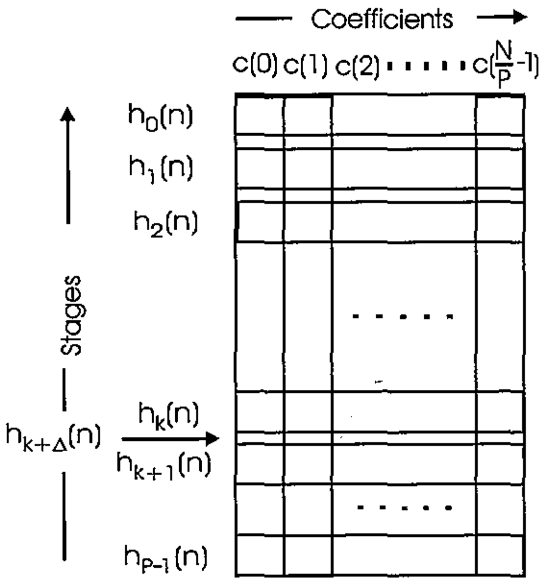


Figure 7.43 Coefficient Mapping of P-stage Polyphase Filter

We examine the polyphase partition or coefficient mapping of the prototype filter to understand how and where to apply the polynomial approximation. The weights $h_r(n)$ of the rth stage of a P-stage polyphase filter are obtained from the prototype filter weights h(n) by the mapping $h_r(n) = h(r+nP)$. This mapping can be easily visualized as mapping the N-taps of the prototype filter into a two-dimensional array with P rows and N/P columns by filling successive columns in natural order. Thus the first column contains the first P points, the second column contains the next P points, and so on. The rows of this two-dimensional array are the polyphase filter sets with the *r*th set holding the weights to compute the *r*th interpolated output point located r/P fraction of the input intersample interval. This two-dimensional mapping is shown in Figure 7.43. Each of the rows of the partition corresponds to a sample point in the interpolated data stream. Row k would compute a sample y(n+k/P), and row k+1 would compute a sample y(n+(k+1)/P). Shown in Figure 7.43 is a pointer pointing to a nonexistent filter at sample y(n+(k+Δ)/P). Our traditional response to computing an output point at this position is to use nearest neighbor y(n+k/P) or compute the two outputs y(n+k/P) and y(n+(k+1)/P) and linearly interpolate by distance Δ between them.

Figure 7.44 shows the time and frequency response of a 250-tap interpolating filter designed for 1-to-50 interpolation of data samples with 12-bit or 72-dB dynamic range. Notice the filter coefficients are samples of a smooth continuous filter.

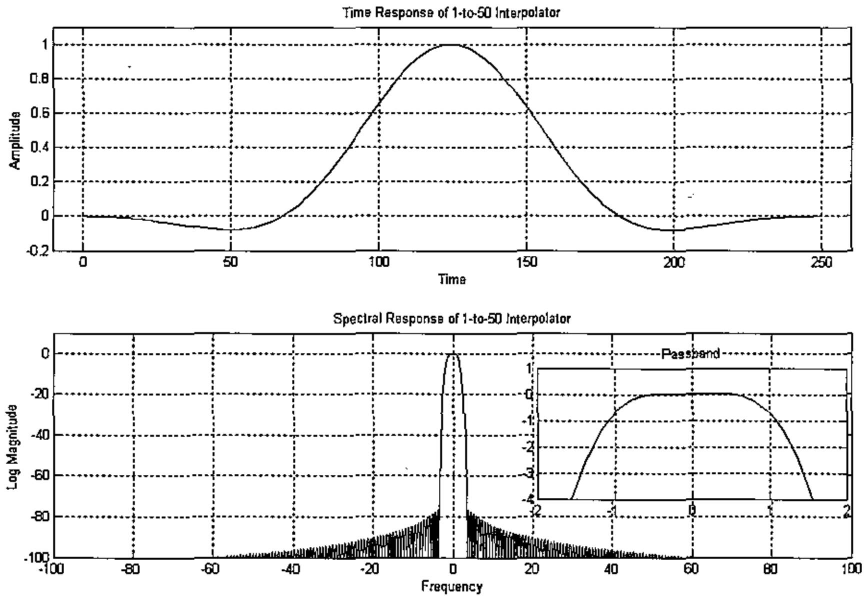


Figure 7.44 Time and Frequency Response of 300-Tap, 1-to-50 Interpolating Filter

The filter in Figure 7.44 is partitioned as shown in Figure 7.43 into five columns of length 50, with each row of the partition forming one path of the 50-path interpolator. Figure 7.45 presents the five 50-point sequences, which are seen to be contiguous segments of the prototype filter. A weight vector containing the first coefficient in each column is the first row of the polyphase filter; similarly, a vector containing the second coefficient in each column is the second row of the filter, and so on. Suppose we wanted a filter coefficient set to compute an output between any two filter sets, say filter set 25 and filter set 26. If we examine the coefficients for this pair of filters we would recognize they are very similar. If we interpolate between these weights we would form a new set of weights that would indeed compute the output sample at the time offset corresponding to the interpolated distance between the weights.

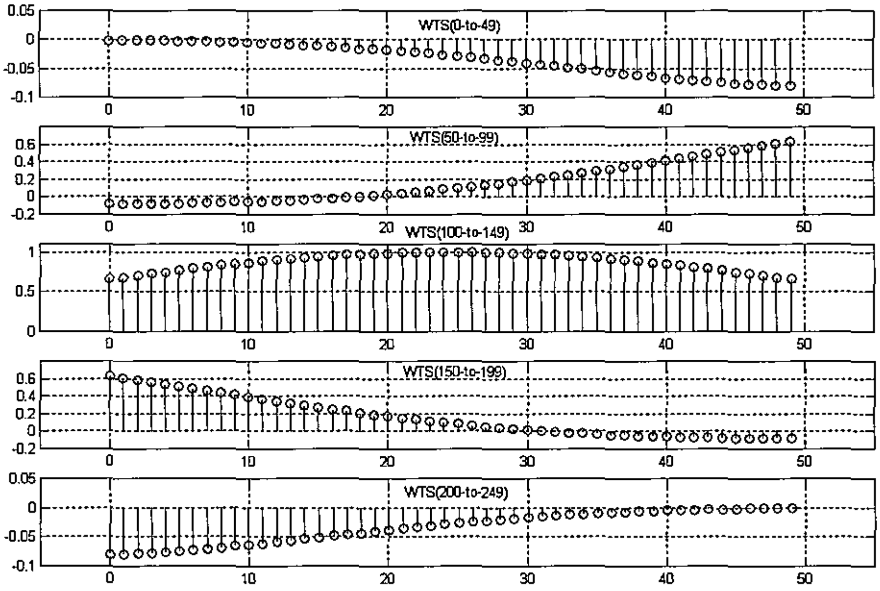


Figure 7.45 Contents of Five Columns in Polyphase Partition of 250-tap Interpolating Filter

The change to interpolating between filter weights rather than between sample outputs is seen in the following manipulations. Equation (7.23) presents the linear interpolation required to obtain an output distance Δ between the output from sub filter r and sub filter r+1.

$$y(n+\frac{r+\Delta}{P}) = y(n+r)\cdot(1-\Delta) + y(k+r+1)\cdot\Delta \tag{7.23}$$

The outputs of these subfilters are obtained as an inner product of the input samples in the filter register and the two sets of weights previously denoted by $h_r(n)$ and $h_{r+1}(n)$. The linear interpolation of the inner products is shown in (7.24).

$$y(n+\frac{r+\Delta}{P}) = (1-\Delta)\sum_{k=0}^{5} x(n-k)\,h_r(k) + \Delta\sum_{k=0}^{5} x(n-k)\,h_{(r+1)}(k) \tag{7.24}$$

Since the same data samples are used in the two inner products, the order of summation and inner product can be interchanged as in (7.25). Here we see that the output sample has been computed at the desired sample location with a filter weight set designed specifically for that output location r+Δ.

$$\begin{aligned} y(n+\frac{r+\Delta}{P}) &= \sum_{k=0}^{5} x(n-k)\,[(1-\Delta)\cdot h_r(k)+\Delta\cdot h_{(r+1)}(k)] \\ &= \sum_{k=0}^{5} x(n-k)\,h_{(r+\Delta)}(k) \end{aligned} \tag{7.25}$$

Examining the factored form shown in (7.24) we can arrive at an alternate version of the interpolated filter set shown in (7.26). Here we recognize that the filter weight, formed as the difference between adjacent filter sets, is a filter that forms the derivative output series.

$$y(n+\frac{r+\Delta}{P}) = \sum_{k=0}^{5} x(n-k)\cdot h_r(k)+\Delta\cdot\sum_{k=0}^{5} x(n-k)\cdot[h_{(r+1)}(k)-h_r(k)] \tag{7.26}$$

We can use the second term in this factored form as a second set of weights that computes the signal derivative to accompany the original interpolator set. We then perform two inner products to form the left sample at position n+r/P and the derivative of the left sample at the same location. This option is shown in (7.27).

$$y(n+\frac{r+\Delta}{P}) = \sum_{k=0}^{5} x(n-k)\cdot h_r(k)+\Delta\cdot\sum_{k=0}^{5} x(n-k)\cdot\dot{h}_r(k) \tag{7.27}$$

We then use the local Taylor series to interpolate to the desired position n+(r+Δ)/P. This local Taylor series option is shown in (7.28).

$$y(n+\frac{r+\Delta}{P}) = y(n+\frac{r}{P})+\Delta\cdot\dot{y}(n+\frac{r}{P}) \tag{7.28}$$

We have just examined two options to perform fine-grain interpolation of an input time series. These were the interpolation of the filter weights or the forming of a Taylor series for the envelope of the output time sequence. These two options are the core of the Farrow filter that we examine next.

7.6.2 Polynomial Approximation

We return to the coefficient columns of the polyphase partition presented in Figure 7.45. We noted that the coefficients of the *r*th stage are located in the *r*th position of the successive columns of the two-dimensional representation of the prototype filter. Similarly, the coefficients of the *(r+1)*th stage are in the *(r+1)*th position of each column. We also noted that if we required a filter stage for an output sample between stages r and r+1, say at r+Δ, we could interpolate filter coefficients, an option presented in (7.25).

We now take another tack and model the coefficient set in each column, as a P-sample wide section of the prototype filter, which can be considered to be samples of a smooth continuous function. We can approximate these sections with low-order polynomials and can compute the coefficients for a filter stage at any position by evaluating these polynomials. Tchebyschev, or equal ripple approximations to the column entries, are easy to compute, and the maximum error can be controlled by selection of the degree of the approximating polynomials.

A useful rule for implementing FIR filters is that out-of-band side-lobe levels are bounded by 5-dB per bit. Thus if we design a FIR filter with minimum side-lobe attenuation of 72-dB we must represent the finite length coefficient set with at least 15-bits. Since leading zeros in a coefficient reduce arithmetic precision we require that the coefficients be scaled to set the maximum coefficient to 1 (i.e., left justified in coefficient space). If we design a filter with a minimum of 72-dB attenuation, we require 15-bit precision of the coefficient list which translates to errors less than 2/32768 or approximately $6\text{x}10^{-5}$. This is the measure we use to specify the acceptable error between the polynomial approximation and the entries in the polyphase columns.

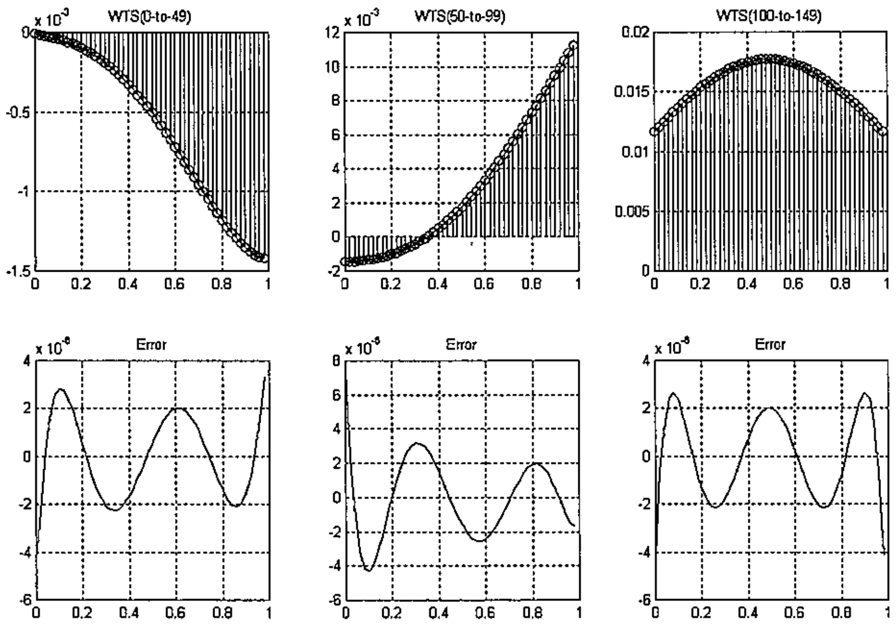


Figure 7.46 First Three Columns of Polyphase Filter, 4th-Order Polynomial Approximation, and Errors Between Columns and Approximation

Figure 7.46 presents the 50 samples of the first three columns of the polyphase partition of the prototype filter along with an approximating curve formed by a 4th-order polynomial. Also shown is the error between the 50 samples and the 4th-order approximating polynomial evaluated at these 50 points. Note the error is everywhere less than 10^{-5}, better by a factor of 6 than required by the approximation requirement.

Figure 7.47 presents the structure of an arbitrary position resampling filter. Rather than use the control mechanism to select the desired sub filter from a polyphase partition, this form of the filter uses the polynomials to compute the filter coefficients corresponding to the desired sample location r. Note that the original 50-stage polyphase filter required storage of 250 coefficients plus a mechanism to interpolate between output sample points. In this alternate form the filter is defined by the five coefficients of each 4th-order polynomial, and since there are five polynomials, the filter is defined by 25 coefficients and a processor to evaluate the five polynomials at the correct time offset to form the filter coefficients $h_r(n)$, where r is not limited to an integer divided by P.

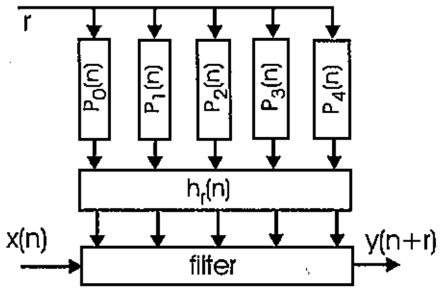


Figure 7.47 Polynomial Form of Arbitrary Resampling Filter

Figure 7.48 presents the time and frequency response of the polyphase filter impulse response formed by samples computed by the five 4th-order polynomials described earlier. Note that the side-lobe levels are maintained at or below the desired -72-dB level.

A word of caution: The Remez algorithm designs filters with equiripple pass band and stop band errors. We know that when the stop band spectrum of filter exhibits constant level side-lobes (or zero-decay rate), the impulse response almost always contains observable impulses at the two ends of the time series. The presence of this impulse precludes the use of a low degree polynomial approximation to the section containing the impulse. We must either design the impulse out of the time response by use of a modified Remez algorithm or, in desperation, we simply cut the offending impulse down to the value determined by its two adjacent neighbors (to obtain continuous slope). The sample slicing may result in a 6-dB increase in nearby side-lobe levels and side-lobe decay rate of 6-dB per octave. Simply design the filter with an additional 6-dB attenuation and proceed with the slicing operation. Having a filter with falling side-lobes offers an incidental system benefit of reduced *integrated side-lobe (ISL) levels*. Lower ISL results in improved filtering performance when the side-lobes alias back into band due to the concurrent down sampling when operating in the arbitrary resampling mode.

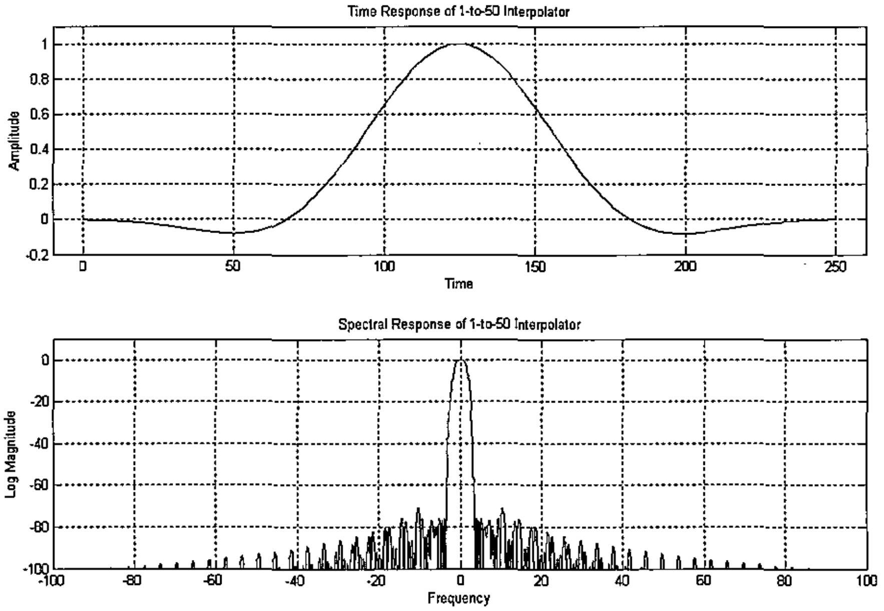


Figure 7.48 Time and Frequency Response of 50-stage Interpolator Formed by 4th-Order Polynomial Approximations to Five Columns of Polyphase Partition

7.6.3 Farrow Filter

The *m*th coefficient of the interpolation polynomial for the offset Δ is computed by evaluating the polynomial $P_m(x)$ at $x = \Delta$. The polynomial P_m is the approximation assigned to the *m*th column of the polynomial interpolator. The form of this polynomial is shown in (7.29).

$$P_m(x) = \sum_{\ell=0}^{4} b(\ell,m)x^\ell \qquad (7.29)$$

The output of the filter using the coefficients from (7.29) is shown in (7.30)

$$y(n+\Delta) = \sum_{m=0}^{4} P_m(\Delta)\cdot x(n-m) \qquad (7.30)$$

We can substitute (7.29) in (7.30) and obtain the double sum of (7.31).

$$y(n+\Delta) = \sum_{m=0}^{4} \sum_{\ell=0}^{4} b(\ell,m)\, \Delta^{\ell} \cdot x(n-m) \tag{7.31}$$

Reordering the summations in (7.31), we obtain the form shown in (7.32).

$$y(n+\Delta) = \sum_{\ell=0}^{4} \Delta^{\ell} \sum_{m=0}^{4} b(\ell,m) \cdot x(n-m) \tag{7.32}$$

Note here that the coefficients b(l,m) are no longer applied as a Taylor series for the filter weights but now are coefficients of a filter applied to the data to form a new set of data dependent coefficients denoted c(n,l) as in (7.33).

$$c(n,\ell) = \sum_{m=0}^{4} b(\ell, m) \cdot x(n-m) \tag{7.33}$$

We substitute the coefficients c(n,l) in (7.32) to obtain (7.34).

$$y(n+\Delta) = \sum_{\ell=0}^{4} c(n,\ell)\Delta^{\ell} \tag{7.34}$$

Examining (7.34), we recognize the form as the Taylor series representation of the output sequence. Consequently, the terms c(n,l) of (7.33) are successive local derivatives of the input series hence of the output series. Equation (7.30) formed the desired output by evaluating the polynomial expansion of the filter weights and then applying these weights to the filter data. By rearranging the double summation the coefficients of the polynomial expansion have become filters applied to the data to form the local data dependent coefficients for the polynomial expansion of the data. These polynomial filters, along with their frequency responses, are shown in Figure 7.49. Here we see that the first filter estimates the local DC term while the next filter estimates the local first derivative and so on.

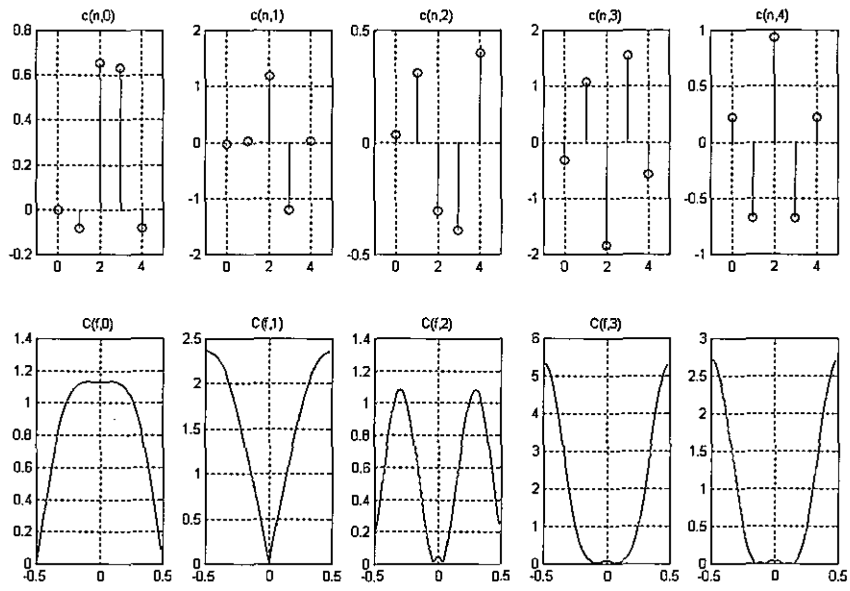


Figure 7.49 Impulse and Frequency Response of Farrow Filters: Filters Form Taylor Series Coefficients of Input Time Series

Table 7-3 lists the coefficients of the polynomial interpolators for the 50-stage filter presented in the previous section. The $P_k(\Delta)$ are the polynomial coefficients of the filter that processes the input data samples x(n–k) to compute the output y(n–Δ). Equation (7.35) shows how the coefficients are generated from the table for a specific Δ and then applied to the input data to compute the desired output sample.

Table 7-4 lists the rearranged coefficient set that forms the Taylor series expansion of the desired output time series. The $C_r(x)$ are the polynomial coefficients of the filter that processes the input data samples x(n–k) to compute the Taylor series coefficients. The Taylor series coefficients are then evaluated at Δ to obtain the output y(n–Δ). Equation (7.36) shows how the coefficients, generated for a specific set of input samples, are evaluated at a specific Δ to compute the desired output sample. It is useful to select a sample x(n) and see how it intersects with both sets of coefficients to contribute to an output sample.

Table 7-3 Polynomial Filter for Output at Δ

	Δ^4	Δ^3	Δ^2	Δ^1	Δ^0
$P_0(\Delta)$	0.2012	–0.2536	–0.0198	–0.0090	–0.0018
$P_1(\Delta)$	–0.6648	1.0550	0.3287	0.0225	–0.0828
$P_2(\Delta)$	0.9231	–1.8092	–0.3134	1.1759	0.6587
$P_3(\Delta)$	–0.6648	1.5511	–0.4007	–1.2035	0.6347
$P_4(\Delta)$	0.2012	–0.5350	0.3940	0.0210	–0.0827

$$y(n+\Delta) = p_0(\Delta)x(n)+p_1(\Delta)x(n-1)+p_2(\Delta)x(n-2)+p_3(\Delta)x(n-3)+p_4(\Delta)x(n-4) \tag{7.35}$$

Table 7-4 Coefficients to Compute Taylor Series Expansion of Input Time Series

	x(n)	x(n–1)	x(n–2)	x(n–3)	x(n–4)
$C_0(x)$	–0.0018	–0.0828	0.06587	0.6347	–0.0827
$C_1(x)$	–0.0090	0.0225	1.1759	–1.2035	0.0210
$C_2(x)$	–0.0198	0.3287	–0.3134	–0.4007	0.3940
$C_3(x)$	–0.2536	1.0550	–1.8092	1.5511	–0.5350
$C_4(x)$	0.2012	–0.6648	0.9231	–0.6648	0.2012

$$y(n+\Delta) = c_0(x)+c_1(x)\cdot\Delta+c_2(x)\cdot\Delta^2+c_3(x)\cdot\Delta^3+c_4(x)\cdot\Delta^4 \tag{7.36}$$

The output samples of the interpolator are computed by evaluating the Taylor series terms of (7.36) for each value of desired output Δ. Note that the coefficients are not updated until a new input sample is delivered to the coefficient engine. By comparison, a new polynomial must be formed and applied to the input data when the polynomial form of the interpolator is used. The Taylor series is most efficiently evaluated using Horner's rule for evaluating a polynomial. This is shown in (7.37).

$$\begin{aligned} y(\Delta) &= c_0+c_1\Delta+c_2\Delta^2+c_3\Delta^3+c_4\Delta^4 \\ &= c_0+\Delta\cdot(c_1+\Delta\cdot(c_2+\Delta\cdot(c_3+\Delta\cdot c_4))) \end{aligned} \tag{7.37}$$

A processor flow diagram that matches Horner's rule is shown in Figure 7.50, and an efficient hardware version of the same process is shown in Figure 7.51.

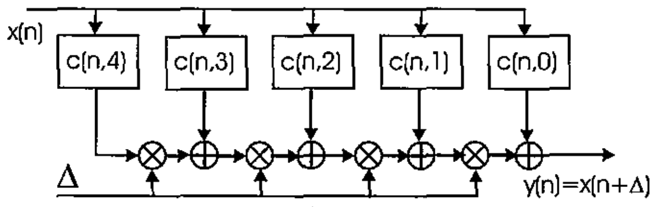


Figure 7.50 Signal Flow Diagram of Horner's Rule for Evaluating Polynomial

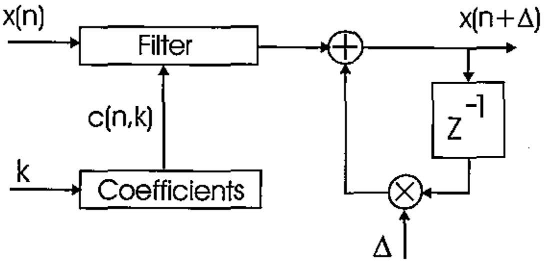


Figure 7.51 Efficient Hardware Version of Horner's Rule Processor

References

Crochiere, Ronald, and Lawrence Rabiner, *Multirate Signal Processing*, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1983.

Farrow, C. W., "A Continuously Variable Digital Delay Element," *Proc. IEEE Int. Symp. Circuits Systems*, (ICAS-88), Vol. 3, pp. 2642 - 2645, Espoo, Finland, June 6-9, 1988.

Fliege, Norbert, *Multirate Digital Signal Processing: Multirate Systems, Filter Banks, Wavelets*, West Sussex, John Wiley & Sons, Ltd., 1994.

Gardner, Floyd M., "Interpolation in Digital Modems: Part I: Fundamentals," *IEEE Trans. Comm.*, Vol. 41, No. 3, pp. 502 - 508, Mar. 1993.

harris, fred, "Forming Arbitrary Length Windows or Filter Sequences from a Fixed Length Reference Table," 18-th Annual Asilomar Conf. on Signals Systems, and Computers, Pacific Grove, CA., Nov. 1984.

harris, fred, "Performance and Design of Farrow Filter Used for Arbitrary Resampling," 31st Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, Nov. 1997.

Hentschel, Tim, *Sample Rate Conversion in Software Configurable Radios*, Norwood, MA, Artech House, Inc., 2002.

Jovanovic-Dolecek, Gordana, *Multirate Systems: Design and Applications*, London, Idea Group, 2002.

Laakso, Timo, Vesa Välimäki, Matti Karjalainen, and Unto Laine, "Splitting the Unit Delay: Tools for Fractional Delay Filter Design," *IEEE Signal Processing Magazine*, Jan. 1996, Vol. 13, No.1, pp. 30 - 60.

Mitra, Sanjit, *Digital Signal Processing: A Computer-Based Approach*, 2nd ed., New York, McGraw-Hill, 2001.

Mitra, Sanjit, and James Kaiser, *Handbook for Digital Signal Processing*, New York, John Wiley & Sons, 1993.

Sable, Les, and fred harris, "Using the High Order Nyquist Zones in the Design of Efficient Multi-channel Digital Upconverters," *IEEE Personal, Indoor and Mobile Radio Conference*, Sept. 1 - 4 1997, Helsinki, Finland.

Vaidyanathan, P. P., *Multirate Systems and Filter Banks*, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1993.

Problems

7.1 Design a FIR filter to be used as a 1-to-20 up sampler for an input signal initially oversampled by a factor of 4. The specifications for this design are as follows:

Sample Rate	400 kHz		
Pass Band	0-4 kHz	Pass band Ripple	0.1-dB
Stop Band	16-200 kHz	Stop band Atten.	60-dB, slope–6-dB/Octave

Plot the impulse response and log magnitude frequency response of the prototype filter to verify that it meets design specifications.

Form an input signal as the impulse response of a Nyquist filter, with the following specifications:

Sample Rate	20 kHz
Symbol Rate	5 kHz
Roll-off Factor α	0.2
Pass band Ripple	0.1-dB
Stop band Attenuation	60-dB

Zero pack the input signal 1-to-20 and form and plot its log magnitude spectrum along with an overlay of the log magnitude spectrum of the prototype interpolating filter. In a subplot, zoom to the spectral region at the edge of the interpolating filter's transition bandwidth to view the rejection level of the replica spectra.

Convolve the zero-packed input signal with the prototype interpolating filter and display in subplots the time response and log magnitude spectrum of the interpolated signal. Verify that the residual spectral levels in the spectra of the interpolated signal have been suppressed at least 60-dB.

Partition the prototype interpolator into a 20-path polyphase filter and up sample the input signal using the polyphase filter structure. Display in subplots the time response and log magnitude spectrum of the interpolated signal. Verify that the signal obtained in the polyphase form of the filter is identical to that obtained by the zero-packed and processed form of the process.

7.2 Repeat Problem 7.1 except, design the prototype low pass filter with multiple stop bands and interleaved don't care bands. Keep the filter length the same for both designs. Be sure to note the difference in the replica attenuation in this design as well as the difference in the shape of the filter impulse response.

7.3 Repeat Problem 7.1 except, this time, cosine heterodyne the impulse response of the prototype interpolating filter to the center of the third Nyquist zone, at 60 kHz. The filter will now simultaneously up sample, reject undesired spectral replicates, and translate.

7.4 Design a FIR filter to be used as a 1-to-40 up sampler for an input signal initially oversampled by a factor of 4. The specifications for this design are as follows:

Sample Rate	800 kHz		
Pass Band	0-4 kHz	Pass band Ripple	0.1-dB
Stop Band	16-400 kHz	Stop band Atten.	60-dB, slope –6-dB/Octave

Plot the impulse response and log magnitude frequency response of the prototype filter to verify that it meets design specifications.

Form an input signal as the impulse response of a Nyquist filter, with the following specifications:

Sample Rate	20 kHz
Symbol Rate	5 kHz
Roll-off Factor α	0.2
Pass band Ripple	0.1-dB
Stop band Atten	60-dB

Zero-pack the input signal 1-to-40 and form and plot its log magnitude spectrum along with an overlay of the log magnitude spectrum of the prototype interpolating filter. In a subplot, zoom to the spectral region at the edge of the interpolating filter's transition bandwidth to view the rejection level of the replica spectra.

Convolve the zero-packed input signal with the prototype interpolating filter and display in subplots the time response and log magnitude spectrum of the interpolated signal. Verify that the residual spectral levels in the spectra of the interpolated signal have been suppressed at least 60-dB.

Partition the prototype interpolator into a 40-path polyphase filter and up sample the input signal using the polyphase filter structure. Display in subplots the time response and log magnitude spectrum of the interpolated signal. Verify that the signal obtained in the polyphase form of the filter is identical to that obtained by the zero-packed and processed form of the process.

7.5 Repeat the filter design of Problem 7.4 except this time we will arrange to up sample by 40 and down sample by 12. We accomplish this by incrementing through the polyphase filter paths in stride of 40/12. Here we form an output sample from every 12th filter path (modulo 40). This accomplishes a sample rate change of 10/3 or 3.333, a rational ratio. Plot the time response and the log magnitude frequency response of the interpolated output time series.

7.6 Repeat the filter design of Problem 7.4 except this time we will arrange to up sample by 3.14159 (look familiar?). We accomplish this by incrementing through the polyphase filter paths in stride of 40/3.14159 or 12.73239..., using the integer part of the index accumulator (modulo 40) to select the nearest neighbor interpolating path. Plot the time response and the log magnitude frequency response of the interpolated output time series.

7.7 Repeat the filter design of Problem 7.4 except this time we will arrange to up sample by 127. We accomplish this by incrementing through the polyphase filter paths in stride of 40/127 or 0.31496..., using the integer part of the index accumulator (modulo 40) to select the nearest neighbor interpolating path. Because the increment is less than 1, there will be 3 and occasionally 4 repeats of a given output sample. Plot the time response and the log magnitude frequency response of the interpolated output time series. Be sure to zoom in on a segment of the interpolated time series to see the repeated output samples.

7.8 Repeat the filter design of Problem 7.4 and then the polyphase partition of the prototype interpolating filter. Use the MATLAB script *polyfit* to form 4th-order polynomial approximations to the individual columns of the partition. Use the vector [0:1/40:1−1/40] to define the values of x for the corresponding values of y, the elements of the columns. To test the quality of the approximation use the MATLAB script *polyval* to evaluate the polynomials over the same input vector values and take the difference between the sample values and the polynomial approximants. Plot the error values.

Now use the polynomials that represent the columns of the original polyphase filter to form a new set of column values over the input grid x2 = [0 1/80:1−1/80]. This forms a filter with impulse response twice the length of the original design. Plot the time response and log magnitude frequency response to see the quality of the polynomial approximation process. Note the levels of the spectral artifacts. Is the order of the polynomial approximation sufficient for this filtering process?

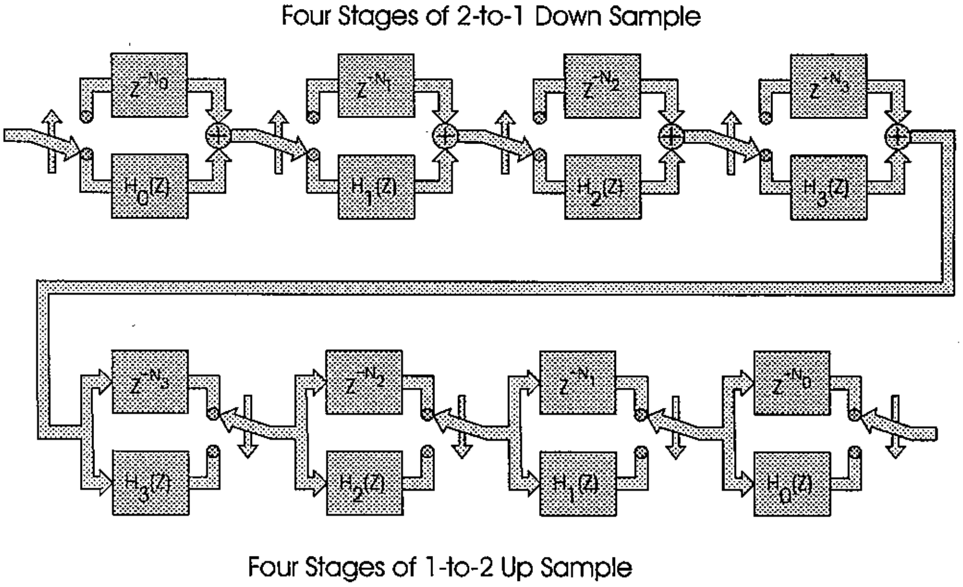
7.9 This problem is a repeat of Problem 7.8 except here we will use 5th-order polynomials to approximate the columns of the polyphase filter. Repeat the filter design of Problem 7.4 and then the polyphase partition of the prototype interpolating filter. Use the MATLAB script *polyfit* to form 5th-order polynomial approximations to the individual columns of the partition. Use the vector [0:1/40:1−1/40] to define the values of x for the corresponding values of y, the elements of the columns. To test the quality of the approximation use the MATLAB script *polyval* to evaluate the polynomials over the same input vector values and take the difference between the sample values and the polynomial approximants. Plot the error values. Is the order of the polynomial approximation sufficient for this filtering process?

7.10 This problem is a repeat of Problem 7.8 except here we will use 3rd-order polynomials to approximate the columns of the polyphase filter. Repeat the filter design of Problem 7.4 and then the polyphase partition of the prototype interpolating filter. Use the MATLAB script *polyfit* to form 3rd-order polynomial approximations to the individual columns of the partition. Use the vector [0:1/40:1−1/40] to define the values of x for the corresponding values of y, the elements of the columns. To test the quality of the approximation, use the MATLAB script *polyval* to evaluate the polynomials over the same input vector values, and take the difference between the sample values and the polynomial approximants. Plot the error values. Is the order of the polynomial approximation sufficient for this filtering process?

7.11 Use the polynomial approximations formed in Problem 7.8 as a replacement for the polyphase interpolator shown in Figure 7.47 and use this process to up sample by 3.14 the impulse response formed in Problem 7.4. We accomplish this by using an increment of 1/3.14 or 0.31847 (modulo–1) of the incrementing accumulator. Plot the time response and the log magnitude frequency response of the time series obtained from the polynomial-based polyphase interpolator.

CHAPTER 8

Half-band Filters



Polyphase M-path filters can be designed with any number of paths. A particularly attractive M-path filter is to be had when the filter has two paths. Many of the properties of multirate filters are easily demonstrated with half-band filters and as such the half-band filter is often used to introduce the reader to multirate filters. From the perspectives and insights gathered from the half-band filter, the M-path filter is presented as a reasonable extension. The half-band filter is used in many configurations including cascade resampling stages, iterated filter stages, and Hilbert transform filters.

8.1 Half-band Low Pass Filters

The prototype half-band low pass filter has the spectral characteristics shown in Figure 8.1.

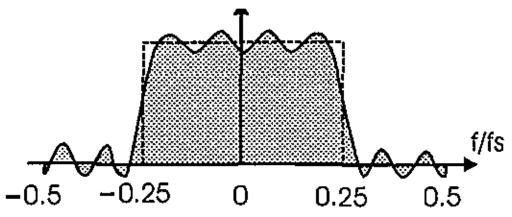


Figure 8.1 Spectral Characteristics of Half-band Low Pass Filter

The filter is designed for a pass band bandwidth between the ± 1/4 sample rate for a two-sided bandwidth equal to half the sample rate. The finite length implementation of the filter has a transition bandwidth that passes through the midpoint, 0.5, of the discontinuity at ± quarter sample rate.

The impulse response of the ideal noncasual continuous filter with two-sided bandwidth fs/2 is shown in (8.1).

$$h_{LP}(t) = \frac{f_S}{2}\frac{\sin(\frac{2\pi f_S/2}{2}t)}{(\frac{2\pi f_S/2}{2}t)} \tag{8.1}$$

When sampled at multiples on nT or n/fs, we obtain, after scaling by fs, the sample data form of the ideal half-band filter as shown in (8.2) and canceling terms results in the form shown in (8.3).

$$h_{LP}(n) = \frac{f_S/2}{f_S} \frac{\sin(\frac{2\pi f_S/2}{2}\frac{n}{f_S})}{(\frac{2\pi f_S/2}{2}\frac{n}{f_S})} \tag{8.2}$$

$$h_{LP}(n) = \frac{1}{2} \frac{\sin(\frac{n\pi}{2})}{(\frac{n\pi}{2})} \tag{8.3}$$

The impulse response of the half-band filter formed by the sampling operation contains one sample at the point of symmetry with matching left and right samples about the symmetry point, as shown in Figure 8.2.

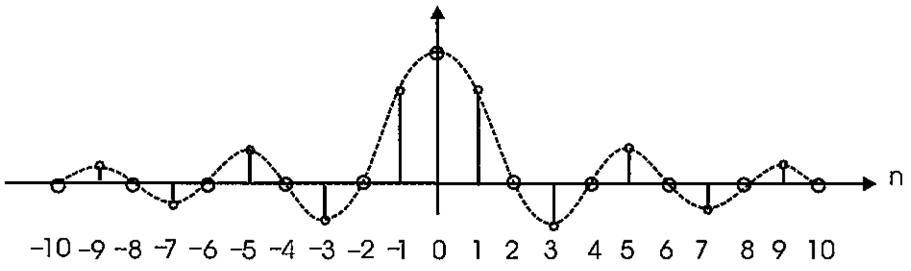


Figure 8.2 Impulse Response of Noncausal Half-band Low Pass Filter

The impulse response samples of the half-band filter are seen to be zero at the even index offsets from the center point of the filter. The coefficients with odd index offset are seen to exhibit even symmetry about the filter center point. These two properties permit the non polyphase form of the filter of length 2N+1 to be implemented with only N/2 multiplications per output sample. When partitioned into a two-path filter and used for down sampling, the N/2 multiplications are distributed over two input samples for a workload of N/4 multiplications per input point in the 2N+1-length filter.

8.2 HALF-BAND HIGH PASS FILTER

The half-band high pass filter has the spectral form shown in Figure 8.3.

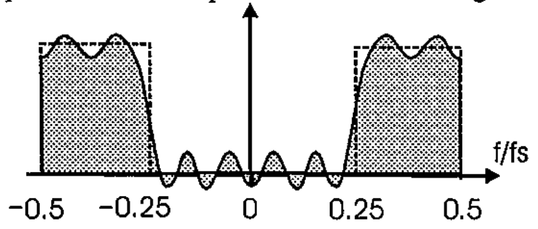


Figure 8.3 Spectral Characteristics of Half-band High pass Filter

The impulse response of the high pass half-band filter is obtained by heterodyning the low pass impulse response to the half sample rate. This is shown in (8.4) and (8.5).

$$h_{HP}(n) = h_{LP}(n)\, e^{jn\pi} \tag{8.4}$$

$$h_{HP}(n) = \frac{1}{2}\frac{\sin(\frac{n\pi}{2})}{(\frac{n\pi}{2})}\cos(n\pi) \tag{8.5}$$

The heterodyne indicated in (8.5) is a sequence of unit-valued samples with alternating signs. The unit-valued samples with a positive sign correspond to the even offset index samples of the low pass filter, which are zero except for the sample at the symmetry point. The unit-valued samples with a negative sign correspond to the odd offset index samples of the low pass filter. The heterodyne to the half sample rate has the effect of reversing the signs of all the samples of the low pass filter except for the sample at the symmetry point. The resultant impulse response is shown in Figure 8.4.

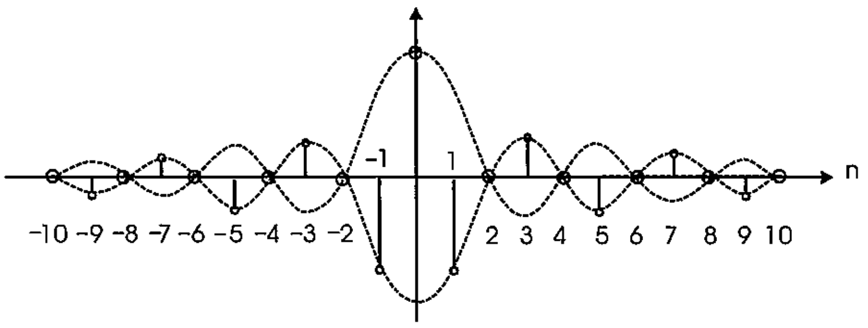


Figure 8.4 Impulse Response of Noncausal Half-band High pass Filter

We note that the sum of the impulse responses of the low pass half-band filter and high pass half-band filter is a single unit-valued sample at its point of symmetry. The canceling of all but one of the weights in the summation of the two impulse responses is shown in (8.6) for the noncausal form of the filter and in (8.7) for the causal form.

$$h_{LP}(n) + h_{HP}(n) = \delta(n) : -N \le n \le +N \tag{8.6}$$

$$h_{LP}(n) + h_{HP}(n) = \delta(n-N) : \ 0 \le n \le +2N \tag{8.7}$$

Equations (8.6) and (8.7) tell us that the low pass and high pass half-band filters are complimentary filters and satisfy the relationships shown in equations (8.8) and (8.9).

$$h_{HP}(n) = \delta(n) - h_{LP}(n) : \ -N \le n \le +N \tag{8.8}$$

$$h_{HP}(n) = \delta(n-N) - h_{LP}(n) : \ 0 \le n \le +2N \tag{8.9}$$

8.3 Window Design of Half-band Filter

The impulse response of a half-band filter can be obtained by windowing the sinc(n) series presented in (8.3). This is shown in (8.10). The window length must be selected to obtain the specified transition bandwidth and the window shape must be chosen to obtain the specified stop-band side-lobe level. The design rules developed in Chapter 3 for the Kaiser window can be easily applied to the windowed design of the half-band filter.

$$h_{LP}(n)=\frac{1}{2}\frac{\sin(\frac{n\pi}{2})}{(\frac{n\pi}{2})}w(n): \quad -N\le n\le +N \tag{8.10}$$

The coefficients in the half-band filter will be partitioned into two paths of a polyphase filter. One path will contain coefficients with indices 2n while the other path will contain coefficients with indices 2n+1. In order for the upper path to contain the zero-value samples and the center tap of the half-band filter of length 2N+1, N must be an even number. When N is an even number we are able to invoke the noble identity and pull the 1-to-2 up sampler at the input to a half-band filter through the filter to perform the up sampling at the filter output. We can always append a zero to the two ends of the filter when the N of a 2N+1 point filter is odd. The appended zero-value endpoints in the filter do not affect the workload of the filter .but do assure that the upper path of a half-band polyphase partition contain an integer number of delays to the filter's midpoint.

The MATLAB script to design a half-band filter and the resultant filter is shown in Design Example 8.1 and in Figure 8. 5.

Design Example 8.1

Design a half-band filter with the following specifications:

Sample Rate	20 kHz
Transition BW	4 kHz
Out-of-Band Attenuation	60-dB

The estimated filter length is obtained from (3.11) in Chapter 3 and repeated here in (8.11).

$$N=\frac{f_S}{\Delta f}\frac{Atten(dB)-8}{14}=\frac{20}{4}\frac{60-8}{14}=18.6\Rightarrow 19 \tag{8.11}$$

The parameter β of the Kaiser window is estimated from Figure 3.8 of Chapter 3 to be 5.8.

The MATLAB script for the half-band design is shown next. We increased the length of the filter by two samples from N=19 to N=21 to assure that the center point of the filter is located in the upper path of a two path partition.

```
h=0.5*sinc(5:0.5:5).*kaiser(21,5.8)';
```

The impulse response and frequency response of this filter are shown in Figure 8.5. The spectrum is seen to satisfy the filter specifications as well as exhibit the window-related 1/f fall-off rate of side-lobes in the pass band and stop band.

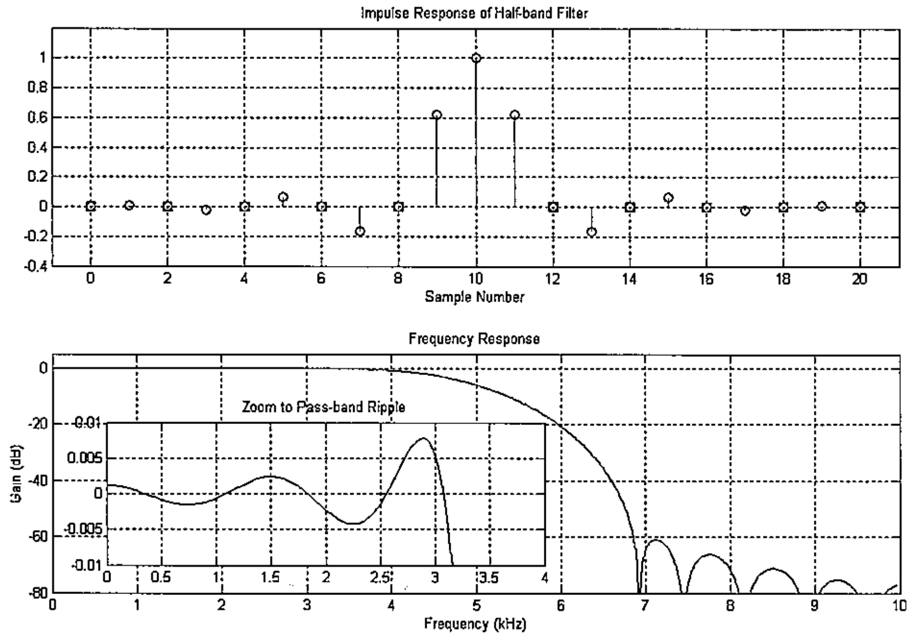


Figure 8.5 Impulse Response and Frequency Response of Half-band Low pass Filter Designed with Kaiser Window

8.4 Remez Algorithm Design of Half-band Filters

The Remez algorithm performs the filter design in the frequency domain. The impulse response of a half-band filter is restricted to have all but one of the even indexed samples be equal to zero. This restriction can be folded into the Remez algorithm by restricting the filter specifications to have equal offsets of pass band and stop band edges from the quarter sample rate and equal weight pass band and stop band penalty functions. A MATLAB script that uses a call to the Remez algorithm to satisfy the specifications for the example presented in the previous section is shown here.

```
h2=remez(20,[0 5-2 5+2 10]/10,[1 1 0 0],[1 1]);
```

The impulse response and frequency response of this filter are shown in Figure 8.6. The spectrum is seen to satisfy the filter specifications as well as exhibit the equal ripple side-lobes in the pass band and stop band. Note the side-lobe levels are lower than those obtained in the window design shown in Figure 8.5.

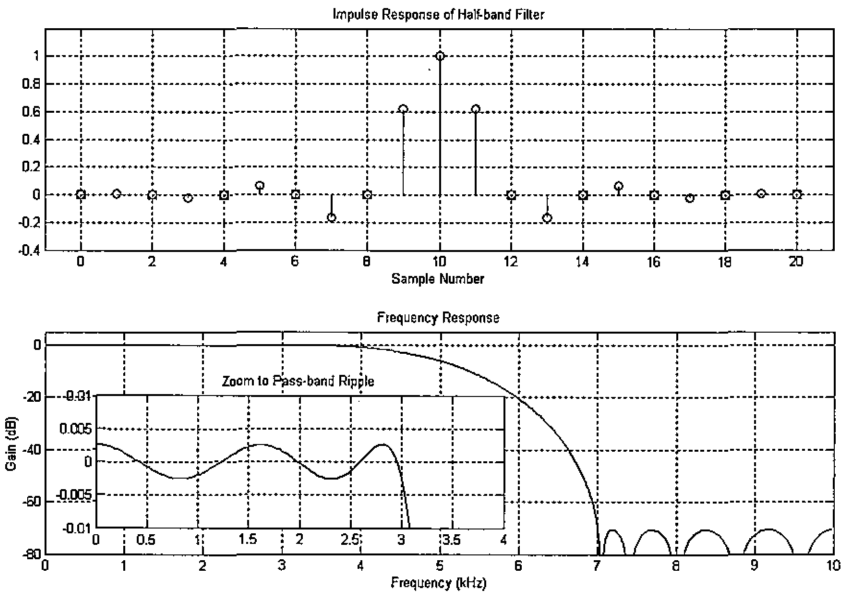


Figure 8.6 Impulse Response and Frequency Response of Half-band Low pass Filter Designed with Remez Algorithm

8.4.1 Half-band Remez Algorithm Design Trick

In the previous section, we obtained the zero-valued samples of the half-band filter impulse response by restricting the band edge specifications to be symmetric about the quarter-sample rate. A half-band design trick can be used to have the Remez algorithm design only the odd-indexed coefficients of the desired half-band filter. We finish the design by inserting the even-indexed samples composed of the zero-valued samples and the center tap sample. The odd-indexed samples form a one-band filter with a transition bandwidth at the half sample rate. We ask the Remez algorithm to design this filter, then modify it. A MATLAB script that calls the Remez algorithm to form the design trick impulse response matching the example presented in the previous two sections is shown here.

```
h3a=remez(9, [0 5-2 5 5]/5, [1 1 0 0]);
h3 = zeros(1, 21);
h3(2:2:21)=0.5*h3a;
h3(10)=0.5;
```

The impulse response and frequency response of the h3a, the Remez-designed filter is shown in Figure 8.7. The spectrum is seen to be the pass band of the desired filter with half its transition band.

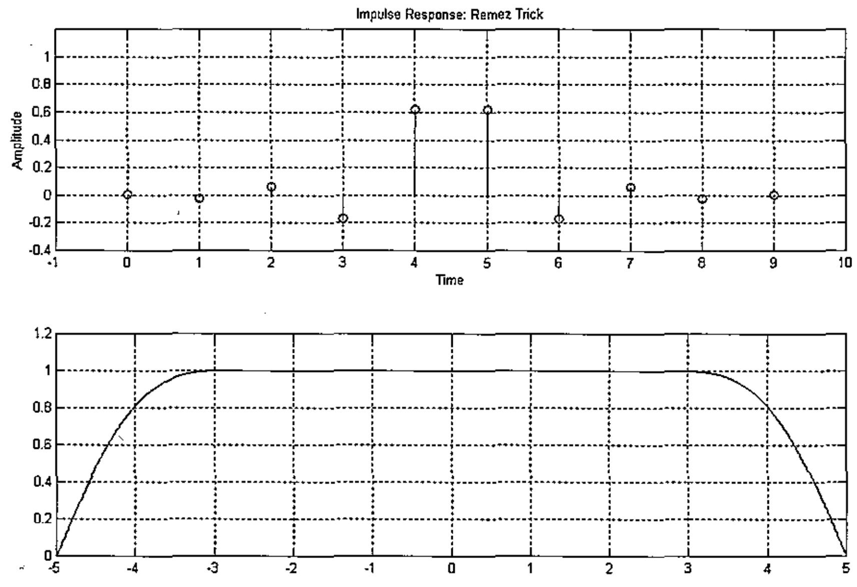


Figure 8.7 Impulse Response and Frequency Response of Trick One-Band Filter Designed with Remez Algorithm

The impulse response and frequency response of h3, the corrected filter response is shown in Figure 8.8. The correction scales the filter by 0.5 and inserts the zero-valued samples at the even-indexed positions with a single 0.5-valued sample at the filter center position. The spectrum is seen to now contain both the pass band and stop band of the filter.

The zero packing is seen to double the filter sample rate and offer two copies of the spectrum, with the copy at the half sample rate exhibiting 180-degree phase shift. The scaling by 0.5 sets the gain in the two spectral regions to nominal values of +0.5 and –0.5. The inserted 0.5 at the filter midpoint raises the spectrum by amplitude 0.5. The effect of the raised spectrum is that in the spectral interval centered at 0 frequency the added 0.5 raises the 0.5 gain to the pass band gain with a nominal value of 1.0 while in the spectral interval centered at the half-sample rate the added 0.5 raises the –0.5 gain to the stop band gain a nominal value of 0.0.

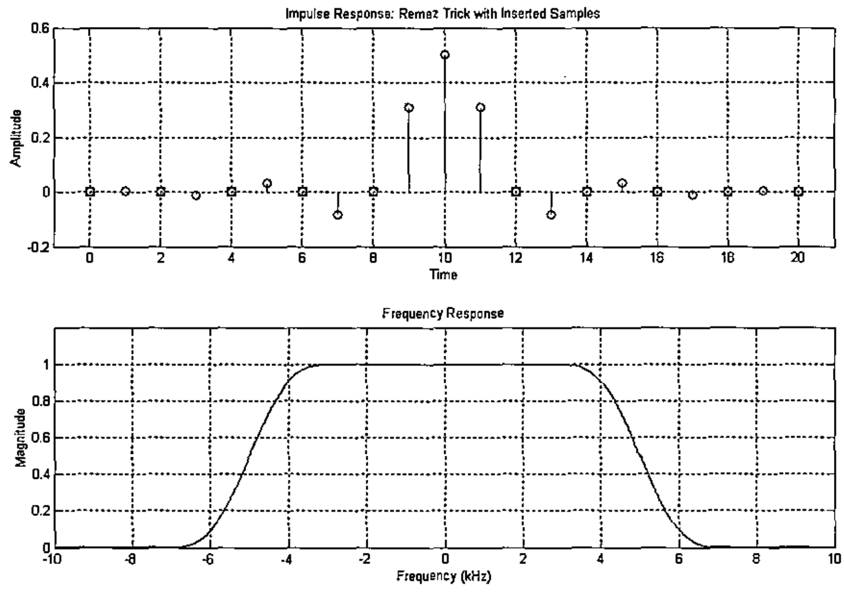


Figure 8.8 Impulse Response and Frequency Response of Trick One-Band Filter with Inserted Zero Samples and Symmetry Point Sample

8.5 Hilbert Transform Band-pass Filter

The half-band filters described and designed in the previous sections have been low pass and high pass filters. A trivial variation of this design leads to a half-band filter that performs the Hilbert transform. The Hilbert transform has a frequency response with a nominal unity gain over the positive frequencies and nominal zero gain over the negative frequencies. The frequency response of the Hilbert transform form of the half-band filter is shown in Figure 8.9.

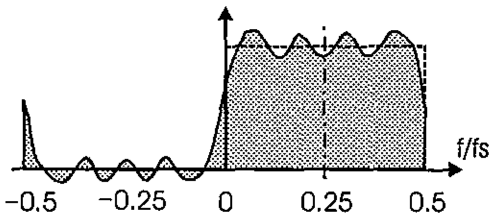


Figure 8.9 Spectral Characteristics of Hilbert Transform Half-band Filter

The spectrum of the Hilbert transform filter is obtained by translating the low pass half-band filter to the quarter-sample rate. This form of this translation for a noncausal filter is shown in (8.12).

$$h_{HT}(n) = h_{LP}(n)\, e^{jn\pi/2} \;:\; -N \le n \le +N \tag{8.12}$$

The heterodyne indicated in (8.12) could be visualized as a quadrature heterodyne with a cosine and a sine series as shown in (8.13).

$$h_{HT}(n) = h_{LP}(n)\, \{\cos(n\pi/2) + j\sin(n\pi/2)\} : \quad -N \le n \le +N \tag{8.13}$$

We note that the cosine component of the heterodyne is a sequence comprised of unit amplitude samples with alternating signs on the even indices and zero-valued samples on the odd indices. The sequence is of the form { 1 0 –1 0 }. The half-band filter is zero-valued at the even indices except for the sample at the origin, the symmetry point. Thus the cosine heterodyne, the real part of the heterodyne, contains a single non-zero sample at the origin. The sine component of the heterodyne is a sequence comprised of unit amplitude samples with alternating signs on the odd indices and zero-valued samples on the even indices. The sequence is of the form{ 0 1 0 –1 }. The odd indexed samples of the half-band filter contain the alternating sign side-lobe samples of the sinc(n π/2) series. The samples of the sine heterodyne interact with the samples of the sinc sequence to remove the alternating signs of the sequence. The complex impulse response and the corresponding spectrum are shown in Figure 8.10.

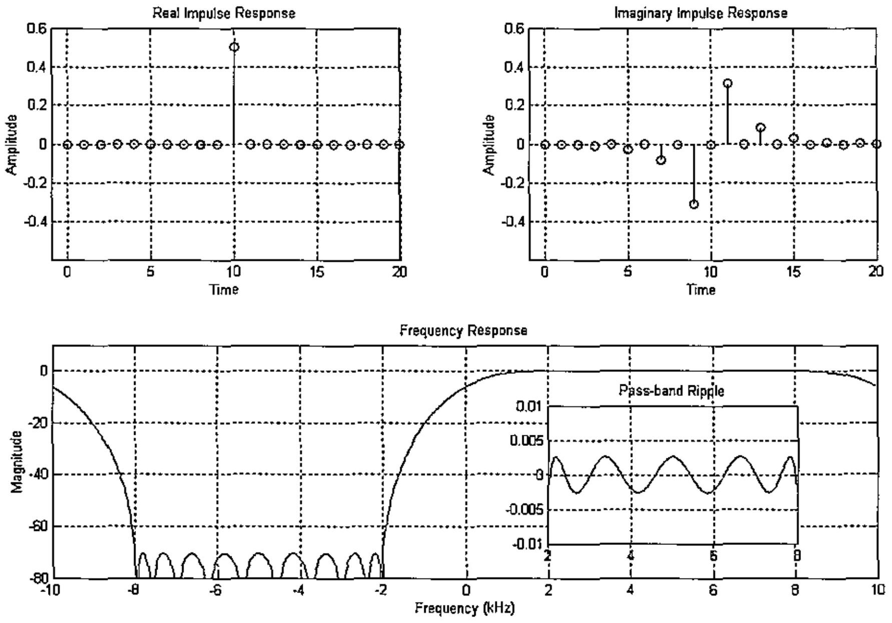


Figure 8.10 Impulse Response and Frequency Response of Half-band Filter Designed by Remez Algorithm and Converted to Hilbert Transform by Complex Heterodyne to fs/4

8.5.1 Applying the Hilbert Transform Filter

The discrete Hilbert transform filter can be used to generate a complex signal from a real input sequence by eliminating the spectral components residing in the negative frequency band. After suppressing the negative frequencies, thus reducing the bandwidth by a factor of 2, it is common to also reduce the sample rate by the same factor. Thus the Hilbert transform filter can be used as a half-band multirate filter. As an aside, this differs from the conventional Weaver modulator that translates the center of the positive frequency interval to baseband for processing by the half-band filter. The Hilbert transform filter translates the baseband half-band filter to the center of the positive frequency interval. This is the "which do we move, filter or signal spectrum" question we discussed in Chapter 6. Moving the filter to the spectral center is equivalent to moving the spectral center to the filter. Figure 8.11 illustrates spectral translation that forms the Hilbert transform filter.

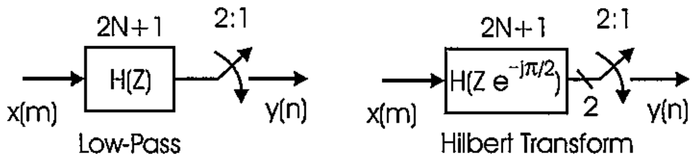


Figure 8.11 Hilbert Transform Filter as a Spectrally Translated Version of Low Pass Half-band Filter

The complex heterodyne of the low pass filter's impulse response results in a filter with complex impulse response. Such a filter can be formed as a two-path filter, one forming the real part and one forming the imaginary part of the impulse response. This form of the Hilbert transform filter is shown in Figure 8.12.

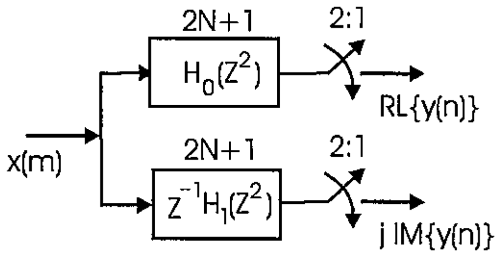


Figure 8.12 Two-path Model of Hilbert Transform Filter with Complex Impulse Response

Due to the zeros of the quarter sample rate cosine and sine, the impulse response of the upper path is seen to be zero at the even indices while the impulse response of the lower path is seen to be zero at the odd indices. The zeros permit application of the noble identity in which we interchange the filter with the 2-to-1 resample to operate the filters at the reduced output sample rate. This interchange is demonstrated in Figure 8.13.

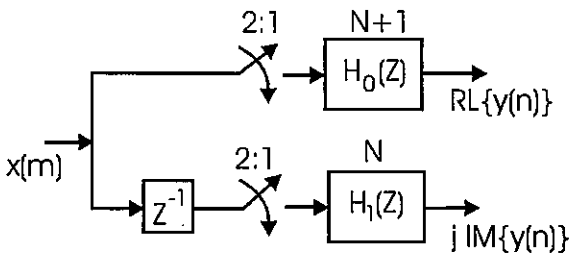


Figure 8.13 Noble Identity Applied to Hilbert Transform Filter

Finally, the interaction of the pair of 2-to-1 resampler switches and the input delay line can be replaced with a 2-input commutator that performs the same function. This form of the resampling half-band Hilbert transform filter is shown in Figure 8.14.

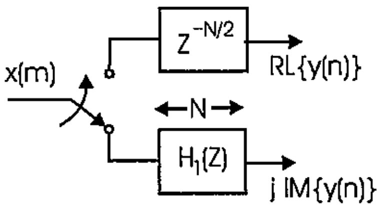


Figure 8.14 Two-path Commutator -driven Hilbert Transform Filter

8.6 Interpolating with Low Pass Half-band Filters

The half-band low pass filter can be used to up sample a time series by a factor of two. The initial form of the1-to-2 up sampling process, based on zero insertion to raise the input sample rate, is shown in Figure 8.15.

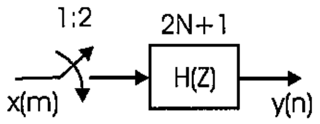


Figure 8.15 One-to-Two Up Sampling Process with Half-band Filter

The half-band filter H(Z) can be partitioned into a pair of polyphase filters as shown in (8.14) and (8.15) and illustrated in Figure 8.16.

$$H(Z)=\sum_{n=0}^{2N} h(n)\, Z^{-n} \tag{8.14}$$

$$H(Z)=\sum_{n=0}^{N} h(2n)\, Z^{-2n}+Z^{-1}\sum_{n=0}^{N-1} h(2n+1)\, Z^{-2n} \tag{8.15}$$

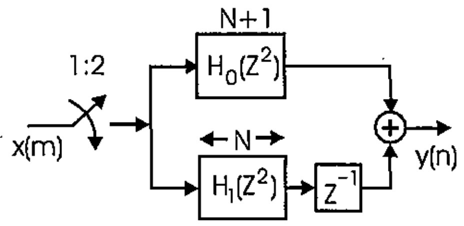


Figure 8.16 One-to-Two Up Sampling with Polyphase Half-band Filter

The order of the resampling and the filtering performed in the filter can be reversed leading to the form shown in Figure 8.17.

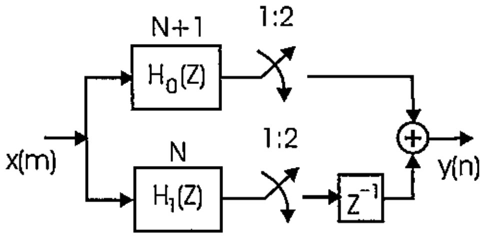


Figure 8.17 One-to-Two Up Sampling at Output of Polyphase Half-band Filter.

Finally, as shown in Figure 8.18, we can replace the pair of 1-to-2 up sampling switches and sample delay with a two-tap commutator that performs the equivalent scheduling of path outputs to the output sample stream.

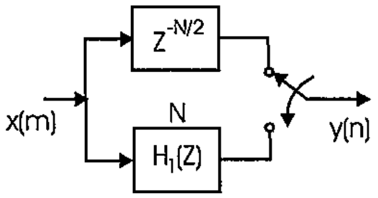


Figure 8.18 One-to-Two Up Sampling with Commutated Two-path Half-band Filter.

We note that the prototype filter is designed with 2N+1 taps and then partitioned into two paths with one path containing the zero-valued samples and implemented as a delay-only path with the other path containing the remaining N non-zero samples. The filter requires N arithmetic operations (ops) to generate two output samples in response to each input sample. When we distribute the N ops per input over the two outputs, we find the filter workload is N/2 ops per output. The number of multiplies per output can be reduced by another factor of 2 by taking advantage of the even symmetry of the coefficient set in the lower path filter.

The question we now address is what is the workload per output for a half-band resampling filter operating at a particular set of performance specifications. We can examine Figure 8.19 to see the performance parameters that affect the filter length and workload per output point.

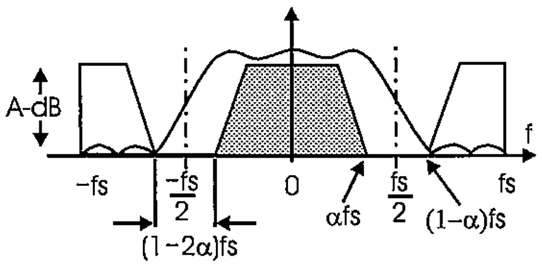


Figure 8.19 Spectral Characteristics of Half-band Filter Suppressing Spectral Replicate

The length of the filter shown in Figure 8.19 can be estimated from the harris approximation presented in Chapter 3 and reproduced here as (8.16).

$$N \cong \frac{f_{SMPL}}{\Delta f}\,\frac{Atten(dB)}{22} \tag{8.16}$$

Substituting the transition bandwidth and sample rate shown in Figure 8.19 into (8.16) we obtain (8.17).

$$\begin{aligned} N &\cong \frac{2f_S}{(1-2\alpha)f_S}\,\frac{Atten(dB)}{22} \\ &= \frac{2}{(1-2\alpha)}\,\frac{A(dB)}{22} = K_1(\alpha)\frac{A(dB)}{22} \end{aligned} \tag{8.17}$$

The relationship between filter length and fractional bandwidth is shown in Figure 8.20. We can clearly see that as the fractional bandwidth of the filter increases, which causes a decrease of transition bandwidth, the filter length increases. For a specific example, consider a half-band filter with fractional bandwidth of 0.45 with 60-dB attenuation. From Figure 8.20, we can estimate the half-band filter length N_{Len} as 20 60/22 or 54.54…taps which gets converted into 57 taps in order to locate the center tap in the upper path of a two-path partition.

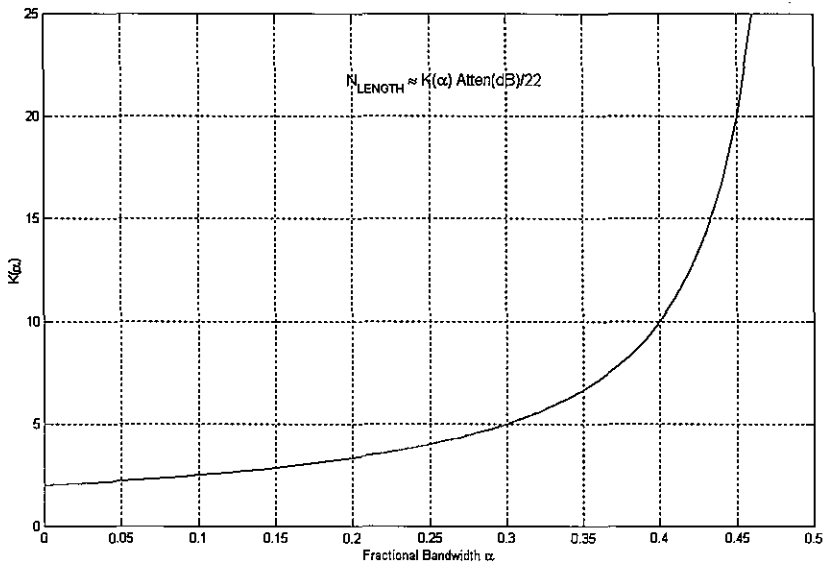


Figure 8.20 Filter Length Parameter K(α) as Function of Fractional Bandwidth α

8.7 DYADIC HALF-BAND FILTERS

We now consider the use of a cascade of half-band filters to obtain a sample rate increase of any power of 2 such as increase by 8 or by 16. Suppose, for instance, we want to increase the sample rate of an input sequence by a factor of 8. We have two primary options available to us. We can use an 8-path polyphase filter to accomplish this task, or we can use a cascade of three half-band filters. We first examine the workload for the sequence of half-band filters and then compare this workload to the M-path filter. The sequence of half-band filters operates at successively higher sample rates but with transfer functions that have successively wider transition bandwidths. There is a processing advantage to the cascade when the reduction in processing due to the wider transition bandwidth in successive filter stages compensates for operating the consecutive filter stages at successively higher sample rates.

Figure 8.21 is a block diagram showing four stages of half-band filters that raise the sample rate by a factor of 16. Shown here is the length of each polyphase filter stage as well as an index indication of how often each stage is used each time a data sample is delivered to the input port.

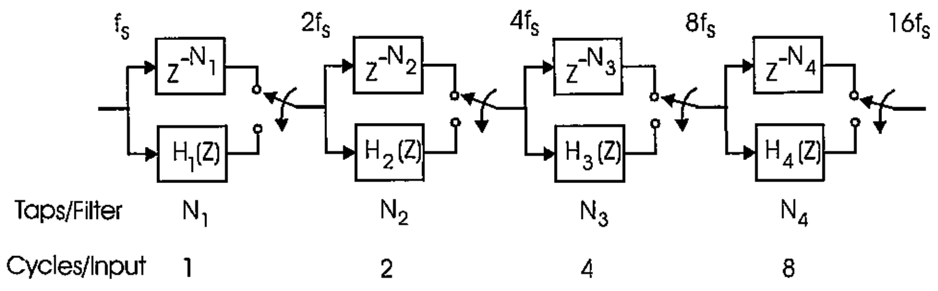


Figure 8.21 Cascade of Four Half-band Filters to Raise the Sample Rate by 16 in Steps of Increase by 2 per Stage

The frequency response of successive filters in the cascade is shown in Figure 8.22. Here we see that the successive filters operate at twice the sample rate of the previous stage and have transition bandwidths successively wider than the previous stage.

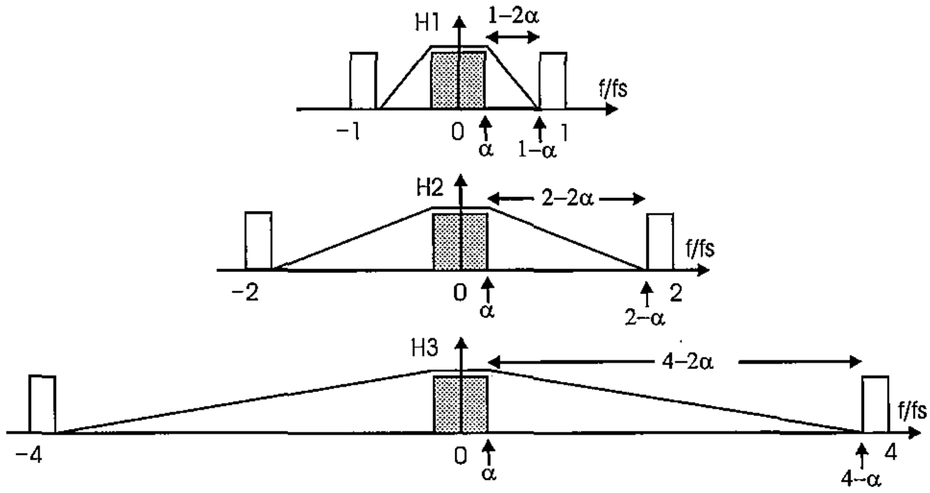


Figure 8.22 Frequency Plan of Three Successive Half-band Filters Showing Transition Bandwidth and Sample Rate

As shown in (8.16), the length of successive filters is proportional to the ratio of their sample rates to their transition bandwidths. A sequence of proportionality factors for the first four filters is shown in Table 8-1. Also indicated is the workload performed by each filter in response to an input sample delivered to the input of the cascade.

Table 8-1 Length of Four Successive Half-band Filters and Number of Arithmetic Operations Performed by Each Filter per Input Sample

	First Stage	Second Stage	Third Stage	Fourth Stage
$\frac{f_S}{\Delta f}$	$\frac{2}{1-2\alpha}$	$\frac{4}{2-2\alpha}$	$\frac{8}{4-2\alpha}$	$\frac{16}{8-2\alpha}$
Ops/Input $K_1(\alpha)$	$\frac{2}{1-2\alpha}$	$2\frac{4}{2-2\alpha}$	$4\frac{8}{4-2\alpha}$	$8\frac{16}{8-2\alpha}$

Table 8-2 lists the workload proportionality factors for the sequence of cascade stages containing one-to-four stages of half-band filtering. The number of output samples has been normalized so that the proportionality factors in this list have units of ops/output sample.

Table 8-2 Workload Proportionality Factor per Output Sample for Cascade of Half-band Filters

	Workload Proportionality Factor per Output Sample
One Stage $K_2(\alpha)$	$\left[\frac{2}{1-2\alpha}\right]\frac{1}{2}$
Two Stages $K_2(\alpha)$	$\left[\frac{2}{1-2\alpha}+2\frac{4}{2-2\alpha}\right]\frac{1}{4}$
Three Stages $K_2(\alpha)$	$\left[\frac{2}{1-2\alpha}+2\frac{4}{2-2\alpha}+4\frac{8}{4-2\alpha}\right]\frac{1}{8}$
Four Stages $K_2(\alpha)$	$\left[\frac{2}{1-2\alpha}+2\frac{4}{2-2\alpha}+4\frac{8}{4-2\alpha}+8\frac{16}{8-2\alpha}\right]\frac{1}{16}$

Table 8-3 presents essentially the same information as Table 8-2 but has factored the normalization factor into the terms in each sum. This form of presenting the proportionality factor emphasizes the reduction in contribution of early stages in the cascade to each output of the cascade as the number of stages, hence the number of output samples, increases. With a higher sample rate increase, the input workload can be amortized over a greater number of output samples. Note, for instance, the contribution of the first stage to the workload per output sample is reduced by a factor of 2 each time another stage is added to the cascade.

Table 8-3 Alternate Presentation of Workload Proportionality Factor per Output Sample for Cascade of Half-band Filters

	Workload Proportionality Factor per Output Sample
One Stage $K_2(\alpha)$	$\left[\frac{1}{1-2\alpha}\right]$
Two Stages $K_2(\alpha)$	$\frac{1}{2}\left[\frac{1}{1-2\alpha}\right]+\left[\frac{1}{1-\alpha}\right]$
Three Stages $K_2(\alpha)$	$\frac{1}{4}\left[\frac{1}{1-2\alpha}\right]+\frac{1}{2}\left[\frac{1}{1-\alpha}\right]+\left[\frac{1}{1-\alpha/2}\right]$
Four Stages $K_2(\alpha)$	$\frac{1}{8}\left[\frac{1}{1-2\alpha}\right]+\frac{1}{4}\left[\frac{1}{1-\alpha}\right]+\frac{1}{2}\left[\frac{1}{1-\alpha/2}\right]+\left[\frac{1}{1-\alpha/4}\right]$

Figure 8.23 presents curves of the expressions shown in Tables 8-2 and 8-3. We see that as the number of stages increases, the proportionality factor $K_1(\alpha)$ in ops per input sample increases, but when distributed over the number of output samples, the factor $K_2(\alpha)$ in ops per output sample is seen to decrease. We also see that the factor $K_2(\alpha)$ in ops per output sample asymptotically approaches 2 and in fact is close to 2 for a wide range of the fractional bandwidth and number of stages.

For comparison, we can use an M-path polyphase filter to change the sample rate by a factor of M. We can recast equations (8.16) and (8.17) for the M-path filter to obtain (8.18).

$$\begin{aligned} N &\cong \frac{M f_S}{(1-2\alpha) f_S}\frac{Atten(dB)}{22} \\ &= \frac{M}{(1-2\alpha)}\frac{A(dB)}{22} = K_3(\alpha)\frac{A(dB)}{22} \end{aligned} \tag{8.18}$$

As shown in (8.19) we can determine the length of each path of the M-path filter by distributing the N weights over the M-paths. If we assume that the top path, path-0, of the M-path filter contains only delays, then only (M–1) of the paths contributes to the workload, and removing one of the M-paths from the workload estimate reduces the average workload. This scaled workload is shown in (8.20). Figure 8.24 presents graphical representations of (8.18) and (8.20).

$$\frac{N}{M} = \frac{1}{(1-2\alpha)}\frac{A(dB)}{22} \tag{8.19}$$

$$\frac{N}{M}(1-\frac{1}{M}) = \frac{1}{(1-2\alpha)}(1-\frac{1}{M})\frac{A(dB)}{22}$$
$$= K_4(\alpha)\frac{A(dB)}{22} \tag{8.20}$$

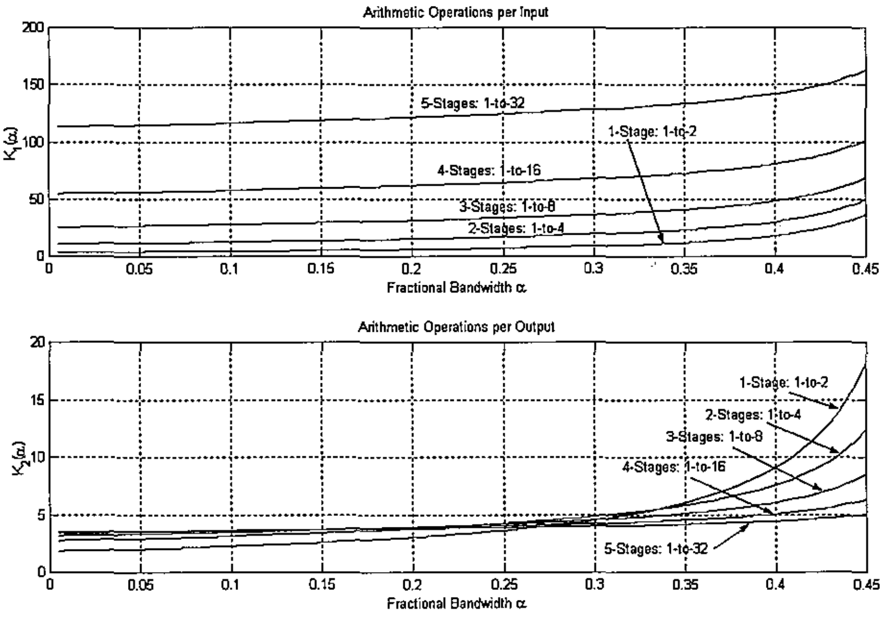


Figure 8.23 Proportionality Factor $K_1(\alpha)$ in Ops/Input and $K_2(\alpha)$ in Ops/Output for One to Five Cascade Stages of Half-band Filters.

We see in Figure 8.24 that the workload proportionality factor $K_4(\alpha)$ in ops per output point in the M-path filter always exceeds the factor $K_4(\alpha)$ of a two-path filter. By contrast, the related proportionality factor $K_2(\alpha)$ for the multiple half-band stages shown in Figure 8.23 is almost everywhere below the factor $K_2(\alpha)$ of the single half-band or two path filter. We thus conclude that the cascade half-band filter is less costly in ops/output point than the M-path filter for the same interpolation factor. The ratio of workloads for the two options, cascade half-band filters and the M-path filter, approaches 1-to-10 for large resampling ratios.

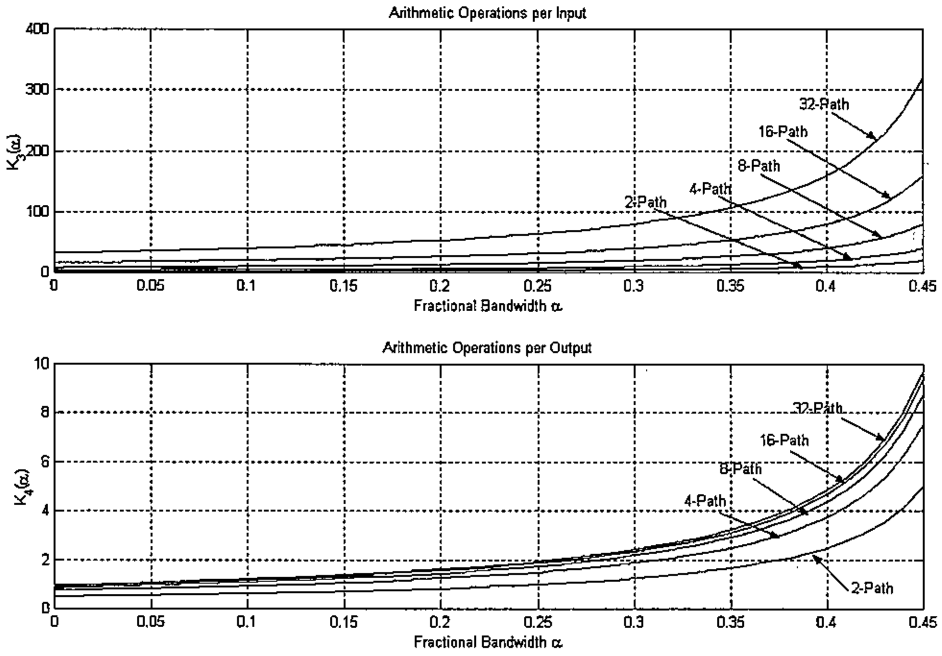


Figure 8.24 Proportionality Factor $K_3(\alpha)$ in Ops/Input and $K_4(\alpha)$ in Ops/Output in M-Path Filter for Values of M = 2, 4, 8, 16, and 32

References

Crochiere, Ronald, and Lawrence Rabiner, *Multirate Signal Processing*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1983.

Fliege, Norbert, *Multirate Digital Signal Processing: Multirate Systems, Filter Banks, Wavelets*, West Sussex, John Wiley & Sons, Ltd., 1994.

Hentschel, Tim, *Sample Rate Conversion in Software Configurable Radios*, Norwood, MA, Artech House, Inc., 2002.

Jovanovic-Dolecek, Gordana, *Multirate Systems: Design and Applications*, London, Idea Group, 2002.

Mitra, Sanjit, and James Kaiser, *Handbook for Digital Signal Processing*, New York, John Wiley & Sons, 1993.

Mitra, Sanjit, *Digital Signal Processing: A Computer-Based Approach*, 2nd ed., New York, McGraw-Hill, 2001.

Vaidyanathan, P. P. and Ngyuen, T. O., "A Trick for the Design of FIR Half-band Filters," *IEEE Trans. on Circuits and Systems* - II, Vol. 34, Mar. 1987, pp. 297 - 300.

Vaidyanathan, P. P., *Multirate Systems and Filter Banks*, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1993.

Problems

8.1 Use the MATLAB *sinc* script file to form a 21-tap low pass, half-band filter.

Use subplots to stem the impulse response and to plot the magnitude spectrum and then the log magnitude spectrum of the filter. What is the DC gain of the filter? What is the appropriate scale factor to apply to the impulse response to set the DC gain to unity? Note the location of the impulse response zeros. After scaling, note the value of normalized frequency corresponding to amplitude 0.5.

8.2 Use the MATLAB *sinc* script file to form a 23-tap low pass, half-band filter.

Use subplots to stem the impulse response and to plot the magnitude spectrum and then the log magnitude spectrum of the filter. What is the DC gain of the filter? What is the appropriate scale factor to apply to the impulse response to set the DC gain to unity? Note the locations of the impulse response zeros. Compare the zero locations of this filter to the zero locations of the filter in Problem 8.1. What can you say about the length of the filter to set the first sample to zero? This is a desired property for a polyphase partition! After scaling, note the value of normalized frequency corresponding to amplitude 0.5.

8.3 Use the MATLAB *sinc* script file to form a 22-tap low pass, half-band filter.

Use subplots to stem the impulse response and to plot the magnitude spectrum and then the log magnitude spectrum of the filter. What is the DC gain of the filter? What is the appropriate scale factor to apply to the impulse response to set the DC gain to unity? Note the locations of the impulse response zeros. Where did they go? What can you say about the length of the filter to have alternate values of the impulse response be zero? This is a desired property for a half-band filter and for a polyphase partition! After scaling, note the value of normalized frequency corresponding to amplitude 0.5.

8.4 Design a low pass, half-band FIR filter using a windowed *sinc* function to obtain a transition bandwidth of 0.1 and a stop band attenuation of 60-dB. Use subplots to plot the impulse response and the log magnitude spectrum of the filter.

8.5 Design a low pass, half-band FIR filter using the Remez algorithm to obtain a transition bandwidth of 0.1 and a stop band attenuation of 60-dB. Use subplots to plot the impulse response and the log magnitude spectrum of the filter.

8.6 Design a low pass, half-band FIR filter using the Remez algorithm trick to obtain a transition bandwidth of 0.1 and a stop-band attenuation of 60-dB. Use subplots to plot the impulse response and the log magnitude spectrum of the filter.

8.7 Design a low pass, half-band FIR filter using the Remez algorithm to obtain a transition bandwidth of 0.1 and a stop band attenuation of 60-dB. Heterodyne the impulse response to center the band center at the quarter-sample rate. Use subplots to plot the real and imaginary part of the impulse response and the log magnitude spectrum of the filter.

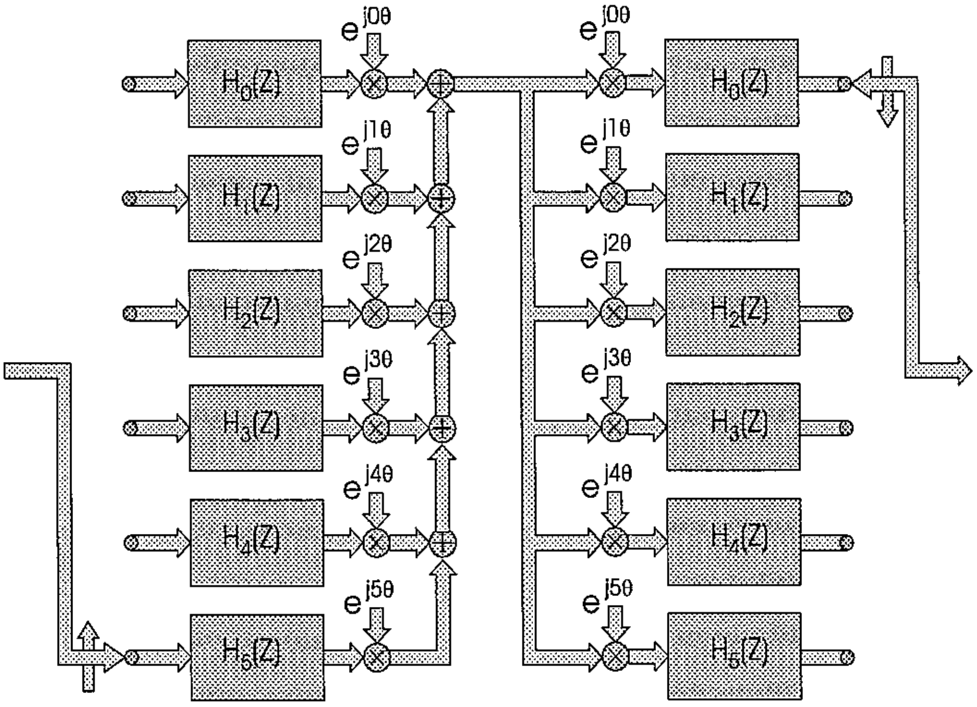
8.8 Repeat the design of the Hilbert transform filter described in Problem 8.7 and then apply to the filter 200 samples a real cosine of frequency 0.11. Plot the input and output time series (real and imaginary) and the windowed log magnitude spectrum of the input and output sequences.

