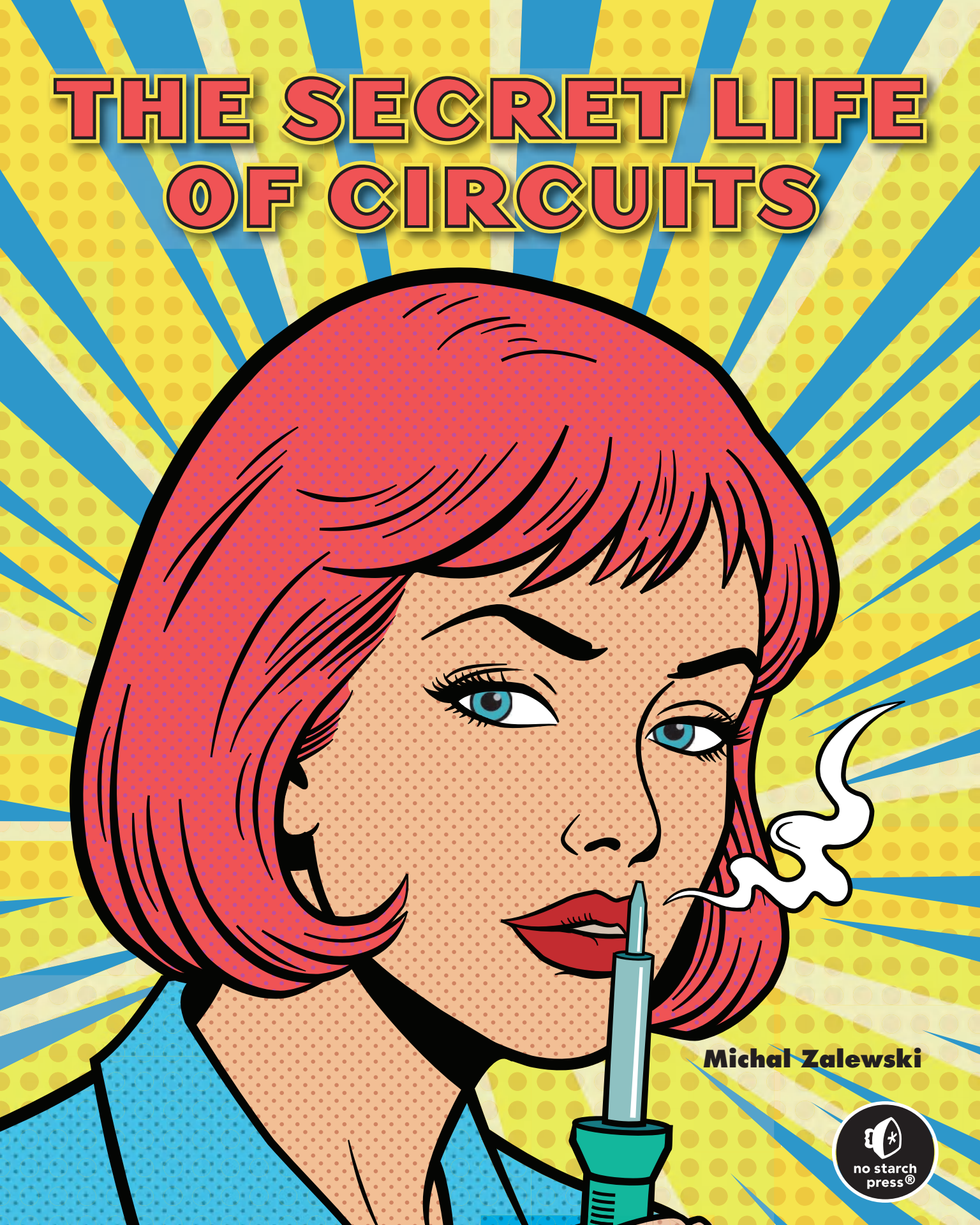
THE SECRET LIFE OF CIRCUITS



Michal Zalewski

no starch press®

PRAISE FOR
THE SECRET LIFE OF CIRCUITS

"Reading this gem of a book is like exploring the component drawers at RadioShack, with an expert on hand to explain what each part does and how to design electronics with them. There is no better introduction."

—TRAVIS GOODSPEED, AUTHOR OF *MICROCONTROLLER EXPLOITS*

"*The Secret Life of Circuits* is neither a textbook nor a trivial introduction; it's a balanced mix of practical knowledge, math, and conceptual explanations free of clichéd analogies."

