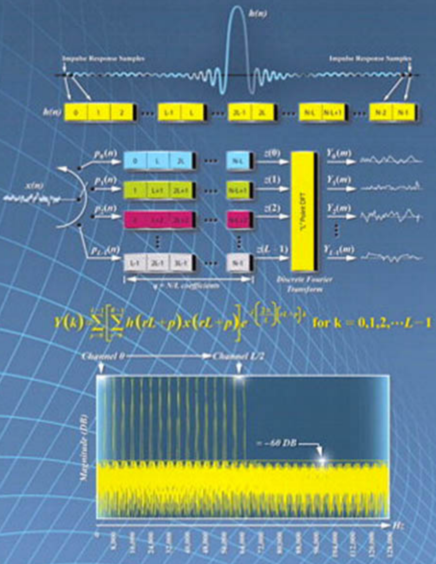
PRENTICE HALL

Practical Applications in Digital Signal Processing



RICHARD NEWBOLD

FREE SAMPLE CHAPTER

SHARE WITH OTHERS

Practical Applications in Digital Signal Processing

This page intentionally left blank

Practical Applications in Digital Signal Processing

Richard Newbold

PRENTICE HALL

Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The author and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U.S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the United States please contact:

International Sales
international@pearson.com

Visit us on the Web: informit.com/ph

Library of Congress Cataloging-in-Publication Data
Newbold, Richard.
Practical applications in digital signal processing / Richard Newbold.
pages cm
Includes bibliographical references and index.
ISBN-13: 978-0-13-303838-5 (hardcover : alk. paper)
ISBN-10: 0-13-303838-6 (hardcover : alk. paper)
1. Signal processing—Digital techniques. 2. Electric filters, Digital. I. Title.
TK5102.9.N49 2013
621.382'2—dc23 2012024511

Copyright © 2013 Pearson Education, Inc.

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission to use material from this work, please submit a written request to Pearson Education, Inc., Permissions Department, One Lake Street, Upper Saddle River, New Jersey 07458, or you may fax your request to (201) 236-3290.

ISBN-13: 978-0-13-303838-5
ISBN-10: 0-13-303838-6

Text printed in the United States on recycled paper at Edwards Brothers Malloy in Ann Arbor, Michigan.
First printing, October 2012

Executive Editor
Bernard Goodwin

Managing Editor
John Fuller

Project Manager
Caroline Senay

Project Editor
Scribe Inc.

Copy Editor
Scribe Inc.

Indexer
Scribe Inc.

Proofreader
Scribe Inc.

Publishing Coordinator
Michelle Housley

Cover Designer
Anne Jones

Cover Art
Laura Robbins

Compositor
Scribe Inc.

To my wife, Mary, who has always stood by me through thick and thin. To my son Shannon, a brilliant software design engineer, and to my son Daniel, an accomplished attorney, both of whom were self-sufficient right out of college.

This page intentionally left blank

Contents

PREFACE xiii
ACKNOWLEDGMENTS xxi
ABOUT THE AUTHOR xxiii

1 REVIEW OF DIGITAL FREQUENCY 1

1.1 Definitions 2
1.2 Defining Digital Frequencies 2
1.3 Mathematical Representation of Digital Frequencies 9
1.4 Normalized Frequency 12
1.5 Representation of Digital Frequencies 13

2 REVIEW OF COMPLEX VARIABLES 15

2.1 Cartesian Form of Complex Numbers 17
2.2 Polar Form of Complex Numbers 21
2.3 Roots of Complex Numbers 27
2.4 Absolute Value of Complex Numbers 35
2.5 Exponential Form of Complex Numbers 36
2.6 Graphs of the Complex Variable z 38
2.7 Limits 40
2.8 Analytic Functions 41
2.9 Singularity 42
2.10 Entire Functions 42
2.11 The Complex Number ∞ 42
2.12 Complex Differentiation 43
2.13 Cauchy-Riemann Equations 47
2.14 Simply Connected Region 51

2.15 Contours 51
2.16 Line Integrals 52
2.17 Real Line Integrals 54
2.18 Complex Line Integrals 84
2.19 Cauchy's Theorem 96
2.20 Table of Common Integrals 109
2.21 Cauchy's Integral 109
2.22 Residue Theory 120
2.23 References 127

3 REVIEW OF THE FOURIER TRANSFORM 129

3.1 A Brief Review of the Fourier Series 129
3.2 A Brief Review of the Fourier Transform 157
3.3 Review of the Discrete Fourier Transform (DFT) 187
3.4 DFT Processing Gain 254
3.5 Example DFT Signal Processing Application 261
3.6 Discrete Time Fourier Transform (DTFT) 263
3.7 Fast Fourier Transform (FFT) 267
3.8 References 268

4 REVIEW OF THE Z-TRANSFORM 271

4.1 Complex Number Representation 271
4.2 Mechanics of the Z-Transform 274
4.3 Left-Sided Z-Transform 277
4.4 Right-Sided Z-Transform 278
4.5 Two-Sided Z-Transform 278
4.6 Convergence of the Z-Transform 279
4.7 System Stability 290
4.8 Properties of the Z-Transform 292
4.9 Common Z-Transform Pairs 304
4.10 Inverse Z-Transform 308
4.11 Pole and Zero Standard Form Plug-In Equations 334
4.12 Applications of the Z-Transform 350
4.13 Summary of Useful Equations 380
4.14 References 381

5 FINITE IMPULSE RESPONSE DIGITAL FILTERING 383

5.1 Review of Digital FIR Filters 384
5.2 Parks-McClellan Method of FIR Filter Design 392
5.3 PM Implementation of Half Band Filters 425
5.4 References 433

6 MULTIRATE FINITE IMPULSE RESPONSE FILTER DESIGN 435

6.1 Poly Phase Filter (PPF) 436
6.2 Half Band Filter 465
6.3 Cascaded Integrator Comb (CIC) Filter 470
6.4 References 531

7 COMPLEX TO REAL CONVERSION 533

7.1 A Typical Digital Signal Processing (DSP) System 534
7.2 Conversion of a Complex Signal to a Real Signal 540
7.3 Complex to Real Simulation Results 560
7.4 Reference 573

8 DIGITAL FREQUENCY SYNTHESIS 575

8.1 Numerically Controlled Oscillator (NCO) 575
8.2 Enhanced NCO Phase Accumulator 608
8.3 NCO Synthesized Output Frequency Error 613
8.4 Adding a Programmable Phase Offset to the NCO Output 622
8.5 Design of an Industry-Grade NCO 628
8.6 NCO Phase Dither 641
8.7 References 644

9 SIGNAL TUNING 645

9.1 Continuous Time (Analog) Fourier Transform 647
9.2 Discrete Time (Digital) Fourier Transform 689
9.3 Useful Equations 754
9.4 References 759

10 ELASTIC STORE MEMORY 761

10.1 Example Application of an Elastic Store Memory 762
10.2 PCM Multiplexing Hierarchy 763
10.3 DS-1C Multiplexer Design Overview 768
10.4 Design of the Elastic Store Memory 774
10.5 Hardware Implementation of the Elastic Store Memory 792
10.6 Overall DS-1C Multiplexer Design Block Diagram 801
10.7 Additional Information 803
10.8 References 805

11 DIGITAL DATA LOCKED LOOPS 807

11.1 Digital Data Locked Design 808
11.2 Digital Data Locked Steady State Behavior 829
11.3 Digital Data Locked Transient Behavior 834
11.4 Data Locked Loop Bit-Level Simulation 845
11.5 Engineering Note 869
11.6 Summary of Useful Equations 869
11.7 References 871

12 CHANNELIZED FILTER BANK 873

12.1 Introductory Description 873
12.2 Channelizer Functional Overview 877
12.3 Channelizer Detailed Design Concepts 919
12.4 Channelizer Software Simulation Results 962
12.5 Channelizer Hardware Design Example 967
12.6 Summary of Useful Equations 974
12.7 References 975

13 DIGITAL AUTOMATIC GAIN CONTROL 977

13.1 Design of a Type I RMS AGC Circuit 981
13.2 Design of a Type II RMS AGC Circuit 1044
13.3 References 1047

A MIXED LANGUAGE C/C++ FORTRAN PROGRAMMING 1049

A.1 Writing a C/C++ Main Program 1051
A.2 Calling Subroutines and Functions from a C/C++ Main 1051
A.3 Writing a FORTRAN Subroutine 1054
A.4 Writing a FORTRAN Function 1055
A.5 Passing Integer Arguments 1055
A.6 Passing Floating Point Arguments 1057
A.7 Passing Array Arguments 1059
A.8 Passing Pointer Arguments 1060
A.9 Compile/Link Mixed Language C/C++ FORTRAN Programs 1063
A.10 Parks-McClellan FORTRAN Subroutine Called from C Main 1064
A.11 References 1091

INDEX 1093

This page intentionally left blank

Preface

I have spent more than 30 years toiling away as a digital hardware design engineer and as an unsophisticated self-taught software designer. Most of my software efforts were in support of my hardware designs and included endeavors such as bit-level simulations, microcode generation, assembly code, FORTRAN, C/C++, and writing Microsoft Windows application graphics-oriented test stations, which I utilized to verify the proper operation of my digital creations.

I began my digital design career when digital signal processing (DSP) was still in its infancy. In those days, all digital designs were implemented with small-scale integrated (SSI) circuits that weren't much more sophisticated than 4-bit adders and 8- to 1-bit multiplexers. The first company I worked for after graduation was heavily into the early phases of DSP.

DSP algorithms are for the most part dependent on repetitive multiplications and summation operations. The first digital multiplier I ever saw required an entire chassis of equipment to do a 16-by-16 multiplication. This multiplier consumed so much hardware that it was efficient to time-share it with other hardware that was engaged in processing independent tasks. Device propagation delays were so huge that building hardware systems that utilized a 5-MHz system clock was considered high tech.

To give some perspective about the state of the art at the time, the term *Silicon Valley* had not been coined yet. It was during this time that a little-known, small company that went by the name of Intel was operating out of a very tiny building located at 365 Middlefield Road in Mountain View, California. Intel had just introduced the world's first microprocessor. It was a 4-bit machine called the 4004 microcomputer. It was built under contract to the Nippon Calculating Machine Corporation in Tokyo, Japan. With the introduction of the 4004, the digital age changed gears. Digital technology soon began to evolve so quickly that hardware designed one year was almost obsolete by the next.

Program requirements always seemed to demand technology that wasn't developed yet. Design engineers were constantly tasked with implementing tomorrow's designs with today's technology. This struggle, in a large sense, fueled an atmosphere of intense research and development and drove the industry to continuously produce lower power, faster, and more complex devices and systems. Looking back, it seems like the world of DSP just exploded on all fronts. Start-up companies sprouted up in the Silicon Valley almost on a daily basis.

During this time, the science and technology of DSP grew and matured as integrated circuit manufacturers strived to produce higher speed signal processing components and lower power processors. Fusible link programmable logic devices were introduced, which quickly evolved into reprogrammable logic devices and, over time, evolved into field programmable gate arrays (FPGAs), complex programmable logic devices (CPLDs), and application-specific integrated circuits (ASICs), which are still in use today. Other companies began to prosper by serving as fabrication houses for extremely high-speed gallium arsenide and indium phosphide integrated circuits. They would teach engineers how to design using their processes and then fabricate their application-specific designs.

The design tools necessary to support the programming and testing of these complex devices have evolved into big-time software applications. FPGA companies are even taking most of the challenges out of DSP design by offering a library of DSP circuits called *cores* that can be incorporated into an FPGA design with a simple keystroke, without much knowledge on the designer's part of how these circuits operate.

During my 30-year career I have accumulated a fairly large library of DSP textbooks. With few exceptions, these books all cover the same basic topics. Different authors address the same subjects but each with their own unique approach. Reading several authors' treatment of the same subject helped me view DSP processing techniques from different perspectives and tended to fill a lot of the blanks in my understanding of the subject. These books were well written by astute people in the field, and they all provided an excellent technical baseline for DSP design.

However, there have been few textbooks written that deal specifically with the many DSP topics and algorithms that are commonly used in everyday applied DSP. As a rule, a good working knowledge of these applied DSP algorithms usually comes from word of mouth, design mentoring, and design experience. Over time, all design engineers accumulate (in their minds) a toolbox of circuits, procedures, algorithms, and techniques that are a product of years of long hours, a lot of sweat, tears, successes, failures, hand-wringing, and a fair amount of banging one's head against the wall. Unfortunately these toolboxes are not documented, and thus it is hard for other engineers to access the wealth of information contained within these toolboxes. Engineers for the most part are a secretive species and in their quest for job security are reluctant to publicize their hard-earned trade secrets.

There are many gray areas in DSP design that have not been addressed in detail by any of the engineering textbooks that I am familiar with. These gray areas usually don't address questions like *How do I design a circuit that will perform this or that critical DSP function?*

For example, no DSP textbook I am familiar with has discussed in detail applications that are heavy into the use of complex digital signals, the spectra of real and complex digital signals, the science of complex to real signal conversion, digital signal translation, or the concept of digital frequency synthesis.

I have not seen any text that provided a detailed analysis on how to design a numerically controlled oscillator (NCO) used in digital tuning applications, or how to design an elastic store memory used in pulse code modulation (PCM) multiplexing applications, or how to design a digital data locked loop (DLL) or a digital automatic gain control (dAGC).

Other design topics rarely discussed in application-oriented detail by the myriad of DSP books available today include applications of poly phase filters (PPF) and cascaded integrator comb (CIC) filters, and applications like digital channelizers, sometimes referred to as *transmultiplexers*. This versatile circuit is found in many applications, such as frequency division multiplex (FDM) to time division multiplex (TDM) conversion, mixing consoles, wideband scanners, and the processing of wideband intercepts in radio astronomy, to name just a few. All these subjects and more can be lumped into the general topic of *Practical Applications in Digital Signal Processing*.

THE PURPOSE OF THIS BOOK

The purpose of this book is to unlock and dispense some of the contents of my own personal toolbox in the hope of filling in some of these DSP gray areas. It is my hope to provide a source of usable information and DSP design techniques suitable for use in real-world design applications.

There are a great many DSP textbooks that are considered bibles of the DSP design world. Many of these books, along with technical papers written by astute people in the field, are referenced within this book. It is not the intention of this book to repeat the work that has been done by so many previous authors. This book does not deal with the derivation and treatment of standard DSP concepts, which have been thoroughly addressed in great detail by many other authors. The sole purpose of this book is to serve as an application-oriented addendum to the many great DSP textbooks that have already been published.

WHO SHOULD READ THIS BOOK

This book is not intended for a person with no previous DSP knowledge or experience. This book is intended for the undergraduate and graduate student who will soon enter the signal processing industry. It is also intended for the engineer already in the industry who has some experience in DSP design and who is now searching for additional information regarding the design and implementation of common but largely undocumented DSP hardware or software applications.

HOW THIS BOOK IS ORGANIZED

This book is organized as a collection of tutorials on common DSP applications. The first four chapters are detailed reviews on the mathematical tools necessary to successfully analyze, design, and build complex digital processing systems. The remaining nine chapters provide detailed tutorials on independent signal processing applications commonly used in the industry. An appendix is included that provides an in-depth discussion on mixed language programming. The content of each chapter is summarized in the following sections.