8.9 Repeat the design of the Hilbert transform filter described in Problem 8.7 and then apply to the filter 200 samples of a real cosine of frequency 0.11. Plot the absolute value of the input and output time series. Suggest how the Hilbert transform can be used to estimate signal amplitude for operation of an AGC circuit.

8.10 We are to design a 1-to-4 up sampler as a cascade of two half-band filters. The input signal is the 200 samples of equal amplitude random-phase sinusoids of normalized frequency 0, 0.05, 0.11, 0.155, 0.21, and 0.26. Design two half-band filters that will reject spectral replicates of this signal in two steps. Apply a 1-to-2 zero-packed version of the input signal to the first half-band filter. Plot the time response of the first filter output and the input and output log-magnitude spectrum of the input and output signals. Zero-pack 1-to-2 the output of the first filter and deliver the zero-packed sequence to the second half-band filter. Plot the time response of the second filter output and the input and output log-magnitude spectrum of the input and output signals.

CHAPTER 9

Polyphase Channelizers



We have discussed and demonstrated in earlier chapters that multirate filters exhibit unique properties related to their being operated as LTV filters. One interesting property is the use of aliasing rather than a heterodyne to move a spectral region to or from baseband while down sampling or up sampling, respectively. We learned that, when down sampling, any multiple of the output sample rate could be aliased to baseband. Similarly, when up sampling, the baseband signal can be aliased to any multiple of the input sample rate. We quickly conclude that since any of the multiple bands are available, perhaps all the bands are available. In fact that is true. We now examine how the architecture of the polyphase filter structure can be applied to the task of simultaneously servicing multiple narrowband channels. The multiple channel version of the polyphase filter is often described as a channelizer or a transmultiplexer. We will first examine the use of the system as a multiple channel demodulator and then its dual form as a multiple channel modulator. Variants of the two techniques are then examined in the final section illustrating a number of specific applications of the process.

9.1 Demodulator Channel Bank

In Chapter 6 we derived the polyphase partition of a band-pass FIR filter. We showed how the M-to-1 down-sampled input series aliases spectra centered at any multiple of the output sample rate to baseband. We then described how the complex heterodyne embedded in the band-pass filter is mapped through the polyphase partition to a set of frequency-dependent phase rotators on the separate paths of the polyphase partition. This structure, repeated here, is shown in Figure 9.1.

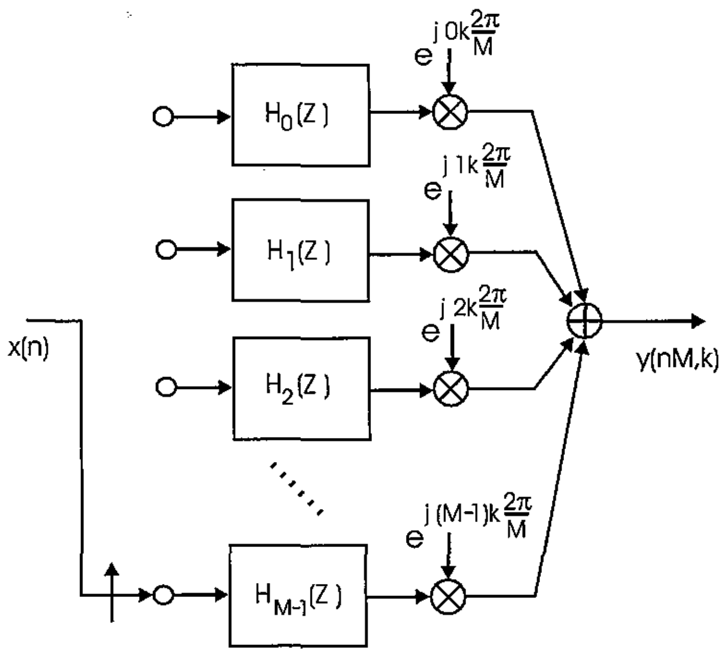


Figure 9.1 Polyphase Partition of M-to-1 Down Sampled Band-pass Filter

The partition and signal flow shown in Figure 9.1 collectively perform the three operations of spectral translation, band-pass filtering, and resampling. The phase-coherent summation performed at the final output accumulator has the form shown in (9.1). Here the variable $y_r(nM)$ represents the *nM*th time sample from the *r*th path of the M-path polyphase partition while the variable y(nM,k) represents the *nM*th time sample of the time series from the center frequency at the *k*th multiple of the output sample rate.

$$y(nM,k)=\sum_{r=0}^{M-1} y_r(nM)\, e^{j\frac{2\pi}{M}rk} \qquad (9.1)$$

We recognize that the summation in (9.1) is the DFT of the vector of M-time samples formed at time nM from the array of M-polyphase output paths. Since the summation can be performed for any index k, it can in fact be performed for all indices. When the process is performed for a range of indices it is described as a polyphase filter bank implementation of a channelized filter bank. In this mode, the phase rotators are applied by the DFT for each output index. The block diagram of this form of the process is shown in Figure 9.2. Here we see that the input commutator delivers an FDM signal to the polyphase stages and that the output commutator extracts a time division multiplexed (TDM) frame from the DFT output.

This change from one multiplex form to another is why the system is known as a transmultiplexer.

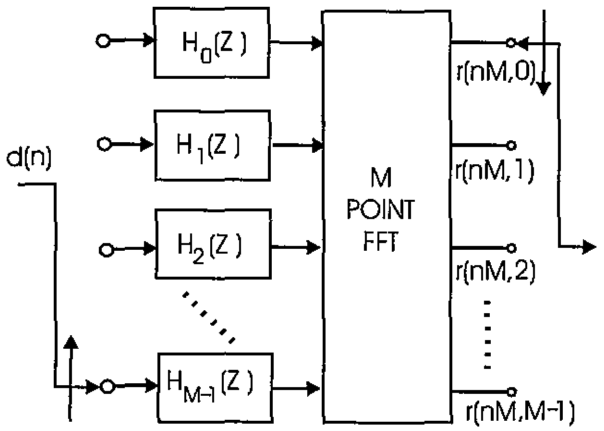


Figure 9.2 Polyphase Filter Bank as a Polyphase Filter Input and a DFT Output Process

We now address the interaction and coupling, or lack of coupling, between the parameters that define the polyphase bank. We observe that the DFT performs the task of separating the channels after the polyphase filter, so it is natural to conclude that the transform size is related to the number of channels spanning the input Nyquist span defined by the input sample rate. This is a correct assessment! We also note that the filter spectral characteristics are defined by the weights of the low pass prototype filter residing in the polyphase partition and that all filters in the filter bank exhibit these spectral characteristics.

In standard channelizer designs, the bandwidth of the prototype is specified in accord with the end use of the channelizers. For instance, when the channelizer is used for spectral decomposition as in a spectrum analyzer, or for source coding, the channels are designed to exhibit a specified attenuation, such as –3.0, –1.0, or –0.1-dB, at their crossover frequency with a specified attenuation at their adjacent center frequency. Overlap of adjacent channel responses permits a narrowband input signal to straddle one or more output channel filters. This is a common practice in the spectral analysis of signals with arbitrary bandwidths and center frequencies. On the other hand, when a channelizer is used to separate adjacent communication channels, channels with known center frequencies and known nonoverlapping bandwidths, the channelizers must maintain the separation of the channel outputs. Typical spectral responses for channel bandwidths corresponding to the two conditions just described are shown in Figure 9.3.

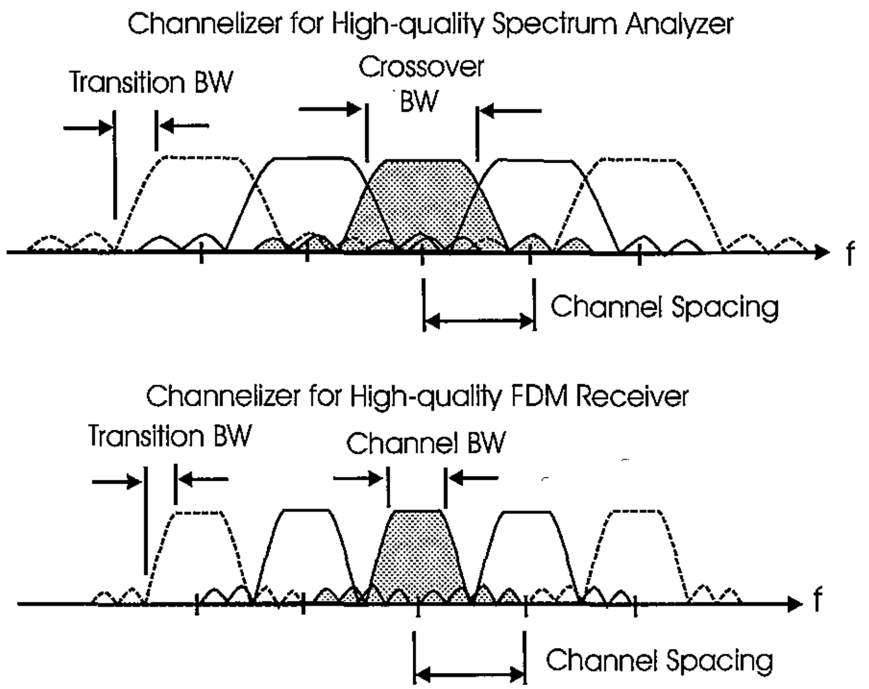

Channelizer for High-quality Spectrum Analyzer
Transition BW
Crossover BW
f
Channel Spacing
Channelizer for High-quality FDM Receiver
Transition BW
Channel BW
f
Channel Spacing

Figure 9.3 Spectral Response of Two Filter Banks with Same Channel Spacing: One for Spectral Analysis and One for FDM Channel Separation

The polyphase filter normally operates with an input M-to-1 resampling commutator. This M-to-1 down sampling aliases to baseband the spectral terms residing at multiples of the output sample rate. The DFT following the polyphase filter defines the channel spacing, which is one-*M*th of the input sample rate. Consequently, the standard polyphase channelizer has an output sample rate matching the channel spacing. For the case of the spectrum analyzer application operating at this sample rate permits aliasing of the spectral band edge into the down sampled pass band. When operated in this mode, the system is called a maximally decimated filter bank. By proper design of the prototype filter, this aliasing can be reversed if the signal is synthesized from the down-sampled and aliased analysis process. A system so designed is known as a perfect reconstruction filter bank.

9.2 Arbitrary Output Sample Rates

For the case of the communication filter bank, an output sample rate matching the channel spacing can avoid adjacent channel cross talk since the two-sided bandwidth of each channel is less than the channel spacing. An example of a signal that would require this mode of operation is the Quadrature Amplitude Modulation (QAM) channels of a digital cable system. In North America, the channels are separated by 6-MHz centers and operate with square-root cosine tapered Nyquist-shaped spectra with 18% or 12% excess bandwidth, at symbol rates of approximately 5.0 MHz. The minimum sample rate required of a cable channelizer to satisfy the Nyquist criterion would be 6.0 MHz. (The European cable plants have channel spacing of 8.0 MHz and symbol rates of 7.0 MHz.) The actual sample rate would likely be selected as a multiple of the symbol rate rather than a multiple of the channel spacing.

Systems that channelize and form samples of the Nyquist-shaped spectrum often present the sampled data to an interpolator to resample the time series from the collected bandwidth-related Nyquist rate to a rate offering two samples per symbol or twice symbol rate. For the TV example just cited, the 6-Ms/s, 5-M symbol signal would have to be resampled by 5/3 to obtain the desired 10 Ms/s. This is not a difficult task, and it is done quite regularly in single-channel receivers. This resampling may represent a significant computational burden if we are required to perform this interpolative resampling for every output channel.

The conventional way we use the M-path polyphase filter bank is to deliver M-input samples to the M-paths and then compute outputs from each channel at the rate fs/M. The thought may occur to us, "Is it possible to operate the polyphase filter bank in a manner that the output rate is higher than one *M*th of the input rate?" For instance, can we operate the bank so that we deliver M/2 inputs prior to computing an output sample rather than delivering M input samples before computing an output sample? Increasing the output sample rate of the polyphase channel bank by a factor of two makes subsequent interpolation tasks less expensive since the spectra of the output signals would already be oversampled by a factor of two with increased spectral separation. Operation in this mode would also permit channelization of overlapped channels without aliasing of the spectral transition bands. The alias-free partition is handy in applications requiring perfect reconstruction of the original time series from spectrally partitioned subchannels. For the record, a polyphase filter bank can be operated with an output sample rate of any rational ratio times the input sample rate. With minor modifications the filter can be operated with totally arbitrary ratios between input and output sample rates. This is true for the sample rate reduction imbedded in a polyphase receiver as well as for the sample rate increase embedded in a polyphase modulator.

We first examine the task of increasing the output sample rate from the polyphase filter bank from fs/M to 2 fs/M. We accomplish this by controlling the commutator delivering input data samples to the polyphase stages. We normally deliver M inputs to the M-path filter by delivering successive input samples starting at Port M-1 and progressing up the stack to Port 0 and by doing so delivering M inputs per output for an M-to-1 down-sampling operation. To obtain the desired (M/2)-to-1 down sampling, we deliver M/2 successive input samples starting at Port (M/2)-1 and progressing up the stack to Port 0. The M/2 addresses to which the new M/2 input samples are delivered are first vacated by their former contents, the M/2 previous input samples. All the samples in the two-dimensional filter undergo a serpentine shift of M/2 samples with the M/2 samples in the bottom half of the first column sliding into the M/2 top addresses of the second column while the M/2 samples in the top half of the second column slide into the M/2 addresses in the bottom half of the second column, and so on. This is equivalent to performing a linear shift through the prototype one-dimensional filter prior to the polyphase partition. In reality, we do not perform the serpentine shift but rather perform an addressing manipulation that swaps two memory banks. This is shown in Figure 9.4 where successive sequences of length 32 are delivered to a filter bank with 64 stages.

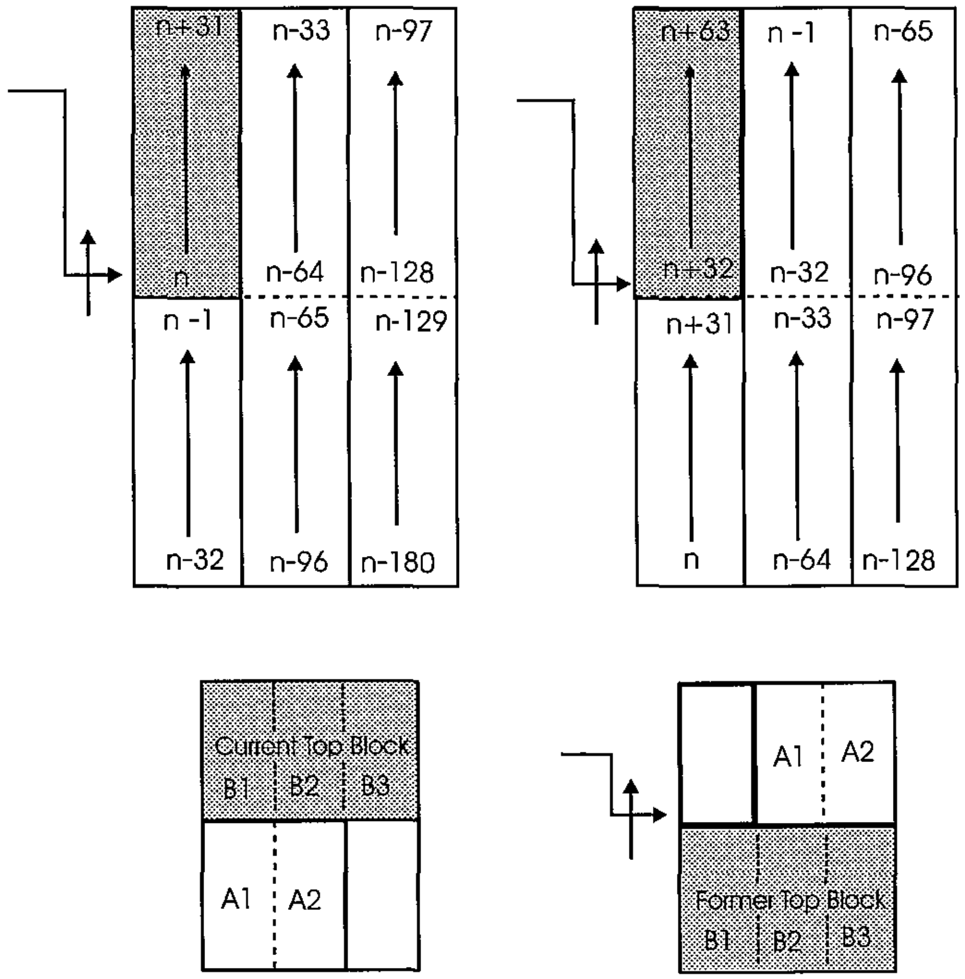


Figure 9.4 Data Memory Loading for Successive 32-point Sequences in a 64-stage Polyphase Filter

We continue this discussion with comments on the 64-stage example. After each 32-point data sequence is delivered to the partitioned 64-stage polyphase filter, the outputs of the 64 stages are computed and conditioned for delivery to the 64-point FFT. The data shifting into the polyphase filter stages causes a frequency-dependent phase shift of the form shown in (9.2). The time delay due to shifting is nT where n is the number of samples, and T is the interval between samples. The frequencies of interest are integer multiple k of 1/Mth of the sample rate 2π/T. Substituting these terms in (9.2) and canceling terms, we obtain the frequency-dependent phase shift shown in (9.3). From this relationship we see that for time shifts n equal to multiples of M, as demonstrated in (9.4), the phase shift is a multiple of 2π and contributes no offset to the spectra observed at the output of the FFT. The M-sample time shift is the time shift applied to the data in the normal use of the polyphase filter. Now suppose that the time shift is M/2 time samples. When substituted in (9.3) we find, as shown in (9.5), a frequency dependent phase shift of kπ from which we conclude that odd-indexed frequency terms experience a phase shift of π radians for each successive N/2 shift of input data.

$$\theta(\omega) = \Delta t \cdot \omega \tag{9.2}$$

$$\theta(\omega_k) = nT \cdot k\frac{1}{M}\,\frac{2\pi}{T} = \frac{nk}{M}\,2\pi \tag{9.3}$$

$$\theta(\omega_k)\Big|_{n=M} = \frac{nk}{M}2\pi\Big|_{n=M} = k \cdot 2\pi \tag{9.4}$$

$$\theta(\omega_k)\Big|_{n=M/2} = \frac{nk}{M}2\pi\Big|_{n=M/2} = k \cdot \pi \tag{9.5}$$

This π radian phase shift is due to the fact that the odd-indexed frequencies alias to the half sample rate when the input signal is down sampled by M/2. We can compensate for the alternating signs in successive output samples by applying the appropriate phase correction to the spectral data as we extract successive time samples from the odd-indexed frequency bins of the FFT. The phase correction here is trivial, but for other down-sampling ratios, the residual phase correction would require a complex multiply at each transform output port. Alternatively, since time delay imposes a frequency-dependent phase shift, we can use time shifts to cancel the frequency-dependent phase shifts. We accomplish this by applying a circular time shift of N/2 samples to the vector of samples prior to their presentation to the FFT. As in the case of the serpentine shift of the input data, the circular shift of the polyphase filter output data is implemented as an address-manipulated data swap. This data swap occurs on alternate input cycles and a simple two-state machine determines for which input cycle the output data swap is applied. This option is shown in Figure 9.5.

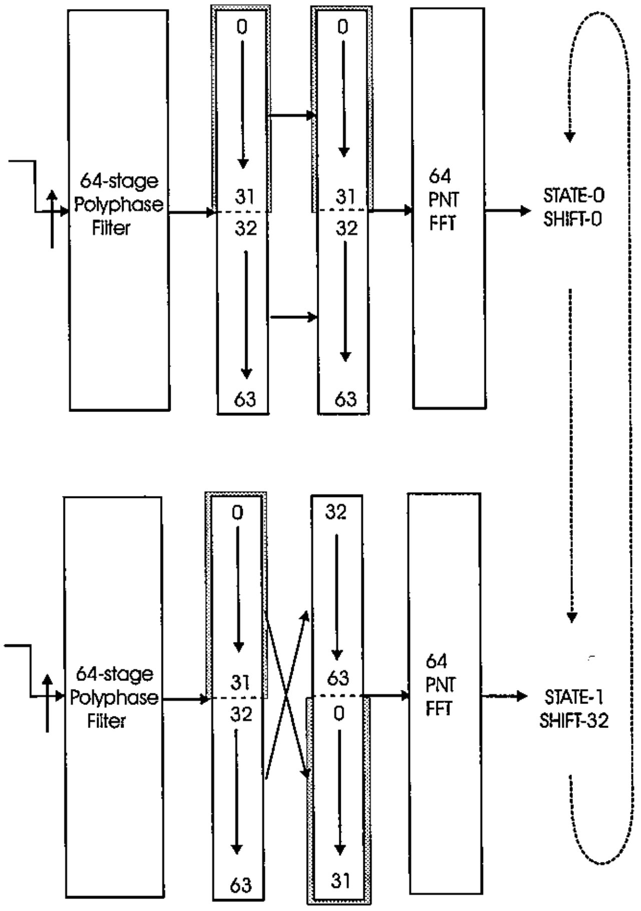


Figure 9.5 Cyclic Shift of Input Data to FFT to Absorb Phase Shift Due to Linear Time Shift of Data Through Polyphase Filter

We are now prepared to examine the process that permits resampling of the polyphase filter bank by any rational ratio. We first demonstrated the modification to the standard polyphase structure to support N/2 down sampling. The modifications involved a serpentine shift of input memory and a circular shift of output memory that are both implemented by memory bank data swaps.

The technique is based on the observation that the commutator is the component in the polyphase filter bank that effects and controls the resampling, not the spacing between the adjacent channels. This is true even though it was the resampling process that first guided us to select the channel spacing so we could access the aliasing to baseband. We now enlarge the set of options to permit aliasing to other frequencies. As we just demonstrated, there are two modifications to the polyphase-resampling filter required to obtain arbitrary resampling. These modifications would normally lead to an exercise in time-varying residue index-mapping of the two-dimensional input data array. If we limit the presentation to the index-mapping process we would develop little insight into the process and further would be bored to tears. Instead we derive and illustrate the modifications by examining a specific example and observing the process develop.

We have just examined the case in which the embedded resampling ratio of a polyphase filter was trivially modified from M-to-1 to M/2-to-1. We now examine the case in which the embedded resampling ratio is modified to (3/4 M)-to-1. The extension to any rational ratio is a trivial extension of this process. To aid in understanding the process we describe a specific resampling channelizer and modify the process to guide us to the desired solution.

The problem we examine is this: We have a signal containing 50 FDM channels separated by 192 kHz centers containing symbols modulated at 128 kHz by square-root Nyquist filters with 50% excess bandwidth. Our task is to baseband channelize all 50 channels and output data samples from each channel at 256 ks/s, which is two samples per symbol.

The specifications of the process are listed next and the spectrum of the FDM input signal and of one of the 50 output signals is shown in Figure 9.6.

Number of Channels:	50
Channels Spacing:	192 kHz
Channel Symbol Rate:	128 kHz
Shaping Filter:	SQRT Cosine Taper
Roll Off Factor α:	50%
Output Sample Rate:	256 kHz

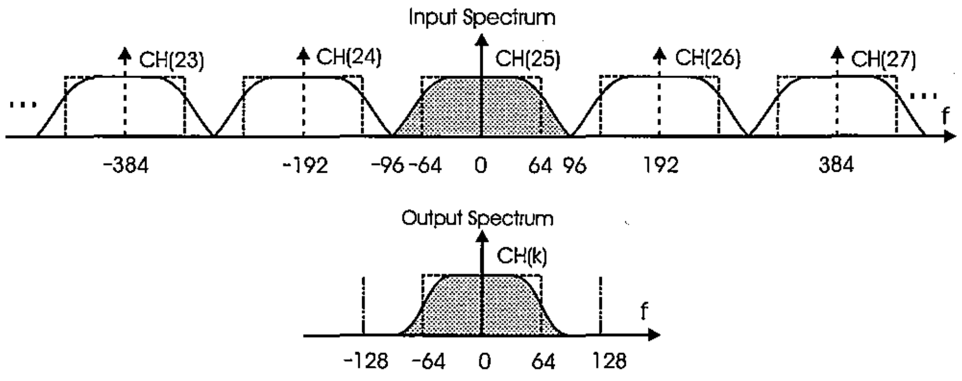


Figure 9.6. Input and Output Spectra of 50-channel Channelizer and Resampler

We start by selecting a transform size N greater than the number of FDM channels to be processed. As indicated in (9.6), the product of the transform size and the channel spacing defines the input sample rate of the data collection process. As shown in (9.7), a re-statement of the Nyquist sampling criterion, the excess bandwidth spanned by the extra channels in the transform is allocated to the transition bandwidth of the analog anti-alias filter.

$$f_s = N \cdot \Delta f \tag{9.6}$$

$$\begin{aligned} f_s &= \text{2-Sided BW} + \text{Transition BW} \\ \text{2-Sided BW} &= \text{\# Data Channels} \cdot \text{Channel Spacing} \\ \text{Transition BW} &= \text{\# Nondata Channels} \cdot \text{Channel Spacing} \\ \text{Transform Size} &= \text{\# Data Channels} + \text{\# Nondata Channels} \end{aligned} \tag{9.7}$$

We note the engineering compromise to be made here: A larger transform size would reduce the cost of the analog anti-alias filter by permitting wider transition bandwidth, while a larger transform would increase the input sample rate, the number of polyphase stages in the filter bank, and the arithmetic processing burden required to implement the bank and the transform.

The size of the selected transform should be highly composite so that it can be implemented as an efficient FFT. Table 9.1 lists transform sizes appropriate for the polyphase channelizer along with required data sample rate and order of anti-aliasing filter.

Table 9-1 List of Highly Composite Transform Sizes Considered for 50-stage Polyphase Channelizer

Transform Size N	Factors	Sample Rate (MHz)	Pass-band Edge in FFT Bins	Stop-band Edge in FFT Bins	Filter Order 0.1-dB Pass 60.-dB Stop
54	2,3,3,3	10.368	25.5	28.5	9
60	3,4,5	11.520	25.5	34.5	7
64	2,2,2,2,2,2	12.288	25.5	38.5	6
72	2,2,2,3,3	13.824	25.5	46.5	6
80	2,2,2,2,5	15.360	25.5	54.5	5

We select a 64-point FFT to span the 50 channels with the excess bandwidth allocated to the analog anti-alias filter. Thus the sample rate for the collected spectra is 64 times the 192 kHz channel spacing or 12.288 MHz. These are complex samples formed from either a baseband block conversion or a digital down conversion and resampling from a digital IF, often centered at the quarter sample rate. The desired output sample rate is 2 times 128 or 256 kHz. The ratio between the input and output sample rates is the resampling ratio, which is 12288/256 or 48-to-1. Thus our task is to use the 64-point DFT to separate and deliver 50 of the possible 64 channels spanned by the sample rate, but to deliver one output sample for every 48 input samples.

Figure 9.7 is a block diagram of the original maximally decimated version of the 64-stage polyphase channelizer and the modified form of the same channelizer. The difference in the two systems resides in the block inserted between the 64-stage polyphase filter and the 64-point FFT. As indicated earlier, the inserted block performs no computation but rather only performs a set of scheduled circular buffer shifts. Next we will develop and describe the operation of the circular buffer stage and state machine scheduler.

Our first task is to modify the input commutator to support the 48-to-1 down sample rather than the standard 64-to-1 down sample. This is an almost trivial task. We arrange for the modified resampling by keeping the 64-path filter but stripping 16 ports from the commutator. The commutator for the standard 64-point polyphase filter starts at port 63 and delivers 64 successive inputs to ports 64, 63, 62, and so on through 0. The modified commutator starts at port 47 and delivers 48 successive inputs to ports 47, 46, 45, and so on through 0. Input memory for the 64-path filter must be modified to support this shortened commutator input schedule. The mapping structure of the reindexing scheme is best seen in the original, one-dimensional prototype filter shown in Figure 9.8 and then transferred to the two-dimensional polyphase partition.

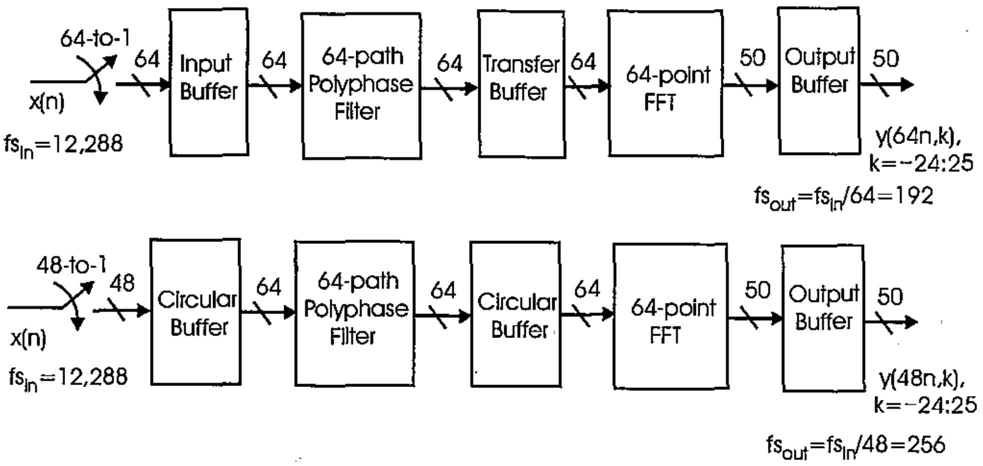


Figure 9.7 Maximally Decimated Filter Bank Structure and Modified Two-Sample-per-Symbol Filter Bank Structure

Figure 9.8 presents the memory content for a sequence of successive 48-point input data blocks presented to the 64-point partitioned prototype filter. In this figure we have indicated the interval of 64-tap boundaries that become the columns of the two-dimensional array as well as the boundaries of successive 48-point input blocks that are presented to the input array. Successive input blocks start loading at Address 47 and work up to Address 0. The beginning and end of this interval are denoted by the tail and arrow, respectively, of the left-most input interval in the filter array. As each new 48-point input array is delivered, the earlier arrays must shift to the right. These shifting array blocks cross the 64-point column boundaries and hence move to adjacent columns in the equivalent two-dimensional partition. This crossing can be visualized as a serpentine shift of data in the two-dimensional array, or equivalently as a circular row buffer down shift of 48 rows in the polyphase memory with a simultaneous column buffer right shift of the input data column. The operation of this circular buffer is illustrated in Figure 9.9, which indicates the indices of input data for two input cycles. Here we see that between two successive input cycles the rows in the top one-fourth of memory translate to the bottom fourth while the bottom three-fourths of rows translate up one-fourth of memory. We also see that the columns in the bottom three-fourths shift to the right one column during the circular row translations. The next input array is loaded in the left-most column of this group of addresses.

Returning to Figure 9.8, the one-dimensional prototype, we note that every new data block shifts the input data origin to the right by 48 samples. The vector $\hat{y}(r,48n)$ formed as the polyphase filter output from all 64-path filters is processed by the FFT to form the vector $\hat{Y}(k,48n)$ of channelized (index k) output time series (index 48n). On each successive call to the FFT, the origin of the sinusoids in the FFT is reset to the beginning of the input array. Since the origin of the input array shifts to the right on successive inputs while the origin of the FFT simultaneously resets to the beginning of the input array, a precessing offset exists between the origins of the polyphase filter and of the FFT. We align the origins, removing the offsets, by performing a circular shift of the vector $\hat{y}(r,48n)$ prior to passing it to the FFT. Since the offset is periodic and is a known function of the input array index, the circular offset of the vector can be scheduled and controlled by a simple state machine. Figure 9.8 shows the location of the two origins for four successive 48-point input arrays and the amount of circular offset required to align the two prior to the FFT. Note that the offset schedule repeats in four cycles, four being the number of input intervals of length 48 that is a multiple of 64. The cyclic shift for schedule for the array $\hat{y}(r,48n)$ prior to the FFT is shown in Figure 9.10.

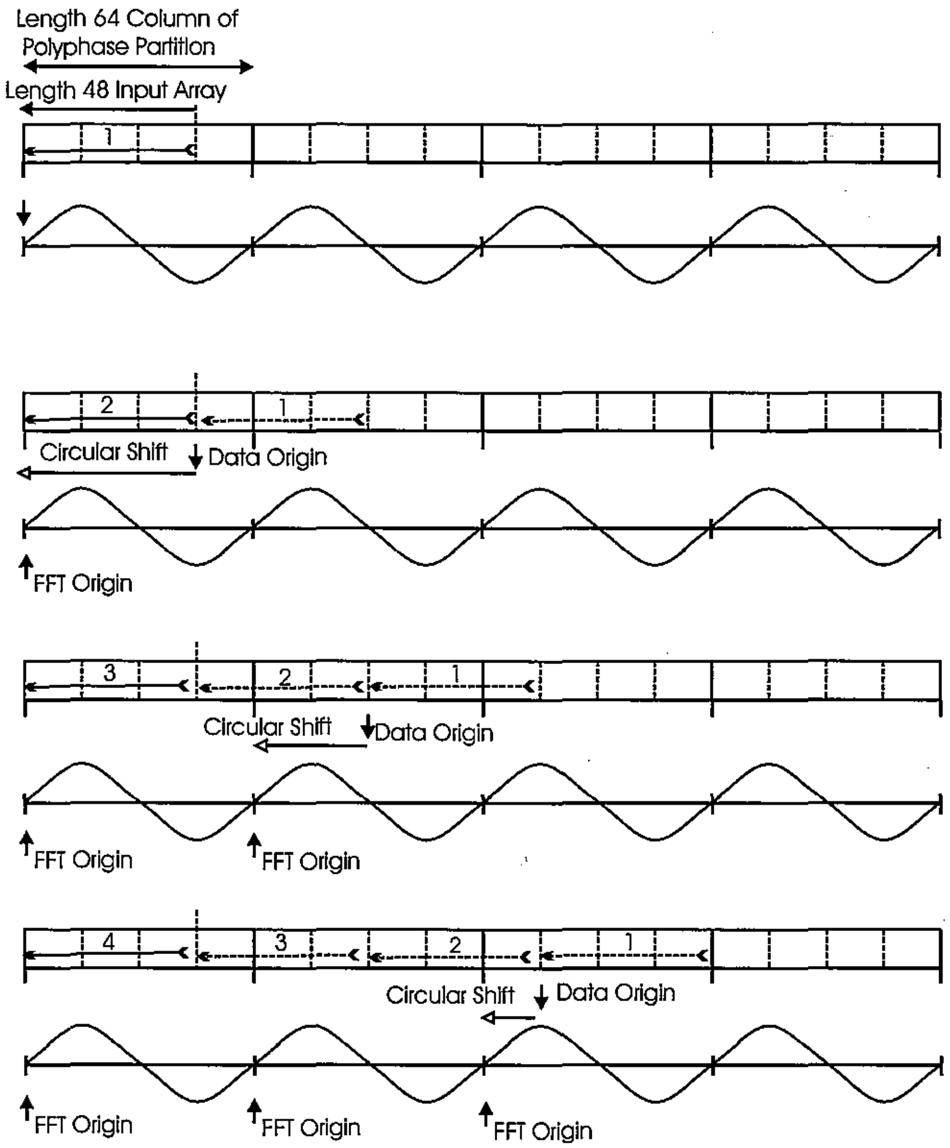


Figure 9.8 Memory Contents for Successive 48-point Input Data Blocks Into a 64-point Prototype Pre-polyphase Partitioned Filter and FFT

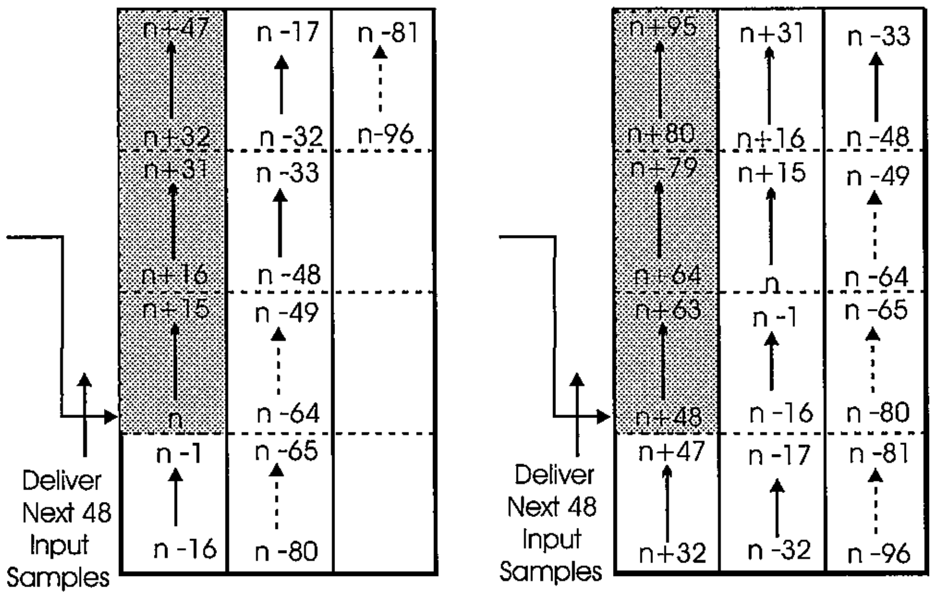


Figure 9.9 Memory Contents for Successive 48-point Input Data Blocks Into a 64-point Polyphase Filter

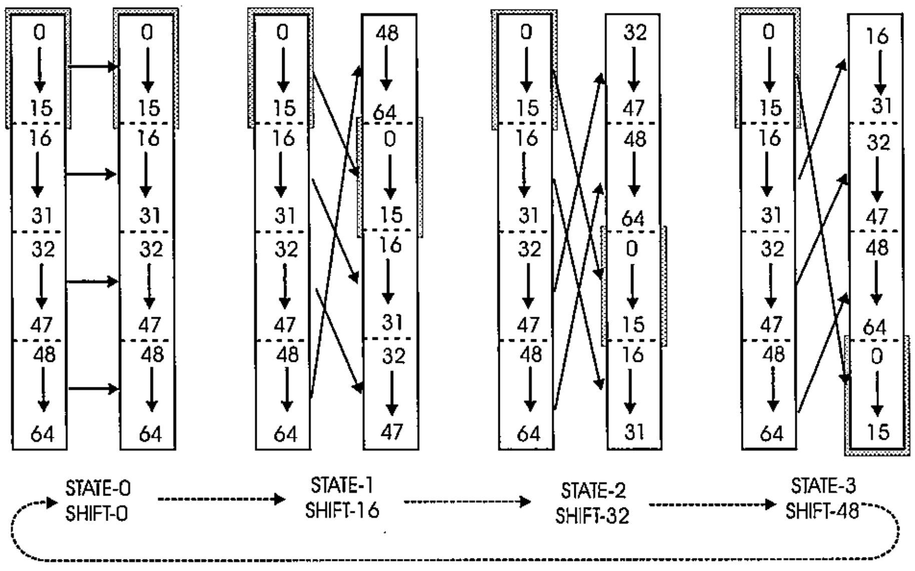


Figure 9.10 Cyclic Shift Schedule for Input Array to FFT

9.2.1 Comparison of Design Options

The design example presented in the previous section embedded the matched filtering, channelizer, and resampler in a modified polyphase filter. In this section we examine other design options that perform the same tasks with different combinations of resources. This will allow us to compare the relative efficiency of the different approaches. With a fresh slate we address the task of allocating various combinations of a resampling polyphase channelizer, output interpolators, and matched filters. The matched filter can be folded into the channelizer or the interpolator. Similarly, the resampling can be performed in two steps by interpolating the output of a maximally decimated polyphase filter, by interpolating the output of a nonmaximally decimated polyphase filter, or in a single step by operating the polyphase filter at the desired output sample rate. In particular, we offer and examine five of the many possible design combinations that satisfy the requirements of this processing task. The options we examine are listed and described next and are illustrated in Figure 9.11.

1. ***Design 1.*** Channelize the FDM signal with 64-to-1 down sampling in the 64-path polyphase channelizing filter to obtain a per-channel output sample rate of 192 kHz, and then up sample each channel by 4/3 to obtain the desired 256 kHz output sample rate, and finally pass each correctly sampled channel series through a matched filter. To reduce processing burden and cost, the matched filter is embedded in the 4/3-interpolator filter.

2. ***Design 2.*** Channelize the FDM signal with 64-to-1 down sampling in the 64-path polyphase matched filter to obtain a per-channel output sample rate of 192 kHz, and then up sample each matched filter series by 4/3 to obtain the desired 256 kHz output sample rate.

3. ***Design 3.*** Channelize the FDM signal with a 32-to-1 down sampling in the 64-path polyphase channelizing filter to obtain a per-channel output sample rate of 384 kHz, then down sample by 3/2 to obtain the desired 256 kHz output sample rate, and finally pass each correctly sampled channel series through a matched filter. Here too, to reduce processing burden and cost, the matched filter is embedded in the 3/2-interpolator filter.

4. ***Design 4.*** Channelize the FDM signal with a 48-to-1 down sampling in the 64-path polyphase channelizing filter to obtain a per-channel output sample rate of 256 kHz, and finally pass each correctly sampled channel series through a matched filter.

5. ***Design 5.*** Channelize the FDM signal with a 48-to-1 down sampling in the 64-path polyphase matched filter to directly obtain a per-channel matched filter output sample rate of 256 kHz.

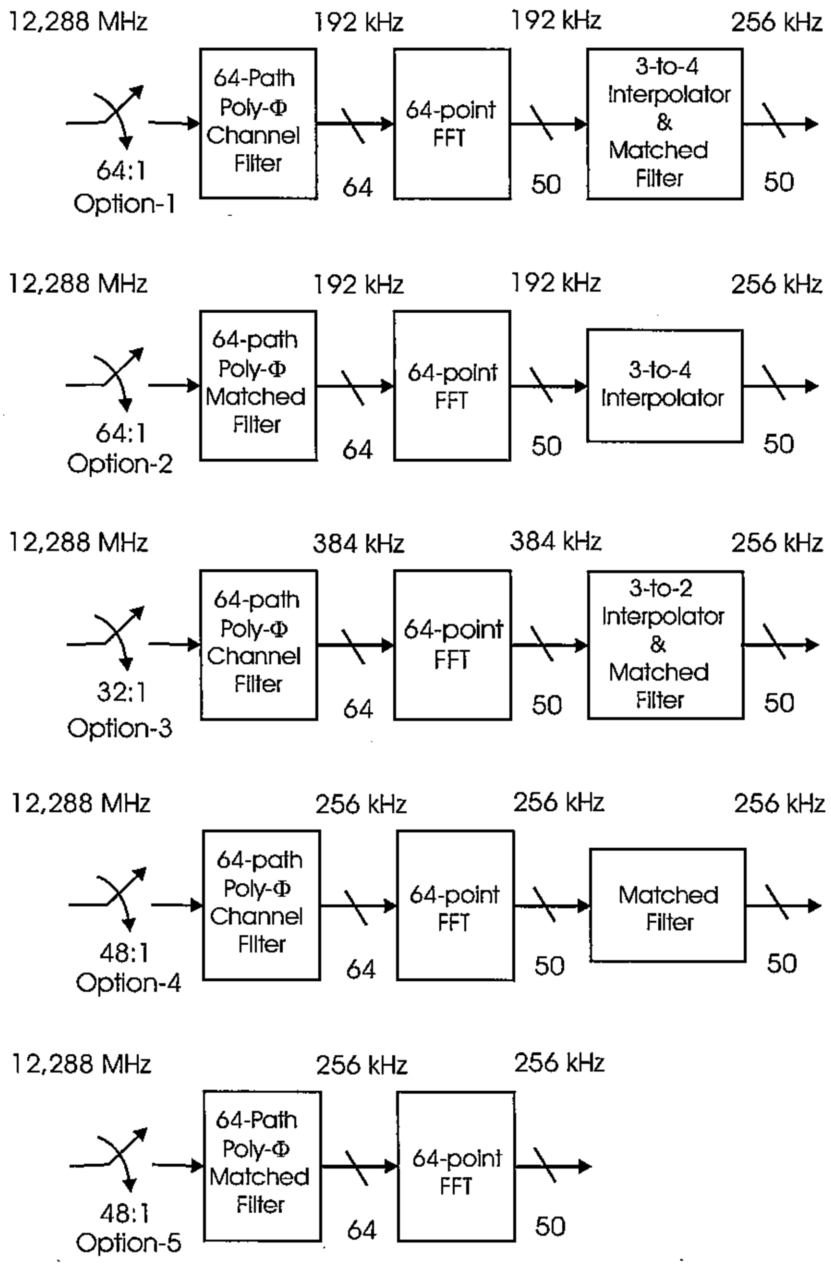


Figure 9.11 Signal Flow Block Diagrams Indicating Data Rates and Number of Signal Paths in Different Implementations of 50-channel Resampling Filter Bank

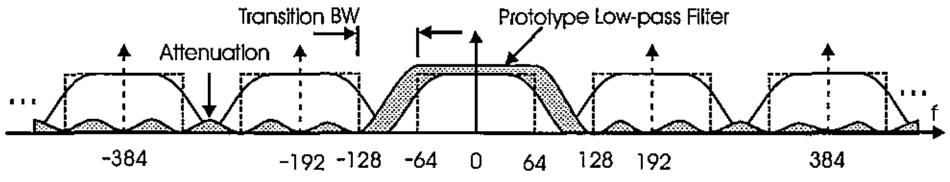


Figure 9.12 Frequency Response of Prototype Low pass Filter Used in Channelizer

Option-1 Design
The prototype filter used in the channelizer options has the spectral response shown in Figure 9.12 with these specific parameters:

Prototype Channel Filter Specifications

Sample Rate:	12.288 MHz
Pass-band Span:	0 to 64 kHz
Stop-band Span:	128 kHz to 6.444 MHz
Pass-band Ripple:	0.1-dB
Stop-band Attenuation:	60-dB
Attenuation Decay Rate:	6.0-dB/Octave

The filter length N_1 is estimated in (9.8) to be 524 taps. Since the filter is to be partitioned into 64 polyphase stages it is convenient, but not necessary, to have the filter length be an integer multiple of 64. The two possible lengths that bracket the estimate from (9.8) are 512 and 576. Filters of both lengths were designed with the Remez algorithm, and the length 512 filter did not meet the filter specifications while the length 576 filter met the specifications with slight margin.

$$N_1 \cong \frac{f_S}{\Delta f}\frac{Atten(dB)}{22} = \frac{12{,}288}{64}\,\frac{60}{22} \approx 524 \tag{9.8}$$

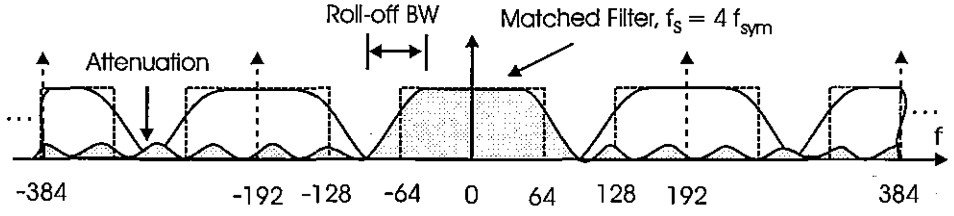


Figure 9.13 Frequency Response of Prototype Matched Filter and 3-to-4 Interpolator Applied to Each Channel Following Channelizer

The prototype filter used in the matched filter performing double duty as the 3-to-4 interpolator following the channelizer has the spectral response shown in Figure 9.13 with these specific parameters:

Prototype Filter Specifications

Root-raised Cosine Nyquist Filter

Input Sample Rate:	192 kHz
Output Sample Rate:	768 kHz
Symbol Bandwidth:	128 kHz
Roll-off Bandwidth:	64 kHz
Pass-band Ripple:	0.1-dB
Stop-band Attenuation:	60-dB
Attenuation Decay Rate:	6.0-dB/Octave

The filter length N_2 is estimated in (9.9) to be 59 taps. Since the filter is to be partitioned into 4 polyphase stages it is convenient, but not necessary, to have the filter length be an integer multiple of 4. The next possible length matching the estimate from (9.9) is 60. A SQRT-Nyquist filter of length 60 designed by nyq_2 met the specifications with slight margin.

$$N_2 \cong 1.8 \cdot \frac{f_S}{\Delta f} \frac{Atten(dB)}{22} = 1.8 \cdot \frac{768}{64} \frac{60}{22} \approx 59 \tag{9.9}$$

The 60 coefficients of the matched filter are partitioned into a 4-path polyphase interpolating filter so that each stage requires 15 taps. The filter dimensions required to implement design option 1 are shown in Figure 9.14.

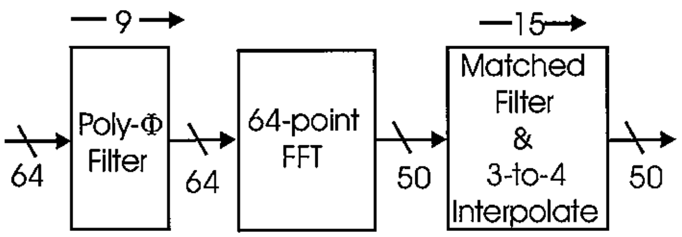


Figure 9.14 Filter Dimensions of Design Option 1

The process shown in Figure 9.14 proceeds as follows: 64 complex input samples are delivered to and are processed by the 64-stage polyphase input filter. The 64 output samples formed by the polyphase partition are processed by the 64-point FFT and 50 outputs, corresponding to a time sample from each of the 50 FDM channels, and are delivered to the matched filter and interpolator. The interpolator is a bank of 50 resamplers that process the input samples from the 50 selected channels to up sample the time series for each channel. It is interesting to estimate the workload for the process and to see how it is distributed over the processing blocks. The workload to compute the 50 output samples is shown in (9.10), where we count a real multiply and add as an OP for operation.

$$\begin{array}{lll} \text{Polyphase Filter: } 2\cdot(64\cdot 9) & = 1152 \\ \text{FFT:} \quad 4\cdot\frac{64}{2}\log_2(64) & = 758 \\ \text{Matched Filter: } 2\cdot(50\cdot 15) & = 1500 \\ \text{Total Number of OPS:} & = 3410 \\ \text{OPS per Channel:} & = 68.2 \end{array} \tag{9.10}$$

For comparison we can estimate the workload for a very efficient conventional single-channel down converter as shown in Figure 9.15. Here the input signal, sampled at 12.288 MHz, is down converted to baseband by a complex heterodyne, filtered by a pair of low pass filters that output data at the desired 256 kHz, and then match filtered by a 20-tap SQRT-Nyquist filter operating at 2 samples per symbol.

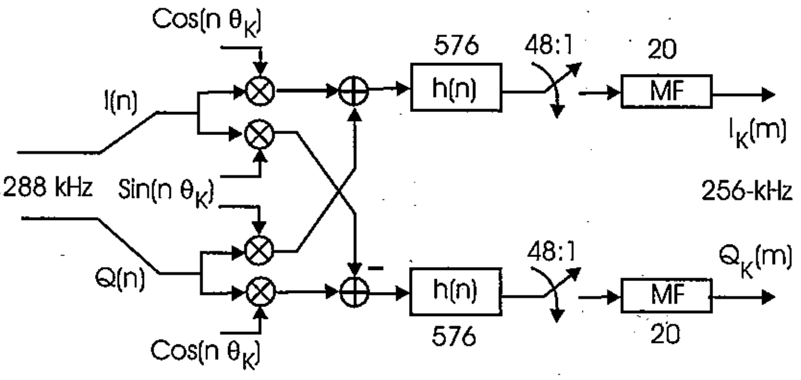


Figure 9.15 Block Diagram of Efficient Single Channel Down Converter, Resampler, and Matched Filter Processor

Here 48 complex input samples are processed to compute one complex output sample. The required workload is shown in (9.11).

$$\begin{array}{lll} \text{Complex Heterodyne: } 4\cdot(48) & = 192 \\ \text{Filter:} \quad 2\cdot(576) & = 1152 \\ \text{Matched Filter:} \quad 2\cdot(20) & = 40 \\ \text{Total Number of OPS:} & = 1384 \end{array} \tag{9.11}$$

We now compare the relative workload of the two processing techniques: The polyphase filter bank and the multiple single-channel down converter. We first compare the workload per channel. From (9.10) and (9.11) we obtain the results shown in (9.12). Here we see that the per-channel workload for the polyphase channelizer is less than 5% of the workload of the single-channel down converter. In a sense, the polyphase channelizer offers 20 channels for the price of a single-channel down converter.

Polyphase Bank, OPS per Channel:	68.2	
Single Channel, OPS per Channel:	1384	(9.12)
Workload Ratio:	0.0493	

From another perspective, we can compare the total workload of the polyphase filter bank to that of the single-channel down converter. Again, from (9.10) and (9.11) we obtain the results shown in (9.13). Here we see that the workload for the polyphase channelizer is approximately 2.5 times the workload of a single-channel down converter. We can conclude from this comparison that if we require one or two channels, we build the required one or two single channel converters. On the other hand, if we require three or more channels, there is a cost advantage to building the polyphase channelizer and simply discarding the outputs of the unused channels.

Polyphase Bank, Total OPS:	3410	
Single Channel, Total OPS:	1384	(9.13)
Workload Ratio:	2.4639	

Option-2 Design

Examining (9.10), we note that the matched filter and interpolation performed in the final output stage represented nearly half the workload of the option 1 design. In the option-2 design we use the channel filter as the matched filter and have the final output stage perform only the 3-to-4 interpolation. The prototype filter used in the matched filter channelizer option must meet these specification parameters:

Prototype Filter Specifications

Root-raised Cosine Nyquist Filter

Input Sample Rate:	12.288 MHz
Symbol Bandwidth:	128 kHz
Roll-off Bandwidth:	64 kHz
Pass-band Ripple:	0.1-dB
Stop-band Attenuation:	60-dB
Attenuation Decay Rate:	6.0-dB/Octave

The filter length required to meet these specifications is found in (9.14) to be approximately 943 taps. Since we would like the number of taps in the channelizing to be a multiple of 64, we chose the number of taps to be 960. The 960-tap SQRT-Nyquist filter designed by nyq_2 was able to meet the channelizer specifications.

$$N_2 \cong 1.8 \cdot \frac{f_S}{\Delta f} \frac{Atten(dB)}{22} = 1.8 \cdot \frac{12{,}288}{64} \frac{60}{22} \approx 943 \tag{9.14}$$

The prototype filter used in the 3-to-4 interpolator following the matched filter channelizer must meet these specification parameters:

Prototype Filter Specifications

3-to-4 Interpolator

Input Sample Rate	192 kHz
Output Sample Rate	768 kHz
Symbol Bandwidth	128 kHz
Roll-off Bandwidth	64 kHz
Pass-band Ripple	0.1-dB
Stop-band Attenuation	60-dB
Attenuation Decay Rate	6.0-dB/Octave

The filter length required to meet these specifications is found in (9.15) to be approximately 34 taps. Since we would like the number of taps in the channelizing to be a multiple of 4, we chose the number of taps to be 36. The Remez design of 36-tap interpolating filter met the channelizer specifications.

$$N_1 \cong \frac{f_S}{\Delta f} \frac{Atten(dB)}{22} = \frac{4 \cdot 192}{64} \frac{60}{22} \approx 34 \tag{9.15}$$

The 36 coefficients of the matched filter are partitioned into a 4-path polyphase interpolating filter so that each stage requires 9 taps. The filter dimensions required to implement design option 2 are shown in Figure 9.16.

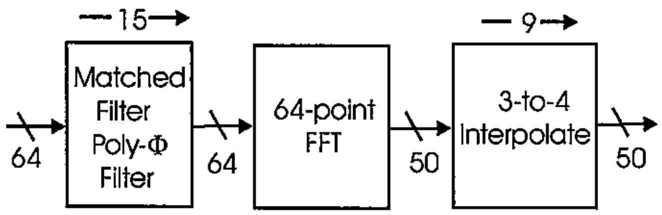


Figure 9.16 Filter Dimensions of Design Option 2

The workload to compute the 50 output samples is shown in (9.16) where, as earlier, we count a real multiply and add as an OP for operation. Comparing (9.16) with (9.10) we see that as a result of embedding the matched filter in the channelizer there is a 5% increase in workload from 68.2 to 71.6 ops per channel. This increase is due to the longer matched filter operating in all 64 paths of the input filter as opposed to operating in the 50 paths of the output filter.

$$\begin{aligned}
&\text{Polyphase Filter: } 2\cdot(64\cdot 15) &&= 1{,}920\\
&\text{FFT: } \quad 4\cdot\frac{64}{2}\log_2(64) &&= 758\\
&\text{Matched Filter: } 2\cdot(50\cdot 9) &&= 900\\
&\text{Total Number of OPS:} &&= 3{,}578\\
&\text{OPS per Channel:} &&= 71.6
\end{aligned} \tag{9.16}$$

Option-3 Design

In option 3 we attempt to reduce the system workload by reducing the length of the prototype filter in the polyphase stage with the goal of reducing the processing performed at the high input rate in exchange for additional processing at the lower output rate. Increasing the transition bandwidth of the filter reduces the length of the filter. Since the output sample rate is defined by the sum of the two-sided bandwidth and the transition bandwidth, the shortened filter must operate at a high output sample rate. We elect to operate the polyphase filter at 384 kHz, which is twice the rate of two previous designs. We accomplish this feat by operating the polyphase filter bank once per 32 input samples rather than the standard mode of once per 64 input samples.

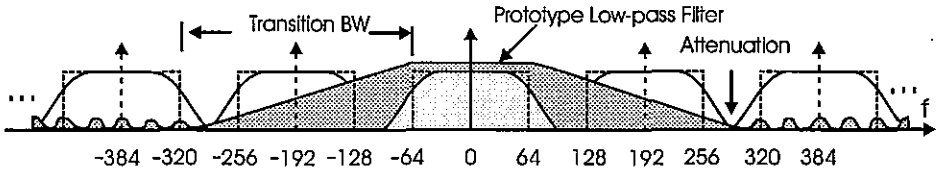


Figure 9.17 Frequency Response of Wide Transition Bandwidth Prototype Low pass Filter Used in Channelizer

The prototype filter used in the wide transition band channelizer option has the spectral response shown in Figure 9.17 with specific parameters listed next. Compare the transition bandwidth of this filter with the one presented for the option 1 design in Figure 9.12.

Prototype Channel Filter Specifications

Sample Rate:	12.288 MHz
Pass-band Span:	0 to 64 kHz
Stop-band Span:	320 kHz to 6.444 MHz
Pass-band Ripple:	0.1-dB
Stop-band Attenuation:	60-dB
Attenuation Decay Rate:	6.0-dB/Octave

The filter length N_1 is estimated in (9.17) to be 131 taps. This reduction in filter length by a factor of 4 relative to the option 1 design is expected since the transition bandwidth for this option is wider by a factor of 4. Since the filter is to be partitioned into 64 polyphase stages it is convenient, but not necessary, to have the filter length be an integer multiple of 64. The two possible lengths that bracket the estimate from (9.17) are 128 and 192. Filters of both lengths were designed with the Remez algorithm, and the length 128 filter did not quite meet the filter specifications while the length 192 filter met the specifications with significant margin.

$$N_1 \cong \frac{f_S}{\Delta f}\frac{Atten(dB)}{22} = \frac{12{,}288}{256}\frac{60}{22} \approx 131 \tag{9.17}$$

The length of the 3-to-2 interpolator required to bring the channelizer output sample rate of 384 kHz to the desired output rate of 256 kHz is the same as that required in the previous option 2 design. From (9.15), the interpolator length is seen to be 34 taps. The filter required for this application is partitioned into a 2-path filter with 17 taps each. The scheduling mode to obtain 2 outputs for every 3 inputs with this filter is shown in (9.18). Zero packing the input data 1-to-2 and noting the location of the non-zero data samples after each input shift of 3 input samples easily derives this schedule.

Input Data Index	Filter Phase	Output Data Index
n, n+1	h(2n)	m
n+2	h(2n+1)	m+1

(9.18)

The filter dimensions required to implement design option 2 are shown in Figure 9.18.

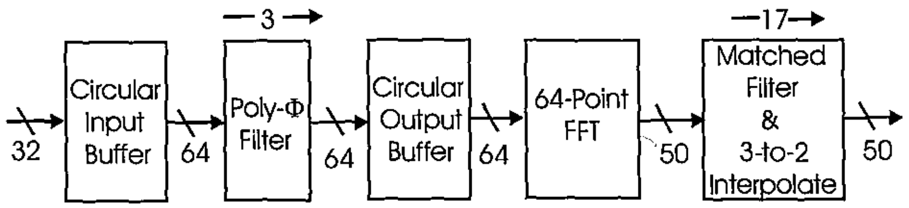


Figure 9.18 Filter Dimensions of Design Option 3

The workload to compute the 50 output samples is shown in (9.19) where, as earlier, e count a real multiply and add as an OP for operation. Comparing (9.19) with (9.10) we e that reducing the filter length and then operating the channelizer at a higher data rate sults in essentially the same workload 68.3 as opposed to 68.2 ops per channel. The ab-nce of expected improvement due to the shorter filter is attributed to the increase in the ımber of transforms required to compute an output block.

$$\begin{aligned}
&\text{Workload for three input blocks of length 32}\\
&\text{required to form two output blocks of length 50:}\\
&\text{Polyphase Filter: } 3\{2\cdot(64\cdot 3)\} &&= 1{,}152\\
&\text{FFT:} \qquad 3\{4\cdot\frac{64}{2}\log_2(64)\} &&= 2{,}274\\
&\text{Matched filter: } 2\{2\cdot(50\cdot 17)\} &&= 3{,}400\\
&\text{Total Number of OPS:} &&= 6{,}826\\
&\text{OPS per Output frame: } (\div 2) &&= 3{,}413\\
&\text{OPS per Channel:} &&= 68.3
\end{aligned} \qquad (9.19)$$

)ption-4 Design

n the option 4 design we perform the channelizing and 48-to-1 resampling with the cycli-:ally shifted buffers around the 64-path polyphase filter, with the FFT to separate the chan-ıels, and with an output filter per path to only perform the matched filtering task at 2 sam-ɔles per symbol. This differs from earlier versions in which the output stage performed in-:erpolation and filtering. The polyphase filter length is the same as derived in (9.8) for op-tion 1. The output filter length differs from previous options due to its distinct filtering task. The specifications for the output filter are listed next.

Prototype Filter Specifications

Root raised Cosine Nyquist Filter

Input Sample Rate:	256 kHz
Symbol Bandwidth:	128 kHz
Roll-off Bandwidth:	64 kHz
Pass-band Ripple:	0.1-dB
Stop-band Attenuation:	60-dB
Attenuation Decay Rate:	6.0-dB/Octave

The filter length N_2 is estimated in (9.20) to be 20 taps. Since the filter is only filtering we have no need to be a multiple of the number of polyphase stages, and we select the matched filter length to be 20 taps. A SQRT-Nyquist filter of length 20 designed by nyq_2 met the specifications with slight margin.

$$N_2 \cong 1.8 \cdot \frac{f_S}{\Delta f} \frac{Atten(dB)}{22} = 1.8 \cdot \frac{256}{64} \frac{60}{22} \approx 19.6 \qquad (9.20)$$

The filter dimensions required to implement design option 4 are shown in Figure 9.19.

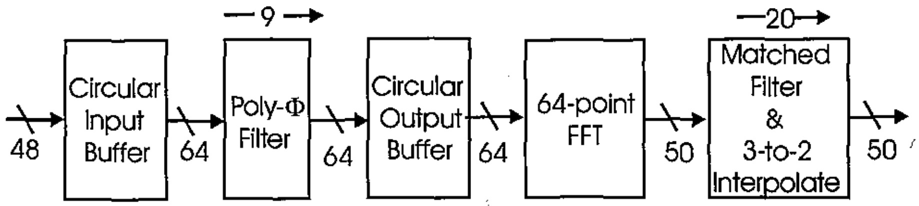


Figure 9.19 Filter Dimensions of Design Option 4

The workload to compute the 50 output samples is shown in (9.21) where, as earlier, we count a real multiply and add as an OP for operation. Comparing (9.21) with (9.10) we see that performing the channelizing and the full resampling task in the channelizer results in an increased workload from 68.3 to 78.4 ops per channel. The increase in workload is attributed to operating the matched filter at the higher input rate than in previous design options.

$$
\begin{aligned}
&\text{Polyphase Filter: } 2\cdot(64\cdot 9) &&= 1{,}152\\
&\text{FFT: } \quad 4\cdot\frac{64}{2}\log_2(64) &&= 768\\
&\text{Matched Filter: } 2\cdot(50\cdot 20) &&= 2{,}000\\
&\text{Total Number of OPS:} &&= 3{,}920\\
&\text{OPS per Channel:} &&= 78.4
\end{aligned}
\qquad (9.21)
$$

Option-5 Design:
In the option 5 design we perform the channelizing, matched filtering, and 48-to-1 resampling with the cyclically shifted buffers around the 64-path polyphase filter and with the FFT to separate the channels. This differs from earlier versions in which an output stage was required to perform interpolation and filtering. The polyphase filter length is the same as derived in (9.14) for option 2. The filter dimensions required to implement design option 5 are shown in Figure 9.20.

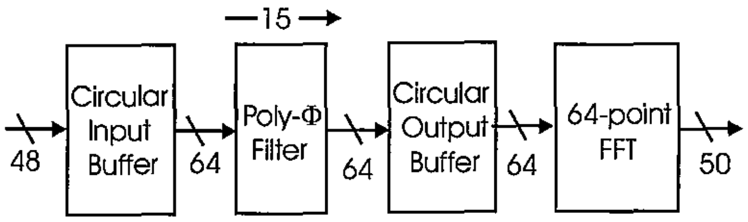


Figure 9.20 Filter Dimensions of Design Option 5

The workload to compute the 50 output samples is shown in (9.22) where, as earlier, we count a real multiply and add as an OP for operation. Comparing (9.22) with (9.10) we see that performing the channelizing, matched filtering, and the full resampling task in the channelizer results in an 22% decrease in workload from 68.3 to 53.7 ops per channel. The decrease in workload is attributed to operating only a single filtering process.

$$
\begin{aligned}
&\text{Polyphase Filter: } 2\cdot(64\cdot 15) &&= 1{,}920\\
&\text{FFT: } \quad 4\cdot\frac{64}{2}\log_2(64) &&= 768\\
&\text{Matched Filter: } 0 &&= 0\\
&\text{Total Number of OPS:} &&= 2{,}688\\
&\text{OPS per Channel:} &&= 53.7
\end{aligned}
\qquad (9.22)
$$

A final comparison of the workload per channel for the various channelizer approaches is found in Table 9-2. Here we see that for the 50-channel system we have examined, all the polyphase options require significantly less resources than the direct down-conversion option with typical workload ratios on the order of 20-to-1. We also see that there was only a 20% range of workload spread within the polyphase options examined from which we conclude the polyphase channelizer, the common element in the options, is the primary contributor to system efficiency.

Table 9-2 Comparison of Workloads per Channel for Various Channelizer Options

Design Option	Ops/Channel	Relative Efficiency wrt Direct Conversion	Relative Efficiency wrt 48-to-1 Polyphase
Direct Down Conversion	1384	100%	25.8
64-to-1 Polyphase Channelize & 3-to-4 MF	68.2	4.93%	1.27
64-to-1 Polyphase MF & 3-to-4 Interpolate	71.6	5.17%	1.33
32-to-1 Polyphase MF & 3-to-2 MF	68.3	4.93%	1.27
48-to-1 Polyphase Channelizer & MF	78.4	5.66%	1.46
48-to-1 Polyphase MF	53.7	3.88%	1.00

References

Bellanger, M., and Daguet, J., "TDM-FDM Transmultiplexer: Digital Polyphase and FFT," *IEEE Trans. Communications*, Vol. COM-22, Sept. 1974, pp.1199 - 1204.

Crochiere, Ronald, and Rabiner, Lawrence, *Multirate Signal Processing*, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1983.

Elliot, Doug, ed., *Handbook of Digital Signal Processing: Engineering Applications*, Chapter 8, "Time Domain Signal Processing with the DFT," pp. 639 - 666, Academic Press, 1987.

Fliege, Norbert, *Multirate Digital Signal Processing: Multirate Systems, Filter Banks, Wavelets*, West Sussex, John Wiley & Sons, Ltd., 1994.

Fliege, Norbert, "Polyphase FFT Filter Bank for QAM Data Transmission," *Proc. IEEE ISCAS'90*, pp. 654 - 657, 1990.

harris, fred, "On the Relationship Between Multirate Polyphase FIR Filters and Windowed, Overlapped FFT Processing," Twenty-third Annual Asilomar Conference on Signals and Computers, 1989.

Jovanovic-Dolecek, Gordana, *Multirate Systems: Design and Applications*, London, Idea Group, 2002.

Koilpillai, R. D., T. Q. Nguyen, and P. P. Vaidyanathan, "Some Results in the Theory of Crosstalk Free Transmultiplexer," *IEEE Trans. SP*. Vol.39, pp. 2174 - 2183, Oct. 1991.

Mitra, Sanjit, *Digital Signal Processing: A Computer-Based Approach*, 2nd ed., New York, McGraw-Hill, 2001.

Mitra, Sanjit, and James Kaiser, *Handbook for Digital Signal Processing*, New York, John Wiley & Sons, 1993.

Renfors, Markku, and T. Saramaki, "Recursive N-th Band Digital Filters, Parts I and II," *IEEE Trans. CAS*, Vol. 34, pp. 24 - 51, Jan. 1987.

Scheuermann, H., and Göckler, H.,"A Comprehensive Survey of Digital Transmultiplexing Methods," *Proc. IEEE*, Vol. 69, No. 11, Nov. 1981, pp. 1419 - 1450.

Vaidyanathan, P. P., *Multirate Systems and Filter Banks*, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1993.

Vaidyanathan, P. P., "Multirate Digital Filters, Filter Banks, Polyphase Networks and Applications: A Tutorial," *Proc. IEEE*, Vol. 78, pp. 56 - 93, Jan. 1990.

Problems

9.1 Design a prototype low-pass filter for a 10-to-1 polyphase partition with the following specifications:

Sample Rate	100 kHz		
Pass Band	0–4 kHz...........	Pass-band Ripple	0.1-dB
Stop Band	6–50 kHz	Stop-band Atten.	60-dB, –6-dB/Octave

What filter length satisfies the specifications?

What is the length of each path in a 10-path polyphase partition?

What is the frequency spacing obtained from the 10-path polyphase filter followed by a 10-point FFT?

Sketch the spectral response of three spectral bands at frequency –10, 0, and 10 kHz when the polyphase filter is operated at 10 kHz output rate, 10 inputs per output.

Sketch the spectral response of three spectral bands at frequency –10, 0, and 10 kHz when the polyphase filter is operated at 20 kHz output rate, 5 inputs per output.

9.2 Design a prototype low pass filter for a 10-to-1 polyphase partition with the following specifications:

Sample Rate	100 kHz		
Pass Band	0–4 kHz	Pass-band Ripple	0.1-dB
Stop Band	5–50 kHz	Stop-band Atten.	60-dB, –6-dB/Octave

What filter length satisfies the specifications?

What is the length of each path in a 10-path polyphase partition?

What is the frequency spacing obtained from the 10-path polyphase filter followed by a 10-point FFT?

Sketch the spectral response of three spectral bands at frequency –10, 0, and 10 kHz when the polyphase filter is operated at 10 kHz output rate, 10 inputs per output.

Sketch the spectral response of three spectral bands at frequency –10, 0, and 10 kHz when the polyphase filter is operated at 20 kHz output rate, 5-inputs per output.

9.3 Design a prototype low pass filter for a 10-to-1 polyphase partition with the following specifications:

Sample Rate	100 kHz		
Pass Band	0–4 kHz	Pass-band Ripple	0.1-dB
Stop Band	6–50 kHz	Stop-band Atten.	60-dB, –6-dB/Octave

Partition the prototype into a 10-path polyphase filter with a 10-point FFT following the partition. Deliver 1,000 samples of the input signal formed as a sum of equal amplitude, real sinusoids with random phase, and of frequencies 0.5, 1.1, 1.6, 2.1, 2.6, 3.1, 3.6, and 4.1 kHz, and the same list of frequencies with a 20 kHz offset.

Perform the fully decimated, 10-to-1 down sample, polyphase filtering and plot the time series and the spectra of each time series output from the 10 ports of the FFT following the polyphase partition. Comment on the signal levels in the channels adjacent to the occupied channels.

9.4 Design a prototype low pass filter for a 10-to-1 polyphase partition with the following specifications:

Sample Rate	100 kHz		
Pass Band	0–4 kHz	Pass band Ripple	0.1-dB
Stop Band	5–50 kHz	Stop band Atten.	60-dB, –6-dB/Octave

Partition the prototype into a 10-path polyphase filter with a 10-point FFT following the partition. Deliver 1,000 samples of the input signal formed as a sum of equal amplitude, real sinusoids with random phase, and of frequencies 0.5, 1.1, 1.6, 2.1, 2.6, 3.1, 3.6, and 4.1 kHz, and the same list of frequencies with a 20 kHz offset.

Perform the fully decimated, 10-to-1 down sample, polyphase filtering, and plot the time series and the spectra of each time series output from the 10 ports of the FFT following the polyphase partition. Comment on the signal levels in the channels adjacent to the occupied channels.

9.5 Repeat Problem 9.4 except that the filter should now be operated as a 5-to-1 down sample polyphase filter with output sample rate of 20 kHz rather than 10 kHz. Plot the time and spectra of each time series output from the 10 ports of the FFT following the polyphase partition. Comment on the signal levels in the channels adjacent to the occupied channels.

9.6 Design a prototype low pass filter for a 16-to-1 polyphase partition with the following specifications:

Sample Rate	320 kHz		
Pass Band	0-5 kHz	Pass band Ripple	0.1-dB
Stop Band	10–160 kHz	Stop band Atten	60-dB, -6-dB/Octave

What filter length satisfies the specifications?

What is the length of each path in a 16-path polyphase partition?

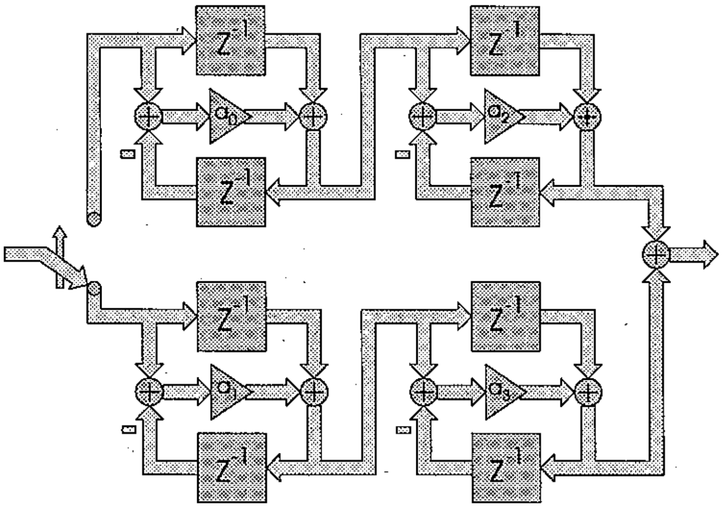
What is the frequency spacing obtained from the 16-path polyphase filter followed by a 16-point FFT?

Sketch the spectral response of three spectral bands at frequency –20, 0, and 20 kHz when the polyphase filter is operated at 20 kHz output rate, 16 inputs per output.

Sketch the spectral response of three spectral bands at frequency –20, 0, and 20 kHz when the polyphase filter is operated at 40 kHz output rate, 8-inputs per output.

CHAPTER 10

Recursive Polyphase Filters



When we first start a description of the resampling process we often use the model of a bandwidth-reducing filter followed by down sampling to a reduced sample rate commensurate with the reduced bandwidth. We argue that it is foolish to compute output samples that are destined to be discarded by the down sampler. This leads us to systems that compute only the required output samples. Such a filter does not compute an output sample for each input sample. Somewhere in the back of our mind we form this vague concept that the resampling operation can only be performed in nonrecursive filters. After all, a nonrecursive filter does not require a prior output to compute the next output and thus we are permitted to skip output samples. A recursive filter, on the other hand, does require prior outputs to form the next output. This suggests that the recursive filter must compute each output for its own benefit even if the down sampler has no need for it in the output data stream.

Recursive filters can in fact be resampled, and most of the structures we have developed for nonrecursive filters are applicable to recursive structures. Having direct access to the impulse response of the nonrecursive prototype filter facilitates the partition of the filter to form the stages of its polyphase version. While recursive filters can be similarly partitioned, not having direct access to the filter impulse response requires us to perform the partition with a different set of tools. The most effective approach to the partition of a recursive filter is to change the design procedure so that the partition is built into its structure. In this chapter we describe this embedding process that leads to efficient resampling of recursive filters

Nearly all of the filtering characteristics of a multirate partition are related to the phase versus frequency characteristics of the elementary building blocks forming the filter. It is the phase behavior that permits signals flowing through different paths to combine constructively or destructively and thus establish pass band and stop band regions. We know that spectral phase is preserved when a spectral region is translated by a heterodyne, and it is pleasing to know this property is also valid when the spectral region is translated by aliasing. This permits the constructive or destructive combining to occur prior to, or following, an aliasing process. Since phase is an important key to understanding the behavior of multirate systems, we examine recursive filters that are characterized as all-pass structures, filters that pass all input frequencies only affecting their phase response.

10.1 All-pass Recursive Filters

A very rich class of multirate filters can be formed from a set of elemental recursive all-pass subfilters. The structure of these filters is very different from traditional recursive digital filters. The filter structure can be introduced a number of ways, and a particularly enlightening form is as a tapped delay line, reminiscent of an M-point FIR filter, whose taps are all-pass filters formed by polynomials in Z^M. This structure is illustrated in Figure 10.1and then again in Figure 10.2 with the *k*th delay in the tapped delay line explicitly assigned to the *k*th path. The transfer function describing these structures is shown in (10.1).

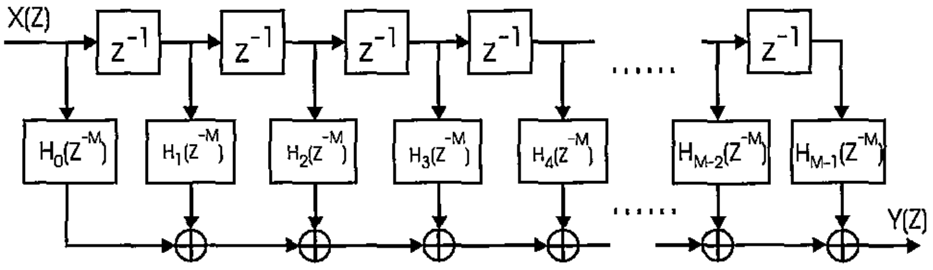


Figure 10.1 M-path Polyphase All-pass Filter Structure

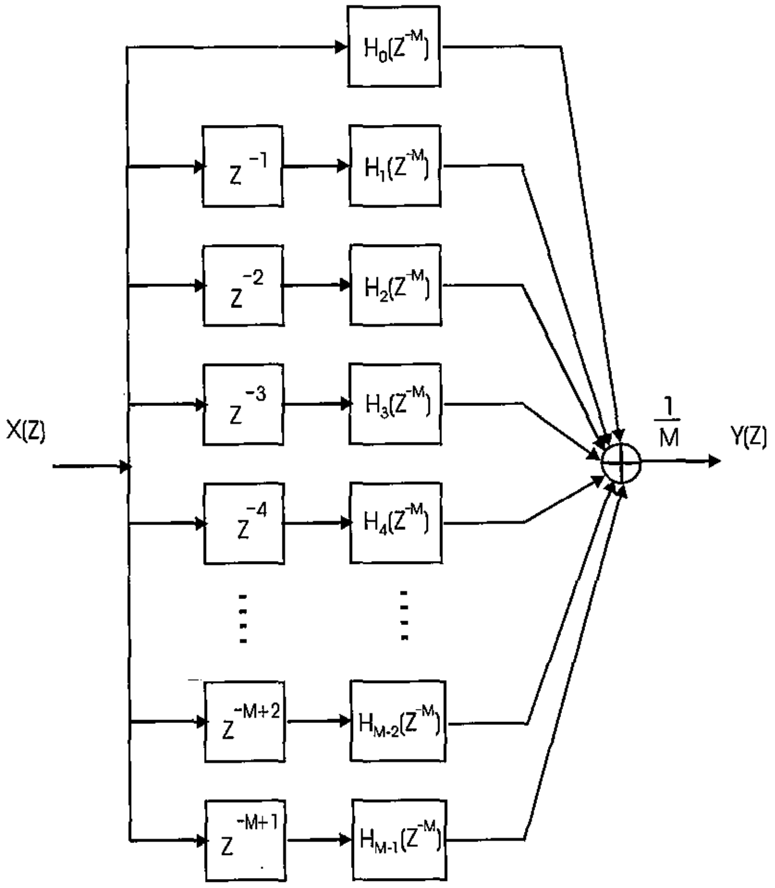


Figure 10.2 M-path Polyphase Filter with Delays Z^{-K} Allocated to the *K*th Path

$$H(Z) = \sum_{m=0}^{M-1} H_m(Z^{-M})Z^{-m} \tag{10.1}$$

We recognize the transfer function is the same form of a polyphase partition of a FIR filter. The difference in this structure is that the path transfer functions $H_m(Z^M)$ are all-pass recursive filters rather than the nonrecursive approximations to an all-pass filter obtained by partitioning a FIR filter. The most common application of the M-path polyphase filter is the case of a two-path filter (M=2), and we will spend a fair amount of time examining this form. Before we do so we will review the properties of recursive polyphase filters and examine useful architectures.

10.1.1 Properties of All-pass Filters

All-pass filters have sinusoidal steady state gains of unity for all input frequencies. This is stated concisely in (10.2).

$$|H(Z)|\big|_{Z=\exp(j\theta)} = |H(\theta)| = 1 \tag{10.2}$$

The simplest non-trivial all-pass network is a single delay line, represented by a single pole at the origin, as shown in (10.3). This single pole located at the origin contributes unity gain and a phase shift that varies linearly with input frequency as seen in Figure 10.3. We note that an all-pass network is characterized entirely by its phase shift.

$$H(Z) = \frac{1}{Z} \tag{10.3}$$

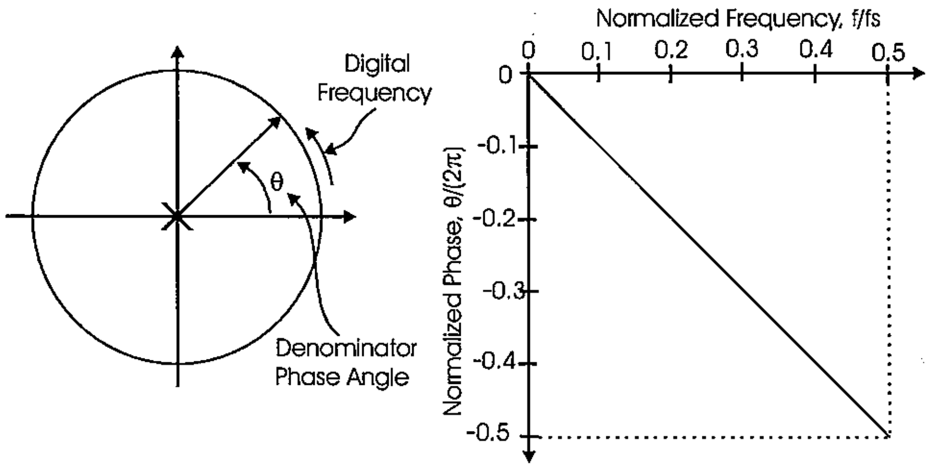


Figure 10.3 Pole Diagram and Phase as Function of Frequency for Single Delay Z^{-1}

The next nontrivial all-pass network is the first order filter in Z shown in (10.4). This form is a first order version of the general relationship presented in (10.5). A characteristic of an all-pass network is that the sequence of coefficients that describe the numerator in descending powers of Z is the index-reversed sequence that describes the denominator in descending powers of Z. The numerator is the reciprocal polynomial of the denominator polynomial. Reciprocal polynomials have reciprocal roots, thus all-pass networks have reciprocal poles and zeros. The pole-zero diagram for this simple first-order all-pass filter is shown in Figure 10.4. The phase response of this filter is the difference of the numerator phase and the denominator phase, which by simple trigonometric consideration is the included angle on the unit circle as indicated on the Figure as $-\phi$. The phase response corresponding to different values of the argument α is shown in Figure 10.5. From this figure we see that the phase angle must be zero at central angle $\theta = 0$, and must be $-\pi$ at central angle π.

$$H(Z) = \frac{1+\alpha Z}{Z+\alpha} \tag{10.4}$$

$$\begin{aligned} H(Z) &= \frac{N_M(Z)}{D_M(Z)} = \frac{Z^M D_M(Z^{-1})}{D_M(Z)} \\ &= \frac{1 + a_1 Z + a_2 Z^2 + \cdots + a_{M-1} Z^{M-1} + a_M Z^M}{Z^M + a_1 Z^{M-1} + a_2 Z^{M-2} + \cdots + a_{M-1} Z + a_M} \end{aligned} \tag{10.5}$$

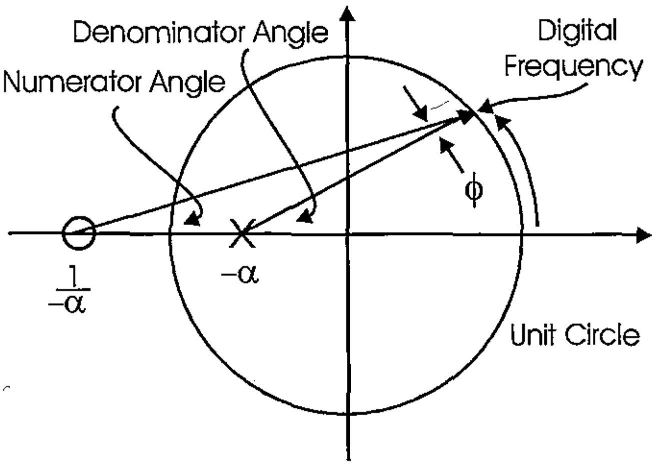


Figure 10.4 Pole-zero Diagram of First Order All-pass Networks [(1+ αZ)/(Z+α)]: Showing Phase Angles that Form Output Phase Angle

Note that for the first-order all-pass filter, the transfer function for $\alpha = 0$ defaults to the pure delay, and as expected, its phase function is linear with frequency. The phase function is a function of the pole position α, and this network can be thought of, and will be used as, the generalized delay element. We observe that the phase function is anchored at its end points (0 degrees at zero frequency and 180 degrees at the half sample rate) and that it warps with variation in α. It bows upward (less phase) for positive α and bows downward (more phase) for negative α. The bowing phase function permits us to use the generalized delay to obtain a specified phase angle at any frequency. For instance, we note that when $\alpha = 0$, the frequency for which we realize 90-degrees phase shift is 0.25 (the quarter sample rate). We can determine a value of α for which the 90 degree phase shift is obtained at any normalized frequency such as at normalized frequency 0.45 ($\alpha = 0.8$) or at normalized frequency 0.05 ($\alpha = -0.73$).

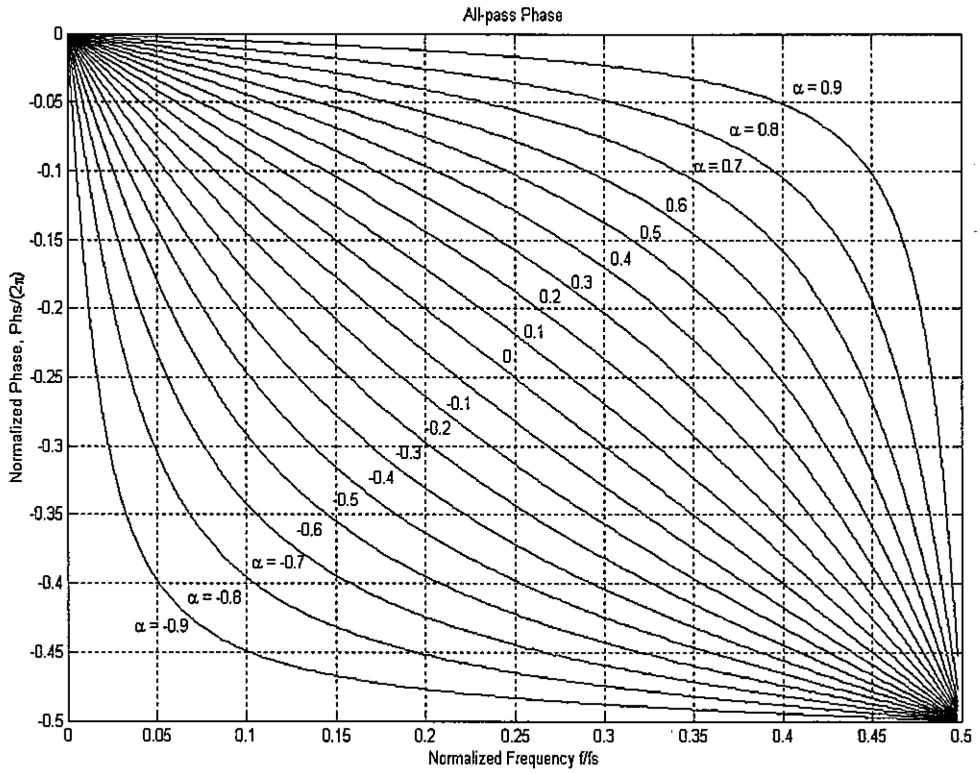


Figure 10.5 Phase Response of First Order in Z All-pass Network as Function of Pole Position α (α = 0.9, 0.8,....,0,....,–0.8, –0.9)

An important variant of this first-order filter is obtained by converting it into a first-order polynomial in Z^2. This is equivalent to zero packing its impulse response, which has the effect of replicating its spectrum as we traverse the unit circle. We see two copies of the spectrum or phase response as we pass the two poles of the transfer function rather than the single pole. The form of this transfer function is shown in (10.6).

$$H(Z)=\frac{1+\alpha Z^2}{Z^2+\alpha} \tag{10.6}$$

The pole-zero diagram of this transfer function is shown in Figure 10.6 where we note that all roots reside on the imaginary axis.