—ERIC SCHLAEPFER, CREATOR OF THE MONSTER 6502 AND SNARK BARKER, AND CO-AUTHOR OF *OPEN CIRCUITS*

"A complete tour of electronics from low-level physics to modern microcontrollers, along the way providing hands-on experiments and insights that make even complex topics like impedance matching and antennas feel intuitive."

—COLIN O'FLYNN, CO-AUTHOR OF *THE HARDWARE HACKING HANDBOOK* AND ASSISTANT PROFESSOR OF ELECTRICAL AND COMPUTER ENGINEERING, DALHOUSIE UNIVERSITY

"Michal's book bridges the difficult chasm between theory and practice with beautiful illustrations and an easy-to-understand framing of the physics of electronics."

—CHRIS GAMMELL, FOUNDER OF CONTEXTUAL ELECTRONICS AND CO-HOST OF *THE AMP HOUR*

"Full color and packed with Zalewski's own hand-drawn diagrams and workbench photographs. This looks like a book that'll sit comfortably next to Charles Platt's *Make: Electronics*."

—*BOING BOING*

THE SECRET LIFE OF CIRCUITS

An Illustrated Guide to Electronic Circuit Design

by Michal Zalewski



no starch press®

San Francisco

THE SECRET LIFE OF CIRCUITS. Copyright © 2026 by Michal Zalewski.

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

First printing

30 29 28 27 26 1 2 3 4 5

ISBN-13: 978-1-7185-0480-6 (print)
ISBN-13: 978-1-7185-0481-3 (ebook)



Published by No Starch Press®, Inc.
245 8th Street, San Francisco, CA 94103
phone: +1.415.863.9900
www.nostarch.com; info@nostarch.com

Publisher: William Pollock
Managing Editor: Jill Franklin
Production Manager: Sabrina Plomitallo-González
Production Editor: Gina Linko
Developmental Editors: Nathan Heidelberger and Sierra Fong
Cover Illustrator: Michal Zalewski
Interior Design: Octopod Studios
Technical Reviewer: Eric Schlaepfer
Copyeditor: Audrey Doyle
Proofreader: Isabel Kunkle
Indexer: BIM Creatives, LLC

Library of Congress Control Number: 2026010527

For customer service inquiries, please contact info@nostarch.com. For information on distribution, bulk sales, corporate sales, or translations: sales@nostarch.com. For permission to translate this work: rights@nostarch.com. To report counterfeit copies or piracy: counterfeit@nostarch.com. The authorized representative in the EU for product safety and compliance is EU Compliance Partner, Pärnu mnt. 139b-14, 11317 Tallinn, Estonia, hello@eucompliancepartner.com, +3375690241.

No Starch Press and the No Starch Press iron logo are registered trademarks of No Starch Press, Inc. Other product and company names mentioned herein may be the trademarks of their respective owners. Rather than use a trademark symbol with every occurrence of a trademarked name, we are using the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The information in this book is distributed on an "As Is" basis, without warranty. While every precaution has been taken in the preparation of this work, neither the author nor No Starch Press, Inc. shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in it.

[E]

I'd like to thank my wife for her patience and support, and my children for being eager testers of wildly impractical electronic devices. I would also like to thank Eric Schlaepfer for going above and beyond in his review of the book's manuscript.

About the Author

Michal Zalewski is a computer security specialist and an avid tinkerer with more than three decades of experience in electronic circuit design. He runs a popular, in-depth electronics blog and has published a number of hardware projects featured on Hackaday and in other venues. He spent more than a decade at Google building its product security program and later served as VP of security and privacy engineering at Snap Inc. He is also the author of two classic information security books—*The Tangled Web* and *Silence on the Wire*—along with *Practical Doomsday*, a guide to level-headed emergency preparedness (all from No Starch Press).

About the Technical Reviewer

Eric Schlaepfer co-authored *Open Circuits*, a book of circuit cross-sections, and runs an engineering-focused Bluesky account, @tubetime.bsky.social. There, he shares cross-section photos, retro computing and reverse-engineering projects, engineering accident analyses, and the occasional vacuum tube or two. Some of his best-known projects include the MOnSter 6502 (the world's largest 6502 microprocessor, built from individual transistors), the Snark Barker (a retro recreation of the classic Sound Blaster sound card), and the Three Fives and XL741 transistor-scale replica chip kits. He holds a BS in electrical engineering from California Polytechnic State University, San Luis Obispo, where his diploma was signed by Arnold Schwarzenegger.