Chapter 1: Review of Digital Frequency

This chapter is a short tutorial on digital frequency and how it is related to the system sample rate. It shows how to mathematically represent the value of a particular digital frequency and how to determine the value of all the samples in a digital sinusoidal waveform.

Chapter 2: Review of Complex Variables

This chapter presents a thorough review of the subject of complex variables. After reading this chapter, it is possible for a person with no prior experience to become proficient in the use of this valuable mathematical tool in the design and development of signal processing circuits and systems. The review starts by defining complex numbers and their properties and progresses all the way to a complete discussion of residue theory. The computation of residues provides the engineer an easy alternative to compute the impulse response of a digital system.

Chapter 3: Review of the Fourier Transform

This chapter provides an in-depth review of the Fourier series and both the continuous and discrete Fourier transform (CFT and DFT, respectively). The discussion includes the derivation of transform properties, transform pairs, Parseval's theorem, and the derivation of energy and power spectral density (PSD) relationships. Attention is also given to the topic of spectral leakage, the band pass filter, and the low pass filter models of the DFT. Signal processing discussions include the use of windows, coherent and incoherent processing gain, and signal recognition. Even though this is an extensive review, it is written so that a reader without any background in the topics of Fourier series or Fourier transforms can proficiently use them when working with signal processing applications.

Chapter 4: Review of the Z-Transform

This chapter provides a comprehensive review of the z-transform. Detailed discussions include the use of pole-zero diagrams, inverse z-transforms, convergence, and system stability. A person with no prior knowledge of z-transforms can, after reading this chapter, utilize the knowledge gained to analyze complex digital systems, thereby enabling them to derive a system frequency response, determine system stability, and compute a system impulse response. In addition, the reader will learn how to use the z-transform in real-world situations to modify existing designs to either enhance performance or alter the specifications for incorporation into other systems.

Chapter 5: Finite Impulse Response Digital Filtering

The focus of this chapter is on the design of finite impulse response (FIR) digital filters. It is not my intent to repeat all of the excellent theoretical material that has already been published by so many astute authors. Almost all DSP texts devote substantial coverage to the history, theory, architecture, mathematics, and legacy design techniques of digital filters. Instead, the intent here is to concentrate solely on a single method for the design and implementation of some of the more common filter types. The purpose of this chapter is twofold. First, in order to establish a communication baseline, we will provide a very brief overview of digital filters. Second, we will demonstrate a computer-aided design methodology based on the Parks-McClellan optimal filter design program to implement several types of digital filters. A complete listing of this program is included in Appendix A.

Chapter 6: Multirate Finite Impulse Response Filter Design

This chapter is a detailed discussion on the design of digital filters used to modify the sample rate of a signal. A designer is often faced with the task of either increasing or decreasing the sample rate of a signal by some integer or fractional amount. There are several methods that can be utilized to change the sample rate of a digital signal. All these methods involve the use of a digital filter, sometimes referred to as a *multirate filter*. Some multirate filters are better suited for specific rate change applications than others. In this chapter we will discuss three rate change methods that use the following three filter types:

1. *Poly phase filters.* The preferred method for moderate sized rate changes.
2. *Half band filters.* An efficient method for factor of two rate changes.
3. *CIC filters.* Computationally efficient filters for large rate changes.

Chapter 7: Complex to Real Conversion

This chapter provides a detailed tutorial on the conversion of a complex signal to a real signal. This is a common signal processing function, yet material dealing with this very important topic is rarely found in engineering textbooks. A very good example of complex signal processing is seen in digital systems that employ a front-end tuner. These systems fall into a category that can be loosely categorized as “digital radio,” in that an input wideband signal is tuned up or down in frequency and passed through a band pass or low pass filter to isolate some narrow band of interest. The mathematics of the tuning function converts the real input signal into a complex signal. The filtered narrow band signal is then processed in its complex form to implement whatever the particular application requires. After the intermediate processing is complete, the complex signal is generally converted back to real and provided as an output.

Chapter 8: Digital Frequency Synthesis

There are numerous applications in the world of DSP that utilize a numerically controlled oscillator, or NCO. An NCO is a programmable oscillator that outputs a digital sinusoid at some user-specified frequency and phase. The sinusoid can be fixed at some programmed frequency, or it can be swept or hopped over a band of frequencies. The sinusoid can have a constant phase or it can be programmed to have multiple or switched phases. It can be a simple or a complex device, depending on the requirements of the application in which the NCO is used. A typical application utilizes the NCO to produce a programmable complex sinusoid to tune band pass signals down to base band for filtering and postprocessing, similar to the local oscillator in an AM radio. This chapter contains detailed figures that clearly illustrate both the design of the NCO and the workings of all the internal processing functions. Extensive simulations graphically illustrate the signals produced by the NCO.

Chapter 9: Signal Tuning

This chapter provides a thorough discussion on the subject of signal tuning in both the continuous analog and discrete digital domains. It is often necessary when processing a signal to move it from one region of the frequency spectrum to another region. This is especially true when processing communications signals, where a band limited signal centered at frequency f_1 is tuned to another center frequency f_2 in order to simplify downstream processing. This chapter illustrates the methods used to translate the spectrum of real and complex signals both up and down in frequency.

Chapter 10: Elastic Store Memory

During their careers, most engineers have designed interfaces between two or more data processing systems that utilized synchronous data streams. There are occasions, however, when a designer must interface two or more processing systems or data streams where the data rates are asynchronous to one another. For purposes of this chapter, the term *asynchronous* refers to the case where each data stream is time aligned to its own clock generated by an independent clock oscillator. The frequency and phase of each clocked data stream are similar but not necessarily identical. Each clock oscillator's output frequency uniquely varies over time and temperature. In many cases, these clocks may differ by as much as a few thousand hertz. In this chapter we illustrate how to synchronize these systems with an elastic store memory.

Chapter 11: Digital Data Locked Loops

Suppose you are presented with a time division multiplex, or TDM, bit stream composed of a multiplex of two or more independent and originally asynchronous tributaries. How can we demultiplex these tributaries and synthesize an independent bit clock for each that is on average identical to its original premultiplex clock? This type of signal is similar to a high-level telephone PCM multiplex that carries several lower level tributaries. This is only one of many possible examples. The same question can be asked of any demultiplex processing where the multiplexed tributaries were originally asynchronous to one another. The answer requires utilizing a digital data locked loop, or DLL. The DLL is a fairly simple device that uses an elastic store memory to synthesize a bit stream clock and then synchronizes the demultiplexed bit stream or tributary with that clock, all with no prior knowledge of the original clock frequency. This chapter provides a thorough tutorial on how to design DLLs for just about any relevant application.

Chapter 12: Channelized Filter Bank

This chapter presents a high-level functional discussion followed by an in-depth, detailed tutorial on the design of a digital channelizer, sometimes referred to as a *transmultiplexer*. As mentioned previously, this versatile circuit is found in many signal processing applications. The channelizer can easily replace hundreds of receivers with not much more than a single integrated circuit. In this chapter, we will design a working channelizer that simultaneously processes up to 2000 independent equal bandwidth signals.

Chapter 13: Digital Automatic Gain Control

This chapter is a thorough discussion of a Type I and Type II digital automatic gain control, or dAGC. This subject matter is rarely covered in any engineering textbook available today, and if it is covered, it is usually given a cursory look amounting to not much more than a paragraph or two. In many electronic systems, one of the most important functions is automatic gain control (AGC). In general, an AGC is a nonlinear feedback circuit that if not designed properly can become unstable. The purpose of this chapter is to design a dAGC circuit; derive its operational parameters; simulate it; and then graphically illustrate the transient response, the steady state operation of the loop error, the loop gain, and the circuit output in response to various input signals and input signal perturbations.

Appendix A: Mixed Language C/C++ FORTRAN Programming

Over the years, there is a good chance that engineers who have been in the business for a while have accumulated a few dusty, old FORTRAN programs, functions, or subroutines that represent some pretty valuable legacy code. If these coded routines weren't considered to be so valuable, the engineers more than likely would never have saved them. Typically, these routines represent a treasure chest of tested, debugged, and proven code that is still relevant in today's engineering environment. The one big problem is that most of the software today is developed in C or C++. If this is the predicament that you find yourself in, there is some good news and some bad news for you. The good news is there is a good chance that the program manager and design engineering staff has at their disposal a wealth of proven FORTRAN code. Incorporating this proven code into a project very well could result in a significant reduction in labor costs and a significant reduction in program schedule. The bad news, of course, is that C or C++ are today's preferred languages; therefore writing deliverable code in FORTRAN is really not a viable option. So if you are a program manager or a design engineer, what can you do in a situation such as this? One alternative is to build a mixed language program, where the bulk of the code including the main is written in C/C++ and linked with one or more valuable FORTRAN legacy functions and/or subroutines. This appendix is a tutorial on how to do just that.

Acknowledgments

I extend my sincere appreciation to Bernard Goodwin, executive editor of Pearson North America, Prentice Hall Professional technical publications, for his generous help and support during the difficult task of introducing a new author to the world of technical publishing. I would also like to acknowledge Dr. John Treichler of Applied Signal Technology for his gracious permission to reference his original work on transmultiplexers.

I would like to thank Richard Lyons, author of *Understanding Digital Signal Processing* (Prentice Hall); David Myers; Jim Kemerling; Michael Myers; and C. Britton Rorabaugh, author of *Notes on Digital Signal Processing* (Prentice Hall), for their technical review of the manuscript.

I would like to give a long overdue thank you to Mike Tate of Electromagnetic Systems Laboratories, the absolute best technician I have ever worked with, who helped me successfully begin my career so many years ago. I would like to thank Tom Ranweiler of Northrop Grumman, the most technically astute software design engineer I have ever worked with, who was the person who helped me finish my career on a successful note with the design of a unique special-purpose signal processing system. I am also thankful for the opportunity to have worked alongside the brilliant systems engineer Dr. Pin-Wei Chen of Northrop Grumman who helped keep me involved in internal research and development design projects.

Finally I would like to extend my deep appreciation to my wife Mary for her patience, support, understanding, and encouragement throughout this very long project.

This page intentionally left blank

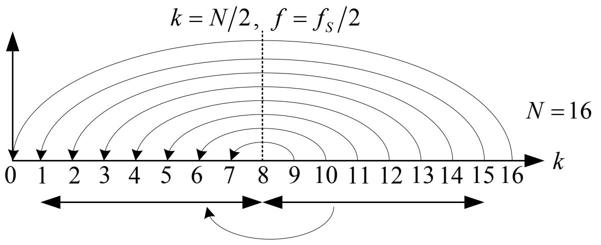
About the Author

Richard Newbold received his B.S.E.E. and M.S.E.E. degrees in 1974 and 1978, respectively, and has spent more than 30 years as a digital hardware design engineer and self-taught software designer. His design experience includes special-purpose signal processing hardware and computers that processed real time wideband signals, direct sequence spread spectrum system processors, PCM multirate processing systems, high-speed signal processing systems implemented on special-purpose gallium arsenide ASICs, transmultiplexers, channelizers, multirate filters, tuners, frequency synthesizers, DLLs, synchronous digital hierarchy (SDH) demultiplexers, fractional resamplers, adaptive filters, elastic store memories, adaptive beam forming, asynchronous clock recovery, and fault tolerant signal processors. His software experience includes real time signal processing, bit-level hardware simulations, microcode and bit slice programming, assembly programming, FORTRAN, C/C++, and Microsoft Windows graphics-oriented test stations, which were used to bit-level simulate, graphically display, and verify the proper operation of his digital creations.

This page intentionally left blank

CHAPTER ONE

Review of Digital Frequency



It is easy to mathematically represent an analog frequency on paper. The range of frequencies in the analog domain is both continuous and theoretically infinite. If we use the symbol f_O to represent some arbitrary analog frequency all we need to do is to equate it with any one of an infinite number of available frequencies. We could, for example, choose f_O to be equal to 23.456 Hz, or we could just as easily choose f_O to be equal to 1.005 MHz. We could choose just about any other value to any precision that we can dream up. As long as we remain realistic, there is no limit on the values that f_O can take on.

However, a digital system operates on digital data and generates digital results that are valid only at discrete increments of time equal to the period of the system sample clock. Therefore the value that a digitally generated discrete frequency can take on is a small subset of the range of values available to analog frequencies. The discrete frequency values within this subset are directly related to and dependent on the sample rate of the digital system clock.

This leads to some confusion when people deal with digital frequencies for the first time. Much of the confusion can be summed up with three frequently asked questions:

1. How do I define a digital frequency?
2. How do I mathematically represent a digital frequency?
3. How do I synthesize a digital frequency in hardware or software?

The scope of this chapter is to provide an answer for the first two of these questions. The answer to question number 3 requires its own chapter and is dealt with in detail in Chapter 8, "Digital Frequency Synthesis."

1.1 DEFINITIONS

In this chapter, we will make the following symbol definitions:

1. f defines any arbitrary analog frequency in hertz.
2. f_O defines a specific analog frequency in hertz.
3. f_K defines a specific digital frequency in hertz.
4. ω_O defines a specific analog radian frequency in radians/second.
5. ω_K defines a specific digital radian frequency in radians/second.
6. f_S defines the sample rate or the frequency of a digital system clock.
7. T defines the period of the digital sample clock $T = 1/f_S$.

1.2 DEFINING DIGITAL FREQUENCIES

Unlike an analog frequency, a digitally generated frequency does not have infinite resolution. A digital frequency can only take on discrete values. A digital sine wave, for example, can only take on discrete values for frequency, phase, and amplitude. For the purposes of this chapter, the frequency resolution of a digitally generated sinusoid is limited by the period of the digital sample clock $T = 1/f_S$, and the precision of the sinusoidal waveform amplitude is limited by the bit width of each digital sample. Let us begin our discussion by considering a digital sinusoidal waveform.

We know that a sine wave has unity amplitude and is repetitive every 2π radians. As illustrated in Figure 1.1, we can draw a circle of radius 1, called the unit circle, and we can visualize a phasor of unity magnitude rotating around the unit circle at some fixed angular rate ω_K. Every time the phasor makes a complete revolution around the unit circle, it has passed through 2π radians and has completed one cycle. We quantify the phasor's rotational speed as being ω_K radians per second. To get started, let us assign the label C to this phasor. Since the phasor C takes on values only at discrete instances of the sample period T, we can represent it as a function of discrete time by writing $C(nT)$ $\{\text{for } n = 0, 1, 2, \cdots\}$.

We can use some simple trigonometry to represent the phasor $C(nT)$ by its vertical and horizontal components, labeled $\mathrm{A}(nT)$ and $B(nT)$ in Figure 1.1. The magnitude of C is related to the value of its components by the Pythagorean theorem: $C(nT) = \sqrt{A^2(nT) + B^2(nT)}$.