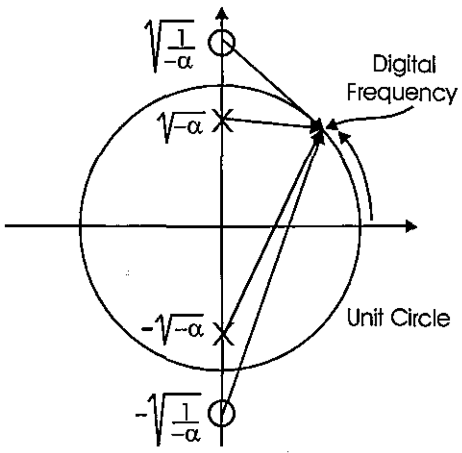


Figure 10.6 Pole-zero Diagram of First Order in Z^2 All-pass Networks $[(1+\alpha Z^2)/(Z^2+\alpha)]$

The phase response of the first-order polynomial in Z^2 is seen in Figure 10.7. Note that the first half of the phase response matches the response seen in Figure 10.5. The second half is the phase-continuous extension we would have seen in Figure 10.5 had we finished traversing the unit circle. The phase response of the first-order filter in Z^2 is seen to be a *lazy S*, which for positive α ramps slowly to the right, falls rapidly, and again continues to ramp slowly to the right. We will be using this lazy-S phase behavior shortly.

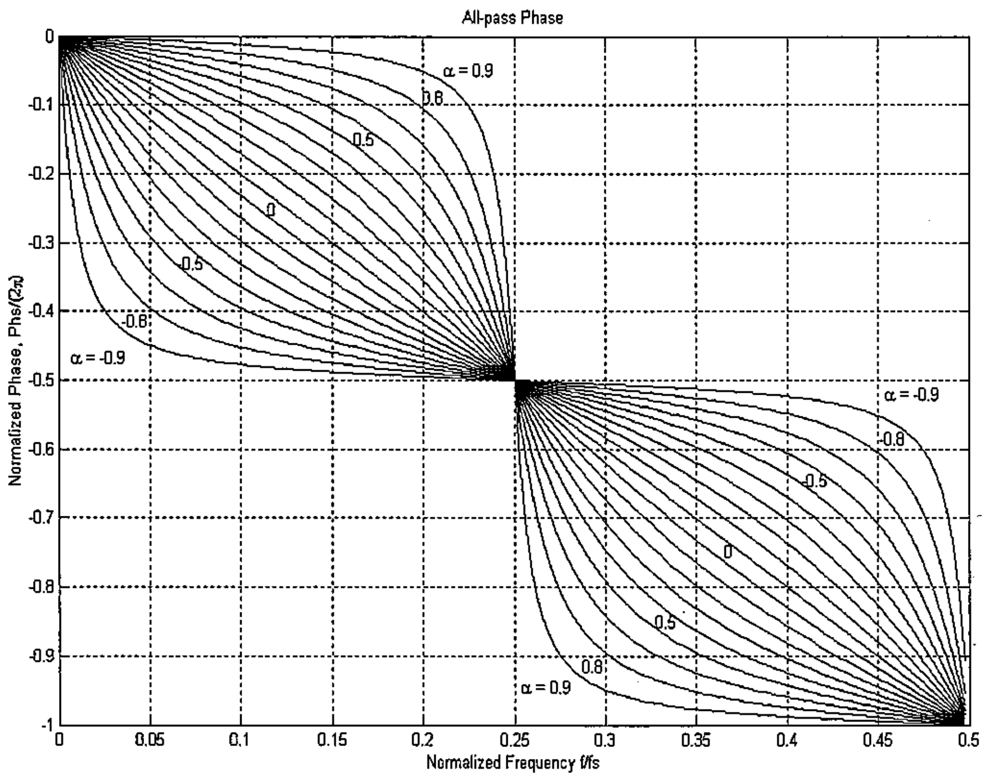


Figure 10.7 Phase Response of First-order (in Z^2) All-pass Network as Function of Pole Position α (α =0.9, 0.8,....,0,.....,–0.8, –0.9)

A closed form expression for the phase function is obtained by evaluating the transfer function on the unit circle. This is done in the following derivation. Equation (10.7) presents the transfer function for the first order filter in Z^M. Initially, we will be interested in this form of the all-pass filter for the two cases, M = 1 and M = 2.

$$H(Z) = \frac{1+\alpha Z^M}{Z^M+\alpha} \tag{10.7}$$

Equation (10.8) starts the process of evaluating the network on the unit circle by replacing the indeterminate Z with $e^{j\theta}$.

$$H(\theta) = \frac{1+\alpha e^{jM\theta}}{e^{jM\theta}+\alpha} \tag{10.8}$$

In (10.9) we factor from numerator and denominator the square root of the exponential term $e^{jM\theta/2}$.

$$H(\theta) = \frac{e^{-jM\theta/2} + \alpha\, e^{jM\theta/2}}{e^{jM\theta/2} + \alpha\, e^{-jM\theta/2}} \tag{10.9}$$

In (10.10) we replace the complex exponential $e^{j(-)}$ with $\cos(-) + j\sin(-)$.

$$H(\theta) = \frac{\cos(M\theta/2) - j\sin(M\theta/2) + \alpha\cos(M\theta/2) + j\alpha\sin(M\theta/2)}{\cos(M\theta/2) + j\sin(M\theta/2) + \alpha\cos(M\theta/2) - j\alpha\sin(M\theta/2)} \tag{10.10}$$

We next gather up the real and imaginary components of the numerator and denominator and divide numerator and denominator by the common real part. These steps are shown in equations (10.11) through (10.13).

$$H(\theta) = \frac{(1+\alpha)\cos(M\theta/2) \;-\; j(1-\alpha)\sin(M\theta/2)}{(1+\alpha)\cos(M\theta/2) \;+\; j(1-\alpha)\sin(M\theta/2)} \tag{10.11}$$

$$H(\theta) = \frac{1 - j\,\dfrac{(1-\alpha)\sin(M\theta/2)}{(1+\alpha)\cos(M\theta/2)}}{1 + j\,\dfrac{(1-\alpha)\sin(M\theta/2)}{(1+\alpha)\cos(M\theta/2)}} \tag{10.12}$$

$$H(\theta) = \frac{1 - j\dfrac{(1-\alpha)}{(1+\alpha)}\tan(M\dfrac{\theta}{2})}{1 + j\dfrac{(1-\alpha)}{(1+\alpha)}\tan(M\dfrac{\theta}{2})} \tag{10.13}$$

The components of the numerator and denominator of (10.13) can be visualized with the aid of Figure 10.8. The numerator and denominator of (10.13) share the same real part and differ only by the sign of their imaginary parts. The magnitude of the numerator and denominator of (10.13) are seen to be the hypotenuse of the two right triangles with base of unit length. Thus the magnitude of the transfer function (the ratio of the numerator and denominator magnitudes) must be unity, and the phase angle of the transfer function is minus twice the angle formed by the numerator right triangle. The phase of the first-order low pass network is shown in (10.14). Note that if $\alpha = 0$, the phase 2ϕ reverts to a linear phase proportional to frequency θ, with the proportionality factor M, originally the degree of the first-order polynomial, now identifying the number of linear delay elements.

$$\phi = -2\,\text{ATAN}\left[\frac{(1\text{-}\alpha)}{(1+\alpha)}\,TAN(M\frac{\theta}{2})\right] \tag{10.14}$$

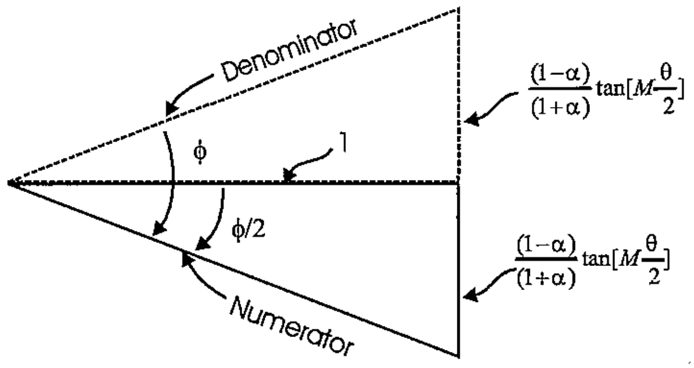

Denominator
ϕ
1
$\phi/2$
Numerator
$\frac{(1-\alpha)}{(1+\alpha)}\tan[M\frac{\theta}{2}]$
$\frac{(1-\alpha)}{(1+\alpha)}\tan[M\frac{\theta}{2}]$

Figure 10.8 Visualization of Components in Equation (10.13)

10.1.2 Implementing First-order All-pass Networks

The first-order all-pass network can be implemented in a number of architectures. We now review a few of them and identify those with desirable finite-precision arithmetic performance or with a reduced number of arithmetic operations. The obvious implementations of the all-pass transfer function, shown in (10.15), is the factored form shown to represent the filter in a canonic form with one feedback and one feed forward path from the common delay element. The structure of this filter is presented in Figure 10.9.

$$H(Z) = \frac{1+\alpha Z}{Z+\alpha} = \frac{Z^{-1}+\alpha}{1+\alpha Z^{-1}} = \frac{1}{1+\alpha Z^{-1}}[\alpha+Z^{-1}] \tag{10.15}$$

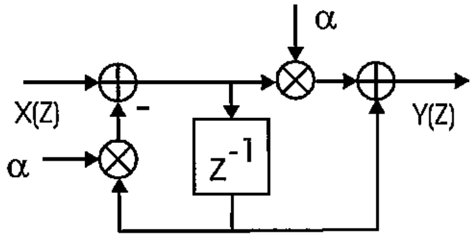


Figure 10.9 Canonic Implementation of First-order All-pass Filter

This form of the all-pass filter does not have two different coefficients for the roots α and 1/α. This avoids the common problem in finite precision implementations related to the fact that Q(α) and Q(1/α), where Q( ) represents the quantization process, are not reciprocals so that the filter is not all-pass. For example, with α =1/3, a 12-bit quantized version of Q(α) = 0.3330078 while Q(1/α) = 3, with product Q(α) Q(1/α) equal to 0.9990234. The primary objection to this form of the filter is that it requires 2 multiplications by the same coefficient α.

An alternate representation of the all-pass filter reverses the order of the generating the pole-zero pair. In the previous version we implement the pole at the input to the filter and the zero at its output. In the second option we implement the zero at the input to the filter and the pole at its output. The appropriate factoring of the transfer function is shown in (10.16), and the form of the reordered implementation is shown in Figure 10.10.

$$H(Z) = \frac{1+\alpha Z}{Z+\alpha} = \frac{Z^{-1}+\alpha}{1+\alpha Z^{-1}} = [Z^{-1}+\alpha]\frac{1}{1+\alpha Z^{-1}} \tag{10.16}$$

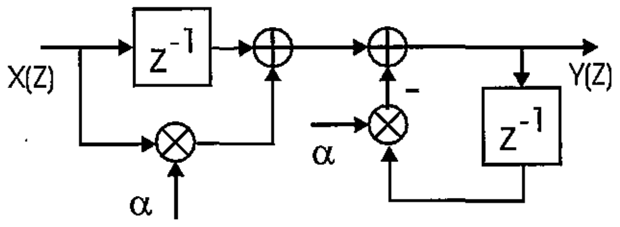


Figure 10.10 Reordered All-pass Filter Structure

We note that the feed forward and feedback sections of the transfer function have the same valued coefficient and that the outputs of their products are summed to form the output Y(Z). The two inputs to the multipliers can first be combined in a pre-sum and a single multiply can be used to apply the coefficient to both the feed forward and the feedback paths. This structure is seen in Figure 10.11 in the general form, as a first-order filter in Z^M and as a processing block indicating a single coefficient α.

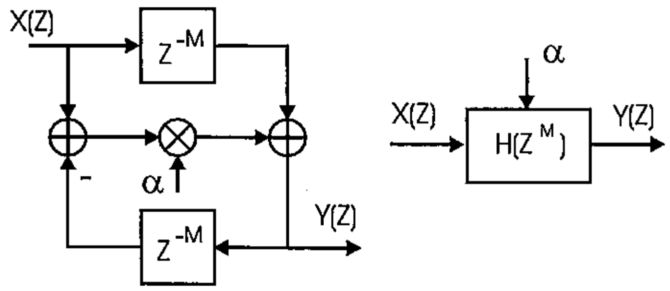


Figure 10.11 Single Coefficient All-pass Filter Structure

A major attraction of this form of the all-pass filter structure is that since the same coefficient is applied to the numerator and denominator, the system maintains all-pass properties for any degree of coefficient quantization (but not quantized arithmetic). The primary feature of this structure is that a single multiplication forms both the numerator and denominator. One multiply here replaces two in the previous version of the filter. An interesting property of this form is that the feed forward register, containing delayed versions of the input, and the feedback register, containing delayed versions of the unity-gain output signal, do not require wider bit fields to accommodate processing gain word growth. The feedback register, the multiplier, and adders do however require extension into lower order bits to preserve significance in the difference formed at the input adder. This form has another advantage when all-pass stages are cascaded. Since the feedback register of one stage contains the same data as the feed forward register of the next stage, the two registers can be folded into a single register serving both feedback and feed forward. This will be demonstrated in the next section.

One final advantage of this structure is that we have easy access to a delayed version of the output of the all-pass filter. This delayed output is available from the feedback delay, and when M = 1, the delayed sample is available at the output of the feedback register. We will have need for a delayed version of the all-pass output when we implement the low pass to band-pass transformation. We discuss this option in detail in a later section when we translate the frequency response of a prototype low pass filter to a different center frequency. This transformation replaces a delay (a linear phase shift) with a delay plus a generalized delay (a nonlinear phase shift). The transformation is shown in (10.17).

$$\frac{1}{Z} \Rightarrow -\frac{1}{Z}\,\frac{1-cZ}{Z-c}, \quad c=\cos(2\pi\frac{f_C}{f_S}) \tag{10.17}$$

The dual graph of the structure presented in Figure 10.11 is also a single coefficient version of the all-pass filter. Recall that in the dual structure, a summing junction replaces each node, a node replaces each summing junction, and the direction of the signal flow is reversed. The dual version of this graph is shown in Figure 10.12.

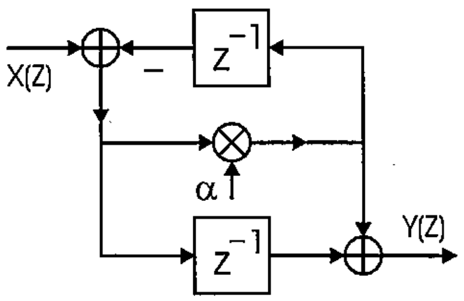


Figure 10.12 Single Coefficient All-pass Filter Dual Structure

The dual graph structure is not as interesting as the original and is included only for completeness. This form of the filter does not exhibit any attractive efficiency options comparable to the register folding and sharing that is possible for the nondual form when successive stages are operated in cascade. While on the topic of completeness, another common, single multiply all-pass structure is shown in Figure 10.13. This structure requires a single delay element but uses three adds instead of the two adds required in the structure presented in Figure 10.11.

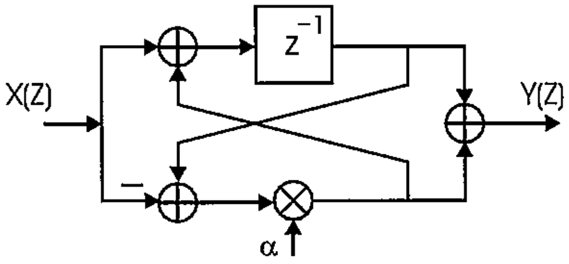
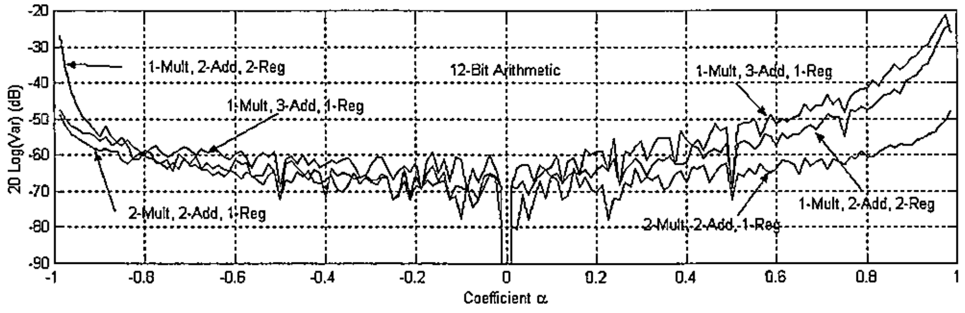


Figure 10.13 Alternate Single Coefficient All-pass Structure

The all-pass filter structures exhibit similar range of frequency-dependent sensitivity to finite arithmetic noise. The magnitude responses of the filters deviate slightly from unity gain when we truncate finite arithmetic multiplication and addition. Figure 10.14 compares the average spectral variance from the response of three forms of the filters implemented with 12-bit and with 16-bit coefficients and with 18-bit and 24-bit feedback registers respectively. Note the slight asymmetry in variance between positive and negative values of the coefficient α and the increased variance as the roots approach the unit circle ($\alpha \rightarrow 1$) and become high Q roots. The canonic filter with 2-multiplies, 2-adds, and 1-register has the lowest variance while the filter with 1-multiply, 3-adds, and 1-register has the highest. The filter with 1-multiply, 2-adds, and 2-registers has the smallest number of arithmetic operations and exhibits a variance between the other filter options. The advantage of the register-sharing options available in the two-register form of the filter warrants its use in most applications.



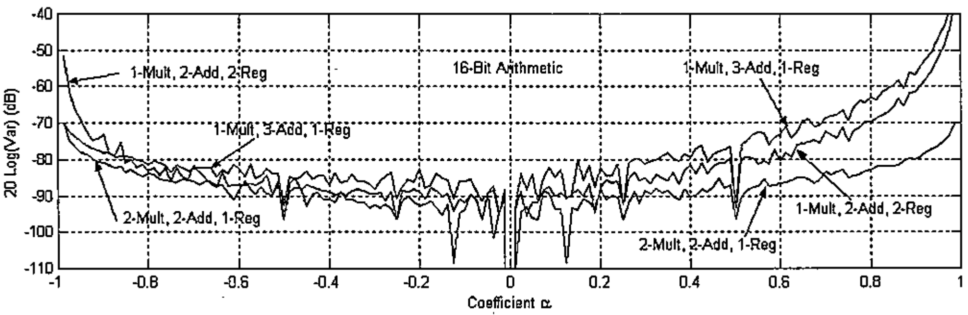


Figure 10.14 Spectral Variance for Three Forms of All-pass Filter for 12-bit and 16-bit Coefficients as Function of Coefficient α

10.2 Two-path All-pass Recursive Filters

We now have access to elemental all-pass structures required to form the filters we will be using as building blocks for our final designs. Rather than examine other low-level building block filters, we change our approach and use a specific filter structure to view the composite filter from a top-down view and develop a perspective of how the all-pass structure leads to this different class of filters. Once we have developed the structure and perspective for a particularly simple class of filters we will expand the class by invoking a simple set of frequency transformations. The structure we eventually master, formed with the all-pass filters, is a 2-path polyphase filter structure. This is essentially a tapped delay line, of length 2, with each tap formed as a cascade of filters in powers of Z^2. This structure is presented in Figure 10.15.

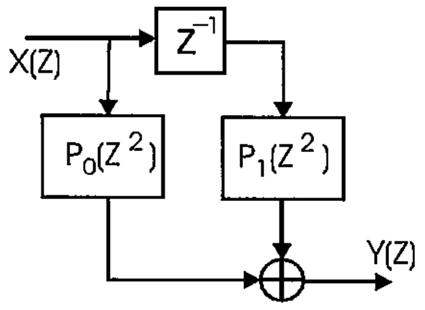


Figure 10.15 Two-path Polyphase Filter

The 2-path structure can be redrawn as shown in Figure 10.16. The delay in the lower path can be placed on either side of the all-pass network. As observed earlier, this delay already resides in the feedback path of the all-pass filter and may be extracted from the filter as opposed to being inserted prior to the filter. When the filter participates in a multirate configuration, a commutator port replaces the delay, and the delay is originally positioned on the side of the filter operating at the higher of the two rates.

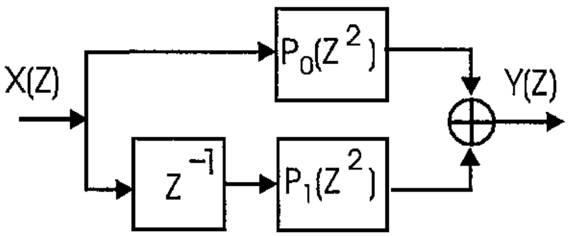


Figure 10.16 Redrawn 2-path Polyphase Filter

10.2.1 Two-path Half-band Filters: Nonuniform Phase

This filter structure shown in Figure 10.16 offers a surprisingly rich class of filter responses that can be formed by the all-pass building blocks. By restricting our initial dialogue to 2-path filters, we can visit familiar filter characteristics and demonstrate the wide versatility of the 2-path structure while comparing relative efficiencies of the all-pass and classical structures. The 2-path structure can implement half-band low pass and high pass filters, as well as Hilbert transform filters that exhibit minimum or nonminimum phase response. The two-path filter can implement standard recursive filters such as the Butterworth and Elliptic filters. MATLAB routines, *tony_des2,* and *tonyc* that compute the coefficients of the two-path filter and a number of its variants, are include in the software available from the book's companion website. Also, the half-band filters can be configured to embed a 1-to-2 up sampling or a 2-to-1 down sampling operation within the filtering process. The prototype half-band filters have their 3-dB band edge at the quarter sample rate. All-pass filters can be used to apply frequency transformations to the 2-path prototype, to form arbitrary bandwidth low pass and high pass complementary filters, and arbitrary center frequency pass band and stop band complementary filters. These same all-pass frequency transformations can be applied to any standard filter structure to obtain desired parameter adjustable spectral transformations. Zero packing the time response of the 2-path filter, another trivial all-pass transformation, causes spectral scaling and replication. The zero-packed structures used in cascade with other filters in iterative filter designs achieve composite spectral responses exhibiting very narrow transition bandwidths with low-order filters.

The specific form of the prototype 2-path filter is shown in Figure 10.17. The number of poles (or order of the polynomials) in the two paths differs by precisely one, with the one extra pole located at the origin and conveyed by the single delay in the lower leg. Since the denominator terms in the all-pass filters are of the form ($Z^2+\alpha_k$), for positive α_k, all poles of this filter are restricted to the imaginary axis. The structure forms complementary low pass and high pass filters, from the scaled sum and difference respectively, of the outputs from the two paths. An important idea related to polyphase filter partitions is that the scaled sum and difference are in fact a 2-point transform of the pair of outputs and represent a phase-rotated coherent sum of the path outputs. The scale factor of this sum and difference is 1/2.

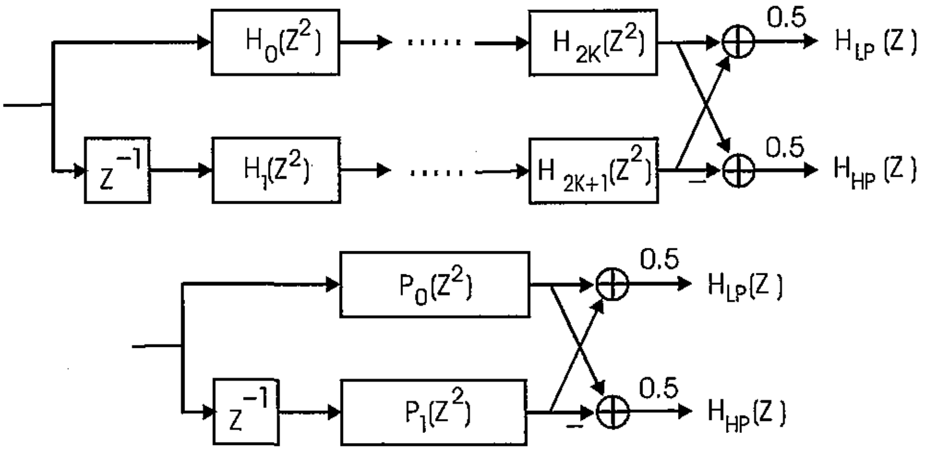


Figure 10.17 Two-path All-pass Filter

The cascade of first-order polynomials in Z^2 in each of the legs of the two-path filter can be implemented by a set of identical stages with different coefficients. This model, shown in Figure 10.18, reflects our standard way of implementing a recursive filter as a sequence of first order in Z^2 filter stages. The particular form we present here can also be re-cast in a more efficient coupled architecture. This architecture takes advantage of the earlier observation that the feedback registers of one stage contain the same data as the feed forward registers of the next stage and consequently can be shared rather than replicated. The architecture that shares the feedback and feed forward registers is shown in Figure 10.19.

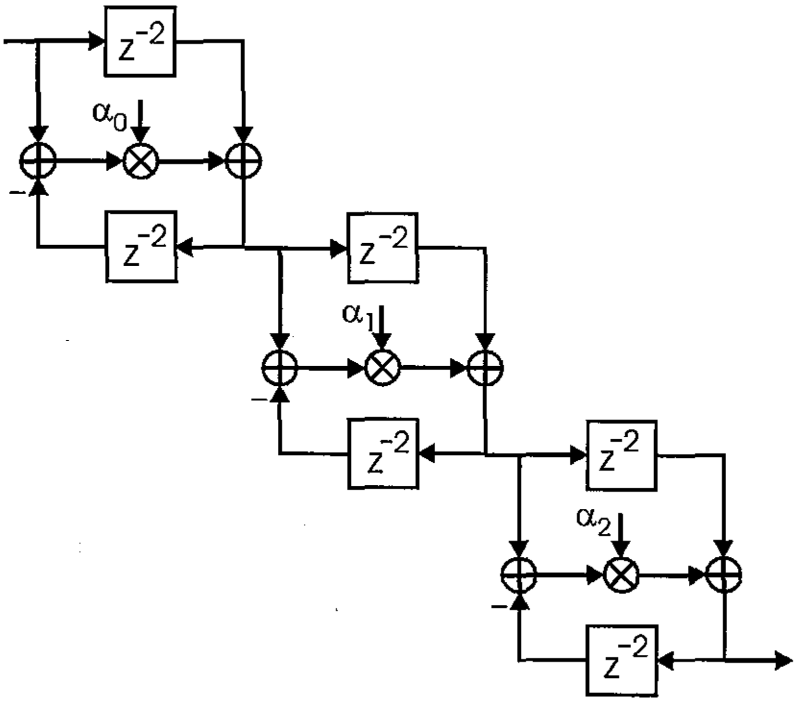


Figure 10.18 Direct Implementation of Cascade First-order Filters in Z^2

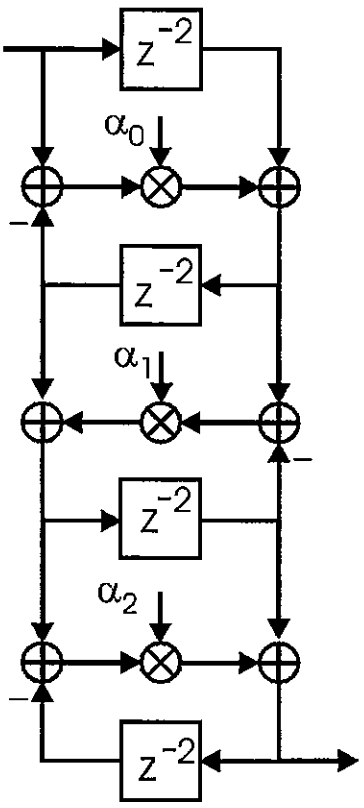


Figure 10.19 Folded Implementation of Cascade First-order Filters in Z^2

The transfer function of the 2-path filter shown in Figure 10.17 is shown in (10.18).

$$
\begin{aligned}
&H(Z) = P_0(Z^2) \pm Z^{-1} P_1(Z^2) \\
&P_i(Z^2) = \prod_{k=0}^{K_i} H_{i,k}(Z^2), \ i = 0,1 \\
&H_{i,k}(Z^2) = \frac{1+\alpha(i,k)Z^2}{Z^2+\alpha(i,k)}
\end{aligned}
\tag{10.18}
$$

In particular, we can examine the trivial case of a single all-pass filter in path-0 and a delay only in path-1. The transfer function for this case is shown in (10.19).

$$\begin{aligned} H(Z) &= \frac{1+\alpha_0 Z^2}{Z^2+\alpha_0} \pm \frac{1}{Z} = \frac{\alpha_0 Z^3 \pm Z^2 + Z \pm \alpha_0}{Z(Z^2+\alpha_0)} \\ &= \frac{\alpha_0(Z \pm 1)[Z^2 \mp (\frac{1}{\alpha_0} - 1) Z + 1]}{Z(Z^2+\alpha_0)} \end{aligned} \tag{10.19}$$

We note a number of interesting properties of this transfer function, applicable to all the 2-path prototype filters. The denominator roots are on the imaginary axis restricted to the interval ± 1 to assure stability. The numerator is a linear-phase FIR filter with an even symmetric weight vector. As such the numerator roots must appear either on the unit circle, or if off and real, in reciprocal pairs, and if off and complex, in reciprocal conjugate quads. Thus for appropriate choice of the filter weight(s), the zeros of the transfer function can be placed on the unit circle and can be distributed to obtain equal ripple stop band response. In addition, due to the one-pole difference between the two paths, the numerator must have a root at ± 1. When the two paths are added, the numerator roots are located in the left half plane, and when subtracted, the numerator roots are mirror imaged to the right half plane forming low pass and high pass filters respectively.

For the example shown in (10.19), the numerator roots are on the unit circle for a limited range of α_0. If we compare the numerator with the sequence obtained from Pascal's triangle we can determine the parameter α_0 to obtain a Butterworth filter, with repeated zeros at –1, the half sample rate. This value is $\alpha_0 = 1/3$. For values of α_0 greater than 1/3, two of the roots leave –1 and migrate as conjugate roots along the unit circle until they reach ± j1 where they meet and annihilate the poles migrating away from the origin along the y axis. As the zeros start their journey around the unit circle the transition bandwidth is reduced while a stop band side-lobe rises between the zeros. A value of α_0 can be found to realize any side-lobe level. For this example, the filter contains three poles and three unit-circle zeros, a total of 5 nontrivial roots for the cost of one multiply per input point.

The attraction of this class of filters is the unusual manner in which the transfer function zeros are formed. The zeros of the all-pass sub filters reside outside the unit circle (at the reciprocal of the stable pole positions) but migrate to the unit circle as a result of the sum or difference of the two paths. The zeros occur on the unit circle because of destructive cancellation of spectral components delivered to the summing junction via the two distinct paths, as opposed to being formed by numerator weights in the feed forward path of a standard biquadratic filter. The stop band zeros are a windfall. They start as the maximum phase all-pass zeros formed concurrently with the all-pass denominator roots by a single shared coefficient and migrate to the unit circle in response to addition of the two path signals.

We now examine a filter containing a recursive all-pass stage in path-0 and a delay and recursive all-pass filter in path-1. This transfer function is shown in (10.20).

$$\begin{aligned} H(Z) &= \frac{1+\alpha_0 Z^2}{Z^2+\alpha_0} \pm \frac{1}{Z}\frac{1+\alpha_1 Z^2}{Z^2+\alpha_1} \\ &= \frac{\alpha_0 Z^5 \pm \alpha_1 Z^4 + (1+\alpha_0\ \alpha_1)Z^3 \mp (1+\alpha_0\ \alpha_1)Z^2 + \alpha_1 Z + \alpha_0]}{Z\,(Z^2+\alpha_0)\,(Z^2+\alpha_1)} \end{aligned} \tag{10.20}$$

This transfer function has 5-poles on the imaginary axis in the interval ±1, and for appropriate selection of the two coefficients five unit-circle zeros on the left half of the unit circle (for the sum of the paths) and the mirror reflection zeros on the right half of the unit circle (for the difference of the paths). As in the previous example, coefficients of the numerator can be compared to that of the appropriate Pascal triangle coefficient list to place all zeros at −1. This occurs when $\alpha_0 = 0.1055728$ and $\alpha_1 = 0.5278640$. For increased values, the roots migrate along the unit circle towards $\pm j1$, reducing the transition bandwidth and forming stop band side-lobes. By judicious choice of the two weights, the filter can be designed for equal-ripple side-lobes of any level. For instance we obtain a stop band edge at 0.370 with −60 equal-ripple side-lobes when $\alpha_0 = 0.1413486$ and $\alpha_1 = 0.5899948$. These weights correspond to a half-band elliptic filter with restricted pole positions.

Figure 10.20 presents the pole-zero diagram for the 2-path filter formed with these weights. In this example the path-0 is formed with one first-order section in Z^2 and path-1 is formed with one first-order section in Z^2 and a single delay Z^{-1}. The composite filter thus contains 5 poles and 5 zeros and requires one coefficient for path-0 and one coefficient for path-1. The design routine *Tony_des2* computed weights for the fifth-order polynomial to obtain −60-dB equal-ripple side-lobes. The −3-dB pass band edge is located at a normalized frequency of 0.25, and the stop band edge that achieved the desired −60-dB stop band ripple is located at a normalized frequency of 0.370. The coefficients of the filter are listed here in decreasing order of Z:

Path-0	**Polynomial Coefficients:**
Filter-0	[1 0 0.1413486]
Path-1	**Polynomial Coefficients:**
Filter-1	[1 0 0.5899948]

The roots presented in Figure 10.20 describe the low pass filter formed from the 2-path filter. Figure 10.21 presents the phase slopes of the two paths of this filter as well as the frequency response, impulse response, and group-delay response. We note that the zeros of the spectrum correspond to the zero locations on the unit circle in the pole-zero diagram. The low-frequency group-delay matches the delay to the peak time response, and we note the expected increased group delay at the filter band edge due to the dominant poles near the unit circle.

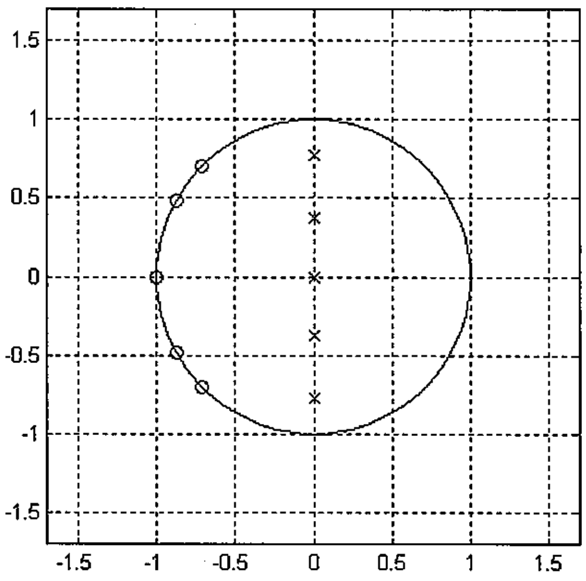


Figure 10.20 Pole-zero Diagram of 2-path, 5-pole, 2-multiplier Filter

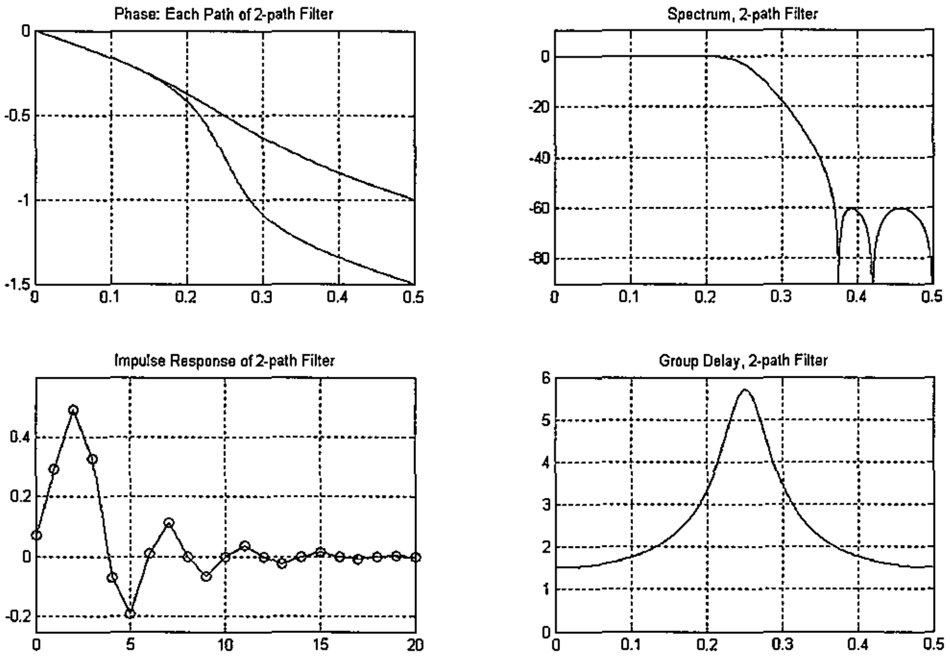


Figure 10.21 Two-path Phase Slopes, Frequency, Impulse, and Phase Responses of 2-path, 5-pole, 2-multiplier Filter

Figure 10.22 presents the pole-zero diagram for a higher-order 2-path filter. In this example the path-0 is formed with two first-order sections in Z^2 and path-1 is formed with two first-order sections in Z^2 and a single delay Z^{-1}. The composite filter thus contains 9 poles and 9 zeros and requires two coefficients for path-0 and two coefficients for path-1.

The *Tony_des2* design routine was used to compute weights for the ninth-order filter with –60-dB equal-ripple stop band. The –3-dB pass band edge is located at a normalized frequency of 0.25 and the stop band edge that achieved the desired 60-dB stop band attenuation is located at a normalized frequency of 0.284. This too is an elliptic filter with constraints on the pole positions. The coefficients of the filter are listed here in decreasing order of Z:

Path-0	**Polynomial Coefficients**
Filter-0	[1 0 0.101467517]
Filter-2	[1 0 0.612422841]

Path-1	**Polynomial Coefficients**
Filter-1	[1 0 0.342095596]
Filter-3	[1 0 0.867647439]

Figure 10.22 presents the roots of the low pass filter formed by the sum of the all-pass transfer functions of the 2-path filter. The high pass roots of the same filter would have the same poles but the zeros would be reflected about the imaginary axis appearing in the low-frequency band on the unit circle. Figure 10.23 presents the phase slopes of the two paths of this filter as well as the frequency response, impulse response, and group-delay response. We note that the zeros of the spectrum correspond to the zero locations on the unit circle in the pole-zero diagram. The low-frequency group delay matches the delay to the peak time response, and we note the expected larger group delay at the filter band edge due to the dominant poles being nearer the unit circle.

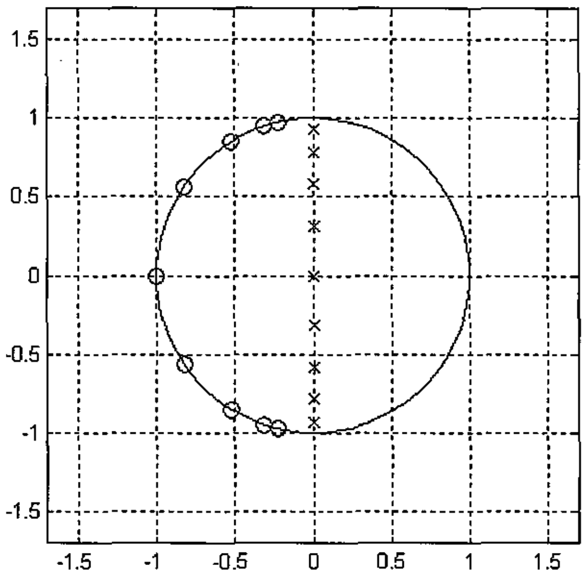


Figure 10.22 Pole-zero Diagram of 2-path, 9-pole, 4-multiplier Filter

Figure 10.24 presents a more detailed view of the phase response for each path of the 9-pole, 2-path filter. The dashed lines represent the phase response of the two paths when the filter coefficients are set to zero. In this case the two paths default to 2 delays in the top path and 3 delays in the bottom path. Since the two paths differ by one delay, the phase shift difference is precisely 180 degrees at the half-sample rate. When the coefficients of the filters in each path are adjusted to their design values, the phase responses of both paths assume the bowed lazy S curve described in Figure 10.17. Note that at low frequencies, the two phase curves exhibit the same phase profile and that at high frequencies, the two phase curves maintain the same 180-degree phase difference. Thus the addition of the signals from the two paths will lead to a gain of 2 in the band of frequencies with the same phase and will result in destructive cancellation in the band of frequencies with 180-degree phase difference. These two bands, of course, are the pass band and stop band, respectively. We note that the two phase curves differ by exactly 180 degrees at only four distinct frequencies as well as the half-sample rate: the frequencies corresponding to the spectral zeros of the filter. Between these zeros, the filter exhibits stop band side-lobes that by design are equal ripple.

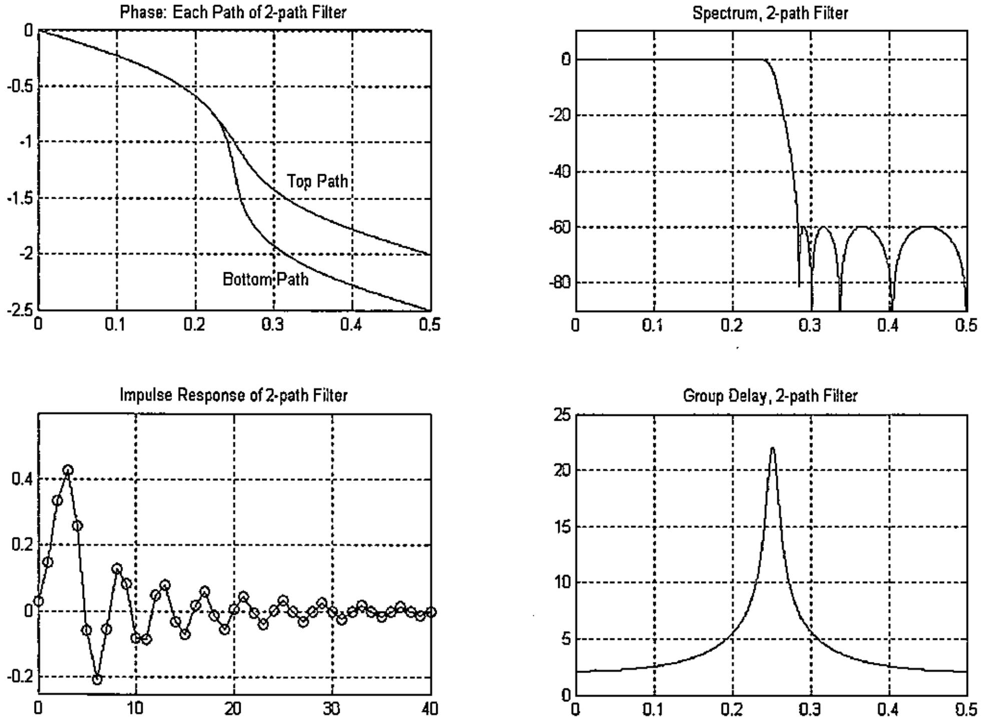


Figure 10.23 Two-path Phase Slopes, Frequency, Impulse, and Phase Responses of 2-path, 9-pole, 4-multiplier Filter

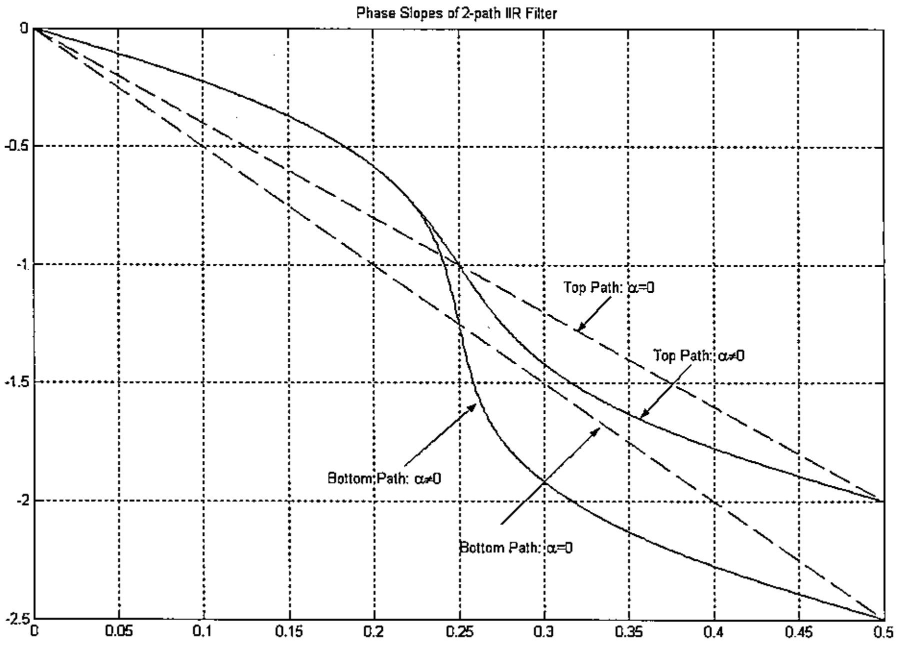


Figure 10.24 Phase Response of Both Paths of 2-path, 9-pole, 4-multiplier Filter

10.2.2 Two-path Half-band Filters: Linear Phase

We can modify the structure of the 2-path filter to form filters with approximately linear-phase response by restricting one of the paths to be pure delay. We accomplish this by setting all the coefficients of the filter in the upper leg to be zero. This forces the all-pass filters in this leg to revert to their default responses of pure delay with poles at the origin. As we pursue the solution to the phase matching problem in the equal-ripple approximation, we find that the all-pass poles must move off the imaginary axis. In order to keep real coefficients for the all-pass filters, we make one minor change in the all-pass filter structure. The change is to permit the cascade filters in the lower path to contain first- and second-order filters in Z^2. We lose degrees of freedom in the filter design process when we set the phase slope in one path to be a constant. Consequently, when we design an equal-ripple group delay approximation to a specified performance we will need additional all-pass sections. To meet the same out-of-band attenuation and the same stop band band edge as the nonlinear phase design of the previous section, our design routine, *lineardesign*, determined that we require two first-order filters in Z^2 and three second-order filters in Z^2. This means that 8 coefficients are required to meet the specifications that in the nonlinear phase design only required 4 coefficients. Path-0 (the delay-only path) requires 16 units of delay while the path-1 all-pass coefficient list is presented next in decreasing powers of Z which along with its single delay element formed a 17th order denominator.

Path-0	**Polynomial Coefficients**				
Delay	[zeros(1,16) 1]				

Path-1	**Polynomial Coefficients:**				
Delay	[0	1]			
Filter-0	[1	0	0.832280776]		
Filter-1	[1	0	−0.421241137]		
Filter-2	[1	0	0.67623706	0	0.23192313]
Filter-3	[1	0	0.00359228	0	0.19159423]
Filter-4	[1	0	−0.59689082	0	0.18016931]

Figure 10.25 presents the pole-zero diagram of the linear phase all-pass filter structure that meets the same spectral characteristics as those outlined in the previous section. We first note that the filter is nonminimum phase due to the zeros outside the unit circle. We also note the near cancellation of the right half plane pole cluster with the reciprocal zeros of the nonminimum phase zeros.

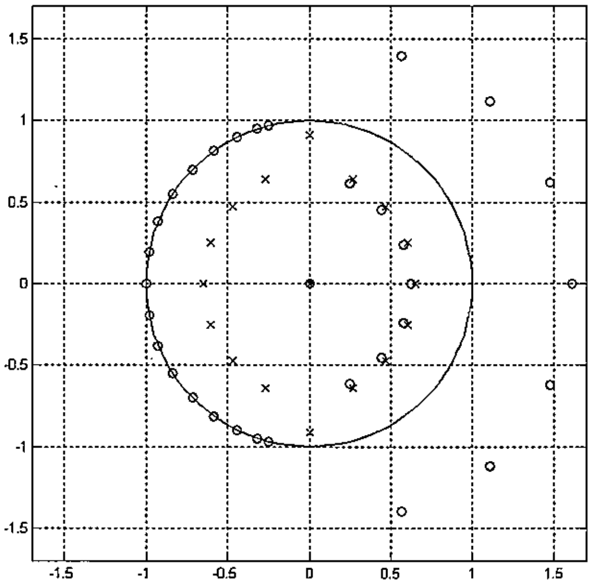


Figure 10.25 Pole-zero Diagram of 2-path, 33 pole, 8-multiplier Filter

Figure 10.26 presents the phase slopes of the two filter paths, the filter frequency response, the filter impulse response, and the filter group delay. We first note that the phase slopes of the paths are linear; consequently the group delay is also linear over most of the filter pass band. Due to the nonminimum phase zeros of this filter the group delay is greater than for the previous filter. The constant group delay matches the 16-sample time delay to the peak of the impulse response. The group delay increases at the filter band edge due to the dominant pole on the imaginary axis near the unit circle. Of course the spectral zeros of the frequency response coincide with the transfer function zeros on the unit circle.

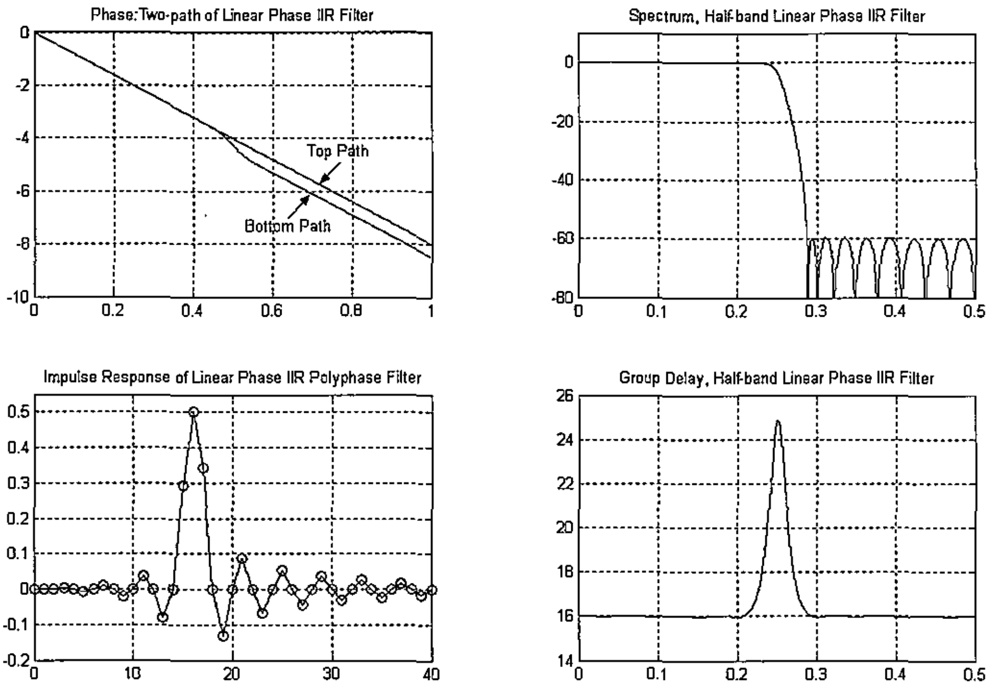


Figure 10.26 Two-path Phase Slopes, Frequency, Impulse, and Phase Responses of 2-path, 33-Pole, 8-multiplier Filter

10.3 Comparison of Nonuniform and Equal-ripple Phase Two-path Filters

We have discussed two forms of the 2-path recursive all-pass filter. In the case of nonuniform phase, the two paths contain one or more all-pass filters formed by first order polynomials in Z^2, while in the case of linear phase, one path is limited to pure delay and the other contains one or more all-pass filters formed by first- and second-order polynomials in

Z^2. In the following discussion we distinguish roots at the origin from roots not at the origin. We say a root at the origin is inactive since it only affects phase and that a root off the origin is active since it affects both phase and magnitude. In the nonuniform phase filter, each coefficient contributes a second-order polynomial to the filter. The second path of this filter also contains an extra delay that contributes a pole at the origin. The total number of poles in the composite filter is 2n + 1, where n is the number of coefficients. The number of zeros in the filter equals the number of poles with all the zeros active, and, in fact, on the unit circle. In the uniform-phase filter, each coefficient in the second path contributes a second-order polynomial in Z^2 that forms two active poles and a pair of inactive poles due to the matching delays in the first path. There is one more delay in the second path than in the first. Thus the total number of poles in this composite filter is 4n + 1, where n is the number of coefficients. The delays in the first path are responsible for a large number of active zeros as the phases of the two paths interact by destructive cancellation.

A final comparison of the efficiency of the two forms of 2-path recursive filters is to be found in Figures 10.27 and 10.28. Both figures present sets of curves showing the trade between out-of-band attenuation and transition bandwidth for 2-path half-band filters for a range of filter complexity. The family of curves is indexed both by number of coefficients, m, and the degree, n, of the composite filter.

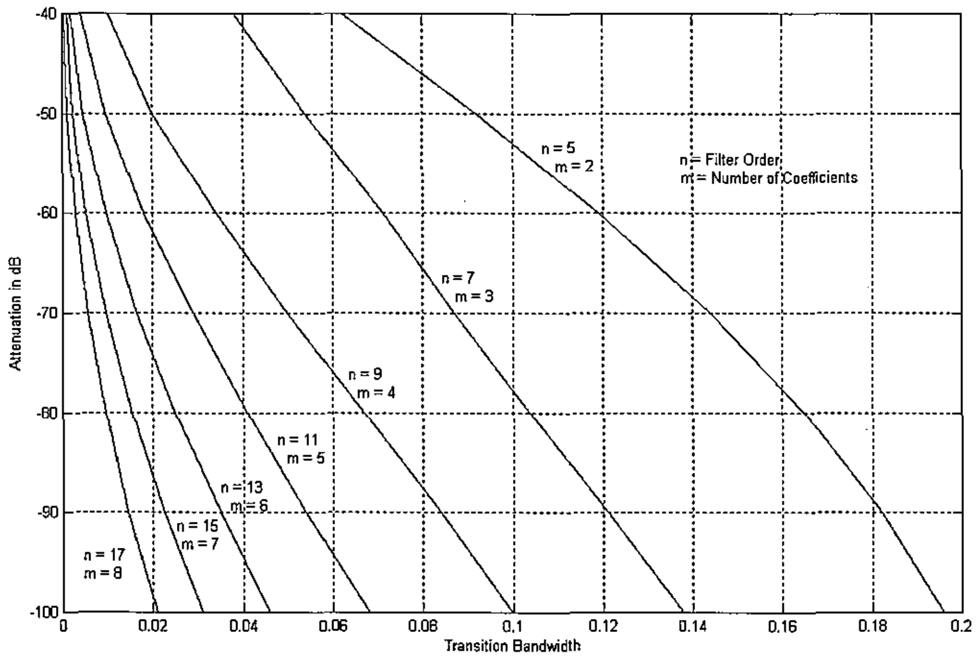


Figure 10.27 Variation of Out-of-band Attenuation versus Transition Bandwidth for Nonuniform Phase 2-path Filters Containing 2 to 8 Coefficients

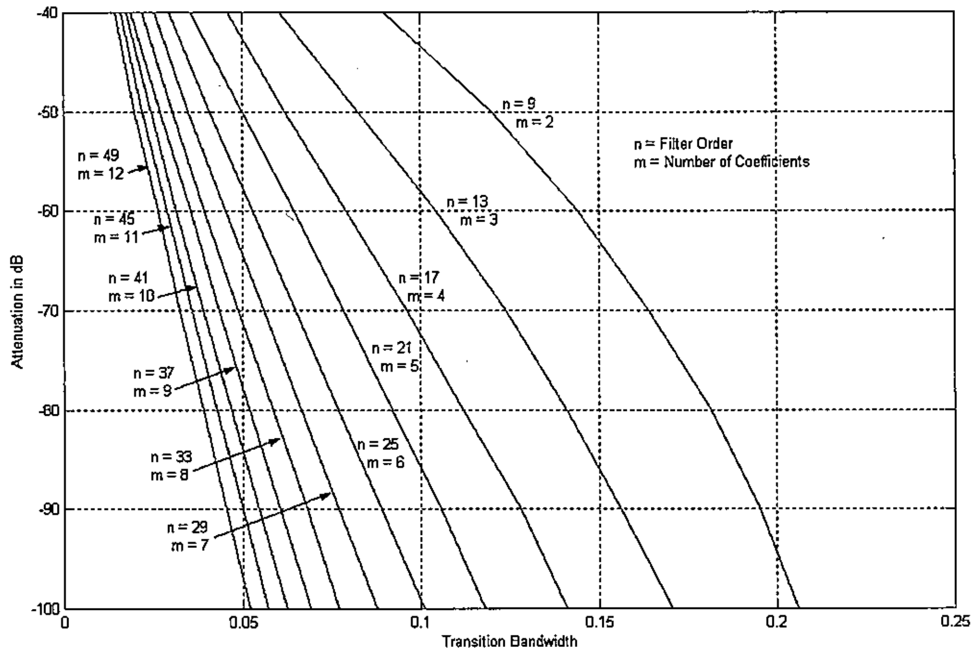


Figure 10.28 Variation of Out-of-band Attenuation versus Transition Bandwidth for Uniform Phase 2-path Filters Containing 2 to 12 Coefficients

As a comparison, suppose we require a half-band filter with transition bandwidth of 3.0% and with 60-dB out-of-band attenuation. We find from Figure 10.27 that we can meet these requirements with a 9th degree, 4-coefficient non-uniform phase filter and from Figure 10.28 that we can meet these requirements with a 33rd degree, 8-coefficient linear-phase filter. To complete this section we include for comparison the pole-zero plot and the frequency response of a linear-phase 65-tap FIR filter that meets the same specifications as the linear-phase IIR filter.

The pole-zero diagram of the FIR filter is presented in Figure 10.29 and the impulse response and frequency response of the same filter is presented in Figure 10.30. Figure 10.31 presents for comparison the impulse response of the linear-phase IIR and FIR filters. Note the shorter delay to the main lobe response of the IIR filter. Figure 10.32 presents the frequency response of the linear phase IIR and FIR filters. As can be seen, these filters have the same pass band and stop band edges as well as the same out-of-band attenuation level. Finally, Figure 10.33 presents a comparison of the in-band magnitude ripple of the two filters as well as the group-delay of the equal ripple approximation to linear-phase IIR filter. We expect that the group delay response of the two filters differs in that the FIR group delay is constant and the IIR is approximately equal ripple. We now observe that the amplitude response of the two filters differs in a similar fashion. The FIR amplitude is equal ripple while the IIR appears to be constant. In fact, the IIR ripple response is merely very small, for this example, on the order of 5-μ-dB. Note that the FIR filter requires 65-multiplies per output (32 if we take advantage of symmetry), while the IIR filter requires 8-multiplies. Now that we have highlighted the relative performance of the linear phase IIR and FIR filters, we leave the FIR filter and return our attention to the primary topic, all-pass recursive filters.

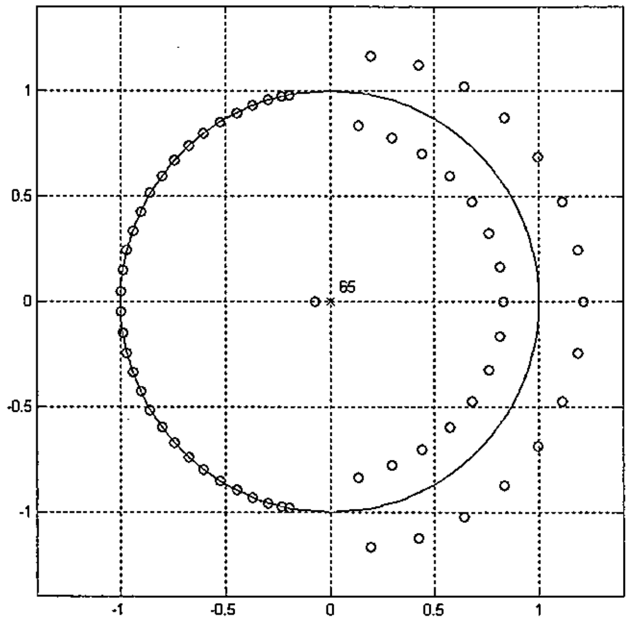


Figure 10.29 Pole-zero Plot of 65-tap FIR Filter

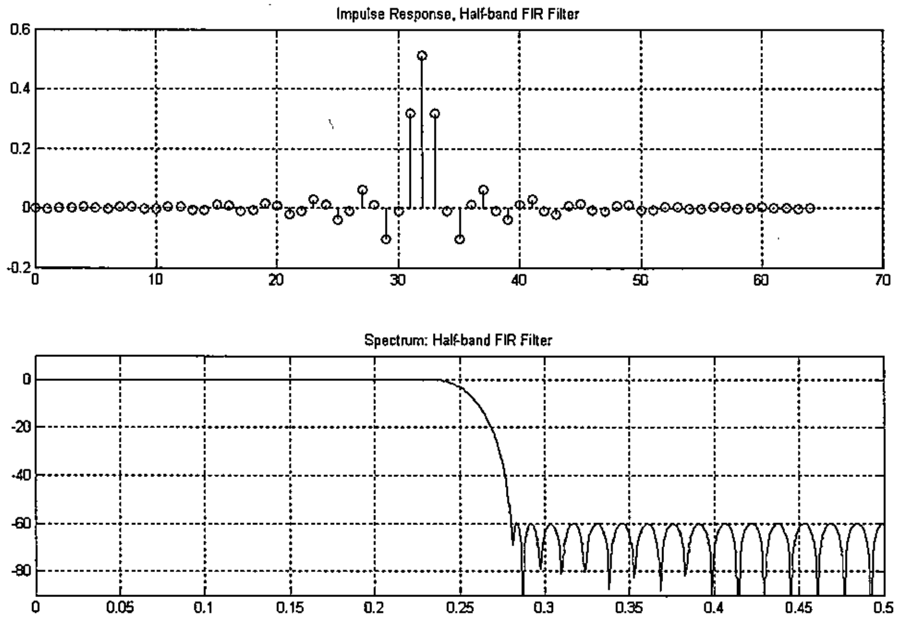


Figure 10.30 Impulse Response and Frequency Response of 65-tap FIR Filter

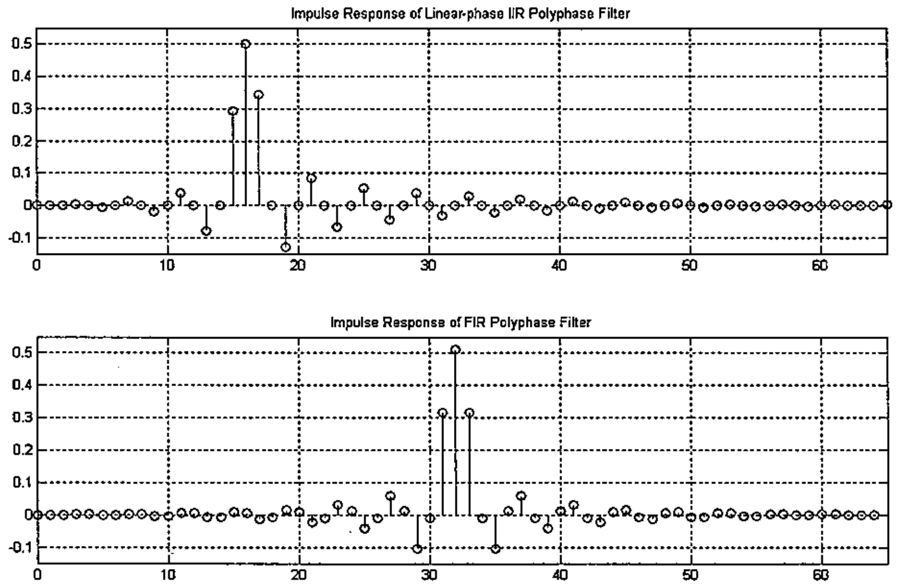


Figure 10.31 Comparison of Impulse Responses of Linear-phase IIR and FIR Filters

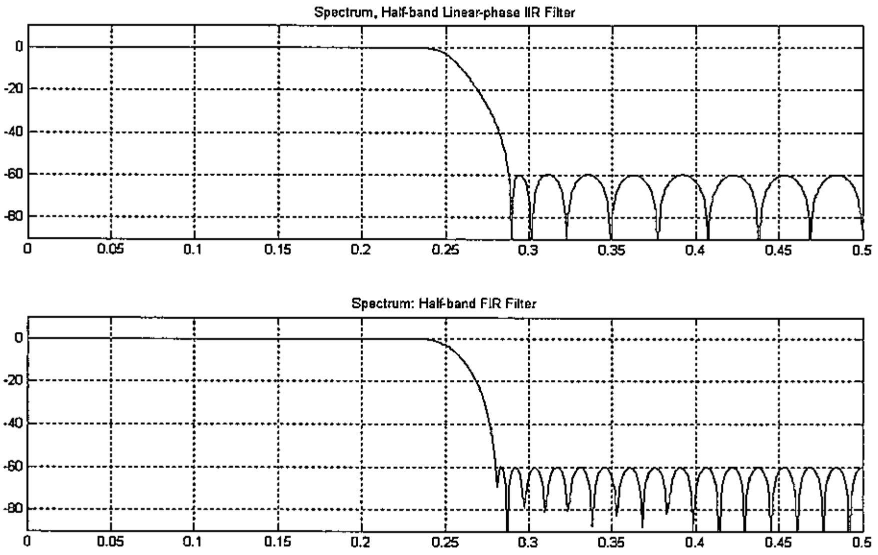


Figure 10.32 Comparison of Frequency Responses of Linear-phase IIR and FIR Filters

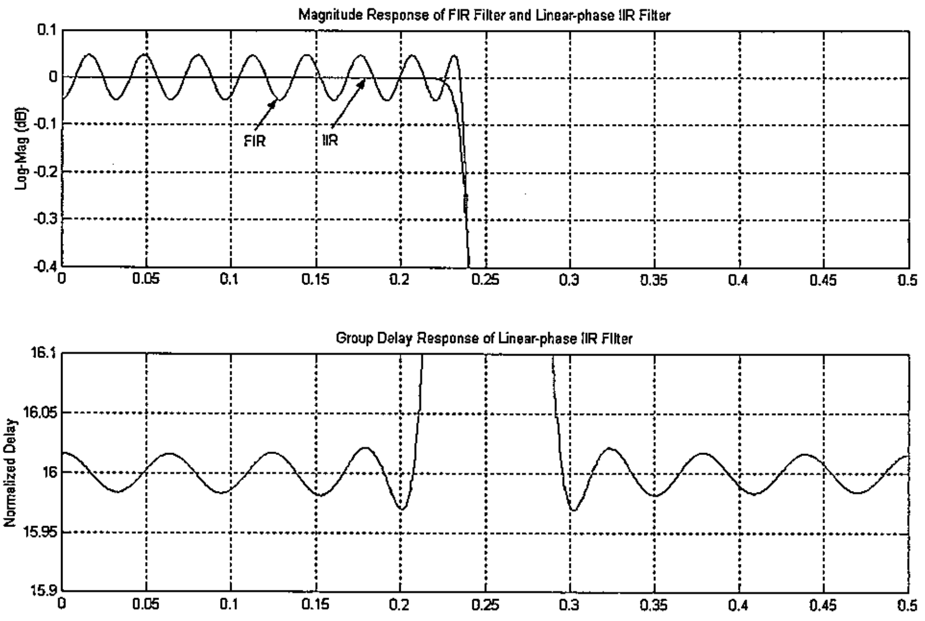


Figure 10.33 Detail Comparison of In-band Magnitude Ripple of Linear-Phase IIR and FIR and of Group Delay of Equal Ripple Linear-phase IIR Filter

10.4 Pass band and Stop band Response in Half-band Filters

The all-pass networks that formed the half-band filter exhibit unity gain at all frequencies. These are lossless filters affecting only the phase response of signals they process. This leads to an interesting relationship between the pass band and stop band response of the half-band filter and, in fact, for any of the 2-path filters discussed in this chapter. Examining Figure 10.17, we note that we have access to complementary filters, the low pass and the high pass versions of the half-band filter, the frequency responses of which are shown in Figure 10.34, where pass band and stop band ripples have been denoted by δ1 and δ2 respectively.

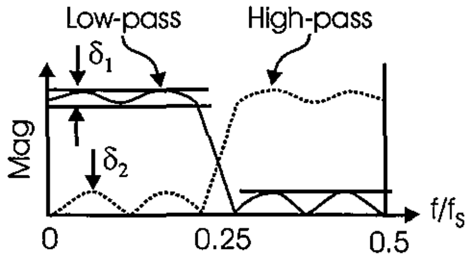


Figure 10.34 Magnitude Response of Low pass and High pass Half-band Filter

The transfer functions of the low pass and of the high pass filters are shown in (10.21), where $P_0(Z)$ and $P_1(Z)$ are the transfer functions of the all-pass filters in each of the two paths. The power gain of the low pass and high pass filters is shown in (10.22). When we form the sum of the power gains, the cross terms in the pair of products cancel, and we obtain the results shown in (10.23).

$$\begin{aligned} H_{LOW}(Z) &= 0.5\cdot[P_0(Z)+Z^{-1}P_1(Z)] \\ H_{HIGH}(Z) &= 0.5\cdot[P_0(Z)-Z^{-1}P_1(Z)] \end{aligned} \tag{10.21}$$

$$\begin{aligned} |H_{LOW}(Z)|^2 &= H_{LOW}(Z)\,H_{LOW}(Z^{-1}) \\ &= 0.25\cdot[P_0(Z)+Z^{-1}P_1(Z)]\cdot[P_0(Z^{-1})+Z\,P_1(Z^{-1})] \\ |H_{HIGH}(Z)|^2 &= H_{HIGH}(Z)\,H_{HIGH}(Z^{-1}) \\ &= 0.25\cdot[P_0(Z)-Z^{-1}P_1(Z)]\cdot[P_0(Z^{-1})-Z\,P_1(Z^{-1})] \end{aligned} \tag{10.22}$$

$$\begin{aligned}|H_{LOW}(Z)|^2 + |H_{HIGH}(Z)|^2 &= 0.25\ [2\,|P_0(Z)|^2 + 2|P_1(Z)|^2] \\ &= 1\end{aligned} \tag{10.23}$$

Equation (10.23) tells us that at any frequency the magnitude square of the low pass gain and the magnitude square of the high pass gain is equal to unity. This is a consequence of the filters being lossless. Energy that enters the filter is never dissipated, a fraction of it is available at the low pass output, and the rest of it is available at the high pass output. This property is the reason the complementary low pass and high pass filters cross at their 3-dB points. If we substitute the gains at peak ripple of the low pass and high pass filters into (10.23), we obtain (10.24), which we can rearrange and solve for the relationship between $\delta_1$and δ_2. The result is interesting. We learn here that the in-band ripple is approximately half the square of the out-of-band ripple. Thus if the out-of-band ripple is –60-dB or one-part in a 1,000, then the in-band ripple is half of one-part in 1,000,000, which is on the order of 5 μ-dB (4.34 μ-dB). The half-band recursive all-pass filters exhibit a very small in-band ripple.

$$\begin{aligned}&[1-\delta_1]^2 + [\delta_2]^2 = 1 \\ &[1-\delta_1] = \sqrt{1-\delta_2^2} \cong 1-0.5\cdot\delta_2^2 \\ &\delta_1 \cong 0.5\cdot\delta_2^2\end{aligned} \tag{10.24}$$

10.5 Transforming Half-band to Arbitrary Band-width

In the previous section we examined the design of 2-path half-band filters formed from recursive all-pass first-order filters in the variable Z^2. We did this because we have easy access to the weights of this simple constrained filter, the constraint being stated in (10.24). If we include a requirement that the stop band be equal ripple, the half-band filters we examine are elliptic filters that can be designed from standard design routines. The filter so designed is our prototype that will be transformed to form other filters with specified (arbitrary) bandwidth and center frequency. In this section, elementary frequency transformations performed with our all-pass filters are introduced, and their impact on the prototype architecture as well as on the system response is reviewed. In particular, frequency transformations that permit bandwidth tuning of the prototype are introduced first. Additional transformations that permit tuning of the center frequency of the prototype filter are also discussed.

10.5.1 Low pass to Low-pass Transformation

We now address the first transformation to perform a transformation from the low pass half-band filter to a low pass arbitrary-bandwidth filter. Frequency transformations occur when an existing all-pass sub network in a filter is replaced by another all-pass sub network. In particular, we now examine the transformation shown in (10.25).

$$\frac{1}{Z} \Rightarrow \frac{1+bZ}{Z+b}; \; b = \frac{1-\tan(\theta_b/2)}{1+\tan(\theta_b/2)}; \; \theta_b = 2\pi\frac{f_b}{f_S} \tag{10.25}$$

This is the generalized delay element we introduced in the initial discussion of first-order all-pass networks. In many applications, we can physically replace each delay in the prototype filter with the all-pass network and then tune the prototype by adjusting the parameter b. We have fielded many designs in which we perform this substitution. Some of these designs are cited in the chapter references. For the purpose of this paper we perform the substitution algebraically in the all-pass filters comprising the 2-path half-band filter and in doing so generate a second structure for which we will develop and present an appropriate architecture.

We substitute (10.25) into the all-pass filter formed by first order polynomials in Z^2 that was introduced in (10.18) and rewritten here in (10.26).

$$\begin{aligned} G(Z) &= H(Z^2)\Big|_{Z \Rightarrow \frac{Z+b}{1+bZ}} \\ G(Z) &= \left.\frac{1+\alpha Z^2}{Z^2+\alpha}\right|_{Z \Rightarrow \frac{Z+b}{1+bZ}} \end{aligned} \tag{10.26}$$

After performing the indicated substitution and gathering terms, we find the form of the transformed transfer function is as shown in (10.27).

$$G(Z) = \frac{1 + c_1 Z + c_2 Z^2}{Z^2 + c_1 Z + c_2} \quad \text{with } c_1 = \frac{2b(1+\alpha)}{1+\alpha b^2} \quad c_2 = \frac{b^2+\alpha}{1+\alpha b^2} \tag{10.27}$$

As expected, when b $\Rightarrow$ 0, $c_1 \Rightarrow 0$, and c2 $\Rightarrow \alpha$, the transformed all-pass filter reverts back to the original first-order filter in Z^2. The architecture of the transformed filter, which permits one multiplier to form the matching numerator and denominator coefficient simultaneously, is shown in Figure 10.35. Also shown is a processing block G(Z) that uses two coefficients c_1 and c_2. This is seen to be an extension of the 1-multiply structure presented in Figure 10.12. The primary difference in the two architectures is the presence of the coefficient and multiplier associated with the power of Z^{-1}. This term, formerly zero, is the sum of the polynomial roots, and hence is minus twice the real part of the roots. With this coefficient being non-zero, the roots of the polynomial are no longer restricted to the imaginary axis.

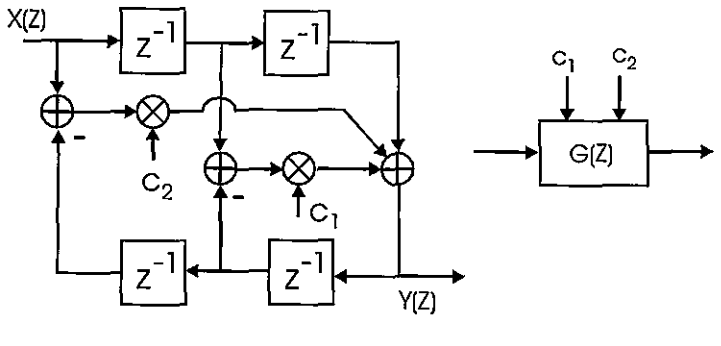


Figure 10.35 Block Diagram of General Second-order All-pass Filter

The root locations of the transformed, or generalized, second-order all-pass filter are arbitrary except that they appear as conjugates inside the unit circle, and the poles and zeros appear in reciprocal sets as indicated in Figure 10.36.

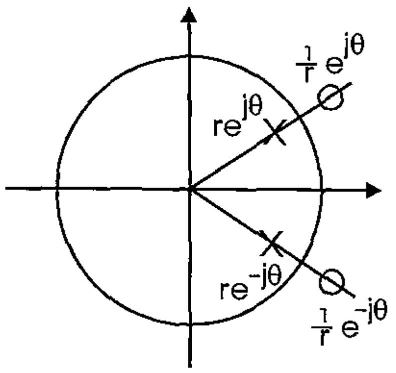


Figure 10.36 Pole-zero Diagram of Generalized Second-order All-pass Filter

The two-path prototype filter contained one or more 1-multiply first-order recursive filters in Z^2 and a single delay. We effect a frequency transformation on the prototype filter by applying the low pass to low pass transformation shown in (10.25). Doing so converts the 1-multiply, first-order in Z^2, all-pass filter to the generalized two-multiply second-order all-pass filter, and converts the delay, a zero-multiply all-pass filter to the generalized one-multiply first-order in Z all-pass filter. Figure 10.37 shows how applying the frequency transformation affects the structure of the prototype. Note that the 5-pole, 5-zero half-band filter, which is implemented with only two multipliers, now requires five multipliers to form the same 5-poles and 5-zeros for the arbitrary-bandwidth version of the 2-path network. This is still significantly less than the standard cascade of first- and second-order canonic filters for which the same 5-pole, 5-zero filter would require 13 multipliers.

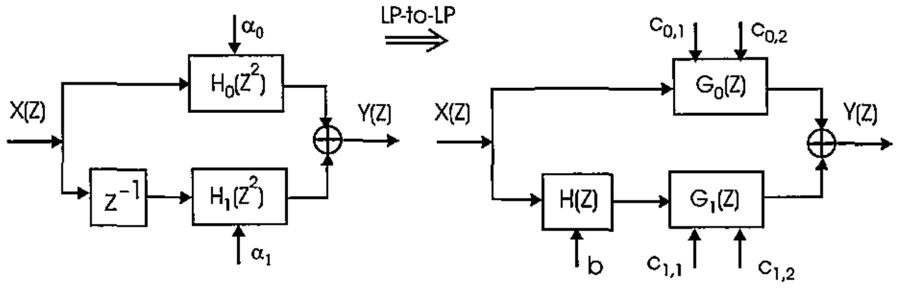


Figure 10.37 Effect on Architecture of Frequency Transformation Applied to 2-path Half-band All-pass Filter

Figure 10.38 presents the phase response of the two paths and the frequency response obtained by applying the low pass to low pass frequency transformation to the prototype 2-path, 2-multiply, half-band filter presented in Figure 10.21. The bandedge was moved from normalized frequency 0.25 to normalized frequency 0.1. Figure 10.39 presents the pole-zero diagram of the frequency transformed prototype filter. The 5-poles have been pulled off the imaginary axis, and the 5-zeros migrated around the unit circle to form the reduced-bandwidth version of the prototype.

Figure 10.40 presents the two path phase responses and the frequency response obtained by applying the low pass to low pass frequency transformation to move the bandedge from the normalized frequency 0.25 to normalized frequency 0.02. Figure 10.41 presents the pole-zero diagram of this frequency transformed prototype filter.

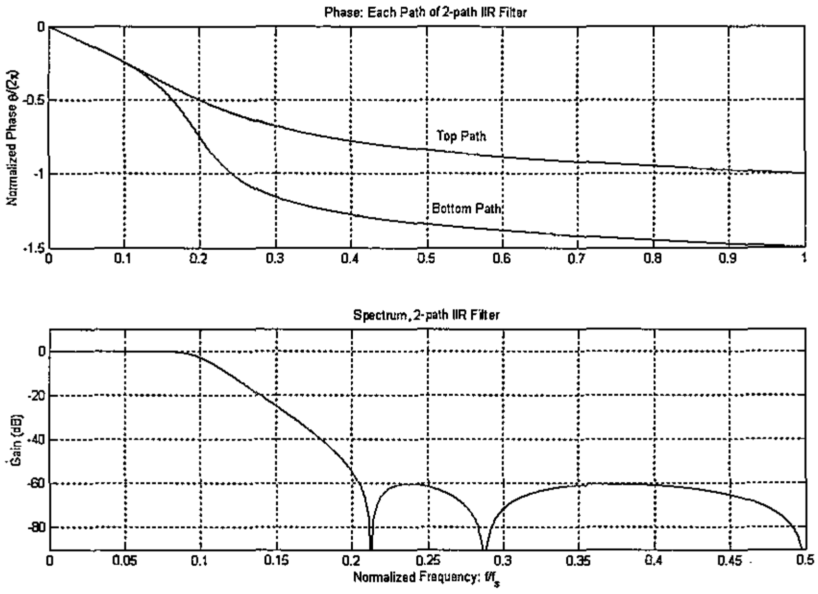


Figure 10.38 Two-path Phase Responses and Frequency Response Obtained by Frequency Transforming Half-band Filter to Normalized Frequency 0.1

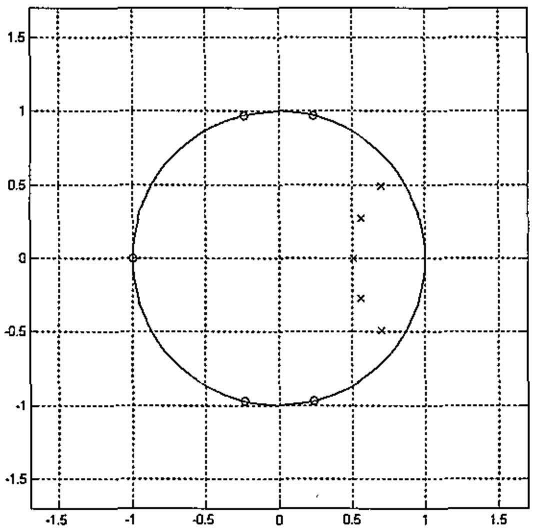


Figure 10.39 Pole-zero Diagram Obtained by Frequency Transforming Half-band Filter to Normalized Frequency 0.1

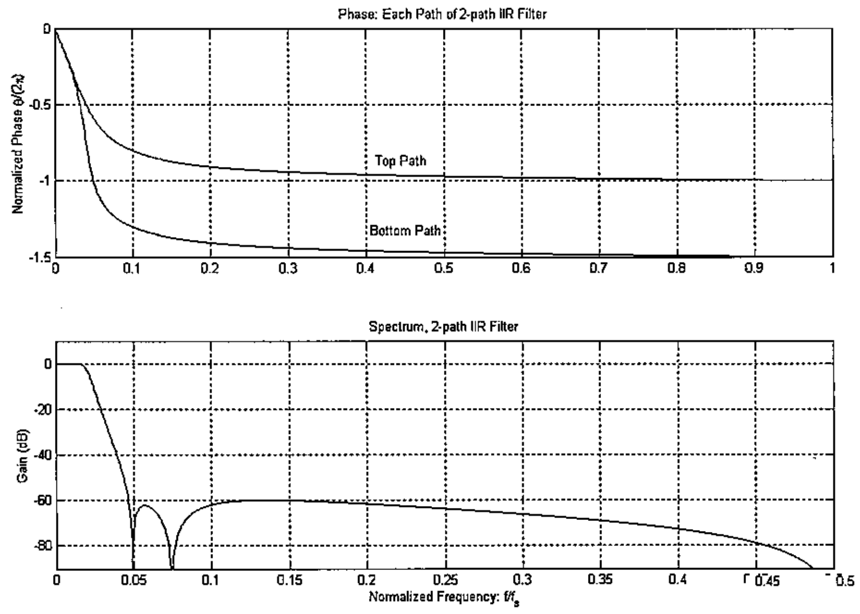


Figure 10.40 Two Path Phase Responses and Frequency Response Obtained by Frequency Transforming Half-band Filter to Normalized Frequency 0.02

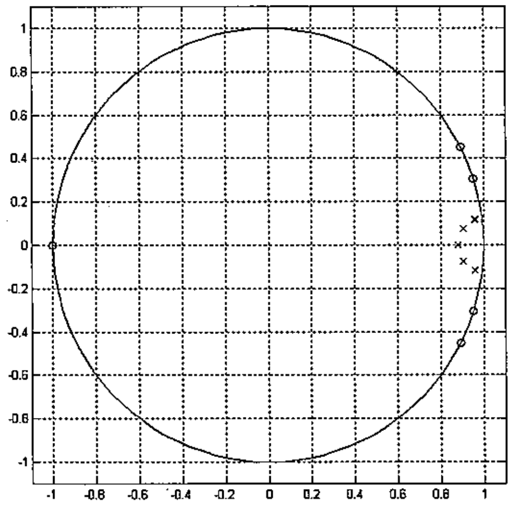


Figure 10.41 Pole-zero Diagram Obtained by Frequency Transforming Half-band Filter to Normalized Frequency 0.02

10.5.2 Low pass to Band-pass Transformation

In the previous section we examined the design of 2-path, arbitrary-bandwidth low pass filters formed from recursive all-pass first- and second-order filters as shown in Figure 10.36. We formed this filter by a transformation of a prototype half-band filter. We now address the second transformation, one that performs the low pass to band-pass transformation. As in the previous section we can invoke a frequency transformation wherein an existing all-pass sub network in a filter is replaced by another all-pass sub network. In particular, we now examine the transformation shown in (10.28).

$$\frac{1}{Z} \Rightarrow -\frac{1}{Z}\frac{1-cZ}{Z-c}; \quad c=\cos(\theta_C); \quad \theta_C = 2\pi\frac{f_C}{f_S} \tag{10.28}$$

This, except for the sign, is a cascade of a delay element with the generalized delay element we introduced in the initial discussion of first-order all-pass networks. We can physically replace each delay in the prototype filter with this all-pass network and then tune the center frequency of the low pass prototype by adjusting the parameter c. For the purpose of this book we perform the substitution algebraically in the all-pass filters comprising the 2-path frequency transformed arbitrary-bandwidth filter, and in doing so generate yet a third structure for which we will develop and present an appropriate architecture.

We substitute (10.28) into the second-order all-pass filter derived in (10.27) and rewritten in (10.29).

$$\begin{aligned} F(Z) &= G(Z)\Big|_{Z \Rightarrow \frac{Z(Z-c)}{(cZ-1)}} \\ &= \frac{(b^2+\alpha)+2b\,(1+\alpha)Z+(1+\alpha b^2)Z^2}{(1+\alpha b^2)+2b\,(1+\alpha)Z+(b^2+\alpha)Z^2}\Bigg|_{Z \Rightarrow \frac{Z(Z-c)}{(cZ-1)}} \end{aligned} \tag{10.29}$$

After performing the indicated substitution and gathering up terms, we find the form of the transformed transfer function is as shown in (10.30).

$$\begin{aligned} F(Z) &= \frac{1+d_1Z+d_2Z^2+d_3Z^3+d_4Z^4}{Z^4+d_1Z^3+d_2Z^2+d_3Z+d_4} \\ d_1 &= \frac{-2c(1+b)(1+\alpha b)}{1+\alpha b^2} \quad d_2 = \frac{(1+\alpha)(c^2(1+b)^2+2b)}{1+\alpha b^2} \\ d_3 &= \frac{-2c(1+b)(1+\alpha b)}{1+\alpha b^2} \quad d_4 = \frac{\alpha+b^2}{1+\alpha b^2} \end{aligned} \tag{10.30}$$

As expected, when we let $c \Rightarrow 0$, $d_1$and $d_2 \Rightarrow 0$, while $d_2 \Rightarrow c_1$ and $d_4 \Rightarrow c_2$ the weights default to those of the prototype, arbitrary-bandwidth, filter. The transformation from low pass to band-pass generates two spectral copies of the original spectrum, one each at the positive- and negative- tuned center frequency. The architecture of the transformed filter, which permits one multiplier to simultaneously form the matching numerator and denominator coefficients, is shown in Figure 10.42. Also shown is a processing block F(Z) which uses the four coefficients d_1, d_2, d_3, and d_4. This is seen to be an extension of the 2-multiply structure presented in Figure 10.35.