BRIEF CONTENTS

Introduction xxi

PART I: THE FUNDAMENTALS

Chapter 1: A Primer on Conduction 3
Chapter 2: The Curious Case of Semiconductors 11
Chapter 3: Time-Invariant Circuit Characteristics 17
Chapter 4: Signals That Change Over Time 27
Chapter 5: Ideal Inductors and Capacitors 53
Chapter 6: Electronics at the Speed of Light 71

PART II: CIRCUIT BUILDING BLOCKS

Chapter 7: Voltage and Current Sources 101
Chapter 8: Common Passive Components 111
Chapter 9: Basic Electromechanical Devices 129
Chapter 10: Discrete Semiconductors 145
Chapter 11: Creating Circuit Schematics 167

PART III: EXPLORING ANALOG CIRCUITS

Chapter 12: Arrangements of Passive Components 175
Chapter 13: Applications of Diodes 205
Chapter 14: Working with Transistors 217
Chapter 15: Operational Amplifiers 235
Chapter 16: Supply Regulation 275
Chapter 17: Other Analog ICs 285

PART IV: THE DIGITAL REALM

Chapter 18: Logic Gates 293
Chapter 19: Adders and Binary Math 305
Chapter 20: Memory 313
Chapter 21: Sequential Operation 321
Chapter 22: Analog to Digital and Back 331
Chapter 23: Clocks in Digital Circuits 345
Chapter 24: A Computer at a Glance 353

PART V: SOFTWARE EATS THE WORLD

Chapter 25: Exploring Microcontrollers 359
Chapter 26: Working with Peripherals 365
Chapter 27: Hello, World 373

PART VI: BEYOND PROTOTYPES

Chapter 28: Surface-Mount Components 387
Chapter 29: The Basics of PCB Design 393
Chapter 30: High-Speed Signals 401
Chapter 31: Electronics That Last 409

Epilogue 413
Appendix: Tools and Supplies 415
Index 427

CONTENTS IN DETAIL

INTRODUCTION xxi

The Challenge of Learning Electronics xxii
What's in the Book xxii

PART I
THE FUNDAMENTALS

1
A PRIMER ON CONDUCTION 3

The Structure of the Atom 3
The Physics of Conduction 5
- Insulators vs. Conductors 6
- Conduction in a Metal Wire 6
- Other Forms of Conduction 7

2
THE CURIOUS CASE OF SEMICONDUCTORS 11

Properties of Elemental Semiconductors 11
Doped Semiconductors 12
The p–n Junction 13
A Sneak Peek at Field-Effect Transistors 14

3
TIME-INVARIANT CIRCUIT CHARACTERISTICS 17

Charge 17
Current 18
Energy and Power 19
Voltage 20
Resistance 22
- Ohm's Law 22
- Joule's Law 23

The Question of Signs 24

4
SIGNALS THAT CHANGE OVER TIME 27

What Is a Signal? 28
- Frequency and Amplitude Measurements 29
- Root-Mean-Square Voltage 30

Properties of Sinusoids 32
- Sine Wave Basics 32

The Example of Simple Harmonic Motion 36
Sine Rate of Change 44
Pythagorean Identity and Sine-Wave RMS 46
Working with Non-Sine Signals 48
Discrete-Time Models 48
Sum-of-Sines Approximation 50

5
IDEAL INDUCTORS AND CAPACITORS 53

Capacitance 53
Capacitor Voltage vs. Current 55
A Discrete-Time Capacitor 55
Capacitive Reactance 57
Inductance 59
A Discrete-Time Inductor 59
Inductive Reactance 60
Impedance 61
L-C in Series 62
R-L and R-C in Series 65

6
ELECTRONICS AT THE SPEED OF LIGHT 71

Measuring the Speed of Electricity 71
How Signals Interact with Matter 73
How Signal Velocity Relates to Inductance and Capacitance 76
Signal Reflections 77
Characteristic Impedance 81
Quarter-Wavelength and Half-Wavelength Lines 82
Radio Waves 84
A Bad Antenna (and a Good One Too) 84
Modulation Schemes 86
Beat Frequencies 89
Superheterodyne Receivers 92
Magnets—How Do They Work? 94
All Aboard the Relativity Train 94
From Electrostatic Fields to Magnetism 95
Back to the Classics 97