We can see that as the phasor C rotates around the unit circle, the magnitude of the A component cyclically grows from 0 at 0 radians to +1 at $\pi/2$ radians. It then attenuates back to 0 at π radians, grows to –1 at $3\pi/2$ radians, and finally attenuates back to a value of 0 as the phasor passes through

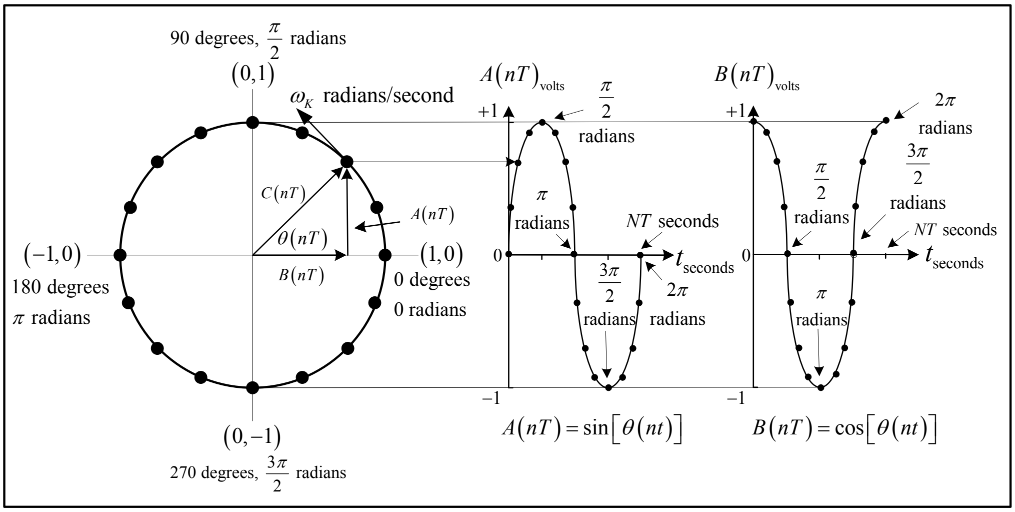


Figure 1.1 Unit circle

2π radians. Each time the phasor completes a rotation of 2π radians, the amplitude of the A component traces out a sine wave and the amplitude of the B component traces out a cosine wave, as illustrated in Figure 1.1.

Since we are dealing with a digital system, we know that the values of the phasor components *A*, *B*, and *C*, and the phasor phase angle θ, shown in Figure 1.1, take on values only at discrete instants of time equal to the period of the sample frequency, or $T = 1/f_S$. Sequential sampling instants can be represented by the infinite series

$$1T,\ 2T,\ 3T,\cdots,nT,\cdots$$

We can represent any arbitrary sampling instant as nT for $0 \le n < \infty$. Incorporating this notation into the unit circle phasor representation Figure 1.1, we see that the only values that can be represented by the phasor *C*, its vertical and horizontal components A and B, and the phase angle θ are at the instants of time equal to nT. We know from trigonometry that

$$\sin(\theta) = \frac{A}{C} = \frac{A}{\sqrt{A^2 + B^2}}$$

and

$$\cos(\theta) = \frac{B}{C} = \frac{B}{\sqrt{A^2 + B^2}}$$

We also know that since we are working on the unit circle, the magnitude of $C = \sqrt{A^2 + B^2} = 1$. Therefore we can state that at any sampling instant nT,

$$A(nT) = \sin\left[\theta(nT)\right]$$

and

$$B(nT) = \cos\left[\theta(nT)\right]$$

Equation 1.1

So far so good, but how do we quantify the discrete values taken on by the sinusoidal waveforms? Well, we can start by dividing the unit circle into N equal arc segments, illustrated by the black dots on the unit circle in Figure 1.1. Each arch segment is a portion of the unit circle scribed by the tip of the phasor C as the phase angle θ is incremented by $2\pi/N$ radians.

We will take this opportunity to coin a new and highly technical term. Let us define the angle $2\pi/N$ and the arch segment it describes as a *radian chunk*. The angle between the adjacent black dots on the unit circle in Figure 1.1 is equal to a radian chunk. The phasor $C(nT)$ and its components $A(nT)$ and $B(nT)$ can only be evaluated at each black dot corresponding to each radian chunk on the unit circle. Therefore the maximum number of samples that can represent a single cycle of a digital sine wave is equal to N.

If the digital oscillator that is generating the digital sinusoid is operating with a sample clock of f_S Hz, then the phasor $C(nT)$ would rotate around the unit circle in discrete radian chunks of $2\pi/N$ at the clock rate of f_S Hz. The lowest radian frequency of the digital oscillator can be mathematically defined as

$$\omega_K = \left(\frac{2\pi \text{ radians}}{N \text{ sample}}\right)\left(f_S \ \frac{\text{sample}}{\text{second}}\right) = \frac{2\pi}{N} f_S \text{ radians/second}$$

$$\omega_K = \frac{2\pi}{N} f_S \text{ radians/second}$$

Equation 1.2

The lowest or fundamental frequency of the digital oscillator can be mathematically defined as

$$f_K = \left(\frac{1}{2\pi \text{ radians}}\right)\left(\frac{2\pi \text{ radians}}{N \text{ sample}}\right)\left(f_S \ \frac{\text{sample}}{\text{second}}\right) = \frac{1}{N} f_S \text{ second}^{-1} = \frac{1}{N} f_S \text{ Hz}$$

$$f_K = \frac{1}{N} f_S \text{ Hz}$$

Equation 1.3

If we think about it for a minute, Equation 1.2 and Equation 1.3 make sense. If the phasor points to each black dot on the unit circle for one sample period, it will take N sample periods for it to move from dot zero to dot $N-1$. In doing so, it will make one revolution of the unit circle stopping once at each black dot in N sample periods. It will take N sample periods to trace out exactly one sinusoidal cycle. The total period for each cycle will be equal to NT sec. The frequency of this sinusoidal cycle is then given by $f = 1/(NT \text{ second}) = (f_S/N)\text{Hz}$. Therefore the frequency of the sinusoids traced by the $A(nT)$ and $B(nT)$ components will be equal to $(f_S/N)\text{Hz}$.

Let us look at a very simple example. Suppose we have a digital oscillator clocked with a sample clock of $f_S = 32$ Hz, and suppose we decided that N will be 16. The unit circle is subdivided into 16 equal arc lengths, giving us 16 equal radian chunks. The rotating phasor $C(nT)$ will be evaluated at 16 locations around the unit circle. This means there will be 16 samples per each period of the synthesized sinusoidal waveform. The digital radian frequency would be

$$\omega_K = \left(\frac{2\pi \text{ radians}}{16}\right)\left(\frac{32}{\text{sec}}\right) = 4\pi \text{ radians/second}$$

or, since $f_K = \omega_K/2\pi$, we can easily compute the digital oscillator frequency to be

$$f_K = \frac{\omega_K}{2\pi} = \left(\frac{4\pi \text{ radians/second}}{2\pi \text{ radians}}\right) = 2 \text{ second}^{-1} = 2 \text{ Hz}$$

In this simple example, 2 Hz is the lowest frequency other than zero that our simple digital oscillator can generate. This is based on the value of the sample frequency f_S and our choice for the value of N. If, for the same sample rate, we had chosen N to be a larger number, then the resolution of f_K would have been greater. For example, if we had selected $N = 64$, the lowest frequency other than 0 Hz that our oscillator could produce would be

$$f_K = \frac{1}{N} f_S \text{ Hz} = \frac{1}{64} 32 \text{ Hz} = 0.5 \text{ Hz}$$

This is a good start, but a digital oscillator that can produce only a single frequency isn't as useful as an oscillator that can be programmed to produce any one of a whole range of discrete frequencies. Ideally, we would like to be able to program the digital oscillator to output any one of a wide range of discrete frequencies. We can achieve this enhancement with the addition of a multiplier "k" in Equation 1.2 and Equation 1.3. We can rewrite these equations to include the multiplier k such that

$$\left.\begin{aligned} \omega_K &= \frac{2\pi k}{N} f_S \text{ radians/second} \\ f_K &= \frac{k}{N} f_S \text{ Hz} \end{aligned}\right. \qquad \left\{\text{where } k = 0,1,2,\ldots N/2\right\}$$

Equation 1.4

If $k = 1$, then the frequencies represented by Equation 1.4 are identical to those represented by Equation 1.2 and Equation 1.3. The value of k can take on discrete integer values ranging from 0 to $N/2$. In our previous example, we set $f_S = 32$ Hz, and $N = 16$ so k could take on values of $0,1,2,3,4,5,6,7,8$. All the possible frequency values that this example oscillator can take on are tabulated in Table 1.1.

As we can see from the table, this oscillator can be programmed to produce one of nine possible frequencies with a resolution of 2 Hz. The addition of the variable k in Equation 1.4 causes the phasor $C(nT)$ to rotate around the unit circle in multiples of $2\pi/N$ radian chunks at a rate of f_S sample per second. When k is set to unity, the phasor will take on values at every black dot on the unit circle producing the oscillator's lowest or fundamental frequency. In this case, each cycle of the generated sinusoid will be composed of N samples.

When k is set to 2, the phase angle of the phasor $\theta(nT)$ will increase in increments of two radian chunks each tick of the sample clock. The phasor $C(nT)$ will take on the values of every second dot, and it will rotate around the unit circle twice as fast, producing a sine or cosine wave that is twice the fundamental frequency. In this case, each cycle of the sinusoid will be composed of half or $N/2$ samples. Similarly, when k is set to 4 the phasor will travel around the unit circle at four times the fundamental rate, taking on values at every fourth dot to produce an output frequency that is four times the fundamental frequency. Each cycle, however, will be composed of $N/4$ number of samples per cycle.

Table 1.1 Example Digital Oscillator Frequencies

k	$\omega_k \dfrac{\text{radians}}{\text{second}}$	f_K Hz	Samples per cycle
0	0.00	0	—
1	$\pi/8$	2	16
2	$2\pi/8$	4	8
3	$3\pi/8$	6	16/3
4	$4\pi/8$	8	4
5	$5\pi/8$	10	16/5
6	$6\pi/8$	12	16/6
7	$7\pi/8$	14	16/7
8	$8\pi/8$	16	2

When k reaches its maximum value of $N/2$, which in this example is 8, there will be just two samples per sinusoidal period. The Nyquist rule states that two samples per cycle is the minimum number of samples allowed in order to be able to reconstruct an analog waveform from a digital waveform. This means that the highest frequency we can theoretically generate with a digital oscillator is half the sample rate or $f_S/2$. In our example, when $k = N/2 = 8$, we were able to generate a sinusoid of 16 Hz, which is exactly half the 32 Hz sample rate.

In Table 1.1, a few of the entries for the "Samples per cycle" column are in fractions. All this means is that each cycle of the sinusoid is generated using a different subset of the N possible samples. That is, successive cycles of the sinusoidal waveform begin on a different sample value.

What happens if we continue to increase the value of k beyond $N/2$ (which in our example is 8)? If we were to let $k = 9$, there would be less than two samples per cycle, the Nyquist rule would be violated, and the output waveform would take on the same frequency as if the value of k had been set to 7. The resulting frequency is said to have been aliased or folded, about $f_S/2$. Undersampling an analog sinusoid with an analog to digital converter will cause the digital output sinusoid it to alias down in frequency. This is identical to setting the multiplication factor k of a digital frequency to some number greater than $N/2$, which will result in the folding or aliasing about the Nyquist frequency of $f_S/2$.

In our example, if we were to say $k = 10$, the resulting digital frequency would be identical to that obtained by saying $k = 6$. If we choose $k = N - 1$

or 15 in our example, the resulting frequency would be identical to the case where $k = 1$. We can see that all frequencies above $k = 8$ are folded or aliased down in frequency about the so-called folding frequency of $f_S/2 = 16$ Hz. In general, the frequency f_{N-K} aliases down to $f_K \{\text{for } (N/2) < k \le (N-1)\}$.

A simple frequency folding diagram for the case where $N = 16$ is illustrated in Figure 1.2.

Care must be taken at the higher frequencies where the value of k approaches the upper limit of $N/2$. On paper there is no problem as $k \to N/2$, but in a hardware or software implementation the samples must be carefully selected. An extreme example would be setting $k = N/2$ and using the samples at 0 and π radians. This represents the optimum sample selection for a cosine wave, since the cosine sequence will take on the form

$$B(T) = \cos(n\pi) = +1, -1, +1, -1, \cdots \quad \{\text{for } n = 0, 1, 2, 3, \cdots\}$$

but these same samples will produce a DC output of 0 for a sine waveform

$$A(T) = \sin(n\pi) = 0, 0, 0, 0, \cdots \quad \{\text{for } n = 0, 1, 2, 3, \cdots\}$$

For this reason, in most designs dealing with narrow band signals, the minimum number of samples per cycle is usually held to some number around 2.5.

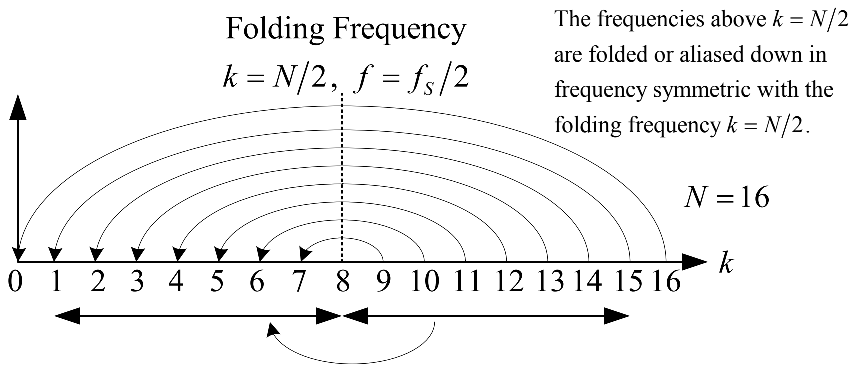

Folding Frequency
$k = N/2$, $f = f_S/2$
The frequencies above $k = N/2$ are folded or aliased down in frequency symmetric with the folding frequency $k = N/2$.
$N = 16$
k
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Figure 1.2 Frequency folding diagram

1.3 MATHEMATICAL REPRESENTATION OF DIGITAL FREQUENCIES

The notation in Equation 1.4 gives us a valid method to represent digital frequencies using a pencil and paper, but it doesn't help us when it comes to the generation of a digital sinusoidal waveform in hardware or software. In Equation 1.4, we explicitly included the sample rate represented by the term f_S. This is fine for mathematical computations on paper but the sample rate is a fixed entity that is already implicit in the operation of the digital hardware. It does not make sense to include the f_S term in frequency synthesis because it is already present due to the fact that each digital computation takes place at the sample rate.

In addition, Equation 1.4 provides us with a single value for any particular frequency—something that we can write down on paper like the value 16 Hz or 48 Hz. This is not appropriate for the actual synthesis of a complete period of a sinusoidal frequency in hardware. In hardware or software, we need to generate all N samples of a sinusoid at the sample rate f_S. To do that, we need to take into account the sample index n of the generated sinusoid, and we need to normalize the sample frequency to 1. We normalize the frequency by dividing the frequency expression by f_S. In doing so, we can rewrite Equation 1.4 to include these changes and produce

$$\begin{aligned} \omega_K(n) &= \frac{2\pi kn}{N} \text{ radians/second} \\ f_K(n) &= \frac{kn}{N} \text{ Hz} \end{aligned} \qquad \text{where} \left\{ \begin{aligned} k &= 0,1,2,\ldots N/2 \\ n &= 0,1,2,3,\ldots\infty \end{aligned} \right\}$$

Equation 1.5

In Equation 1.5 we have normalized the sample frequency to 1. The normalized frequency of a digital waveform is usually expressed as a fraction given by k/N. To convert a normalized frequency back to an unnormalized frequency, we simply multiply by the sample rate, or

$$f_K = \frac{k}{N} f_S$$

Remember the sample rate is implicit in a hardware or software implementation. It is the rate at which the computations are performed (i.e., the rate at which the phasor advances between black dots on the unit circle).