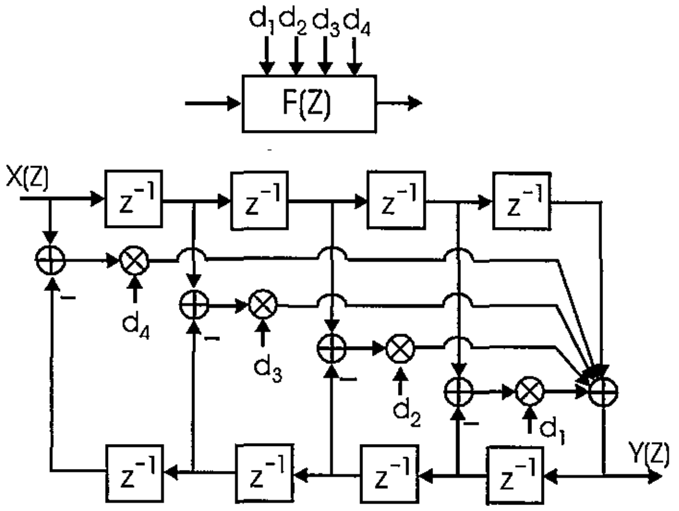


Figure 10.42 Block Diagram of General Fourth-order All-pass Filter

We have just described the low pass to band-pass transformation that is applied to the second-order all-pass networks of the 2-path filter. One additional transformation that requires attention is the low pass to band-pass transformation that must be applied to the generalized delay or bandwidth-transformed delay from the prototype half-band filter. We substitute (10.28) into the first-order all-pass filter derived in (10.25) and rewritten in (10.31).

$$\begin{aligned} E(Z) &= \left.\frac{1+bZ}{Z+b}\right|_{Z\Rightarrow\frac{Z(Z-c)}{(cZ-1)}} \\ &= \frac{(cZ-1)+bZ(Z-c)}{Z(Z-c)+b(cZ-1)} = \frac{-1+c(1-b)Z+bZ^2}{Z^2-c(1-b)Z-b} \end{aligned} \tag{10.31}$$

As expected, when c ⇒ 1 the denominator goes to (Z + b)(Z – 1) while the numerator goes to (1 + bZ)(Z – 1) so that the transformed all-pass filter reverts to the original first-order filter. The distributed minus sign in the numerator modifies the architecture of the transformed second-order filter by shuffling signs in Figure 10.35 to form the filter shown in Figure 10.43. Also shown is a processing block E(Z), which uses two coefficients, e_1 and e_2.

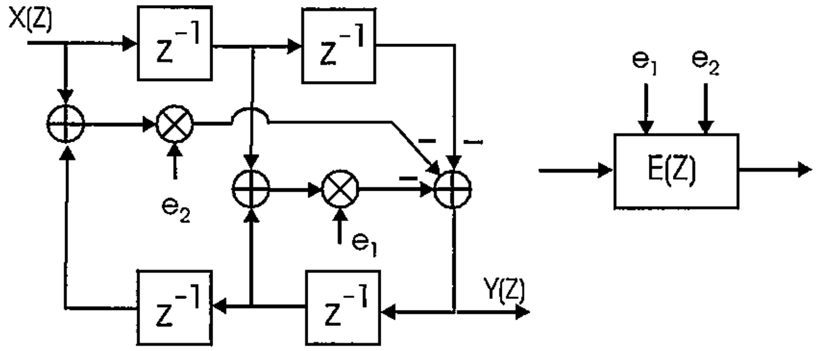


Figure 10.43 Block Diagram of Low pass to Band-pass Transformation Applied to Low pass to Low pass Transformed Delay Element

In the process of transforming the low pass filter to a band-pass filter we convert the 2-multiply second-order all-pass filter to a 4-multiply fourth-order all-pass filter, and convert the 1-multiply low pass to low pass filter to a 2-multiply all-pass filter. The doubling of the number of multiplies is the consequence of replicating the spectral response at two spectral centers of the real band-pass system. Note that the 5-pole, 5-zero arbitrary low pass filter now requires 10 multipliers to form the 10-poles and 10-zeros for the band-pass version of the 2-path network. This is still significantly less than the standard cascade of first- and second-order canonic filters for which the same 10-pole, 10-zero filter would require 25 multipliers. Figure 10.44 shows how the structure of the prototype is affected by applying the low pass to band-pass frequency transformation.

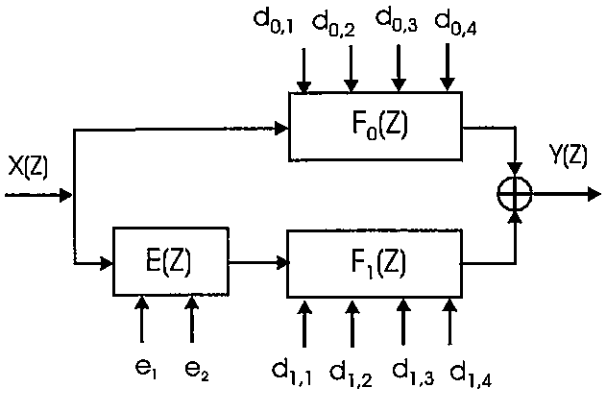

$d_{0,1}$ $d_{0,2}$ $d_{0,3}$ $d_{0,4}$
$F_0(Z)$
X(Z)
Y(Z)
E(Z)
$F_1(Z)$
e_1 e_2
$d_{1,1}$ $d_{1,2}$ $d_{1,3}$ $d_{1,4}$

Figure 10.44 Effect on Architecture of Low pass to Band-pass Frequency Transformation Applied to 2-path Arbitrary Bandwidth All-pass Filter

Figure 10.45 presents the two path phase responses and the frequency response obtained by applying the low pass to band-pass frequency transformation to the prototype 2-path, 4-multiply, low pass filter presented in Figure 10.38. The one-sided bandwidth was originally adjusted to a normalized frequency of 0.1 and is now translated to a center frequency of 0.22. Figure 10.46 presents the pole-zero diagram of the frequency-transformed prototype filter. The 5-poles defining the low pass filter have been pulled to the neighborhood of the band-pass center frequency. The 5-zeros have also replicated, appearing both below and above the pass band frequency.

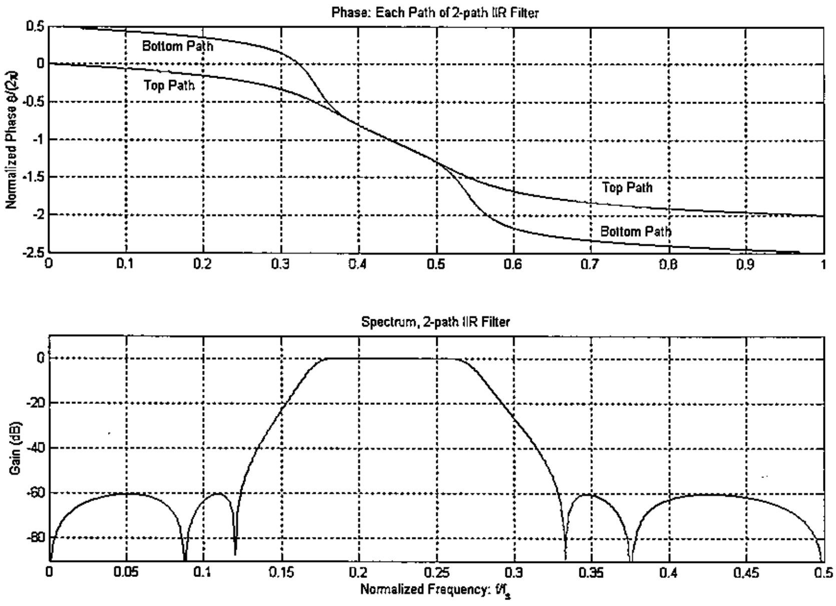


Figure 10.45 Two Path Phase Responses and the Frequency Response of 2-path All-pass Filter Subjected to Low pass to Low pass and then Low pass to Band-pass Transformations

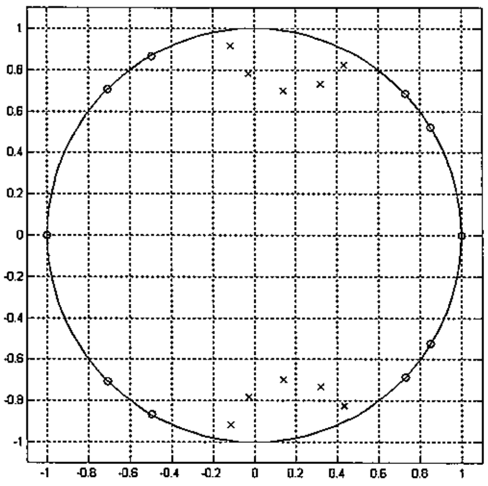


Figure 10.46 Pole-zero Plot of 2-path All-pass Filter Subjected to Low pass to Low pass and then Low pass to Band-pass Transformations

10.6 Multirate Considerations of Recursive Half-band Filters

The half-band filters can be used in a number of multirate applications to perform up sampling and down sampling with imbedded resampling or in a variety of iterated filter applications. Since the half-band filter is formed with polynomials in Z^2, we can invoke the noble identity to slide a 2-to-1 down sampler from the filter output to the filter input or a 1-to-2 up sampler from the filter input to the filter output. This permits us to operate the half-band filter at the lower of the two rates when there is a rate change. The five multiplications required to obtain one data point from a 5-coefficient down sampling half-band filter are distributed over two input samples so that the workload for the half-band filter is 5/2 multiplies per input. Figure 10.47 illustrates the use of the noble identity applied to a 2-to-1 down sampler and Figure 10.48 illustrates the same relationship applied to a 1-to-2 up sampler. In each case the resampler is slid through the all-pass filters, changing them from polynomials in Z^2 to polynomials in Z. The resampler cannot move through the delay. Recognizing the interaction of the delay and two resampling switches, we replace the three elements with a commutator that delivers successive inputs to the 2-path filter for the down sampler or accepts successive outputs from the two paths of the up sampler.

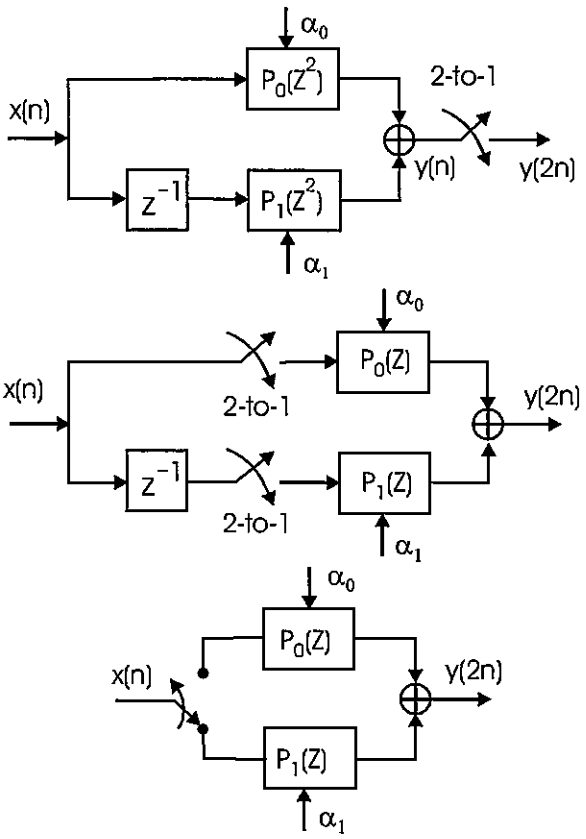


Figure 10.47 Half-band, All-pass Polyphase Filter with 2-to-1 Down Sampler Moved from Output to Input and Replaced with Commutator

The resampling half-band filter can be applied iteratively in a cascade of half-band filters to obtain sample rate changes that are powers of 2. Thus three successive stages of half-band filtering can be cascaded to obtain efficient 8-to-1 down sampling or 1-to-8 up sampling. The efficiency of the half-band cascade suggests filter applications in which we use the cascade to down sample, perform desired filtering at a reduced rate, and then up sample with a second cascade. The following example demonstrates the significant savings due to applications of the half-band, 2-path recursive all-pass filter

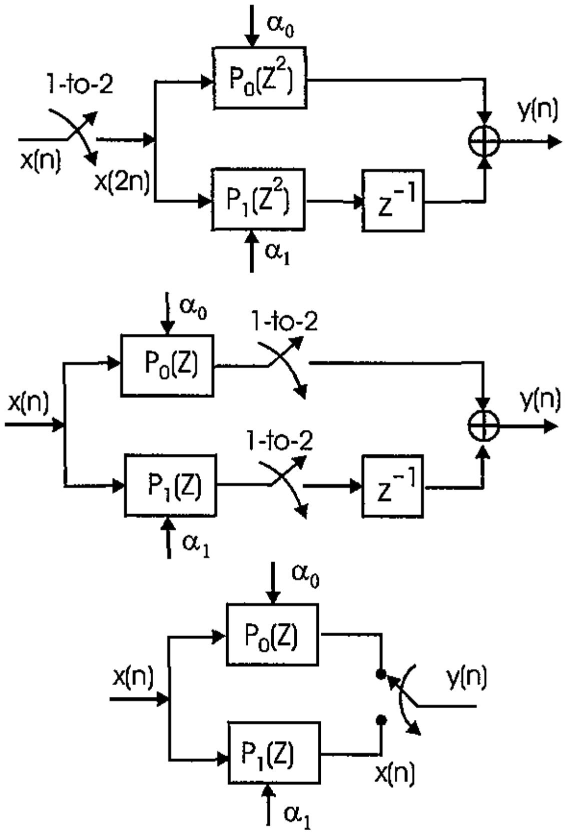


Figure 10.48 Half-band, All-pass Polyphase Filter 1-to-2 Up Sampler Moved from Input to Output and Replaced with Commutator

EXAMPLE 10.1 Applications of Half-band Filters

We examine a low-pass digital filter that meets the following specifications:

Sample Rate:	=	400 kHz
Pass Band:	=	8 kHz
Stop Band:	=	12 kHz
In-band Ripple:	<	0.1-dB
Stop band Attenuation:	>	75-dB

We first try the brute force approach and seek a conventional recursive filter solution. We find from a standard filter design package, such as MATLAB or QED-2000, that an 8th-order elliptic filter meets these specifications. Figure 10.49 presents the impulse response and the frequency response of the elliptic filter.

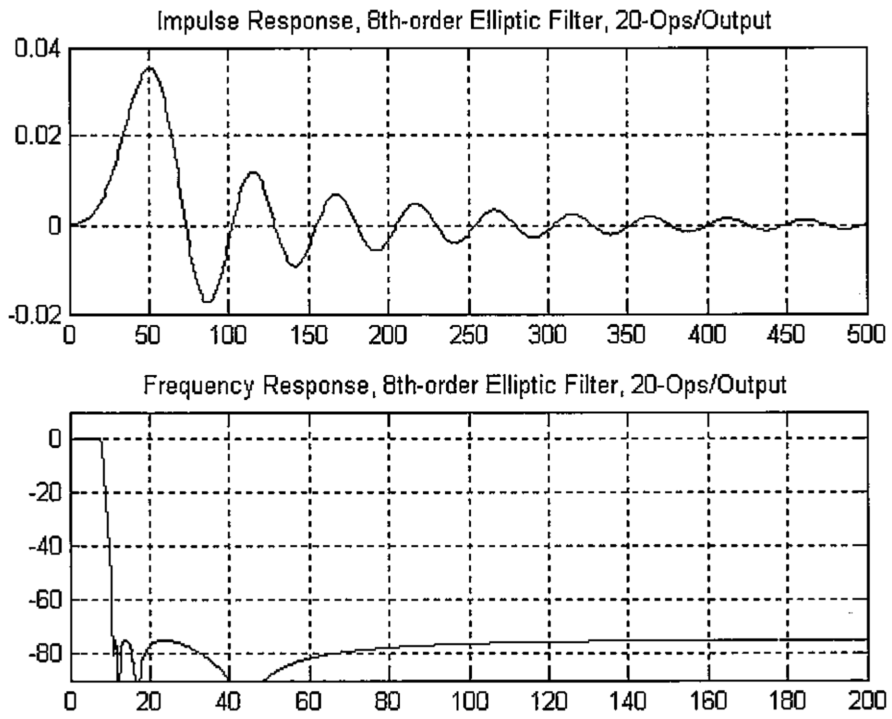


Figure 10.49 Time and Frequency Response of 8th-order Recursive Filter

Figure 10.50 shows a standard implementation of the 8th-order filter as a cascade of four 2nd-order canonic filters sometimes called biquads. Each filter requires 5-multiplies, 2 for the feedback path and 3 for the scaled feed forward path. The 8th-order filter requires four of these segments, hence the filter requires 20-multiplies per input-output data pair. This filter is our reference design against which we compare other designs that offer reduced computation.

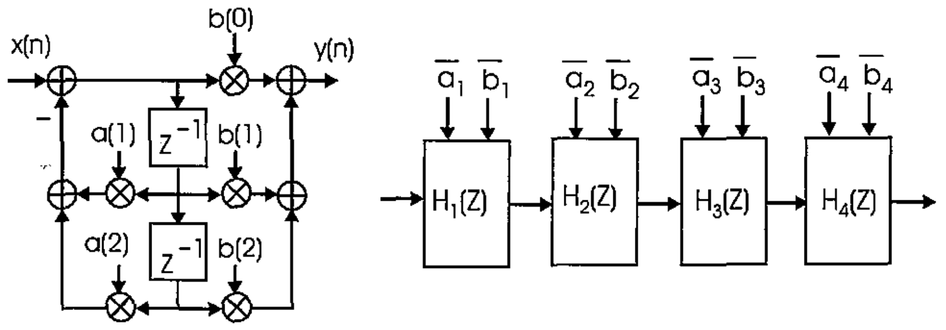


Figure 10.50 Cascade of Four 2nd-order Canonic Filters to Form 8th-order Elliptic Filter

Our first alternate design uses the two-path all-pass recursive filter initially designed as a half-band with the desired stop band attenuation and then transformed to the desired low pass bandwidth by the low pass to low pass transformation discussed in (10.27). The attraction of this option is that the system poles are formed by the filter coefficients and the system zeros are free by virtue of the destructive cancellation that occurs as the two paths are summed. Using the MATLAB design routine *tony_des2* we learn that a 9th-order 2-path filter will meet the design specifications. Note that the 2-path filter must always contain an odd number of roots. The structure of this filter is shown in Figure 10.51 where the segments G(Z) and H(Z) are the forms introduced in Figure 10.37. Note this filter only requires 9-multiplies to form its 9-poles and 9-zeros. This represents a savings of more than a half relative to our cascade biquads. Figure 10.52 presents the time and frequency response of this two-path filter. Note that the extra zero resides at the half-sample rate, a characteristic of the 2-path filter.

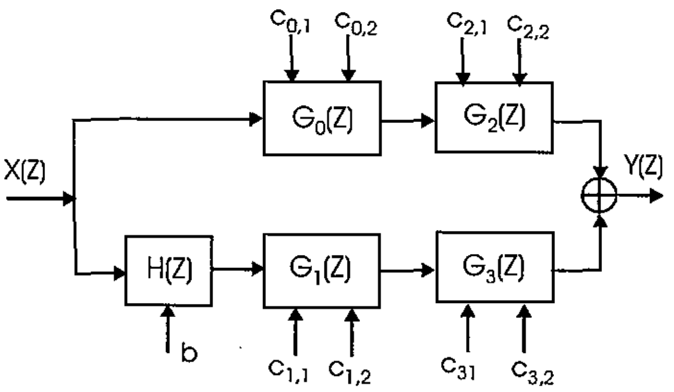


Figure 10.51 Structure of 9th-order 2-path Recursive All-pass Filter

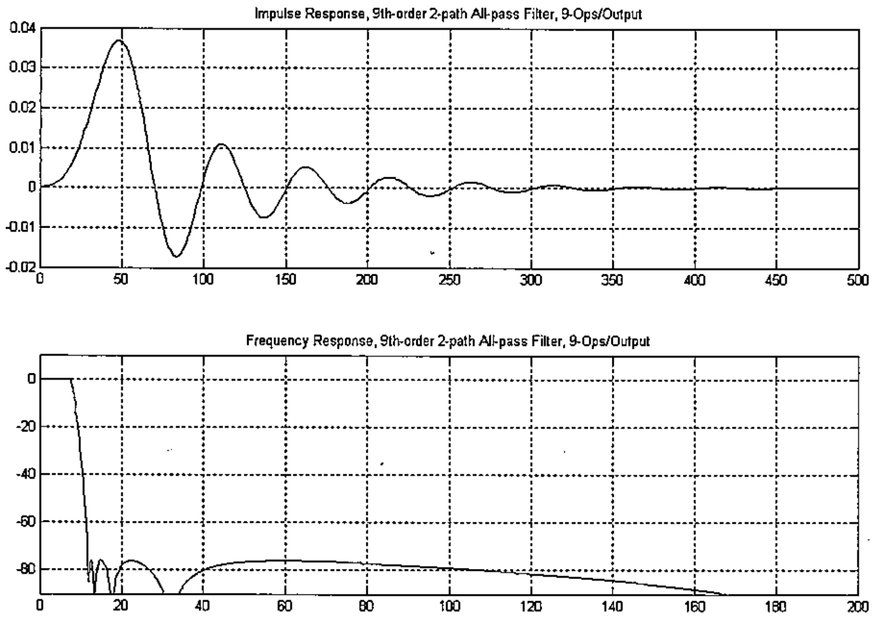


Figure 10.52 Time and Frequency Response of 9th-order 2-path Recursive Filter

The next alternate design takes advantage of the fact that the bandwidth of the desired filter is a small fraction of the sample rate. We implement this filter by first reducing the sample rate with a cascade of half-band filters, build the filter that meets the desired specification at the reduced rate, and then increasing the sample rate with another cascade of half-band filters. The structure of this cascade is shown in Figure 10.53 where we see that the input filter chain contains three resampling half-band filters that reduce the sample rate 8-to-1 from 400 kHz to 50 kHz. The first stage is a 3rd-order 1-multiply filter, the second stage is a 5th-order 2-multiply filter, and the third stage is a 7th-order 3-multiply filter. The low pass filter that performs the bandwidth control is a 9th-order 2-path filter that now operates at 50 kHz rather than the original 400 kHz. The output filter chain contains the same filter set as the input chain but is applied as an up-sampling filter in the opposite order.

The easiest way to determine the workload for this resampling filter is to deliver eight input samples to the chain and determine the workload for the entire chain, then divide by the number of input samples. Starting at the input of the entire cascade we note the first filter will perform 4-multiplies and deliver 4-samples to the next stage. The second stage will also perform 4-multiplies and deliver 2-samples to its next stage. The third stage will perform 3-multiplies and deliver 1-sample to the central filter. This filter performs 9-multiplies and delivers 1-sample to the start of the up-sample chain. The up-sample chain performs the same workload as the down-sample chain for which we conclude that the total workload is (4 + 4 + 3 + 9 + 3 + 4 + 4) or 31-multiplies. These 31-multiplies are amortized over eight input-output sample pairs for a workload of 3.875 or approximately 4-multiplies per input sample. The budget can be visualized as approximately 1.5-multiplies per input point to reduce the sample rate, approximately 1-multiply per sample point to filter the input signal, and approximately 1.5-multiplies per output point to raise the sample rate to the original input rate. Comparing filter workloads we recall that the original filter requires 20, the 2-path requires 9, and the cascade-resampling filter requires 4-multiplies per sample point, which translates into successive savings of about a factor of 2.

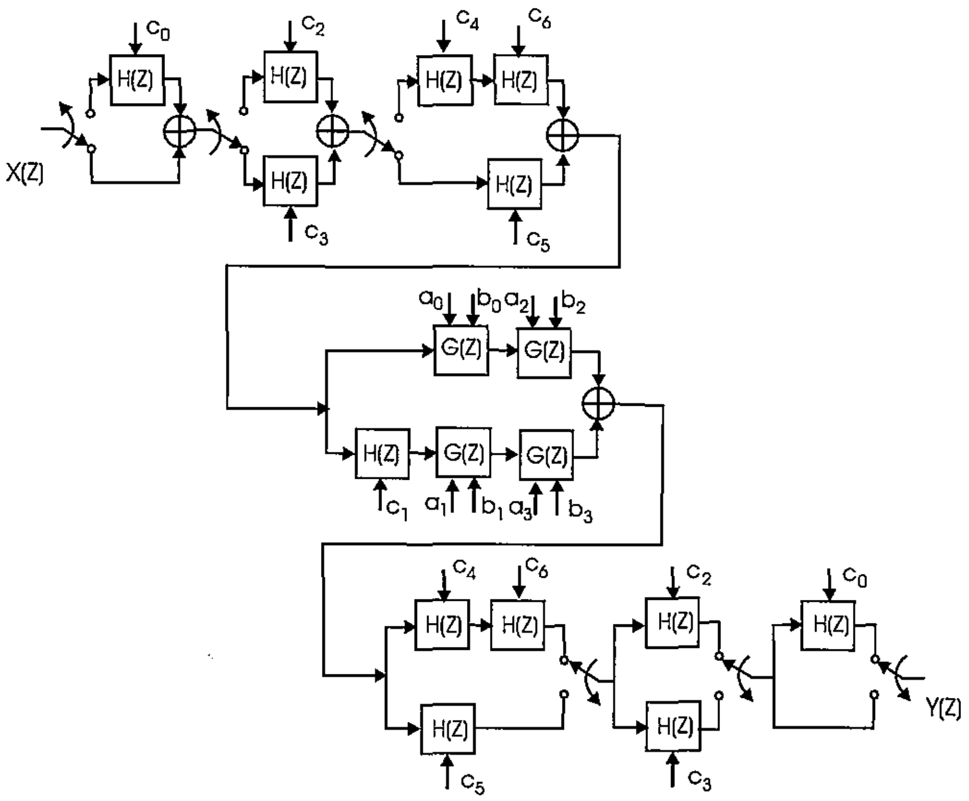


Figure 10.53 Structure of Low-bandwidth Filter as a Cascade of Down-sampling Half-band Filters, the Low pass Filter, and a Cascade of Up-sampling Half-band Filters

Figure 10.54 presents the impulse response and frequency response of the composite resampling chain of filters. The group delay of the chain is seen to be greater than either of the earlier filter options. Figure 10.55 is a set of spectra showing the spectral transformations performed by the filtering chain. This explicitly shows the spectra obtained by zero insertion up sampling followed by filtering to remove the spectral replicate due to the up-sampling operation. The figure presents the frequency response of the filter sequence starting at the low pass central filter and then that of the zero-packed up-sampled version, the half-band filter following the up-sampling, and the result of passing through the half-band filter. The sequence starts in the upper left and proceeds by successive columns to the lower right with the figure at the end of one column becoming the start of the next column. Remember, the process we describe in Figure 10.53 does not separately zero pack and filter, so these spectra are not actually available in the processing chain but are shown to offer insight to the sequence of operations.

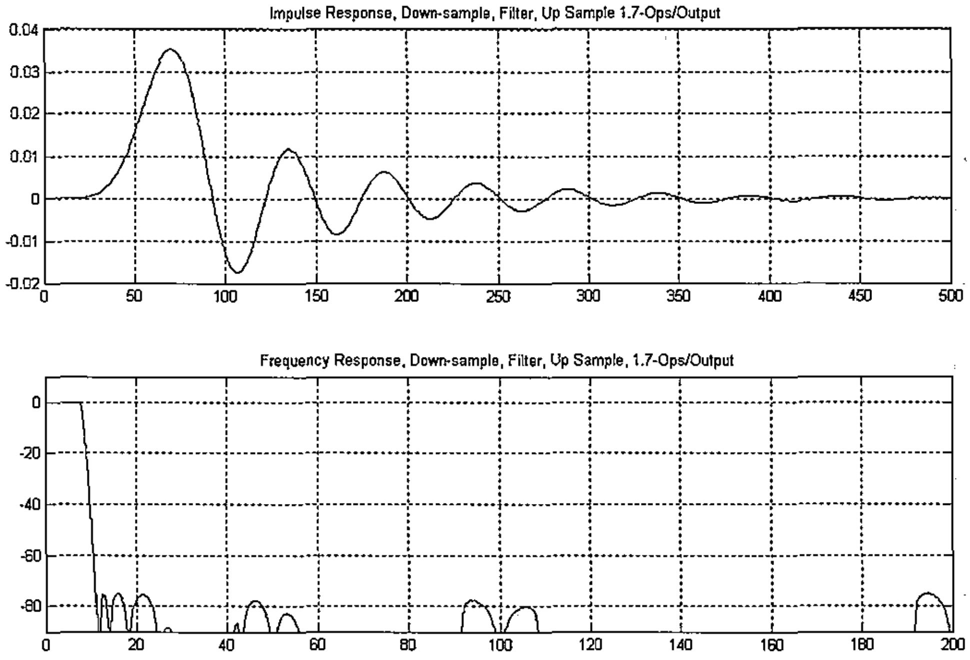


Figure 10.54 Time and Frequency Response of 8-to-1 Down-sampled and 1-to-8 Up-sampled 9th-order 2-path Recursive Filter

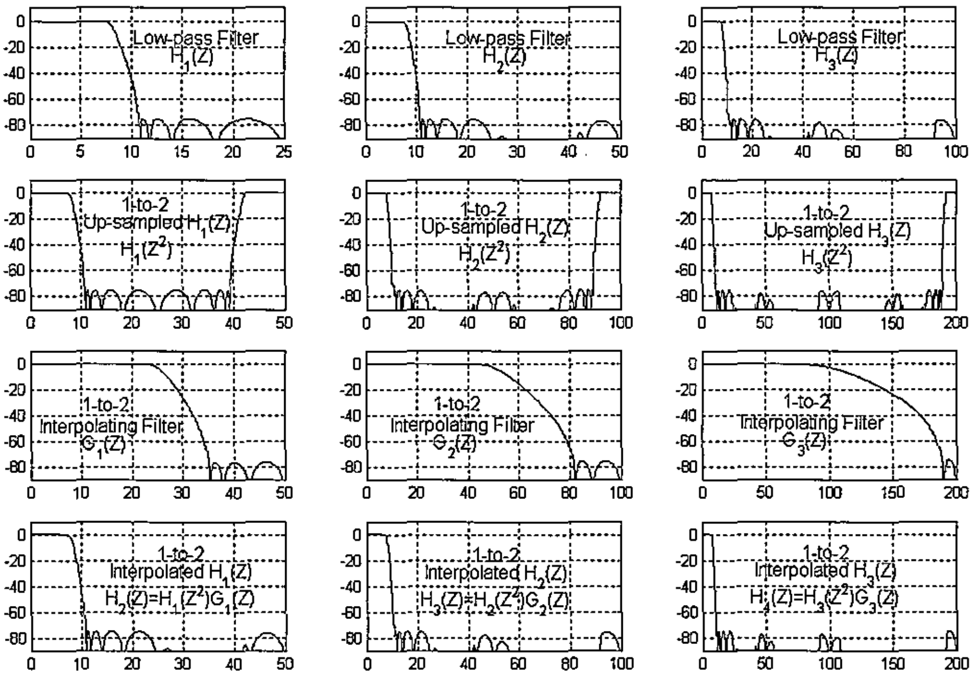


Figure 10.55 Frequency Response for Two of Three Segments of Successive Filtering: Zero-packed Bandwidth Limiting Filter and Interpolating Filter

10.7 Hilbert-transform Filter Variant of Two-path All-pass Filter

A trivial variant of the 2-path all-pass filter leads to a filter with wide application. This is the Hilbert-transform filter that is formed by a trivial modification to the basic 2-path half-band filter. We can translate the half-band filter centered at baseband to another half-band filter centered at the quarter-sample rate. The simple transformation is related to the modulation theorem for Z-transforms. The theorem states that the Z-transform of a heterodyned time series is the Z-transform of the original time series with the variable Z replaced with a phase-rotated Z. This relationship is illustrated in (10.32) through (10.34). In Equations (10.32) and (10.33) we note the Z-transforms of sequences h(n) and of the heterodyned sequence h(n) exp($j\theta_0$n). Here we see that the phase rotation of the time series heterodyne has the same exponent as the position index exponent, and we are free to associate the phase rotator with the delay rather than with the time samples. This is shown in (10.34).

$$\begin{aligned} H(Z) &= \sum_{n=0}^{\infty} h(n) Z^{-n} \\ &= h(0) + h(1)Z^{-1} + h(2)Z^{-2} + \cdots + h(k)Z^{-k} + \cdots \end{aligned} \tag{10.32}$$

$$\begin{aligned} G(Z) &= \sum_{n=0}^{\infty} [h(n)\, e^{j\theta n}] Z^{-n} \\ &= h(0) + h(1)\, e^{j\theta} Z^{-1} + h(2)\, e^{j2\theta} Z^{-2} + \cdots + h(k)\, e^{jk\theta} Z^{-k} + \cdots \end{aligned} \tag{10.33}$$

$$\begin{aligned} G(Z) &= H(Z)\big|_{Z \Rightarrow e^{-j\theta} Z} \\ &= H(e^{-j\theta} Z) \end{aligned} \tag{10.34}$$

If we heterodyne the impulse response of a filter to the quarter-sample rate, the phase rotator is exp(j $\pi/2$), which we can write more compactly as the operator j. Thus we can replace each Z^{-1} in a Z-transform with jZ^{-1} to reflect the heterodyne of the time series. We now apply this transformation to the transfer function of the half-band filter. Since the filter is formed by polynomials in Z^{-2}, the substitution is equivalent to replacing each Z^{-2} with $-Z^{-2}$. The effect of this transformation on the all-pass sections in the 2-path filter is shown in (10.35). Except for a sign change on the transfer function, the phase substitution on the delays Z^{-2} is equivalent to a sign change applied to the coefficients of the all-pass stages. The negative sign of the transfer function is canceled if there is an even number of stages in a filter path.

The one exception to the filter coefficient absorbing the sign change occurs at the single delay in the lower leg of the 2-path filter. Here we must associate the j operator with the delay. This means that we declare the second leg of the 2-path filter to be the imaginary part of the time series. This means, in turn, that the summation performed at the output of the two paths is no longer a true sum, but rather an association, pairing the real and imaginary outputs as a complex sample of the filter.

$$H(Z^2) = \frac{1 + \alpha Z^2}{Z^2 + \alpha} \Rightarrow G(Z) = H(-Z^2) = -\frac{1 - \alpha Z^2}{Z^2 - \alpha} \tag{10.35}$$

Figure 10.56 shows how the transformation is applied to the signal-flow representation of the half-band filter. Note the sign changes and the assignment of the second path to the imaginary output component. Here we continue to invoke the option of down sampling the output of the half-band filter by a factor of 2-to-1 in concert with the filter's bandwidth reduction. Since the filter still contains polynomials in Z^2 we can use the noble identity to interchange the order of filtering and down sampling which results in the compact form shown in the bottom right of the figure.

The Hilbert-transform filter has always held the position of being our first exposure to a polyphase filter structure. In fact it is a 2-path filter, the precursor of the general M-path structure. It uses a combination of frequency-dependent phase shift and time-domain phase rotation to obtain destructive cancellation of signals over selected frequency bands. The Hilbert-transform cancels the negative-frequency segment of the spectra arranging for a broadband frequency-dependent 90-degree phase shift between the two arms of the filter. The time-domain phase shift applied through the j operator rotates the entire spectral response another 90 degrees to effect a 180-degree phase difference along the two paths over the negative frequency band.

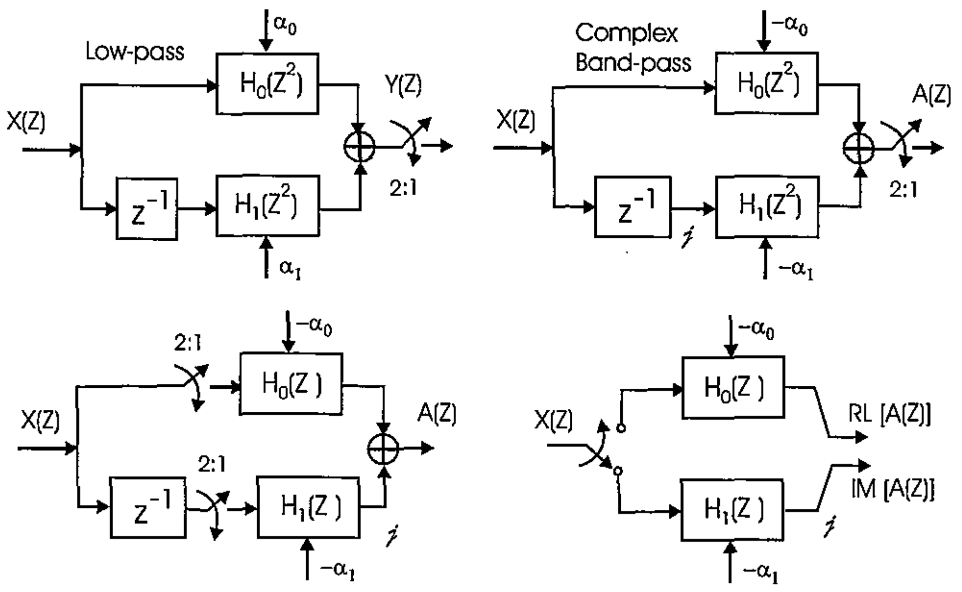


Figure 10.56 Low-pass Half-band Filter Translated to Positive-frequency Half-band Filter

We have two basic options on how to achieve the frequency-dependent 90 degree phase shift. The minimum resource option places recursive first-order all-pass polynomials in Z^2 in both arms and uses the lazy-S phase contours to obtain the desired phase difference between the two arms. While this option realizes the desired phase difference between the two arms, the phase slope between input and output of the filter is not linear. There are many applications in which linear phase is not a requirement. Automatic gain control is one and digital receivers containing an equalizer are another. If linear phase is a requirement, we expend additional resources, restrict the upper path to contain only delay elements, and have the lower path supply the desired phase difference.

Figure 10.57 illustrates the phase structure of a non uniform and a uniform phase filter designed for the same 5% transition bandwidth and 60-dB out-of-band attenuation. The two filters require 4-coefficients and 7-coefficients respectively to meet these specifications. Figure 10.58 presents the pole-zero diagrams for the two options. We see here the effect of the center frequency transformation on the system roots. Replacing Z^{-2} with $-Z^{-2}$ in the path filters rotated the pole positions 90 degrees, which is easily seen in the left figure where the poles have been moved from the Y axis to the X axis. Similarly, applying the j operator to the lower path rotated the zero positions the same 90 degrees. Compare the pole-zero diagram of Figure 10.58 with Figure 10.22.

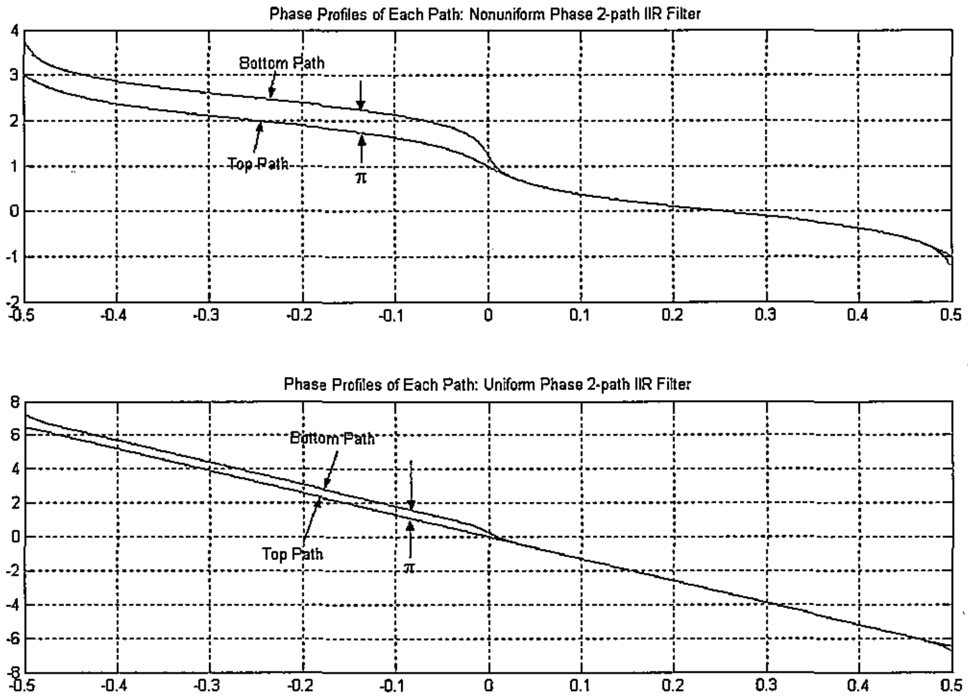
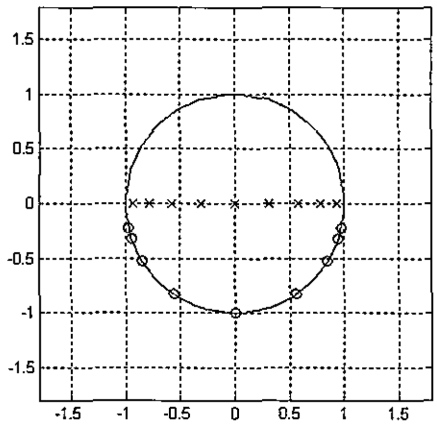


Figure 10.57 Path Phases of Two-path Recursive All-pass, Nonuniform Phase and Uniform Phase Hilbert-transform Filters



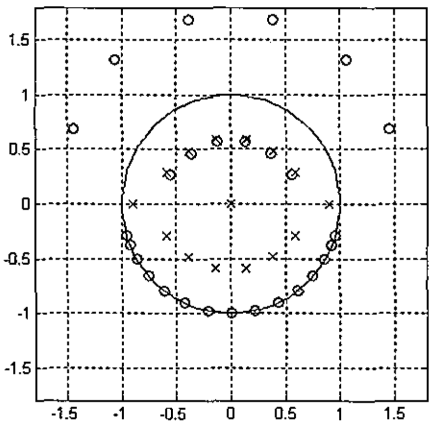


Figure 10.58 Pole-zero Diagrams for Nonuniform and Uniform Phase Hilbert-transform Filters

Figure 10.59 presents the frequency and phase responses of the two Hilbert-transform variants. The in-band phase profiles are inherited from the corresponding phase profiles of the two paths as seen in Figure 10.57. We easily see the stop band zero locations. When used in the resampling mode, the 4-coefficient, 9-pole/zero nonuniform phase filter requires 2-multiplies per input point to cancel the negative frequencies, while the 7-coefficient, 27-pole/zero uniform phase filter requires 3.5-multiplies per input point. For comparison, consider a 147-tap true half-band FIR filter that would be required to meet the same specification. When partitioned into a 2-to-1 polyphase filter, the upper path would contain the even-indexed sample points, which implement the delay to the center tap of the filter, while the lower path would contain the 74 symmetric odd-indexed samples. The filter would require 18-multiplies per input point if the architecture used the coefficient symmetry or 36-multiplies per input point if not. Thus the best linear phase FIR, the linear phase IIR, and the nonlinear phase filters would require 18-, 3.5-, and 2- multiplies respectively per input point to implement comparably performing Hilbert-transform filters.

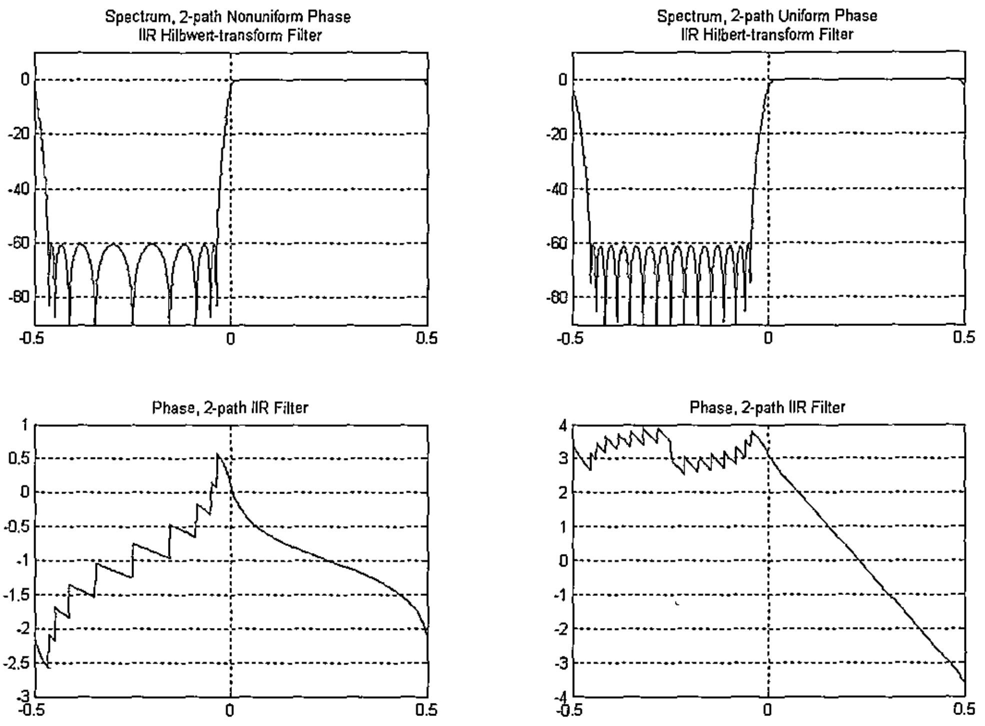


Figure 10.59 Frequency and Phase Response of Nonuniform and Uniform Phase 2-path Recursive All-pass Hilbert-transform Filters

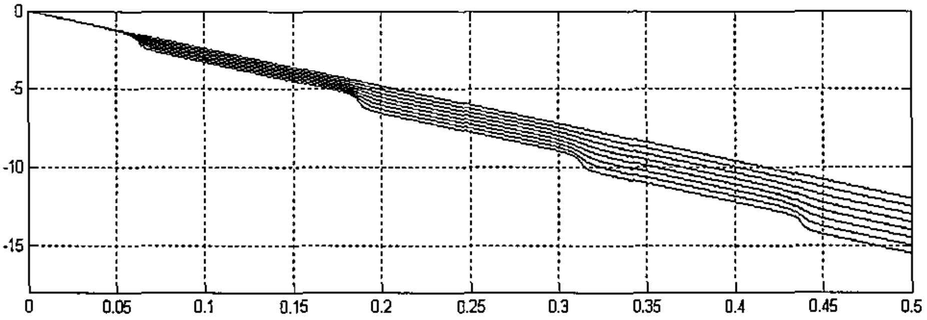
10.8 M-PATH RECURSIVE ALL-PASS FILTERS

This chapter started with a discussion of M-path recursive all-pass filters. We then introduced recursive all-pass filter structures and demonstrated their use in a rich subset of M-path filters, 2-path filters, and variants related to them. We now return to the M-path filter that can be thought of as an extension of the 2-path structure. We presented the standard structure of the M-path filter in Figure 10.2. In the case of nonlinear phase, the paths contain a delay and one or more all-pass filters formed as first-order polynomials in Z^{-M}. In the case of linear phase, the first path is only delay, and the remaining paths contain a delay and one or more all-pass filters formed as first- and second-order polynomials in Z^{-M}. The polynomials in Z^{-M} enable use of the noble identity, the interchange of the resampler, and the M-stage filter prior to the filtering operation rather than after.

As a specific example to demonstrate the characteristics of the recursive all-pass M-path filters we have designed and programmed a pair of 8-path filters with the same pass band and attenuation characteristics of the composite three-stage half-band filter discussed and presented in Figure 10.49. The specifications, recast in normalized frequency units, that the filter must satisfy are that pass band frequency is 0.0625 (1/16), stop band frequency is 0.09375 (1.5/16), pass band ripple is less than 0.1-dB, and stop band attenuation is at least 75-dB. The design routine for the nonuniform phase filter determined that these specifications required 2-stages in paths 0, 1, 2, 3, 4, and 5, and 1-stage in paths 6 and 7, each stage being a first-order polynomial in Z^{-8}. Thus 14 coefficients define this 8-path filter. Remarkably, since these coefficients describe 14 first-order polynomials in Z^{-8}, the composite filter contains 112 pole-zero pairs. In like fashion the design routine for the linear-phase filter determined that these specifications required 24-delays in path-0 and both a first-order and a second-order stage in paths 1, 2, 3, 4, 5, 6, and 7. These stages are polynomials in Z^{-8}. Here 21-coefficients define the eight path filter. Since these coefficients form the equivalent of 24 first-order polynomials in Z^{-8}, the composite filter contains 192 pole-zero pairs.

The poles of the polynomials in Z^{-8} have 8-fold symmetry around the unit circle, and as we traverse the unit circle we observe rapid changes in phase as we pass these poles. The spectral regions in the vicinity of the poles represent transition bands. These phase-change regions are easily seen in Figure 10.60, which presents the phase profiles of the 8-paths of two 8-path filters. Between the transitions, the separate phase profiles differ by constant phase differences. In the five intervals seen in the positive frequency span, the phase differences are 0, 45, 90, 135, and 180 degrees. The three unseen intervals in the negative frequency span have phase differences of –45, –90, and –135 degrees. When the signals in these 8-paths are summed they constructively add in the 0-degree difference frequency interval (the pass band) and destructively add in the remaining intervals (the stop bands). The transition regions between the bands represent transition regions, either transitioning from pass band to stop band or from stop band to stop band.

Figure 10.61 show the pole-zero diagrams for the pair of 8-path filters, and Figure 10.62 shows a detail section of the unit circle near the first set of poles. We see the zeros on the unit circle that define the multiple stop bands as well as the pole clusters on the 8-roots of $-\alpha$ formed by the first-order polynomials in Z^{-8}. We also see the cluster of zeros alongside the pole clusters shielding the stop band intervals from the pole clusters. Figures 10.63 and 10.64 present the frequency response, a pass band detail of the frequency response, the impulse response, and the pass band phase response for the two versions of the 8-path recursive filter. The first characteristic that strikes our attention is the multiple transition bands between successive stop band intervals. We explained this as the traversal of the pole clusters near the unit circle at the roots of the first-order polynomials in Z^{-8}. When the output of these filters is maximally decimated (8-to-1) the multiple transition bands alias to a common interval, the transition interval from the pass band to stop band.



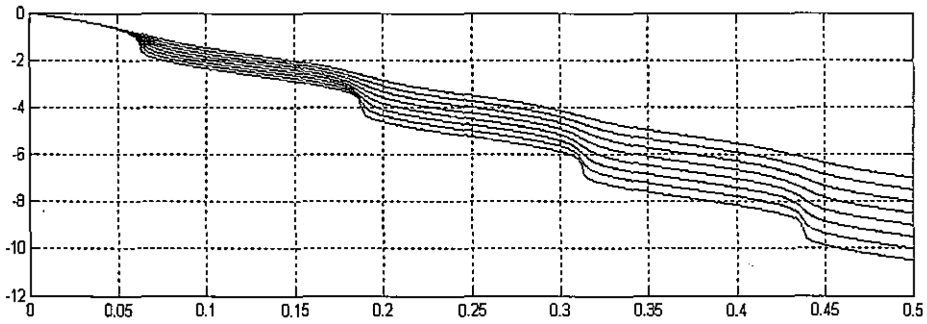
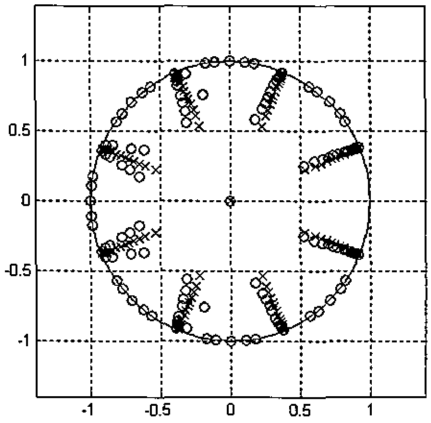


Figure 10.60 Phase Profiles of 8-paths of 8-path Nonuniform and Uniform Phase Polyphase Filter



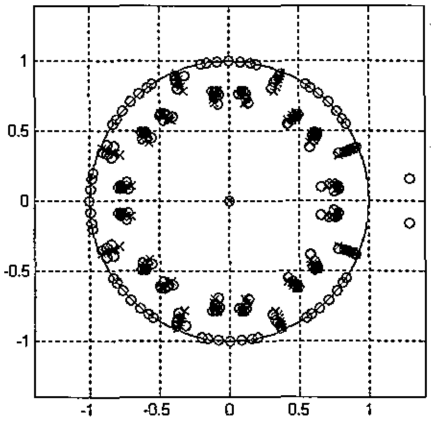


Figure 10.61 Pole-zero Plots of Composite 8-path Recursive Nonuniform and Uniform Phase Filters

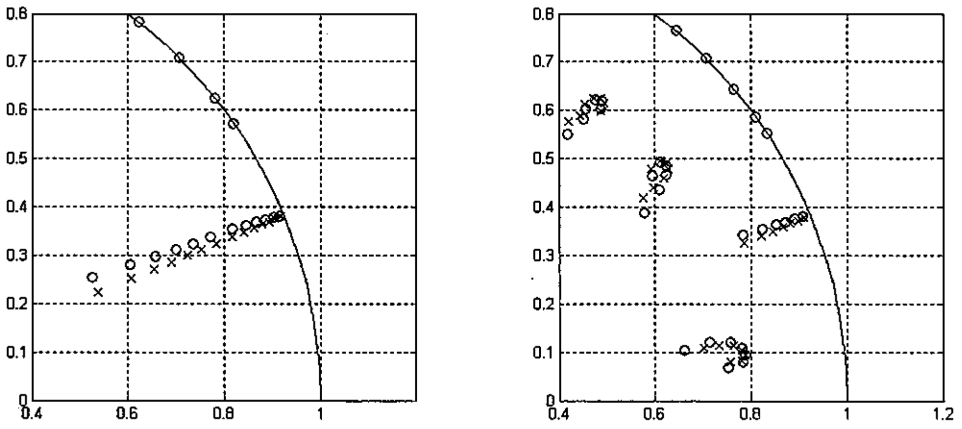


Figure 10.62 Detail of Pole-zero Plots of Composite 8-path Filters

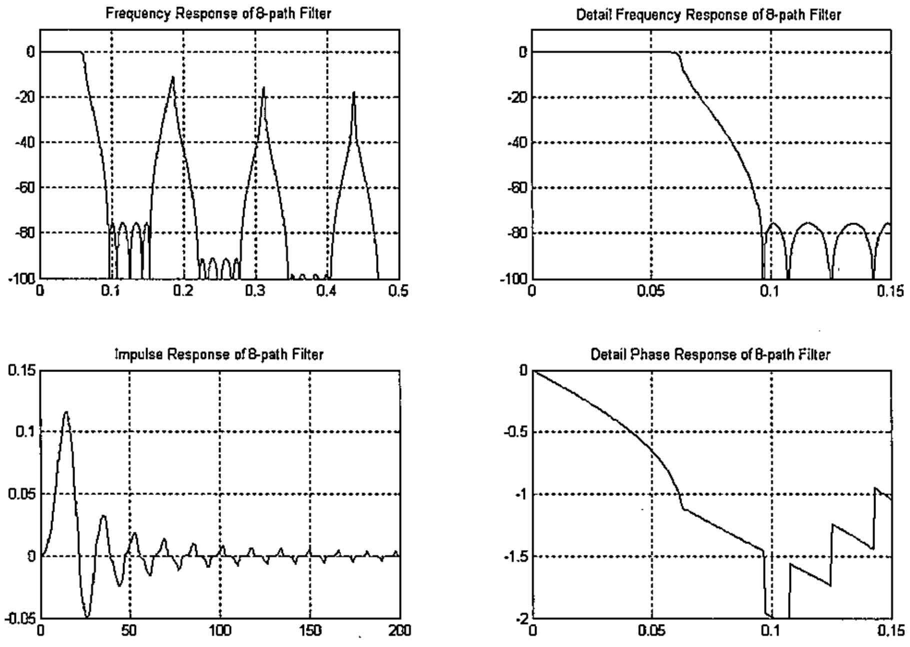


Figure 10.63 Frequency, Impulse, and Phase Responses of Nonuniform Phase 8-path Filter

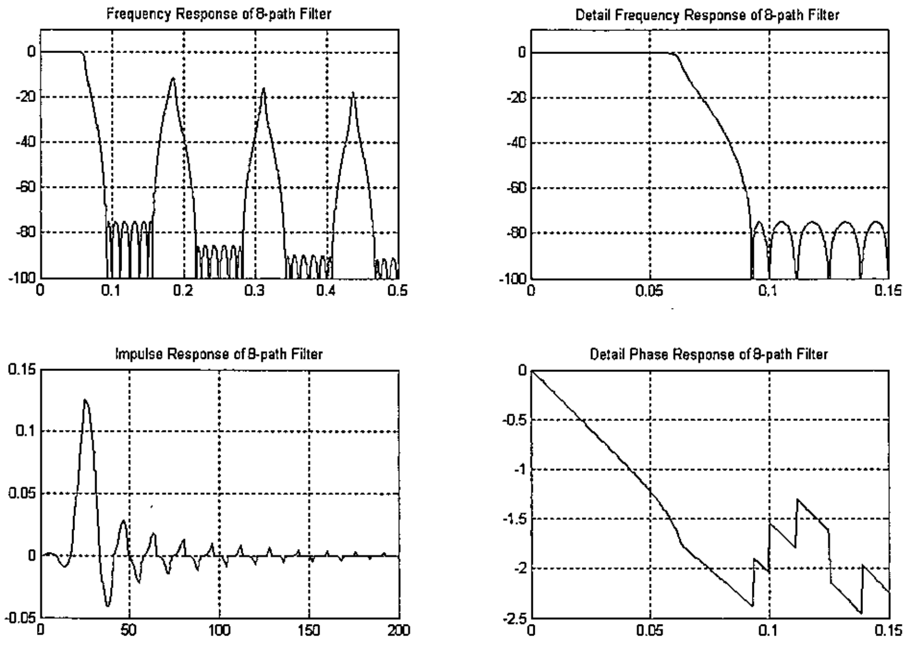


Figure 10.64 Frequency, Impulse, and Phase Responses of Uniform Phase 8-path Filter

10.9 Iterated Half-band Filters

The final architecture we consider for the all-pass structures is that of an iterated recursive filter. The iterated filter concept is useful when the filter bandwidth is a small fraction of the signal sample rate. Rather than design the filter at the input sample rate, we design it at a fraction of the input rate, which often leads to shorter filters and always results in filters exhibiting reduced sensitivity to finite arithmetic. This reduced sensitivity is the result of using lower Q-poles in the recursion process. The implementation of the iterated filter involves designing the filter at a reduced sample rate, then zero packing its impulse response and interpolating up to the desired higher sample rate with postprocessing filters. The iterative filter up samples the filter impulse response to the higher sample rate rather than down samples the data to the reduced sample rate, as we did earlier.

To demonstrate the iterative filter process, let us again consider a filter meeting the specifications outlined in the previous section. These specifications, in normalized frequency units, are pass band frequency of 0.0625 (1/16), stop band frequency of 0.09375 (1.5/16), pass band ripple of less than 0.1-dB, and stop band attenuation of at least 75-dB. We will perform the design for a sample rate one-fourth of the original sample rate but operate the filter at the system rate by 1-to-4 zero packing and then remove the spectral replicates with two half-band filters. The structure of the iterative design is shown in Figure 10.65. We determined that a 7th-order input filter was required to satisfy the designed goals when cast as a half-band filter with pass band at 0.25 (1/4) and stop band at 0.375 (1.5/4). This filter requires 3-coefficients. Since the half-band filter is designed as polynomials in Z^{-2}, the 1-to-4 zero packing converts it to polynomials in Z^{-8}. The second-stage filter is also a 7th-order, 3-coefficient half-band filter that is zero packed 1-to-2. Finally, the third stage is a 5th-order, 2-coefficient half-band without zero packing. The spectral responses of these filters are shown in Figure 10.66, and the composite spectra at successive filter outputs are shown in Figure 10.67. The spectra shown in each panel are the output spectra from the previous stage, which is the input to the current stage and the spectral response of the current stage.

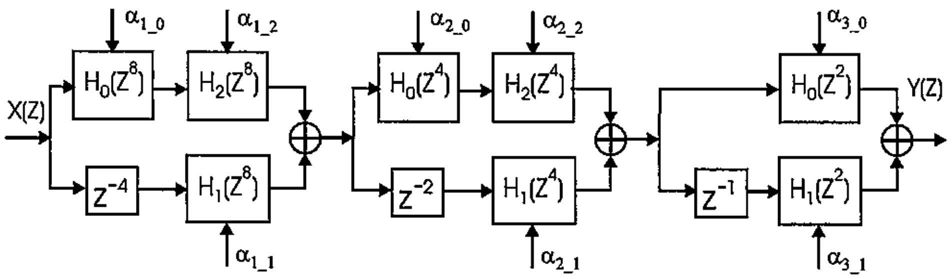


Figure 10.65 Iterated Filter, Up Sampled 1-to-4 by Zero Packing and Then Filtered to Suppress Spectral Replicates

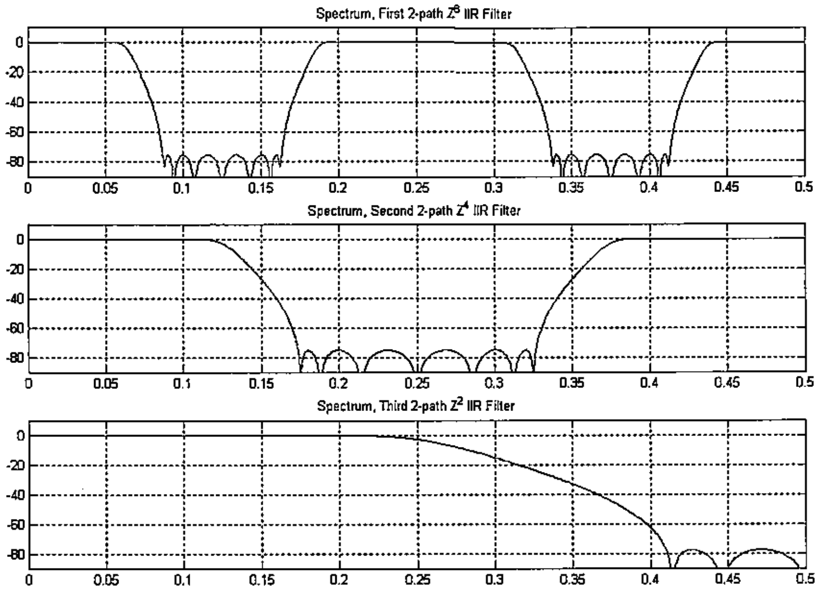


Figure 10.66 Spectra of 1-to-4 Zero-packed First Stage, of 1-to-2 Zero-packed Second Stage, and Non-Zero-packed Third Stage of Iterated Filter

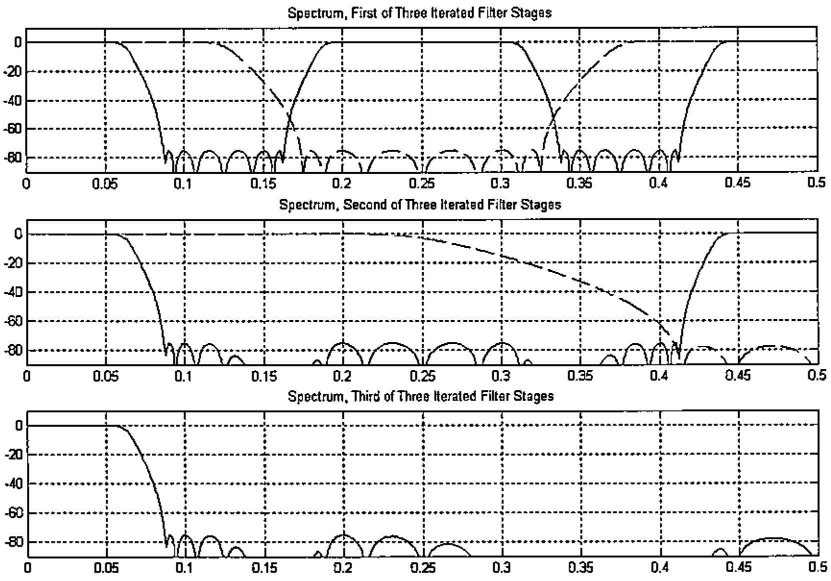


Figure 10.67 Spectral Response of Cascade Filters in Iterated Filter Chain

10.9.1 Final Comparisons

We can now compare the workload to reduce the bandwidth and sample rate by a factor of 8-to-1 for various resampling options. Our reference design was a standard 8th-order elliptic filter implemented as a cascade of 4-biquadratic stages that required 20-multiplies per input data point. We then considered a 2-path tuned half-band filter that required a 9th-order half-band prototype that required 9-multiplies per data point, resulting in more than a 2-to-1 savings. We then examined the option of a cascade of three half-band filtering stages and found the total workload to be approximately 4.0-multiplies per input. Continuing with the other option, we looked at the 8-path resampling filters. The nonuniform phase 8-path filter requires 14-multiples per 8-input data samples, which is equivalent to 1.8-multiplies per input while the uniform phase 8-path filter requires 21-multiplies per 8-input data samples, which is equivalent to 2.7-multiplies per input. For comparison, an 8-path polyphase partitioned FIR filter with the same spectral characteristics requires 13-multiplies per input sample point.

Finally we looked at the iterated half-band filter option that required 8-multiplies per input point. We conclude that the 8-path nonuniform phase and the 8-path uniform phase polyphase implementation are ranked first and second as the most efficient of the options for a single channel-filtering task. The 8-path implementations become even more efficient when the polyphase partition is used as a channelizer. Suppose, for instance, five channels centered baseband (–2/8, –1/8, 0, 1/8, 2/8) are to be down converted by a set of phase rotators following the path processing. The phase rotators are nearly trivial for this example, but could be applied with an 8-point FFT with three of the terms discarded. Now the 1.8 or 2.7 multiplies per input point is amortized over 5-output points for a filter processing 0.36 or 0.54 multiplies per input per channel.

References

Ansari, R. and B. Liu, "Efficient Sampling Rate Alteration Using Recursive (IIR) Digital Filters," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, Vol. 31, Dec. 1983, pp. 1366 - 1373.

Drews, W. and L. Gaszi, "A New Design Method for Polyphase Filters Using All-pass Sections," *IEEE Trans. on Circuits and Systems*, Vol. 33, Mar. 1986, pp. 346 - 348.

harris, fred, "On the Design and Performance of Efficient and Novel Filter Structures Using Polyphase Recursive All-pass Filters," Keynote Presentation ISSPA-92 (International Symposium on Signal Processing), Gold Coast, Australia, Aug. 16-21, 1992.

harris, fred and Eric Brooking, "A Versatile Parametric Filter Using an Imbedded All-Pass Sub-Filter to Independently Adjust Bandwidth, Center Frequency, and Boost or Cut," 95th Audio Engineering Society (AES) Convention, New York, New York, Oct. 7-10, 1993.

harris, fred, Robert Caulfied, and William McKnight, "Use of All-pass Networks to Tune the Center Frequency of Sigma Delta Modulators," 27th Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, Oct. 31 to Nov. 3, 1993.

harris, fred, Itzhak Gurantz and Shimon Tzukerman, "Digital (T/2) Low pass Nyquist Filter Using Recursive All-pass Polyphase Resampling Filter for Signaling Rates 10 kHz to 10 MHz," 26th

Annual Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, Oct. 26-28, 1992.

harris, fred, Maximillien d'Oreye de Lantramange, and Anton Constantinides, "Design and Implementation of Efficient Resampling Filters Using Polyphase Recursive All-Pass Filters," 25th Annual Asilomar Conference on Signals, Systems, and Computer, Pacific Grove, CA, Nov. 4-6, 1991.

Jovanovic-Dolecek, Gordana, *Multirate Systems: Design and Applications*, London, Idea Group, 2002.

Renfors, Markku. and T. Saramaki, "Recursive N-th Band Digital Filters, Parts I and II," *IEEE Trans. CAS*, Vol. 34, pp. 24 - 51, Jan. 1987.

Vaidyanathan, P. P., *Multirate Systems and Filter Banks,* Englewood Cliffs, NJ, Prentice-Hall, Inc., 1993.

Valenzuela, R. A. and Anton Constantinides, "Digital Signal Processing Schemes for Efficient Interpolation and Decimation," *IEE Proceedings*, Dec. 1983.

Problems

10.1 Verify that any transfer function of the form shown in the equation is an all-pass network; that is, $|H(\theta)] = 1$ for all θ. Sketch the location of the numerator and denominator roots for M = 1, 2, 3, and 4.

$$H(Z) = \frac{1-\alpha Z^M}{Z^M-\alpha}$$

10.2 Group delay is the phase slope $d(\phi)/d(\theta)$ of a transfer function. Determine the group delay of a first-order all-pass network shown here. Plot the group-delay for various values of the parameter α.

$$H(Z) = \frac{1-\alpha Z}{Z-\alpha}$$

10.3 Form a transfer function H(Z) shown next, as the sum of the two trivial all-pass networks $P_0(Z)$ and $P_1(Z)$. Comment on the root locations, and determine the scale factor required to obtain a unity gain filter for arbitrary value of the parameter α.

$$P_0(Z) = \frac{1-\alpha Z}{Z-\alpha}, P_1(Z) = 1$$

$$H(Z) = P_0(Z) + P_1(Z)$$

10.4 Form a transfer function H(Z) shown next, as the sum of the two trivial all-pass networks $P_0(Z)$ and $P_1(Z)$. Comment on the root locations as functions of the parameters α and β.and determine the scale factor required to obtain a unity gain filter for arbitrary value of the parameters α and β.

$$P_0(Z) = \frac{1-\alpha Z}{Z-\alpha}, P_1(Z) = \beta$$

$$H(Z) = P_0(Z) + P_1(Z)$$

10.5 Form a transfer function H(Z) shown here, as the sum of the two trivial all-pass networks $P_0(Z)$ and $P_1(Z)$. Determine and plot the root locations as a function of the parameter α.

$$P_0(Z) = \frac{1+\alpha Z^2}{Z^2+\alpha}, P_1(Z) = \frac{1}{Z}$$
$$H(Z) = P_0(Z) + P_1(Z)$$

10.6 Form a transfer function H(Z) shown here, as the sum of the two all-pass networks $P_0(Z)$ and $P_1(Z)$. Determine the range of α_0 and α_1 for which the numerator roots are confined to the unit circle.

$$P_0(Z) = \frac{1+\alpha_0 Z^2}{Z^2+\alpha_0}, P_1(Z) = \frac{1}{Z}\frac{1+\alpha_1 Z^2}{Z^2+\alpha_1}$$
$$H(Z) = P_0(Z) + P_1(Z)$$

10.7 Use the MATLAB design program *tony_des* to design a recursive half-band low pass filter with stop band normalized frequency edge equal to 0.30 and with at least 60-dB stop band attenuation. Design an IIR Tchebyschev-II filter with the same 3-dB edge, stop band edge, and stop band attenuation. Form and compare the frequency response of the two filters. Comment on their relative implementation complexity.

10.8 Use the MATLAB design program *tony_des* to design a recursive half-band low pass filter with stop band normalized frequency edge equal to 0.30 and with at least 60-dB stop band attenuation. Design a Remez-based FIR filter with the same 3-dB edge, stop band edge, and stop band attenuation. Form and compare the frequency response of the two filters. Comment on their relative implementation complexity.

10.9 Use the MATLAB design program *Lineardesign* to design a recursive linear phase half-band low pass filter with stop band normalized frequency edge equal to 0.30 and at least 60-dB out-of-band attenuation. Design a Remez-based FIR filter with the same 3-dB edge, stop band edge, and stop band attenuation. Form and compare the frequency response of the two filters. Comment on their relative implementation complexity.

10.10 Use the MATLAB design program *tony_des* to design a recursive low pass filter with pass band and stop band normalized frequency edges equal to 0.1 and 0.2 respectively and with at least 60-dB stop band attenuation. Design an IIR Tchebyschev-II filter with the same 3-dB edge, stop band edge, and stop band attenuation. Form and compare the frequency response of the two filters. Comment on their relative implementation complexity.

10.11 Use the MATLAB design program *tony_des* to design a recursive low pass filter with pass band and stop band normalized frequency edges equal to 0.1 and 0.2 respectively and with at least 60-dB stop band attenuation. Design a Remez based FIR filter with the same 3-dB edge, stop band edge, and stop band attenuation. Form and compare the frequency response of the two filters. Comment on their relative implementation complexity.

10.12 We want to design a low pass filter with the following specifications:

Sample Rate	100 kHz		
Pass Band	0-to-5 kHz	In-band Ripple	< 0.1-dB

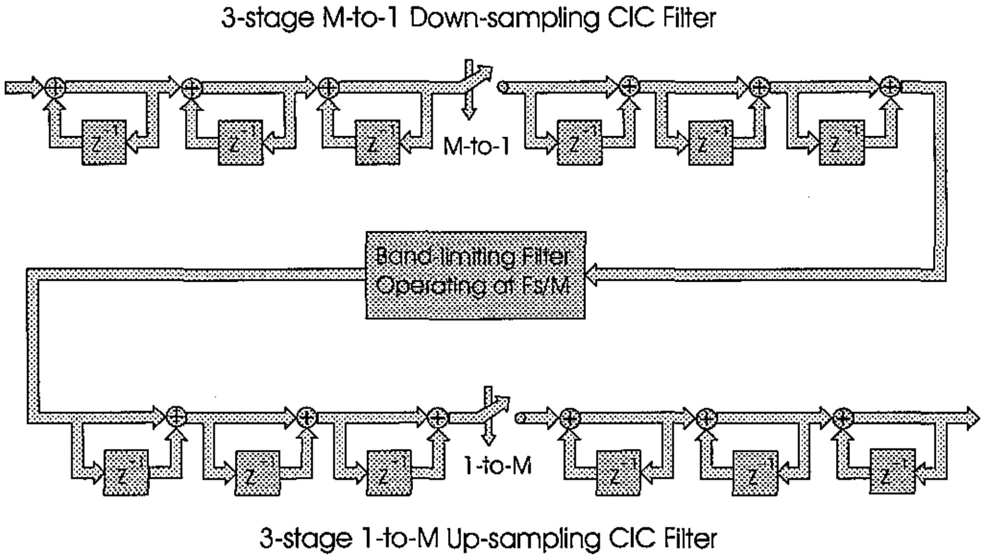
Stop Band	8-to-50 kHz	Stop band Atten.	60-dB

Use *tony_des* to design a recursive nonuniform phase filter that satisfies this specification.

We now implement a cascade of two resampling half-band filters and a final spectral shaping filter to achieve the same filter specifications. Design the pair of half-band 2-to-1 resampling filters and the final output stage. What is the relative work between the two implementations?

CHAPTER 11

Cascade Integrator Comb Filters


3-stage M-to-1 Down-sampling CIC Filter
z^{-1}
M-to-1
Band-limiting Filter Operating at Fs/M
1-to-M
3-stage 1-to-M Up-sampling CIC Filter

Digital filters are formed by a standard set of resources, memory or delays, summing junctions, multipliers, and resamplers. Architectures that combine these resources build a variety of filtering systems. When faced with two or more contending filtering architectures to solve a given problem, we compare their relative cost as well as their relative performance. In the early days of *digital signal processing* (DSP) the number of multiplies and data transfers per data sample were standard cost measures. Another comparative cost is sensitivity to finite precision coefficients and required register and accumulator widths. A delight to *application specific integrated circuit* (ASIC) designers, and to a lesser extent *field programmable gate array* (FPGA) designers, is a class of filters that does not require multipliers, requires few data transfers, and further exhibits an easy to derive relationship between filter specifications and accumulator widths. These filters are variants of a structure based on the sliding average filter. This chapter presents these filters and discusses their performance under finite arithmetic.

11.1 A Multiply-free Filter

A filter with a rectangle-shaped impulse response is called a boxcar or a sliding average filter. It is a simple FIR filter with unit-valued coefficients that performs a filtering task without multiplies. The quality of the filtering is actually not very good since the spectral response of the boxcar filter, as shown in (11.1) and illustrated in Figure 11.1, exhibits only 13-dB attenuation. The simplicity of one form of implementation is so attractive we are drawn to this filter even though the filtering performance is not very good. We will simply have to find a way to improve its performance.

$$H(\theta) = \frac{\sin(\theta \frac{M}{2})}{\sin(\theta \frac{1}{2})} \tag{11.1}$$

The FIR filter implementation of the boxcar filter is shown in Figure 11.2.

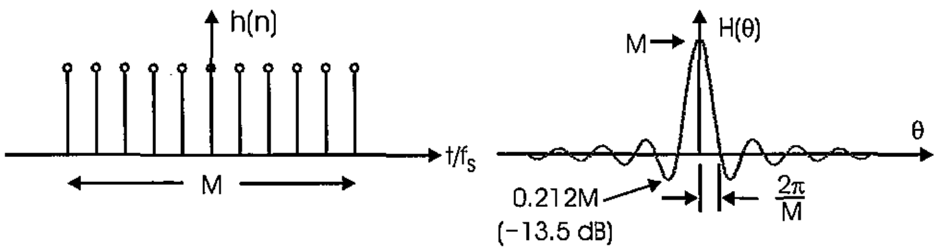


Figure 11.1 Boxcar Filter Impulse Response and Frequency Response

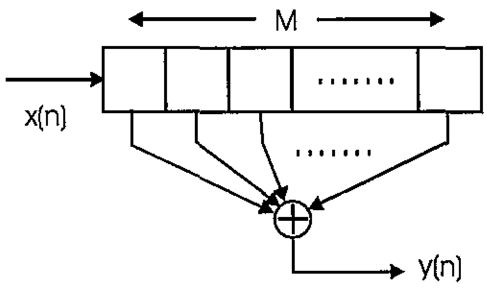


Figure 11.2 FIR Filter Implementation of Boxcar Filter

Conceptually, the filter computes an output as follows: An input arrives, and the data in the register shifts one place to the right to accommodate the new arrival. The filter forms the sum of the contents of the registers and outputs this sum. This is shown in (11.2).

$$y(n) = \sum_{k=0}^{M-1} x(n-k) \tag{11.2}$$

The process is repeated upon the arrival of each new input sample. If we think about it, this is not a very efficient implementation of the filter. When the new input arrives, the shift to the right of the register contents discards the sample that had arrived M-samples ago to make room for the latest input sample. The new output sum differs from the previous sum by the addition of the new data point and the removal of the M-sample old data point. A recursive form of the boxcar filter can be implemented by altering the previous sum by the known difference. This form of the sum is shown in (11.3).

$$\begin{aligned} y(n) &= \sum_{k=0}^{M-1} x(n-k) = x(n) - x(n-M) + \sum_{k=0}^{M-1} x(n-1-k) \\ &= [x(n) - x(n-M)] + y(n-1) \end{aligned} \tag{11.3}$$

This form of the filter is known as the CIC. The block diagram of this filter is shown in Figure 11.3. The integrator is the "I" in the name CIC, and it is easily identified as the recursive accumulator. By default, the M-units of delay and the subtraction preceding the integrator must be the comb filter, the second "C" in the name CIC. This section has a simple impulse response a 1 at time index 0 followed by a –1 at time index M. The Z-transform of this impulse response is shown in (11.4). The reason this filter is called a comb filter can be traced to its frequency response, which as shown in (11.5) is a sinusoid. The magnitude response of the filter, shown in Figure 11.4, has the appearance of a rectified sine wave with M-periodic zeros spanning the frequency axis. The periodic zeros remind us of the teeth of a comb, hence the name, comb filter.

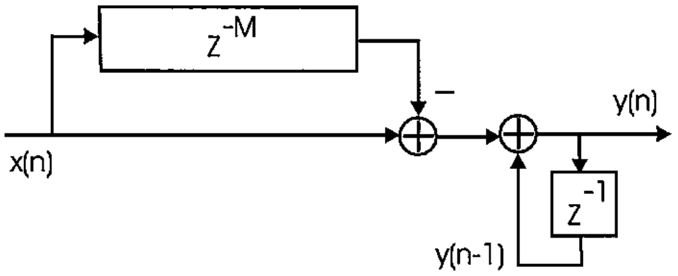


Figure 11.3 Cascade Integrator Comb Filter

$$C(Z)=1-Z^{-M} \tag{11.4}$$

$$\begin{aligned} C(\theta) &= 1 - e^{-jM\theta} \\ &= e^{-jM\theta/2}[e^{+jM\theta/2} - e^{-jM\theta/2}] \\ &= 2je^{-jM\theta/2}\sin(M\theta/2) \end{aligned} \tag{11.5}$$

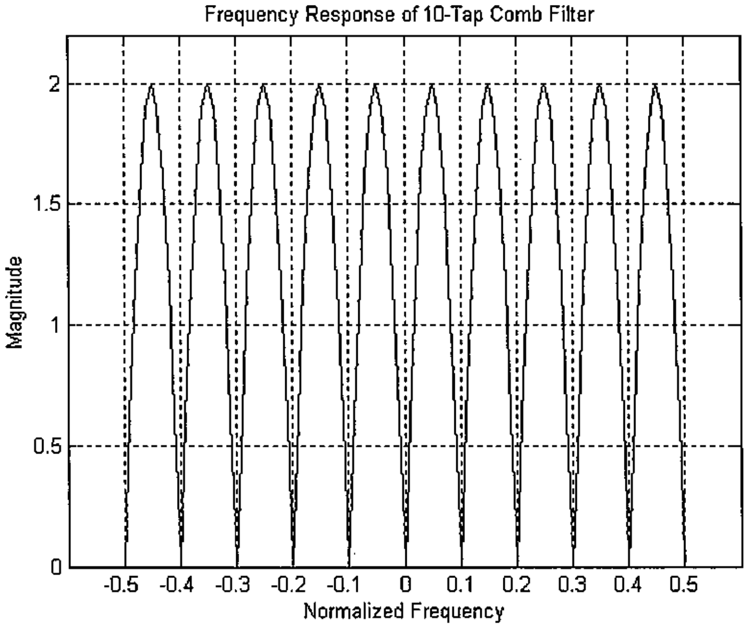


Figure 11.4 Frequency Response of 10-tap Comb Filter

Notice from Figure 11.3 that in this form of the filter only two summations are required to form the filter output while in the direct implementation, indicated in Figure 11.2, M-additions are required to form the filter output. This is valid for any M. We are thus able to replace 100 adds with 2 adds if we implement the boxcar filter as a CIC filter. Both implementations require the same amount of memory but modifications yet to come will reduce the memory required for the CIC.

An alternate derivation of the CIC structure is available from the Z-transform of the boxcar filter's impulse response. This is shown in (11.6), which we recognize as the first M-terms of the geometric series as shown in (11.7).

$$H(Z) = \sum_{n=0}^{M-1} Z^{-n} \tag{11.6}$$

$$H(Z) = 1 + Z^{-1} + Z^{-2} + Z^{-3} + \cdots + Z^{-(M-1)} \tag{11.7}$$

The closed form of this series is shown in (11.8).

$$H(Z) = \frac{1 - Z^{-M}}{1 - Z^{-1}} \tag{11.8}$$

Equation (11.8) can be cast as a polynomial in Z as in (11.9).

$$H(Z) = \frac{1}{Z^{(M-1)}} \frac{Z^M - 1}{Z - 1} \tag{11.9}$$

In this form of the Z-transform we can easily see that the zeros, the numerator roots, are uniformly distributed around the unit circle at the M-roots of unity and that the zero at z = 1 is canceled by the pole, a denominator root, at the same location. This pole zero cancellation is certainly valid in the ratio of polynomial representation of the filter but may not be in the following partition.

Returning to (11.8) we perform a questionable step and then examine the consequences of that maneuver. We treat the numerator and denominator of (11.8) as if they reside in different filters. Normally we are free to do this; what we have to question here is are we able to separate the numerator and denominator when they contain canceling roots? Technically we cannot place canceling roots in separate filters. I once had a student get angry with me for separating them. We will shortly see why we are not permitted do so and then find a work around. The separated form of the filter is shown in (11.10).

$$H(Z) = [1 - Z^{-M}]\left[\frac{1}{1 - Z^{-1}}\right] \tag{11.10}$$

An obvious, but as just mentioned perhaps incorrect, conclusion to be drawn from this partitioned form of the boxcar filter is that the filter can be implemented as the cascade of two filters, an integrator and a comb. This is the structure shown in Figure 11.3, a structure arrived at from a different perspective.

Figure 11.5 presents two filter structures that we now compare to the original boxcar filter. The two structures are reordered versions of the comb filter and the integrator. This reordering is standard fare for linear systems since linear time invariant systems commute. Shown by each of the three filters is its impulse response. The impulse response of the tapped delay line version of the boxcar is simple and obvious.

The impulse response of the comb and integrator cascade is formed in two phases. We first see the comb response, which is a sequence of two impulses separated by M-samples. The first impulse applied to the integrator circulates around the integrator, outputting the same unit valued samples. When the second impulse arrives at the integrator it cancels the circulating first impulse, and the integrator response goes to zero. The composite response of the cascade filters matches the response of the boxcar. No surprise yet!

The impulse response of the integrator and comb cascade is also formed in two phases. We first see the integrator response, which is a recirculating copy of the input impulse that is a step or DC response. This step is delivered directly to the output and matches the boxcar response. M samples into the response, and an M-unit delayed version of the output step arrives at the output summing junction where they subtract from the direct path output of the integrator and set each output sample to zero. This output sequence also matches the response of the boxcar filter. If an observer joined us now and examined only the input and output of this filter she would see zero input and zero output and might conclude that the filter is at rest, a standard attribute of a stable system. In fact it isn't; it is circulating the original input impulse in the integrator and delivering pairs of samples via the comb filter to sum to zero and give the impression the system is at rest. What we are witnessing is the effect of an internal state of the system, the integrator that is neither observable nor controllable from external ports. Remember that this state did not exist in the original boxcar filter, it is the denominator root canceled by a zero of the numerator.

Figure 11.6 presents the same filter structures we just examined but this time we compare their responses to an input step. Shown by each of the three filters is its step response. The step response of the tapped delay line version of the boxcar contains two distinct intervals. The first interval of length M is a ramping transient response that lasts until the filter is filled. The second interval is the steady state response to the step of constant level M.

The step response of the comb and integrator cascade is also formed in two phases. We first see the comb response, which is a square wave sequence of length M-samples. The square wave forms the transient ramp in the integrator. At the end of M-samples the comb response goes to zero and the integrator stops accumulating inputs. At this point the integrator output has reached steady state and circulates and outputs its constant level of M. The composite response of the cascade filters matches the response of the boxcar. Still no surprise yet!

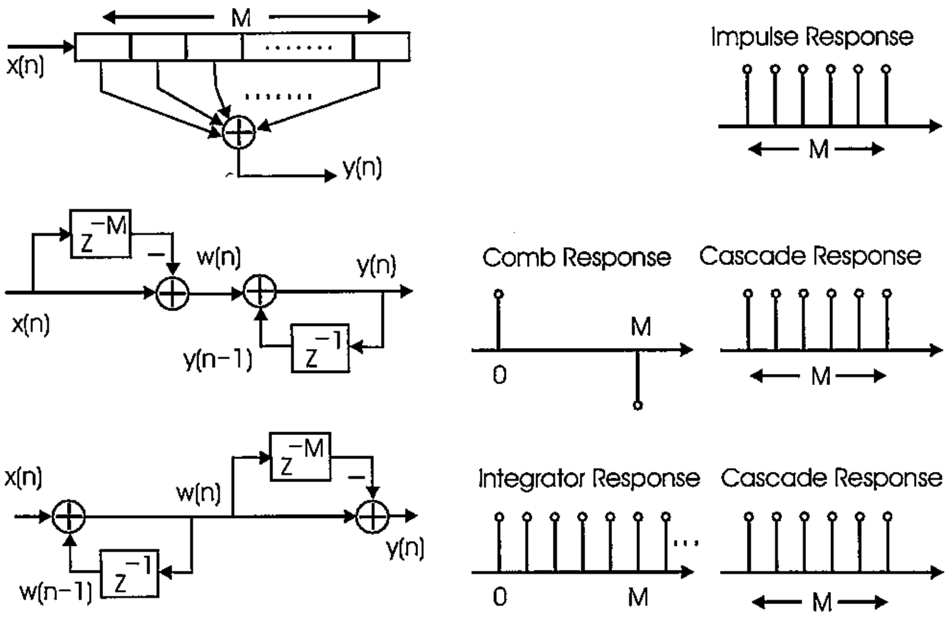


Figure 11.5 Structures and Associated Impulse Responses of Boxcar, of Cascade Comb and of Integrator, and Cascade Integrator and Comb

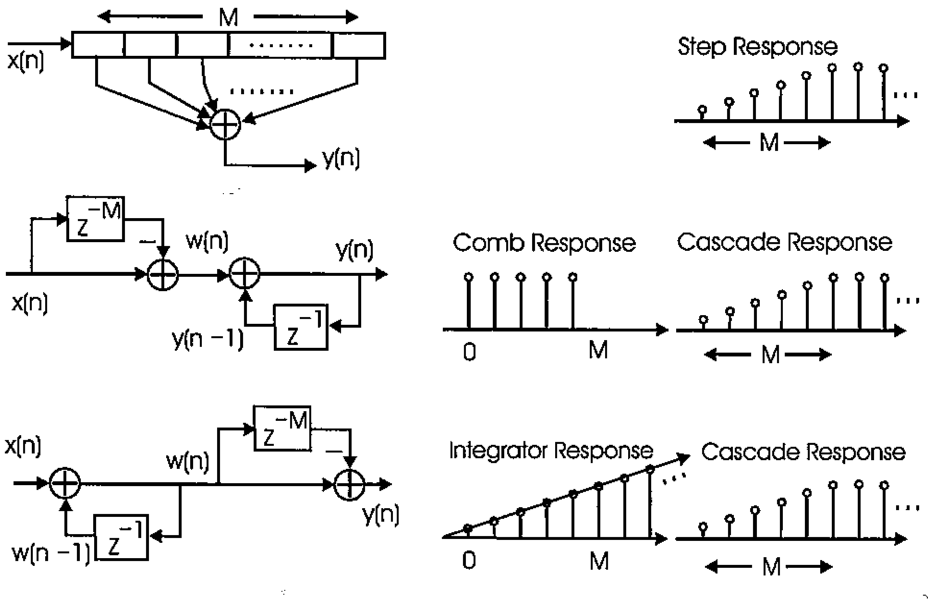


Figure 11.6 Structures and Associated Step Responses of Boxcar, of Cascade Comb and of Integrator, and Cascade Integrator and Comb

The step response of the integrator and comb cascade is also formed in two phases. We first see the integrator response, which is a ramp sequence formed by integrating the input step. This ramp is delivered directly to the output as the start of the boxcar response. M samples into the response, an M-unit delayed version of the output ramp arrives at the output summing junction where they subtract from the direct path delivered by the integrator to form a constant difference of amplitude M for each output sample. This output sequence also matches the response of the boxcar filter. If an observer joined us now and examined only the input and output of this filter she would see a finite input and a finite output and conclude that the filter is stable, an alternate definition of a stable system. In fact it is not stable in this sense since the integrator is on the way to infinity. The difference between the ramp and the M-unit delayed ramp formed in the comb filter is a constant even as the ramp climbs without bound. Here again we are witnessing the effect of an internal state of the system that is neither observable nor controllable. Any finite state machine that tries to implement this structure is bound to overflow its accumulator.