PART II
CIRCUIT BUILDING BLOCKS

7
VOLTAGE AND CURRENT SOURCES 101

Ideal Power Supplies 101
Batteries 102
Current and Voltage Characteristics 104
Rechargeable Cells 106
AC Adapters 106

Solar Cells . . . 108
Odds and Ends . . . 109
Schematic Symbols for Power Supplies . . . 109

8
COMMON PASSIVE COMPONENTS 111

Resistors . . . 111
Fixed Resistance . . . 112
Variable Resistance . . . 112
Resistor Power Ratings . . . 113
Component Parasitics . . . 114
Johnson–Nyquist Noise . . . 116
Capacitors . . . 119
Types and Ratings . . . 119
Component Parasitics . . . 120
Effects of Temperature . . . 121
Dielectric Saturation . . . 121
Age-Related Considerations . . . 123
Inductors and Chokes . . . 123
Component Parasitics . . . 123
Core Saturation and Hysteresis . . . 125
Ferrite Beads and Common-Mode Chokes . . . 126
Transformers . . . 127

9
BASIC ELECTROMECHANICAL DEVICES 129

Connectors . . . 129
Switches . . . 130
Electrical Ratings and Architecture . . . 131
Switch Bounce . . . 132
Switch-Based Sensors . . . 132
Simple Magnetic Actuators . . . 133
Speakers . . . 134
Relays . . . 135
Motors . . . 136
Brushed DC Motors . . . 137
Brushless DC Motors . . . 137
Piezoelectric Crystals . . . 138
Microphones . . . 141
Deciphering Decibels . . . 142
Quantifying Microphone Sensitivity and Noise . . . 142
Circuit Protection Components . . . 143

10
DISCRETE SEMICONDUCTORS 145

Conventional Diodes . . . 145
Reverse-Bias Operation . . . 147
Other Diode Behaviors . . . 148
Specialized Variants . . . 149

Light-Emitting and Light-Sensing Diodes 150
Transistors 153
JFETs 153
Metal-Oxide-Semiconductor FETs 156
Bipolar Junction Transistors 160
Remembering the Vacuum Tube 164

11
CREATING CIRCUIT SCHEMATICS **167**

Representing Connections 168
Component Drawing Styles 169
Logical Flow 170

PART III
EXPLORING ANALOG CIRCUITS

12
ARRANGEMENTS OF PASSIVE COMPONENTS **175**

A Brief Review of Circuit Theory 175
Kirchhoff's Circuit Laws 176
Impedance in Series 178
Impedance in Parallel 179
Mixed Impedances 180
Complex Currents, Voltages, and Impedances 181
Uses of Resistors 183
Current Limiters 183
Current Sensors 184
Voltage Dividers 185
Analog Voltage Shifters 186
Pull-Ups and Pull-Downs 188
Uses of Capacitors 189
The Dark Art of Charging a Capacitor 189
R-C Low-Pass and High-Pass Filters 192
Uses of Inductors 201

13
APPLICATIONS OF DIODES **205**

Circuit Protection and Signal Clipping 205
Voltage References 207
Rectification 209
Voltage Doubling 213
DC Restoration 214
Diode-Based Logic Gates 215

14
WORKING WITH TRANSISTORS 217

Switching Circuits 218
Common-Source Switch 218
Polarity-Detecting Switch 219
Signal Followers 220
Voltage Follower 220
Push-Pull Stage 223
Current Mirror 224
Basic Amplifiers 225
Inverting Amplifier 226
Differential Amplifier 227
Relaxation Oscillator 230

15
OPERATIONAL AMPLIFIERS 235

Op-Amps at a Glance 235
IC Packages and Symbols 237
DC Characteristics 238
AC Characteristics 239
A Note on Old vs. New 241
Op-Amp Voltage Amplifiers 242
Voltage Follower 242
Noninverting Amplifier 244
Inverting Amplifier 247
Bandwidth Control 248
Current-to-Voltage Amplifier 249
Frequency Response 249
Designing a Microphone Amplifier 251
Oscillators 253
Relaxation 253
Phase-Shift 256
Active Filters 259
Bending the R-C Slope 259
Anatomy of a Catastrophe 261
Multistage R-C Filter 263
Sallen–Key Topology 264
True Mathematics 268
Summing Amplifier 268
Difference Amplifier 270
Op-Amp Multiplication and Division 271
Integrator 272
An Improved Integrator 273