For example, if our sample frequency $f_S = 256$ KHz, $k = 16$, and $N = 128$, then the normalized frequency would be expressed as follows:

$f_K = k/N = 16/128 = 0.125$. The actual frequency in Hz can be determined by multiplying the fraction k/N by the actual sample frequency f_S or

$$f_K = \frac{k}{N} f_S = (0.125)(256 \text{ KHz}) = 32 \text{ KHz}$$

The function of the sample index n in Equation 1.5 is to index successive samples and to advance the phasor around the unit circle by acting as an incrementing multiplier to the fixed value k/N. The term nk/N can be thought of as an infinite series given by

$$\frac{k}{N}, \frac{2k}{N}, \frac{3k}{N}, \cdots, \frac{nk}{N}, \cdots$$

The correct way to generate a digital sinusoid waveform of unity magnitude and normalized frequency k/N is given by

$$\begin{aligned} A(n) &= \sin\left(\frac{2\pi k}{N} n\right) \\ B(n) &= \cos\left(\frac{2\pi k}{N} n\right) \end{aligned} \qquad \text{for} \left\{ \begin{array}{l} n = 0, 1, 2, \cdots \\ k = 0, 1, 2, \cdots, N/2 \end{array} \right.$$

Equation 1.6

It is important to remember that the term $\frac{2\pi}{N}$ is a radian chunk. The term $\frac{2\pi}{N} k$ is a k multiple of a radian chunk. Since the sample rate is implicit to the hardware and since we are dealing with normalized values of frequency, we can now drop the index notation nT as illustrated in Figure 1.1 and simply refer to the index as n as we did in Equation 1.5 and Equation 1.6. The phasor $C(n)$ will rotate around the unit circle at the sample clock rate in increments of $2\pi k/N$ radian chunks. This means that the phasor sine and cosine components $A(n)$ and $B(n)$ will take on discrete values at each kth radian chunk and will do so at the clock rate.

Nothing is stopping us from writing down on a sheet of paper that the radian frequency associated with this radian chunk is $(2\pi k/N) f_S$, but keep in mind that this is just a number on a sheet of paper; it is not the sine or cosine argument that will trace out a sinusoidal waveform. This is where the sample index n comes into play. The index increments by one every sample clock so the argument $\frac{2\pi}{N}nk$ takes on incremental k radian chunk values with each clock period. This means that $\omega_K = (2\pi nk/N)$ will take on successive values of $0, 2\pi k/N, 4\pi k/N, 6\pi k/N, \cdots$ at the sample rate. This is the dynamic argument of the sine and cosine function. If we plot $A(n)$ and $B(n)$ for this dynamic argument as $n = 0,1,2,3,\cdots$, we will trace out a sine and a cosine waveform at the radian frequency $(2\pi k/N) f_S$.

For example, suppose we let $k = 1$, $N = 8$, and $f_S = 32$ KHz. The sine argument would be implemented as

$$\frac{2\pi kn}{N} = \frac{2\pi n}{8} \quad \text{for } n = 0,1,2,3,\cdots$$

The sine waveform for the first eight values of the sample index n is illustrated in Figure 1.3. The fundamental frequency of this sinusoid is computed by

$$\begin{aligned}
\omega_K = \omega_1 &= \left(\frac{2\pi k}{N}\right) f_S &= \left(\frac{2\pi}{8}\right) 32 \text{ KHz} &= 8\pi \text{ K radians/second} \\
f_K = f_1 &= \left(\frac{k}{N}\right) f_S &= \left(\frac{1}{8}\right) 32 \text{ KHz} &= 4 \text{ KHz}
\end{aligned}$$

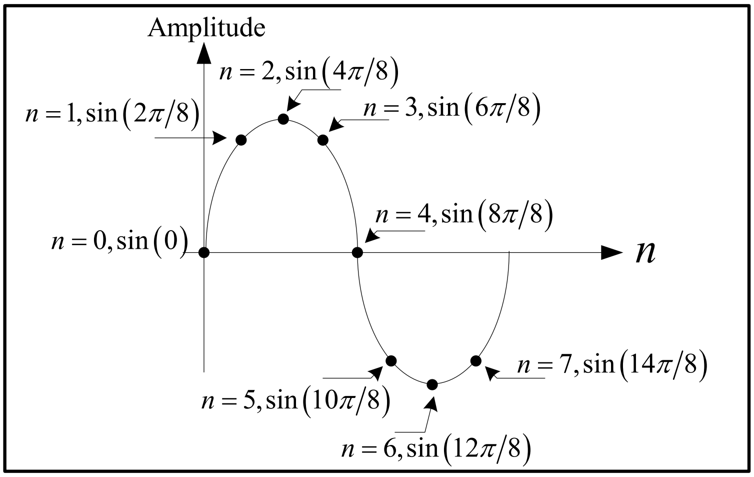


Figure 1.3 Mapped sine wave using a dynamic argument

The fundamental frequency is generated when $k=1$. Multiples of the fundamental frequency are referred to as harmonic frequencies. Frequencies generated for $k=2,3,4,\cdots,N/2$ are harmonics of the fundamental. When the multiplier $k=0$, we end up with a frequency of 0 Hz, or DC.

We can represent the phase offset in a digital sinusoid by the expression

$$\sin\left(\frac{2\pi kn}{N}+\Phi\right) \text{ where } \Phi=\frac{2\pi p}{N} \quad 0\le p\le N-1$$

The phase offset can only take on values equal to p radian chunks.

When implementing a digital sinusoidal generator in hardware, it is of paramount importance to remember the following seven points:

1. Since Equation 1.6 is computed every T seconds, it implicitly includes the sample rate term f_S. This is because the index n increments at the sample rate.
2. In normalized notation, the frequency of the sinusoid in Equation 1.6 is given by the ratio of k/N, where $0\le k/N\le 0.5$.
3. The frequency resolution of the digital sinusoid waveforms $A(n)$ and $B(n)$ is determined by the size of N.
4. The phase resolution of the digital sinusoid waveforms $A(n)$ and $B(n)$ is determined by the size of N.
5. The precision of the amplitude of the sinusoidal waveform is determined by the bit width of the samples used to represent both $A(n)$ and $B(n)$.
6. The base or fundamental frequency of the sinusoids in Equation 1.6 is equal to $1/N$ normalized and f_S/N unnormalized.
7. The set of discrete frequencies that can be output by a programmable oscillator is given by k/N for $k=0,1,2,\ldots,N/2$, which equates to a normalized frequency range of $0 \text{ to } 0.5$ in steps of $1/N$, or an actual frequency range of $0 \text{ Hz to } f_S/2 \text{ Hz}$ in steps of f_S/N.

We will show in Chapter 8, "Digital Frequency Synthesis," that the value of k in item 7 can take on fractional values, which allows the engineer to design synthesizers that have much finer frequency resolution.

1.4 NORMALIZED FREQUENCY

The notation in Equation 1.4 is based on the concept of dividing the unit circle into N equal arc segments. We can derive an equivalent expression simply by observing that the digital frequency

$$f_K = \frac{k}{N} f_S \quad \text{where } 0 \le k \le \frac{N}{2}$$

can be normalized by dividing both sides of the equation by the sample frequency f_S to arrive at

$$\frac{f_K}{f_S} = \frac{k}{N} \quad \text{where } 0 \le f_K \le \frac{f_S}{2} \quad \text{and} \quad 0 \le k \le \frac{N}{2}$$

Equation 1.7

Both sides of Equation 1.7 are now expressed as a fraction between 0 and 0.5. We can drop the subscript K such that f_K becomes f. When we do we can think of both f and f_S as being two analog frequencies whose ratio just happens to be k/N. This notation is useful if the unit circle is not specifically considered in the derivation of digital frequencies. The two methods of notation are equivalent, as illustrated in Equation 1.8.

$$\cos\left(\frac{2\pi f}{f_S} n\right) = \cos\left(\frac{2\pi k}{N} n\right) \quad \text{where} \left\{ \begin{array}{c} n = 0,1,2,3\cdots \\ \dfrac{f}{f_S} = \dfrac{k}{N} \end{array} \right\}$$

Equation 1.8

It's a matter of preference. Either notation is correct. This book uses both notations where appropriate.

1.5 REPRESENTATION OF DIGITAL FREQUENCIES

A digital frequency can be written on paper in units of Hz or in units of radians per second as

$$f_K = \frac{k}{N} f_S \quad \text{or} \quad \omega_K = \frac{2\pi k}{N} f_S$$

A more common method is to express a digital frequency as a fraction where the sampling frequency has been normalized to 1, such as

$$f_K = \frac{k}{N} \quad \text{or} \quad \omega_K = \frac{2\pi k}{N}$$

Table 1.2 Four Ways to Represent a Digital Frequency

		$k=0$	$k=1$	$k=2$	$\cdots$	$k=N/2$
Normalized radians	$\omega_k=\frac{2\pi k}{N}$	0	$\frac{2\pi}{N}$	$\frac{4\pi}{N}$	$\cdots$	π
Radians	$\omega_k=\frac{2\pi k}{N}f_S$	0	$\frac{2\pi}{N}f_S$	$\frac{4\pi}{N}f_S$	$\cdots$	πf_S
Normalized frequency	$f_k=\frac{k}{N}$	0	$\frac{1}{N}$	$\frac{2}{N}$	$\cdots$	0.5
Frequency	$f_k=\frac{k}{N}f_S$	0	$\frac{f_S}{N}$	$\frac{2f_S}{N}$	$\cdots$	$\frac{f_S}{2}$

In all cases, the value of k can range from 0 to $N/2$. Examples of the four representations of digital frequencies are illustrated in Table 1.2 for several values of k.

A frequency in the analog domain has infinite resolution and therefore can take on all possible values. A frequency in the digital domain can only take on specific discrete values, which are multiples of f_S/N. The value of an analog frequency can always be made to match exactly the value of a digital frequency, but since the value of the digital frequency does not have infinite resolution, the opposite is not true.

The amplitude of an analog frequency can take on an infinite number of different values, whereas the amplitude of a digital frequency can only take on discrete values. The number of amplitude values that can be represented by a digital sinusoid is dependent on the bit width of the individual digital samples. For example, suppose the bit width of a bipolar sinusoidal sequence is given by B. The sinusoidal amplitude can take on values equal to 0 and $\pm\left[1,2,3,\cdots,\left(2^{B-1}-1\right)\right]$.

The sole purpose of this chapter is to introduce the mathematical representation of frequency in the digital domain. This book contains a great deal more information on the subject. For a detailed analysis and tutorial on the synthesis of digital frequencies the reader is encouraged to read Chapter 8, "Digital Frequency Synthesis."

Index

Note: The letters *f*, *e*, and *t* indicate that the entry refers to a page's figure, equation, or table, respectively.

A

acceptance test, 628
address generation, 960–961, 960*f*
address ramps, 581–583, 609–611, 615–617
  cycling and, 583, 610
  frequency of, 615
  least significant bits of, 641
  linear, 581
  look up tables, 581*f*, 581, 583, 589, 590, 609, 611*t*, 611, 643
  numerical clock oscillator design and, 581*f*, 610*f*, 625*f*, 625, 634, 634*f*, 638, 638*f*
  offset, 638*f*
  output, 615
  period of, 582, 595, 616
  phase accumulator, 582, 615, 616, 616*f*, 617, 617*f*, 643
  phase dithered, 643, 643*f*
  sample clock offset and, 617*f*
  width of, 582, 632
address space, 776
  circular memory, 813
  decision points and, 776
  DS-1 bit stream generation, 809–810
  variance, 776
alarm bit generators, 769, 802*f*, 803
aliased region, 487*f*
aliased spectra, 193
aliased zeroes, 521*f*
aliasing, 1, 7–8, 192–195, 204, 438–439, 439*f*, 443–445, 466, 468, 469*f*, 483–490, 485*f*, 486*f*, 487*f*, 489*f*, 496–498, 496–498, 496*f*, 497*f*, 510, 514–515, 519, 521–523, 521*f*, 527–536, 528–529, 529–536, 529*f*, 535*f*, 581, 955, 970
  attenuation and, 496*f*
  bands, 496*f*
  decimation and, 439*f*, 510*t*
  first edge, 497*f*
  imaging attenuation and, 497*f*
  imaging bands and, 483*f*, 485*f*, 486*f*, 487*f*
  levels, 487*f*, 521*f*
  nonaliased spectra, 439*f*
  pass band, 489*t*, 510*t*, 515*e*, 519*f*
alternating series test, 284
amplitude, 2, 3, 11*f*, 174, 227, 235, 257, 875*f*, 903*f*, 905*f*
  analog versus digital, 14, 630–631
  Blackman window spectral main lobe, 246
  constant, 290
  discrete Fourier transform, 199–200
  double sideband amplitude modulation, 665
  Euler's formula and, 215
  filtering, 386, 467, 451
  fluctuations, 378
  Fourier series and, 130, 136–137, 139–147, 141*e*, 143*f*, 144*t*
  frequency response, 342, 425, 465

amplitude (*continued*)
  fundamental frequencies and, 598
  growth and decay properties, 292
  Hamming window spectral main lobe, 244
  harmonic frequencies and, 598
  identical, 654, 656, 717, 719
  impulse, 752–753
  increasing, 323
  inverting, 668
  loop gain and, 1020
  main lobe, 241, 243
  maximum, 216, 243, 253*f*, 257, 583, 585–586, 630
  modulation, 665, 979, 1020–1022
  numerical clock oscillator phase offsets and, 583–486
  negative, 666, 732
  noise, 257–258, 586
  overall system, 342
  pass band, 465
  plotting, 666, 667*f*, 669*f*, 707, 732*f*, 732
  pole location and, 339–340
  precision of, 2, 12, 188
  pulse, 154, 159–162
  real poles and, 292
  rectangular windowing and, 240, 253*f*
  reducing, 630–631
  response, 342, 465, 979, 980
  saw tooth waveforms and, 615
  sine waves and, 2, 12
  single-sided coefficients, 150–151
  spectral line, 207
  spectral lines, 182
  spectrum, 171, 171*e*, 174–175, 225, 243, 652, 666, 667*f*, 668, 669*f*, 729–730
  spectrum for cos($2\pi f_0 t$), 169*e*
  spectrum of sin($2\pi f_0 t$), 171*e*
  square waveforms and, 596
  system response term, 480
  translating, 657, 664–669, 667*f*, 669*f*, 729–730, 734
  triangle wave, 155
  unity, 2, 213, 226, 752–753
  voltage waveforms and, 182
  zero, 226, 338
analog AM radio receiver, 575
analog capacitors, 363
analog signal processing, 1–2, 13, 16, 187, 200, 595
  amplitude of, 630
  analog-processing block, 970
  bandwidth, 972
  circuitry, 363, 970, 973–974
  clipping of, 630
  continuous time Fourier transform, 235, 647–688, 689, 692, 711
  conversion from digital, 7, 535–536, 584, 630, 696, 970, 973
  conversion to digital, 7, 194–195, 204, 355, 534, 535*f*, 596, 630, 878*f*, 879–880, 970–972, 971*f*
  filters, 238
  frequency, 14–15
  impulse response, 238
  input, 535*f*, 970
  narrow band, 194
  output, 535*f*, 820*t*
  sampling, 630
  spectrum analyzer, 371
  spectrum analyzers, 371
  telephone channels, 968
  time variable *t*, 582, 623
  tuning, 646
  undersampling, 7, 194
  wideband, 880*f*
analysis frequencies, 254, 255, 712, 881, 890, 1040
analytic functions, 41, 127, 329
annular convergence ring, 282
antennas, 646
antialiasing, 194, 438, 534, 536, 970
  filters, 438, 535*f*
application-specific integrated circuits, 385–386, 513, 633, 769, 774, 792, 961
Applied Signal Technology, 873
approximation errors, 394, 403, 424*f*, 425*f*
approximation expression, 488*e*
arc intervals, 29
argument sequences, 950*e*
arithmetic logic unit, 540*f*, 543, 558, 558*f*, 559*f*, 577, 577*f*, 581*f*, 610*f*, 612*f*, 625, 625*f*, 634*f*, 637, 638*f*, 643*f*, 948, 949*f*, 959*f*, 962, 962*f*
arithmetic noise, 209
attack rate, 826, 853, 978, 978
automatic gain control, xx, 977–1047