11.2 Binary Integers and Overflow

We have identified a concern about register overflow in a CIC filter. To help understand the overflow we now review a number of ways binary numbers are used to represent integers. Equation (11.11) presents the standard notation for binary numbers.

$$N = \sum_{i=0}^{b-1} a_i 2^i; \quad a_i = 0,1 \tag{11.11}$$

Standard notation assigns the least significant bit to the right so that (11.11) can be written specifically as in (11.12).

$$N = a_{b-1}2^{b-1} + a_{b-2}2^{b-2} + \cdots + a_2 2^2 + a_1 2^1 + a_0 2^0 \tag{11.12}$$

Alternatively, we can represent the binary number by an ordered set, with position in the ordered set implying the appropriate power of 2. This is the standard notation we use in decimal notation. This notation form is shown in (11.13).

$$\begin{array}{ll} \text{Implied Weight:} & 2^{b-1}\ 2^{b-2}\ 2^{b-3} \cdots\cdots 2^2\ 2^1\ 2^0 \\ & N = a_{b-1}\ a_{b-2}\ a_{b-3} \cdots\cdots a_2\ a_1\ a_0 \end{array} \tag{11.13}$$

An example of the binary representation of a set of decimal integers with implied powers of 2 is shown in Table 11-1.

Table 11-1 Decimal Integers and Their 3-bit Binary Representation

Decimal	Binary
0	0 0 0
1	0 0 1
2	0 1 0
3	0 1 1
4	1 0 0
5	1 0 1
6	1 1 0
7	1 1 1

The integers identified in Table 11-1 are all positive. If we require both positive and negative representations of our integers we have to add a sign modifier. The form of a binary number with a sign modifier is shown in (11.14)

$$\begin{array}{ll} \text{Implied Weight:} & s \quad 2^{b-1} \; 2^{b-2} \; 2^{b-3} \cdots\cdots 2^2 \; 2^1 \; 2^0 \\ & N = a_b \quad a_{b-1} \; a_{b-2} \; a_{b-3} \cdots\cdots a_2 \; a_1 \; a_0 \end{array} \tag{11.14}$$

In *sign-magnitude* notation, the sign bit is a multiplier +1 or –1. In *offset-binary* notation, the sign bit is an added term a_b2^b and the number is decoded by subtracting 2^b from its binary representation. In *2's-complement* notation, the sign bit is an added $-a_b2^b$. Examples of 4-bit binary numbers in the three binary representations are shown in Table 11-2.

Table 11-2 Decimal Integers and Their Signed 4-Bit Binary Representation

Decimal	Sign-magnitude	Offset-binary	2's-complement
+7	0 1 1 1	1 1 1 1	0 1 1 1
+6	0 1 1 0	1 1 1 0	0 1 1 0
+5	0 1 0 1	1 1 0 1	0 1 0 1
+4	0 1 0 0	1 1 0 0	0 1 0 0
+3	0 0 1 1	1 0 1 1	0 0 1 1
+2	0 0 1 0	1 0 1 0	0 0 1 0
+1	0 0 0 1	1 0 0 1	0 0 0 1
+0	0 0 0 0	1 0 0 0	0 0 0 0
–0	1 0 0 0	N/A	N/A
–1	1 0 0 1	0 1 1 1	1 1 1 1
–2	1 0 1 0	0 1 1 0	1 1 1 0
–3	1 0 1 1	0 1 0 1	1 1 0 1
–4	1 1 0 0	0 1 0 0	1 1 0 0
–5	1 1 0 1	0 0 1 1	1 0 1 1
–6	1 1 1 0	0 0 1 0	1 0 1 0
–7	1 1 1 1	0 0 0 1	1 0 0 1
–8	N/A	0 0 0 0	1 0 0 0

Figure 11.7 presents the overflow behavior of a 2's-complement binary counter. The overflow is, as expected, periodic. The unique behavior of the overflow is that the difference between points in the counter (or on circle) is correct even if the counter has experienced an overflow. It is well known that intermediate overflows of a 2's-complement accumulator lead to the correct answer as long as the accumulator is wide enough to hold the correct answer.

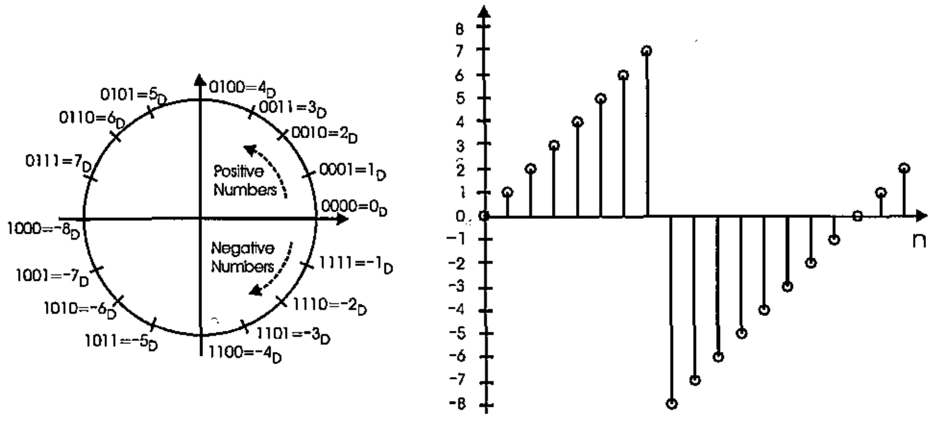


Figure 11.7 Overflow Behavior of 2's-Complement Binary Counter

As an example, let us examine the step response of a CIC with 4-units of delay in the comb filter. Here the number 1 is added to the accumulator on each clock cycle, and the comb filter following the accumulator forms the difference between the current input and the input formed 4-units ago. The sequence of outputs values is presented in Table 11-3, where we see that the output of the comb filter is correct in spite of the overflowing accumulator.

Table 11-3 Comb Filter Operating on Output of Overflowing Accumulator

w(n)	math	w(n-4)		y(n)
0	–	0	=	0
1	–	0	=	1
2	–	0	=	2
3	–	0	=	3
4	–	0	=	4
5	–	1	=	4
6	–	2	=	4
7	–	3	=	4
–8	–	4	=	–12 = 4
–7	–	5	=	–12 = 4
–6	–	6	=	–12 = 4
–5	–	–7	=	–12 = 4
–4	–	–8	=	4
–3	–	–7	=	4
–2	–	–6	=	4
–1	–	–5	=	4

If the accumulator is sufficiently wide, and if the CIC filter is performed with 2's-complement arithmetic, the CIC output will be correct in spite of the internal overflow of the integrator. By sufficiently wide we mean that the accumulator width must be the sum of the number of input bits and the number of bits required to accommodate the growth M of the M-tap prototype filter upon which the CIC is based. For instance, with 10-bit input data and a 100-tap boxcar filter with a gain of 100, we require 7-bits of growth for a bit field width of 17-bits. The accumulator must have 17 or more bits to implement the 100-tap box-car filter as a CIC filter. Bit width is addressed in more detail in a later section.

11.3 Multistage CIC

We now address the problem that the boxcar filter or its surrogate, the CIC filter, is really not a good filter. We improve performance of the boxcar by forming a filter as a cascade of multiple boxcar filters. It is common to use 3-to-5 cascade stages with many applications using 3 or 4 stages. We can examine the impulse response of a multiple-stage boxcar to gain insight into how a set of identical relatively poor performing filters work together to become a pretty good filter. The transfer function for a K-stage boxcar filter is shown in (11.15), and the corresponding frequency response is shown in (11.16). The two parameters that define this filter are M, the length of each sub filter, and K, the number of filters in the cascade.

$$H_K(Z) = \left[\frac{1-Z^{-M}}{1-Z^{-1}}\right]^K \tag{11.15}$$

$$\mathrm{H}_\mathrm{K}(\theta) = \left[\frac{\sin(\theta\frac{\mathrm{M}}{2})}{\sin(\theta\frac{1}{2})}\right]^\mathrm{K} \tag{11.16}$$

For ease of understanding, we examine a specific cascade filter. The form of a 4-stage cascade is shown in Figure 11.8. Here we see that the transfer functions from the input to the output of successive filter stages are increasing integer powers of the first stage transfer function $(1-Z^{-M})/(1-Z^{-1})$.

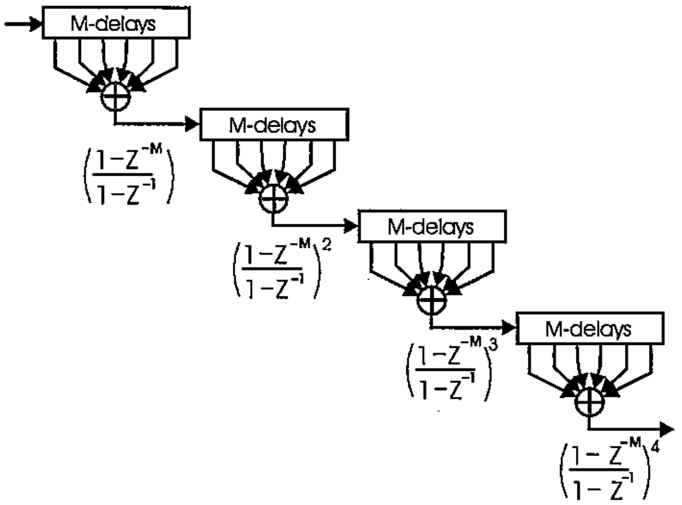

M-delays
M-delays
M-delays
M-delays
$\left(\frac{1-Z^{-M}}{1-Z^{-1}}\right)$
$\left(\frac{1-Z^{-M}}{1-Z^{-1}}\right)^2$
$\left(\frac{1-Z^{-M}}{1-Z^{-1}}\right)^3$
$\left(\frac{1-Z^{-M}}{1-Z^{-1}}\right)^4$

Figure 11.8 Cascade of Four Length-M Boxcar Filters

Figure 11.9 presents the impulse response observed at the outputs of the four successive length-10 boxcar filters. For ease of comparison, each response has been normalized to unity peak amplitude. As expected, the successive outputs are a rectangle, a triangle formed by convolving two rectangles, a piecewise quadratic formed by convolving three rectangles, and a piecewise cubic formed by convolving four rectangles. These are successively smoother sequences and are expected to exhibit successively lower side-lobe levels. Figure

11.10 presents the frequency responses of the impulse responses shown in Figure 11.8. Here too, the spectra have been normalized for unity gain. The most obvious feature of these spectra is the reduction in maximum side-lobe level as well as rate of side-lobe attenuation in successive filters. The maximum side-lobe level of the first filter is –13-dB, and since the spectra of successive filters are multiples of the first spectrum, the level of the maximum side-lobe in the successive filters is multiples of –13-dB. The second, and more important, feature of these spectra can be seen in Figure 11.11, which presents the main lobe response and a detailed look at the spectra at the first zero crossing. Here we see the effect of the repeated zeros on the ability of the filter to suppress spectra centered at normalized frequency 1/10 or 1/M for the general case. We noted the additional rejection capabilities of the higher order spectral zeros in our discussion of arbitrary interpolators.

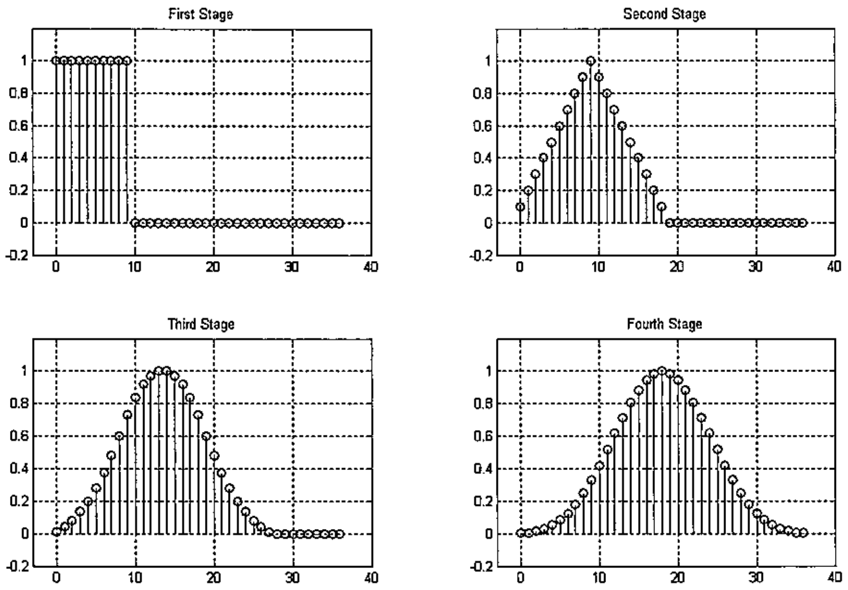


Figure 11.9 Scaled Impulse Responses of Four-stage, Length-10 Boxcar Filter

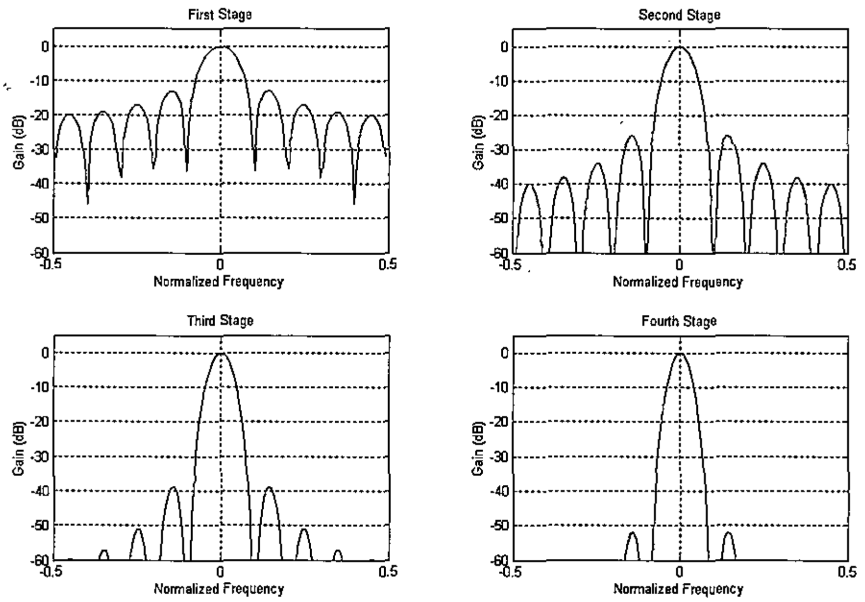


Figure 11.10 Scaled Frequency Responses of Four-stage, Length-10 Boxcar Filter

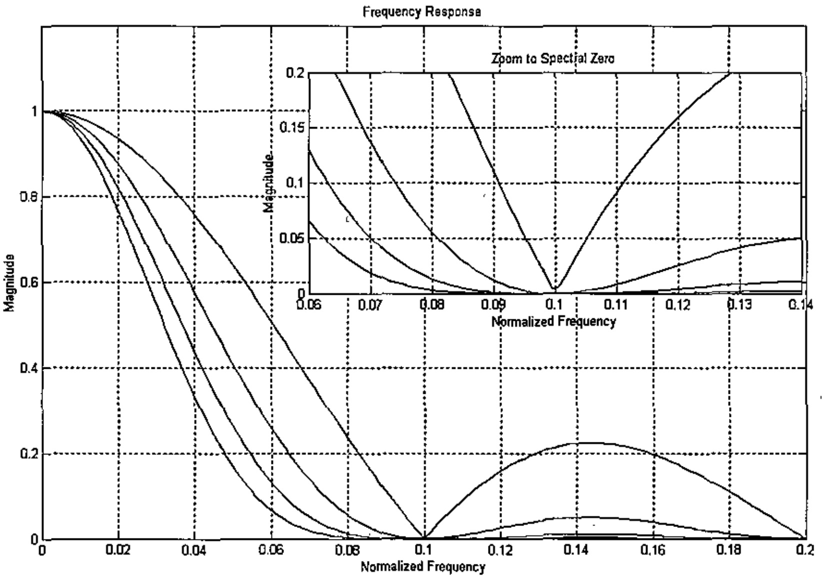


Figure 11.11 Zoom to Zero of Frequency Responses of Four-stage, Length-10 Boxcar Filter

Figure 11.12 shows how the CIC filter with two, three, and four stages is able to suppress spectral copies of a baseband signal, originally oversampled 4-to-1, that has been up sampled again by 1-to-10 zero packing. We can easily see that the higher order filter with higher order zeros at the spectral copies are better able to suppress the undesired spectral copies.

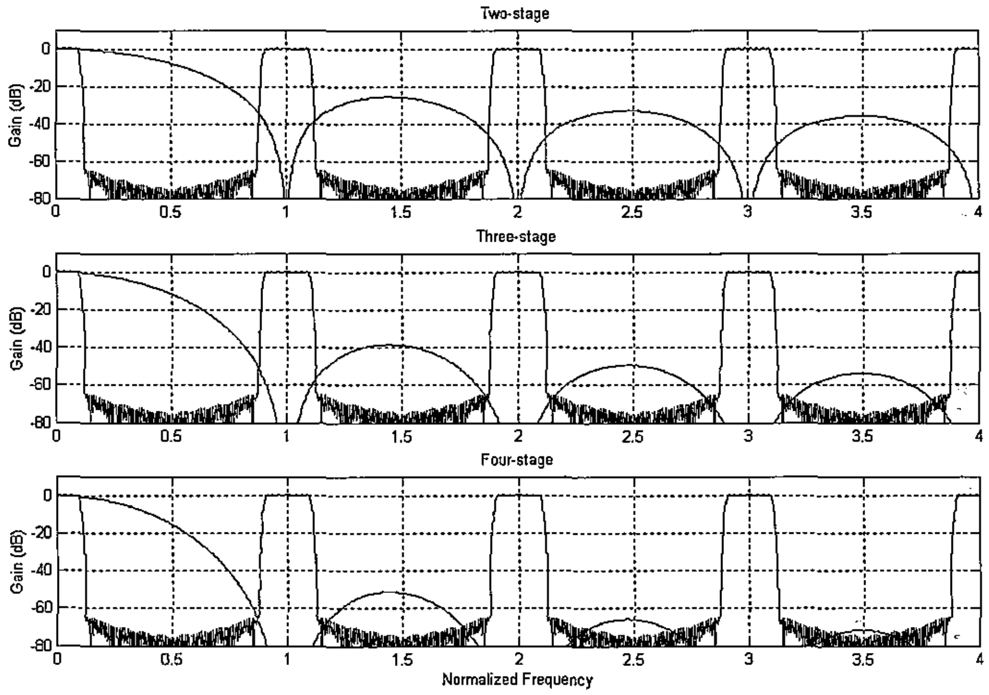


Figure 11.12 Spectral Replica Suppression by Two-, Three-, and Four-stage CIC Filter

An effect we observe in Figure 11.10 and hint at in Figure 11.11 is the frequency dependent gain reduction of the filter's main lobe response. This nonuniform pass-band gain will distort a baseband spectrum being processed by this filter. The main lobe gain reduction usually limits the input signal bandwidth to be less than 25% of the main lobe width. For this example, input bandwidth would be less than 0.025 or 1/4 of 1/10, the 1/10 being the main lobe width of the 10-tap boxcar filter. The spectral distortion due to the main lobe is corrected by embedding the inverse response in a FIR filter preceding or following the CIC. Additional comment on the correction process is postponed until we examine a system using the CIC with an embedded resampling operation.

11.4 Hogenauer Filter

The CIC filter is used in both up sampling and down sampling applications. The CIC contains two subfilters, the comb and the integrator, which can be applied in either order. If the filter is applied to an up-sampling task, the comb filter is placed at the input to the CIC filter. On the other hand, if the CIC is applied to a down-sampling task, the comb filter is placed at the output of the CIC filter. This ordering is established to permit the application of the noble identity and thus enable the reordering of the resampling switch and the comb filter. This reordering is illustrated in Figure 11.13 for the two cases of up sampling and of down sampling. Note that the comb filter becomes a differentiator at the low data rate after it is reordered with the resampling switch. When the CIC absorbs the resampling switch the filter structure is known as a Hogenauer filter. Remember that when ordering the three components in a resampling mode, the integrator always operates at the higher of the two rates, the differentiator operates at the lower rate, and the resampler resides between the pair.

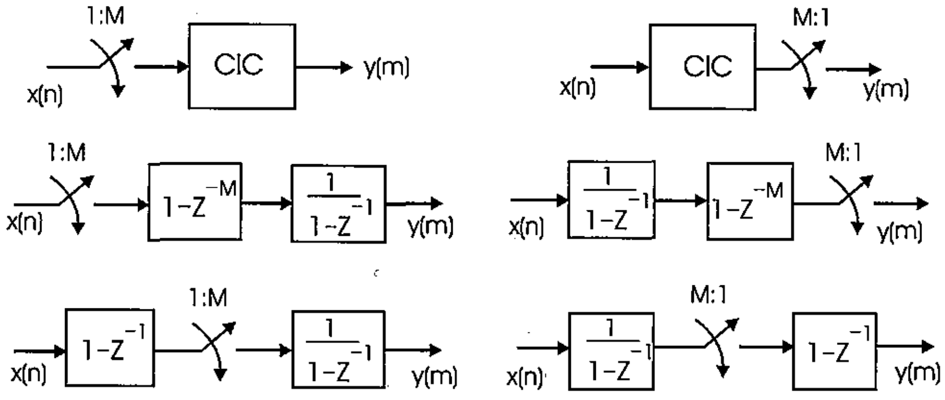


Figure 11.13 Order and Reordering of Resampling Switch and Subfilters of CIC Filter for Up Sampling and for Down Sampling

A CIC filter with any number of stages can be similarly converted to a Hogenauer filter by first ordering all the integrators on one side of the filter and the comb filters on the other side, then applying the noble identity to interchange the resampling switch and the multiple comb filters. This reordering for a three-stage down-sampling CIC filter is shown in Figure 11.14. The up-sampling version of this same process is similar except that the derivatives are at the input and the integrators are at the output.

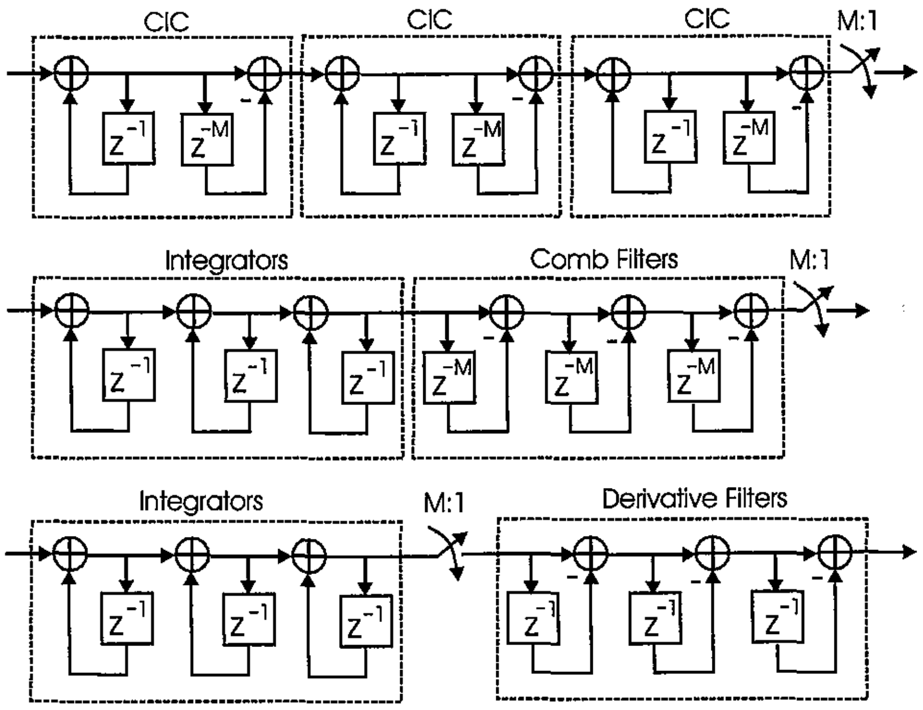


Figure 11.14 Down-sampled Three-stage CIC Filter, Rearranged and Converted to a Hogenauer Filter

11.4.1 Accumulator Bit Width

We observed earlier that in spite of overflowing accumulators the composite CIC filter would compute correct filter outputs provided the additions were performed with 2's-complement arithmetic and provided the bit field width of the accumulator exceeded the word width required by the output sequence. The required bit width is the number of bits in the input data words plus the number of bits required to accommodate the maximum filter gain. This is shown in (11.17). The gain of a K-stage filter with length-M comb filters is the product of the DC gains of each prototype boxcar filter as shown in (11.18). Substituting (11.18) in (11.17) we determine the bit width required of the CIC accumulators as indicated in (11.19).

$$b_{ACCUM} = b_{DATA} + CEIL[\log_2(GAIN)] \tag{11.17}$$

$$GAIN = M^K \tag{11.18}$$

$$b_{ACCUM} = b_{DATA} + CEIL[K \cdot \log_2(M)] \tag{11.19}$$

As an example, suppose we have a 2-stage CIC with length-20 comb filters processing 7-bit input. The gain of the filter is 20*20 or 400 for which we require 9-bits to provide for the signal growth through the filter. The required accumulator width is 9-bits of growth added to the 7-bits of input data or 16 bits. Figure 11.15 presents the time series at the outputs of a CICs two 16-bit integrators and two comb filters. The input to the CIC is a cosine of amplitude 63 with period equal to 1,000 samples. Note the overflow of the second integrator and the recovery from the overflow by the two comb filters.

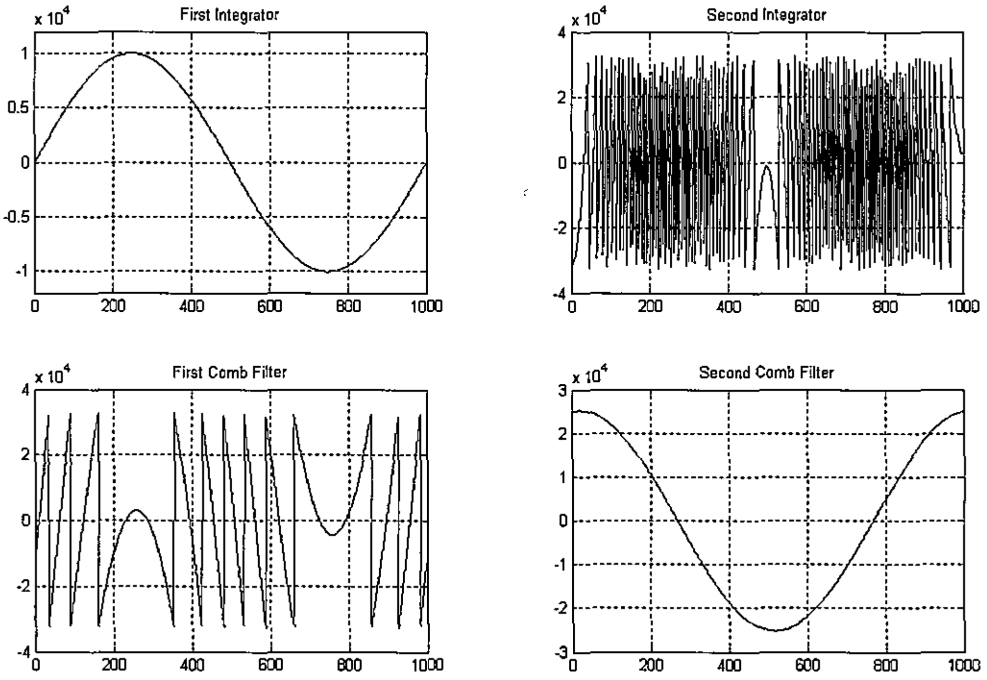


Figure 11.15 Two-stage CIC Filter, 20-tap Comb, 16-bit Accumulator, 7-bit Input

For comparison with a properly operating CIC we examine the performance of an improperly operating filter. Figure 11.16 presents the time series from a similar two-stage CIC filter operating on the same input signal except that the accumulator bit width of this CIC filter is 15-bits. As in the previous example, the second accumulator overflows, and the two comb filters attempt to reverse the overflow. We observe that the output of the second comb filter has not successfully recovered from the overflow and that there is a residual overflow of a single bit at its output. When the input signal level is reduced by a factor of two, the filter successfully recovers from the internal overflow since the reduced input level results in a reduced output level supportable by the 15-bit accumulator width.

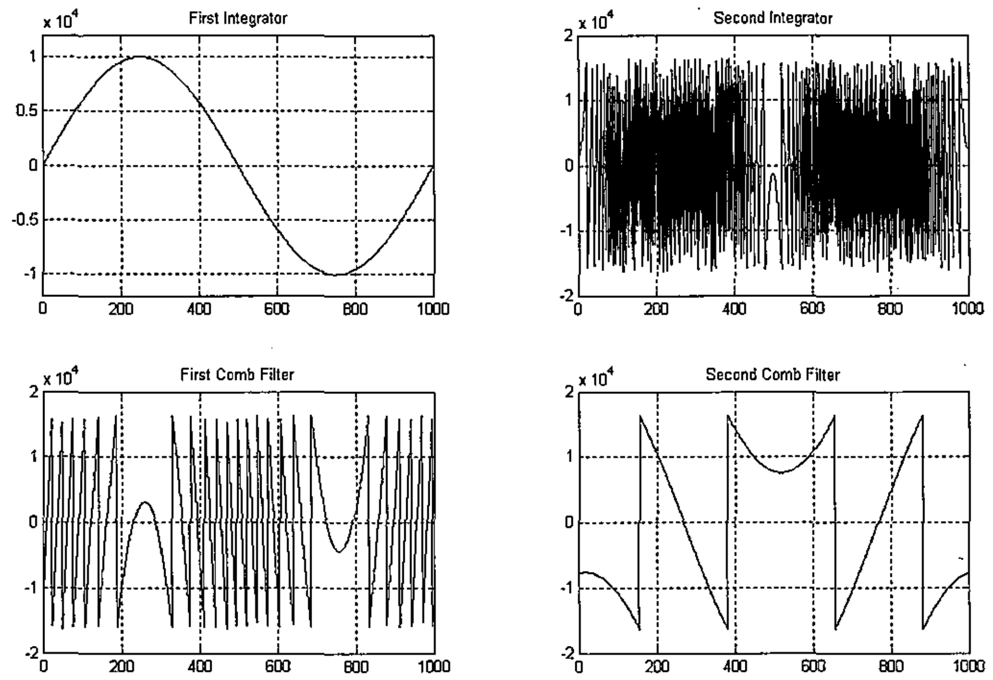


Figure 11.16 Two-stage CIC Filter, 20-tap Comb, 15-bit Accumulator, 7-bit Input

11.4.2 Pruning Accumulator Width

In the previous section we identified the bit growth required in a CIC to successfully recover from internal accumulator overflow. The bit growth reflects the filter gain between input and output of the filter. In an up-sampling filter, this growth occurs in successive integrator stages, and the integrators at the beginning of the chain do not require the same bit width as does the final integrator. We can prune the most significant bits of accumulators to the level corresponding to the maximum gain of each integrator. In a down-sampling filter the growth appears immediately in the first integrator stage, and all subsequent integrators and comb filters must honor the most significant bit of the first integrator stage. Scaling applied at the output of the CIC to remove the filter processing gain discards the lower order bits of the CIC process. We can prune lower order bits early in the filtering chain to the bit position in any stage that cannot grow beyond the least significant bit of the output word. We now examine the two cases of *most significant bit* (MSB) pruning for up-sampling CIC filters and of *least significant bit* (LSB) pruning for down-sampling CIC filters.

11.4.2.1 Up-sampling CIC

Figure 11.17 is a block diagram of a 4-stage CIC up-sampling filter. The output of each integrator in the chain is identified and is available at the indicated tap points.

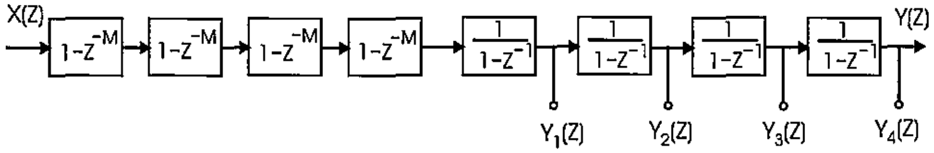


Figure 11.17 Four-stage Up-sampling CIC Filter with Integrator Outputs Identified

The transfer functions from the input to each integrator output are identified in (11.20) . Here, at each position, the included integrators are coupled with a matching comb filter with the unmatched or excess comb filters listed separately as a preprocessor to the equivalent boxcar filters.

$$\frac{Y_K(Z)}{X(Z)} = H_K(Z) = (1 - Z^{-M})^{(4-k)} \left[\frac{1 - Z^{-M}}{1 - Z^{-1}} \right]^K \tag{11.20}$$

Figure 11.18 presents the impulse response from the input to the four integrator positions when there are 20 delays in each comb filter. Note the successively larger output levels at the outputs of the successive integrators.

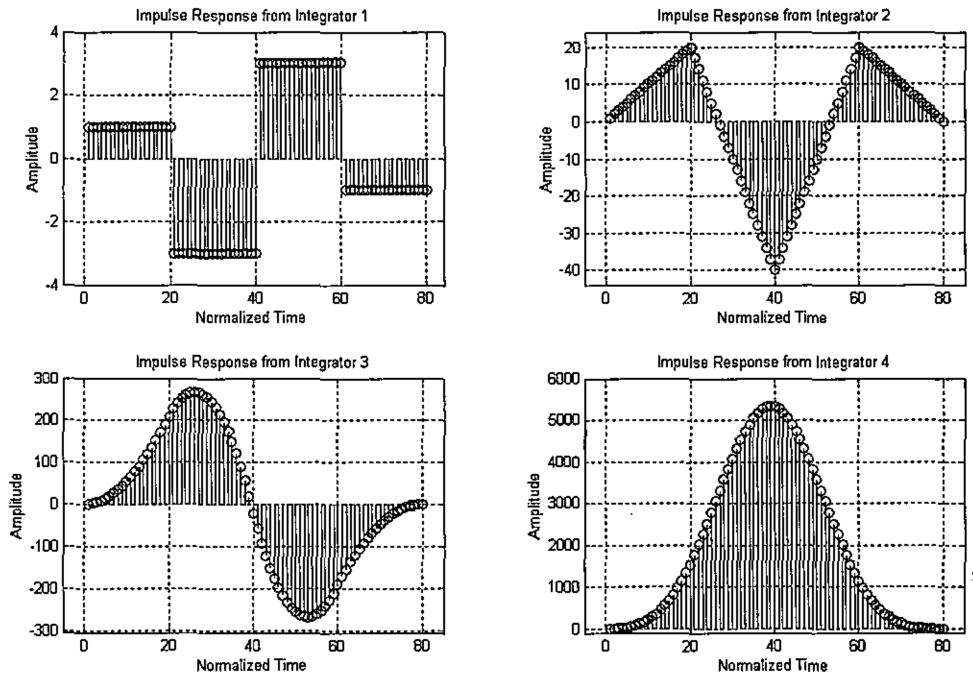


Figure 11.18 Impulse Responses at Integrators of Four-stage CIC Filter with 20-tap Comb

To determine the maximum signal gain from the input to $y_k(n)$ the output of integrator k we apply the input sequences $x_k(n)$ satisfying (11.21).

$$x_K(n) = sign[y_K(n)] \tag{11.21}$$

The sequences that maximize the output levels at each of the integrator stages are shown in Figure 11.19.

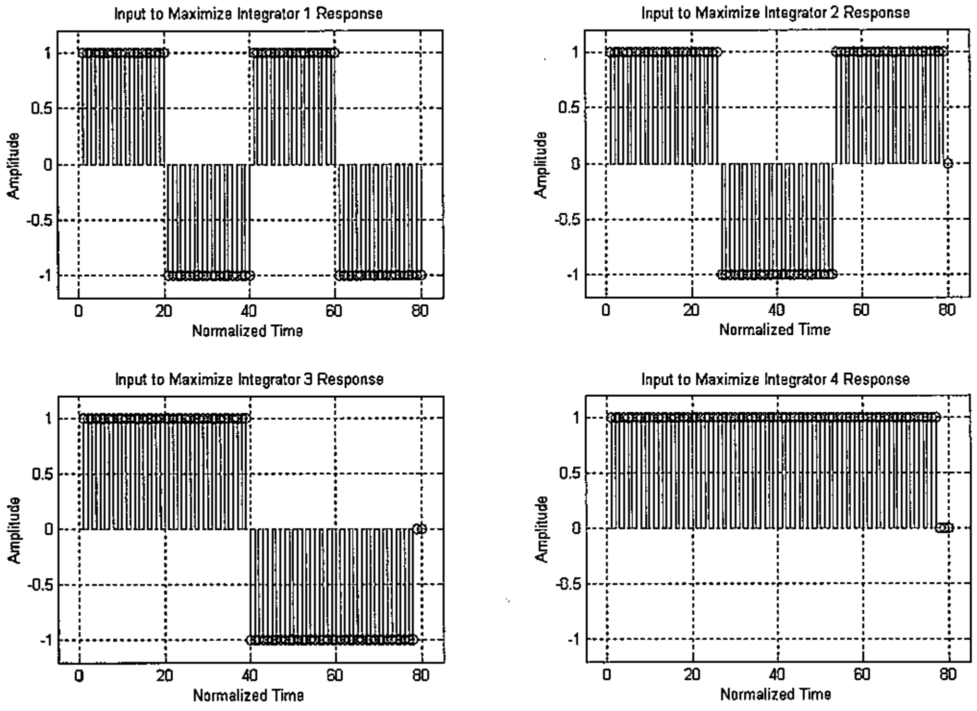


Figure 11.19 Input Sequences to Maximize Output from Integrators in Four-Stage CIC

The sequence that maximizes the output at a selected integrator lead to nonmaximum output levels at the remaining integrators. This can be seen in Table 11-4 which lists the maximum level obtained at each integrator for the four input sequences identified in Figure 11.19. The bold entries in the table correspond to the maximum level that integrator can exhibit for sequences that result in stable responses at the final output stage. Figure 11.20 presents the integrator responses for the input sequences shown in Figure 11.19 designed to maximize the amplitude response for the selected integrator.

Table 11-4 Maximum Integrator Response Levels for Sequences that Maximize Selected Integrator Response

	Sequence Int-1	Sequence Int-2	Sequence Int-3	Sequence Int-4
Integrator 1	**–160**	109	80	40
Integrator 2	858	**–1068**	640	267
Integrator 3	–6,821	9,848	**–10,680**	5,340
Integrator 4	73,725	113,496	141,626	**160,000**

Each comb filter in the processing sequence, with one addition per stage, has a gain of 2 with a cumulative gain at the *k*th stage of 2^k. The impulse weights of a comb filter cascade are the terms of Pascal's triangle with alternating signs. These terms are shown in Figure 11.21. To maximize the sum at the output of each comb filter, as we did for the integrators, we set the input sequence to the sign of the comb filter impulse response. To also probe the response of the integrator chain while probing the comb filters, we extend the samples for the M-sample interval between samples of the comb filter response. The probe signals to test both the comb and the integrator segments of the CIC are shown in Figure 11.22. We note that the successive sequences use the previous sequence as a prefix and simply add one more segment. Consequently, we will find that the sequence that maximizes the output for comb filter k also maximizes the output for comb filter k–1.

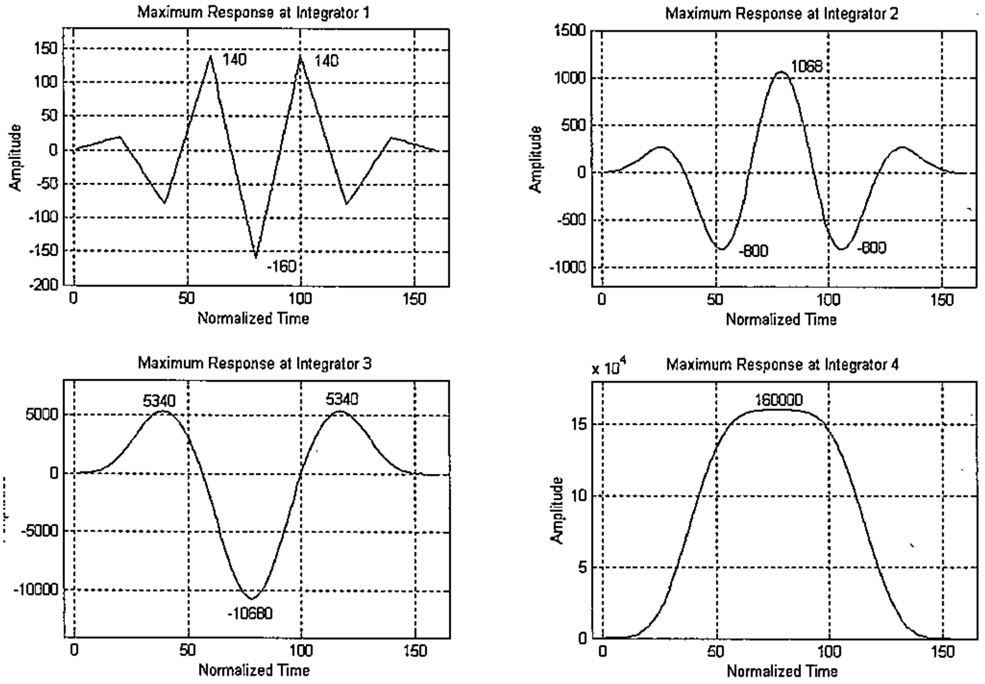


Figure 11.20 Integrator Responses for Input Sequence Selected to Maximize Output Level

Table 11-5 lists the maximum level obtained at each comb filter and at each integrator output for the four input sequences identified in Figure 11.21. The bold entries in the table correspond to the predicted maximum levels that comb filter can exhibit; we also see the peak values of the integrators obtained while maximizing the output of the comb filters. For ease of comparison, we also list from Table 11-4, the maximum values of the integrator outputs.

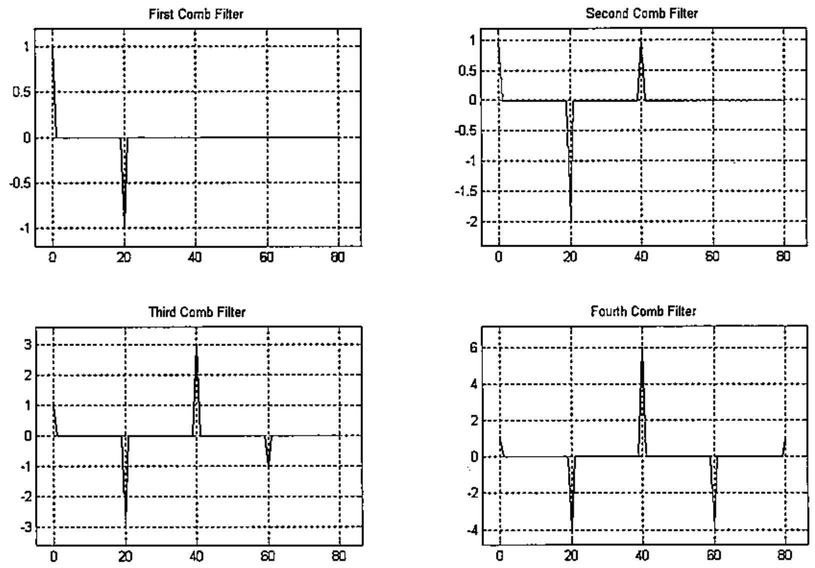


Figure 11.21 Impulse Responses at Comb Filters of Four-stage CIC Filter with 20-tap Comb

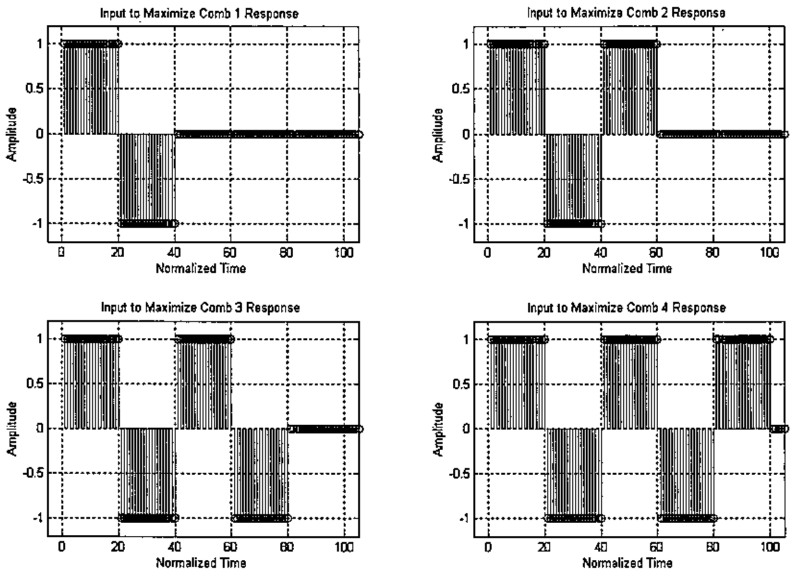


Figure 11.22 Input Sequences to Maximize Output from Comb Filters in Four-stage CIC

Table 11-5 Maximum Comb Response Levels for Sequences that Maximize Selected Comb Response

	Sequence Comb-1	Sequence Comb-2	Sequence Comb-3	Sequence Comb-4	Maximum Level
Comb-1	**2**	**2**	**2**	**2**	**2**
Comb-2	3	**4**	4	4	**4**
Comb-3	6	7	**8**	**8**	**8**
Comb-4	10	14	15	**16**	**16**
Integrator-1	120	140	−160	160	**160**
Integrator-2	720	900	855	855	**1068**
Integrator-3	−8,020	6,821	−6,821	6,821	**10,680**
Integrator-4	73,710	73,725	73,725	73,725	**160,000**

The bit-width results determined from the two sets of probes applied to the up-sampling CIC filter of Figure 11.17 are tabulated in Table 11-6.

Table 11-6 Maximum Comb and Integrator Response and Required Bit Field Width

	Maximum Gain	Growth Bits
Comb-1	2	1
Comb-2	4	2
Comb-3	8	3
Comb-4	16	4
Integrator-1	160	9
Integrator-2	1068	10
Integrator-3	10,680	14
Integrator-4	160,000	18

The bit growth per processing stage can be visualized by the graph of bit-field width shown in Figure 11.23. Each stripe in the graph indicates the number of bits entering and the number of bits leaving the indicated process. For the example just concluded we can see how the total of 18-bit growth from input to output is distributed between the comb filters and the integrators.

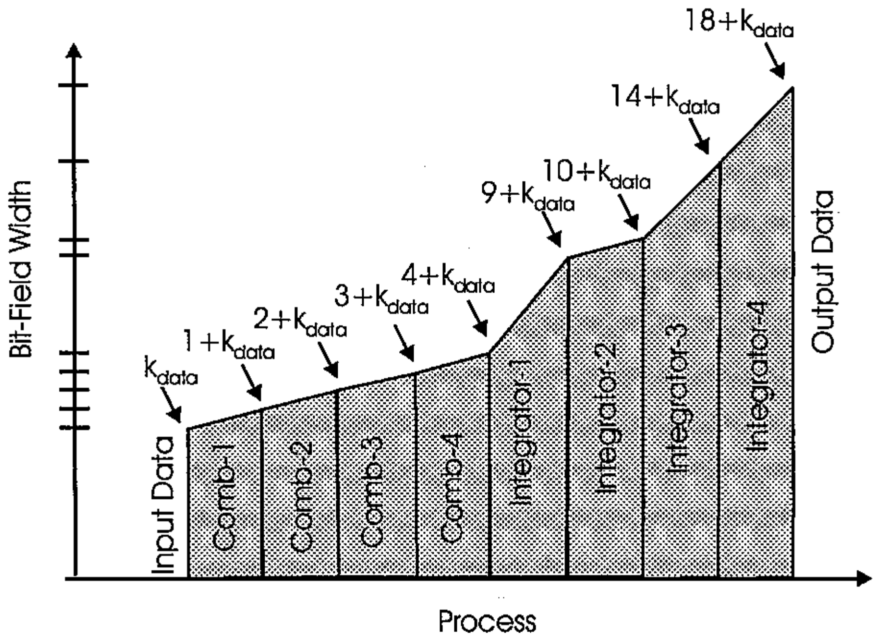


Figure 11.23 Graphs Showing Required Bit Field at Input and Output of Each Process in CIC

11.4.2.2 Down-Sampling CIC

Figure 11.24 is a block diagram of a 4-stage CIC down-sampling filter. Pruning of this process will by performed be discarding lower-order bits in each accumulator. The discarded bits will be treated as additive noise to each integrator. Our interest here is the noise gain from each integrator and from each comb filter to the output port. The input of each integrator in the chain is identified and is available at the indicated input points from which we will determine the noise gains.

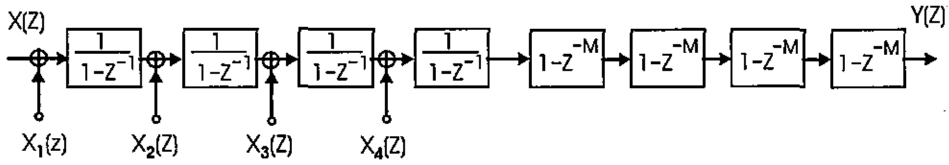


Figure 11.24 Four-stage Down-sampling CIC Filter with Integrator Inputs Identified

The transfer functions from the input of each integrator to the CIC output are identical to the transfer functions identified in (11.20) and repeated here as (11.22) to reflect the change of input and output variables. Here, at each position, the integrators included in the signal path are coupled with a matching comb filter, with the unmatched or excess comb filters listed separately as post processor to the equivalent boxcar filters.

$$\frac{Y(Z)}{X_K(Z)} = G_K(Z) = \left[\frac{1-Z^{-M}}{1-Z^{-1}}\right]^K (1-Z^{-M})^{(4-K)} \qquad (11.22)$$

Figure 11.25 presents the impulse response from the four-integrator input positions to the output when there are 20 delays in each comb filter. Note the successively reduced output levels at the output when there are a reduced number of integrators between the selected input and the CIC output.

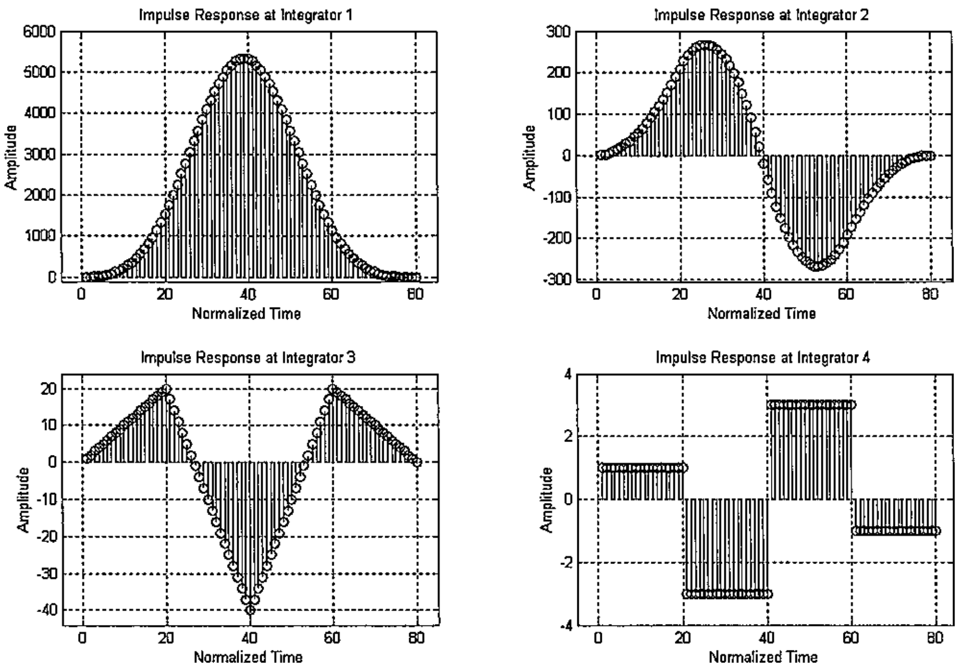


Figure 11.25 Impulse Responses from Integrators of 4-Stage CIC Filter with 20-Tap Comb

The noise power measured at the output due to noise inserted at the input to integrator k is shown in (11.23).

$$\sigma^2_{OUT_K} = \sigma^2_K \sum_{n=0}^{N-1} g^2_K(n) = \sigma^2_K NG_K \tag{11.23}$$

The standard deviation of the output noise due to the gain from each integrator is listed in Table 11-7. In a similar manner, noise inserted at the input to the comb filters can be formed using the comb filter impulse response. The comb filter impulse responses are the same as those shown in Figure 11.21. Table 11-7 shows the noise gain for each of the integrator and comb filter stages in the CIC.

Table 11-7 Noise Gain from Each CIC Stage to Output

	SQRT(Noise Gain)	Noise Bit Growth
Integrator-1	24,785	14.6
Integrator-2	1,462.4	10.5
Integrator-3	146.3	7.2
Integrator-4	20	4.3
Comb-1	8.4	3.1
Comb-2	4.5	2.2
Comb-3	2.5	1.3
Comb-4	1.4	0.5

The total noise contributed to the output by the separate noise sources injected at each stage due to pruning the *least significant bits* (LSB) in each of the 2K filter segment is shown in (11.24).

$$\sigma_{TOTAL}^2 = \sum_{K=0}^{2K-1} \sigma_{OUT_K}^2 = \sum_{K=0}^{2K-1} \sigma_K^2 NG_K \tag{11.24}$$

It is reasonable to assign equal levels of noise contribution from each noise source. For this condition, the noise contributed by each source must satisfy (11.25).

$$\sigma_{OUT_K}^2 = \frac{1}{2K}\sigma_{TOTAL}^2 \tag{11.25}$$

The acceptable noise level that can be inserted at each noise source must account for the 1/2K of (11.25). Since the least significant bit level defines the additive noise level at each insertion point, we convert the 1/2K to bits as is done in (11.26).

$$\text{Additional Number Bits} = \log_2(2K) \tag{11.26}$$

Our 4-stage example, with 8-noise terms, requires the noise terms to drop an additional 3-bits for the overall noise to be below the specified output LSB. When we incorporate the additional attenuation of the scaled noise in our example, we obtain the parameters listed in Table 11-8.

Table 11-8 Bit Locations Below Output LSB for Register Pruning

	Bits Below Output LSB	Integer Number of Bits
Integrator-1	$14.6 + 3 = 17.6$	18
Integrator-2	$10.5 + 3 = 13.5$	14
Integrator-3	$7.2 + 3 = 10.2$	11
Integrator-4	$4.3 + 3 = 7.3$	8
Comb-1	$3.1 + 3 = 6.1$	7
Comb-2	$2.2 + 3 = 5.2$	6
Comb-3	$1.3 + 3 = 4.3$	5
Comb-4	$0.5 + 3 = 3.5$	4

The LSB bit pruning per processing stage can be visualized by the graph of bit-field width shown in Figure 11.26. Each stripe in the graph indicates the number of bits entering and the number of bits leaving the indicated process. For the example just concluded we can see how the total of 18-bit growth occurring at the input stage permits the LSB pruning as we progress from input to output through the integrators and the comb filters.

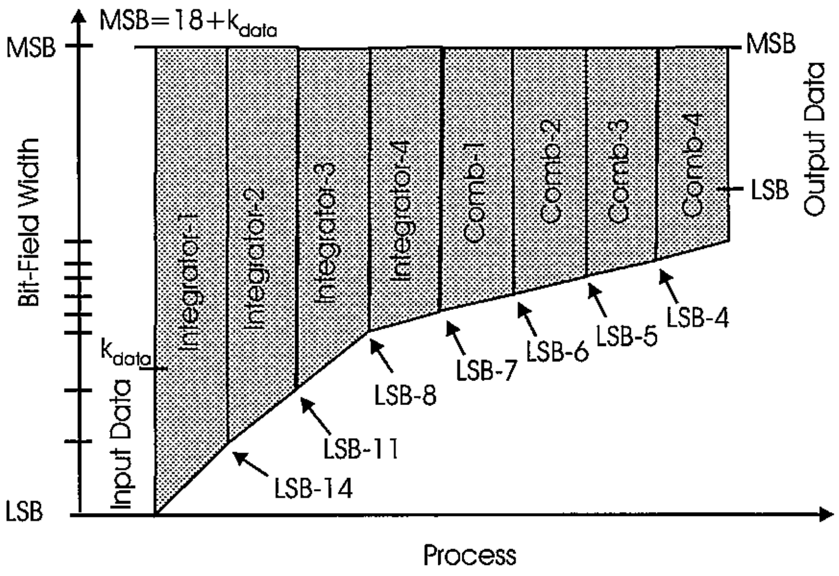


Figure 11.26 Graph Showing Required Bit Field at Input and Output of Each Process in CIC

11.5 CIC Interpolator Example

We now examine an example of the use of the Hogenauer (resampled CIC) filter. The example is part of a modulator that shapes and up samples in a polyphase filter and then interpolates to a higher output sample rate with a CIC filter. Figure 11.27 presents an example in which the shaping is performed by a 1-to-5 polyphase filter and the additional interpolation is performed in a 3-stage 1-to-16 CIC filter. The CIC is implemented as a Hogenauer filter with three resampled comb filters at the input, a 1-to-16 resampler, and three integrators at the output.

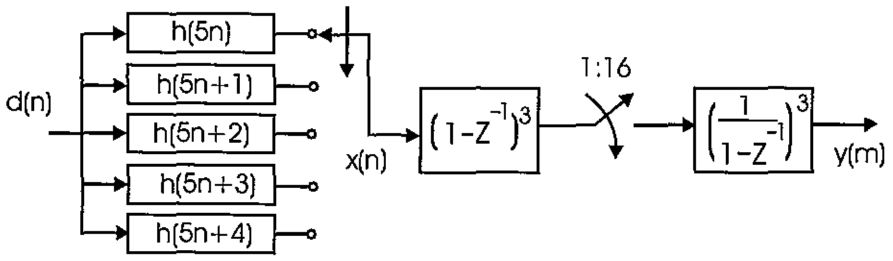


Figure 11.27 Cascade of 1-to-5 Shaping Filter and 1-to-16 CIC Interpolating Filter

Figure 11.28 presents the time and frequency response of the output series obtained by applying an impulse to the input of the shaping filter. The time axis is normalized to input symbol rate of unity, and the frequency axis is normalized to unit 3-dB bandwidth or unity symbol bandwidth.

We gain insight into the operation of the CIC filter by tracking the impulse response of the shaping filter through the CIC filter. By this process, we obtain the impulse response of the composite shaping and interpolating filter. Figure 11.29 follows the response from the shaping filter through the three input comb filters at the input rate and then follows the 1-to-16 up-sampled sequence through the three output integrators. We can clearly see the equivalent *zero order hold* (ZOH) effect of the 1-to-16 up-sampling operation at the output of the first integrator.

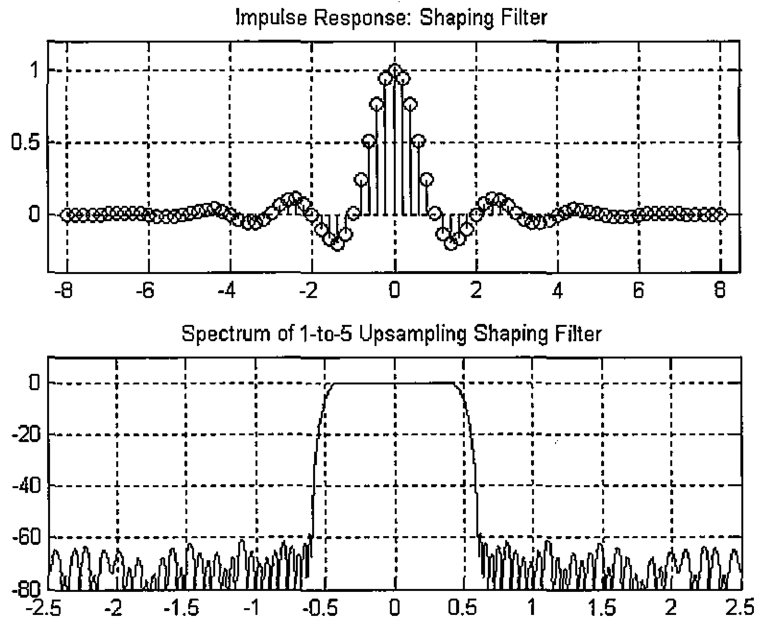


Figure 11.28 Time and Frequency Response of Time Series from 1-to-5 Shaping Filter

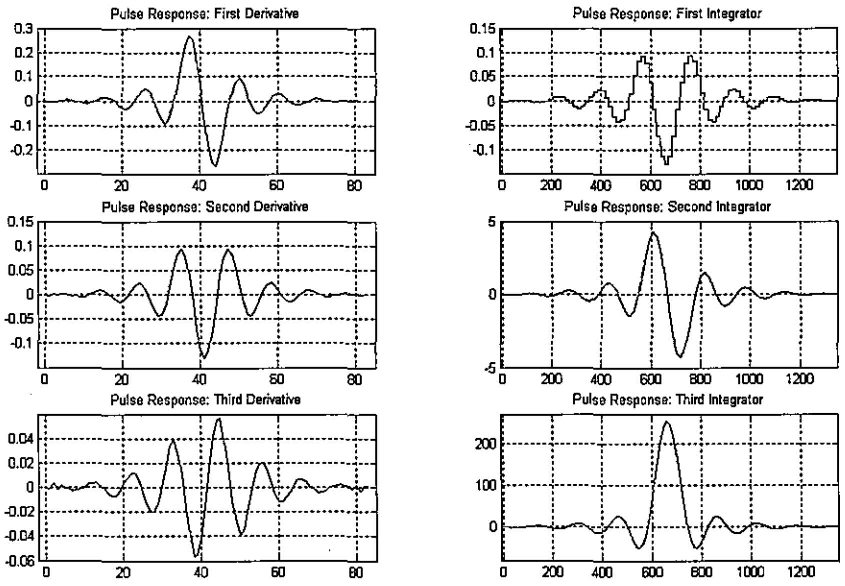


Figure 11.29 Shaping Pulse Response of Three-stage 1-to-16 Hogenauer Filter

Figure 11.30 presents the frequency response obtained from the output of the composite polyphase shaping filter and 3-stage CIC interpolating filter. We can clearly see the effect of the CIC spectral zeros on the up-sampled input time series. The maximum residual spectral level brackets the first spectral zeros of the CIC filter. This level is seen to be –60-dB, which is the target level of 60-dB attenuation. These spectral levels fall below the noise floor of a 10-bit *digital to analog converter* (DAC) and will be concealed by the system noise. An interesting note is that if the shaping filter were to operate as a 1-to-4 up-sampling filter, the 3-stage CIC would not obtain the desired 60-dB attenuation. Thus the choice is to replace the 3-stage CIC with a 4-stage CIC or to raise the output rate of the shaping filter from 1-to-4 to 1-to-5. This second option is more cost effective since the addition of a fourth CIC stage would not only add another integrator but would also increase the bi- field width of the other three integrators.

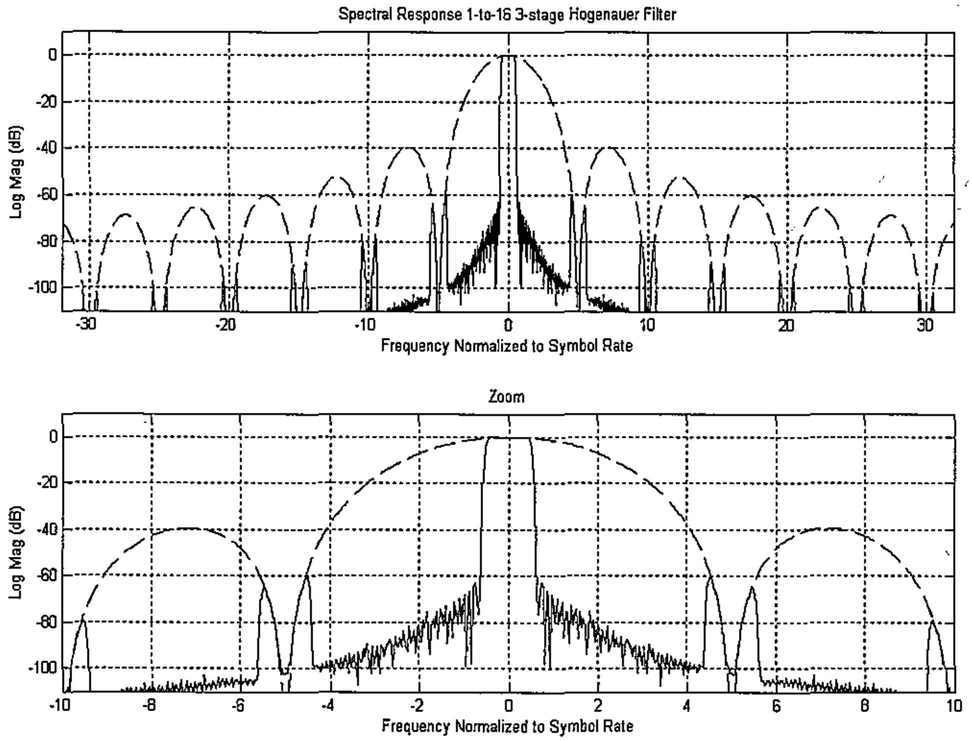
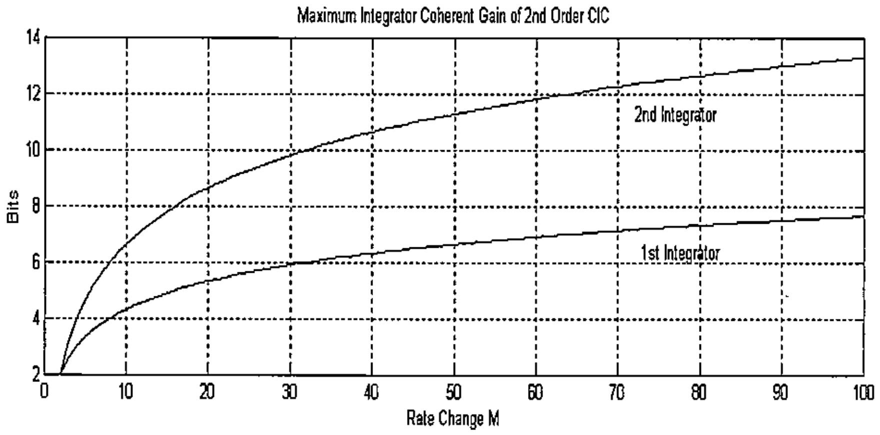


Figure 11.30 Spectrum of Composite Impulse Response with CIC Spectral Overlay

11.6 COHERENT AND INCOHERENT GAIN IN CIC INTEGRATORS

Figures 11.31a and b present curves showing the maximum coherent gain between input and integrators of up-sampling CIC filters of order 2 through 5 for up-sampling rates from 2 through 100. These curves were generated in a manner similar to the process described in the up sampling CIC example that examined the specific case of up sampling by 20 with a 4th order CIC. The input sequences that probed the CIC are of the form shown in Figure 11.18. These sequences maximize the peak amplitude at each integrator. The curves contain the information required to determine the required bit width for each integrator in an up-sampling CIC operating at a specific resampling rate.

Figures 11.32a and b present curves showing the maximum incoherent gain between integrators and output of down-sampling CIC filters of order 2 through 5 for down-sampling rates from 2 through 100. These curves were generated in a manner similar to the process described in the down sampling CIC example that examined the specific case of down sampling by 20 with a 4th order CIC. A single impulse at each integrator probed the CIC to form the impulse response from which the sum of squares was computed to determine the incoherent gain. The curves contain the information required to determine the position in a bit field that can be pruned to reduce the bit width for successive integrators in a down sampling CIC operating at a specific resampling rate. Figures 11.33a and b present the same curves shown in Figures 11.28 and 11.29 for a wider range of up- and down-sampling rates from 2 to 1000.



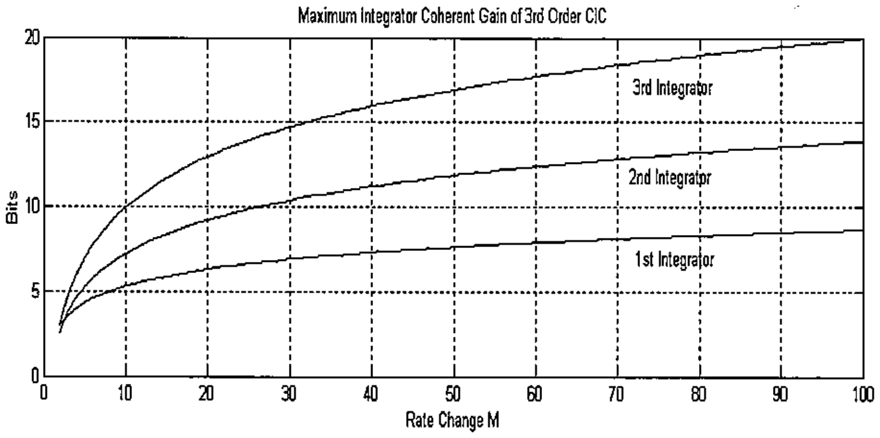
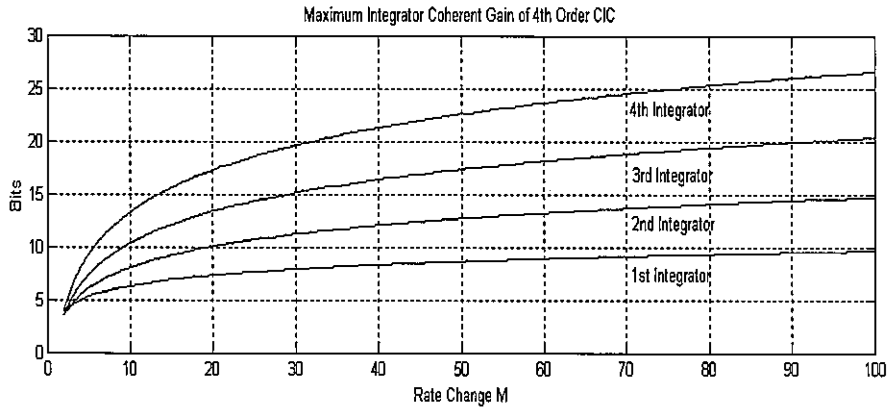


Figure 11.31a Coherent Bit Growth of Integrators of 2nd and 3rd Order CIC Filters as Function of Rate Change M



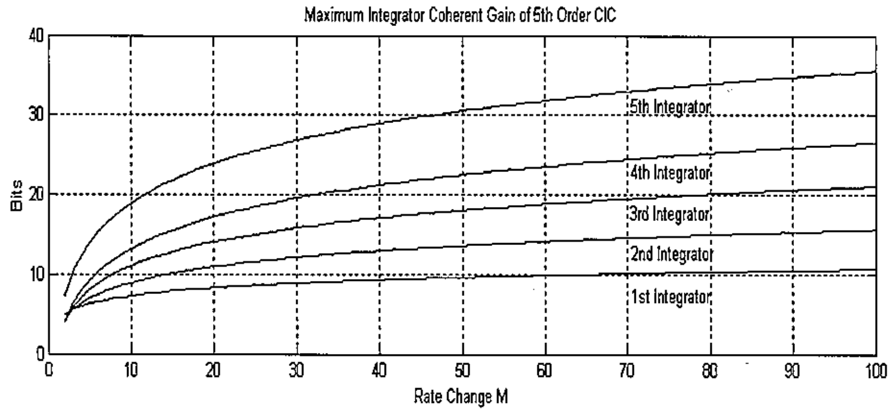


Figure 11.31b Coherent Bit Growth of Integrators of 4th and 5th Order CIC Filters as Function of Rate Change M

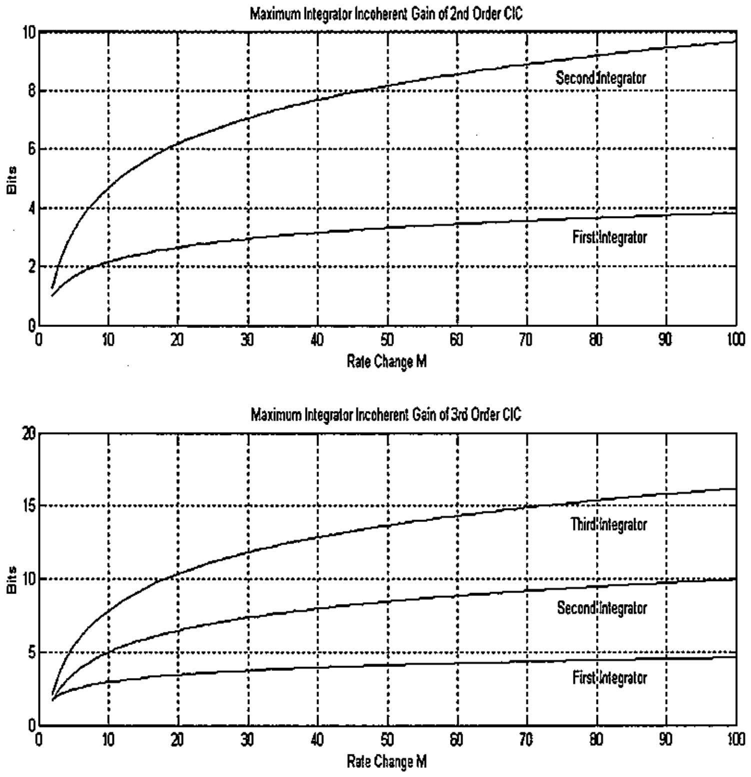
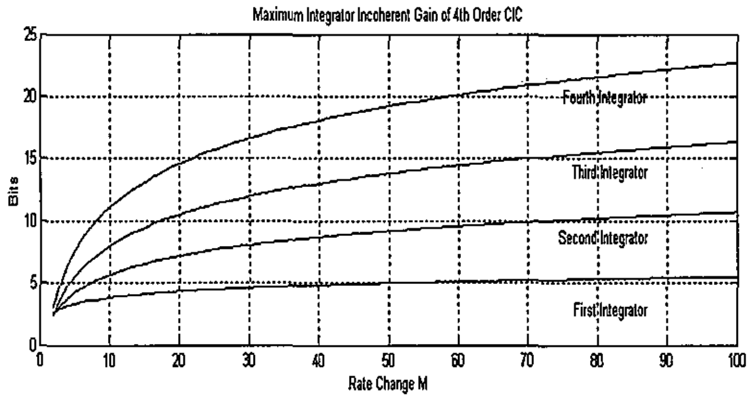


Figure 11.32a Incoherent Bit Growth of Integrators of 2nd and 3rd Order CIC Filters as Function of Rate Change M



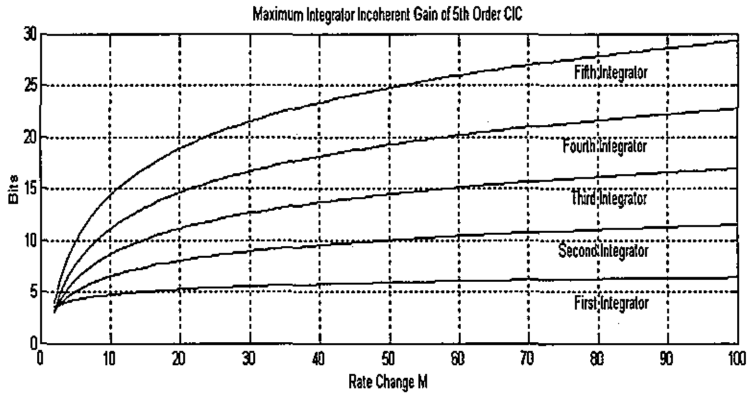


Figure 11.32b Incoherent Bit Growth of Integrators of 4th and 5th Order CIC Filters as Function of Rate Change M

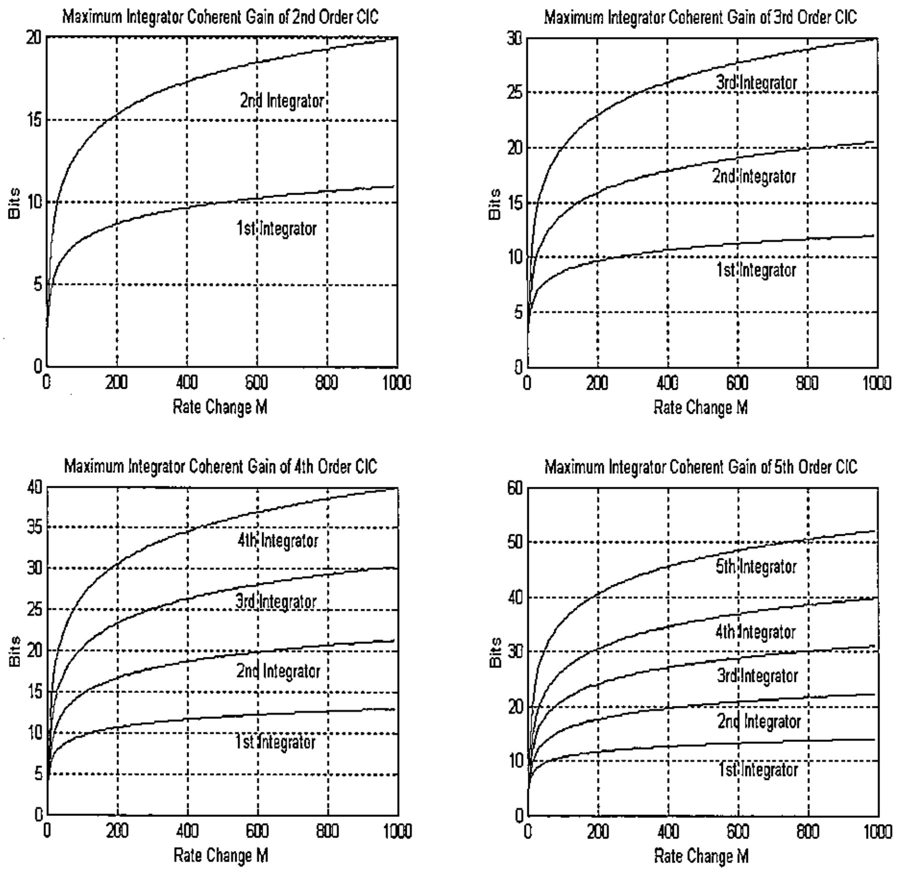


Figure 11.33a Coherent Bit Growth of Integrators of 2nd, 3rd, 4th, and 5th Order CIC Filters as Function of Rate Change M

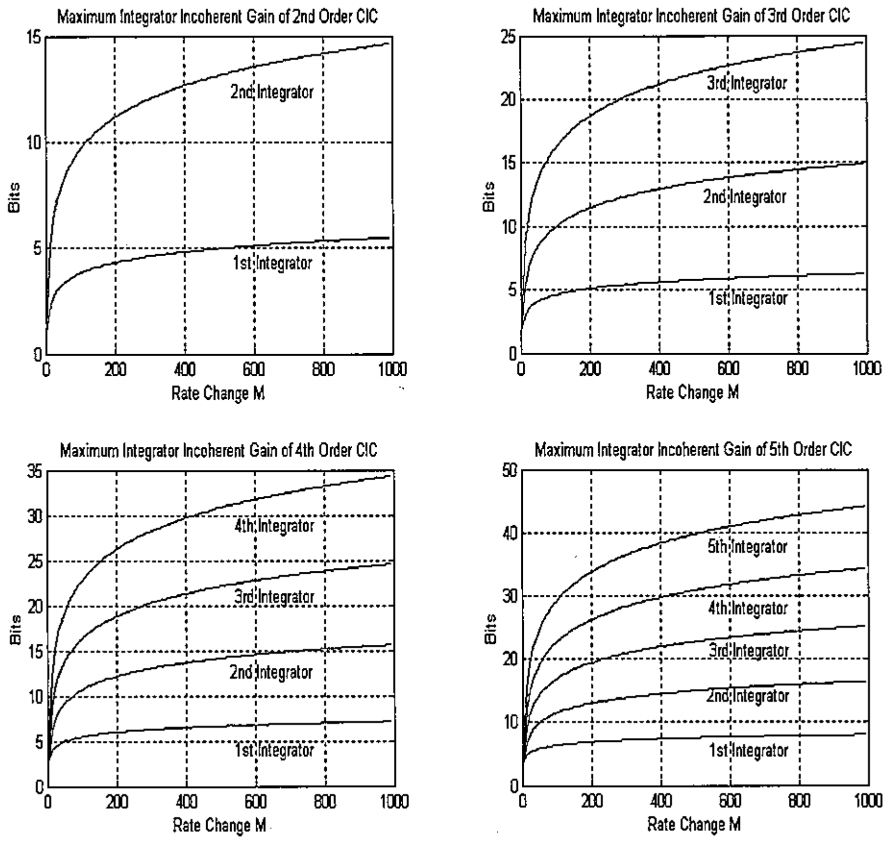


Figure 11.33b Incoherent Bit Growth of Integrators of 2nd, 3rd, 4th, and 5th Order CIC Filters as Function of Rate Change M

References

Chu, S. and Burrus, C. S., "Multirate Filter Design Using Comb Filters," *IEEE Trans. on Circuits and Systems*, Vol. 31, Nov. 1984, pp. 913 - 924.

Crochiere, Ronald and Lawrence Rabiner, "Multirate Signal Processing," Englewood Cliffs, NJ, Prentice-Hall, Inc., 1983.

Fliege, Norbert, *Multirate Digital Signal Processing: Multirate Systems, Filter Banks, Wavelets*, West Sussex, John Wiley & Sons, Ltd., 1994.

Hentschel, Tim, *Sample Rate Conversion in Software Configurable Radios*, Norwood, MA, Artech House, Inc., 2002

Hogenauer, E. B., "An Economical Class of Digital Filters for Decimation and Interpolation," *IEEE Trans. Acoustics. Speech Signal Proc.*, Vol. ASSP-29, April 1981, pp. 155 - 162

Jovanovic-Dolecek, Gordana, *Multirate Systems: Design and Applications*, London, Idea Group, 2002.

Mitra, Sanjit, *Digital Signal Processing: A Computer-Based Approach*, 2nd ed., New York, McGraw-Hill, 2001.

Mitra Sanjit and James Kaiser, *Handbook for Digital Signal Processing*, New York, John Wiley & Sons, 1993.

Vaidyanathan, P. P., "Multirate Systems and Filter Banks," Englewood Cliffs, NJ, Prentice-Hall, Inc., 1993.

Problems

11.1 Program a 20-tap version of the three forms of the boxcar integrator shown in Figure 11.5 and determine the impulse response of all three versions. Note the state of the integrator in the two forms of the CIC.

11.2 Program a 20-tap version of the three forms of the boxcar integrator shown in Figure 11.5 and determine the step response of all three versions. Note the state of the integrator in the two forms of the CIC.

11.3 Program a 20-tap version of the three forms of the boxcar integrator shown in Figure 11.5 using integer arithmetic with a 5-bit 2's-compliment accumulator and with a 4-bit 2's-compliment accumulator, and then determine the step response of all three versions. Note the state of the integrator in the two forms of the CIC for the two different width accumulators.

11.4 The spectrum of a P-stage CIC filter exhibits multiple zeros at multiples of 1/Mth of the sample rate. Determine the attenuation available at an offset of 1/(4M) from the zero at 1/M; i.e., at (3/4M). This is the frequency that aliases into the bandwidth of 1/(4M), the bandwidth of the final cascade of the CIC, and the 4-to-1 FIR filter following the CIC filter. In particular, how many stages of CIC are required to obtain 60-dB, 80-dB, 100-dB, and 120-dB attenuation at the edge of the first Nyquist zone to alias to baseband?

11.5 A P-stage CIC filter of length M has a steady state DC gain of M^P. For an M = 100 and P = 5, determine the width required of the accumulators when the input data is 16 bits. Repeat for M = 1,000.

11.6 For the block diagram in Figure 11.17, imagine an input signal consisting of samples of a unity amplitude sine wave of normalized frequency 0.1. Determine the amplitude of the sinusoid observed at the 4 integrator output ports as a function of M. In particular, what is the set of amplitudes for M = 100? Repeat for normalized frequency of 0.01.

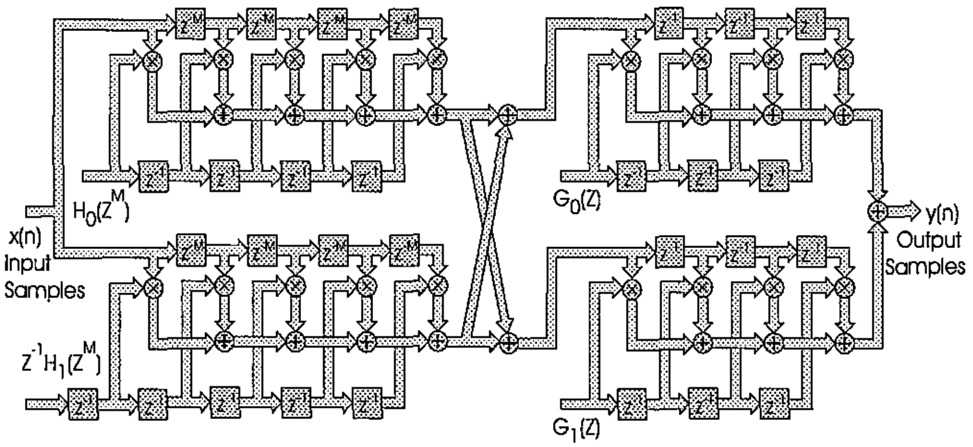
11.7 For the block diagram in Figure 11.24, imagine injecting an input signal consisting of samples of a unity amplitude sine wave of normalized frequency 0.1 in any of the 4 input ports of the integrator train. Determine the amplitude of the sinusoid observed at the output port as a function of M. In particular, what is the set of amplitudes for M = 100? Repeat for normalized frequency of 0.01.

11.8 Program the filter chain consisting of a 1-to-4 polyphase Nyquist filter followed by a 4-stage CIC filter that will up sample the filter output by a factor of 20. The Nyquist filter has a 50% excess bandwidth and a side-lobe level 60-dB below pass band response. Apply an impulse to the input and record and plot the time response of the filter chain at each stage as well as the frequency response at the final output. Repeat this programming task but replace the Nyquist filter with a 1-to-5 polyphase filter and a 4-stage CIC filter that will up sample by a factor of 16. What is the peak output amplitude of the final output for the two filter implementations? Can either implementation be realized with a 3-stage CIC? If so, which one, and what is the peak output amplitude for that case?

11.9 A CIC filter of length M does not have to be down sampled by M-to-1. An interesting option is to resample by M/2. When we operate the filter in this manner, the folding frequency for the filter output is fs/M rather than fs/(2M). Program a 3rd-order CIC filter of length 2M operating at an M-to-1 down sample rate. In this test form a 1-to-4 polyphase Nyquist filter with 50% excess bandwidth and 60-dB side-lobes. Zero pack this signal 1-to-20 and process it with a 3rd-order CIC filter of length 40. Examine the frequency response of the filter. Compare the response to a 1-to-20 zero-packed signal processed with a 3rd-order CIC of length 20. Also examine the response of 1-to-20 zero-packed signal processed with a 2nd-order CIC filter of length 40. Comment on the performance differences and the workload difference of the three options.

CHAPTER 12

Cascade and Multiple Stage Filter Structures



Often a cascade of two or more stages of filtering and resampling best serves a multirate signal-processing task. Various structures that emphasize this perspective have been developed. This chapter presents a number of systems in which cascade filtering is applied to the signal to obtain multiresolution partition of a signal or to obtain significant reduction in signal processing workload.

12.1 INTERPOLATED FIR (IFIR) FILTERS

The major emphasis of this text has been the application of resampling techniques to time series. We now switch perspectives and apply the resampling procedure to the filter processing the time series. We know that the length of a FIR filter is proportional to the ratio of the filter sample rate to the filter's transition bandwidth. In many applications, transition bandwidth is small because the filter bandwidth is small, so that narrow transition bandwidth is often synonymous with narrow pass band bandwidth. Thus we find that for many filters operating with a large ratio of sample rate to pass band bandwidth the filter length is prohibitively large. In other chapters, we addressed this problem by preprocessing the input time series with a resampling filter in which the signal bandwidth and the sample rate of the output time series is simultaneously reduced. The bandwidth-limiting filter, now designed to operate at the reduced sample rate, is shorter in proportion to the sample rate reduction. If it is necessary to preserve the original input sample rate, the output of the bandwidth-limiting filter is then up sampled back to the original input sample rate. This cascade structure is shown in Figure 12.1 where $G(Z^M)$ performs the two resampling operations while $H(Z)$ performs the desired bandwidth limiting function.