16
SUPPLY REGULATION 275

Linear Regulators 275
Pulse-Width Modulation 277
Charge Pumps 278

Buck Converters 280
Boost Converters 282

17
OTHER ANALOG ICS **285**

Timers and Oscillators 285
Interface Chips 286
Switched-Capacitor Filters 286
Sensors 288
Odds and Ends 289

PART IV
THE DIGITAL REALM

18
LOGIC GATES **293**

Defining Digital Circuits 293
Truth Tables 295
Logic Gate Design 296
    Buffer and Inverter 296
    OR, AND, NOR, and NAND 298
    Exclusive OR 301
Universal Gates 302

19
ADDERS AND BINARY MATH **305**

An Introduction to Binary Arithmetic 305
Building an Adding Circuit 307
    Half Adder 307
    Full Adder 308
Toward Multiplication and Division 311

20
MEMORY **313**

Latches 313
    S-R Latch 313
    Gated S-R Latch 315
    D Latch 316
Flip-Flop Circuits 316
    D Flip-Flop 317
    T Flip-Flop 318
Beyond Logic Gates 318

21
SEQUENTIAL OPERATION 321

Counters 321
- Ring Counter 322
- Ripple Counter 323
- Synchronous Counter 325

Data Routing 325
- Line Decoder 326
- Multiplexer 327
- Demultiplexer 328
- Shift Register 329

22
ANALOG TO DIGITAL AND BACK 331

Digital-to-Analog Converters 331
- Binary-Weighted DAC 333
- R-2R DAC 334
- Oversampling Averaging DAC 336

Analog-to-Digital Converters 338
- Flash and Pipelined ADCs 339
- Slope-Based ADC 340
- Successive Approximation Register ADCs 341
- Delta-Sigma ADC 342

23
CLOCKS IN DIGITAL CIRCUITS 345

Clock Sources 345
- R-C Oscillators 346
- Crystal Oscillators 346

Clock Adjustment 348
- Division 348
- Multiplication 349

24
A COMPUTER AT A GLANCE 353

Architectural Outline 353
Instruction Set 355
Further Optimizations 356

PART V
SOFTWARE EATS THE WORLD

25
EXPLORING MICROCONTROLLERS 359

Features of a Microcontroller 360
Choosing the Right Device 361
- Process Control 361

Serious Compute . . . 362
Internet of Things . . . 362
Essential MCU Toolkit . . . 363
Programming Language . . . 363
Programmers . . . 363
Build Environment . . . 364

26
WORKING WITH PERIPHERALS **365**

Memory-Mapped Registers . . . 366
Digital Port I/O . . . 366
Parallel 8080/6800 Protocols . . . 367
Serial Peripheral Interface . . . 368
Pulse Width Modulation . . . 369
Analog Accessories . . . 371

27
HELLO, WORLD **373**

Blinking LED . . . 373
The Software Side . . . 375
Programming the Chip . . . 376
LED Chaser . . . 377
Push-to-Advance LEDs . . . 378
ADC Bar Meter . . . 379
PWM Audio Toy . . . 381

PART VI
BEYOND PROTOTYPES

28
SURFACE-MOUNT COMPONENTS **387**

Common Two-Terminal Components . . . 388
Transistors and Friends . . . 389
Integrated Circuits . . . 389
Odds and Ends . . . 390

29
THE BASICS OF PCB DESIGN **393**

Hobby PCB Vendors . . . 394
Your First PCB . . . 395
Schematic Capture . . . 395
Board Setup and Layer Navigation . . . 395
Placing Footprints . . . 396
Nets and Rough Layout . . . 396
Routing the Traces . . . 397
Design Validation . . . 398
Production Files and Process Choices . . . 399

30
HIGH-SPEED SIGNALS **401**

What's "High Speed"? . . . 401
On Picking Your Fights . . . 402
Supply Decoupling . . . 403
Current Return Paths . . . 404
Copper Pours and Ground Planes . . . 405
EMI Shielding . . . 406
Signal Reflections and Trace Length Matching . . . 407

31
ELECTRONICS THAT LAST **409**

Supply Testing . . . 409
Polarity and ESD Protection . . . 410
Cable Strain Relief . . . 410
Environmental Protection . . . 410
Serviceability . . . 411