amplitude response, 979*f*, 980*f*
attack rate of, 978
circuit parameters, 1030*t*
circuitry, 1033*f*
DC signal response, 1000*f*, 1002*f*, 1005*f*
dead band, 978–979, 980*f*
decay rate, 978, 1015, 1045
feedback coefficient, 988–999, 990*e*, 991*e*, 993*e*, 995*e*, 1032–1037, 1033*f*, 1034*f*, 1035*f*, 1036*f*, 1037*f*
initial response, 1017*f*
noise, 1037–1044, 1037*f*, 1038*e*, 1039*e*, 1039*f*, 1040*f*, 1042*f*, 1043*f*
output signal RMS, 1010*f*
relationship between output and input signals, 987*e*
root mean square of signals, 1005*f*
simulations, 999–1023, 1000*f*, 1002*f*, 1005*f*, 1006*f*, 1007*f*, 1008*f*, 1009*f*, 1010*f*, 1011*f*, 1012*f*, 1013*f*, 1014*f*, 1015*f*, 1016*f*, 1017*f*, 1019*f*, 1022*f*, 1032–1037
transient response, 979*f*, 980*f*, 1005*f*, 1015*f*, 1023–1032, 1023*e*, 1024*e*, 1026*e*, 1026*f*, 1027*e*, 1030*t*, 1031*e*, 1031*f*, 1032*f*, 1033*f*
Type I RMS AGC Circuit, 979*f*, 981–1044, 981*e*, 982*f*, 984*f*, 985*e*, 986*f*, 987*f*, 988*f*, 990*e*, 991*e*, 993*e*
Type II RMS AGC Circuit, 980*f*, 1044–1047, 1045*f*, 1046*f*

B

band limited signals, 172, 177, 529–646, 657–658
analog, 877
discrete Fourier transform bin, 910
discrete Fourier transform filter, 895
input message signal, 661
multiplication, 174–175
spectral inversion of, 195
spectrum, 740
band of interest, 194–195, 468–469, 482–488, 483*f*, 485*f*, 486*f*, 487*f*, 498, 520*f*, 527, 528, 529*f*, 533
band pass filters, 403*t*, 411*e*, 417*t*, 423*t*, 426*t*, 427*t*, 432*t*, 565*t*, 687*f*, 878–884, 883*f*, 899*f*, 914–918
adjacent, 899*f*
banks of, 905*f*
discrete Fourier transform summation bank, 894*f*
enhanced discrete Fourier transform, 905*f*
filter response, 905*f*
overlap, 224
Parks-McClellan routine output, 411*e*
spectral leakage from, 899*f*
bands
lower sideband, 666, 731, 731*f*, 874, 875*f*, 879
mirror image spectral, 740
upper sideband, 665, 665*f*, 731, 731*f*, 874, 875*f*
base band, 194, 204, 219, 482–485, 518, 520, 522–525, 528, 535, 541, 567, 575, 608, 645–646, 645, 657, 669, 672, 679–680, 685, 687, 738, 744, 873, 895, 902, 964, 970
Bell Laboratories, 245
binary points, 609–610, 609*f*, 612, 612*f*, 634–635, 638, 643, 643*f*
bit accumulation, 774, 776–777
bit generators, 802–803, 802*f*
bit location, 768, 770, 782–783, 795–796, 803
cascaded integrator comb decimators, 501*t*, 503*e*, 504*e*, 505*e*
example, 502*t*
Hogenauer LSB procedure, 499–500
Hogenauer LSB procedure, 517
bit rate, 764, 766–769, 774–782, 784, 804, 829, 869, 879, 973
data stream, 766, 768
elastic store memory, 771–774, 771*t*
modifying, 768
bit variance, 777, 779
bit width, 2
cascaded integrator comb, 511*t*
error signal, 843*t*, 866*t*
minimal, 359
of digital errors, 843*t*
register, 511*t*
register, 517*t*
bits
do not insert information, 777*f*
fill, 768

bits (*continued*)
garbage, 768
information, 768
insert information bits, 775*f*
insert information bits, 777*f*
least significant bits, 372, 501–502, 501*t*, 503*e*, 504*e*, 505*e*, 825
noninformation, 768
register bit width, 355, 356
stuff or garbage, 766, 768*f*, 768, 770*f*
truncated, 499
Blackman, Ralph Beebe, 245
block diagrams, 204*f*, 317
channelizers, 880*f*, 962*f*
complex signal translation, 746*f*, 748*f*
converting complex signals to real, 540*f*
decimate by two filter, 446*f*
decimation, 446*f*
DS-1C multiplexer, 802*f*
elastic store memory, 793*f*
folding memory, 926*f*
frequency division multiplex signal extraction, 687*f*
numerically controlled oscillator, 577*f*, 581*f*, 610*f*, 625*f*, 634*f*, 638*f*, 643*f*
phase offset, 638*f*
simplified demultiplexer, 808*f*
spectral inversion control, 942*f*
translation by ej 20*pt* t, 677*f*
translation of a complex signal, 680*f*
typical signal processing system, 535*f*
Brigham, Oran, 268
buffers
double, 923*f*, 925*f*
input memory, 923*f*, 925*f*, 928*f*, 962*f*
split, 959*f*, 960*f*
split addresses, 961*t*

C

carrier signals, 664, 979
cascaded integrator comb filters, 435, 470–531
bit pruning, 499–506, 501*t*, 503*e*, 503*e*, 505*e*
cleanup filter, 526–531
comb filter stage, 471–473
composite, 473
composite filter z-transform, 475–477
decimation filter, 477–478, 483, 499–506, 514–523, 515*t*, 516*f*, 517*t*, 518*f*, 519*f*, 520*f*, 521*f*, 527–528, 529*f*
filter gain, 498
filter response, 473
frequency response, 473
impulse response, 491–492
integrator stage, 474–475
interpolation filter, 478–479, 506–514, 508, 511*t*, 523–526, 524*f*, 528–531, 530*f*
pass band droop, 506–508
phase response, 490–491
pole-zero placement, 493–498
sample rate frequency response, 479–490
Cauchy, Augustin-Louis, 111
Cauchy-Riemann conditions, 42, 47–51, 48*t*, 49*t*
Cauchy's theorem, 96–108, 98*f*, 102*f*, 105*f*, 109–120, 111*e*, 113*e*, 121*e*
ceiling line, 53, 53*f*, 62
center frequency, 261–262, 537, 621–622, 645, 664–665, 681, 729, 730, 809, 822–824, 845, 847, 854–855, 855
constant increments of, 684
converting to, 194, 262
deviation from, 869, 857*f*, 858*f*
filter response, 563
generate, 822
image bands, 525
loop, 862
max variance, 845
normalized, 935
normalized bin, 712
output, 823
point, 847
pull range, 822
sequential discrete Fourier transform bins, 895
simulations of, 866
specifying, 822
temperature, 614
tuning signal, 675, 743, 744
variance, 845
voltage controlled crystal oscillator, 823–824, 829–831, 845–850, 869
wideband frequency division multiplex signal, 892
channelized filter banks, xx, 873–975, 974*e*

block diagrams, 958–962, 959*f*, 960*e*, 960*f*, 961*t*, 962*f*
design of, 919–962, 967–974
equations for, 974–975
function of, 877–919
hardware, 967–974, 968*f*, 969*f*, 970*f*, 971*f*
software simulation results, 962–967, 963*f*, 965*t*, 966*f*, 967*f*
channelizers, 873–879, 875*f*, 876*f*, 878*f*, 880*f*, 958*f*
complex to real conversion, xviii, 942–948, 943*e*, 944*e*, 945*e*, 946*e*, 947*f*, 948*t*
cosine look up tables, 948–955, 949*e*, 949*f*, 950*e*, 950*t*, 951*e*, 951*t*, 952*e*, 953*t*, 954*t*, 955*f*
data, 942*f*
data processing, 907–910, 909*f*
discrete Fourier transform, 881–900, 881*e*, 883*f*, 884*e*, 885*f*, 887*e*, 887*f*, 889*f*, 890*e*, 890*f*, 891*f*, 892*f*, 893*f*, 894*e*, 894*f*, 896*f*, 898*f*, 899*f*, 913*e*
functional, 880*f*
hardware design, 971*f*
implementation, 873
input signal processing, 919–923, 920*t*, 921*f*, 923*f*, 930–930, 930*e*, 931*e*, 931*f*
input test signal, 963*f*
mathematical concepts, 910–919, 911*e*, 911*f*, 912*f*, 913*e*, 915*e*, 916*e*, 917*e*, 918*e*
operational parameters, 919
sample rate decimation, 932–933
segment folding memory, 923–930, 924*e*, 925*f*, 926*f*, 927*f*, 928*e*, 928*f*, 929*e*, 930*e*
software simulation design parameters, 965*t*
stand-alone, 945*e*, 974*e*
time varying phase terms, 933–942, 933*e*, 934*e*, 936*e*, 937*e*, 938*f*, 939*e*, 940*e*, 941*e*, 942*f*
channels, 646
bandwith, 684*f*, 685*f*
channel stack, 537–538, 874, 972–973
identity tones, 966*f*, 967*f*, 968*f*, 969*f*, 970*f*
reference tone, 615, 964–965, 966*f*, 967*f*, 968*f*, 969*f*, 970*f*
circuitry
analog, 970, 973–974
analog-processing block, 970
attack rate of, 978
board geometry, 237
calculating time constant *k*, 854
decay rate, 978, 1015, 1045
feedback coefficient, 985
fixed parameters, 839
frequency response, 376
layout, 237
load, 238
loop gain, 987*e*
LSI 16-by-16-bit multiplier chips, 431
macros, 238
scale factor, 985
small scale integrated circuits, 237
circular memory, 761*f*, 794–796, 794*f*, 797*f*, 798*f*, 799, 800*f*, 811, 813, 814*f*, 816*f*, 817, 825, 835
clock jitter, 825–826, 828, 844, 851, 852*f*, 854, 855*f*, 857*f*, 857*f*, 857*f*, 857*f*, 857*f*, 858*f*, 859*f*
clock oscillators, 455*f*, 456*f*, 613–637, 761, 768, 790, 820, 827, 859
clock teeth, 809, 814
clock tree skew, 238
code
converter code glitches, 534
length, 642
snippets, 788, 947, 1058, 1060, 1083
coefficient memory, 389, 456, 457*f*, 461, 925*f*, 928*f*, 962*f*
coefficients, 388*e*
double-precision, 404, 564
filter, 904
floating point, 564
sliding filter vectors, 548
coherent integration method, 260–261
comparison tests, 284
complementary metal-oxide semiconductor devices, 237–238
complex numbers, 17–37, 38*f*, 39*f*
absolute value of, 35–36
addition, 18*e*
Cartesian form of, 17–21, 18*f*, 18*e*, 19*e*, 20*e*, 21*e*
conjugates, 19*e*, 20*e*, 21*e*
converting Cartesian and polar forms, 23*e*
converting to polar form, 106–107

complex numbers (*continued*)
  division, 18*e*, 24*e*
  dot product, 27*f*
  equality, 18*e*, 23*e*
  exponential form of, 36–37
  graphs of, 38*f*, 39*f*
  magnitude, 19*e*
  multiplication, 18*e*
  multiplication, 23*e*
  polar form of, 21–27, 21*e*, 22*e*, 22*f*, 23*e*, 24*e*, 25*e*, 26*e*
  powers of, 25*e*, 26*e*
  properties of, 18–21, 18*e*, 23–27
  real numbers versus, 16
  representation of, 271–274
  roots of, 27–34, 27*e*, 28*e*, 29*e*, 31*f*, 33*f*, 34*f*
complex variables, 15–127, 272*f*
  analytic functions, 41–42
  Cauchy-Riemann equations, 47–51
  Cauchy's theorem, 96–108, 109–120
  complex differentiation and, 43–46
  contours, 51
  entire functions, 42
  limits and, 40–41
  line integrals, 51–54, 84–96
  real line integrals, 54–84
  residue theory and, 120–127
  simply connected regions, 51
components
  aging, 614, 618, 637
  alternating current, 863–864
  direct current components, 863–864
  in-phase, 16–17
  quadrature, 16–17
  real versus imaginary, 92
  scalable libraries, 386
conjugate pairs, 342–344, 367
constant group delay, 385, 387, 470, 490–491
constant phase delay, 387
contours
  Cauchy's theorem, 98*f*, 102*f*, 105*f*
  circular, 64–65, 81*f*
  closed, 51, 54
  equivalent, 117*f*
  examples of, 51*f*, 56*f*, 67*f*
  integral, 52–53, 53*f*
  integral summation curtain, 53*f*
  integration, 55
  line integrals and, 52
  singularities and, 99*f*
  triangular, 62
control bus interface, 793*f*
Cooley, J. W., 267
Cooley-Tukey FFT algorithm, 268
copper wire, 646, 678, 762, 874, 973
core memory, 632