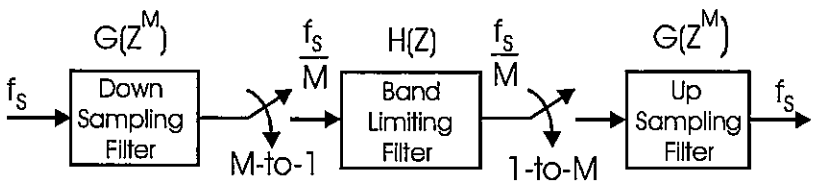


Figure 12.1 Resampling Filtering Option When Ratio of Sample Rate to Transition Bandwidth Is Large

An alternate to down sampling the input time series to the sample rate of the bandwidth-limiting filter is to raise the sample rate of the bandwidth-limiting filter. The filter is up sampled by zero packing its impulse response 1-to-M. The up-sampled filter now has the longer impulse response expected for operation at the higher sample rate but does not require any additional arithmetic operations. It does, however, require additional memory to hold the larger set of input data samples. The Z-transform of the up-sampled filter is $H(Z^M)$.

The frequency response of the up-sampled filter has M spectral replicates at multiples of the baseband design sample rate of fs/M. A second filter, following the zero-packed filter, preserves the baseband spectra while filtering out the multiple spectral replicates. This filter structure is shown in Figure 12.2, while the frequency responses observable at various positions in the cascade are shown in Figure 12.3.

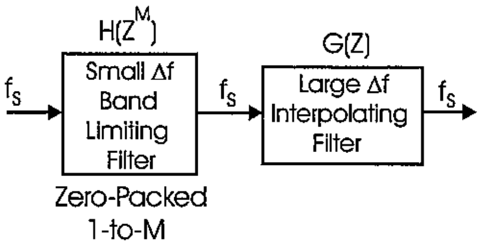


Figure 12.2 Interpolated FIR (IFIR) Filter: Up-sampled Prototype and Interpolating Clean-up Filter

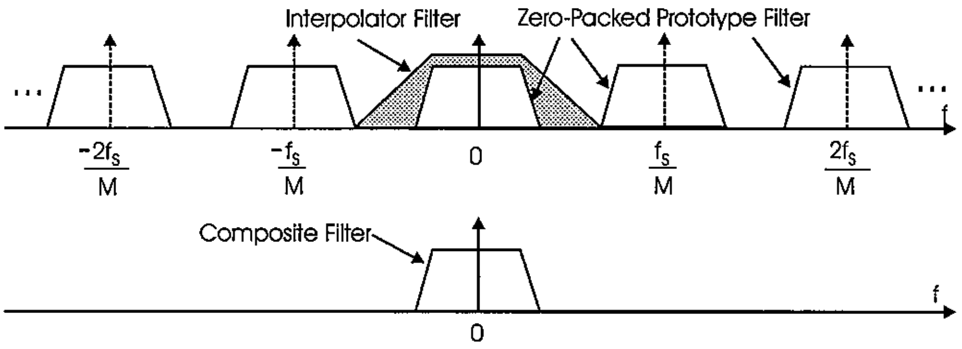


Figure 12.3 Spectral Description of Zero-packed, Interpolating, and Composite Filters

The second filter interpolates the impulse response of the first. In the composite impulse response, the zeros in the impulse response of the first filter are replaced by interpolated samples. This occurs while the filtering action of the second filter suppresses the spectral replicates of the first. By maintaining two separate filters, the composite workload of the two filters is the sum of the lengths of the prototype non-zero-packed filter and that of the following interpolating filter. In the filtering operation, the zero-packed filter with few active coefficients accomplishes the desired spectral bandwidth-limiting process with fewer arithmetic operations but exhibits multiple pass bands. The second filter following the zero-packed filter passes the desired spectra while rejecting the replica spectral bands. We do pay a small penalty in that the cascade filter requires additional data register space to operate the pair of filters. The composite filter also exhibits additional group delay relative to the single filter realization as well as the in-band ripple of both filters.

12.1.1 Interpolated FIR Example

As an example of this process consider the implementation of a digital filter with the following specifications:

Filter Type	Low pass
Sample Rate	100 kHz
Pass Band	0-to-5 kHz
Stop Band	7-to 50 kHz
In-band Ripple	0.1-dB pk-to-pk
Stop band Attenuation	60-dB
Filter Length:	150 Taps

A single-stage filter that meets these specifications requires 150 taps. Figure 12.4 presents the impulse response and frequency response of the single-stage filter.

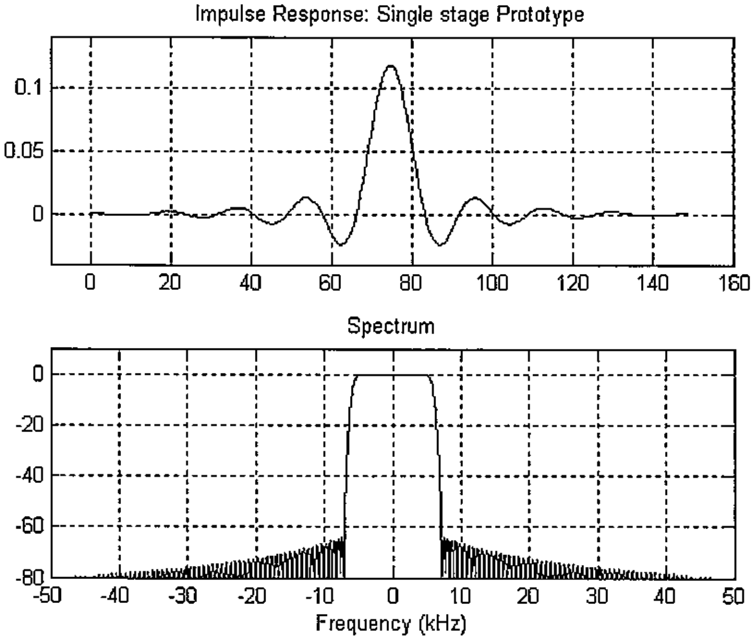


Figure 12.4 Impulse Response and Frequency Response of Prototype 150-tap Filter

The specifications for an interpolated 2-stage filter designed to operate the input stage at one-fifth of the input rate, or 20 kHz, followed by the interpolating filter operating at the input rate of 100 kHz are listed next. As expected the first stage filter length is one-fifth the length of the full sample rate prototype filter. The interpolating filter has a length of 40 taps, so the arithmetic workload for the cascade is 30 + 40 or 70 ops per input as opposed to the 150 ops per input for the single-stage prototype. Figure 12.5 presents the impulse response and frequency response of the two filter stages as well as the cascade of the two stages.

	First Stage	**Second Stage**
Filter Type:	Low pass	Low pass
Sample Rate:	20 kHz	100 kHz
Pass Band:	0-to-5 kHz	0-to-5 kHz
Stop Band:	7-to 10 kHz	15-to-50 kHz
In-band Ripple:	0.08-dB pk-to-pk	0.02-dB pk-to-pk
Stop band Attenuation:	60-dB	60-dB
Filter Length:	30 taps	40 taps

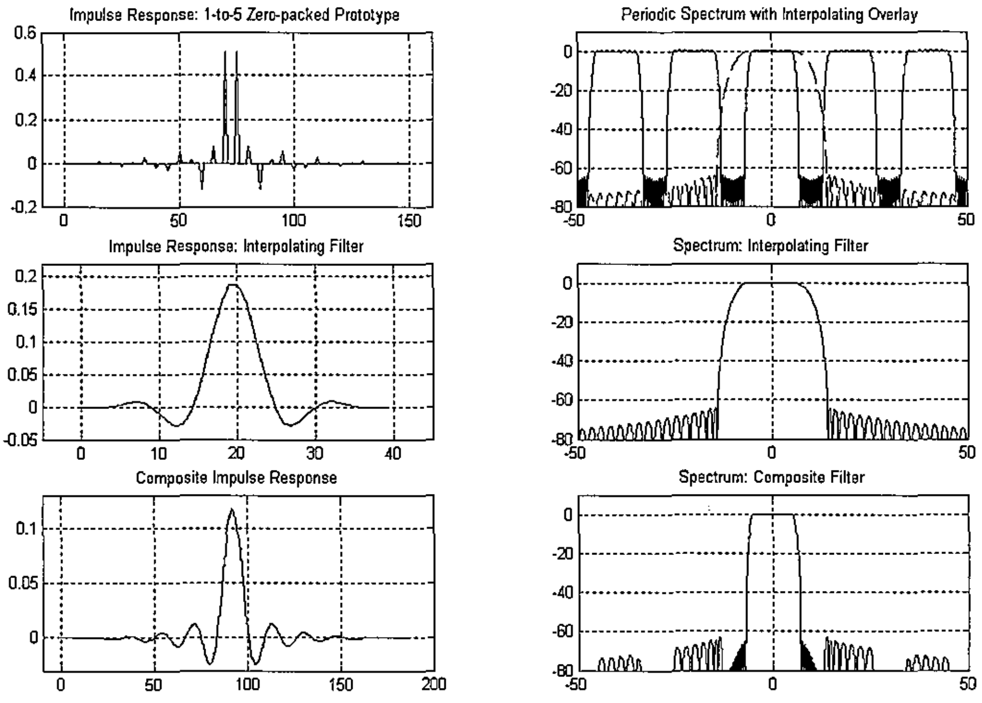


Figure 12.5 Impulse Response of Two Cascade Filters and Composite Interpolated Response Along with Frequency Responses of Each Impulse Response

As a final comparison we can form a cascade of three filters that perform 5-to-1 down sampling, band limiting, and 1-to-5 up sampling of the form shown in Figure 12.1. The input and output stages of this cascade are polyphase partitions of the interpolator in the example just described while the band limiting filter is the non-zero-packed version of the first-stage filter from the same example. Figure 12.6 presents one of the five impulse responses that can be observed at the filter output plus the frequency response of the three stages and the composite response of the three stages. The frequency responses have all been generated at the final output sample rate

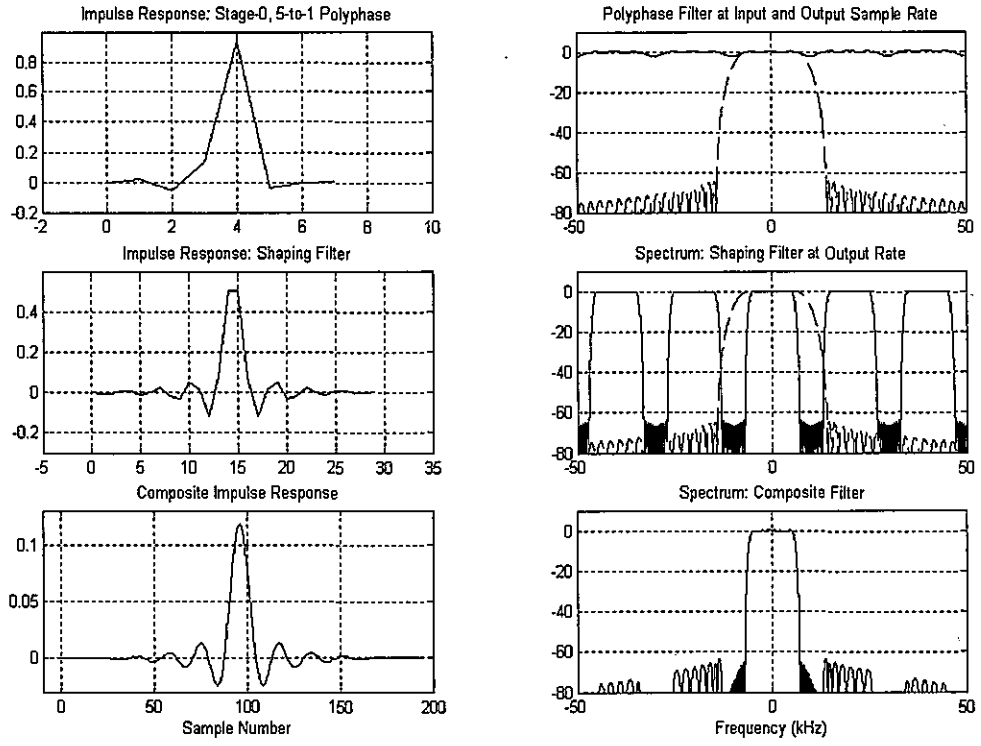


Figure 12.6 Impulse Response of Three Cascade Resampling Filters and Composite Impulse Response Along with Frequency Responses of Each Impulse Response

When this third-option filter operates, for every 5-input points, the workload performed by the three stages are 40 ops, 30 ops, and 40 ops respectively for a total of 110 ops per 5-input samples. The workload per sample for this option is 22 ops per input sample. Interestingly, this option exhibits the smallest arithmetic workload even though it exhibits the longest composite impulse response. The length of the impulse response can be determined in the following manner: an applied impulse outputs an 8-point sequence from the first filter corresponding to one of the 5-legs of the input polyphase filter. This sequence then passes through the 30-tap filter to output a sequence of length 37 points. This 37-tap sequence in turn moves through the five legs of the output polyphase filter to output a sequence of length 224 points (37 * 5 + 40 – 1). Table 12-1 compares the arithmetic workload and the equivalent filter length of the three filter options examined in this example.

Table 12-1 Comparison of Three Filter Structures: Coefficient Lengths, Arithmetic Workload, and Length of Composite Filter Impulse Response

Option	Filter Coefficients	Arithmetic Workload	Impulse Response Length
Single Stage	150 Taps	150 ops/input	150 Taps
Interpolated Filter	30 and 40 Taps	70 ops/input	189 Taps
5-to-1 and 1-to-5 Resampling Filter	40, 30, and 40 Taps	22 ops/input	224 Taps

In this comparison we note an observable truism about signal processing: To obtain the lowest processing burden, a signal-processing task should always be conducted at the lowest possible sample rate commensurate with the signal bandwidth!

12.2 Spectral Masking Filters Based of Half-band Filters

In the previous section we discussed the task of controlling filter length where filter length is proportional to the ratio of sample rate to transition bandwidth. We also noted that narrow transition bandwidth is often associated with narrow pass band bandwidth. The large ratio of sample rate to filter bandwidth permitted us to reduce the sample rate for the filter design and thus obtain a system design requiring a reduced number of coefficients. What design option is available when the ratio of sample rate to transition bandwidth is large but the pass band bandwidth is too large to permit changes in sample rate? The primary design technique is known as *frequency masking*. This technique can be used to obtain narrow transition bandwidths for arbitrary pass band or stop band bandwidths.

The spectral masking design technique is an extension of the IFIR filter design technique. For clarity, we first examine a restricted subset of this procedure that starts with a half-band filter and its complementary high pass filter. Later we will extend the process to an arbitrary low pass filter and its complementary high pass filter. In this procedure a half-band filter H(Z) is designed with a desired transition bandwidth at 1/Mth of the desired sample rate. This filter is zero packed 1-to-M to form the filter $H(Z^M)$. The zero packed filter maintains the narrow transition bandwidth but exhibits a periodic pass band due to the zero-packing. The complementary filter of the zero-packed two-path polyphase filter also exhibits a periodic spectral pass band.

Masking filters following both filters eliminate selected subsets of their periodic pass bands. The signal paths with the two masked spectra are then added to form the composite spectrum. Equation (12.1) describes the Z-transform of the composite spectral masking filter while Figure 12.7 presents the block diagram of the same filter. Figure 12.8 presents the spectral response of the separate filters and the response of them in cascade as well as the composite system.

$$T(Z) = H(Z^M)G_0(Z) + H_C(Z^M)G_1(Z) \tag{12.1}$$

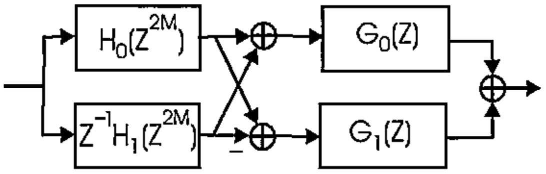
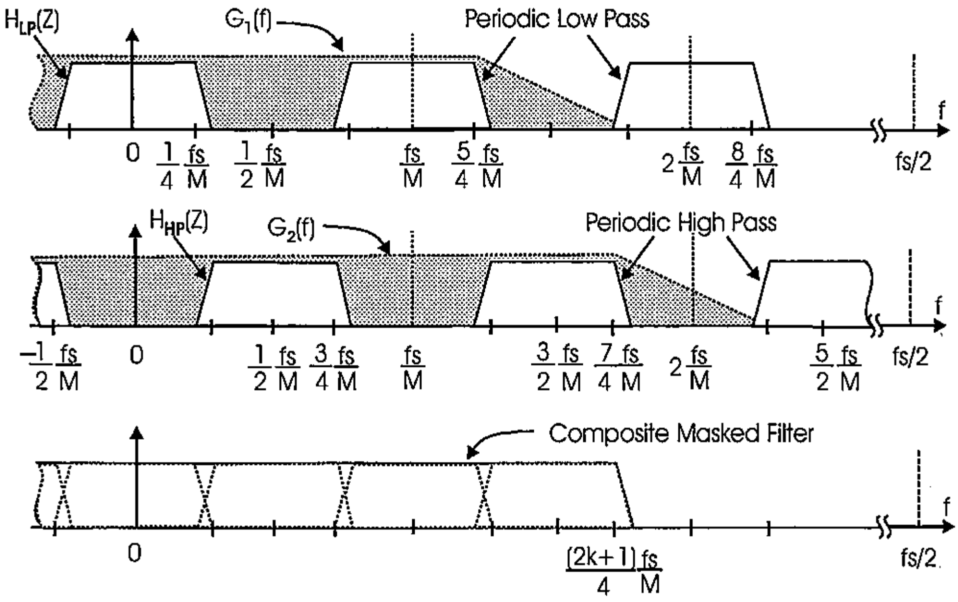


Figure 12.7 Block Diagram of Half-band Spectral Masking Filter



$H_{LP}(Z)$
$G_1(f)$
Periodic Low Pass
0
$\frac{1}{4}\frac{fs}{M}$
$\frac{1}{2}\frac{fs}{M}$
$\frac{fs}{M}$
$\frac{5}{4}\frac{fs}{M}$
$2\frac{fs}{M}$
$\frac{8}{4}\frac{fs}{M}$
fs/2
f
$H_{HP}(Z)$
$G_2(f)$
Periodic High Pass
$-\frac{1}{2}\frac{fs}{M}$
0
$\frac{1}{2}\frac{fs}{M}$
$\frac{3}{4}\frac{fs}{M}$
$\frac{fs}{M}$
$\frac{3}{2}\frac{fs}{M}$
$\frac{7}{4}\frac{fs}{M}$
$2\frac{fs}{M}$
$\frac{5}{2}\frac{fs}{M}$
fs/2
f
Composite Masked Filter
0
$\frac{(2k+1)}{4}\frac{fs}{M}$
fs/2
f

Figure 12.8 Spectra of Top Path, $H(Z^M)$ and Masking Filter $G_1(Z)$, Spectra of Bottom Path, $H_C(Z^M)$ and Masking Filter $G_2(Z)$, and Spectra of Sum of Two Paths

We see that the frequency response of the composite filter has a pass band edge that matches the upper edge of one of the spectral replicates and that the transition bandwidth matches that of the prototype low pass filter. We also note from Figure 12.8 that the cutoff frequency of the composite masked filter is fs/4M above one of the spectral replicate center frequencies. The cutoff frequency is the term presented in (12.2). Equation (12.3) reflects the constraint that the cutoff frequency must be less than the half sample rate. Combining these equations, we obtain (12.4), which expresses all possible cutoff frequencies obtainable with masking filters for a range of M in a 1-to-M zero-packed half-band filter. Table 12-2 lists a set of frequencies that satisfies (12.4) for a small range of up-sampling factor M. Here we see, for instance, that in a 1-to-5 up-sampled masking filter we can obtain low pass filters with band-edge frequencies of 3/20, 5/20, 7/20, and 9/20 of the sample rate. As an example, if we have a sample rate of 100 kHz, we can design a masking filter based on a half-band prototype with a 6-dB cut-off frequency of 35 kHz.

$$f_C = \frac{(2k+1)}{4M} f_S \tag{12.2}$$

$$2k+1 < 2M \tag{12.3}$$

$$\frac{f_C}{f_S} = \frac{2k+1}{4M}, \quad 2k+1 < 2M \tag{12.4}$$

Table 12-2 Possible Ratios of 6-dB Pass-band Bandwidth-to-Sample Rate (fc/fs) for Range of Small Integer (M) for 1-to-M Zero Packing in Masking Filter

	K = 1	K = 2	K = 3	K = 3	K = 4
M = 2	3/8	---	---	---	---
M = 3	3/12	5/12	---	---	---
M = 4	3/16	5/16	7/16	---	---
M = 5	3/20	5/20	7/20	9/20	---
M = 6	3/24	5/24	7/24	9/24	11/24

We now illustrate the masking filter design process with an example that meets the following specifications based on a 6-dB frequency equal to 7/20 of the sample rate. From the listed specifications we estimate the required filter length to be 151 taps.

Filter Type	Low pass
Sample Rate	100 kHz
Pass Band	0-to-34 kHz
Stop Band	36-to 50 kHz
In-band Ripple	0.1-dB pk-to-pk
Stop band Attenuation	60-dB
Filter Length:	151 Taps

Figure 12.9 presents the impulse response and frequency response of the single-stage 151-tap filter designed to meet these specifications.

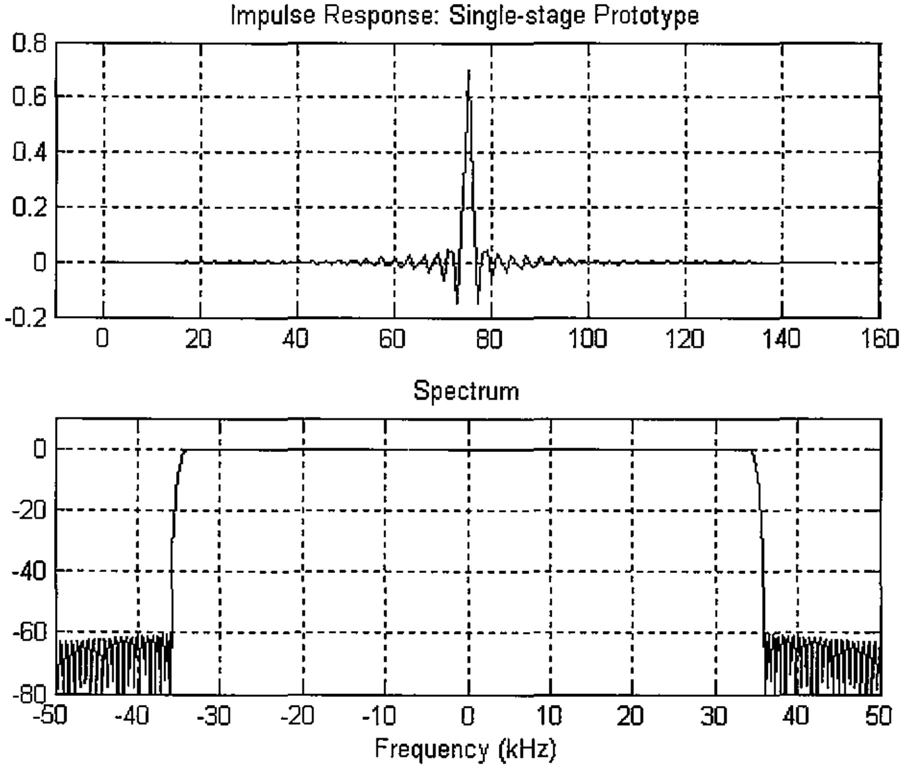


Figure 12.9 Impulse Response and Frequency Response of Prototype 151-tap Filter

The specifications for a frequency masked 2-path filter designed to operate the input stage at one-fifth of the input rate, or 20 kHz, followed by the two masking filters operating at the input rate of 100 kHz, are listed next. We note the need for the stages to exhibit additional out-of-band attenuation to accommodate the increase in side-lobe levels that occur when summing the two paths. The half-band filter, selected to have a 2-kHz transition bandwidth centered at 5 kHz, requires 37 taps. The masking filters with equal transition bandwidths are of the same length and require 35 taps to meet the listed specifications. The arithmetic workload for the cascade is 37 + 35 + 35 or 107 ops per input as opposed to the 151 ops per input for the single-stage prototype. The computational savings is approximately 30%. Figure 12.10 presents the impulse response and frequency response of the three filter stages as well as the composite response of the two paths.

	First Stage	**Mask-1**	**Mask-2**
Filter Type:	Half-band	Low pass	Low pass
Sample Rate:	20 kHz	100 kHz	100 kHz
Pass Band:	0-to-4 kHz	0-to-26 kHz	0-to36 kHz
Stop Band:	6-to 10 kHz	35-to-50 kHz	45-to-50 kHz
In-band Ripple:	0.025-dB	0.025-dB	0.025-dB
Stop band Atten:	66-dB	65-dB	65-dB
Filter Length:	37 taps	35 taps	35 Taps

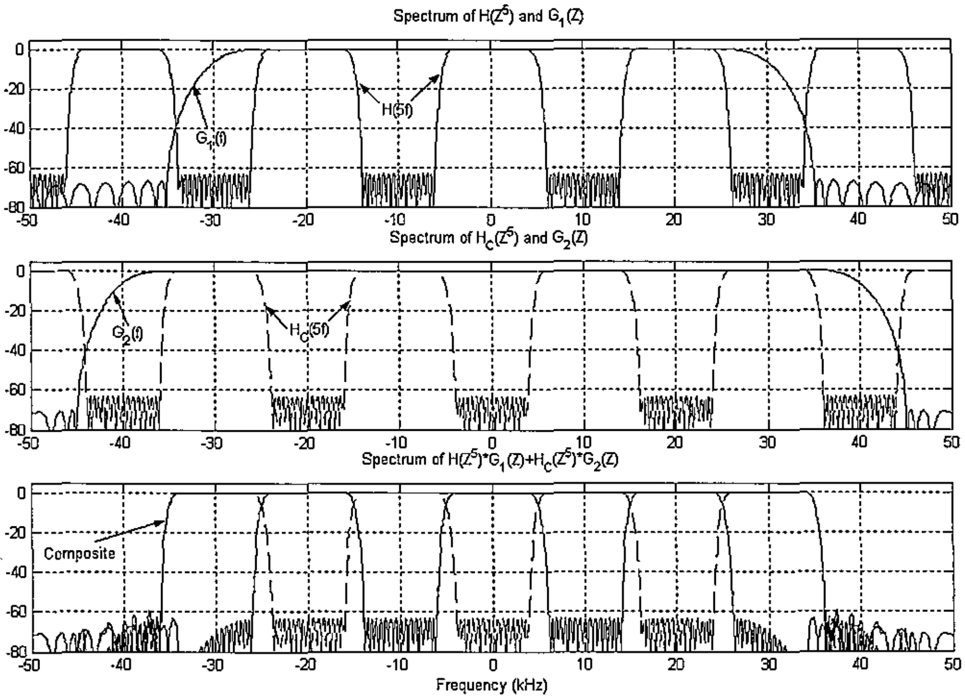


Figure 12.10 Spectra of Up-sampled 1-to-5 Half-band Low pass Filter and First Masking Filter, of Up-sampled 1-to-5 Half-band High pass Filter and Second Masking Filter, and of Composite Sum Filter

12.3 Spectral Masking Filters Based on Complementary Filters

A half-band filter was selected for the prototype in the previous section because it permitted us to determine a simple relationship for the possible band-edge frequencies available in the masking structure. We can modify the design process slightly so that arbitrary band-edge frequencies can be obtained from the same structure. The more general procedure starts with a prototype filter H(Z) with bandwidth $fc = \alpha fs$. The parameter α is nominally between 0 and 0.5, but is usually near 0.25. This filter is designed with the desired transition bandwidth at 1/Mth of the sample rate and, as earlier, is zero packed to form the filter $H(Z^M)$. The up-sampled filter maintains the narrow transition bandwidth but exhibits the periodic pass band due to the zero packing. The filter must have an odd number of coefficients to be able to form its complement as the difference between a delay path and the zero-packed filter. The

(N-1)/2 units of delay is the midpoint of the filter $H(Z^M)$. The Z-transform of the complementary filter $H_C(Z^M)$ is shown in (12.5).

$$H_C(Z^M) = (Z^{-(N-1)/2} - 1)\, H(Z^M) \tag{12.5}$$

We want the complementary filter to have the same stop band attenuation as the prototype filter. To achieve this goal the prototype filter must have equal valued pass band and stop-band ripple. This is accomplished by setting equal valued penalty function weights in the Remez algorithm. The procedure now follows the steps derived in the previous example except we have to select an appropriate resampling rate. We select this rate by comparing the ratios of Table 12-2 to the desired ratio of bandwidth to sample rate and selecting an up-sample rate corresponding to a close ratio. We then sketch the replicated spectra at this up-sample ratio and adjust the bandwidth of the baseband prototype to move the replica band edge to the desired cutoff frequency.

We now illustrate the general masking filter design process with an example that meets the specifications outlined in the previous example except that the band edge has been changed from 34 kHz to 30 kHz.

Filter Type	Low pass
Sample Rate	100 kHz
Pass Band	0-to-30 kHz
Stop Band	32-to 50 kHz
In-band Ripple	0.1-dB pk-to-pk
Stop band Attenuation	60-dB
Filter Length:	151 Taps

Again we find that a single stage filter that meets these specifications requires 151 taps but we seek a more efficient solution. We require a masking filter with a normalized band edge of 0.3. Examining the entries in Table 12-2 we find that the ratio 7/24 or 0.292 in the 1-to-6 up sample is the closest entry with the ratio 5/16 or 0.312 in the 1-to-4 up-sample row a close second. Selecting the 1-to-4 up sample we form the sketch shown in Figure 12.11 and reduce the upper band edge at the first replicate from 31.25 to 30 by reducing the baseband bandwidth from 6.25 to 5.

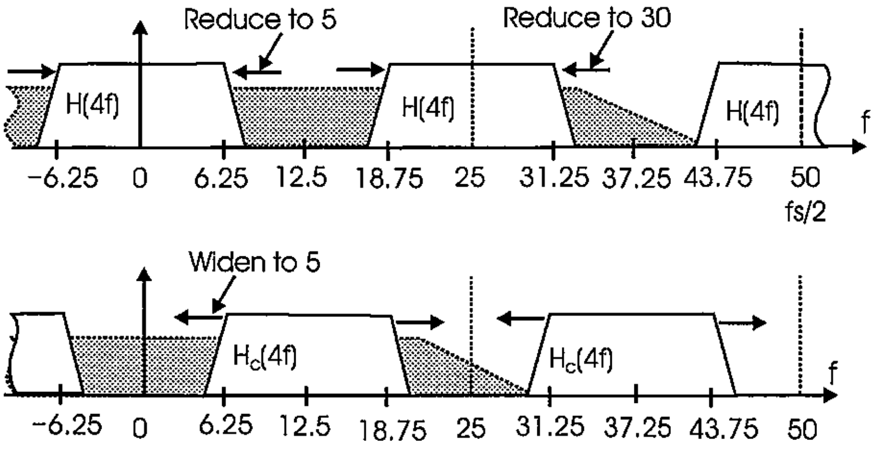


Figure 12.11 Initial Spectral Representation of 1-to-4 Up-sampled Baseband Filter

The specifications for a shifted frequency mask 2-path filter reflecting the shifted edges indicated in Figure 12.11 are listed next. We note the additional out-of-band attenuation required to accommodate the increase in side-lobe levels that occur when summing the two paths. Also note that the in-band ripple and out-of-band attenuation of the first stage correspond to equal values of ripple ($0.56*10^{-3}$). The masking filters now have different transition bandwidths, hence have different lengths of 25 and 33 taps. The time series output from the two paths must be time aligned by adding 4-units of delay to the output of the top path, which has the shorter filter. The arithmetic workload for this frequency-masked filter is 45 + 25 + 33 or 103 ops per input as opposed to the 151 ops per input for the single-stage prototype. This represents a computational savings of approximately 32%. Figure 12.12 presents the impulse response and frequency response of the separate paths and of the composite filter.

	First Stage	**Mask-1**	**Mask-2**
Filter Type	Low pass	Low pass	Low pass
Sample Rate	20 kHz	100 kHz	100 kHz
Pass Band	0-to-5 kHz	0-to-30 kHz	0-to-20 kHz
Stop Band	7-to 10 kHz	45-to-50 kHz	30-to-50 kHz
In-band Ripple	0.005-dB	0.045-dB	0.045-dB
Stop band Attenuation	65-dB	65-dB	65-dB
Filter Length:	45 taps	25 taps	33 taps

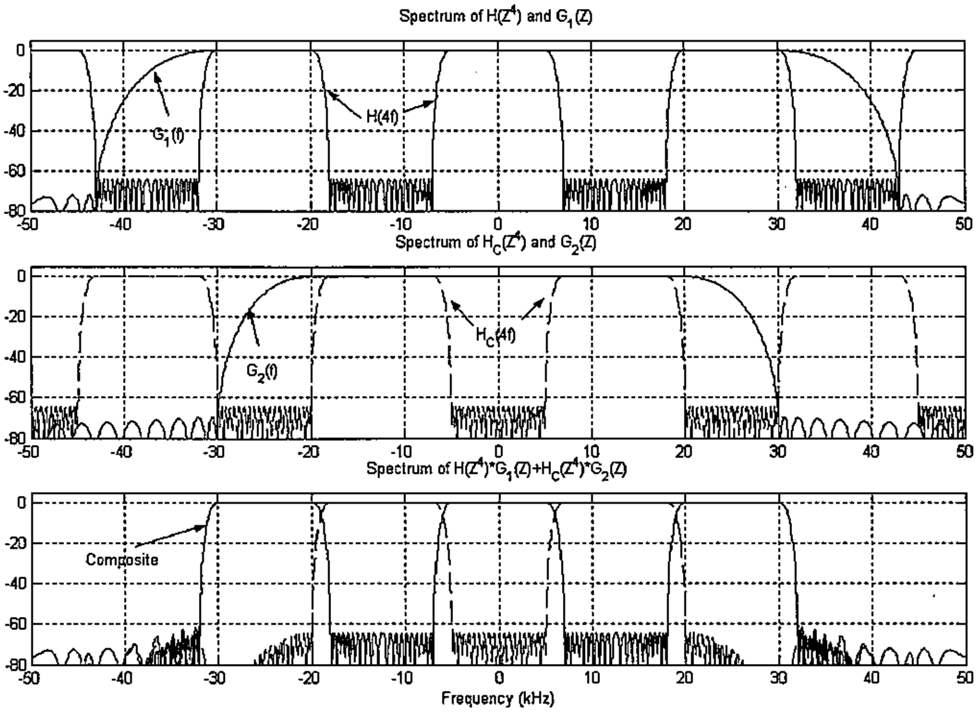


Figure 12.12 Spectra of Up-sampled 1-to-4 Low pass Filter and First Masking Filter, of Up-sampled 1-to-4 Complementary Filter and Second Masking Filter, and of Composite Sum Filter

12.4 PROPORTIONAL BANDWIDTH FILTER BANKS

Proportional bandwidth filter banks are used to obtain a spectral partition in which the center frequencies and the bandwidths of the filters are equally spaced on a logarithmic scale or logarithmically spaced on a linear scale. Filter banks with this property are often called constant Q filter banks where Q in this application is the ratio of center frequency to bandwidth. Vibration analysis, audio analysis, and signal detection and classification of periodic signals with harmonic components are best performed in constant Q filter banks. The graphic equalizer of a home entertainment system has a constant Q filter bank. Audio instruments monitoring sound pressure level for noise abatement have constant Q filter banks with each filter spanning one third or one sixth of an octave.

12.4.1 Octave Partition

In this section we examine the standard approach to proportional bandwidth filtering. A processing scheme is designed to decompose a single octave of bandwidth located near the quarter-sample rate. With appropriate half-band filtering and sample-rate reduction, the frequency band occupying the next lowest octave slides up to occupy the same spectral interval at the reduced sample rate. This octave shifting due to 2-to-1 resampling is illustrated in Figure 12.13.

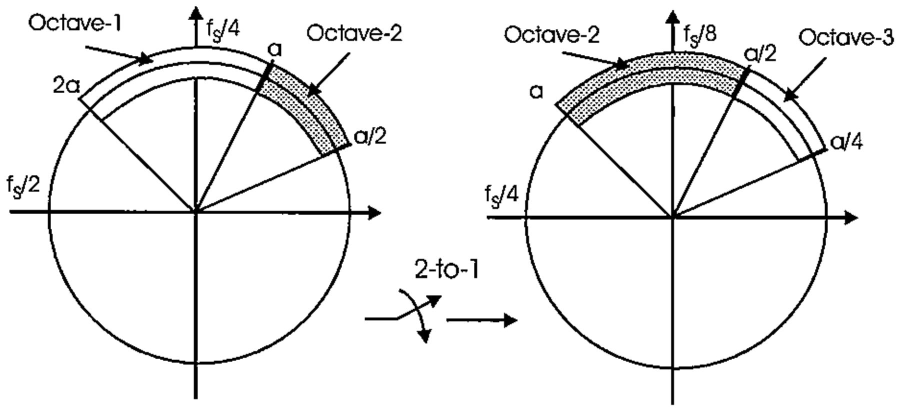


Figure 12.13 Spectral Band, Octave K+1 (Octave-2) Sampled at fs Shifts to Location of Octave K (Octave-1) when 2-to-1 Down Sampled to fs/2

Thus a single octave processing scheme is applied to successively lower octaves as prefiltering and resampling iteratively delivers the sequence of octaves to the same fractional bandwidth at successively reduced sample rates. The signal flow of the multioctave process is shown in Figure 12.14. The suboctave processor extracts spectra from the top octave partitioning that octave with a bank of fractional octave filters. The filters used for the octave processing and for the half-band processing can be FIR or IIR. IIR filters are often used to perform the spectral partition when the application is spectral analysis, audio equalization, or audio compression. FIR filters are used for image processing or other applications requiring linear phase.

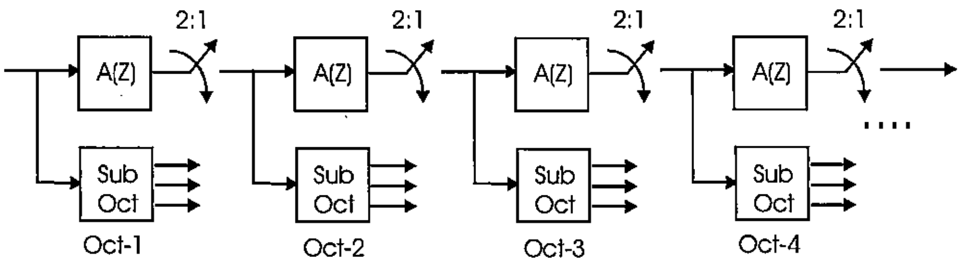


Figure 12.14 Multioctave Processing Iteratively Filter and Down Sample to Access Successive Octaves

An interesting feature of the multioctave processor is the aggregate workload to implement the system. Let us assume that the processing workload of the first stage, the input stage that forms the top octave and the first half-band filter, is N operations per input sample. The next stage requires the same workload but operating at half the clock rate requires N/2 operations per input sample. The workload for the sequence of stages is the sum shown in (12.6). Here we see that the workload for an arbitrarily large number of octaves is less than twice the workload of the first octave.

$$\frac{\text{Ops}}{\text{Input}} < \{N + N/2 + N/4 + N/8 + \ldots..\} < 2N \tag{12.6}$$

12.4.2 Proportional Bandwidth Filters

Suppose we have M filters spanning a frequency interval in equal increments on a log scale. We first address the center frequencies of the filter bank. Let f_0 be the center frequency of the lowest frequency filter in the filter bank. The *k*th proportional bandwidth filter is located at the center frequency shown in (12.7).

$$f_k = f_0(1+r)^k \tag{12.7}$$

We can solve for the parameter r if we evaluate (12.7) with f_k set to the *M*th center frequency and the exponent k set to M–1 as shown in (12.8).

$$f_M = f_0(1+r)^{M-1} \tag{12.8}$$

To design the filters we have to identify the band edges of each filter as well as assure that the lower band edge of the first filter and the upper band edge of the *M*th filter correspond to the bandwidth spanned by the filter bank. We modify equations (12.7) and (12.8) to partition the inclusive interval into 2M increments and then use odd indices in the exponent to identify the band centers and even indices to identify band edges. The modified form is shown in (12.9).

$$f_{\text{Low-Edge}} = f_{\text{High-Edge}}(1+s)^{2M} \tag{12.9}$$

After solving for the parameter s we can use the expressions of (12.10) to solve for the critical frequencies of the proportional bandwidth filter bank.

$$\left.\begin{aligned} &k\text{th Lower Band Edge} = f_{\text{Low-Edge}}(1+s)^{2k-2} \\ &k\text{th Center Frequency} = f_{\text{Low-Edge}}(1+s)^{2k-1} \\ &k\text{th Upper Band Edge} = f_{\text{Low-Edge}}(1+s)^{2k} \end{aligned}\right\} \quad k=1,2,\ldots,M \tag{12.10}$$

12.4.2.1 Example of Proportional Bandwidth Design

Let us examine a standard 15-band, International Organization for Standardization (ISO) 2/3 octave filter bank used as a graphic equalizer in a high-end audio system. For this design the input sample rate is 48 kHz, with 15 frequency bands spanning the frequency interval 20 Hz through 20 kHz with center frequencies listed here.

ISO Center Frequencies

25, 40, 63, 100, 160, 250, 400, 630, 1000, 1600, 2500, 4000, 6300, 10000, 16000

If we apply (12.10) to these requirements we can solve for the term (1 + s) and then verify the center frequencies of the filter bank. This is done in (12.11) and (12.12).

$$\begin{aligned} &20000 = 20(1+s)^{30} \\ &\log_{10}(1+s) = \frac{1}{30}\log_{10}(1000) \\ &(1+s) = 10^{3/30} = 1.2589\ldots \end{aligned} \tag{12.11}$$

$$f_k = 20\,(1.2589..)^{(2k-1)},\ k = 1, 2, 3, \ldots, M \tag{12.12}$$

Table 12-3 lists the center frequencies obtained by solving (12.12) and the corresponding center frequencies identified in the ISO standard. We see that, applying standard rules for identifying parameters for instruments, the ISO frequencies have been adjusted so that in every 5 intervals the values repeat in increments of 10, as in 25, 250, and 2500 Hz. The table also lists the computed band edges of the proportional bandwidth filters without the shifted correction to match the ISO frequency list.

Table 12-3 Computed and ISO Standard Center Frequencies of 15-band Proportional Bandwidth Filter Bank; Also, Lower and Upper Edge of Same Bands

Filter Number	Computed Frequency	ISO Standard Frequency	Lower Edge	Upper Edge
1	25.18	25	20	32
2	39.91	40	32	50
3	63.25	63	50	80
4	100.24	100	80	125
5	158.87	160	126	200
6	251.79	250	200	317
7	399.05	400	317	502
8	632.46	630	502	796
9	1 002.38	1 000	796	1 262
10	1 588.66	1 600	1 262	2 000
11	2 517.85	2 500	2 000	3 170
12	3 990.52	4 000	3 170	5 024
13	6 324.56	6 300	5 024	7 962
14	10 023.74	10 000	7 962	12 619
15	15 886.56	16 000	12 619	20 000

Starting at the higher frequency bands, we observe that bands 15 and 14 bracket 12 kHz, the quarter sample rate. These bands must be implemented at the full sample rate of 48 kHz and the remaining bands can be processed at the reduced rate of 24 kHz after a half-band filter and 2-to-1 down sampling. Continuing up the list we note that bands 13 and 12 bracket (or are close) to 6 kHz, the new quarter sample rate and must be implemented at the 24 kHz sample rate. The "close to" consideration is used to avoid the condition that the filter transition bandwidth crosses the quarter sample rate. The remaining filters can be processed at a reduced rate after a second half-band filter and a second 2-to-1 down sampling. In this manner we can allocate where in the down sample chain each channel filter has to be implemented. This partition was performed for the list of frequency edges listed in Table 12-3 and filters were designed to obtain the band edges at the selected sample rate.

To meet the requirements of all 15 filters at their corresponding sample rate the filters were designed with a normalized unity sample rate. Examining the filters we found that only four filters operating at their appropriate sample rate were required to implement all 15 filters. Table 12-4 presents the filters in the bank, the sample rate at which they operate, their (uncorrected) normalized frequency edges, and the implementation assignments. A fifth, half-band, filter was required to finish the design suite. Figure 12.15 presents the filter partition for the 15-band proportional bandwidth filter bank. As shown, the filter bank performs as a spectrum analyzer. To operate as a graphic equalizer, gain control would be applied to the filter outputs, and then the outputs would be up-sampled and merged by a filter bank of dual structures.

Table 12-4 Down-sampled Sample Rate and Normalized Band-edge Frequencies for 15-Band Proportional Bandwidth Filter Bank

Filter Number	Sample Rate	Lower Edge	Upper Edge	Filter Type
1	93.75	0.2133	0.3413	C
2	187.5	0.1707	0.2667	B
3	375	0.1333	0.2133	D
4	375	0.2133	0.3333	C
5	750	0.1680	0.2667	B
6	1 500	0.1333	0.2113	D
7	1 500	0.2113	0.3347	C
8	3 000	0.1673	0.2653	B
9	3 000	0.2653	0.4207	A
10	6 000	0.2103	0.3333	C
11	12 000	0.1667	0.2642	B
12	24 000	0.1321	0.2093	D
13	24 000	0.2093	0.3318	C
14	48 000	0.1659	0.2629	B
15	48 000	0.2629	0.4167	A

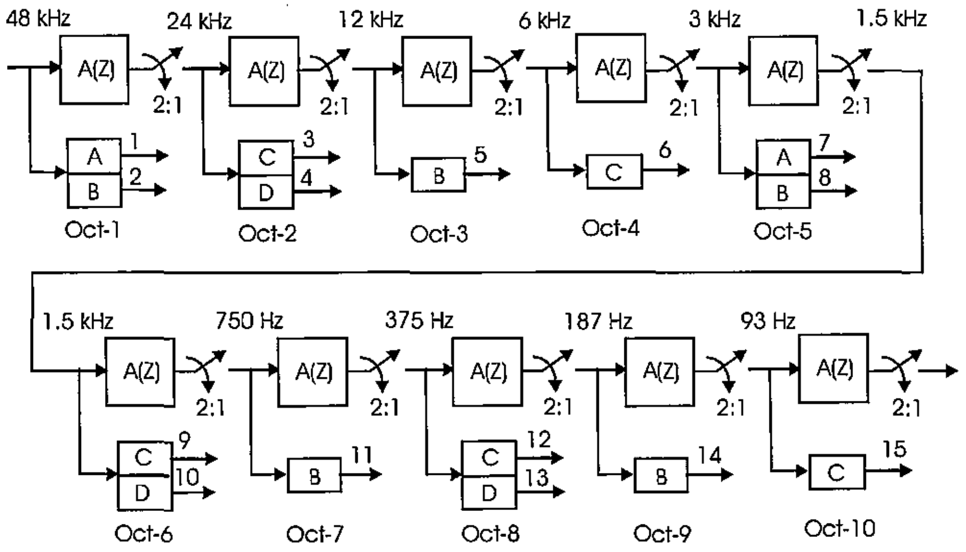


Figure 12.15 Filter Partition for 15-band Proportional Bandwidth Analysis Filter Bank

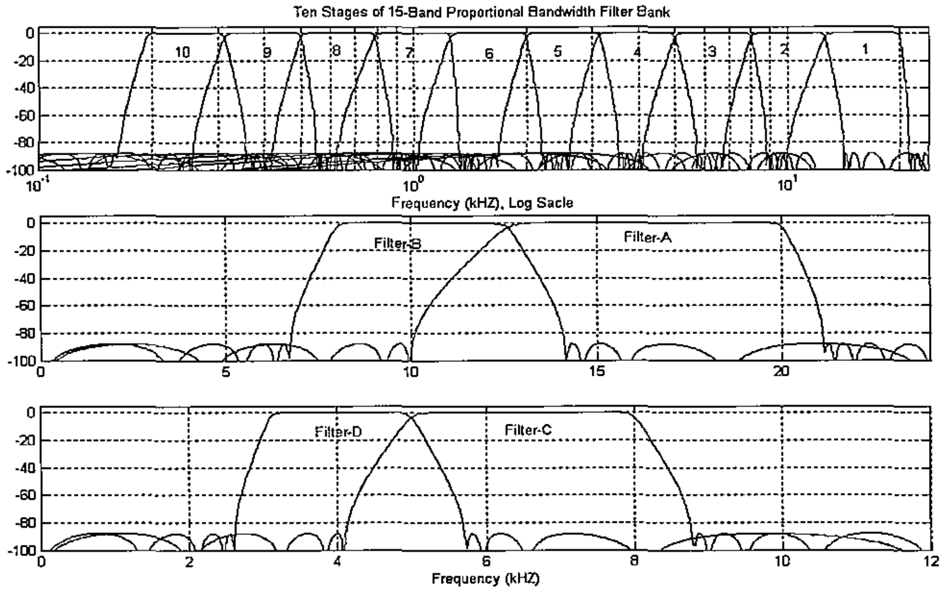


Figure 12.16 Frequency Response of First 10 Bands of 15-stage Proportional Bandwidth Filter Bank and Four Filter Types Distributed Through Half -band Filter Chain

Figure 12.16 presents the frequency response of the first 10 proportional bandwidth filters in the filter bank, plus the four prototype filters identified in Table 12-4 as filter types A, B, C, and D. Each of these absurdly good filters was designed as a two-path polyphase IIR filter of the type presented in Chapter 10. Each is a 9-pole band-pass filter requiring 19-multiplies per output sample. More traditional filter banks use 3-pole Butterworth filters that in the polyphase IIR form would require 6-multiplies per output sample. The half-band filter is also a 2-path recursive polyphase structure requiring 4-multiplies per output sample to form the 9-poles of the low pass. The half-band filter operates at half rate with the resampling being performed prior to the filtering operation. An upper bound to the processing workload for the entire filter chain can be estimated by examining the workload for the first stage. This workload is seen to be:

First Stage:	Half Band	2 ops/input
	Filter A	19 ops/input
	Filter B	19 ops/input
	Total First Stage	40 ops/input

The workload for the entire chain must be less than twice this workload, which is less than 80 arithmetic operations per input sample to obtain the output of 15 filters. The actual count is 72 operations per input sample. The workload per filter is seen to less than 5 operations per input sample. Implementing the proportional bandwidth filters, as 3rd-order Butterworth prototypes would reduce the workload for the entire 15-band filter bank to approximately 26 operations per input sample. Amortized over the 15 filters, this is less than 2-operations per input sample per filter.

12.4.2.2 Fractional Bandwidth Design Example

This example examines a 1/12-octave proportional bandwidth Spectrum analyzer. The design decomposes an octave band of frequencies spanning the normalized frequency interval 3/16 to 6/16 into 12 proportional bandwidth filters. The same 12-band filter bank processes successive octaves after they are filtered and down sampled 2-to-1 by a half-band filter. The chain is repeated 10 times to span the three decades from 20 Hz to 20 kHz. The 12-filters spanning the octave were implemented as 2-path recursive polyphase structures requiring 6-multiplies to form each 3-pole band-pass elliptic filter. The half-band filter was also designed as a 2-path recursive polyphase structure implemented as a resampling filter requiring 3-multiplies per input sample. The workload required to process the first octave is seen to be:

First Stage:	Half-band:	3 ops/input
	12-Proportional Bandwidth Filters:	72 ops/input
	Total First Stage	75 ops/input

The workload for the entire chain must be less than twice this workload, which is less than 150 arithmetic operations per input sample, to obtain the output of the 12 filters per octave for each of the 10 octaves spanned by the process. A total of 120 filter outputs are formed by this process, and amortizing the workload over the 120 filters we find that the workload per filter is 1.25 operations per filter per input sample. Figure 12.17 presents the frequency response of the 12-fractional octave filters spanning the top octave from 3/16 to 6/16 along with the response of the same filters in the next lower octave. Also shown is the frequency response of the first four half-band filters operating at their successively lower normalized sample rates of 1, 1/2., 1/4, and 1/8.

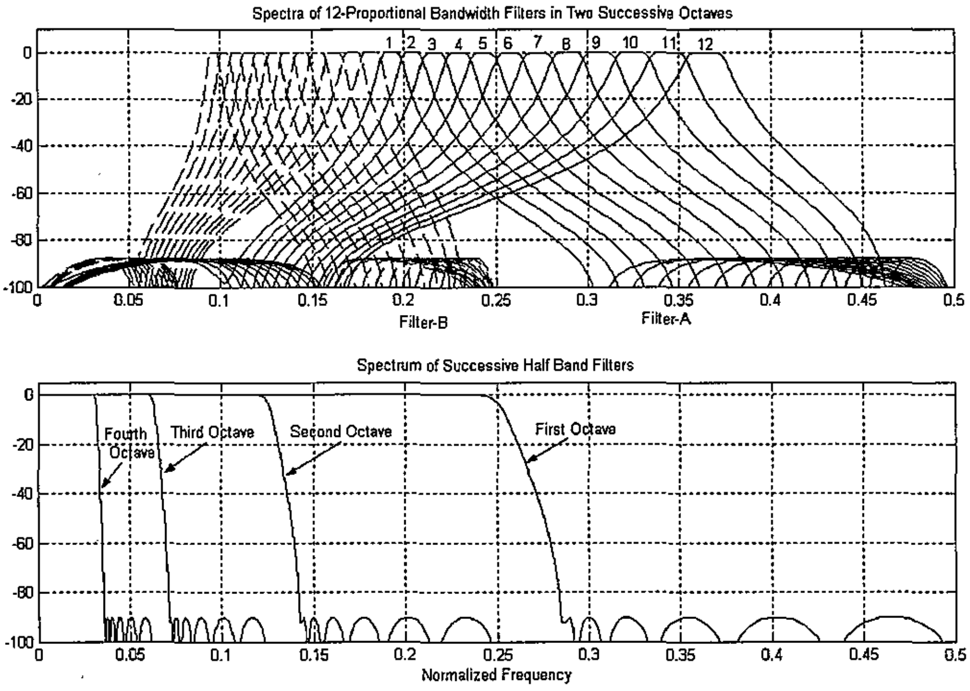


Figure 12.17 Frequency Response of 12-fractional Octave Filter Bank in Top Octave and in Next Lower Octave and Spectra of Four Successive Half-band Filter Responses

References

Crochiere, Ronald and Lawrence Rabiner, *Multirate Signal Processing*, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1983.

Fliege, Norbert, *Multirate Digital Signal Processing: Multirate Systems, Filter Banks, Wavelets*, West Sussex, John Wiley & Sons, Ltd., 1994.

Jovanovic-Dolecek, Gordana, *Multirate Systems: Design and Applications*, London, Idea Group, 2002.

Mitra, Sanjit, *Digital Signal Processing: A Computer-Based Approach*, 2nd ed., New York, McGraw-Hill, 2001.

Mitra, Sanjit and James Kaiser, *Handbook for Digital Signal Processing*, New York, John Wiley & Sons, 1993.

Vaidyanathan, P. P., *Multirate Systems and Filter Banks*," Englewood Cliffs, NJ, Prentice-Hall, Inc., 1993.

Problems

12.1 Design a FIR filter with the following specifications:

Sample Rate	1.0 MHz		
Pass Band	0-to-10 kHz	Pass-band Ripple	0.1-dB
Stop Band	20-to- fs/2 kHz	Stop-band Atten:	60.0-dB

Now repeat the same design for a sample rate of 50 kHz. Note that the filter length is proportional to the sample rate.

Now zero pack the shorter filter to increase its length to the high data rate filter. The zero packing raises its sample rate and accesses the 20 spectral copies at the higher sample rate. Now design an interpolating filter to suppress the spectral copies to –60-dB. Determine and plot the frequency response of the cascade filter pair. Compute and compare the workload required to implement the single large filter and the cascade interpolated filter.

12.2 Design a FIR filter with the following specifications:

Sample Rate	1.0 MHz		
Pass Band	0-to-10 kHz	Pass-band Ripple	0.1-dB
Stop Band	20-to- fs/2 kHz	Stop-band Atten.	60.0-dB

Now repeat the same design for a sample rate of 500 kHz. Note that the filter length is proportional to the sample rate.

Design a half-band filter to lower the bandwidth and sample rate 2-to-1. Form a composite filter as a cascade of three filters, a half-band 2-to-1 filter, the low-pass filter, and a half-band 1-to-2 filter. Determine the workload for the cascade, and compare it to the workload of the single stage filter.

12.3 Design a FIR filter with the following specifications:

Sample Rate	1.0 MHz		
Pass-band	0-to-10 kHz	Pass-band Ripple	0.1-dB
Stop-band	20-to- fs/2 kHz	Stop-band Atten.	60.0-dB

Now repeat the same design for a sample rate of 250 kHz. Note that the filter length is proportional to the sample rate.

Design a cascade of two half-band filters to lower the bandwidth and sample rate 4-to-1. Form a composite filter as a cascade of three filter sets, the two half-band 2-to-1 filters, the low pass filter, and the two half-band 1-to-2 filters. Determine the workload for the cascade, and compare it to the workload of the single-stage filter.

12.4 We are going to design a FIR filter with the following specifications:

Sample Rate	80 kHz		
Pass-band	0-to-30 kHz	Pass-band Ripple	0.1-dB
Stop-band	35-to- 40 kHz	Stop-band Atten.	60.0-dB

This is our reference design. We now design a FIR filter to up-sample and mask. The prototype filter has the following specifications:

Sample Rate	40 kHz		
Pass-band	0-to-10 kHz	Pass band Ripple	0.1-dB
Stop-band	15-to- 20 kHz	Stop Band Atten.	60.0-dB

This second filter is zero packed 1-to-2 and partitioned into a 2-path polyphase filter as shown in Figure 12.7. We form the low pass and high pass legs of this filter. Determine and plot the frequency response of the replicated low pass and high pass filters. Now design the masking filter to reject the high-frequency replicate in the low-pass path. Be sure this filter has an odd number of taps so we can align the lower leg with the filtered upper leg. Mask the upper leg with its second filter, and form the sum with its output and the delayed lower leg. Verify the masked filter satisfies the filter specification. Compare the two filters, the masked and the direct form. Compare the workload required to implement the two filters.

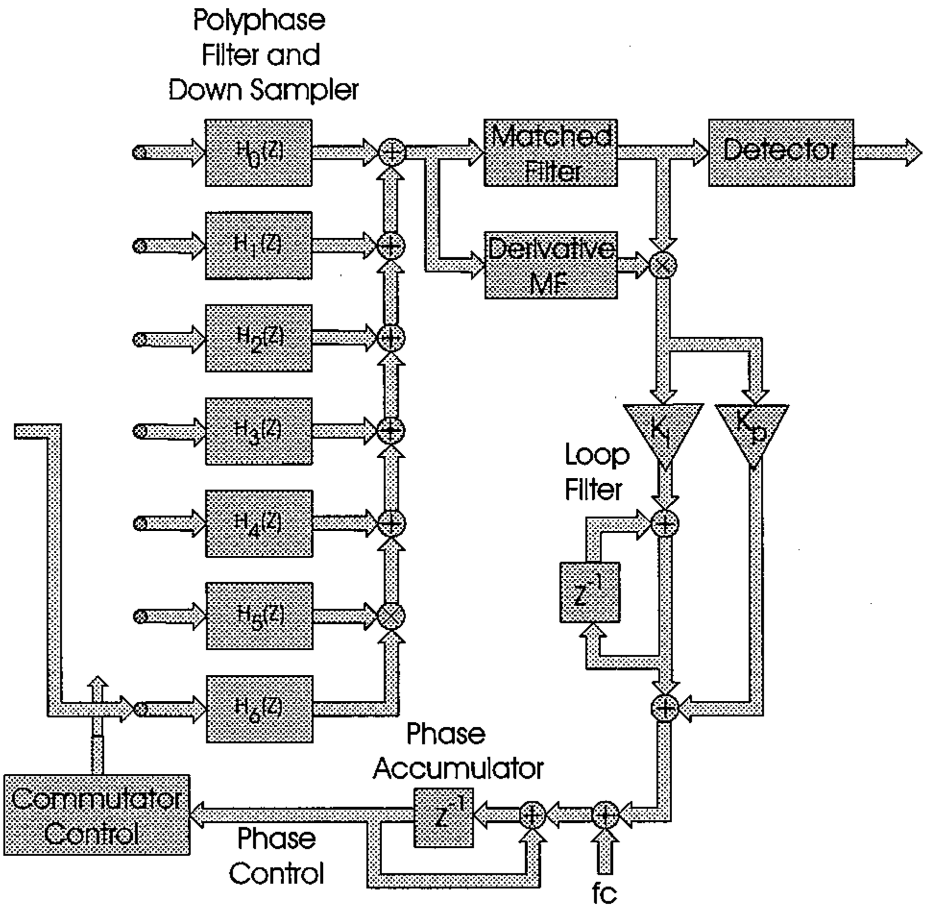
12.5 Design a set of three proportional bandwidth IIR filters that spans an octave of frequency between fs/8 and fs/4. The filters will be 3rd-order inverse Tchebyschev filters, sometimes referred to as Tchebyschev-II filters. Determine the normalized band edges of the filter set. These band edges will be the 3-dB crossover points of adjacent filters, and the filters will exhibit 60-dB out-of-band attenuation. Plot the frequency response of the three filters.

12.6 Design a set of three proportional bandwidth FIR filters that spans an octave of frequency between fs/8 and fs/4. Determine the normalized band edges of the filter set. These band edges will be the 1-dB crossover points of adjacent filters, and the filters will have a transition bandwidth that is 60-dB down in the center of the next lowest frequency band and exhibit 1/f spectral side-lobes thereafter. Plot the frequency response of the three filters.

12.7 Design a set of three proportional bandwidth FIR filters that spans an octave of frequency between 3/16 fs and 3/8 fs. Determine the normalized band edges of the filter set. These band edges will be the 1-dB crossover points of adjacent filters, and the filters will have a transition bandwidth that is 60-dB down in the center of the next lowest frequency band and exhibit 1/f spectral side-lobes thereafter. Plot the frequency response of the three filters. Compare the filter length required to span this octave to filter length required to span the octave defined in problem 12.6. Explain the difference if any.

CHAPTER 13

Communication Systems Applications



Communication systems make liberal use of multirate filters in several ways. Multirate processing finds application in shaping filters, in channelizers, in interpolators, in efficient bandwidth and sample rate reduction schemes, in anti-alias filtering, and in many other applications. This chapter deals with applications that fall under the other category. It is a potpourri of applications we have found over many years of applying signal processing techniques to communication systems. Some of these applications are standard and are well known among practitioners but are included to bring the neophyte into the inner circle. Other applications are unique and will quicken the heart of even the most seasoned practitioner.

13.1 Conventional Digital Down Converters

A radio receiver down converts and demodulates a narrowband radio frequency (RF) signal embedded in a block of frequencies assigned to a particular radio service. For instance, the commercial FM band spans the frequency span from 88 MHz to 108 MHz with multiple 200-kHz channels. Similarly cable TV modems select, down convert, and demodulate 5-MHz symbol rate QAM signals spanning 54 to 216 MHz with 6 MHz channels interspersed with legacy frequency gaps. The traditional architecture of a radio receiver that performs this task is shown in Figure 13.1. This standard architecture performs two frequency translations and is called a dual-conversion receiver. The receiver passes the input signal through an image reject filter, amplifies, and then down converts a selected RF channel to an intermediate frequency (IF) filter that performs initial bandwidth limiting. The output of the IF filter is again down converted to baseband by matched quadrature mixers that are followed by matched analog baseband filters that perform final bandwidth control. Each of the quadrature down-converted signals is then converted to its digital representation by a pair of matched analog-to-digital converters (ADC). The output of the ADCs is processed by DSP engines that perform the required synchronization, equalization, demodulation, detection, and channel decoding.

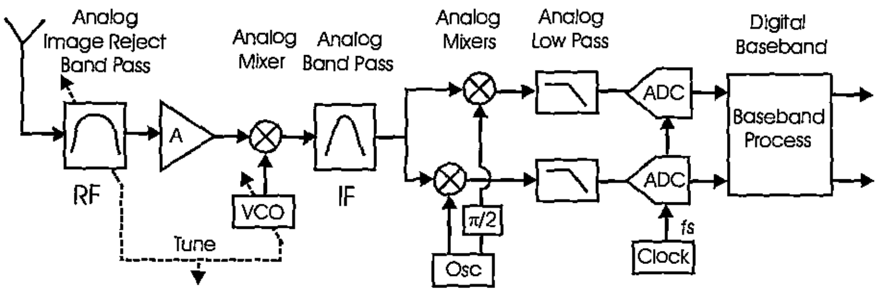


Figure 13.1 Standard Radio Receiver Architecture

Gain and phase imbalance between the two paths containing the quadrature mixers, the analog baseband filters, and the ADCs in the receiver are the cause of cross talk between the in-phase and quadrature (I/Q) components. In addition, the ADCs inject a DC term in the center of the baseband signal, and the analog filters introduce group delay distortion. Adaptive algorithms can remove the imbalances, the DC terms, and the phase distortion as background processing tasks in the DSP segment of the receiver. Rather than repairing the analog defects in the DSP domain, we have high motivation to avoid these distortion effects entirely by performing the entire baseband processing task in the DSP domain. Besides the advantages of avoiding performance degradation due to path imbalance due to analog component tolerance, avoiding performance degradation due to component parameter drift with time and temperature, avoiding the cost of quadrature mixers, and avoiding group delay distortion associated with analog filters, DSP insertion also offers the attraction of flexibility related to filters with programmable bandwidth and sample rates.

Figure 13.2 presents the block diagram of a second generation receiver in which the conversion from analog to digital occurs at IF rather than at baseband. Examining the receiver, we note two significant changes in the processing stream. First, due to the higher center frequency of the IF signal, the ADC must operate at a higher sample rate than in the baseband version. Second, we see that the down conversion of the selected channel is performed by a digital down converter and digital low-pass filter which, because of the higher sample rate, now includes a resampling operation. We are willing to accommodate this extra processing burden to gain the advantage that the DSP-based down conversion is free of imbalance-related distortion terms. A second advantage of the digital translation process is that the digital filters in the process are designed to have linear phase characteristics, a characteristic trivially simple to realize in digital nonrecursive filters. Another option we will examine shortly is the ability to embed the down conversion in the resampling process.

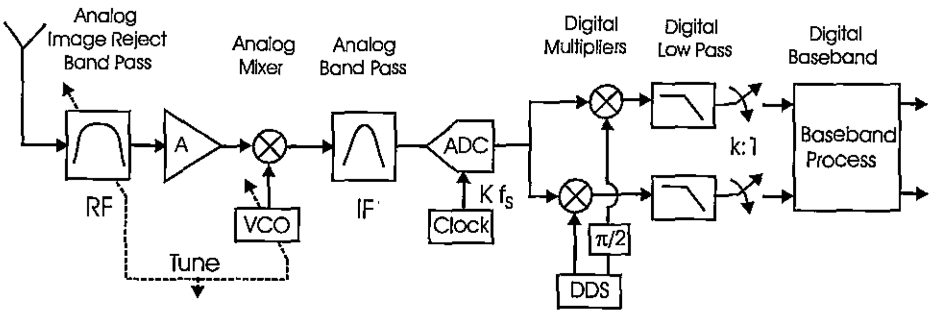


Figure 13.2 Second-Generation Radio Receiver Architecture

To control the computational workload, the filtering and down sampling is usually erformed in two stages, a 4- or 5-stage CIC filter that performs filtering without multiplicaions while performing an internal M-to-1 down sampling to an output rate of 4fs. The CIC s followed by two or more half-band filters that correct for the main lobe gain of the CIC while rejecting the main lobe spectral region containing significant aliased energy due to the M-to-1 down sampling. Figure 13.3 shows a possible realization of the standard second-generation receiver front end implemented by a digital down converter. The converter performs a cascade of simple operations consisting of a quadrature mixer driven by a direct digital synthesizer (DDS), a K-stage CIC filter capable of large integer resampling of M-to-1, a 2-to-1 half-band filter with CIC compensation, a second 2-to-1 half-band to finish the spectral control, and a course gain control.

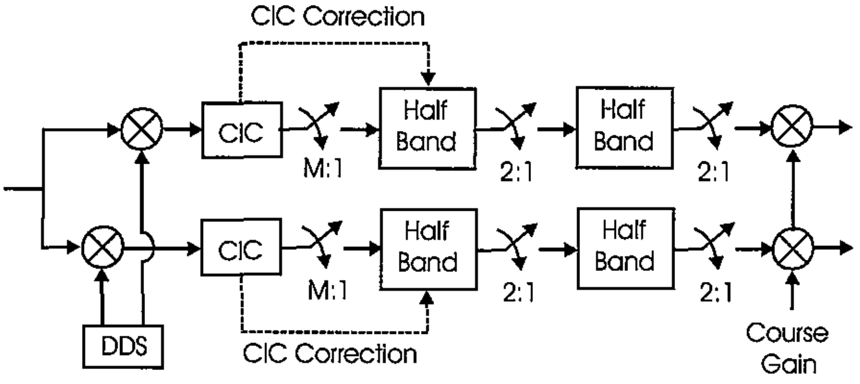


Figure 13.3 Standard Digital Down Converter with CIC and Half-band Filters

Figure 13.4 shows the log magnitude frequency response of a 5-stage CIC filter operating as a 10-to-1 resampling filter. The spectrum is shown at the input and output sample rates. The spectrum at the output rate shows the aliased main-lobe folding back into the main lobe spectral interval due to the 10-to-1 down sampling. The alias-free region, approximately one-fourth of the output sample rate, is extracted from the main lobe by the half-band filters following the CIC. These filters not only extract the desired bandwidth, they correct the droop in pass-band gain due to the curvature of the CIC main-lobe response. Shown, as an overlay on the main lobe, is the desired response of the half-band filters following the CIC filter.

Figure 13.5 shows the frequency response of the CIC filter main lobe along with the frequency response of the compensating first half-band filter and their composite response. Note the spectral peaking of the compensated spectrum beyond the pass-band edges. This peaking is suppressed by the frequency response of the second half-band filter following the compensating half-band filter. The compensating half-band filter was designed using the Remez algorithm with the minor modification that the pass-band gain is the reciprocal of the CIC gain over the normalized frequency interval 0 to 0.5. The gains were set over a grid spanning the normalized frequency interval, and the Remez algorithm does linear interpolation between the grid points. Rather than measure the CIC gain, we computed it at the desired grid points using the first three terms of its Taylor series as indicated in (13.1).

$$gg = (1 - \frac{1}{6}\theta^2 + \frac{1}{120}\theta^4 - \frac{1}{5040}\theta^6)^5 \qquad (13.1)$$

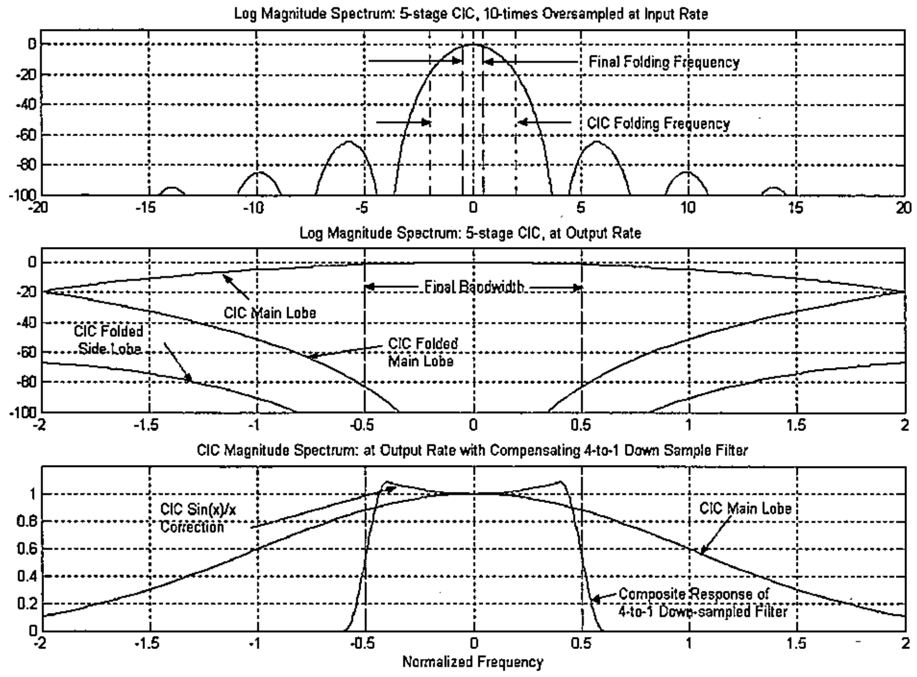


Figure 13.4 Frequency Response of 5-stage CIC Filter at Input Sample Rate, and at Output Sample Rate Illustrating Main-lobe Folding due to 10-to-1 Re-sampling, and Main-lobe Response with Overlaid Compensating 4-to-1 Down-sample filter

The MATLAB call to the Remez routine using this expression for the desired pass-band gain is shown here.

```
phi=[0.00  0.01  0.011  0.02  0.021  0.03  0.031…
           0.04  0.041  0.05  0.051  0.06  0.061…
           0.07  0.071  0.08  0.081  0.09  0.091  0.10];
phi=phi*pi;
tt=((1-(phi.^2)/6+(phi.^4)/120-(phi.^6)/1540)).^5);
hh=remez(20,[phi 0.4 0.5]/0.5,[1./tt  0  0]);
```

MATLAB Call for CIC Compensating Half-band Filter

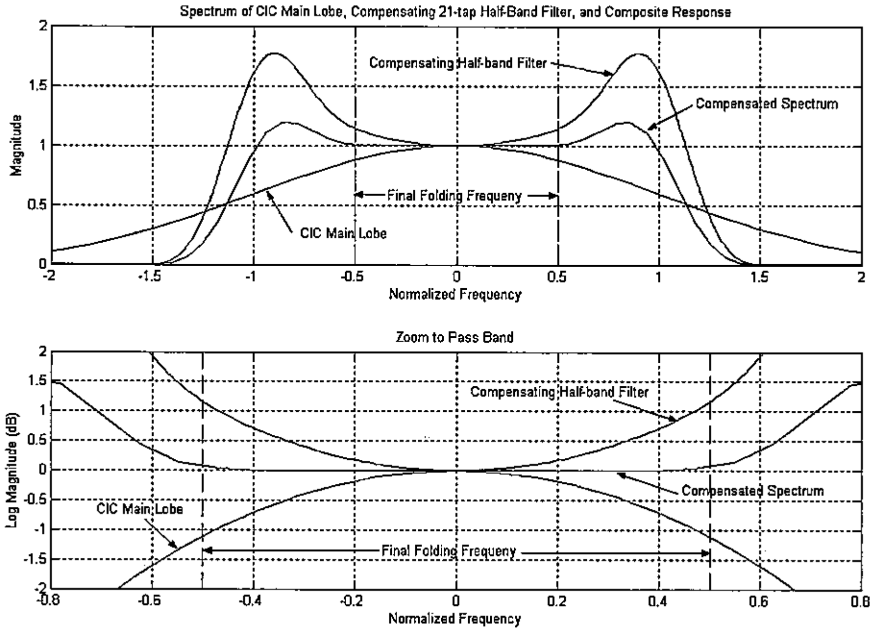


Figure 13.5 Spectra of CIC Main Lobe, of 21-tap Compensating Half-band Filter, and the Composite Response

13.2 Aliasing Digital Down Converters

In the conventional sampled data collection process, the sample rate is selected to satisfy (13.2) where 2 f_{BW} is the two-sided bandwidth of the analog anti-alias filter and f_Δ is the transition bandwidth of the same filter.

$$f_S = 2f_{BW} + f_\Delta \tag{13.2}$$

Signals collected from the output of the anti-alias filter at this data rate are said to satisfy the Nyquist sampling criterion. The Nyquist criterion is sometimes stated as: "The sample rate must be greater than twice the highest frequency of the input signal." This is an over restrictive or a narrow interpretation of the Nyquist criterion. The less restrictive interpretation is that the sample rate must exceed the two-sided bandwidth of the signal. This second interpretation is important when we have a narrow bandwidth signal centered on a high frequency carrier. In a digital receiver, the signal processing following the data collection process removes the carrier to extract the complex envelope of the narrowband signal on the carrier. A digital down-conversion process normally performs this task. Since the carrier frequency is discarded as part of the signal extraction, there is no need to preserve it during the data-sampling process. We are thus free to violate the Nyquist criterion for the carrier frequency as long as we satisfy the criterion for the bandwidth of its complex envelope.

This narrowband interpretation of the Nyquist criterion leads to an alternate data collection process known as subsampling or IF sampling. In this process the sample rate is selected to be less than the signal's center frequency to intentionally alias the center frequency to a lower frequency less than the sample rate. Since we are intentionally violating the Nyquist sampling criterion we must condition the analog signal to prevent multiple frequency intervals from aliasing to the same frequency location to which our desired signal component will alias. To minimize the cost of the analog signal-conditioning filter we arrange for the signal band of interest to alias to one-fourth of the selected sample rate. Aliasing to the quarter-sample rate maximizes the separation between the positive frequency alias and the negative frequency alias, which permits the maximum transition bandwidth of the analog band-pass filter. Aliasing the center frequency f_C to the quarter sample rate during the sampling process is assured if the sample rate satisfies (13.3). The k + 1/4 option aliases the signal to the positive quarter-sample rate while the k – 1/4 option aliases the signal to the negative quarter-sample rate. Use of the two options simply makes available a larger set of possible sample rates with either option equally acceptable.

$$f_C = k f_S \pm \frac{1}{4} f_S \tag{13.3}$$

To better understand the process, we now present an example of IF subsampling with aliasing to the quarter sample rate.

13.2.1 IF Subsampling Example

Signal 2-sided Bandwidth	10 kHz
Center Frequency	450 kHz
Signal Dynamic Range	80-dB
Output Sample Rate	20 kHz

First Option: Our first option is to sample the input signal at 2.0 MHz, placing the band of interest near the quarter-sample rate, then using a standard digital down converter we would perform a complex translation to baseband, filter and down sample 25-to-1 with a CIC, and then filter and down sample 4-to-1 with a pair of half-band filters.

Second Option: Our second option, the focus of this example, is to perform IF sampling at a sample rate satisfying (13.3). A list of possible sample rates that satisfy (13.3) for the center frequency of 450 kHz is presented in Table 13-1.

Table 13-1 List of Possible Sample Rates that Alias 450 kHz to Quarter-sample Rate

Integer k	**fs** for 450 = (k + 1/4) fs	**fs** for 450 = (k – 1/4)fs
0	1800	1800
1	360	600
2	200	257.143...
3	138.46...	163.37...
4	105.88...	120

We will select the 200-kHz sample for this example and suggest that the reader consider how the solution would be different had we selected 120-kHz as the sample rate. The aliasing caused by sampling the signal centered at 450 kHz at a 200-kHz sample rate is illustrated in Figure 13.6. Also note in this figure the transition bandwidth of the analog IF filter is maximized when the alias is to the quarter-sample rate. Figure 13.7 presents the same spectrum on a frequency-scaled periodic circle. Here we can start at 0 and travel in the positive direction around the 200-kHz circle passing the listed frequencies. After two encirclements, we see that frequency 450 kHz is located at the same location as 50 kHz. This is the desired result that we expected from the IF sampling process.

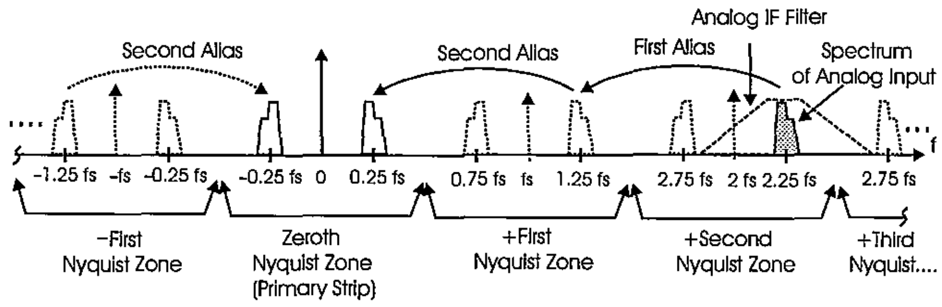


Figure 13.6 Aliasing Spectra at 2.25 fs in Second Nyquist Zone to 0.25 fs in Zeroth Nyquist Zone by IF Sampling at fs

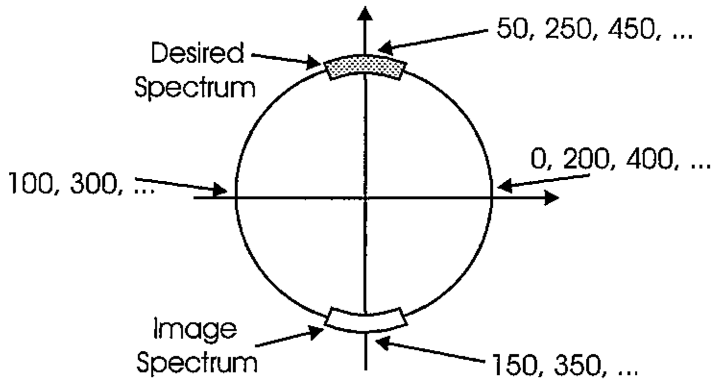


Figure 13.7 Periodic Spectra on Scaled Circle Showing How 450 kHz from Second Nyquist Zone Folds through 250 kHz in First Nyquist Zone to 50 kHz in Zeroth Nyquist Zone

At this point we could return to the standard digital down converter that can translate the spectral region to baseband with a complex heterodyne followed by a 10-to-1 resampling filter. This filter can be a 5-to-1 CIC followed by a 2-to-1 compensating half-band filter. The complex heterodyne required to move the spectrum from the quarter-sample rate is of the form shown in (13.4). The values of the cosine and sine terms in (13.4) are trivially the +1 and –1 and zero values shown in (13.5), so that in fact the heterodyne is trivial.

$$\exp(-j\frac{\pi}{2}n) = \cos(\frac{\pi}{2}n) - j \quad \sin(\frac{\pi}{2}n) \tag{13.4}$$

$$\exp(-j\frac{\pi}{2}n) = \{1, 0, -1, 0, \ldots\} + j\{0, -1, 0, 1, \ldots\} \tag{13.5}$$

We also note that due to the zero-value samples of the heterodyne, half the data samples in the cosine product are set to zero, and the complementary set of the data samples in the sine heterodyne are also set to zero. These zero-valued terms cannot contribute to the outputs of the filter following the heterodyne. Since we know the location of these zeros, we can discard them without error as long as we account for their effect in shifting the non-zero data samples through the filter. It would be a shame to waste the unique attributes of this heterodyne and data variation on a standard digital down converter. Instead, we continue this example with a number of techniques that take advantage of the signal being located at the quarter-sample rate.

The most obvious option to apply at this point is a 4-path polyphase filter to down sample 4-to-1 and thus alias the quarter-sample rate to DC. A phase rotator at the output of the 4-stage filter would unwrap the aliases and extract samples corresponding to the band centered at the quarter-sample rate. By setting the center frequency of the sines and cosines at the quarter-sample rate, the phase rotator sequence exp[j (π/2)r], for r = 0, 1, 2, and 3 is a particularly simple sequence of 1, j, – 1 and –j. Merging the phase-rotated outputs of the polyphase filter stages proves to be a particularly simple task of identifying real and imaginary output sample, and forming the difference of the two real paths and of the two imaginary paths. The 4-path polyphase filter using these rotators is shown in Figure 13.8.

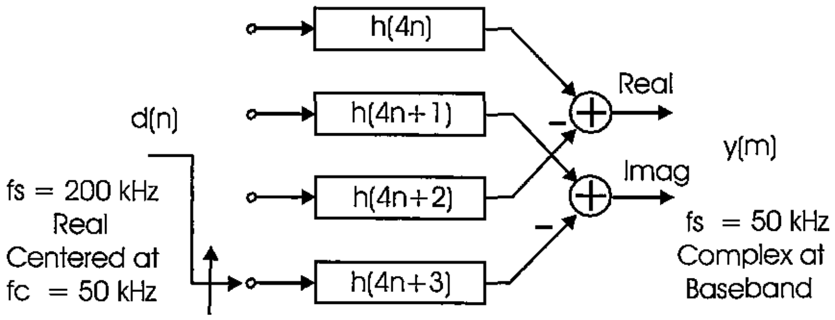


Figure 13.8 Four-path Polyphase Filter: Simultaneously Translates Frequency Band from Quarter-sample Rate to Baseband, Down Samples 4-to-1, and Converts Real Input to Complex Output

The remaining operation is to down sample the output of the 4-path polyphase filter from 50 kHz to the desired 20 kHz. This down-sampling task requires a 2.5-to-1 change in sample rate, which is accomplished as a 1-to-2 up sampling followed by a 5-to-1 down sampling. The 1-to-2 up sampling is performed conceptually by zero packing the input series 1-to-2. Of course the alternate zero-valued samples resulting from the zero packing cannot contribute to the filter output and, as observed earlier, can be discarded providing we account for their data shifting function in the 5-stage polyphase filter. Two successive cycles that deliver five zero-packed data samples to the 5-stage filter are shown in Figure 13.9.

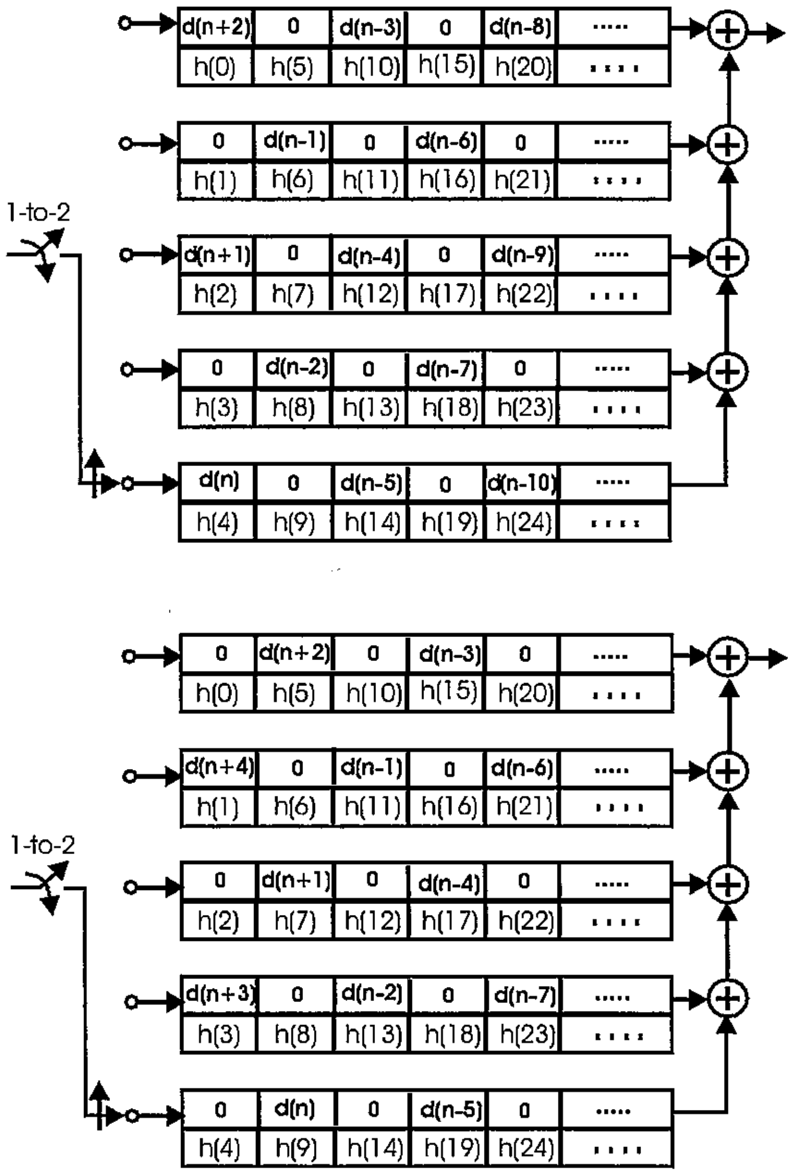

d(n+2) 0 d(n-3) 0 d(n-8) .....
h(0) h(5) h(10) h(15) h(20) ....
0 d(n-1) 0 d(n-6) 0 .....
h(1) h(6) h(11) h(16) h(21) ....
1-to-2
d(n+1) 0 d(n-4) 0 d(n-9) .....
h(2) h(7) h(12) h(17) h(22) ....
0 d(n-2) 0 d(n-7) 0 .....
h(3) h(8) h(13) h(18) h(23) ....
d(n) 0 d(n-5) 0 d(n-10) .....
h(4) h(9) h(14) h(19) h(24) ....
0 d(n+2) 0 d(n-3) 0 .....
h(0) h(5) h(10) h(15) h(20) ....
d(n+4) 0 d(n-1) 0 d(n-6) .....
h(1) h(6) h(11) h(16) h(21) ....
1-to-2
0 d(n+1) 0 d(n-4) 0 .....
h(2) h(7) h(12) h(17) h(22) ....
d(n+3) 0 d(n-2) 0 d(n-7) .....
h(3) h(8) h(13) h(18) h(23) ....
0 d(n) 0 d(n-5) 0 .....
h(4) h(9) h(14) h(19) h(24) ....

Figure 13.9 Indexing of Successive Zero-packed Data Samples to 5-path Polyphase Filter

Note that in the first cycle we deliver three non-zero samples d(n), d(n+1), and d(n+2) and then compute an output. In the second cycle we deliver two non-zero samples d(n+3) and d(n+4) and then compute the next output. Over two successive input cycles, we deliver 5 inputs and compute 2 outputs for a resampling ratio of 5-to-2 or 2.5-to-1. The inserted zero-valued samples simply move the data samples in their paths one data sample to the right. We can omit the inserted zeros by following the 2-cycle state machine rules listed in Table 13-2.