EPILOGUE **413**

APPENDIX: TOOLS AND SUPPLIES **415**

Digital Oscilloscope . . . 415
Digital Multimeter . . . 416
Lab Power Supply . . . 417
Function Generator . . . 418
Soldering Equipment . . . 418
    Irons . . . 418
    Wire and Accessories . . . 419
Prototyping Boards . . . 420
Workholding and Wire Handling . . . 422
Sourcing Tools and Components . . . 424
Lab Safety . . . 424

INDEX **427**

INTRODUCTION



Who deserves the credit for ushering in the information age? The conventional wisdom is that we owe it to Charles Babbage, an eccentric 19th-century polymath who mused about a general-purpose computing machine. His spiritual partner in crime was Alan Turing, a tortured World War II hero who laid out the foundations of the theory of computability.

These are great stories, but I'm not sure they're right. Humanity has tinkered with calculating devices for millennia: The Antikythera mechanism—an automated astronomical instrument recovered from an ancient shipwreck—is more than 2,000 years old. The first well-documented mechanical adder, credited to Blaise Pascal, dates back to 1645. Leibniz's stepped reckoner, the earliest device for rapid multiplication and division, debuted in 1673. The first commercially successful programmable mechanism probably belongs to Joseph Marie Jacquard and his punch card-operated loom of 1804—an improvement on Jacques de Vaucanson's design from 1745.

We've long had the right ideas, but something held us back. That changed in the early 20th century with the invention of the electronic circuit, which moved data with far greater fidelity and flexibility than sprockets, cams, and pawls allowed. The unsung heroes of the revolution might be William Eccles and Frank Wilfred Jordan, who invented an early form of electronic memory: a vacuum-tube circuit that stored a single binary digit. On the heels of these inventions, Konrad Zuse, a Nazi-era German scientist who received little postwar recognition, designed the first programmable computer.

The Challenge of Learning Electronics

We're fuzzy not just on electronics history but also on what makes this technology tick. Many books claim to teach the basics; few succeed. Much of the blame goes to the hydraulic analogy, which compares circuits to plumbing pipes. The idea is seductive but soon springs leaks. By the time you reach transistors, you're building Rube Goldberg–esque contraptions that would get you laughed off any plumbing job. To wit: One reputable educator, Giovanni Organtini, insisted on modeling the transistor as a vintage pull-chain toilet with a bucket attached where the handle ought to be.