D

D flip-flop, 614, 769, 792, 794, 803
data
  aligning, 911*f*
  blocks, 911, 923*f*, 926, 933
  channelized, 942*f*
  channelizer processing paths, 909*f*
  discrete Fourier transform channel outputs, 939*e*, 940*e*
  driver source current, 801
  example data transmission frame, 790*f*
  multiplying, 911*f*
  service level, 764
  shifted, 549*f*, 551*f*, 553*f*, 557*f*, 558*f*, 559*f*, 939*e*
  simulation, 567–573, 568*f*, 571*f*, 572*f*
  synchronous, 968
  throttling, 772, 779
  tributary data bit streams, 802*f*
data locked loops, xix, 807–871, 808*f*, 808*f*, 869*e*, 869*e*, 870*e*, 871*e*
  analog error voltage, 818–820, 819*f*, 820*t*
  circular memory and, 814*f*
  clock frequency synthesis, 820–825, 821*f*, 824*t*
  elastic store memory and, 804, 809–817, 812*f*, 813*f*, 814*f*, 816*f*, 810*f*
  error signal bit width versus time constant, 843*t*
  error signals, 817
  error voltage generation, 819*f*
  fill, 813*f*
  frequency devation and, 867*f*
  frequency response and, 827*f*, 828*f*
  modulo operation, 814*f*
  operational characteristics, 825–829, 827*f*, 828*f*
  read pointers, 812*f*, 816*f*

reference clock synthesis, 821*f*
rise time, 861, 862*f*, 865*e*, 866*t*
simulation, 844–868, 846*f*, 848*t*, 849*f*, 852*f*, 853*f*, 855*f*, 857*f*, 858*f*, 859*f*, 860*f*, 862*f*, 864*f*, 865*e*, 866*e*, 866*t*, 867*f*
steady state behavior, 829–834, 829*e*, 830*e*, 830*f*, 831*f*, 832*e*, 832*f*, 866–868, 866*e*, 867*f*
transient behavior, 834–844, 836*e*, 837*e*, 838*e*, 839*e*, 840*e*, 841*e*, 842*e*, 843*t*, 838*e*, 839*e*, 861–866, 862*f*, 864*f*, 865*e*, 866*t*, 812*f*, 816*f*
decimation, 436, 437–448, 437*f*, 438*f*, 439*f*, 440*e*, 440*f*, 441*e*, 442*f*, 443*f*, 444*e*, 445*f*, 446*f*, 447*f*, 448*f*, 521*f*
block diagram, 443*f*, 446*f*
decimate by two filter, 445*f*, 446*f*, 466
factor, 777*f*
filtering, 535*f*, 1045*f*
magnitude, 539*f*
poly phase filters and, 443*f*
ratio, 909*f*
sample rate, 496*f*, 529*f*, 539*f*
derivatives, 44–45
design review, 422, 509, 514, 545–546, 559, 621, 628, 1021
digital capacitors, 363
"digital radio," 528, 533, 535, 646, 677
digital signal processing systems, xiii–xv
complex, 533–570
conversion from analog, 7, 194–195, 204, 534, 535*f*, 596, 630, 878*f*, 879–880, 970–972, 971*f*
conversion to analog, 535–536, 584, 630, 696, 970, 973
converting complex to real signals, 560–573
real, 533–570
simulating, 560–573
tuning, 645–759
typical, 534–540
digital signals, 1–14
defining, 2–8, 3*f*, 4*e*, 5*e*, 6*e*, 7*t*
discrete values, 4, 14
errors, 613–622
mathematical representations of, 9–12, 9*e*, 10*e*, 14*t*
representations of, 9–12, 13–14
standard implementation representation of, 691*e*
synthesizers, 535*f*, 575–644
direct current, 139
direct current components, 863–864
canceling circuit, 376
removal circuit, 379*f*
steady state behavior, 863–864
Dirichlet conditions, 130
discrete Fourier transform, 187–253, 880*f*, 906*f*
adjacent bins, 246, 714
Bartlett windowing, 241–243, 242*e*, 242*f*
bin band pass filters, 218–220
Blackman window, 245–246, 245*e*, 246*f*, 251*f*
channelizer discrete Fourier transform output, 913*e*, 915*e*, 917*e*, 918*e*
complex output points, 206
discrete power spectrum, 211–212, 212*e*, 213*e*
enhanced filter, 956*f*, 957*f*
example application using, 261–263
filter banks and, 907
filter response, 931*f*
finite impulse response filter window, 246, 247*e*, 248*f*
forward, 209*e*, 210*e*
frequency, 189–195
frequency bins and, 249*f*
Hamming window, 243–244, 244*e*, 244*f*, 245*f*
Hanning window, 243, 243*e*, 252*f*
increasing processing gain, 260*e*
inherent frequency response, 974*e*
inverse, 209–210, 209*e*, 210*e*, 647*e*, 650*e*, 933*e*
L points, 880, 881*e*, 883*f*, 906*f*
leakage, 220–236
pairs, 218, 219*t*
poly phase enhanced, 902*f*, 903*f*
preprocessors, 237
processing gain, 254–261, 255*e*
processing number of cycles with, 234*f*, 236*f*
properties, 217, 218*t*
pulse waveforms, 213–217
rectagular window, 238–241, 239*e*, 239*f*, 240*e*, 241*f*, 250*f*, 252*f*

discrete Fourier transform (*continued*)
representation, 210–211
resolution, 209
scale factors and, 210*e*
shifted input data and, 939*e*
standard display conventions, 519
summation filters, 884, 893*f*, 894*f*, 898*f*
symmetry, 195–209
system interpretation of, 896*f*
time, 263–267, 264*e*
windowing, 236–253, 253*t*
"do not insert information" bits, 775*f*, 777*f*
dot products, 26–27, 27*f*, 384, 440–441, 458, 547–549, 552, 557
double precision array, 396*t*, 1059
double sideband amplitude modulation, 665
down conversion, 645, 685
frequency division multiplex signal, 685*f*
down sampling, 436–438, 463, 478, 516
Drichlet conditions, 130
dual inline packages, 637
duty cycles, 140–142, 141*t*, 143*f*, 144*f*, 144*t*, 614, 809, 814, 974
dynamic arguments, 11*f*

E

elastic store memory, xix, 761–805, 780*t*
bit accumulation levels, 774, 776–777
block diagram, 793*f*
brute force implementation, 775–776, 775*f*
conceptual, 794*f*
data locked loops and, 804, 810*f*
data rates, 770–774, 771*t*
depth, 843*t*
designing, 769–770, 774–777
DS-1C multiplexer design, 768–774, 801–803, 802*f*
efficient, 776–777, 777*f*
example application, 762–763
fill, 779, 793*f*, 797*f*, 798*f*, 800*f*, 802*f*
hardware implementation of, 792–801, 793*f*, 794*f*, 797*f*, 798*f*, 800*f*
limits, 779
North American Digital Hierarchy, 765–768, 766*t*, 768*f*
output bit rate, 771*t*
pulse code modulation multiplexing hierarchy, 763–768, 764*t*
read bit rate, 771*t*
simulation, 778–779, 788–791, 790*f*
size, 774–775
software implementation of, 788–791
steady state, 779–783, 780*t*, 782*t*, 783*t*
threshold value, 793*f*
transient statistics, 784–791, 785*t*, 787*t*
energy spectral density, 185–186, 185*e*
equations
Cauchy-Riemann equations, 47–51
Cauchy-Riemann equations in polar form, 50
channelized filter banks, 974–975
digital data locked loops, 869–871
plug-in, 350
pole, 334–350
signal tuning, 754–759
zero standard form, 334–350
errors
decay, 856
design specific loop digital error signal, 829*e*
error accumulation, 856
frequency error bounds, 620*f*
initial, 1016*f*
loop, 1003, 1005*f*
loop digital error signals, 829*e*
maximum, 418
MDAC voltage to error relationship, 830*e*
periodicity of, 602
predefined error bounds, 157
rounding, 600, 601*f*, 602*f*, 605*f*, 639*f*
scaled feedback signal, 985*e*
truncation, 597*f*, 598*f*
voltage generation, 819*f*, 821*f*
voltage response and, 849*f*
zero error operating point, 832*f*
Euler's equation, 167, 188, 188*e*, 215, 266, 306, 306*e*, 307, 322, 337, 344, 480, 653, 655, 683, 686, 702, 706, 713, 716, 718
European or CEPT Digital Hierarchy, 763

F

factorization, 126, 312
fast edge rate clocks, 801

fast Fourier transform, 267–268
fill bit, 768
fill difference, 793*f*, 794*f*, 795, 797*f*, 798*f*, 800*f*, 812, 813*f*, 814*f*, 816*f*
  adjacent filters, 899*f*
  analog, 238
  antialiasing, 194, 438, 534, 535*f*, 536, 970
  anti-imaging, 449
  banks of, 894*f*, 905*f*, 907
  cascade, 428, 463, 467–468, 527–530
  clean up, 469*f*, 514, 523, 529*f*, 530*f*
  coefficient memory, 962*f*
  coefficients, 388, 565*t*, 904, 925*f*, 928*f*
  composite, 906*f*
  decimate by two, 445*f*, 446*f*
  decimation, 436, 535*f*, 1045*f*
  design, 392–425
  direct form, 384, 427
  discrete Fourier transform filters, 220, 907
  enhanced discrete Fourier transform filter, 905*f*, 931*f*
  examples of, 370*t*
  finite impulse response, 383–531
  four-phase, 447
  half band filters, 425–433
  implementation of, 425–433
  length, 423*t*, 426*t*, 427*t*, 432*t*, 441, 565*t*, 906*f*
  multirate filters, 435–531
  overlap, 224
  parallel filter structures, 328
  poly phase filters, 436–465
  recursive, 364, 371*f*, 372*f*
  simulations of, 566*f*, 567*f*
  sliding coefficient vectors, 548
  summation, 884, 887*f*, 888*f*
  transition bands, 904
  transversal architecture, 427
  two-filter cascade, 463
  two-phase, 443
finite impulse response filters, xvii–xviii, 383–531, 403*t*, 417*t*, 423*t*, 426*t*, 427*t*, 432*t*, 565*t*
  architecture, 546*f*
  cascaded integrator comb filters, 470–531
  clean up filters, 529*f*, 530*f*
  design, 392–425
  direct form, 384
  direct form architecture, 389–392
  discrete time differences and, 384*e*
  even coefficient architecture, 389–390
  filter types, 387–389
  half band filters, 425–433
  linear phase, 386–387, 388*e*, 389–392
  low pass digital, 393
  multirate, 435–531
  odd coefficient architecture, 390–392
  Parks-McClellan design method, 392–425
  poly phase filters, 436–465
  shorthand drawing notation for, 547*f*
  transfer function, 385*e*
  windowing, 248*f*
flash memory, 581
flip-flop, 614, 769, 792, 794, 803
floating point, 586, 592, 632, 641, 1009, 1050
  attenuation of, 417
  double-precision, 401, 403, 408, 410, 414, 417, 421–422, 430, 564, 956
  filter coefficients and, 408, 422
  Microsoft Excel, 209
  Parks-McClellan FOTRAN subroutine, 1057–1060, 1065, 1084*t*, 1090
  processors, 404
  samples, 596
  system, 379
fold and sum operation, 924, 925*f*, 948
folding frequencies, 192, 437, 487*f*, 496*f*, 497*f*, 519*f*, 519, 521*f*
folding memory, 923–924, 926, 946–947, 948–949, 948*t*, 949*f*, 958–961, 962*f*
  block diagram, 926*f*
  computational example, 927*f*
  double buffers and, 928*f*
  inversion term, 947*f*
  modified architecture, 949*f*
  split buffer, 959*f*
follow on system, 372, 435
forward Fourier transform, 647, 647*e*
Fourier series, 129–157, 137*f*, 140*e*
  coefficients, 153
  compact form of, 148*e*
  computed, 153
  graphs of, 143*f*, 149*f*, 153*f*
  harmonics of, 141*t*, 144*t*, 148*t*
  harmonics of, 157
  one sided, 153, 153*f*
  Parseval's theorem, 181

Fourier series (*continued*)
  single-sided, 150*e*
  trigonometric, 130*e*, 131*e*, 132*e*, 133*e*, 135*e*, 139*e*
  two-sided, 153*f*
Fourier transform, xvii, 129–269, 380*e*
  continuous, 180*t*, 181*t*, 647–688
  discrete Fourier transform, 263
  discrete time (digital), 187–253, 188*e*, 191*e*, 201, 203*f*, 205*e*, 254–261, 261–267, 689–754
  fast Fourier transform, 267–268
  forward, 158*e*, 159*e*
  Fourier series, 129–157
  graphs of, 201, 202*f*
  inverse, 158*e*, 159*e*
  pairs, 164*e*, 165*e*, 166*e*, 167*e*, 180*t*, 649*e*, 650*e*, 651*e*
  properties, 179, 181*t*
  time sequences and, 274*e*
  unit circle of, 201
  zoom, 435
  z-transform and, 274*e*
frames, 766, 876*f*
  DS-1, 766*t*
  example data transmission, 790*f*
  frame alignment, 766, 802*f*
  half, 772
  master, 768*f*
  microframes, 767, 768*f*, 780*t*
  pulse synchronizers, 363
  subframes, 766, 768
frequency
  aliasing, 1, 7–8, 192–195, 204, 438–439, 439*f*, 443–444, 443–445, 466, 468, 469*f*, 483–490, 485*f*, 486*f*, 487*f*, 489*f*, 496–498, 496*f*, 497*f*, 510, 514–515, 519, 521–523, 521*f*, 527–536, 528–529, 529–536, 529*f*, 535*f*, 581, 955, 970
  aliasing tree diagram, 194*f*
  axis folding, 192*f*, 193*f*
  cutoff, 430*f*
  folding, 8, 8*f*, 192, 192, 193*f*, 438, 467, 484, 486*f*, 487*f*, 496*f*, 497*f*, 519*f*, 521*f*
  graphs of, 192*f*, 193*f*
  input tributary bit clock, 848*t*
  normalized, 9–10, 12, 192, 211, 217, 225–227, 234–235, 239, 249, 250*f*, 251*f*, 252*f*, 258*f*, 266–267, 362, 389, 393, 396*t*, 396, 399, 405–406, 412, 419, 488–489, 534–536, 538, 539–540, 560, 561*f*, 563–564, 566*f*, 567*f*, 567, 570, 571*f*, 690–714, 717*e*, 889–892, 1007–1008, 1013, 1037
  Nyquist, 7, 191, 438, 701*f*, 704*f*, 708*f*, 709*f*, 718*f*, 720*f*
  optimal, 590, 604
  possible values, 6
  radian, 2, 4–5, 10–11, 158–159, 161, 182, 188–189, 239, 253*t*, 266, 338*f*, 339*f*, 341*f*, 343*f*, 347*f*, 349*f*, 371*f*, 372*f*, 372, 375*f*, 377*f*, 378*f*, 648, 690–692, 712, 754*e*, 755*e*, 888–889, 892
  resolution, 2, 12, 191–192, 232–233, 233–234, 240, 243, 264, 268, 435, 579, 586, 604, 608–609, 611–613, 632–633, 633, 826, 828, 833, 842–844, 851–854, 856–858
  step changes, 844
  step downs, 849*f*
  unnormalized, 224
  voltage controlled crystal oscillator, 832*f*
frequency axis
  calibration of, 201
  folding, 193
frequency deviation, 845, 849*f*, 851, 852*f*, 853*f*, 855*f*, 857*f*, 858*f*, 859*f*, 860*f*, 867*f*, 869
frequency division multiplex processing system, 448, 535, 645–646, 679–680, 684–685, 878*f*, 892, 940, 957–958, 964, 966–970, 972
  block diagram, 687*f*
  channels, 535
  channels, 875*f*
  down conversion, 685*f*
  signal extraction, 687*f*
  signal spectra, 684*f*, 878*f*
  time division multiplex conversion, 873–881
  time division multiplex signal pathway, 878*f*
frequency impulse, 221, 570, 668, 671–672, 675, 678, 738, 743, 747, 752, 891
  cancellation, 656
  cosine, 664, 714, 729
  function, 165, 650
  response, 167

shifted, 650, 650*e*
superimposed, 895
frequency inversion phenomenon, 938
frequency jitter, 825, 860*f*
frequency offset, 848*t*
output versus voltage controlled crystal oscillator control voltage, 824*t*
unwanted, 615
zero, 620*f*
frequency rejection, 572*f*
frequency sampling method, 384
frequency shift keyed communications system, 576
fundamental frequency, 4, 6, 11*f*, 190, 595*t*, 612*f*, 634*f*, 638*f*
direct current components, 130, 142
harmonics in relation to, 12, 136, 140
numerically controlled oscillators, 579–580, 584–585, 587–588, 594–595, 598, 608, 611, 613, 615, 632–634, 638