Table 13-2 Two-state State Machine Input and Inner Product Schedule for 5-path 2.5-to-1 Resampling Filter

Cycle-0	Cycle-1
Input data: d(n) in path-4 d(n+1) in path-2 d(n+2) in path-0	Input data: d(n+3) in path-3 d(n+4) in path-1
Inner products: path-4, data with h(10n+4) path-3, data with h(10n+8) path-2, data with h(10n+2) path-1, data with h(10n+6) path-0, data with h(10n)	Inner products: path-4, data with h(10n+9) path-3, data with h(10n+3) path-2, data with h(10n+7) path-1, data with h(10n+1) path-0, data with h(10n+5)

Figure 13.10 shows the block diagram of the down converter using undersampling to alias the narrowband spectrum to the quarter-sample rate and then a 4-path polyphase filter to alias it again to baseband for a final down sample of 5-to-2 with a 5-path polyphase filter. The first filter requires 16 coefficients, which means that each of the 4-paths require inner products with 4-coefficients while the second filter requires 40 coefficients that are split evenly between the two cycles of the state machine so that each of the 5-paths also require inner products with 4-coefficients.

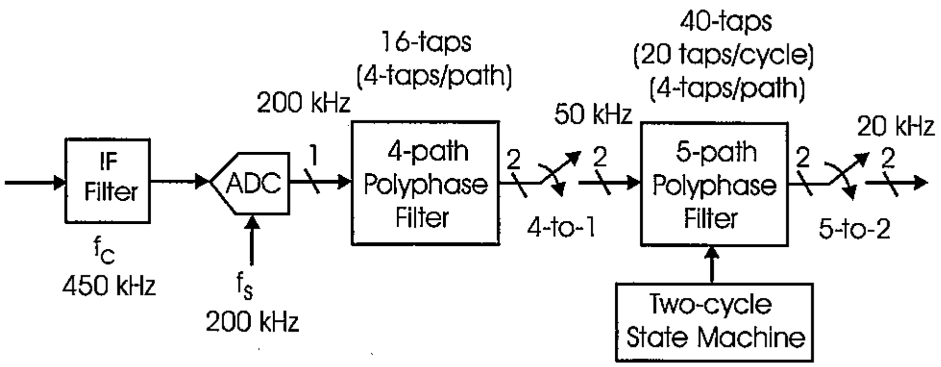


Figure 13.10 Signal Processing Structure of Down Converter Using Aliasing to and from Quarter-sample Rate by Subsampling and Polyphase Filtering

Third Option: In the example we are using to demonstrate subsampling we selected a sample rate of 200 kHz to translate the 450-kHz IF signal to the quarter-sample rate of 50 kHz. We selected the 200 kHz for the ease with which we could perform the remainder of the processing task, filtering and down sampling by a factor of 10 from 200 kHz to 20 kHz. We accomplished the 10-to-1 down sampling in two steps, first the 4-to-1 in the 4-path polyphase filter and then a 5-to-2 in the 5-path polyphase filter. We were guided to the 4-to-1 down sampler because it was natural to alias the quarter sample rate to baseband in this stage. A signal at any center frequency can be aliased to other center frequencies beside baseband under the resampling operation. The quarter-sample rate is quite remarkable in that it is only aliased to the four cardinal directions by other resample ratios besides the obvious 4-to-1. We now consider another option for this example in which we use a 5-to-1 first stage filter followed by a 2-to-1 second stage filter.

We can design a prototype low pass filter that will support a 5-to-1 resampling operation in a 5-path polyphase filter. In the standard way we would apply this filter, we would permit multiples of the output sample rate to alias to baseband where phase rotators would unwrap the desired alias. The problem here is that the signal of interest resides at 50 kHz, which is not one of the multiples of 40-kHz, so the standard polyphase structure appears to be inappropriate for this problem. By a slight modification of how we use the polyphase filter we are able to take advantage of the unique attributes of the quarter-sample rate center frequency. In particular, we translate the prototype low pass filter to the quarter-sample rate prior to the 5-path polyphase partition. We accomplish this translation by the trivial heterodyne of the filter coefficients with the terms $\exp[j(\pi/2)n]$. These heterodyne terms, the periodic sequence $\{1, j, -1, -j\}$, merely guide the output samples to the real or imaginary output and at most effect a sign change in the weighted sum performed in the filtering operation. This form of the filter is a close cousin of the Hilbert transform filter.

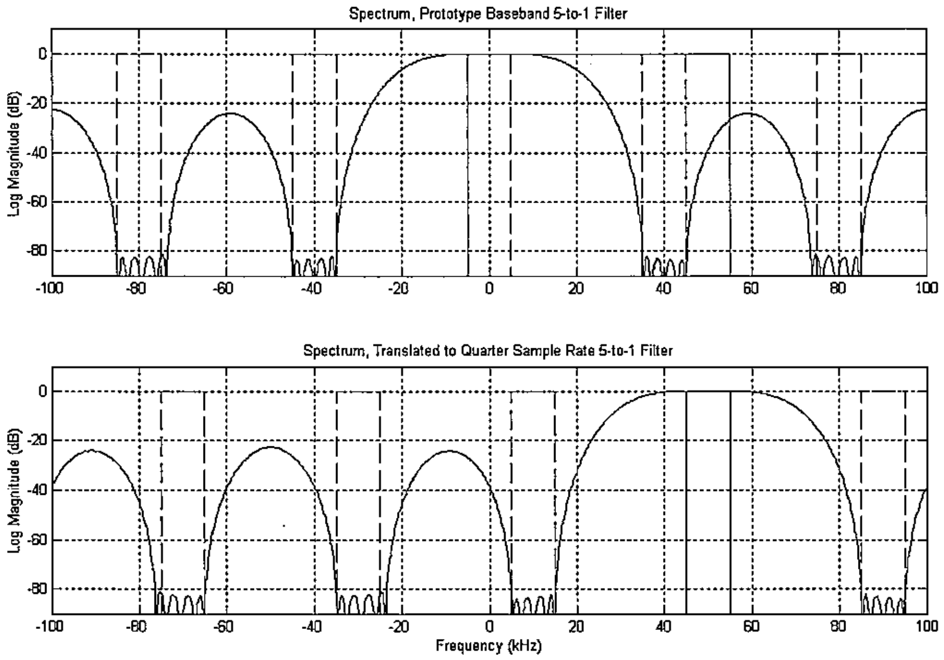


Figure 13.11 Spectrum of 25-tap Prototype Low pass Filter with Frequency Bands that Alias to Baseband when Resampled 5-to-1 (Upper Figure) and Spectrum of Same Filter Translated to Quarter-sample Rate with Frequency Bands that Alias to Quarter-sample Rate when Resampled 5-to-1 (Lower Figure)

When the filter is used in this manner, as a complex polyphase band-pass filter, the resampling operation exhibits an interesting property. This property is that a signal at the quarter-sample rate will always alias to a multiple of the quarter-sample rate under any integer-resampling operation. In particular, for our example, we have a signal centered at 50 kHz and sampled at 200-kHz so that it is located at the quarter-sample rate when it enters the 5-path polyphase filter. Here the signal is resampled 5-to-1 to 40 kHz which aliases the 50 kHz centered signal to 10-kHz, the quarter-sample rate at the output of the filter. Figure 13.11 shows the spectrum of a 25-tap baseband filter with the frequencies that alias to baseband under a 5-to-1 resampling as well as the translated to the input quarter-sample rate filter with the frequencies that alias to the output quarter sample rate when resampled 5-to-1.

We note that the sampled data sequence that enters the 5-path filter is real while the sampled data sequence that leaves the filter is complex and centered at the quarter-sample rate. This output sequence is heterodyned to zero by the trivial heterodyne $(-j)^n$ prior to final processing in the half-band filters. Figure 13.12 presents the block diagram of the signal processing performed by this option.

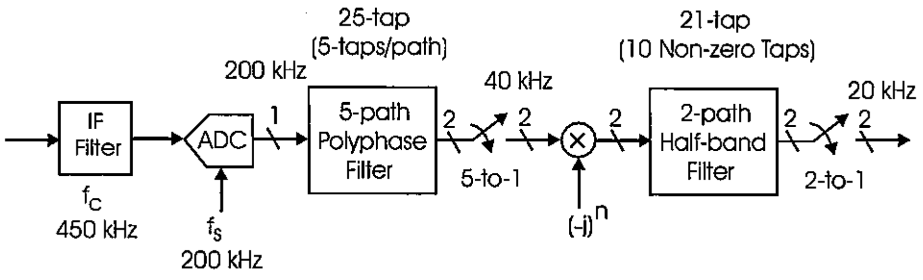


Figure 13.12 Signal Processing Structure of Down Converter Using Aliasing to Quarter-sample Rate by Subsampling and Again by 5-path Polyphase Filter

13.3 Timing Recovery in a Digital Demodulator

Multirate filters have had significant impact on communication systems, often replacing analog prototype systems with DSP based implementations. Filtering for bandwidth control and quadrature mixing for spectral translation are the obvious examples of this digital insertion. A second instance of this process is the timing recovery mechanism in digital communication receivers. In traditional analog systems, the phase of the sampling clock is controlled by a phase lock loop to obtain phase alignment with the symbol epochs in the received modulation signal. In modern digital systems, the phase alignment occurs not by moving the sample instances to the correct position in the time waveform, but rather by interpolation the waveform samples from the collected sample locations to the desired sample locations.

13.3.1 Background

The modulator of a communication system transmits a sequence of scaled and time translated band-limited waveforms of the form shown in (13.6). The scale factors a_n are selected from a finite list of permitted amplitudes in response to a sequence of binary words delivered at a periodic rate to the modulator. The alphabet may, for instance, contain two elements such as +1 and −1 selected by a 1-bit input, or it may contain four elements such as +3, +1, −1, and −3 selected by a 2-bit input, and so on.

$$d(t) = \sum_n a_n h(t - nT) \tag{13.6}$$

In an ideal communication system, the demodulator receives a version of the transmitted waveform that has been corrupted by additive white Gaussian noise (AWGN) as shown in (13.7).

$$d(t) = \sum_{n} a_n h(t-nT) + \mathcal{N}(t) \tag{13.7}$$

To minimize the effects of the received noise in the ensuing decision process, the received noisy signal is passed through a matched filter. The impulse response of the matched filter is, as shown in (13.8), a time reversed and delayed version of the transmitted waveform h(t). Here we assume the wave shape extends over k symbol durations.

$$g(t) = h(kT - t) \tag{13.8}$$

The convolution process performs a running weighted average with the filter's time-reversed impulse response. We purposely time-reversed the filter impulse response in anticipation of the time reversal that will occur in the convolution process. The reversal in the convolution process undoes the initial reversal so that the running weighted average is performed with a replica of the transmitted waveform and the convolver performs a replica correlation. The peak of the correlation function corresponds to the projection of the noisy signal on a noiseless replica of the signal, and this projection exhibits the maximum SNR that can be obtained by any processing of the received signal. In order for the receiver to access these peak values it invokes a timing recovery mechanism to align the frequency and phase of its sampling clock with the epochs of the correlation peaks. The signal flow of the entire process is shown in Figure 13.13.

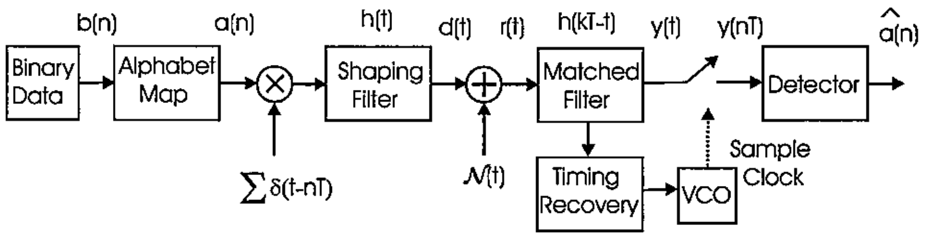


Figure 13.13 Signal Flow in Modulator and Demodulator for Communicating Through Band-limited AWGN Channel

In modern receivers, the matched filter operation is performed in the sampled data domain. The receiver structure changes slightly to accommodate the anti-aliasing filter, the S&H, and the ADC. A first-generation digital demodulator structure is shown in Figure 13.14. Note here that the timing recovery is now controlling the sampling clock to the digital matched filter rather than the sampling at the output of the analog matched filter. The sampling performed at the input to the digital filter must satisfy the Nyquist criterion for the collected wave shape, and in many systems the sample rate is two samples per symbol. The sampling performed at the output of the analog matched filter to supply samples to the detector occurs at the symbol rate or one sample per symbol.

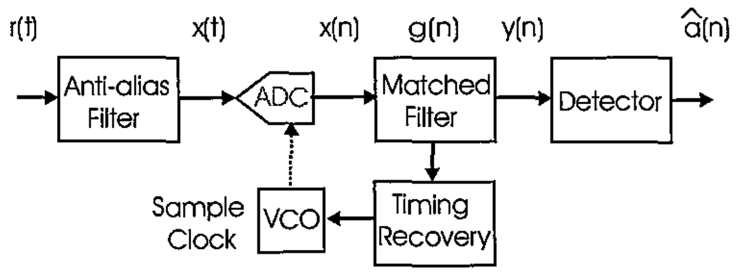


Figure 13.14 Signal Flow in First-generation Digital Demodulator

The timing recovery process must ascertain if the clock sample position is at the correct position or should be advanced or retarded relative to the input time waveform. It does this by posing the question, "How do I know that I am at the local peak of the correlation function?" It knows that at the peak, the correlation function has a zero derivative, so it poses the nearly equivalent question, "What is the derivative of the correlation function zero at this sample time?" Traditionally, auxiliary matched filter outputs, called early and late gates, that are time advanced and time delayed relative to the sample test point, supply an answer to this question as the average difference between the early and late gate output values. This can be visualized in Figure 13.15.

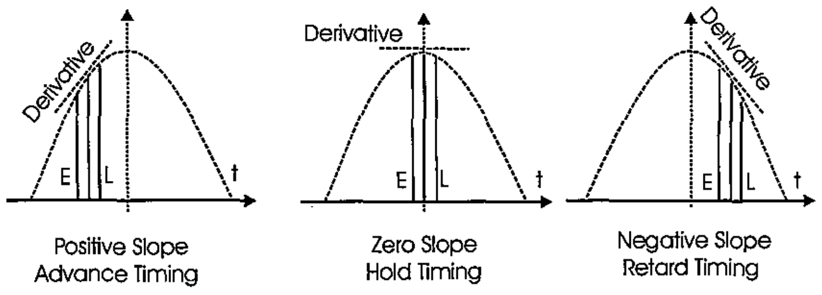


Figure 13.15 Three Samples, Early, Punctual, and Late Samples, on Correlation Function for Three Operating Conditions. (1) Positive Slope: Peak Is Ahead, (2) Zero Slope: At Peak, and (3) Negative Slope: Peak Is Behind

The derivative alone contains insufficient information to determine if the timing should be time advanced, held, or retarded. A conditional piece of information is missing, and this information is the answer to the question "What is the sign of this sample of the correlation function?" As seen in Figure 13.16, a sample set formed prior to the peak will generate a positive slope if the correlation values are positive but will generate a negative slope if the correlation values are negative.

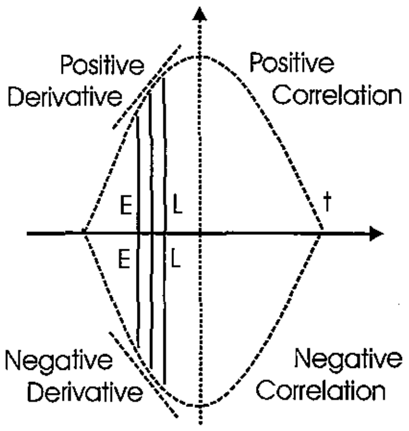


Figure 13.16 Three Samples, Early, Punctual, and Late Samples on Positive and on Negative Correlation

The conditional information is folded into the observable error parameter in the timing recovery process as the products shown in (13.9). When the SNR, the sign of the data sample y(t), is reliable, the sign[y(t)] is a sufficient modifier. When the SNR is small, the modifier is the data itself rather than the sign of the data. The maximum likelihood timing recovery process minimizes the product of the derivative matched filter output $\dot{y}(t)$ with SNR-conditioned selection of the matched filter output y(t) or sign of the matched filter output sign[y(t)]. Many systems do not bother switching modes and simply use $y(t)\dot{y}(t)$ for the full range of SNR.

$$\begin{aligned} e(t) &= \dot{y}(t)\,\mathrm{sign}[y(t)] \cong [y(t+\Delta t) - y(t-\Delta t)]\cdot \mathit{sign}[y(t)], && \text{High SNR} \\ e(t) &= \dot{y}(t)\,y(t) \cong [y(t+\Delta t) - y(t-\Delta t)]\cdot y(t), && \text{Low SNR} \end{aligned} \tag{13.9}$$

13.3.2 Modern Timing Recovery

Examining Figure 13.14 we note that the timing recovery process controls the phase and frequency of the voltage controlled oscillator (VCO) that is supplying the sampling clock to the ADC. It is likely that the error detector, the $y(t)\dot{y}(t)$ of (13.9), is performed in the sampled data domain as $y(n)\dot{y}(n)$ and that the loop filter that controls the transient and steady state behavior of the timing recovery system is also performed in the sampled data domain. The control signal formed to operate the VCO resides, as a set of numbers in the sampled data domain, while the control signal required to operate the VCO is an analog voltage level. To convert the digital control signal to an analog control signal requires a high-precision DAC, a low-bandwidth analog filter, and a bus and control mechanism to transfer the control words to the DAC's internal register.

Rather than incur the overhead of changing between the sampled data and continuous domains, we can perform the entire timing recovery process in the sampled data domain. We can accomplish this in two different ways. We can either move the data samples to be aligned with the filter coefficients, or we can move the filter coefficients to be aligned with the data samples. In the first method, we use an interpolator to raise the input sample rate by a factor of M, say 32, and then down sample back to the same input rate with a fixed time offset to the sample locations required to be aligned with the impulse response of the matched filter. Figure 13.17 presents the signal flow for this demodulator architecture. In the second method we increase the sample rate of the matched filter and then resample the filter response to the original sample rate with successive time offsets of 1/M, 2/M, 3/M, etcetera, to form a set of M filters matched to different time offsets between the input sample location and the envelope of the received waveform. Figure 13.18 presents the signal flow for this demodulator structure. In both cases we are required to operate only one filter path out of the M-possible paths. In this mode, the interpolator is used as a 1-to-M up sampler followed by an M-to-1 down sampler with the desired time offset between input and output sample locations. The timing recovery process determines which filter path is the one required to align the filter with the signal time offset.

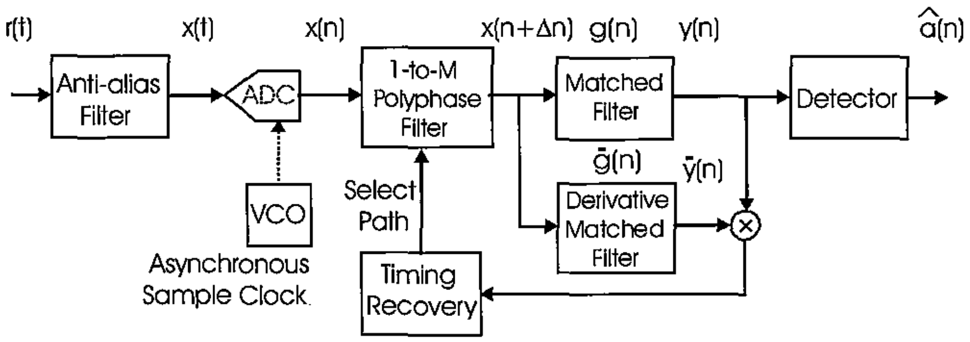


Figure 13.17 Signal Flow for Timing Recovery with Polyphase Interpolator Processing and Shifting Asynchronous Samples to Desired Time Locations

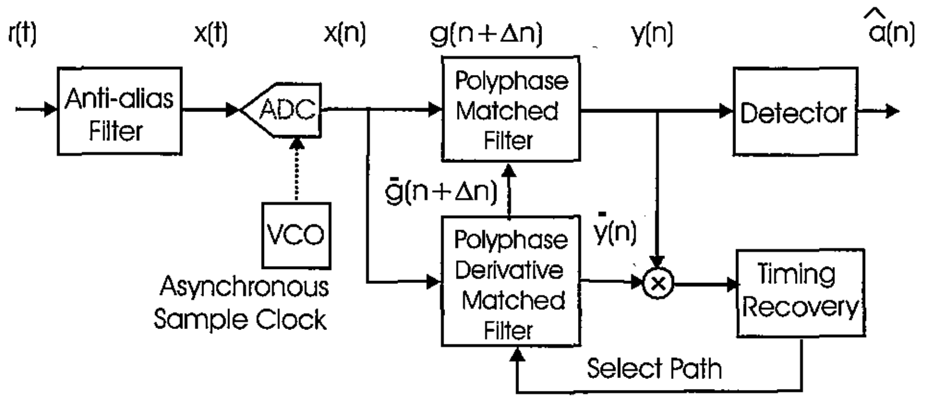


Figure 13.18 Signal Flow for Timing Recovery with Polyphase Matched Filter. Timing Recovery Selects Matched Filter Path Aligned with Input Sample Positions.

In Figure 13.15 we used the analog perspective that used an early and a late gate to determine the derivative at the output of the matched filter. Doing so requires the operation of three filters: early, punctual, and late. Note that the filters corresponding to the early and late gate filters are trivially the polyphase segments (k – 1) and (k + 1) when testing polyphase segment (k). This is seen in Figure 13.19. As shown in (13.10), the same data is convolved with filters (k – 1) and (k + 1) to form corresponding outputs, which we then subtract to form the derivative estimate. Factoring out the common data, we are left with a filter with coefficients obtained as the difference of the two adjacent filters. We recognize this single filter as one that directly computes the desired derivative. We can now replace the early and late gate filters with the derivative filter and thus reduce the workload required to implement control data for the timing recovery process. Figure 13.20 shows a process using two banks of polyphase filters, one for the signal and one for the derivative, while Figure 13.21 shows a simple structure with two simple filters with a coefficient selection process. It is this later structure that is embedded in the block diagrams shown in Figures 13.17 and 13.18.

$$\begin{aligned}\dot{y}(n+k/M) &= y(n)*h_{k+1}(n) - y(n)*h_{k-1}(n)\\ &= y(n)*[h_{k+1}(n) - h_{k-1}(n)] \qquad (13.10)\\ &= y(n)*\dot{h}_k(n)\end{aligned}$$

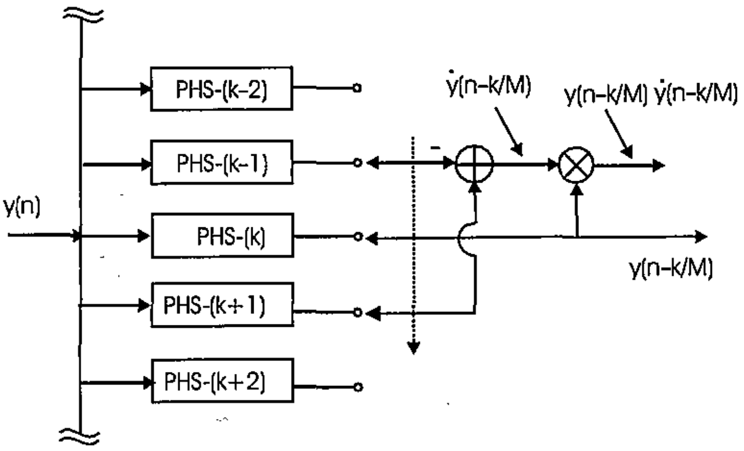


Figure 13.19 Early, Punctual, and Late Gate Filters for Timing Recovery Control Signals

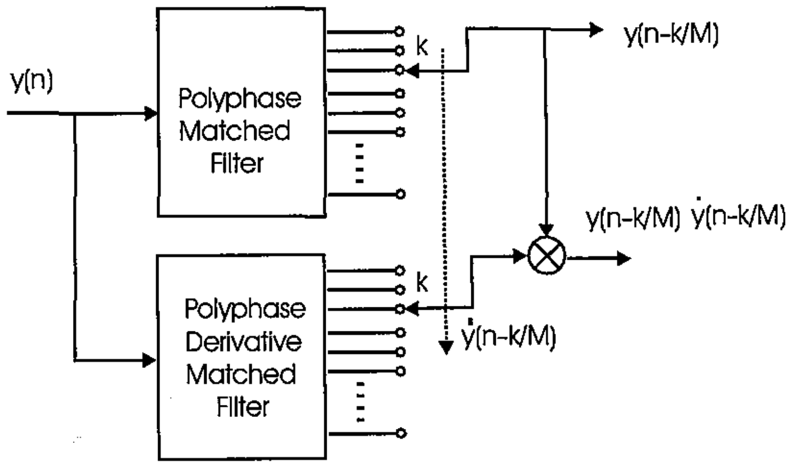

y(n)
Polyphase Matched Filter
Polyphase Derivative Matched Filter
k
k
y(n-k/M)
$y(n-k/M)\,\dot{y}(n-k/M)$
$\dot{y}(n-k/M)$

Figure 13.20 Two Polyphase Filter Banks Forming Filter and Derivative Filter Outputs

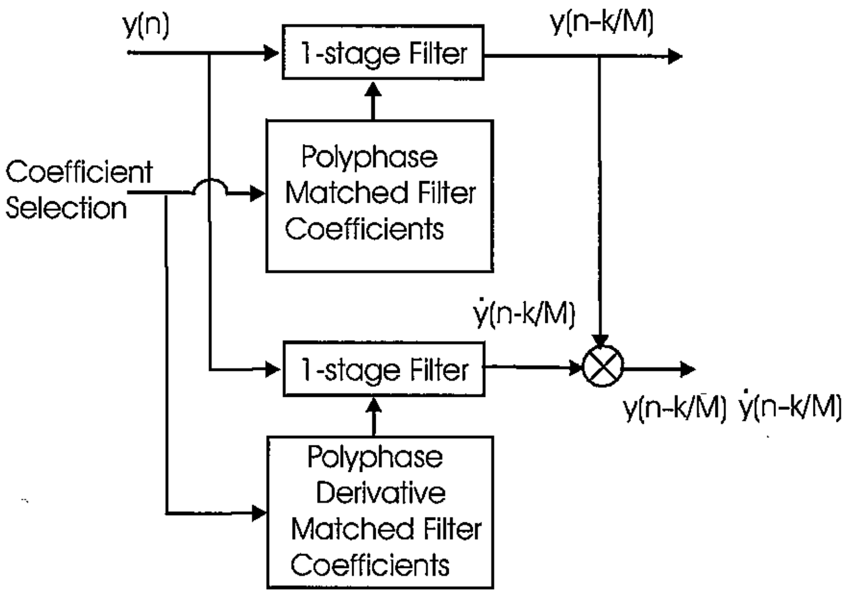
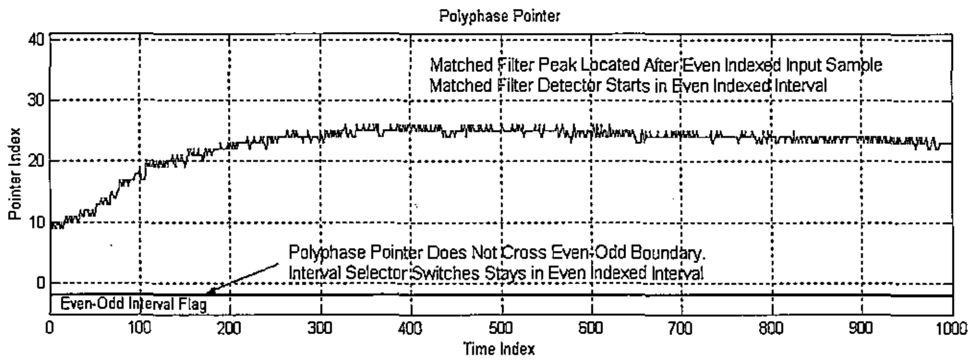

y(n)
1-stage Filter
y(n-k/M)
Polyphase Matched Filter Coefficients
Coefficient Selection
$\dot{y}$(n-k/M)
1-stage Filter
y(n-k/M) $\dot{y}$(n-k/M)
Polyphase Derivative Matched Filter Coefficients

Figure 13.21 Two Single-stage Filters with Selection of Coefficient Sets from Polyphase Filter Bank for Matched Filter and Derivative Matched Filter

Figure 13.22 shows the time history of the address pointer in a modem using a 40-path polyphase-matched filter for timing recovery. Since the input signal is collected at two samples per symbol, one sample, the one with an even index for instance, is chosen as a data sample, and the other is the non data sample sometimes called the timing sample. The data sample is the one delivered to the detector. If the address pointer tries to cross the address boundaries, an address lower than 1 or greater than 40, the address wraps circularly and the input sample identified as the data sample is switched to the odd index. When this happens, the pointer resets from index 0 in interval 1 to index 40 in interval 2. You can see this pointer offset in the lower half of Figure 13.22. Beyond the scope of this discussion is the observation that there may also be data sample insertion (data stuff) or deletion (data skip) at the boundary crossing depending on the direction of the crossing and whether the data sample is an even or odd index input sample.



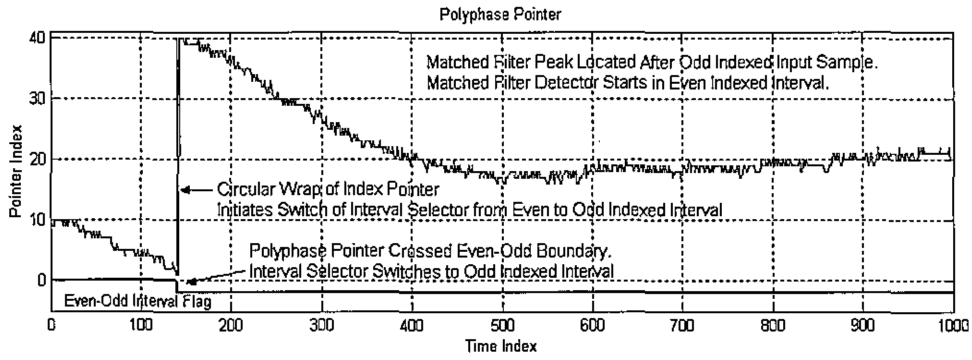


Figure13.22 Polyphase Address Pointer for Timing Recovery Loop with 40-path Filter and Two Samples per Symbol. In Upper Figure, Pointer Does Not Cross Address Pointer Boundary: In Lower Figure, Pointer Crosses Address Pointer Boundary and Converts Address 0 to Address 40 and Switches from Even-indexed to Odd-indexed Data Sample.

13.4 Modem Carrier Recovery

Many communication systems use a complex heterodyne to translate a complex baseband communication signal to a center frequency in the band allocated to the particular radio service. We perform spectral translation to different center frequencies for a number of reasons. One reason is the sharing of the radio spectrum by multiple users through use of frequency division multiplexing. Another reason is the relative ease of designing analog circuits when the signal bandwidth occupies a small fraction of the center frequency. Yet another reason is access to smaller or more efficient antennae at higher frequencies as well as access to spectral bands with specific propagation characteristics.

The transmitter in the communication system performs the spectral translation of the baseband signal to the selected carrier center frequency by an up-conversion process. To cull the desired signal from the numerous channels sharing the spectral region the receiver must invert the spectral translation process. In order to accomplish this task, the receiver requires a phase-coherent replica of the unmodulated carrier delivering the signal. In early radio systems, a copy of the carrier was embedded in the modulated signal to enable the down-conversion process. In modern radios the receiver forms its own copy of the carrier from side information residing in the modulated signal. This process is called carrier recovery.

13.4.1 Background

At the simplest level, modulation is simply the translation of a complex baseband modulation envelope m(t) with baseband spectral characteristic M(ω) to an arbitrary center frequency called the carrier. The relationship between the time and frequency descriptions of the baseband and translated signal is shown in (13.11).

$$\begin{aligned} m(t) &\Leftrightarrow M(\omega) \\ m(t)\exp(j\omega_C t) &\Leftrightarrow M(\omega-\omega_C) \end{aligned} \tag{13.11}$$

Product modulators that modify the amplitude and phase of the carrier perform this translation. To form the signal required for the actual transmission, the conjugate of the complex waveform is added to the expression shown in (13.11) to make the signal real. As shown in (13.12), the real signal is formed as the Cosine heterodyne of the real part of the complex envelope minus the Sine heterodyne of the imaginary part of the complex envelope. The real and imaginary parts of the complex waveform are normally called the In-Phase and Quadrature Phase or the I and Q components. These I, Q designations are inherited from the phase of their respective carriers.

$$\begin{aligned} s(t) &= m(t)\exp(j\omega_C t) + m^*(t)\exp(-j\omega_C t) \\ &= 2\,\mathrm{RL}[m(t)]\cos(\omega_C t) - 2\,\mathrm{IM}[m(t)]\sin(\omega_C t) \\ &= I(t)\cos(\omega_C t) - Q(t)\sin(\omega_C t) \end{aligned} \tag{13.12}$$

The narrowband signal is launched through the channel by the transmitter and is delivered through a noisy channel at the receiver. The receiver must remove the complex envelope from the carrier and present the noisy versions I(t) and Q(t) to the demodulator. To do so it must align the frequency and phase of its local oscillator to match the frequency and phase of the carrier in the received signal. The oscillator must do this in spite of temperature variation of its frequency-dependent components, in spite of manufacturing tolerance spread, in spite of Doppler-related frequency offsets due to a velocity component between platforms, and in spite of the fact that there is no spectral line at the carrier frequency in the received signal. It accomplishes this feat with a carrier recovery loop formed around a phase locked loop (PLL). The loop is composed of a phase detector operating on demodulated data, a loop filter, and a controlled oscillator. The high-level block diagram of the up conversion and down conversion performed at the two ends of the channel is shown in Figure 13.23.

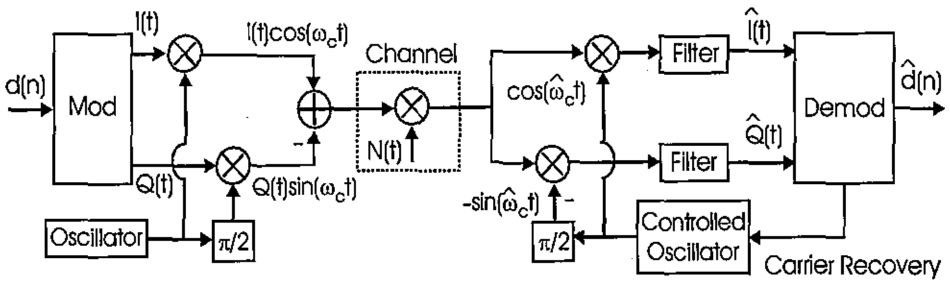


Figure 13.23 Block Diagram of Quadrature Up Converter at Transmitter and Quadrature Down Converter at Receiver.

The PLL has two distinct modes of operation: acquisition and tracking. The tracking bandwidth of a PLL, the range of frequencies over which it can follow frequency offsets, is limited by the control range of the oscillator. The acquisition bandwidth is limited by the pass band bandwidth of the loop filter. The acquisition range is considerably smaller than the tracking range. A receiver is directed to acquire an RF signal by being tuned to the expected center frequency of the signal.

Previous generation, or legacy design, receivers operate in the following manner: If the frequency offset between the local oscillator and the received center frequency is less than the acquisition bandwidth, the carrier recovery loop is able to generate a control signal to shift the local oscillator in the direction to servo the offset to zero, thus accomplishing the acquisition task. If the frequency offset is greater than the acquisition bandwidth, a lock detection signal will not be generated in the time-out interval that monitors the operating state of the receiver. The receiver enters a *failure to acquire* state in which it invokes an acquisition aid. The aid conducts a search for the signal by directing the controlled oscillator to perform a frequency sweep through the expected range of frequency offsets. When the frequency difference between the local oscillator and the errant signal becomes smaller than the acquisition bandwidth, the loop successfully acquires the signal, generates the lock detection signal, and disables the acquisition aid.

13.4.2 Modern Carrier Recovery

A modern receiver will use a maximum likelihood frequency estimator as an aid to pull the local oscillator close to the center frequency of an offset signal and thus bring the offset signal within the acquisition range of the PLL. The derivation of the maximum likelihood frequency estimator is beyond the scope of this text, so we will simply describe its operation and then its implementation. In a manner similar to the maximum likelihood timing-recovery process, we form a filter that is the derivative of the matched filter. In the timing-recovery process we took the time derivative of the matched filter, and then drove the product of the matched filter and the time-derivative matched filter outputs to zero. In the frequency-recovery process we take the frequency derivative of the matched filter and in a similar manner we can drive the product of the matched filter and the frequency-derivative matched filter outputs to zero. The frequency derivative matched filter is called a band-edge filter, and we now describe why it is so called and describe a common variant of its operation.

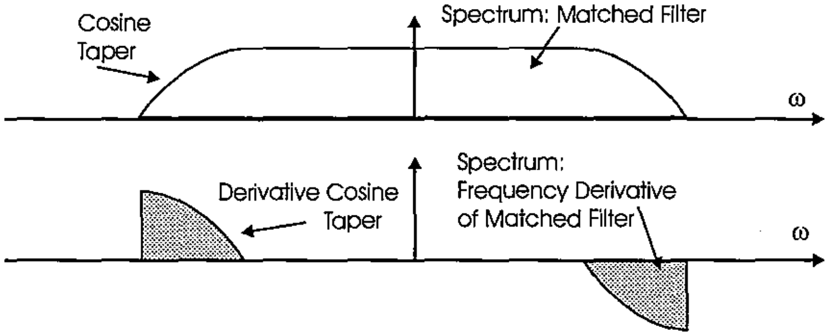


Figure 13.24 Spectra of Matched Filter for Square-root-Nyquist Filter and Frequency-Derivative Matched Filter

Figure 13.24 presents the power spectrum of the matched filter for a square-root-Nyquist filter as well as the frequency derivative of the same filter. Note that the derivative is zero everywhere except in the transition band of the filter. Seeing that the non-zero spectral response of the derivative-matched filter spans the band edges of the matched filter, we can readily understand why it is called a band-edge filter.

Figure 13.25 illustrates how the band-edge filter responds to input signals with significant frequency offsets. The top two figures present the spectrum of an input signal without a frequency offset and the response of the two spectral segments in the band-edge filter. We note that the two band-edge segments contain the same spectral energy. By comparison, the bottom two figures present the spectrum of an input signal with a small frequency offset and the response of the two spectral segments in the band-edge filter. Here we note that, as a result of the spectral shift to the right, the energy contained in the filter segment on the right increased while the energy contained in the filter segment on the left decreased. Energy difference in the two band-edge segments is a simple and accessible observable that can be used to drive the input spectrum to be centered at baseband. The modification to the receiver that incorporates the band-edge filter is shown in Figure 13.26. The polyphase filter following the band-edge filter separates the two band-edge spectral regions, and their outputs become the frequency error detector as the difference of their squared magnitudes.

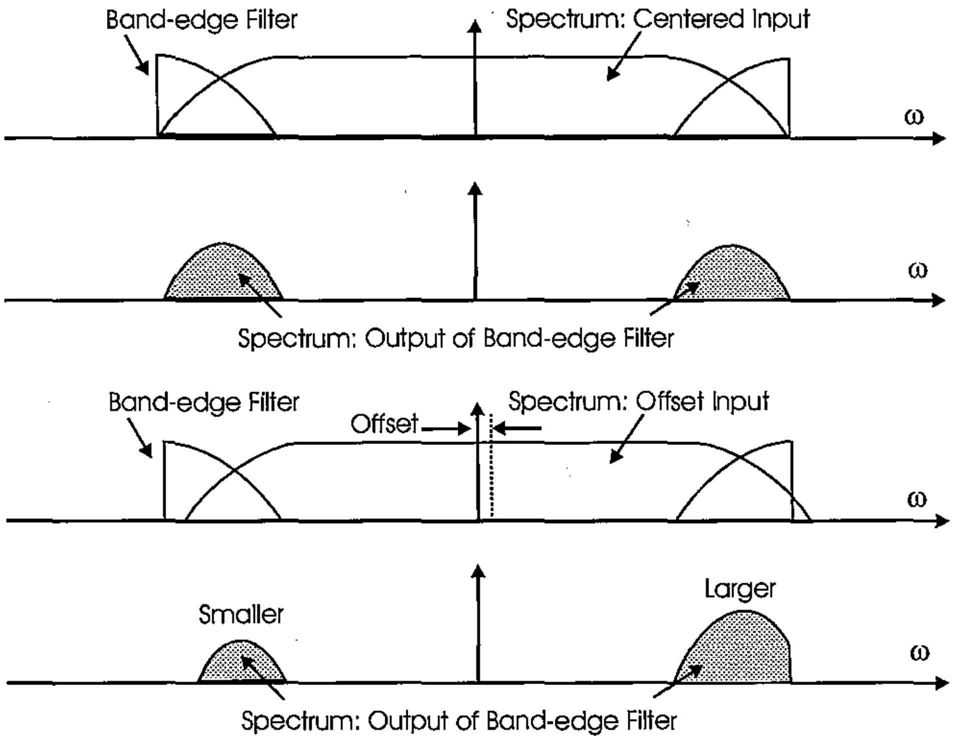


Figure 13.25 Spectra at Input and Output of Band-edge Filter for Centered Input Signal and for Input Signal with Spectral Offset

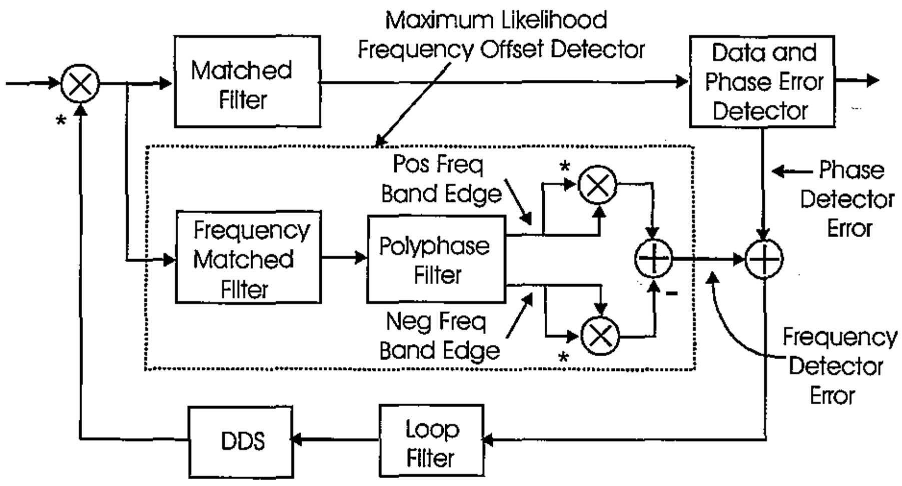


Figure 13.26 Band-edge Filter and Frequency Error Detector Appended to Carrier Recovery Process

13.4.2.1 Design and Partition of Band-edge Filter

We now address the task of designing the band-edge filter and the polyphase filter that separate the positive and negative frequency spectral segments. The first task of designing and implementing the band-edge filter is trivial. In fact, the band-edge filter is essentially free because its desired shape and spectral location resides in the complement of the matched filter. If the matched filter is designed to operate at two samples per symbol, we partition the matched filter into a two-path polyphase filter to form both the low pass, or matched filter and the high pass filter. The overlapped transition band of the high pass filter has the desired frequency-matched spectral response. The flat response region of the high pass filter allows adjacent channel spectra into the band-edge filter response that has to be eliminated in the next stage processing. The spectra presented to the polyphase matched filter pair and the output spectra obtained from this partition are shown in Figure 12.27. Note the presence of the adjacent channel's remnants partially suppressed, but not fully rejected, by prefiltering prior to the matched-filter processing. We note that while the complementary matched filter forms the desired band-edge filter, the spectrum is corrupted by remnants of the adjacent channel spectra.

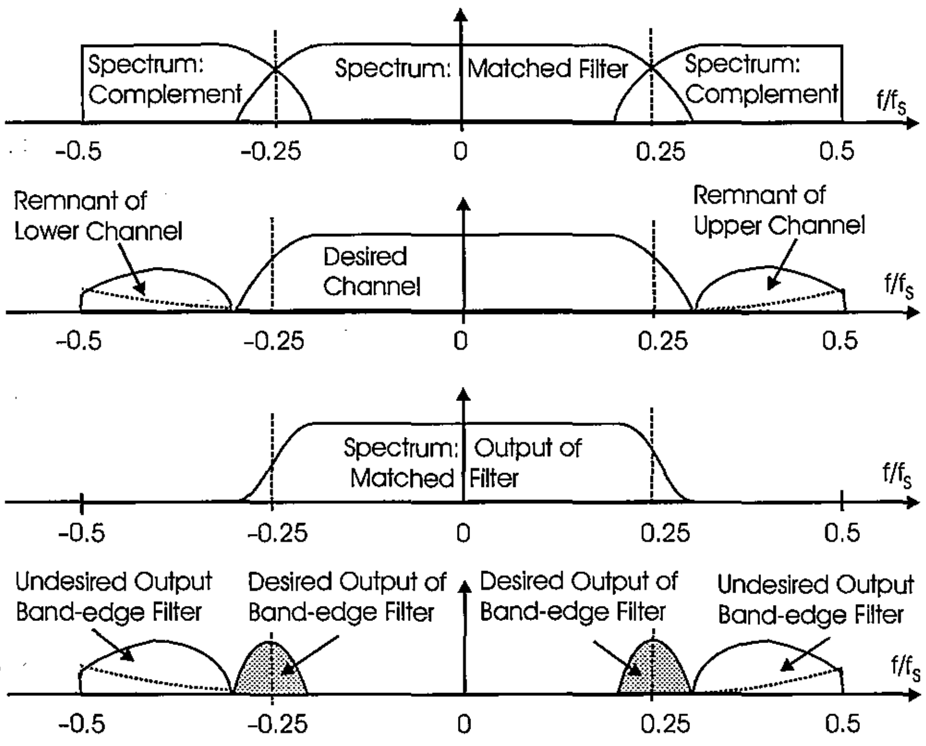


Figure 13.27 Spectrum of Matched Filter and Complement, Spectrum of Composite Signal Presented to Filter Pair, and Spectral Responses from Matched Filter and from Complementary Matched Filter

The polyphase filter following the frequency-matched filter in Figure 13.26 serves three tasks. Its first task is to reject the undesired spectral components adjacent to the band edge spectral regions. This is accomplished by appropriate selection of bandwidth and transition bandwidth of the prototype low pass filter. The next task is that of separating the signal contributions from the positive and negative frequency segments of the spectrum. We can accomplish this by complex translation of the prototype low-pass filter to each of the two spectral locations with trivial heterodynes of the form $\{1, +j, -1\ -j\}$. The resultant filters are narrowband Hilbert-transform filters. The third task is an option, and that is to reduce the sample rate because the band-edge filter has a lower bandwidth than the signal bandwidth. Note that the PLL filter that responds to the output of this band-edge signal has a significantly lower bandwidth than the signal bandwidth and is likely operating at a reduced sample rate. The sample rate reduction and the spectral translation from the center frequencies at the quarter-sample rate can both be folded into a polyphase resampling filter. The polyphase filter can be implemented with four or more paths depending on the band-edge bandwidth. This bandwidth is the same as the excess bandwidth of the square-root-Nyquist matched filter and the input sample rate is twice the signal bandwidth. Thus with a normalized excess bandwidth α equal to 0.25, the band-edge filter is oversampled 8-to-1 and can be down sampled by the same ratio.

There may be applications for which the spectral clean up and channel separation is desired without the sample rate change. Such an example is one in which the band-edge filters are used to generate spectral lines at the symbol rate, which is used as an aid in timing acquisition. For these cases, the polyphase partition can still be applied to the filtering task without invoking the noble identity to resample at the filter input. Figure 13.28 presents the filter chain that implements the matched filter and the band-edge filter pair from the matched filter complement, a polyphase filter, and a pruned 4-point FFT.

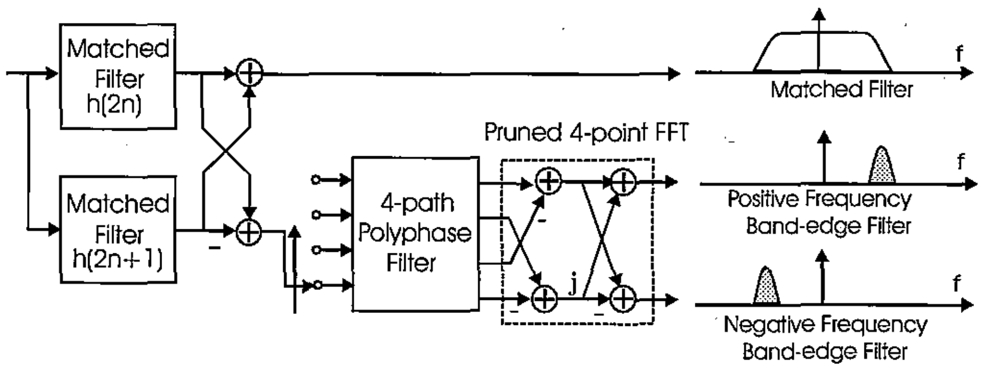


Figure 13.28 Signal Flow for Matched Filter and Upper and Lower Band-edge Filters

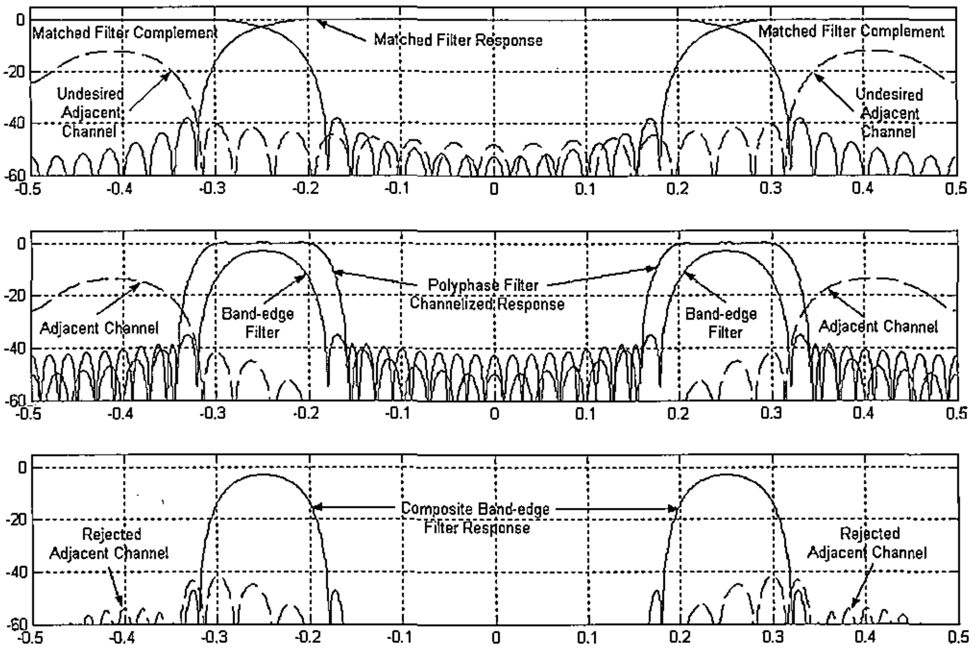


Figure 13.29 Desired and Undesired Spectra at Input to Matched Filter and Complementary Matched Filter, Response of Complementary Matched Filter to Both Inputs, and Response of Polyphase Filter to Both Inputs

Figure 13.29 shows the desired and undesired adjacent channel spectra at input to matched filter and complementary matched filter that forms the initial band-edge filter. We then see the response of the complementary filter to the two input components showing the desired band-edge response and the undesired adjacent channel contribution. Finally we see the spectral response of the 40-tap polyphase filter to the separate components present in the complementary matched filter. As designed, the band-edge filters' responses are isolated and are available and are separated by the polyphase filter.

13.5 Digitally Controlled Sampled Data Delay

A digitally controlled sampled data delay line described here is implemented with recursive all-pass filter sections. This is in marked contrast to the standard implementation of programmable time delays that use fixed nonrecursive polyphase stages or adjustable Farrow FIR filters. The recursive filter exhibits an equal-ripple approximation to constant group delay. The phase slope is programmable to present a continuously variable time-delay network. Applications of a continuously adjustable, linear-phase time-delay structure offers unique signal processing options to address various communication system tasks. These include timing recovery in DSP-based receivers, adaptive beam forming and steering, communication systems channel modeling, and reverberation modeling in acoustic chambers and instruments.

13.5.1 Recursive All-pass Filter Delay Lines

A polyphase M-path filter can be formed with recursive as well as with nonrecursive filter segments. The recursive system is composed of all-pass filters with numerator and denominator formed with reciprocal polynomials in Z^M. For ease of design, and without loss of generality, the polynomials are formed as a cascade of first- and second-order all-pass filter stages. The first-order filter in Z^M forms M-poles and M-zeros with a single multiply while the second-order filter in Z^M forms 2M-poles and 2M-zeros with two multiplies. These filters offer particularly efficient implementations of all-pass transfer functions. Figure 13.30 presents the standard structure for an M-path filter, which because of the Z^M polynomial structure can be used as an M-to-1 down sampler or as a 1-to-M up sampler. When the zero-indexed path in the M-path filter is selected to be pure delay, a particularly simple all-pass filter, every path in the M-path filter becomes an equal-ripple approximation to that path delay. Figure 13.31 presents the structure of a sixth-order recursive all-pass filter that makes up each path in the M-path polyphase filter. Each path contains two first-order polynomials and two second-order polynomials. Each path requires six multiplies per output data point.

Figure 13.32 presents the phase shift of the 10 paths in the 10-path recursive polyphase partition. Note the linear phase as a function of normalized frequency. Figure 13.33 presents the phase slopes for the same 10 paths of the 10-path filter formed with six coefficients per path. Note that the 10 paths present almost linear time delay over ± 35% of sample rate. If a signal is sampled at twice its nominal bandwidth, a common practice in many digital receivers, it will experience linear time delay over its whole bandwidth when processed by the 10 stages shown here. For comparison, a 10-stage polyphase FIR filter with the same linear phase shift over the same fractional bandwidth would require 12 taps per path. Figures 13.34 and 13.35 present the phase and phase slopes of the equivalent 10-path FIR filter.

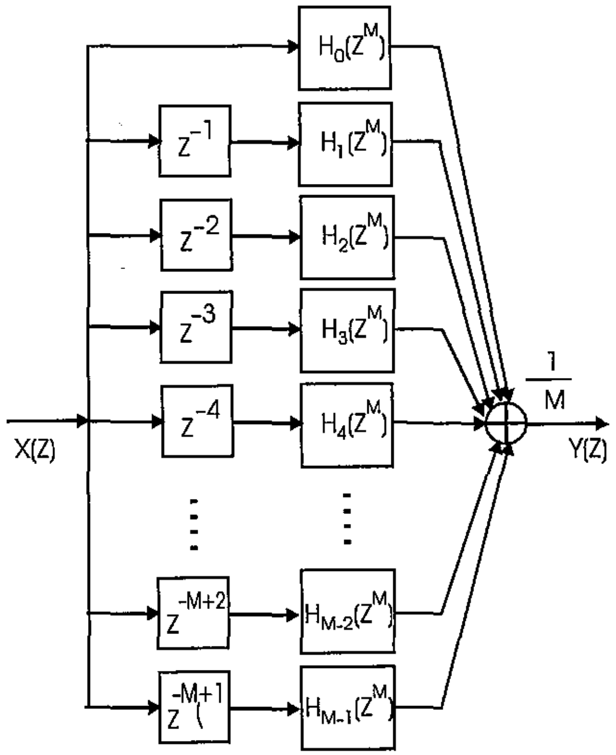


Figure 13.30 M-path Filter Implemented as Recursive All-pass Filter in Z^M

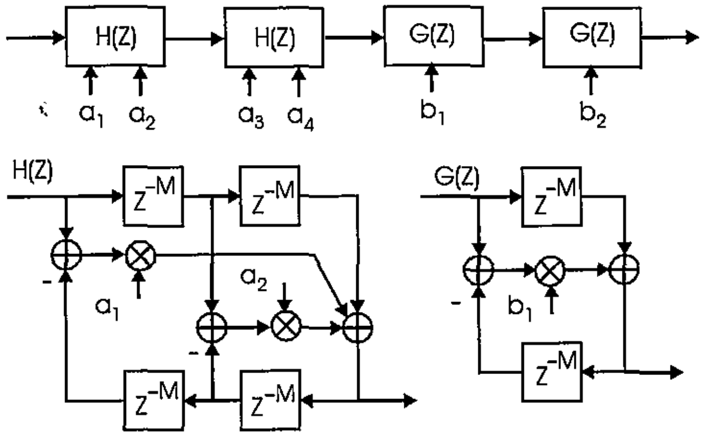


Figure 13.31 Structure of Each Path in M-Path Recursive All-pass Polyphase Filter

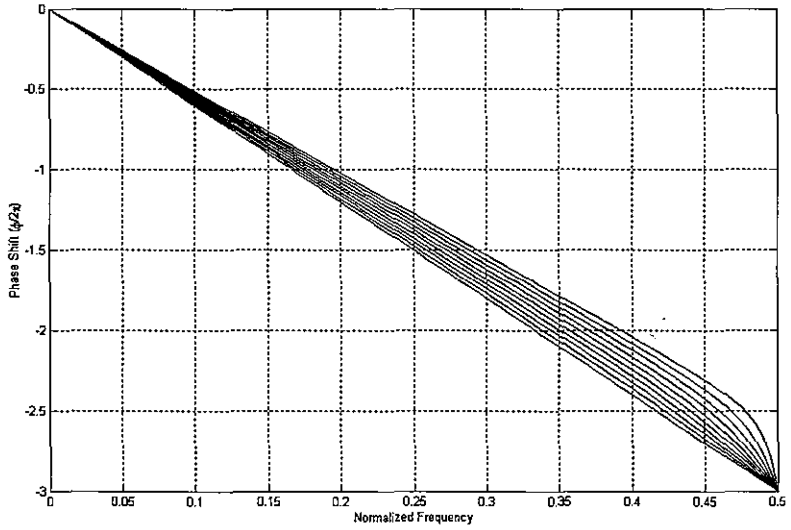


Figure 13.32 Phase Shift for 10 Paths of 10-path Recursive Filter

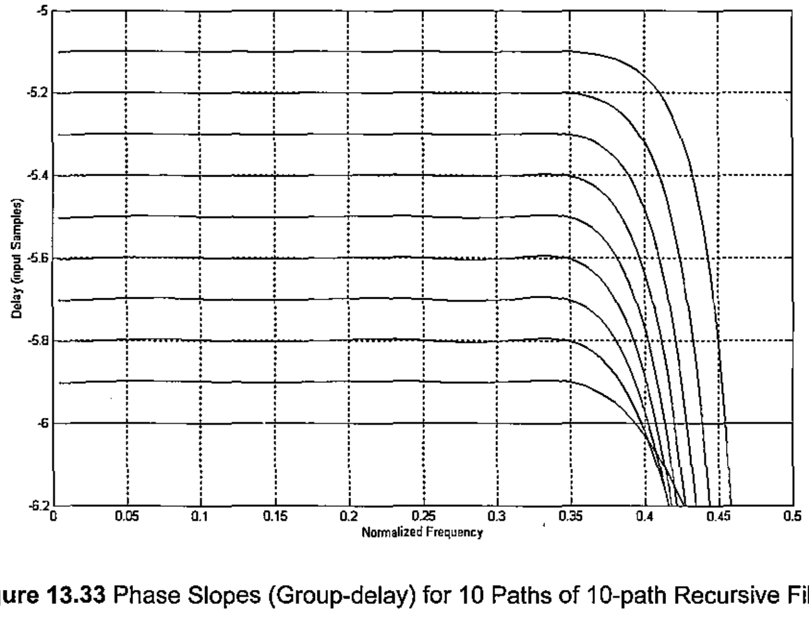


Figure 13.33 Phase Slopes (Group-delay) for 10 Paths of 10-path Recursive Filter

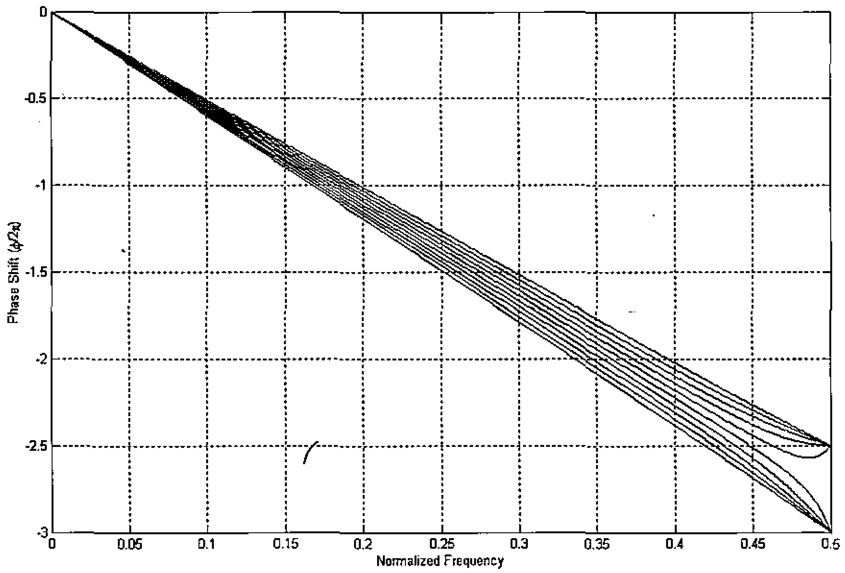


Figure 13.34 Phase Shift for 10 Paths of 10-path Nonrecursive Filter

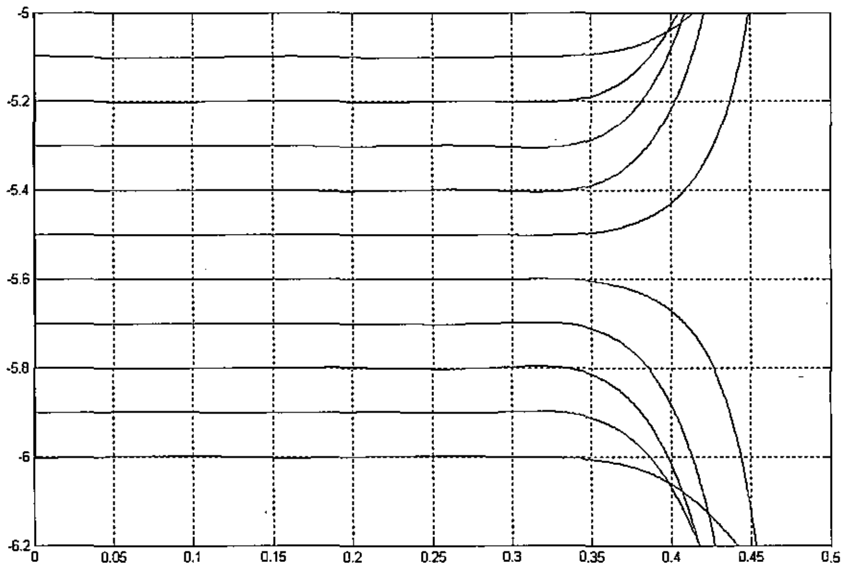


Figure 13.35 Phase Slopes (Group-delay) for 10 Paths of 10-path Nonrecursive Filter

We can imagine that the six coefficients of the all-pass filter on each of the nine non-trivial paths are samples of a smooth continuous curve which, when sampled at increments of 0.1, present the values that define the 9 paths. If we had the smooth continuous functions, we could sample it at any location (besides the multiples of 0.1) to determine the weights required to obtain arbitrary time delay corresponding to the desired sample point. Figure 13.36 presents the coefficients of the 9-path filters and a fifth-order polynomial fitted with equiripple error to the sample values. As seen, the polynomials offer the smooth continuous function that relates weight values to delay. These polynomials can then be sampled at any location to obtain the weights that will present the delay at that sample position. The separate curves are labeled with their parameter values as shown in Figure 13.31.

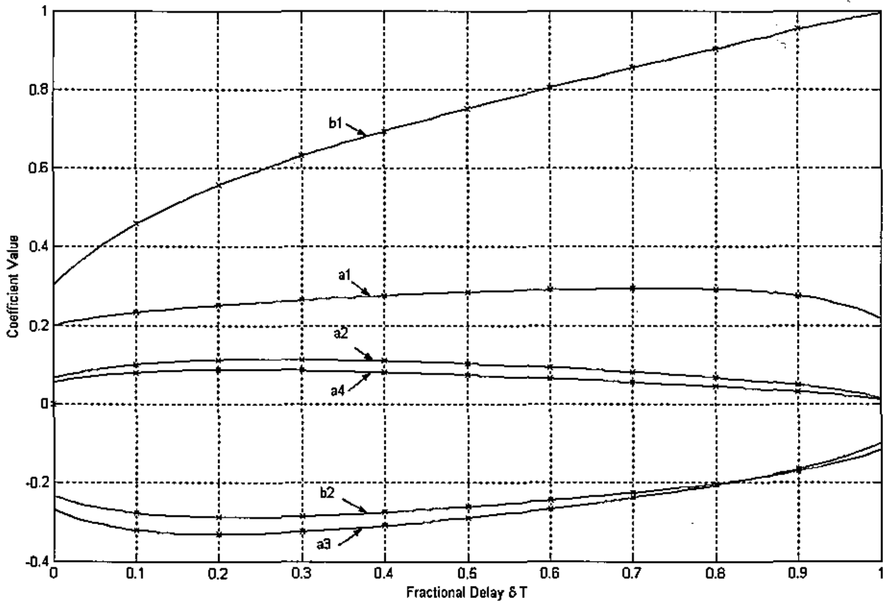


Figure 13.36 Coefficient Values for Non-trivial Paths of the10-path Filter and Fifth-order Polynomial Fitted to Values

These polynomials are embedded in the structure of an arbitrary time delay network as shown in Figure 13.37. Figure 13.38 presents the original delays plus those obtained by evaluating the polynomials at increments of 0.1 starting from an initial offset of 0.05. These samples correspond to delays midway between those obtained from the original 10-path filter bank. As can be seen, the delays from the coefficients obtained by sampling the polynomials are also linear with frequency.

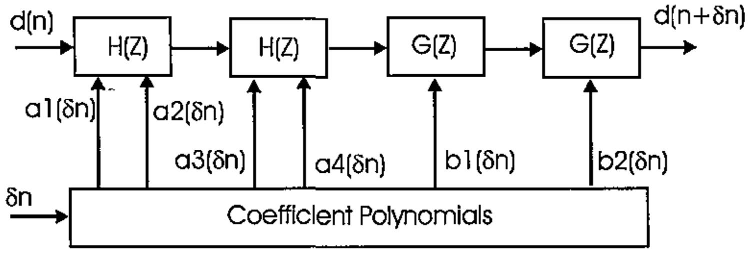


Figure 13.37 Programmable Time-delay Network with Associated Coefficient Generator

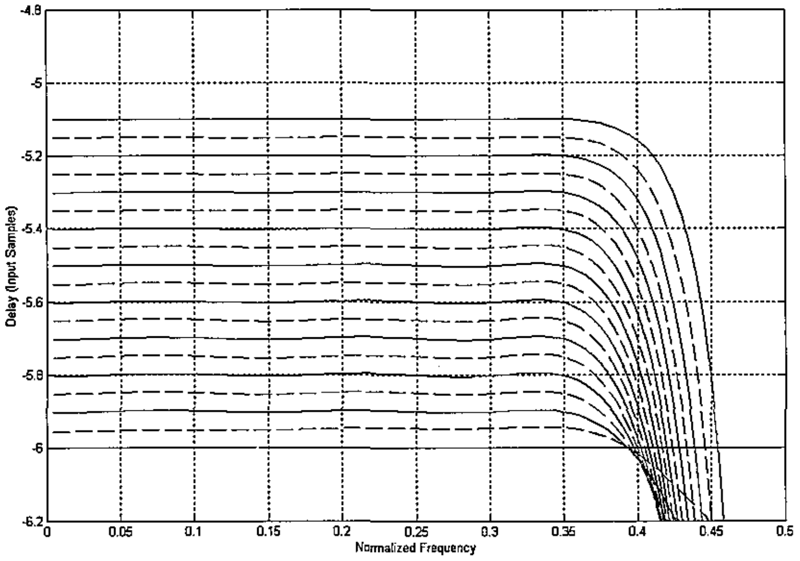


Figure 13.38 Original Delays Plus Midvalue Delays Obtained by Evaluating Coefficient Polynomials Defined by Original 10-path Filter

The digital delay line described here has been used in a timing recovery loop in the receiver structure shown in Figure 13.39. Here the input data is collected at 2-samples per symbol, the envelope is time shifted by the variable digital time delay line, and it is presented to the decision feedback equalizer and detector. Figure 13.40 illustrates various time responses of the receiver structure of Figure 13.39 that uses the variable time-delay network in its timing recovery loop. The variable, labeled timing coefficient, shown in the upper-right corner of Figure 13.40, is the parameter delivered to the coefficient generator shown in Figure 13.37. The timing error is reduced to the minimum level resolvable by the timing error detector. The instantaneous and average time-delay error is shown in the lower-right corner of Figure 13.40.

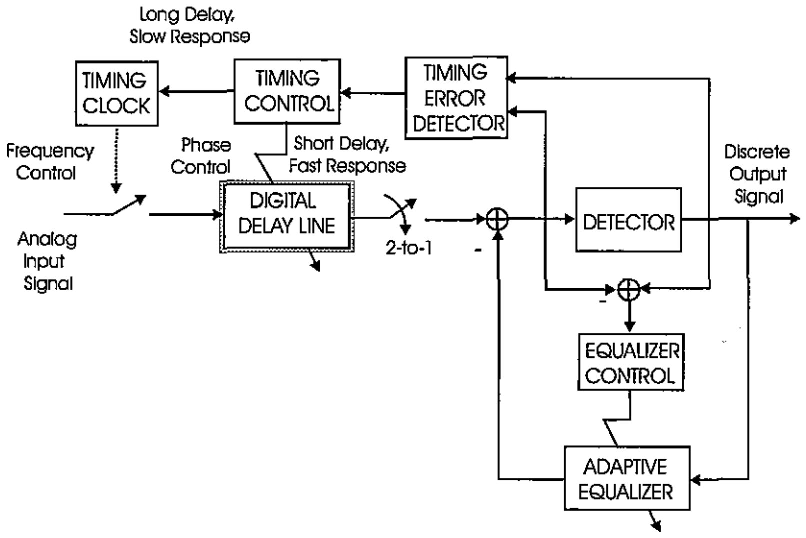


Figure 13.39 Digital Delay Line Embedded in Digital Receiver

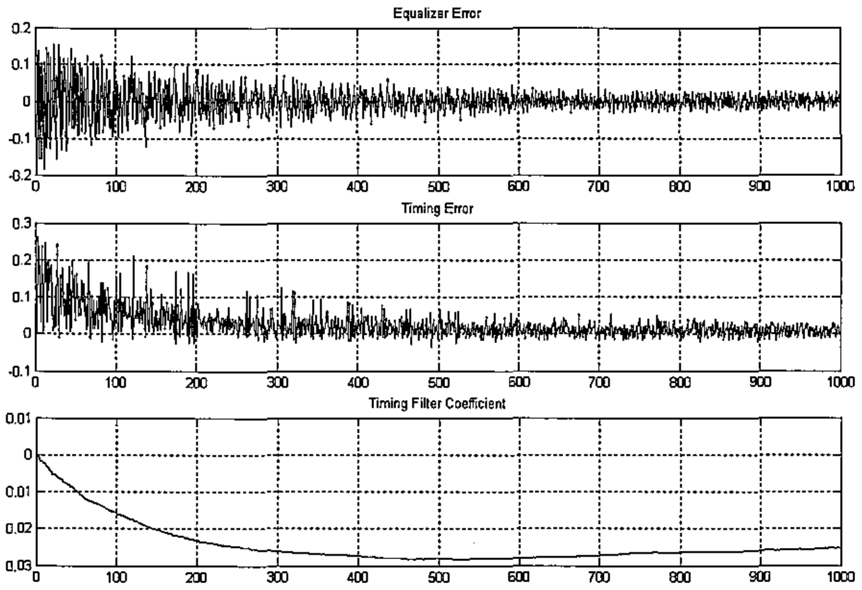


Figure 13.40 Equalizer and Timing Error Response to Timing Coefficient Trajectory of Digital Delay Line Embedded in Digital Receiver

We have demonstrated here that a single path of a standard M-path recursive polyphase filter can be used as a programmable linear-phase time-delay network in a manner similar to the FIR Farrow filter. The example we used to illustrate the performance of this concept used two second-order polynomials in Z^2 and two first-order polynomials in Z^2, for a total of 6-multiplies per filter path, to obtain the variable time-delay process. Time-delay filters can be formed with fewer stages per arm, and we have tested the performance of systems with 1, 2, or 3 multiplies. These filters, with smaller number of computations, perform very well. They simply exhibit a larger ripple in the time-delay versus frequency characteristic, or exhibit linear group delay over a smaller fraction of the input sample rate due to larger transition bandwidth. Interestingly, when coupled with a decision feedback equalizer, the performance of the time-delay network with a small number of coefficients is enhanced since the equalizer attributes residual group-delay distortion to the channel and compensates appropriately. For those interested in playing with the structure we describe here, Table 13-3 lists the six coefficients of the nine nontrivial paths of a linear-phase 10-path recursive polyphase filter. The MATLAB script file *time_10* that contains the same coefficients and generated the phase portraits is also available in the disc accompanying this text.

Table 13-3 Coefficients of All-pass Filters of Paths 1–9 of 10-path Polyphase Filter

Path #	B1	B2	A1	A2	A3	A4
1	0.458758	– 0.276820	0.233603	0.100849	– 0.320550	0.080736
2	0.556790	– 0.287255	0.252152	0.112364	– 0.330782	0.087489
3	0.630817	– 0.284168	0.265170	0.113768	– 0.324013	0.086214
4	0.693992	– 0.274731	0.275704	0.110127	– 0.309201	0.081185
5	0.751025	– 0.261264	0.284359	0.103237	– 0.289275	0.073988
6	0.804237	– 0.244646	0.290925	0.093857	– 0.265387	0.065386
7	0.854981	– 0.224961	0.294424	0.082187	– 0.237745	0.055720
8	0.904135	– 0.201338	0.292323	0.067839	– 0.205482	0.044960
9	0.952320	– 0.170104	0.276523	0.048997	– 0.164700	0.032264

13.6 Interpolated Shaping Filter

Throughout this book we have examined a number of techniques to implement resampling filters. In this section we select a specific task of up sampling by a factor of 8 the impulse response of shaping filter used in a modulator. The particular shaping filter we examine is a square-root cosine-tapered Nyquist filter used in a third-generation cellular phone system. A friend supplied the spectral mask the system had to satisfy and asked us to design the filter and the interpolator to perform the up-sampling task. The question then arose as to what would be an efficient up sampler and what signal degradation effects would be introduced by the up-sampling options. The distortion terms are related to the pre and post echoes that the interpolating filter contributes to the time response due to ripple in spectral amplitude and spectral phase. We also chose to examine distortion levels related to nonuniform phase characteristics of efficient recursive filters. The measure of distortion that is easy to access is the peak and rms level of ISI at the output of the matched filter. The types of interpolators we examined are listed in Table 13-4, along with an indication of their relative computational complexity and their ISI levels. Also included in this table is the level of ISI contributed by the original shaping filter operating at its design frequency of 2-samples per symbol and then down sampling to symbol rate. The remaining table entries were obtained by operating the filters at 16-samples per symbol and then down sampling the filter output to the symbol rate.

Table 13-4 Interpolator Options and their Comparative Measures

Interpolator Type	Length or Number of Coefficients	Operations per Output Sample	Peak and RMS ISI
Shaping Filter No Interpolation	41-taps	41-M, 41-A	PK; 0.0101 RMS; 0.0036
8-path Polyphase FIR	72-taps, 9-taps/Path	9 M, 9A	PK; 0.0158 RMS; 0.0040
3-stage Half band FIR Filter Set	1st Stage; 21-taps, 2nd Stage; 13-taps 3rd Stage; 9-taps	4.8 M, 4.8 A	PK; 0.0109 RMS; 0.0036
8-path Linear-phase Polyphase IIR Filter	21-coef. 3-coef./Path	2.7 M, 5.3 A	PK; 0.0128 RMS; 0.0040
3-stage Linear-phase IIR Half-band Filter Set	1st Stage; 4-coef. 2nd Stage; 2-coef. 3rd Stage; 1-coef.	1.5 M, 3.0 A	PK; 0.011 RMS; 0.0036
8-path Nonlinear Phase Polyphase IIR Filter	14-coef. 2-coef./Path	1.8 M, 3.6 A	PK; 0.0817 RMS; 0.0259
3-stage Nonlinear Phase IIR Half-band Filter Set	1st Stage; 3-coef. 2nd Stage; 2-coef. 3rd Stage; 1-coef.	1.4 M, 2.8 A	PK; 0.0808 RMS; 0.0284

Figure 13.41 shows the impulse response and the frequency response of the shaping filter designed to meet the spectral mask indicated on the spectral plot. Also presented in the spectral plot is an insert figure zoomed to show the in-band ripple levels of the shaping filter. The in-band ripple level was specified to be less than 0.1-dB, and here it is seen to be 0.022-dB. We see here that the frequency of the ripple is 10 cycles per interval of symbol bandwidth that has been normalized to 1.0 in this figure. The ripple frequency tells us that the matched filter output will exhibit pre and post echoes at positions ±10 symbols from the location of its peak value.

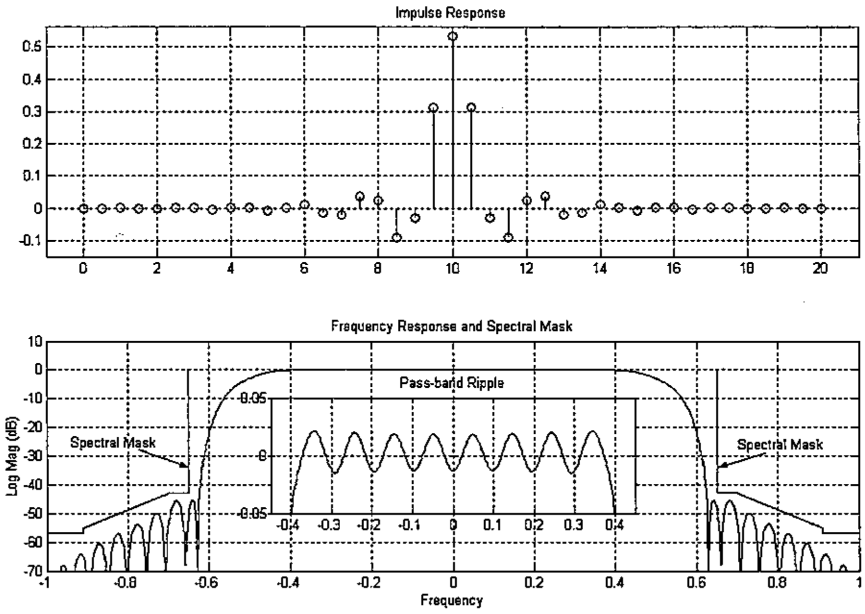


Figure 13.41 Impulse Response and Spectra of Shaping Filter

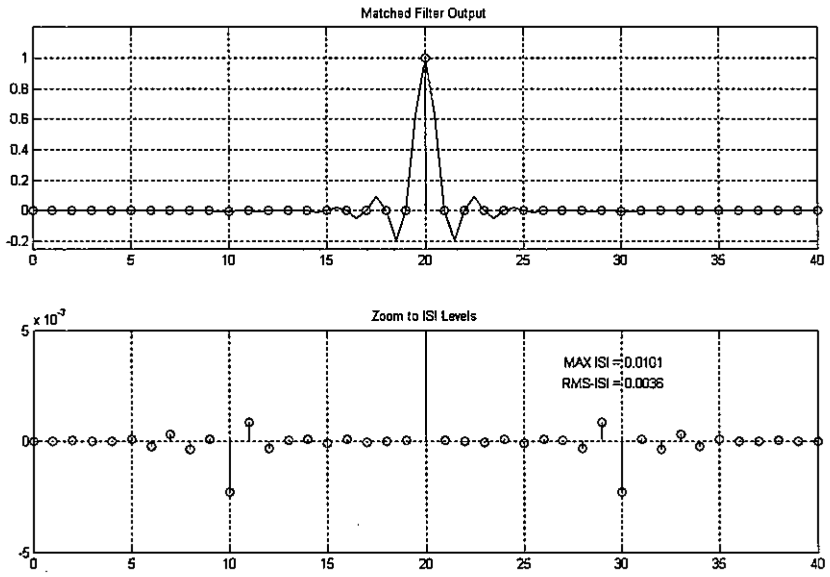


Figure 13.42 Matched Filter Response and Detail Showing ISI levels

Figure 13.42 presents output samples of the matched filter taken at two samples per symbol with circles marking the one-sample-per-symbol time marks aligned with the peak output. The subplot accompanying the matched filter is a zoom to the low ISI levels where we see, as expected, the pre and post echoes at ± 10 symbols from the peak position. The maximum ISI level that the matched filter can exhibit, computed as the sum of the absolute values, was found to be 0.0101, or approximately 1% of the peak output level. The RMS value of the ISI cloud around any constellation point was found to be 0.0036, approximately 1/3 of the peak ISI.

The first contender that comes to mind for use as the 1-to-8 up sampler is an 8-path polyphase FIR filter. A prototype FIR filter of length 72 taps satisfied the filtering requirements dictated by the transition bandwidth and out-of-band attenuation levels. This filter is partitioned into an 8-path polyphase structure that is fed by the shaping filter. The polyphase filter is likely implemented in the form shown in Figure 7.13 rather than in the expanded form suggested by Figure 13.43. We see here that the polyphase form of the filter requires 9 multiplies and adds per interpolated output sample point. We use this workload as the reference against which we compare the other interpolator options.

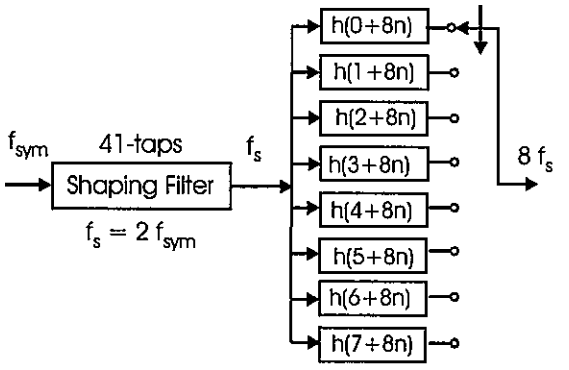


Figure 13.43 Block Diagram of Signal Flow for Shaping Filter and 1-to-8 Polyphase FIR Filter Interpolator

Figure 13.44 shows the phase response of the 8-paths of the polyphase partition of this prototype filter along with its log magnitude frequency response. The phase response imparts little value here except that it does show the constant phase offsets present in the different Nyquist zones that can be used in a polyphase channelizers. The reason we show the phase here is that we will shortly compare this phase profile with corresponding profiles of recursive counterparts of the 8-path filter. Figure 13.45 shows the spectrum of the interpolating filter overlaid on the periodic extension of the shaping filter's spectrum and the spectrum of the composite impulse response of the shaping filter and the 1-to-8 interpolating filter. Also shown in Figure 13.43 is a zoom to the interpolator's in-band ripple. This ripple

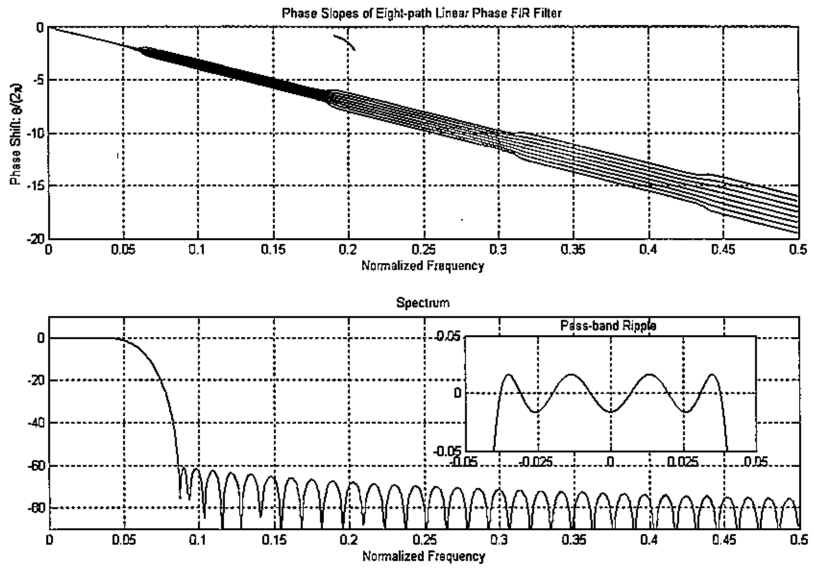


Figure 13.44 Path Phase Responses of 8-Paths Polyphase Filter and Frequency Response of Prototype Filter

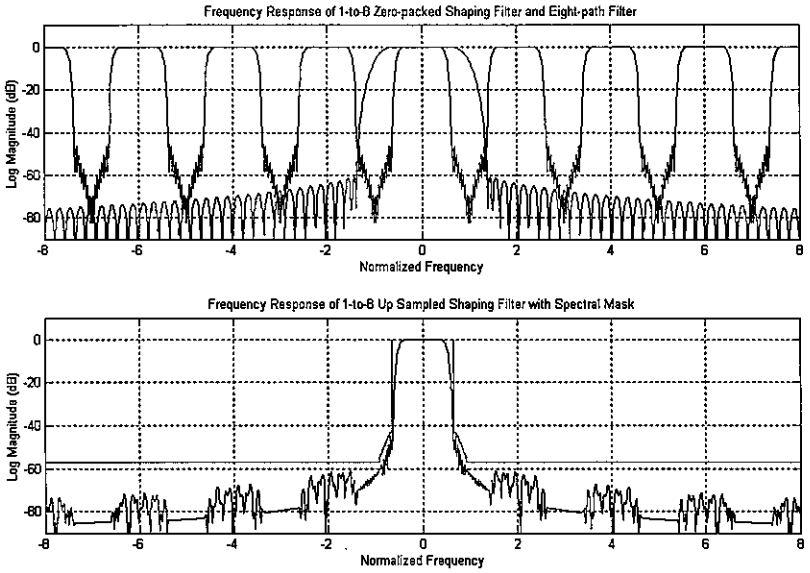


Figure 13.45 Spectra of Up-sampled Shaping Filter with Spectra of Interpolating Filter and Spectra of Interpolated Shaping Filter Response

is seen to be approximately 4-cycles per signal bandwidth and is approximately the same amplitude of the shaping filter ripple. We would expect an increase in ISI level due to the composite filtering action of the two filters. In fact, the ISI exhibited by the time series formed by the 8-path interpolator increased from peak levels of 1% to 1.6% and from RMS levels of 0.36% to 0.4 %.

The next contender for use as the 1-to-8 up sampler is a cascade of 3-stages of half-band FIR filters . The half-band filters offer two primary advantages over a single polyphase filter. The first is that by ratcheting the sample rate up in increments of 2 the successive filters in the cascade, while operating at higher sample rates, do so with shorter filters due to the increased separation between the spectral replicates at the higher sample rates. The second advantage is the nearly 50% reduction in workload due to the zero-valued weights in the half-band filter. A set of prototype FIR filters of lengths 21, 13, and 9 taps respectively satisfied the filtering requirements dictated by the successive transition bandwidths. These filters are partitioned into 2-path polyphase structure in which the upper path of each defaults to a simple delay line. The filter cascade is shown in Figure 13.46, where we see the delays in the upper arm and the nontrivial weights in the lower arm. The lower arm is tagged with its workload per output sample. Note that the first stage is used once per input to output 2 samples with a stage workload of 10 operations. These samples in turn are presented to the next stage, which must operate once per input it sees and thus operates twice to output 4 samples for a stage workload of 12 operations. Finally, the last stage sees 4-inputs and thus operates 4-times to output 8-data samples with a stage workload of 16 operations. The total workload for the three stages is [10 + 12 + 16] or 38 operations, which when amortized over the 8 output samples leads to 4.75 operations per output. This workload is approximately half that of the 8-stage polyphase filter. We note that the half-band filters are too short to exhibit ripple in the signal bandwidth they are interpolating. Thus a second benefit of the half-band filter is that it contributes an insignificant increase to the composite ISI, with the peak ISI changing from 1.0% to 1.1% and the RMS ISI not changing within the measurement resolution of 0.01%. Figures 13.47, 13.48, and 13.49 present the spectra of the successive 1-to-2 up-sampled input data with the overlaid half-band filter responses as well as the spectra of the up-sampled and filtered time responses. These spectra are presented to illustrate the ratcheting process that enables the higher speed processing tasks to be implemented with simpler, shorter filters.

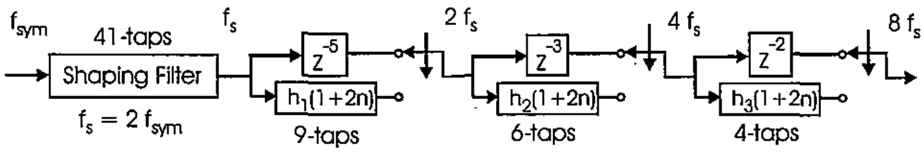


Figure 13.46 Block Diagram of Signal Flow for Shaping Filter and Cascade of Three Levels of Half-band 1-to-2 Up-sampling FIR Interpolating Filter

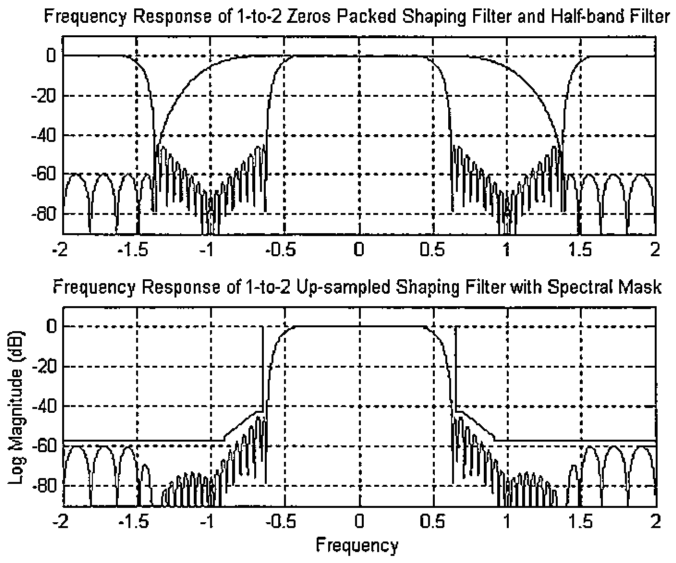


Figure 13.47 Spectra of Up-sampled Shaping Filter with Spectra of First Half-band FIR Interpolating Filter and Spectra of 1-to-2 Interpolated Filter Response

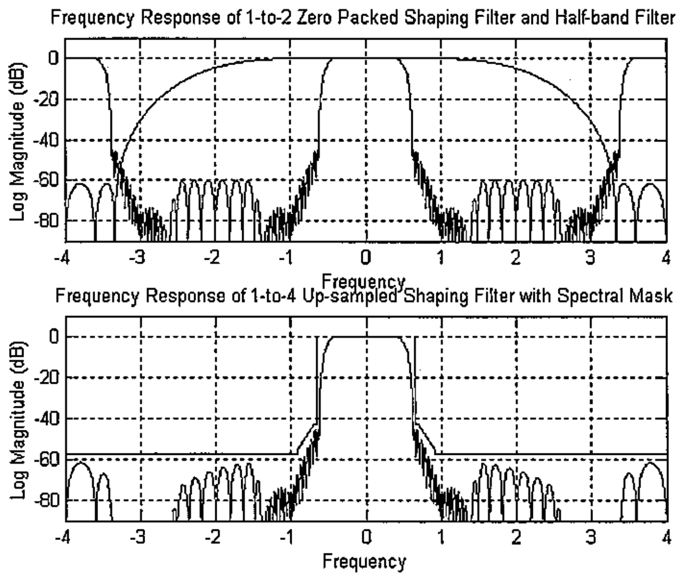


Figure 13.48 Spectra of 1-to-4 Up-sampled Shaping Filter with Spectra of Second Half-band FIR Interpolating Filter and Spectra of 1-to-4 Interpolated Filter Response

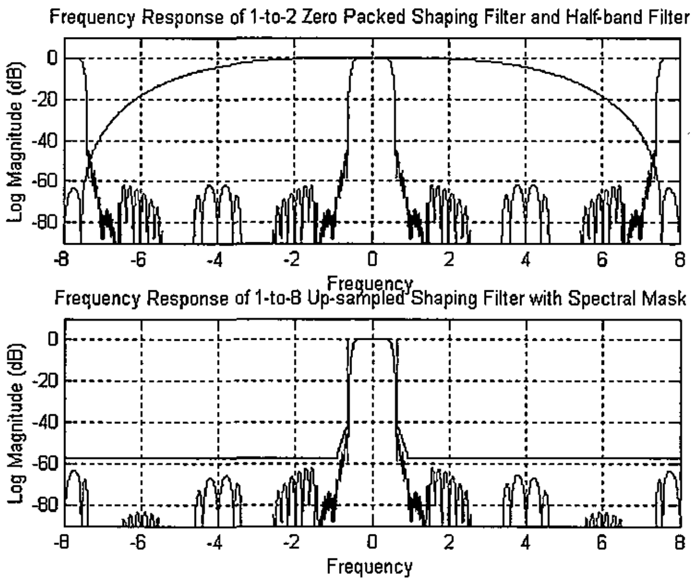


Figure 13.49 Spectra of 1-to-8 Up-sampled Shaping Filter with Spectra of Third Half-band FIR Interpolating Filter and Spectra of 1-to-8 Interpolated Filter Response

At bandwidths, hence sample rates, for which we can use recursive filters for the interpolating function we can bring new contenders to the table. These contenders are the resampling IIR filters described in detail in Chapter 10. There are recursive counterparts to the two nonrecursive options we have just examined. These are an 8-path polyphase filter and a three-stage cascade of half-band filters. Figure 13.50 presents the approximately linear-phase IIR polyphase filter structure. This filter approximates the linear-phase characteristics by setting its top arm to a delay line with linear phase to which the remaining arms approximate with equal-ripple error. The nontrivial arms each have two filters in cascade, one filter requiring a single multiply and two adds and the second filter requiring two multiplies and four adds. The structural detail of these filters is found in Figures 10.11 and 10.35. The workload for this polyphase filter is seen to be less than three operations per output sample. Figure 13.51 shows the group-delay response and the in-band ripple response of the prototype 8-path IIR filter. We first note the remarkable low level of amplitude ripple, the peak ripple being approximately 2 μ-dB presenting an inconsequential level of ISI. The group-delay ripple, on the other hand, has amplitude of 0.05 and a frequency of approximately 2 cycles per signal bandwidth or 32 cycles per sample rate. We would expect this filter to contribute odd symmetric echoes of amplitude 0.05/32 at a 2-sample offset from the maximum filter response. Figure 13.52 presents the matched-filter response and a zoom to the ISI levels to show the ISI contributors. The echoes are at the location and at the levels we expected.

Figure 13.53 shows the phase response of the 8-paths of the approximately linear-phase IIR polyphase filter along with its log magnitude frequency response. The phase response shows the phase of the delay-only reference path, and the phase response of the other paths as they approximate the reference slope. Note that at the Nyquist boundaries, the phase difference expands to the next multiple of $2\pi/8$ and that this expansion occurs by inserting additional phase shift. Compare this behavior to the related phase expansion of the 8-path FIR filter in Figure 13.44. For the IIR, the spectral intervals corresponding to the phase transitions between Nyquist zones, the phase terms do not destructively cancel, and the magnitude response exhibits a transition bandwidth between stop band intervals in successive Nyquist zones. We can clearly see the transition bandwidths in the associated log magnitude spectrum. Our first reaction to these spectral bumps is horror; possibly "Good grief!" These transition intervals do not bother us, they can be thought of as don't care bands matching the spectral intervals in which we know there is no input energy by virtue of the input signal being oversampled. Figure 13.54 shows the spectrum of the IIR interpolating filter overlaid on the periodic extension of the shaping filter's spectrum. As expected the transition regions of the polyphase IIR filter match the stop bands already present in the input spectrum. The spectrum of the up sampled and filtered composite impulse response is seen to meet the spectral ask requirements of the filtering task.

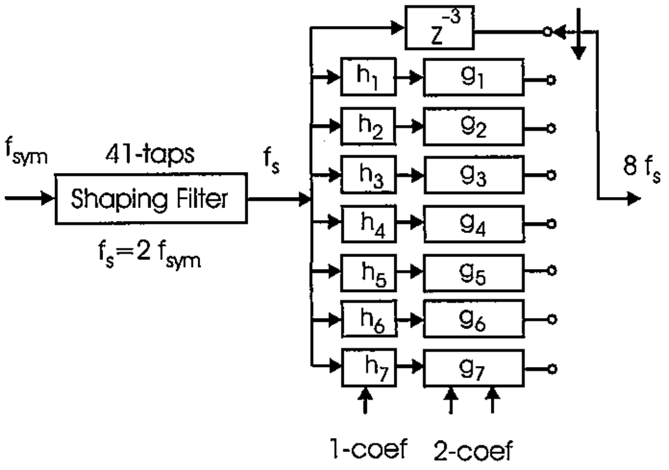


Figure 13.50 Block Diagram of Signal Flow for Shaping Filter and 1-to-8 Polyphase, Approximately Linear, Recursive Filter Interpolator

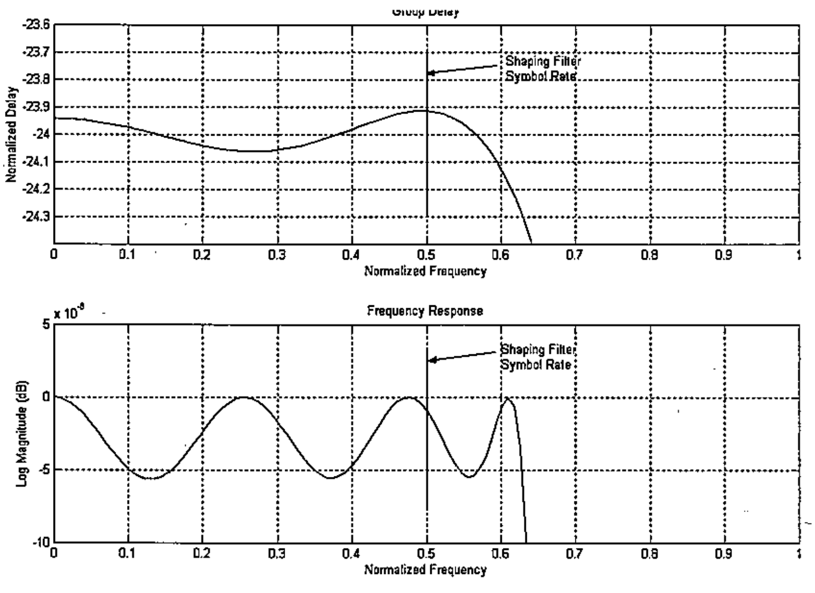


Figure 13.51 Zoom to Group-Delay Ripple and Magnitude Ripple of 1-to-8 Polyphase Recursive, Approximately Linear, Recursive Filter Interpolator

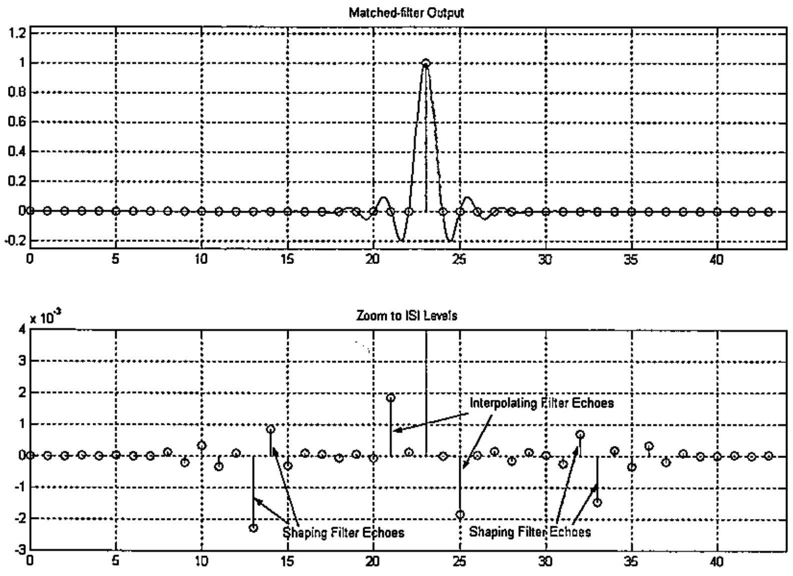


Figure 13.52 Interpolated Matched-filter Response and Detail Showing ISI Levels

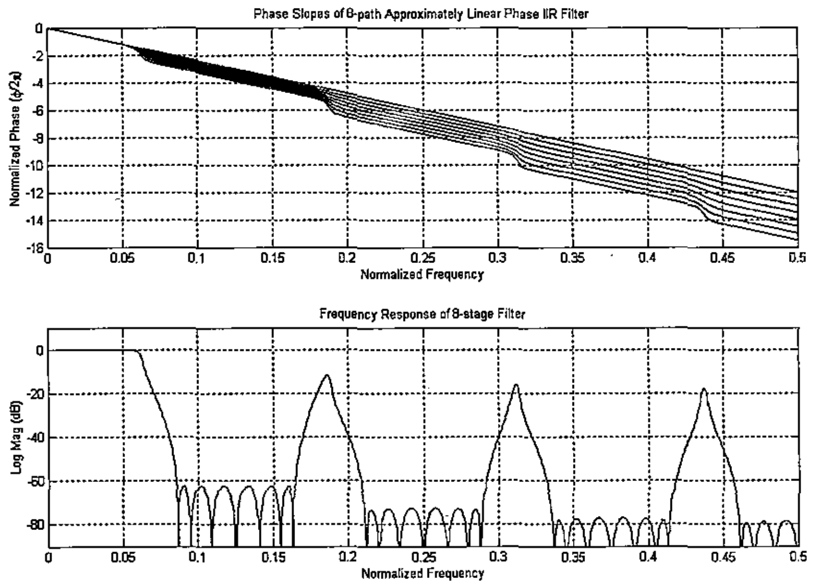


Figure 13.53 Path-phase Responses of 8-path Polyphase IIR Filter and Frequency Response of Prototype Filter

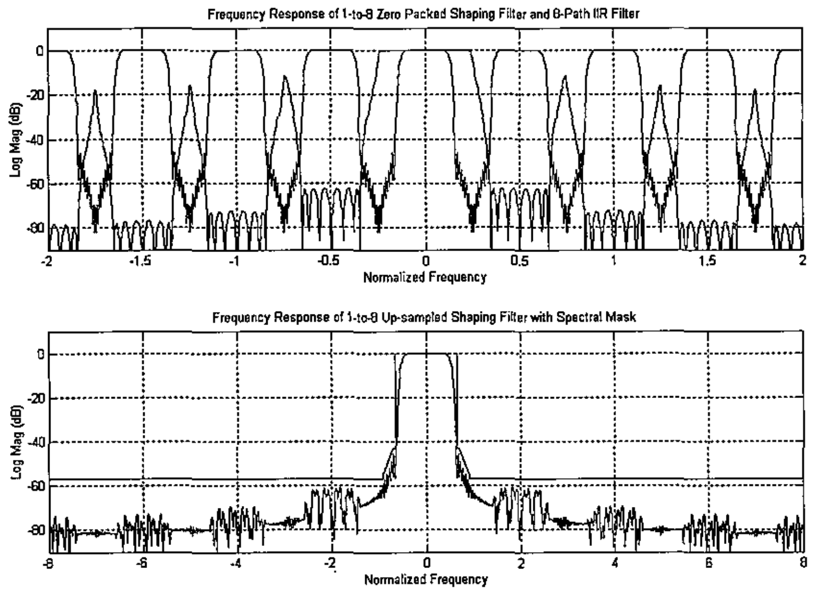


Figure 13.54 Spectra of Up-sampled Shaping Filter with Spectra of IIR Interpolating Filter and Spectra of Interpolated Shaping Filter Response

The final interpolator option we examine in this discussion is the use of the nearly linear-phase recursive half-band filters. These filters have the same property as the FIR half-band filters in that the upper arm of the polyphase half-band is delay only. The IIR version of a 3-stage cascade that meets the filtering requirements of the interpolator task is shown in Figure 13.55. Here we see the delays in the upper arm and the all-pass filter structures in the lower arm. The workload for the filters in the lower arm matches the number of input coefficients that represents a small workload to perform its phase-shift task. In this cascade, the first stage is used once per input to output 2 samples with a stage workload of 4 operations. These samples in turn are presented to the next stage, which must operate once per input it sees and thus operates twice to output 4 samples for a stage workload of 4 operations. Finally, the last stage sees 4-inputs and thus operates 4-times to output 8-data samples with a stage workload of 4 operations. The total workload for the three stages is [4 + 4 + 4] or 12 operations, which when amortized over the 8 output samples leads to 1.5 operations per output. This workload is approximately half that of the 8-stage polyphase IIR filter and one-third of the workload for the FIR half-band cascade. We note that the IIR half-band filters do not exhibit amplitude ripple and that their phase ripple has a long period relative to the bandwidth being processed during the interpolation process. Thus a second benefit of the IIR half-band filter is that it contributes an insignificant increase to the composite ISI, with the peak ISI changing from 1.0% to 1.1% and the RMS ISI not changing within the measurement resolution of 0.01%. Figures 13.56, 13.57, and 13.58 present the spectra of the successive 1-to-2 up-sampled input data with the overlaid half-band IIR filter responses as well as the spectra of the up-sampled and filtered time responses. These spectra are presented to allow comparison between FIR and IIR filters performing the same ratcheting process to permit shorter filters at higher sample rate. Be sure to compare these figures with Figures 13.47, 13.48, and 13.49, keeping in mind the relative workloads expended while performing the interpolation process.

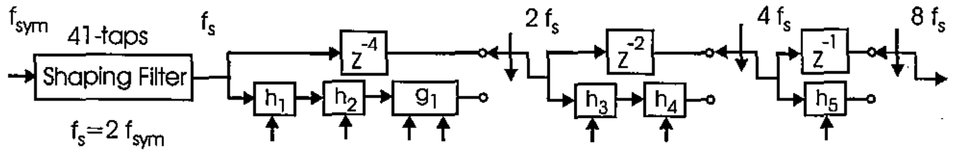


Figure 13.55 Block Diagram of Signal Flow for Shaping Filter and Cascade of Three Levels of Half-band 1-to-2 Up-sampling IIR Interpolating Filter

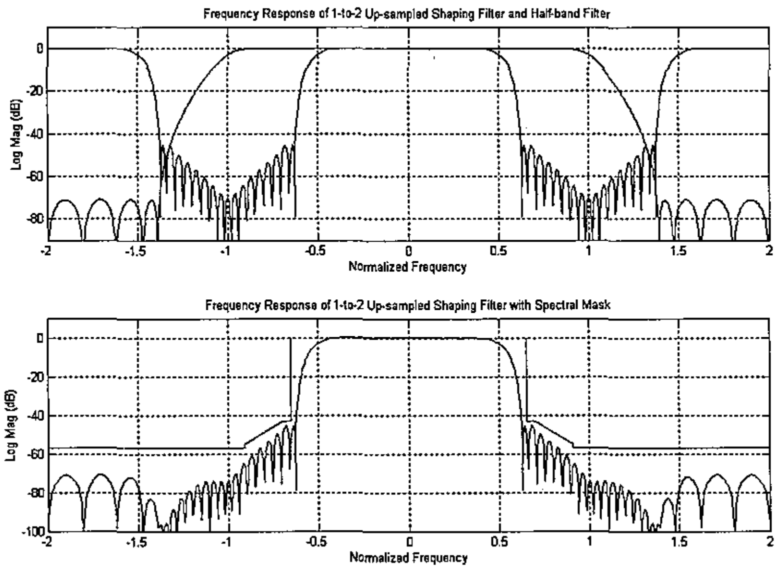


Figure 13.56 Spectra of Up-sampled Shaping Filter with Spectra of First Half-band IIR Interpolating Filter and Spectra of 1-to-2 Interpolated Filter Response

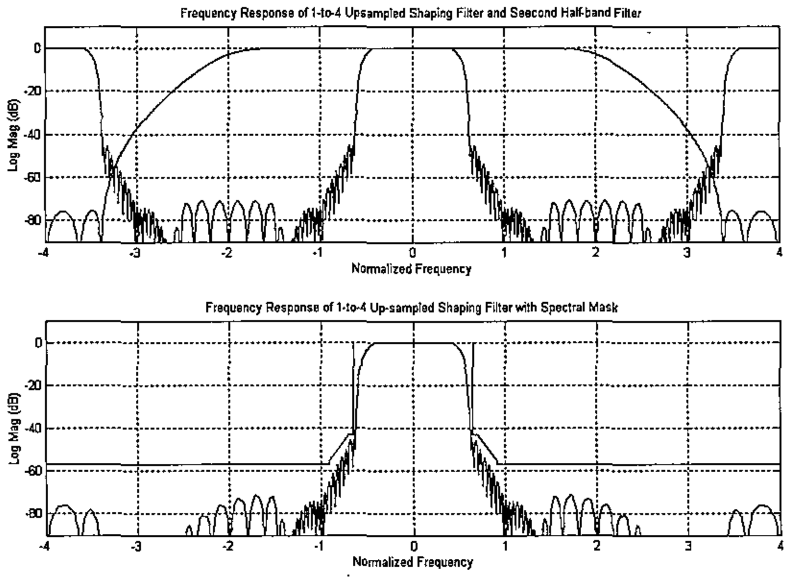


Figure 13.57 Spectra of 1-to-4 Up-sampled Shaping Filter with Spectra of Second Half-band IIR Interpolating Filter and Spectra of 1-to-4 Interpolated Filter Response

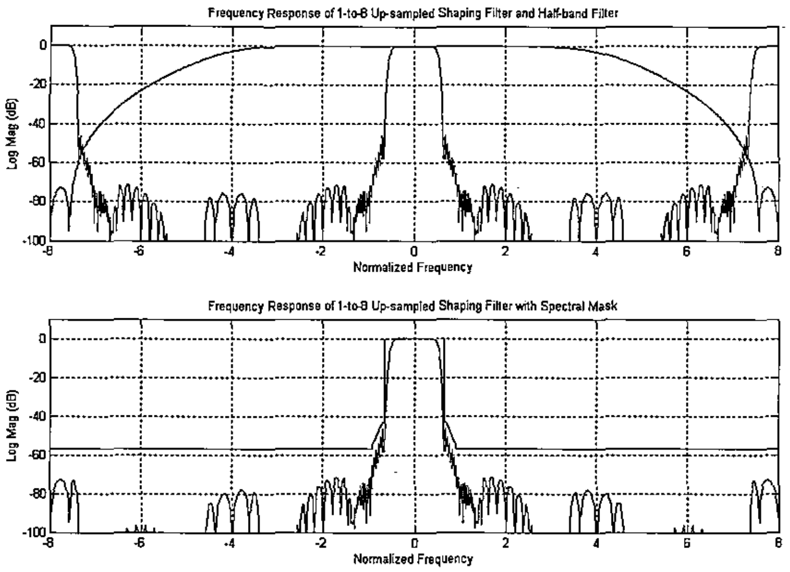


Figure 13.58 Spectra of 1-to-8 Up-sampled Shaping Filter with Spectra of Third Half-band IIR Interpolating Filter and Spectra of 1-to-8 Interpolated Filter Response

A final note in this area is that the 8-path polyphase filter and the 3-stage half-band filter can also be implemented with recursive filters that exhibit nonuniform phase responses. The workload required of these filters is again reduced relative to that of the linear-phase recursive filter set. These filters can be mixed and matched, linear filters at the low sample rate and nonlinear phase filters at the higher sample rate. At the higher rates, the fractional bandwidth is sufficiently small that the nonuniform phase IIR filters are approximately linear-phase filters. Another, tongue in cheek, comment is that group-delay distortion introduced in the shaping filter and interpolator at the modulator will be attributed to channel effects, including analog filters in the signal flow path, and that these small effects will be suppressed by an equalizer in the demodulator. A less onerous comment is that the group-delay distortion terms generated by the low-cost, nonuniform phase recursive interpolating option are known a priori and can be pre-equalized by inserting their conjugate phase in the initial shaping filter.

13.7 Sigma-delta Decimating Filter

Multirate filters find great application in source coding applications. One ubiquitous application is in the filtering and sample-rate change associated with sigma-delta modulators. The sigma-delta modulation process is a source-coding technique that uses memory to represent samples of data at a given level of fidelity with a smaller number of bits than normally required to achieve that fidelity level when used without memory. The source coding performed by an A-to-D converter is that of representing a sampled data process with a finite set of amplitudes in such a way that the error between the two representations is made acceptably small. Many A-to-D and D-to-A converters perform their conversion process on a sample-to-sample basis without regard to past or future conversion steps. These are often called memoryless converters or, as a friend once observed, converters with amnesia. The memory required for this conversion process resides in the correlation between data samples. If data samples are highly correlated, and if we have the current and recent past samples of the data, there is little uncertainty of the value of the next sample. Predictive encoders operate on this basis. We predict the next sample and use the converter to measure or resolve the error in the prediction process. Similarly, if the data is highly correlated, the error generated by the conversion process is also highly correlated and hence predictable. If we are able to predict the error made by the quantizer, we can subtract the estimate prior to the conversion process. Converters that predict and cancel errors prior to their occurrence are called noise feedback or noise shaping converters. The classic sigma-delta converter is a member of this class of systems.

To assure high levels of correlation between data samples, we significantly oversample the process being encoded. Typical oversample ratios are 16 to 128 times the Nyquist rate of the process. Figure 13.59 presents the block diagram of a simple noise feedback quantizer. In this figure the quantizer is modeled as an additive noise source. The difference between input and output of the quantizer is the quantizer error. We measure this error as the difference between the quantizer input and output ports and present the error to the predicting filter with Z-transform P(Z). This filter, based on previous samples of the quantizer error, predicts a value for the current error and subtracts it from the input signal prior to delivering it to the quantizer. The simplest predicting filter uses the previous error as an estimate of the current error and is in fact merely a one-sample delay register, that is $P(Z) = Z^{-1}$. This substitution for the simple predicting filter is shown in Figure 13.60. In this model we see clearly that the noise is being fed back to the input of the system. The block diagram shown in Figure 13.60 is often redrawn as a two-loop feedback system. The first loop starts at the input goes to the quantizer, but not through it, and returns to the input through the delay line. This loop describes a digital integrator. The second loop starts at the input, goes through the quantizer to the output port, and returns to the input through a delay and negative sign. This alternate description of the noise feedback process is shown in Figure 13.61. Finally, the explicit feedback around the delay line in the first loop forming the digital integrator is suppressed by replacing the loop by its transfer function as shown in Figure 13.62. This is the most common representation of a sigma-delta modulator, one or more integrators and a quantizer in a unity gain feedback loop.

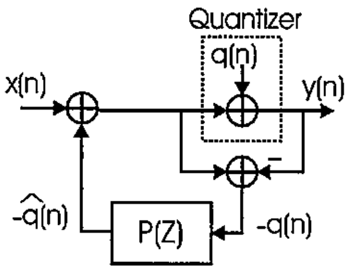


Figure 13.59 Block Diagram of a Noise Feedback Quantizer with Predicting Filter P(Z)

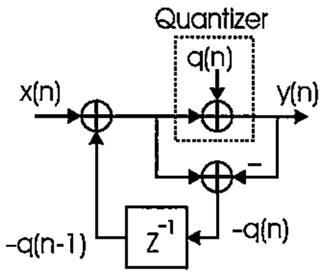


Figure 13.60 Block Diagram of a Noise Feedback Quantizer with Delay Line as Predicting Filter

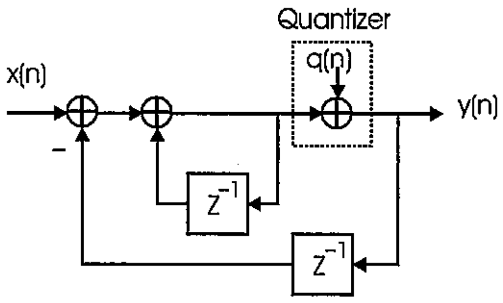


Figure 13.61 Alternate Block Diagram of a Noise Feedback Quantizer Showing Digital Integrator in Feedback Loop

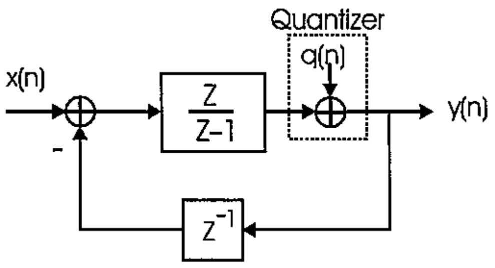


Figure 13.62 Block Diagram of a Noise Feedback Quantizer with Block Diagram of Digital Integrator Replaced by Its Transfer Function

If we model the sigma-delta modulator as a two-input, one-output system we can form the output as the sum of two inputs through their corresponding transfer functions. This is shown in (13.13). When we replace the predicting filter with a delay line Z^{-1} we obtain the relationship shown in (13.14) where we see that the noise transfer function (NTF) is a simple differentiator, $[1-Z^{-1}]$.

$$Y(Z)=X(Z)+Q(Z)[1-P(Z)] \tag{13.13}$$

$$Y(Z)=X(Z)+Q(Z)(1-Z^{-1})=X(Z)+Q(Z)\left[\frac{Z-1}{Z}\right] \tag{13.14}$$

The single zero of the NTF resides at Z = 1, the zero frequency position for sampled data spectra. A stylized power spectrum of the output signal is shown in Figure 13.63. Here we see that the noise has been shaped by the NTF, which has placed a double zero at DC. The double zero suppresses noise in the vicinity of DC but permits noise away from the DC area. Since the input signal has been oversampled by, say, a factor of 64, the spectrum of the input is confined to a small neighborhood around DC, the bottom ± 0.8 % of the sample rate. Thus the sigma-delta loop has arranged to keep noise away from the low pass spectral region that contains the input signal and place it in a spectral region that contains no input signal. Since the zero of the NTF suppresses the noise, the level of noise injected by the quantization process is not important since the product of any finite noise power spectral density and the zero of the NTF is always zero. Consequently, the quantizer is often implemented as a one-bit decision device. While a number of sigma-delta systems are implemented with as many as four bits performing the quantization process, a one-bit quantizer with multiple integrators in the loop is the most common structure. A low pass filter can reject the out-of-band quantization noise, and the sample rate of the filtered data can be reduced in concert with the bandwidth reduction. This certainly sounds like a task for multirate filters.

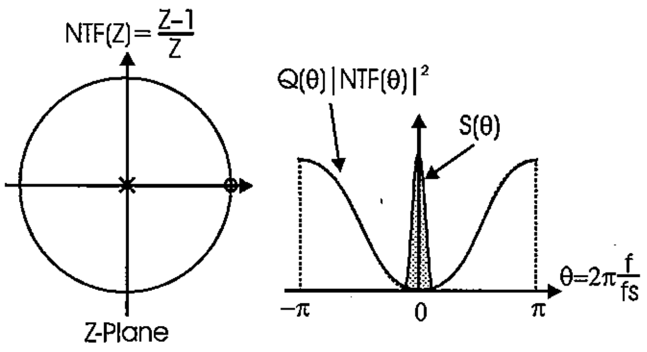


Figure 13.63 Roots of NTF and Power Spectrum of Composite Output Signal of a Noise Feedback Quantizer

13.7.1 Sigma-delta Filter

The filtering task we address here is that of reducing the bandwidth and sample rate of an input data stream formed by a one-bit, two-loop sigma-delta converter operating at 64 times the signal's Nyquist rate. For this example, the signal's two-sided bandwidth is 30 kHz, and we require an output sample rate of 60 kHz. Additional post processing of the oversampled output series is not of interest to us here. The sigma-delta converter input data rate is 3.840 MHz. The dynamic range of the two-loop modulator is 90-dB, so the filtering to be accomplished is equivalent to a single low pass filter with a two-sided bandwidth of 30 kHz, a transition bandwidth of 15 kHz, and a dynamic range of 90-dB with 0.1-dB in-band ripple. The 90-dB is defined by the performance of the 2-loop sigma-delta, which improves quantizing SNR at the rate of 15-dB per doubling of sample rate. The filter following the modulator will of course reduce the sample rate in proportion to the bandwidth reduction. Figure 13.64 is a block diagram of the sigma-delta loop, and Figure 13.65 presents a plot of the input and output time series formed by the modulator as well as the spectrum and a zoomed baseband detail of that spectrum of the output time series.

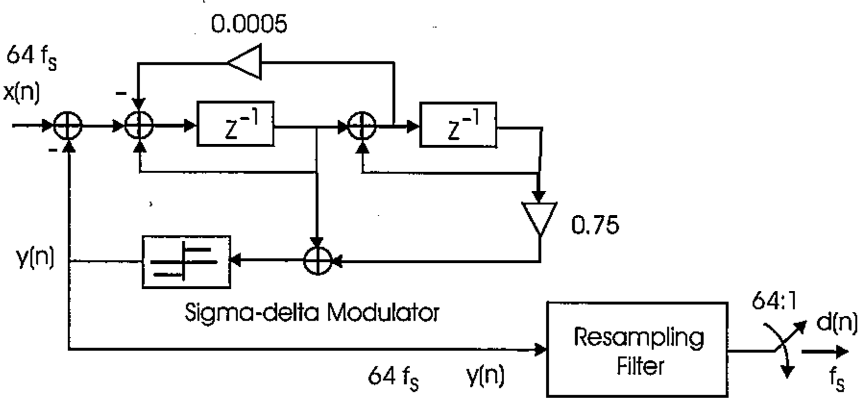


Figure 13.64 Block Diagram of Two-loop, One-bit Sigma-delta Modulator and its Companion Resampling Filter

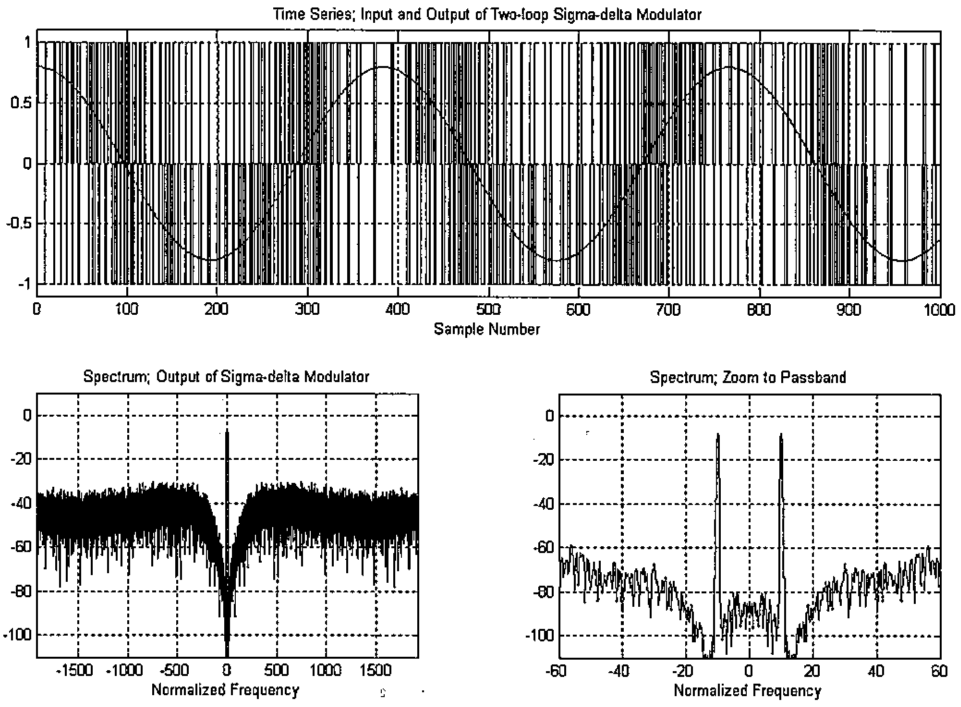


Figure 13.65 Time Series of Input and Output of Sigma-delta Modulator and Spectrum with Zoomed Detail of Output Spectrum from Modulator

The traditional approach to the filtering and resampling task following the sigma-delta modulator is a two-step process comprising a 16-to-1 CIC filter followed by a 4-path polyphase filter or a pair of half-band filters. This approach is similar to the filtering used in digital down converters. A 4th-order CIC is required to meet the 90-dB suppression requirement. The CIC is operated as a Hogenauer filter, simultaneously performing filtering and 16-to-1 down sampling. Figure 13.66 presents the time series formed at the output of the CIC as well as the spectrum of the output and with an overlaid frequency response of the 4-stage CIC filter. Note that the input spectrum contains multiple equal amplitude, non-harmonically related sinusoids. This signal is used to probe the frequency response of the process. Figure 13.67 presents the same information as did Figure 13.66 except here the time series and the spectra are formed after the time series has been down sampled 16-to-1. Here is where we would observe undesired aliasing into the baseband spectral region. None is seen!

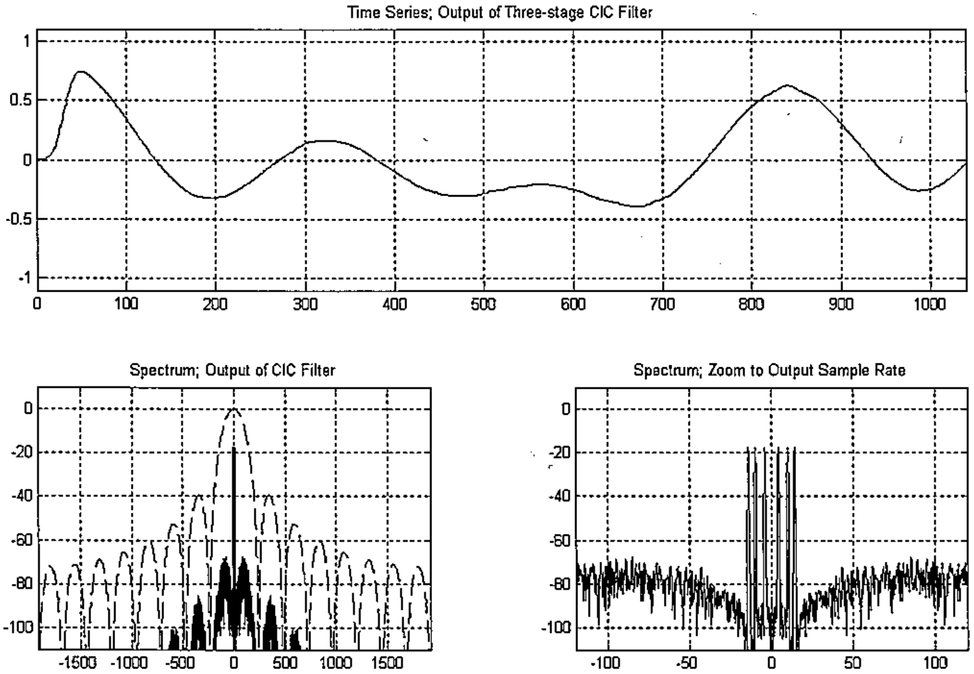


Figure 13.66 Time Series and Spectrum of CIC-filtered Output of Sigma-delta Modulator with Overlaid Filter Response and Zoomed Detail of Output Spectrum

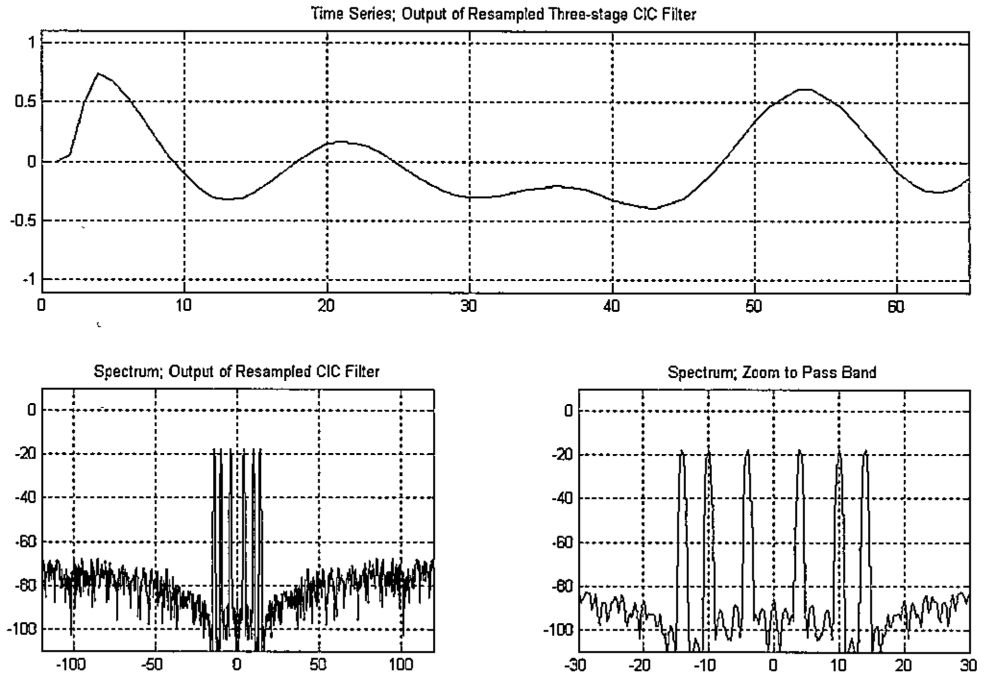


Figure 13.67 Time Series and Spectrum of CIC-filtered and Down-sampled Output of Sigma-delta Modulator and Zoomed Detail of Output Spectrum

The 4-to-1 down-sampling filter following the CIC performs spectral housekeeping, which entails spectral suppression of out-of-band quantizing noise and in-band spectral equalization if required, as well as the sample rate change. The amount of spectral suppression required of this filter is surprisingly small. The required level of additional attenuation can be estimated from the subplot in the lower-left side of Figure 13.67. Here we see that we only require another 20-dB or so to achieve the desired 90-dB attenuation levels. Figure 13.68 presents the time response and spectrum of the output from the final 4-to-1 down sampling filter. Also overlaid on the output spectrum is the frequency response of this 24-tap filter. We also see the spectrum of the final down-sampled time series and note that the spectral terms are 90-dB below full-scale response. What we have illustrated here is that the combination of noise shaping with the 2-loop sigma-delta modulator along with the spectral suppression performed by the cascade of two resampling filters has been able to convert a highly oversampled, hence highly correlated, signal with a 1-bit noise shaping converter to 15-bits of uncorrelated data samples.

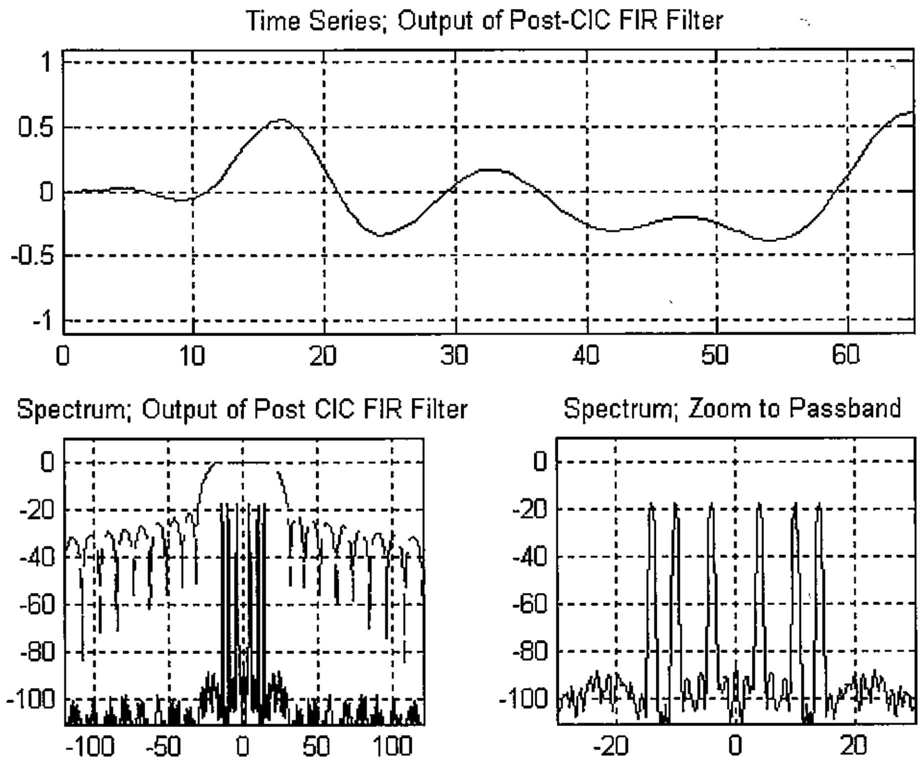


Figure 13.68 Time Series and Spectrum of Second FIR Filtered and Down-sampled Output from CIC Filter with Filter Overlay and Zoomed Detail of Output Spectrum

It may have occurred to you that the filtering performed by the cascade of the CIC and the 4-path polyphase filter did not take advantage of the fact that the data formed by the sigma-delta modulator is bipolar and of fixed amplitude, in fact only the sign of the modulated process. Processing such data in a FIR filter can be accomplished without the need for multiplications. This awareness was not obvious because the CIC filter that processed the modulator output had no multiplies. We depart here from the standard solution and ask about a polyphase FIR filter that can perform the same 16-to-1 bandwidth and sample-rate reduction as the CIC filter. A polyphase FIR filter that accomplishes the same spectral suppression requires four-coefficients per polyphase arm. The data delivered to the polyphase filter from the sigma-delta modulator is a 1-bit sample, the sign-bit, that conceptually is delivered to successive filter paths by the input commutator. Thus the content of each path register is a +1 or −1 in each of the four register positions. The output from any given path is simply the weighted sum of the filter weights where the weighting terms are ±1. The polyphase filter requires four-sums for each input sample, but of course the four-stage CIC filter also requires four-sums for each input sample. That's interesting! The 16-path polyphase filter requires a total of 64 sums to form one output sample at the output rate. The four-stage CIC requires the same number of sums to perform the arithmetic in the four-overflowing accumulators. The output of the final accumulator is then passed to the four-derivatives in the CIC chain to perform four more sums at the output rate. In terms of hardware resources, the four accumulators in the CIC can be mapped to the 4 accumulators in the dual form of the polyphase filter presented in Figures 5.11 and 5.12. This accumulator variation is shown in Figure 13.69. The advantage of using the multiply-free form of the polyphase filter is that we can control the in-band spectral characteristics of the filter, an option not available to the CIC implementation. Another consideration applicable to FPGA implementation is that the content of each polyphase filter path is a four-bit binary word.

There are only 16 possible outputs that any given stage can deliver to its output port. These outputs can be precomputed and stored in a 16-element array unique to each path that is addressable from the four-bit word stored in the input register for that path. The sum of the outputs from the 16 paths is the down -ampled and filtered time series. The 16-path filter then only requires 16 table accesses and 16 sums rather than the 64 adds required by the equivalent CIC. Figure 13.70 shows the look-up table implementation of the 16-path polyphase filter operating with binary inputs from the sigma-delta modulator.

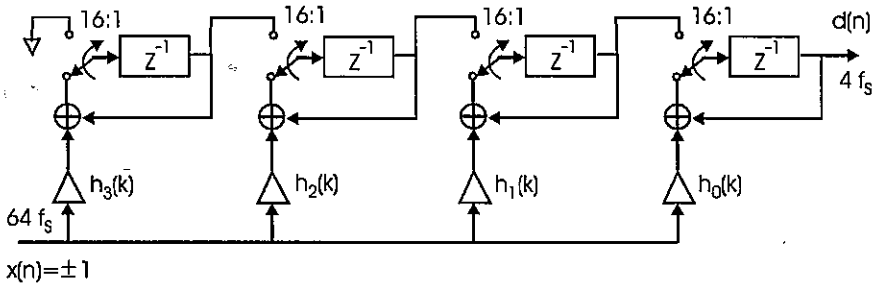


Figure 13.69 Dual Form Filter Showing Accumulators in 4-tap Polyphase Filter

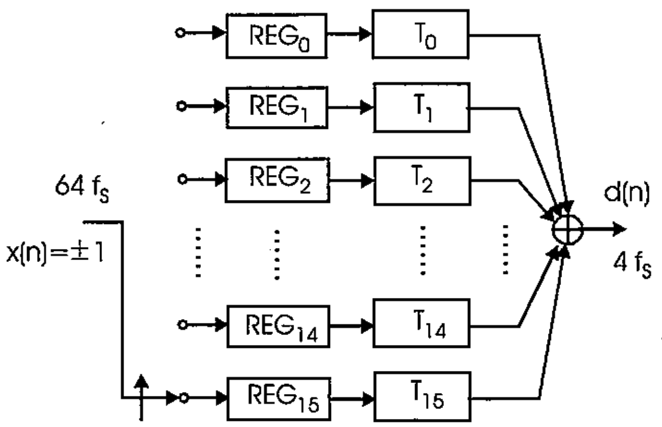


Figure 13.70 Polyphase Filter Implemented with Short Look-up Tables Addressed from Binary Input Data Delivered from Sigma-delta Modulator

Figures 13.71 and 13.72 present the output time series and spectra formed by the polyphase filter performing the first filtering task previously performed by the CIC filter. The two figures correspond to data taken prior to and after the 16-to-1 down sampling operation. Also shown in Figure 13.71 is an overlaid spectral description of the 4-tap 16-arm polyphase filter. Note that the overall filtering effect is the same as that performed by the CIC and presented in Figures 13.66 and 13.67. A housecleaning filter, following the polyphase filter, finishes the filtering task and reduces the sample rate but does not have to correct the sinx/x spectral tilt. Figure 13.73 presents the time response and spectrum of the output from this final 4-to-1 down-sampling filter. Also overlaid on the output spectrum is the frequency response of this 24-tap filter. We also see the spectrum of the final down-sampled time series and note that the spectral terms are 90-dB below full-scale response.

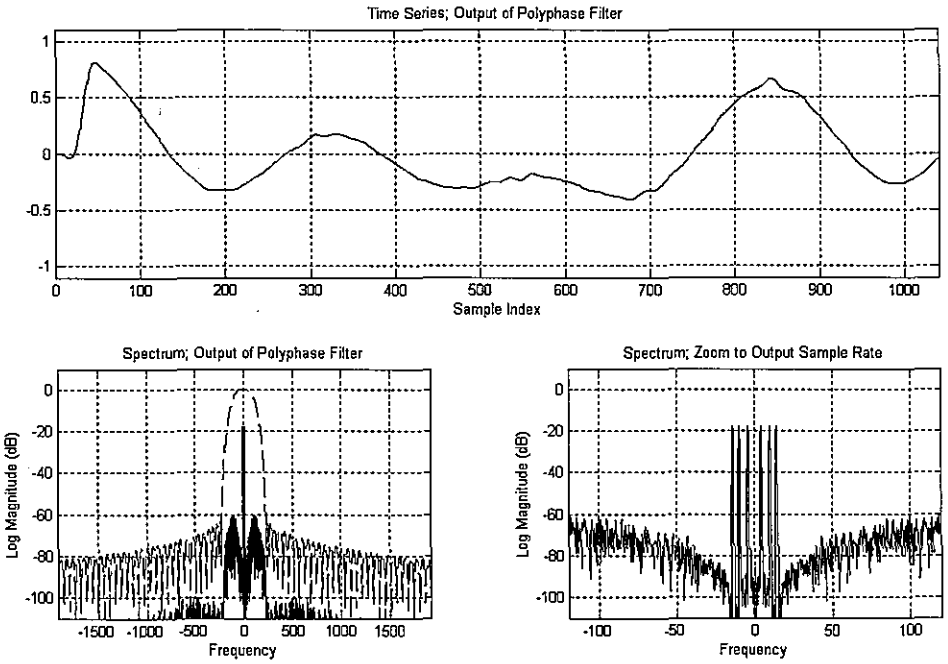


Figure 13.71 Time Series and Spectrum of Polyphase-filtered Output of Sigma-delta Modulator with Overlaid Filter Response and Zoomed Detail of Output Spectrum

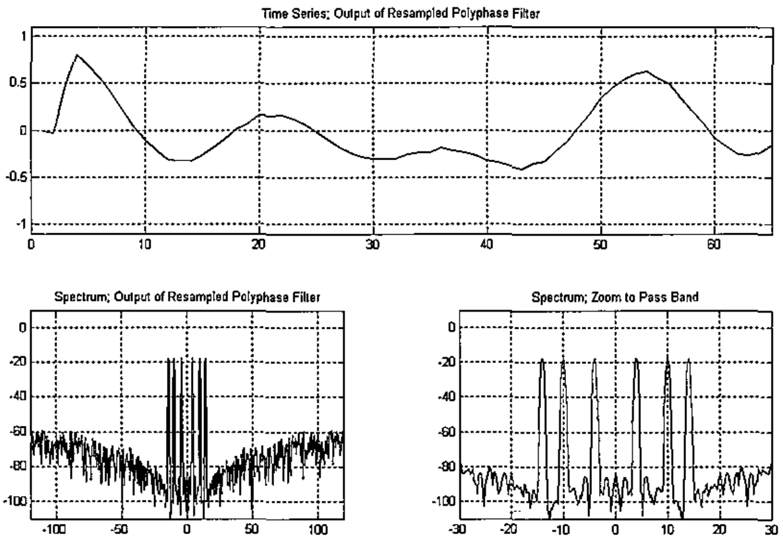


Figure 13.72 Time Series and Spectrum of Polyphase-filtered and Down-sampled Output of Sigma-delta Modulator and Zoomed Detail of Output Spectrum

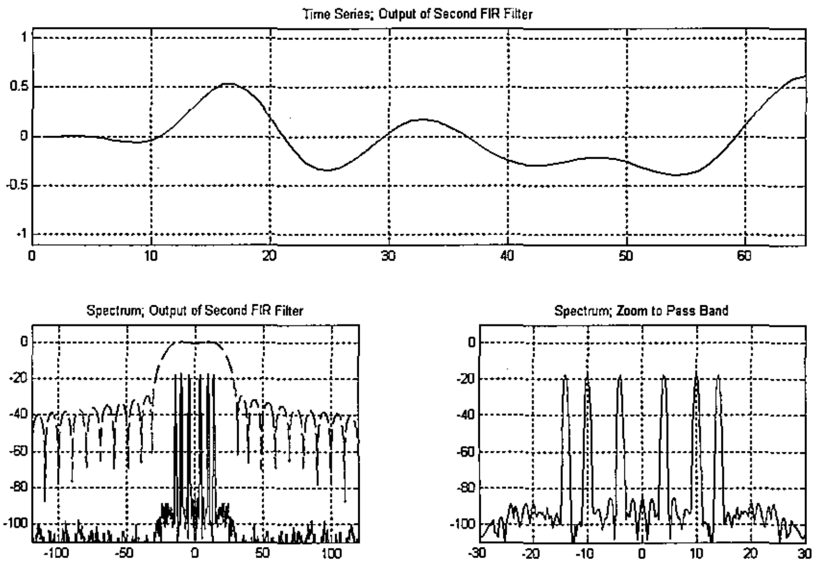


Figure 13.73 Time Series and Spectrum of Final 4-to-1 Down-sampled Filter and Detail of Output Spectrum

13.8 FM Receiver and Demodulator

Multirate signal processing has had a significant influence at the physical or hardware layer of modern communication systems. In particular, multirate signal processing is found at the core of communication systems that couple the Software Defined Radio (SDR) and Software Communications Architecture (SCA) to reconfigure system resources for operation over a wide range of modulation formats and waveforms. In this section we demonstrate one particularly efficient multirate signal-processing solution to the task of implementing a radio configured to select and extract a single FM channel from the commercial FM band, to down convert and demodulate that channel, and to perform stereo demodulation and separation of the resulting baseband signal. Sprinkled through this discussion are descriptions of the necessary background material required to understand the various processing requirements.

In the United States, the commercial FM band spans the frequency interval from 880 MHz to 108 MHz with the different FM stations allocated carriers separated by 200 kHz spacing starting at 88.1 MHz. European FM stations are separated by 100 kHz intervals. The frequency modulation index of the FM signal is 75 kHz, which from Carson's rule leads to a nominal two-sided bandwidth of approximately 180 kHz. The compatible stereo signal, commonly carried by FM stations, is a pair of 15 kHz bandwidth audio signals transmitted as the sum (L + R) and difference (L – R) of the desired audio component signals denoted L and R for left and right respectively. The L + R signal resides at baseband while the L – R signal is double-sideband suppressed carrier (DSB-SC) modulated to a 38 kHz AM subcarrier. A 19-kHz pilot signal, inserted between the L + R baseband signal and the L – R subcarrier signal, is extracted by the receiver and frequency-doubled to obtain a phase-coherent reference signal required for the DSB-SC down conversion. The separated signals are then summed and differenced to form 2L and 2R, the desired stereo components. Figure 13.74 presents a block diagram of a stereo FM receiver as well as the spectral representation of the input and output spectra of the receiver. Also shown in this figure is a second subcarrier, the subsidiary carrier authorization (SCA) signal, carried by many stations.

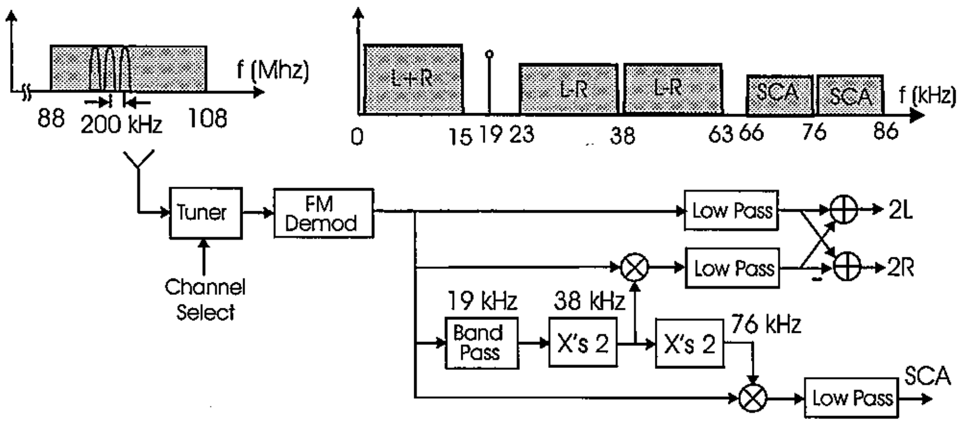


Figure 13.74 Input and Output Spectra of Stereo FM Receiver and Block Diagram of Conventional FM Receiver and Stereo Demodulator

13.8.1 FM Band Channelizer

The tuner section of the FM receiver can be implemented as shown in Figure 13.75. The analog section of the receiver contains an antenna, a band-pass filter with bandwidth matching the 88-to-108 MHz FM band, a gain controllable RF amplifier, and a 10-bit ADC converter operating at an 80-MHz sample rate. The ADC operating at the 80-MHz sample rate performs IF sampling and aliases the FM band centered at 98 MHz to 18 MHz, close to the quarter-sample rate of the converter.

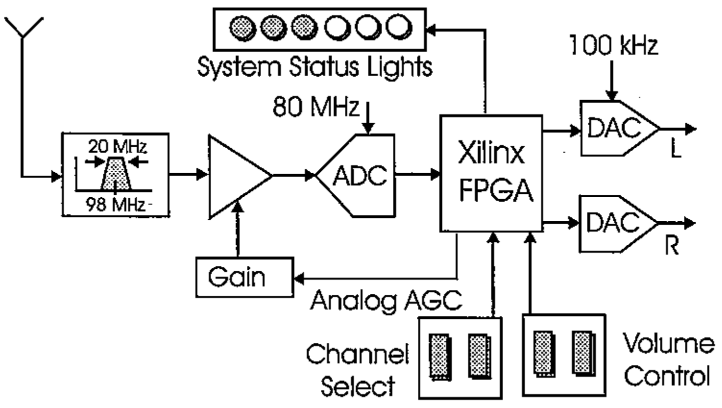


Figure 13.75 Block Diagram of DSP-based FM Channelizer

The digital section of the receiver, residing in the Field Programmable Gate Array (FPGA), contains a polyphase filter that restricts the signal bandwidth and reduces the sample rate 200-to-1 to 400-kHz. The 400-kHz rate was selected to permit wide transition bandwidth in the filter. The spectral response of the baseband prototype filter designed for the 200-to-1 resampling filter is shown in Figure 13.76. A 1200-tap FIR filter operating at 80-MHz sample rate satisfies the spectral specifications indicated in this figure. When implemented as a 200-path polyphase filter, the length of each path is a reasonable 6-taps, which requires six operations per input sample. The total workload per input sample is increased when we include the complex phase rotators required to extract the desired signal. We now examine a few options that accomplish the channel selection along with the bandwidth and sample rate reduction afforded by the polyphase filter partition.

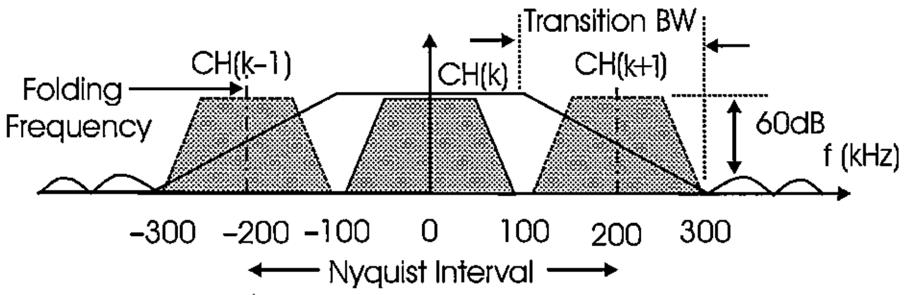


Figure 13.76 Spectral Characteristics of Prototype Low-pass Filter in Polyphase Receiver

The first approach to extract the desired channel from the sampled data stream is a standard complex heterodyne and low pass filter pair with the filters implemented in a polyphase structure that performs simultaneous bandwidth reduction and sample rate reduction. This option is shown in Figure 13.77. Here we see that the workload for the real or imaginary output is 7-multiplies per input sample, 1-multiply for the heterodyne and 6-multiplies for the polyphase filter. The total workload for this implementation is 14-multiplies per input sample point.

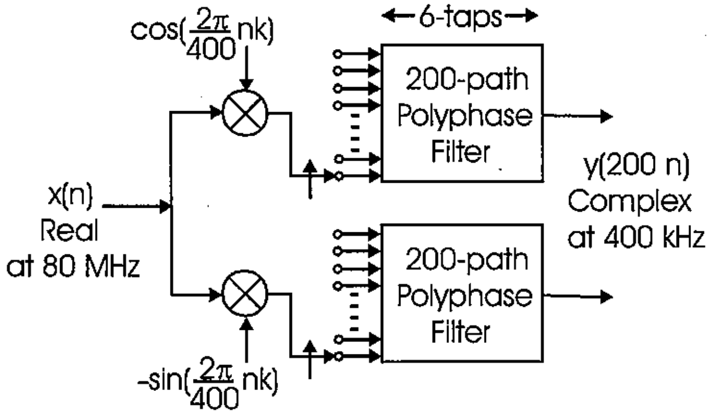

$\cos(\frac{2\pi}{400}nk)$
6-taps
200-path Polyphase Filter
200-path Polyphase Filter
x(n) Real at 80 MHz
y(200 n) Complex at 400 kHz
$-\sin(\frac{2\pi}{400}nk)$

Figure 13.77 Standard Heterodyne, Filter, and Down-sample Architecture Channel Selector

We can apply the equivalency theorem to this channel-selection task to obtain a desired reduction in processing load. Rather than downconvert the selected channel to baseband by an on-line heterodyne, we can upconvert the prototype low pass filter to the selected channel as an off-line heterodyne. The savings we incur is that the on-line post-filtering downconversion, if necessary, occurs at the reduced output rate rather than at the high input rate. The band-pass version of the channel selection process is shown in Figure 13.78. In this system, a signal centered at 8.0 MHz, the image of 88.0 MHz prior to IF sampling, aliases to baseband in the 200-to-1 down sampling to 400 kHz. Thus a signal centered at 8.1 MHz, the first FM channel, aliases to 100 kHz, the quarter-sample rate at the 400 kHz output rate. The required complex heterodyne following the down sampling is a simple translation from the quarter sample rate to baseband. This operation is performed without actual multiplication since the values of the cosine and sine can be restricted to ±1 and 0. The workload for this form of the channelizers is 6-multiplies per input sample for each of the real and imaginary output port, thus the total workload is 12-multiplies per input sample. By embedding the heterodyne in the filter, we have saved the workload required by the heterodyne process.

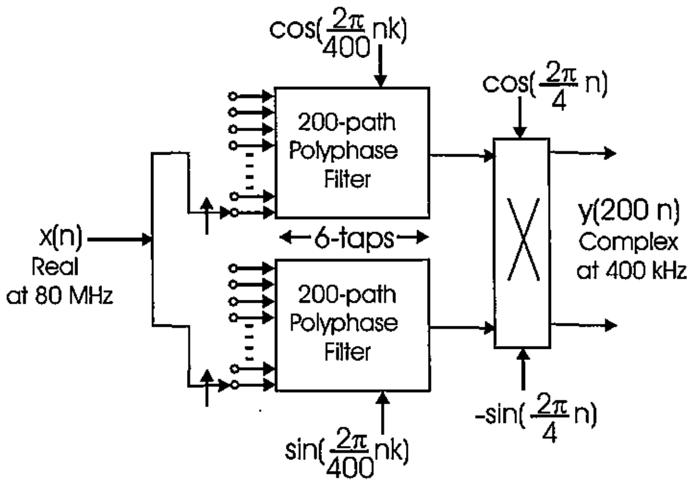

cos(2π/400 nk)
200-path Polyphase Filter
6-taps
200-path Polyphase Filter
sin(2π/400 nk)
x(n) Real at 80 MHz
cos(2π/4 n)
-sin(2π/4 n)
y(200 n) Complex at 400 kHz

Figure 13.78 Heterodyned Filter, Down-sample, and Down-convert Architecture Version of Channel Selector

A lesson learned early in multirate filtering is that it is wiser to make the data complex as we leave a processing algorithm than it is to make it complex as we enter the algorithm. In the standard channel-centered polyphase filter structure, the data is made complex by the output phase rotators that unwrap the desired alias from the multiple aliases that have been baseband by the aliasing caused by the resampling process. In the standard polyphase filter structure all multiples of the output sample rate alias to baseband. In the system we are examining, these frequencies are all multiples of 400 kHz. As noted in the previous paragraph, the original 88 MHz, aliased to 8.0 MHz due to the IF sampling, is a multiple of 400 kHz and folds to zero frequency in the standard polyphase structure. The first FM channel is centered 100 kHz above 88.0 MHz, and this frequency will fold to the quarter sample rate at the output sample rate of the polyphase filter. In fact, all the desired channels will fold to ± 100 kHz. We now examine a variant of the standard polyphase filter structure that will fold and translate the FM spectral centers to zero frequency. Equation (13.15) presents the Z-transform of the frequency-translated version of the prototype filter impulse response while (13.16) presents the 1-to-M polyphase partition of the same.

$$H(Z) = \sum_{n=0}^{N-1} h(n)\, e^{j\frac{2\pi}{M}nk} Z^{-n} \tag{13.15}$$

$$
\begin{aligned}
H(Z) &= \sum_{r=0}^{M-1} \sum_{n=0}^{\frac{N}{M}-1} h(r+nM)\, e^{j\frac{2\pi}{M}(r+nM)k} Z^{-(r+nM)} \\
&= \sum_{r=0}^{M-1} e^{j\frac{2\pi}{M}rk} Z^{-r} \sum_{n=0}^{\frac{N}{M}-1} h(r+nM) e^{j\frac{2\pi}{M}nMk} Z^{-nM}
\end{aligned}
\tag{13.16}
$$

When the frequency index k is an integer, 2π nk is congruent to 2π, and the selected frequency bin, bin k, aliases to zero in the polyphase partition. A variant of this relationship is to be seen by replacing k with k + s/4, for s = 0, 1, 2, or 3. This is shown in (13.17) where we see that the inner sum representing the operation of the polyphase stages still has a phase shift that varies with the time index *n*. For the example developed here, the residual phase-shift term is trivial, simply being powers of j. In operation, when the path coefficients are loaded into the path filters, the coefficients are rotated by the path rotations exp(j 0.5 π n) for s = 1 or exp(−j 0.5 π n) for s = 3. This pre-rotation of the weights results in the successful down conversion, by the resampling operation, of the frequency components of the FM channels offset ± 100 kHz from the multiples of 400 kHz. Note that the offset is also embedded in the phase rotators on each polyphase arm that are applied in the outer summation of (13.17).

$$
\begin{aligned}
H(Z) &= \sum_{r=0}^{M-1} e^{j\frac{2\pi}{M}r(k+s/4)} Z^{-r} \sum_{n=0}^{\frac{N}{M}-1} h(r+nM)\, e^{j\frac{2\pi}{M}nM(k+s/4)} Z^{-nM} \\
&= \sum_{r=0}^{M-1} e^{j\frac{2\pi}{M}r(k+s/4)} Z^{-r} \sum_{n=0}^{\frac{N}{M}-1} h(r+nM)\, e^{j\frac{2\pi}{4}ns} Z^{-nM}
\end{aligned}
\tag{13.17}
$$

embedding the j phase rotator in the path weights has a slight impact on the structure of the polyphase filter arms and the subsequent phase rotator. While no actual complex products are involved in the polyphase arms, the data samples formed by the polyphase arms are now complex rather than real. This means that the formerly complex scalar phase rotators applied at the stage output now requires a full complex product. The structure of the modified polyphase filter is shown in Figure 13.79. The workload for this version of the channel select-process is seen to be 6-multiplies per input sample for the polyphase filter and 4-multiplies for the output complex rotator for a total workload of 10-multiplies per input sample. This is the most efficient of the three variants of the channel selection process we have examined.

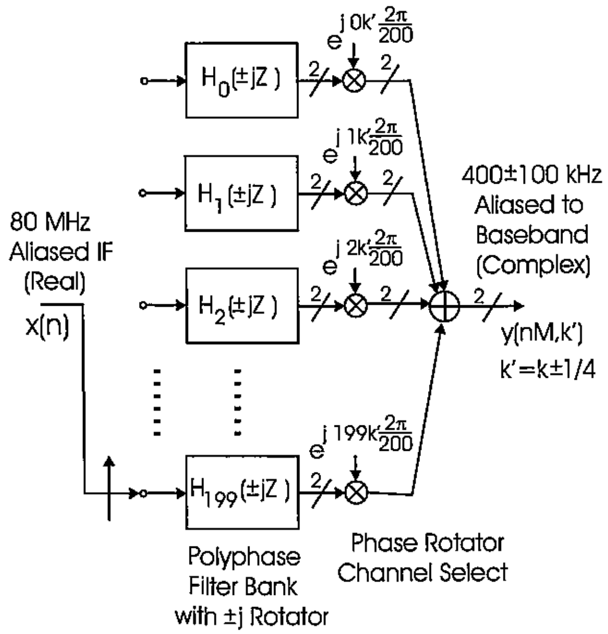


Figure 13.79 Signal Flow Diagram of Modified Polyphase Filter to Permit Frequencies Offset by Quarter of Output Sample Rate to Alias to Baseband

13.8.2 FM Demodulator

For completeness, we now present a short description of the digital FM demodulator. An FM demodulator performs the task of extracting the modulation signal from the FM modulated waveform. Since frequency is the time derivative of a sinusoid's phase angle, an FM demodulator must form the derivative of the received signal's phase. In analog systems this process is accomplished by a circuit called a discriminator that applies a sequence of operators to the carrier-centered FM signal. These operators are a hard-limiter that removes incidental AM from the signal, a derivative circuit that converts the FM signal to an AM signal, and a diode detector that responds to the resulting amplitude variations. A balanced version of this system is used to linearize the derivative process while canceling its DC component.

In the digital receiver, the channel-selection process forms a complex baseband signal with the I(n) and Q(n) components defining the complex envelope of the FM signal. The I-Q components define the modulated phase angle of the original FM signal, and the digital FM demodulator must form the derivative of that phase angle. One implementation forms the angle $\theta(n)$ of the ordered pair by an arctangent or by a CORDIC routine and then forms its derivative with a FIR filter. In another implementation, the derivative of the arctangent is formed in a single step by the relationship shown in equation (13.18) and illustrated in Figure 13.80 without the denominator scaling. The scaling term serves the same function as the hard limiter in an analog discriminator, that of removing amplitude modulation from the input signals.

$$\dot{\theta}(n)=\frac{d}{dn}\arctan\left(\frac{I(n)}{Q(n)}\right)=\frac{I(n)\dot{Q}(n)-\dot{I}(n)Q(n)}{I^2(n)+Q^2(n)} \tag{13.18}$$

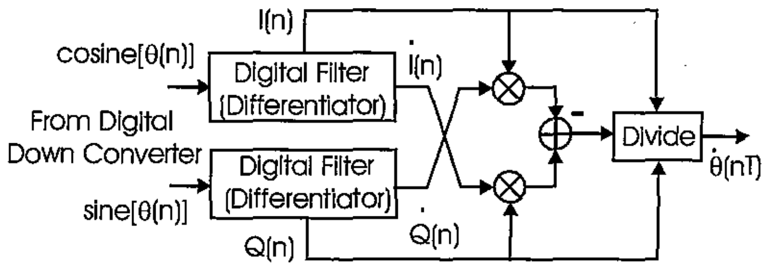

I(n)
cosine[θ(n)]
Digital Filter (Differentiator)
İ(n)
From Digital Down Converter
Digital Filter (Differentiator)
sine[θ(n)]
Q(n)
Q̇(n)
-
Divide
θ̇(nT)

Figure 13.80 Signal-flow Diagram of Digital FM Discriminator

13.8.3 Stereo Decoding

Stereo decoding is a straightforward task that closely follows the signal flow suggested in Figure 13.74. The interesting part of this problem is the processing to extract the pilot signal and then frequency double it for use in the demodulation of the DSB-SC modulated L – R signal. We first examine the length of the filter required to isolate the pilot signal and then present a clever multirate implementation of the pilot extraction and frequency-doubling task. The brute-force filter designed to isolate and extract the pilot signal would have to satisfy the specifications indicated in Figure 13.81. A filter with approximately 273 taps is required to satisfy these specifications. Since the filter bandwidth is a very small fraction of the sample rate, we know that we can use multirate signal processing to implement a more efficient option. We now examine one such option based on down sampling, filtering, and up sampling with a delightful little twist unique to this application.

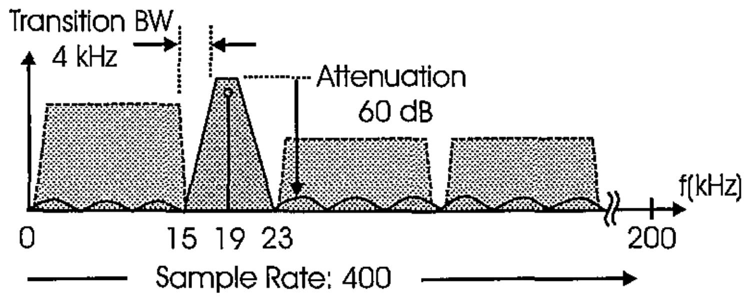


Figure 13.81 Filter Specifications for Pilot Extraction Filter

The structure of an alternate pilot extraction filter is shown in Figure 13.82 and the spectra of the signal at various points in the processing scheme can be seen in Figure 13.83. In this structure we perform a 20-to-1 downsample with a 20-stage polyphase filter with phase rotators that extracts the first Nyquist zone centered at 20 kHz. The prototype low-pass filter used in this partition has a large transition bandwidth equal to 19 kHz, which requires a filter with 60 taps. The output of the phase-rotated polyphase filter contains the alias of the 19-kHz pilot now located at –1.0 kHz with a sample rate of 20 kHz. The filter following the polyphase filter limits the bandwidth around the pilot signal, and it is designed as a low pass filter with two-sided bandwidth of 2 kHz and transition bandwidth of 2.0 kHz. This filter requires 30 taps. The complex signal output by the low pass filter contains only the aliased pilot signal. We now double the frequency of this signal from –1 kHz to –2 kHz by squaring the complex samples. We now up sample the double frequency aliased pilot by a factor of 1-to-20 in a second polyphase filter. The phase rotators at the input to the second polyphase are selected to output a real signal in the second Nyquist zone centered at 40 kHz. This aliases the –2-kHz baseband signal to 38 kHz, the carrier frequency required for the DSB-SC demodulation of the L – R signal component. The delight found in this process is that the frequency doubling is performed at baseband between the down-sampling and up-sampling processes and that we have selected different Nyquist zones in the down-sampling and up-sampling process.

The workload required for this technique of pilot extraction and doubling can be determined as follows. The input polyphase filter requires 3-multiplies and adds for each path filter and 2-multiplies and adds for the path phase rotators for a workload of 5-multiplies per input sample. Similarly, the output polyphase filter, the dual of the input filter, also requires 5-multiplies and adds per output sample. The filtering performed at the reduced 20-kHz sample rate requires 30-multiplies and adds for each of the real and imaginary filter legs and 4-multiplies and 2-adds in the complex product that doubles the frequency of the extracted pilot. The total of 64-multiplies is amortized over 20 input or output samples so that the filtering load is 3.2-multiplies per input. The total workload, the sum of these terms, is seen to be approximately 13.2-multiplies and adds per input-output pair. This is a significant improvement relative to the 273-multiplies and adds required of the brute-force, nonmultirate filter implementation.

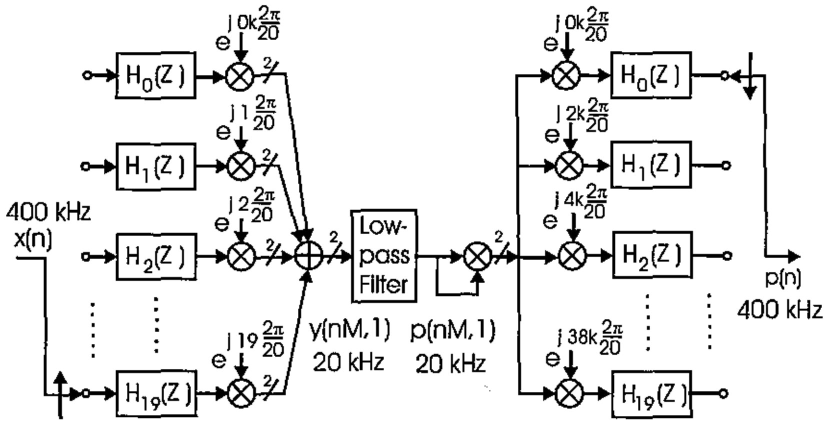


Figure 13.82 Pilot Extraction and Doubling by Polyphase Down-sample, Filter, Frequency Doubling, and Polyphase Up-sample

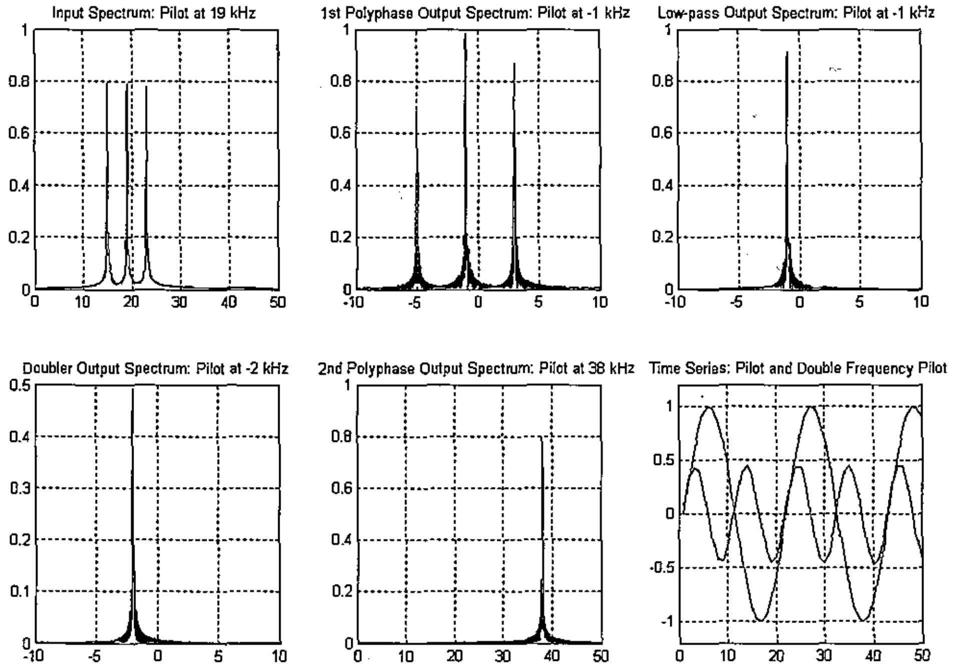


Figure 13.83 Spectrum of Input and Output of Polyphase Down Sampler, at Output of Low-pass Filter, at Output of Squaring Circuit, and at Output of Poly-

phase Up-sampling Filter. Also Time Series of Pilot Component of Input Signal and Double Frequency Pilot at Output of Process

References

Aziz, Pervez, Henrik Sorenson, and Jan Van Der Spiegel, "An Overview of Sigma Delta Converters: How a 1-bit ADC Achieves more than 16-bit Resolution," *IEEE Signal Processing Magazine*, , Vol. 13, No. 1, pp. 61–84, Jan. 1996.

Candy, James, and Gabor Temes, *Oversampling Delta-sigma Data Converters: Theory, Design, and Simulation*, Piscataway, NJ, IEEE Press, 1992.

Crochiere, Ronald, and Lawrence Rabiner, *Multirate Signal Processing*, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1983.

Dick, Chris., fred harris and Michael Rice, "Synchronization in Software Radios: Carrier and Timing Recovery Using FPGAs," IEEE Symposium on Field-Programmable Custom Computing Machines, Napa Valley, CA, April 16–19, 2000.

Fliege, Norbert, *Multirate Digital Signal Processing: Multirate Systems, Filter Banks, Wavelets*, West Sussex, John Wiley & Sons, Ltd., 1994.

harris, fred, "Carrier Synchronization Techniques for DSP-based Modems," *Communication Systems Design Magazine*, (Trade Journal), pp. 32–34, 36, 38–39, 42, July 2000.

harris, fred, "A Fresh View of Digital Signal Processing for Software Defined Radios, Part I and Part II," International Telemetry Conference (ITC), San Diego, CA, Oct. 21–24, 2002.

harris, fred, "Sigma-Delta Converters in Communication Systems," *Encyclopedia of Communications*, John Wiley & Sons, John Proakis, Editor, 2002.

harris, fred, "Teaching MODEM Concepts and Design Procedure with MATLAB Simulations," Signal Processing Education, Workshop, Kent, Texas, Oct. 15–18, 2000.

harris, fred, and Chris Dick, "On Structure and Implementation of Algorithms for Carrier and Symbol Synchronization in Software Defined Radios," URSIPCO-2000, European Association for Signal Processing, Session on Efficient Algorithms for Hardware Implementations of DSP Systems, Tempere, Finland, Sep. 5–8, 2000.

harris, fred, and Chris Dick, "Performing Simultaneous Arbitrary Spectral Translation and Sample Rate Change in Polyphase Interpolating and Decimating Filters in Transmitters and Receivers," 2002-Software Defined Radio Technical Conference, San Diego, CA, Nov. 11–12, 2002.

harris, fred, and Michael Rice, "Multirate Digital Filters for Symbol Timing Synchronization in Software Defined Radios," *IEEE Journal on Selected Areas in Communications*, Vol. 19, pp. 2346-2357, Dec. 2001.

Hentschel, Tim, *Sample Rate Conversion in Software Configurable Radios*, Norwood, MA, Artech House, Inc., 2002.

Jovanovic-Dolecek, Gordana, *Multirate Systems: Design and Applications*, London, Idea Group, 2002.

Mitra, Sanjit, *Digital Signal Processing: A Computer-based Approach*, 2nd ed., New York, McGraw-Hill, 2001.

Mitra, Sanjit, and James Kaiser, *Handbook for Digital Signal Processing*, New York, John Wiley & Sons, 1993.

Norsworthy, Steven, Richard Schreier, and Gabor Temes, *Delta-sigma Data Converters: Theory, Design, and Simulation*, Piscataway, NJ, IEEE Press, 1997.

Rice, M. and harris, f., "Polyphase Filter Banks for Symbol Synchronization in Sampled Data Receivers," MILCOM-2002, Anaheim, CA, Oct. 7–10, 2002.

Rice, M., harris, f., and Chris Dick, "Maximum Likelihood Carrier Phase Synchronization In FPGA-based Software Defined Radios," ICASSP-2001, International Conference on Acoustics, Speech and Signal Processing, Salt Lake City, Utah, May 7–11, 2001

Vaidyanathan, P. P., *Multirate Systems and Filter Banks*, Englewood Cliffs, NJ, Prentice-Hall, Inc., 1993.

Problems

13.1 Equation 13.1 presented the Taylor series for $[\sin(\theta)/\theta]^P$, the main lobe response of the CIC filter. Determine the value of θ for which the droop due to the main lobe curvature is -0.1-dB and then 0.5-dB. What signal bandwidth, as a fraction of the sample rate, corresponds to these values of θ?

13.2 A signal with two-sided bandwidth of 20 kHz is centered on an analog 450 kHz intermediate frequency carrier. Suppose we select a sample rate of 360 kHz to perform IF sampling and alias the signal from the center frequency to a new digital-center frequency. Draw a block diagram of this process. Show where the spectrum is located, and describe the remaining processing required to bring the signal to baseband with filtering and resampling to 90-kHz output sample rate.

13.3 A signal with two-sided bandwidth of 20 kHz is centered on an analog 450-kHz intermediate frequency carrier. Suppose we select a sample rate of 120 kHz to perform IF sampling and alias the signal from the center frequency to a new digital center frequency. Draw a block diagram of this process. Show where the spectrum is located, and describe the remaining processing required to bring the signal to baseband with filtering and resampling to 40-kHz output sample rate. Be careful of spectral inversion.

13.4 The information a receiver needs to determine correct timing information from a received signal is obtained from the output of a matched filter and the derivative matched filter. We now examine the transfer function of the time error detector. Form a SQRT-Nyquist pulse oversampled by 10-to-1. Also form a corresponding matched filter and a derivative matched filter with the same over sampling ratio. Be certain that the derivative filter and the matched filter are time aligned, that is, that the zero crossing of the derivative filter matches the peak of the matched filter. Convolve the impulse response of the shaping filter with impulse response of the matched filter and of the derivative matched filter. Form the dot product of the two responses, and plot the results. This function is the timing error detector function. Determine its slope in units of amplitude per time-offset. If the amplitude of the input signal is doubled, what happens to the gain of the time offset error detector? How does this impact the time-offset error for a QAM signal as opposed to a QPSK signal?

13.5 Determine how to form a band-edge filter from a matched filter operating at 4-samples per symbol. Hint: This will require a 4-path polyphase partition, but no resampling, of the matched filter. Where is the band edge located for this process? Can we reduce the sample rate by a factor of 2 as part of separating the two band edges? Why would one try to do this?

13.6 Form a simple band-edge filter by operating the matched filter at 2-samples per symbol and extracting the overlap of the low pass and high pass polyphase partition with a Hilbert transform filter. Feed the band-edge filters from the output of a shaping filter also running at 2-samples per symbol and modulated with 16-QAM random data. Introduce selectable frequency offsets to the modulated signal, and form a plot of the average output level from the energy difference of the two band-edge filters as a function of frequency offset. Be sure to test both positive and negative offsets. Determine the gain of the frequency-offset detector. Is this gain a function of modulation constellation density? Is this gain a function of input signal level?

13.7 A simple digitally controlled, digital delay line can be formed by a cascade of one or more identical all-pass networks with transfer function $H(Z) = (1 - \alpha\, Z)/(Z - \alpha)$. This is particularly effective when the signal bandwidth is a small fraction of the sample rate, as might occur when a signal is oversampled by a factor of 4. Under this condition, the all-pass network exhibits approximately linear phase shift in the signal bandwidth. Determine the expression for phase slope of the all-pass network. You will find assistance in Chapter 10. Then determine group delay as a function of the parameter α.

13.8 Form a delay line using three stages of the recursive one-tap all-pass filter presented in Chapter 10. Form a SQRT-Nyquist pulse that is oversampled by a factor 10-to-1 and then pass the SQRT-Nyquist pulse through the delay line filter chain with the parameter α set to zero, plot the input and output signal, and verify the delay is 3-samples. Repeat the experiment for a range of α between -0.5 and $+0.5$ in increments of 0.1. Examine the relationship between the observed delay and the setting for the parameter α.

Index

Signal Processing

Using multirate signal processing to reduce costs and improve performance

Multirate signal processing can reduce costs and improve performance in applications ranging from laboratory instruments to cable modems, wireless systems, and consumer entertainment products. This book offers the first systematic, clear, and intuitive introduction to multirate signal processing for working engineers and system designers.

The author uses extensive examples and figures to illuminate a wide range of multirate techniques, from basic resampling to leading-edge cascade and multiple-stage filter structures. Along the way, he draws on extensive research and consulting experience to introduce processing "tricks" shown to maximize performance and efficiency. Coverage includes:

- Effect of sampling and resampling in time and frequency domains
- Relationships between FIR Filter specifications and filter length (taps)
- Window design and equal-ripple (Remez) design techniques
- Square-Root Nyquist and Half Band Filters, including new design enhancements
- Polyphase FIR Filters: up-sampling, down-sampling, and cascade up-down sampling
- Polyphase interpolators and filters that perform arbitrary sample rate change
- Dyadic half band filters, including quadrature mirror FIR filters
- Polyphase Channelizers, including M-path modulators, demodulator channel banks, simultaneous interpolation, and channel bank formation
- Comprehensive coverage of recursive all-pass filters—a topic never before covered in this detail
- Comparisons with traditional DSP design techniques
- Extensive applications coverage throughout

About the Author

fredric j harris is Professor of Electrical and Computer Engineering at San Diego State University, where he holds the CUBIC Signal Processing Chair of the Communication Systems and Signal Processing Institute. He has taught DSP and communication technology at SDSU since 1967. He holds several patents on digital receiver and DSP technology, lectures on DSP worldwide, and consults with many organizations building high-performance DSP systems, including the SPAWAR, Lockheed, Scientific Atlanta, Hewlett-Packard, Tektronix, TRW, CUBIC, and Motorola.

http://authors.phptr.com/harris/

PRENTICE HALL
Professional Technical Reference
Upper Saddle River, NJ 07458
www.phptr.com

METU LIBRARY
0050348939
146511-2
59600
0 76092 02915 1
9 780131 465114