G

gallium arsenide gate arrays, 238, 800
garbage (stuff) bits, 768*f*, 770, 776, 779, 802
generic base band signal spectrum, 518
graphical user interface, 395
Green's theorem, 79–84, 79*e*, 80*f*, 87, 90, 97
ground bounce, 237
grounding techniques, 373
group delay, 386*e*
guard bands, 777, 792, 833–834, 964–965, 973

H

half band filter, 425, 435, 469*f*
frequency response, 469*f*
Parks-McClellan routine output, 432*t*
half frame, 772
Hamming, Richard Wesley, 243
hardware
channelizers, 971*f*
digital sinusoidal generators, 12
elastic store memory and, 792–801
follow on, 372, 435
hardware multipliers, 546
implementation, 12
pole-zero plot of, 297*f*
recursive structure, 295*f*
special purpose digital, 237
harmonic frequencies, 12, 595*t*
fundamental frequencies and, 130
odd, 598
headroom, 801
heap memory, 632
heterodyne, 573
hexadecimal format, 588, 634
Hilbert transforms, 389, 396*e*, 1067, 1072, 1087
histogram, 533
Hogenauer, E. B., 470–471, 488, 512
Hogenauer LSB pruning procedure, 499, 517

I

identity tones, 966*f*, 967*f*, 968*f*, 969*f*, 970*f*
image bands, 512*f*, 568*f*
aliasing and, 483*f*, 485*f*, 486*f*, 487*f*
anti-imaging low pass filter, 449
attenuation, 510*t*, 515*e*, 568*f*
first edge, 497*f*
interpolate by two, 542*f*
residue, 530*f*
image rejection, 512*f*, 572*f*, 932
integers, 396*e*
integral test, 284
integrals
arc length of, 67*f*, 68*e*, 69*f*
common, 110*t*
complex, 110*t*
complex line, 84*e*, 87*e*, 90*e*
contour, 73*e*
definite, 52, 52*f*, 53*f*, 91
double, 79
evaluating, 108*e*
examples of, 57*e*, 57*f*, 58*e*, 59*f*, 61*f*, 63*f*, 65*f*
parametric forms of, 70*e*
path independent, 91*e*
integration
attenuation, 363*t*
frequency response, 362*f*
gain, 260
graphs of, 92*f*, 95*f*

integration (*continued*)
  incoherent, 261
  limits of, 58
  path of, 57–59, 59*f*, 61*f*, 63*f*, 65*f*, 95*f*
  region of convergence of, 352
  register bit width, 355, 356
  register width versus coefficient, 357*t*
interference
  narrow band, 372*f*
  transfer functions and, 375*f*
interpolation, 436, 448–462, 449*f*, 450*f*, 451*f*, 452*e*, 453*f*, 454*e*, 455*f*, 456*f*, 457*f*, 458*f*, 459*e*, 460*f*, 461*f*, 572*f*
  architecture, 559*f*
  cascaded integrator comb, 511*t*
  coefficients, 565*t*
  complex signals, 550–552, 551*f*, 552*e*
  complex to real conversion, 565*t*
  poly phase, 549*f*, 559*f*
  real signals, 543–550, 544*f*, 545*f*, 546*f*, 547*f*, 548*e*, 549*f*, 550*e*
  simulation, 560–567, 561*f*, 565*t*, 566, 567
inverse Fourier transform, 647
invisible functions, 234

J

Japanese Digital Hierarchy, 763
jitter period, 826, 852*f*, 853–857, 855*f*, 857*f*
Joules, 185–187

L

L points, 902*f*, 909*f*, 926*f*, 928*f*, 947*f*, 949*f*
  discrete Fourier transform of, 880, 881*e*, 883*f*, 906*f*
L'Hôpital's rule, 45, 280
Laplace transform, 271
Laurent series expansion, 120–122, 126
lead time, 614, 622, 637
least significant bits, 372, 501*t*, 503*e*, 504*e*, 505*e*, 609, 767
  cascaded integrator comb decimation filter, 499–506, 514–518
  digital data locked design, 818–819, 825–829
  digital data locked loops, 842, 852
  digital data locked transient behavior, 835
  Hogenauer pruning procedure, 499, 517
  phase dithering and, 641–643
  truncation, 501–502, 517*t*
legacy code, 1049–1050, 1064–1065
line integrals, 51, 52–54, 55
  complex, 84–96
  contours and, 52
  integration of, 57
  path-dependent, 63
linear phase digital filters, 403*t*, 411*e*, 417*t*, 423*t*, 426*t*, 427*t*, 432*t*, 565*t*
linear phase, 385
linear time invariant causal systems, 290, 340
linearity, 219*t*, 297
lobes
  main, 163, 220, 222–225, 230, 232, 234–236, 240–244, 246, 248, 250, 253, 373, 509, 884, 886, 892–895, 893*f*, 898*f*, 898, 900–901, 996
  side, 223–225, 230, 232, 236, 240–245, 248, 251, 253–254, 372–373, 884, 886, 898*f*, 899–901, 904
low pass filters, 194–195, 538*f*, 539*f*, 568*f*, 687*f*, 885*f*
  frequency response, 542*f*
  Parks-McClellan routine output, 403*t*
  recursive, 364

M

Maclaurin series expansion, 120
magnitude, 219*e*, 21–25, 31, 35, 37, 68, 99–100, 129, 148–149, 156, 161, 184, 211–213
  aliased frequencies, 522
  applied example, 261–263
  attenuated image frequency, 570
  automatic gain control circuit, 989–990, 1009, 1017*f*, 1034*f*
  averaging, 260–261
  band limited signal, 895
  band pass filter, 897–898, 409*f*, 410*f*
  band pass weight and, 424*f*, 425*f*
  channelized input test signal, 963*f*

channelizer data processing paths and, 909*f*
clock jitter and, 851–860
complex pole, 364
complex variable *z*, 272–273
computing, 266, 345, 361, 374, 1070, 1073–1074
cursor measurements and, 415*f*, 416*f*, 422*f*
digital data locked operations and, 825–829
discrete Fourier transform, 200, 210, 215–217, 220–221, 882–884, 1039–1041
discrete Fourier transform filtering, 956*f*, 957*f*
error signals, 825, 984, 987, 995, 1009
fill, 819*f*, 821*f*
filter lobes, 223–224, 234–236, 240–244, 254, 886
filter response, 232–233
finite impulse response filters and, 386
first order complex conjugate poles, 322–324, 323*f*, 324*f*
floating point sinusoid, 596
frequency discontinuity, 251
frequency response, 222, 334–342, 346–347, 349–350, 365–366, 386
half band filtering, 431*f*
harmonic components and, 599
high pass filtering and, 415*f*, 416*f*
high sample rates and, 480
infinite, 347
input, 355, 895
integrator frequency response, 361
interpolation filter stop band, 570, 573
left-sided sequence, 280–281
log, 595
low pass filtering, 402*f*
maximum, 482, 594, 993
noise, 255–260
output, 235, 371, 903, 966*f*, 968*f*, 969*f*, 970*f*, 984, 987, 995, 1039–1041
pass band ripples and, 396–397
peak, 240
phase response and, 480
reducing, 1035
region of convergence and, 296–297
residual image, 573
right-sided sequences, 280, 284–285
round off error, 600
signal discontinuity, 241
signal energy, 225
spectral leakage, 222–224, 234, 242, 251
spectrum of complex signals, 541
square waveforms and, 599
stabilizing, 825
stop band filtering and, 421*f*, 422*f*
system stability and, 290
truncated sinusoid sample, 596
two-sided sequence, 282
uniform, 649, 694
unity, 2, 10, 222, 234, 473, 692
windowing and, 251, 253*t*
zero, 222, 347, 370*t*, 964
McClellan, James H., 392
microcontroller command, 642
microprocessor bus, 577
Microsoft Excel, 208–209, 221, 227, 228*f*, 322, 338, 340, 341*f*, 342, 343*f*, 347, 349, 350, 365–367, 369, 374, 490, 508, 952, 992, 1029, 1032, 1033*f*
Microsoft FORTRAN Power Station 4.0 compiler, 1063
Microsoft Visual C++ 6.0 Professional compiler, 1063
mixed language programming, xx, 1049–1090
array arguments, 1059–1060
C/C++ main program, 1051, 1083–1090, 1084*t*
calling subroutines and functions, 1051–1054
compiling, 1063–1064, 1090
floating point arguments, 1057–1058
FORTRAN functions, 1055
FORTRAN subroutines, 1054–1055
integer arguments, 1055–1057
Parks-McClellan Fortran subroutine, 1064–1082
pointer arguments, 1060–1063
multiplexing, 443, 688, 802*f*, 808*f*, 880*f*
asynchronous input tributaries, 763–764
DS-1C multiplex elastic store memory, 766
functional DS-0 to DS-1C, 765*f*
input, 448
time division signals, 878*f*
multiply/accumulate operations, 444

multiplying D/A converter, 818, 819*f*, 821*f*
  analog voltage controlled crystal oscillator control voltage, 824*t*
  bipolar configuration, 820*t*
  digital input words, 824*t*
  unipolar configuration, 820*t*
  voltage controlled crystal oscillator control voltage, 824*t*
  voltage reference pin, 822
  voltage to error relationship, 830*e*

N

noise
  additive, 1043*f*
  arithmetic, 209
  floor level, 252, 262
  loop errors and, 1043*f*
  magnitude, 255
  maximum amplitude, 1038*e*
  narrow band, 254
  noiseless input, 1043*f*
  random, 255
noninformation bit, 768
nonlinearity, 822, 834
nonzero, 152
North American Digital Hierarchy, 763, 764*t*, 765–768, 765*f*, 766*f*, 768*f*
numerically controlled oscillators, xviii, 575–644
  accumulator offset and, 577*f*
  block diagrams, 577*f*, 581*f*, 610*f*, 625*f*, 634*f*, 638*f*, 643*f*
  design performance, 638–641, 639*f*, 640*f*
  designing, 628–640
  enhanced phase accumulators, 608–613
  frequency errors from, 613–622
  frequency resolution of, 579, 586, 608–609, 611–613, 633
  harmonics, 595*t*
  industry-grade, 628–640, 629*e*, 630*e*
  operation of, 629*e*
  output, 589*f*, 591*f*, 613–622, 620*f*, 622–628, 626*f*, 635*t*, 639*f*, 640*f*
  phase accumulator, 577–581, 577*f*, 579*e*, 580*e*, 601, 608–613, 609*f*, 610*f*, 611*t*, 612*f*
  phase dither, 641–644, 642*e*, 642*f*, 643*f*, 643*t*
  phase offset, 622–628, 622*e*, 624*e*, 625*e*, 625*f*, 626*f*, 627*f*, 628*f*, 637–638, 638*f*, 640*f*
  phase register, 575–577, 609*f*
  phase words, 577, 633–634, 634*f*, 635*t*
  rounding errors, 639*f*
  sequence length, 606*t*, 607*t*
  sine look up tables, 581–608, 581*f*, 582*e*, 583*e*, 586*e*, 587*e*, 588*t*, 589*f*, 591*f*, 593*f*, 594*e*, 595*e*, 595*t*, 597*f*, 598*e*, 598*f*, 599*e*, 599*f*, 600*f*, 630–633
  synthesized output frequency error, 613–622, 614*e*, 616*f*, 617*f*, 618*e*, 619*e*, 620*f*
  tuning accuracy, 634–637, 635*t*
Nyquist frequency, 7, 204, 191, 536, 541, 559, 573, 636, 688, 696, 932, 938, 942–943

O

oscillators, 4–7, 4*e*, 5*e*, 6*e*, 7*t*
  buying, 614–615, 621–622
  clock, 455*f*
  drift, 618, 620, 621–622
  off the shelf, 586–587, 637, 802*f*
  programmable, 12
  quality of, 614–616
  sample clock, 613–622
  stability, 635
overcorrection, 1018
overflow conditions, 352–354, 356, 358, 474, 769–770, 772, 774, 777, 779, 799, 949, 997, 1021
overlap regions, 304
overshoot, 996, 997, 1018, 1019–1022, 1019*f*, 1020*f*, 1022*f*, 1036

P

Parks-McClellan algorithm, 401
  argument list, 395, 430
  band pass filters and, 411*e*
  FORTRAN subroutine, 1064–1082
  half band filters and, 432*t*
  high pass filters and, 417*t*

low pass filters and, 403*t*
stop band filters and, 423*t*
subroutine arguments, 396*e*
subroutines, 395
Parks, Thomas W., 392
Parseval's theorem, 149–150, 150*e*, 153–155, 179*t*, 181–182, 181*e*, 181–182, 182*e*, 186
partial fractions, 103–104, 312, 328
phase accumulators, 575, 577*f*, 581*f*, 608–613, 610*f*, 625*f*
address ramps and, 616*f*, 617*f*
block diagrams, 577*f*
fractional point, 611*t*, 612*f*
maximum values, 356
numerically controlled oscillators and, 577*f*
numerically controlled oscillators and, 601
output, 577*f*, 611*t*
recursive, 982*f*
phase angle, 274
phase delay, 386*e*
phase dithering, 628
generator, 644
least significant bits and, 641–643
spurious free-dynamic range specifications, 644
variance, 643*f*, 643*t*
poles
complex, 342–350, 344*e*, 344*f*, 345*e*
complex conjugate, 318–322, 345–347, 349
first order, 124
locations and time response, 291*f*
multiple order, 314
real, 335*f*, 337*e*
real versus complex, 291*f*
smallest magnitude, 281
pole-zero analysis, 292, 297*f*, 297, 315, 316*f*, 318, 343, 345, 347, 348*f*, 372–373, 373*f*, 376*f*, 376, 520, 522, 526, 980, 997
CIC filter pole-zero placement design, 493–495, 494*f*
errors, 380
low pass recursive filter example, 364–366, 365*f*
magnitude and, 369
plotting, 358
quantization and, 380
regions of convergence and, 358
poly phase filters, 435, 436–465, 558*f*, 880*f*, 925*f*, 928*f*
block diagrams, 443*f*
channelized filter banks, 955–958, 956*f*, 957*f*, 958
decimation and, 443*f*
decimation filters, 436
discrete Fourier transform, 900–907, 901*e*, 902*f*, 903*f*, 905*f*, 906*f*
enhanced discrete Fourier transform filters, 956*f*
frequency response, 931*f*, 956*f*
impulse response, 909*f*
structure, 906*f*
polynomials, 41
"poor man's tuner", 194–195
power spectral density, 187, 187*e*
power
average, 150, 154, 187*e*
average, 255
bin by bin power comparison, 262
consumption, 514
noise, 255
root mean square of, 980
specific frequency bands and, 187*e*
triangle waveforms and, 154
pseudo random number sequence generator, 642
pseudocode, 789, 791
P-test, 284
pulse code modulation, 763, 968, 1006
multiplexing hierarchy, 763–768
Pythagorean theorem, 2, 25

R

radian chunks, 4–6, 6*e*, 10–12, 190, 691, 888–889, 918
radian frequency, 136, 648*e*, 690*e*, 691*e*
ratio convergence test, 283–285, 283*e*, 287, 352
read pointers, 793*f*, 811, 812*f*, 813*f*, 816*f*, 819*f*, 821*f*
real poles, 280, 282, 296, 366, 368, 379, 381*e*
first order, 315–318
placement, 290–292, 364

real poles (*continued*)
  recursive filter example, 370*t*
  second order, 324–327
  time response and, 291*f*
real zeros, 296, 366–367, 381*e*
  placement, 335–338, 340, 342, 364, 370*t*
recursive accumulator, 352*f*, 353*e*, 354*e*, 370*t*
reference tone, 615, 964–965, 966*f*, 967*f*, 968*f*, 969*f*, 970*f*
reference voltage, 818–823, 819*f*, 821*f*, 830, 845, 869
region of convergence, 275, 279, 286*f*, 288*f*, 289*f*, 352
Remez exchange algorithm, 392, 403*t*, 411*e*, 417*t*, 423*t*, 426*t*, 427*t*, 432*t*, 565*t*
residue, 121, 120–127, 121*e*, 122*e*, 124*e*
  image bands, 530*f*
  postfilter, 523
rise time, 866*t*
  data locked loops, 861, 862*f*, 865*e*, 866*t*
  exponential, 854, 864*f*
  transient, 862*f*
root mean square, 980, 1000*f*, 1010*f*, 1019*f*, 1022*f*, 1030*t*, 1045*f*
  automatic gain control and, 1005*f*
  output signals, 1017*f*
root test, 284

S

sample clocks, 1–2, 4–6, 10–11, 190, 293, 435, 440–441, 455, 462, 544, 546–547, 577–583, 577*f*, 581*f*, 604, 608–625, 610*f*, 611*f*, 612*f*, 613–622, 618*e*, 620*f*, 620*f*, 625*f*, 628–629, 634*f*, 635–636, 643–644, 643*f*, 704, 709, 878*f*, 889, 972, 972, 982*f*
  cycle, 611*t*
  offsets, 617*f*
  oscillator drift, 618, 620, 621–622
  oscillators, 620*f*
  phase accumulators and, 617*f*
sample rates, 612*f*, 909*f*
  changing by noninteger value, 462–465, 463*f*, 464*f*, 465*f*
  decimated, 496*f*, 529*f*, 539*f*, 909*f*
  fractional, 253*t*
  normalization of, 1028*e*
  time series and, 921*f*
sequences
  bipolar sinusoidal, 14
  correlation scores, 189
  current input time series data, 909*f*
  generators, 641
  length, 606*t*, 607*t*
  maximal length, 605, 641
  repetition, 605
sidebands
  double sideband amplitude modulation, 665
  lower, 665*f*, 666, 731*f*, 875*f*
  upper, 665–666, 665*f*, 731*f*, 875*f*
sign manipulation operations, 543, 558*f*, 688
signal to noise ratio, 254–260, 255*e*, 256*e*, 258, 595–596, 595*e*, 608, 628–632, 640, 1038–1041
signal tuning, xix, 645–759
  complex conjugate symmetry, 681–684, 682*f*, 683*f*
  complex exponential tuning signals, 741
  complex frequency translation, 668–688, 669*f*, 684*f*, 685*f*, 686*e*, 687*e*, 687*f*
  continuous time (analog) Fourier transform, 647–688, 647*e*, 648*e*
  discrete complex frequency translation, 734–759, 750*f*, 751*f*, 753*f*, 754*e*, 755*e*, 756*e*, 757*e*, 758*e*, 759*e*
  discrete frequency domain unit impulse response, 696–702, 699*e*, 700*e*, 701*e*, 701*f*
  discrete frequency identities, 720–727, 721*e*, 722*e*, 723*e*, 724*e*, 725*e*, 726*e*
  discrete real frequency translation, 727–734, 728*f*, 729*e*, 730*f*, 731*f*, 732*f*, 733*f*
  discrete time (digital) Fourier transform, 689–716, 689*e*, 690*e*, 691*e*, 692*e*, 710*f*, 713*e*, 714*e*, 714*f*, 715*f*
  discrete time domain unit impulse response, 692–695, 692*e*, 693*e*, 694*e*, 695*f*
  frequency domain unit impulse, 650–651, 650*e*
  frequency domain unit impulse, 651*f*
  frequency translation, 657–661, 657*e*, 658*e*, 659*e*, 660*e*
  incremental, 885*f*

maximum error, 629
quadrature, 622
range, 629
real frequency translation, 661–668, 662*e*, 662*f*, 663*e*, 663*f*, 664*e*, 664*f*, 665*f*, 667*f*, 668*f*
resolution, 629
time domain unit impulse, 648–649, 648*e*, 649*e*
signals
analog, 880*f*
average power of, 186*e*
band limited, 172, 665*f*, 740*f*, 745*f*
bounded, 290
carrier, 615, 664
clipping, 630
converting complex signals to real, 540–560, 540*f*, 542*f*, 544*f*, 545*f*, 546*f*, 547*f*, 548*e*, 549*f*, 550*e*, 551*f*, 552*e*, 553*f*, 554*e*, 555*e*, 555*f*, 556*e*, 556*f*, 557*f*, 558*f*, 559*f*, 535*f*
detection and identification, 254
discontinuity, 241
discrete Fourier transform of, 196*f*, 200*f*, 204*f*, 205*e*, 206*f*
end-to-end signal flow, 808
energy spectral density, 185–186
error signals, 829*e*, 832*f*, 848*t*, 866*e*
interpolated, 530*f*
mean square of, 981
paths, 670
peak, 1040*f*
periodic, 150
perturbation, 1021
preprocessing, 252
root mean square of, 1017*f*
scaled feedback error, 985*e*
time division multiplexed, 878*f*
total energy of, 182–183, 183*e*, 186
truncated, 227*f*, 229*f*
wideband, 906*f*
simulations
automatic gain control, 1002*f*, 1005*f*, 1006*f*, 1007*f*, 1008*f*, 1009*f*, 1010*f*, 1011*f*, 1043*f*
bit-level digital data locked loops, 845–868
channelizer filter bank software simulation, 962–967
code snippets, 788
complex to real signal conversion, 560–573, 565*t*, 567*f*
data locked loops, 848*t*
design parameters, 965*t*
elastic store memory, 785*t*, 787*t*
loop errors, 1043*f*
parameters, 846*f*, 848*t*
software, 566*f*, 567*f*, 644, 846*f*
steady state, 780*t*, 782*t*, 783*t*
translation, 566*f*
truncation errors, 597*f*
Singleton, Richard C., 268
singularities, 122*e*
contours and, 99*f*
first order, 124
graphs of, 99*f*, 108*f*
multiple order, 106–107, 108*f*
multiple, 123*f*
spectral inversion control, 942*f*
spectral inversion, 940–942, 965
spectral leakage, 222, 225–227, 226*f*, 229–230, 230*f*, 232–234, 233*e*, 236, 238, 240, 242, 246, 250–252, 255–257, 593–594, 602, 715–716, 715*f*, 898–901, 903–904, 965, 1038
adjacent filters and, 899*f*
band pass filters and, 899*f*
reduction, 233–234
spectral line, 595*t*
spectral nulls, 163*f*
spectral representation, 878*f*
spurious free dynamic range, 590, 595–596, 600, 607–608, 632–633, 641–642
static random access memory, 632, 769
steady state behavior, 834
DC component, 863–864
loop gain, 1015*f*
output, 979*f*
stuff (garbage) bits, 768*f*, 770, 776, 779, 802
surface mount packages, 637
sweet frequency, 224–225, 250–251
synchronous digital hierarchy, 968
synchronous optical network specifications, 763, 968
synchronous transport module, 968

T

Taylor series expansion, 120
temperature, 614–615, 618, 621–622, 636–637, 761, 772, 834–844, 869, 995, 1004
  operational, 618
  voltage controlled crystal oscillators, 637
time division multiplex processing, 680, 878*f*, 880*f*, 957, 967–969, 973
  frequency division multiplex conversion, 873–881, 878*f*, 880*f*
transient response, 1014–1015
  automatic gain control circuits, 1030*t*, 1033*f*
  data locked loops, 838*e*, 839*e*, 866*t*
  exponential, 1019*f*, 1030*t*
  loop error, 1019*f*
  measuring, 1033*f*
  rise time and, 866*t*
transition bands, 394*f*, 404*f*, 412*f*, 418*f*, 429*f*
translate-and-filter operation, 917
translation, 646, 875*f*
  band limited signals, 665*f*, 676*f*, 731*f*, 740*f*, 745*f*
  block diagram of, 746*f*, 748*f*
  complex signal, 746*f*, 747*f*, 748*f*, 750*f*, 751*f*
  discrete Fourier transform summation filters, 893*f*, 894*f*
  narrow band signals, 733*f*, 735*f*
  quarter band, 543, 553–555, 553*f*, 554*e*, 556*f*, 558*f*, 568*f*
  real signal, 728*f*, 732*f*, 735*f*, 736*f*, 741*f*, 662*f*
  spectral magnitude, 733*f*
Treichler, John, 873
Tukey, James W., 267
two's complement format, 585, 960*f*

U

underflow conditions, 769–770, 772, 774, 777, 779
undersampling, 7, 193–194, 536
undershoot, 996–997, 1018, 1020*f*, 1022*f*
unit circle, 2–6, 3*f*, 4*e*, 9–10, 12–13, 38*f*, 39*f*, 77, 111–112, 189–190, 190*e*, 200, 201, 275–276, 279–282, 285, 286*f*, 290, 291*f*, 296, 297*f*, 318, 322–323, 336, 338–340, 342, 350, 352, 355, 359–361, 365–366, 365*f*, 373*f*, 373, 377, 379, 373, 377, 379, 473–474, 479, 493, 495, 522, 526, 690–691, 694, 696–697, 704, 708–709, 888–889, 889*f*, 915, 917–918, 934, 937
unit delay, 292*f*, 293*e*, 294*e*, 295*f*
unity coefficients, 491
up sampling, 436, 450*f*

V

vector multiplication, 189
VHSIC hardware description language, 543, 792
Visual C/C++, 395
voltage controlled crystal oscillator, 637, 807, 821–826, 828–835, 840–842, 845, 847–851, 866–871
  control voltage, 824*t*
  design example, 832*e*
  digital error signals and, 832*e*, 866*e*
  error response versus voltage, 831*e*
  frequency, 821*f*, 830*f*, 832*f*
  frequency response, 866*e*
  frequency response versus voltage, 830*f*, 831*e*, 866*e*
  ideal frequency, 832*f*
  input error voltage, 849*f*
  linear, 830*f*
  nonlinearity, 822
  operating region, 831
  output frequency, 824*t*, 830*f*
  synthesized frequency, 849*f*
voltage
  analog output, 820*t*
  curve, 832*f*
  error, 849*f*, 819*f*, 821*f*, 849*f*
  fractional output, 824*t*
  multiplying D/A converter reference pin, 822
  reference, 818–823, 819*f*, 821*f*, 830, 845, 869
  voltage controlled crystal oscillator control voltage, 824*t*

voltage controlled crystal oscillator
frequency versus, 830*f*, 831*e*, 832*f*, 866*e*
Von Hann, Julius, 243

W

waveforms
cosine waves, 3, 6, 8, 11, 131, 148, 174, 182, 190, 200, 205, 225, 227, 657, 662, 703–704, 711–716, 728, 950–952, 955*f*
discrete, 213*f*, 217*f*
duty cycle pulsed, 141*t*, 143*f*, 144*t*
even symmetric triangular, 144*f*, 149*f*
frequency spectrum of, 161*e*, 162*f*
graphs of, 160*f*
infinite length, 158
mapped, 11*f*
negative exponential, 748
noise and, 257*f*
normalizing, 149*e*, 150*e*
pulse, 160*f*, 161*e*, 162*f*, 213*f*, 217*f*
purity of, 592
sawtooth, 577*f*, 578
sinusoidal, 2–12, 11*f*, 131, 175, 205, 227, 235, 240, 257*f*, 258, 576, 581, 582–585, 589–590, 599, 604, 626, 629, 641, 704, 708–716, 801, 1011–1012, 1038
spectrum, 217*f*
square, 137*f*, 140, 154, 157–158, 596–600, 597*f*, 598*e*, 598*f*, 599*e*, 602, 614, 801, 821
triangle, 142–143, 147, 148*t*, 154–155, 157
voltage, 130
weighting function, 394, 396*e*
wiper arms, 443, 445, 453–456, 455*f*, 460, 462, 549–552, 555, 557–558, 906*f*, 907, 912
write pointers, 793*f*, 811, 812*f*, 813*f*, 816*f*, 819*f*, 821*f*

Z

zeros
aliased, 521*f*
complex, 344*e*, 344*f*, 345*e*, 345*f*, 347*f*, 375*f*
critical, 388*e*
crossings along the spectrum, 163
inserted, 458*f*, 545*f*
interleaved sequences, 546
magnitude tabulations for, 370*t*
pole-zero, 373*f*
real, 335*f*, 337*e*
z-transform, xvii, 271–382
applications of, 350–380
common pairs, 304–308, 304*e*, 305*t*, 306*e*, 306*t*, 308*e*
convergence of, 279–289, 281*f*, 282*f*, 283*e*, 284*e*, 285*e*, 286*e*, 286*f*, 287*e*, 288*e*, 288*f*, 289*f*
convolution of sequences, 301–303, 302*e*, 303*e*
differentiation and, 299, 299*e*
equations using, 334–350
equations using, 380–381
evaluating, 275*f*
final value theorem, 300–301, 300*e*
Fourier transform and, 274*e*
frequency response, 276*e*
initial value theorem, 300–301, 300*e*
inverse, 308–334, 308*e*, 309*e*, 310*e*, 311*e*, 311*f*, 312*e*, 313*e*, 329*e*, 330*e*
left-sided, 277–278, 277*e*, 281*f*, 286*e*, 287*e*, 288*f*
linearity, 297
mechanics of, 274–277
modifying an existing circuit, 363–380
multiplication of sequences, 303–304, 304*e*
one-sided, 380*e*
partial fraction expansion method, 312–328, 312*e*, 313*e*
power series expansion method, 308–311, 308*e*, 309*e*, 310*e*, 311*e*, 311*f*
properties of, 292–304
recursive accumulator, 350–363
residue method, 328–334, 329*e*, 330*e*
right-sided, 278, 278*e*, 281*f*, 284*e*, 285*e*, 286*f*, 298–299, 298*e*, 300–301, 300*e*, 380*e*
shifting, 297–298, 298*e*, 380*e*
simplifying signal processing with, 275*e*
system stability and, 290–292
time sequences and, 274*e*
two-sided, 278–279, 278*e*, 288*e*, 289*f*, 380*e*
unit delay, 292–297