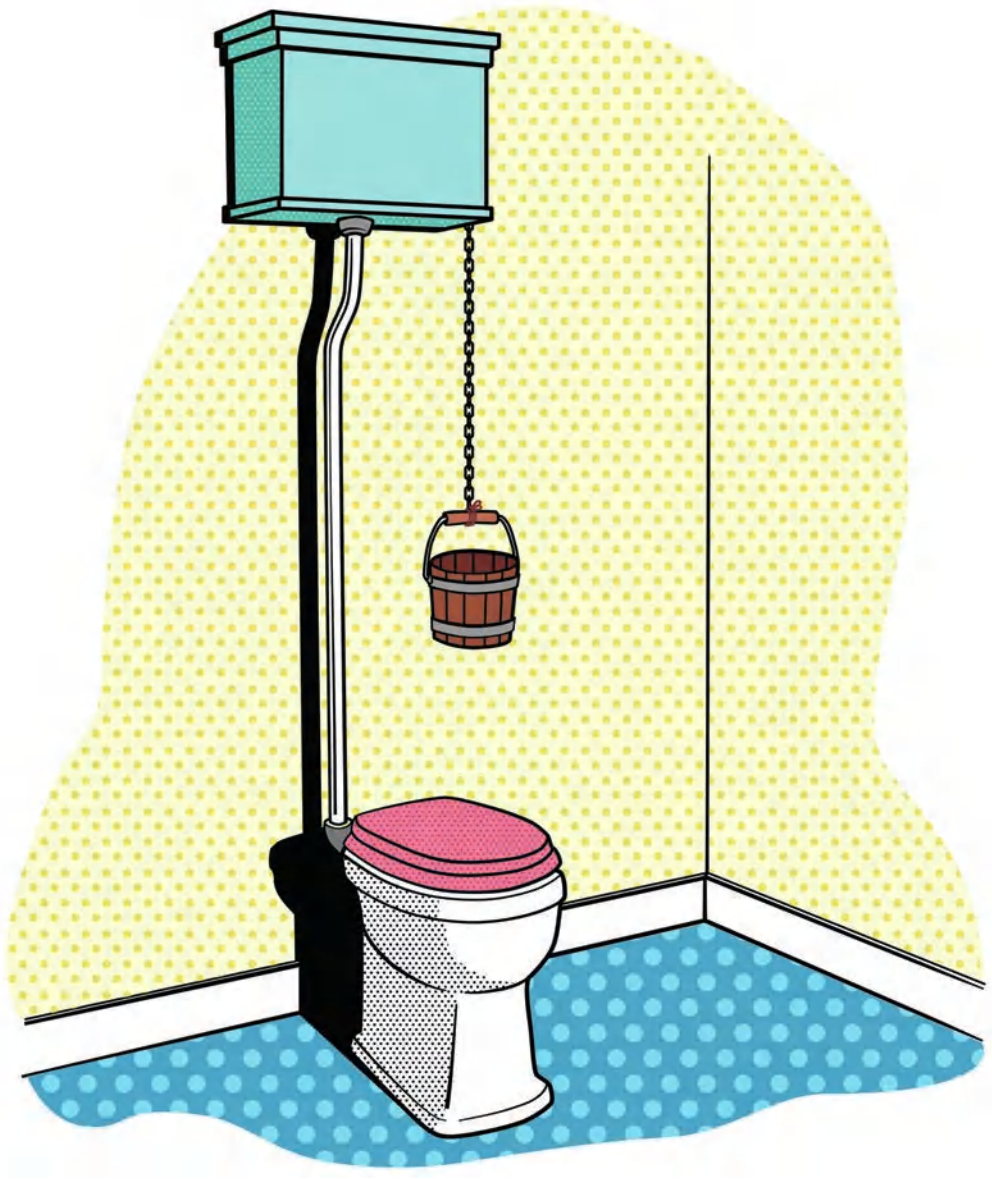
Academic textbooks offer more coherent explanations, but the prose is a tough nut for a layperson to crack. *The Art of Electronics*, the highly regarded tome by Paul Horowitz and Winfield Hill (Cambridge University Press, 2015), introduces differential equations by the end of Chapter 1. Phasor diagrams await those who make it to Chapter 2. Many of the foundational Wikipedia articles are even more challenging—they're written by electrical engineering students for other students, lacking the humor and insight of Horowitz and Hill.

The book you're holding offers a middle ground for DIYers, hackers, and other serious tinkerers from all walks of life. It's the reference I wish I had when I was learning this craft decades ago: an accessible yet accurate explanation of how electronic circuits really work. It's not a substitute for a college textbook, but it will make you fluent in the language. *The Secret Life of Circuits* avoids advanced calculus and obtuse jargon without resorting to plumbing analogies.

What's in the Book

We'll start with prerequisite theory. The chapters in Part I involve some physics and math, but we'll ease in gently—building intuition about how electricity works rather than memorizing formulas. Part II covers common electronic components. Part III puts that knowledge to work with fundamental analog circuits.

In Part IV, we'll venture into the digital realm, progressing from individual logic gates to the architecture of a simple computer. Part V dives into the world of microcontrollers and embedded programming, and Part VI shows how to turn a rough prototype into a polished project ready for Kickstarter or gifting to friends. Finally, the appendix offers up practical tips for setting up an electronics workshop and sourcing the requisite test equipment and parts.



Art based on Giovanni Organtini, "A Flush Toilet Model for the Transistor," The Physics Teacher *50, no. 4 (2012): 221.*

PART I

THE FUNDAMENTALS

1

A PRIMER ON CONDUCTION



Before jumping straight into practical electronic circuits, we need to gain solid footing with some of the underlying physics and math. To keep the book accessible, I don't want to make assumptions about the college courses you might have taken, or the Wikipedia rabbit holes you might have gone down. With that in mind, this chapter aims to tackle two questions that aren't always satisfactorily answered in high school: What is electricity, and what makes certain materials "conduct" it in the first place?

The Structure of the Atom

Most readers should be familiar with the basic structure of the atom: a nucleus consisting of neutrons and protons, surrounded by a swarm of electrons that are stratified into a number of "shells." In pop-science articles and videos, the concept is usually illustrated using the dated but intuitive model developed by Niels Bohr around 1918. The model proposed that electrons revolve around the nucleus in circular orbits, as shown in Figure 1-1.

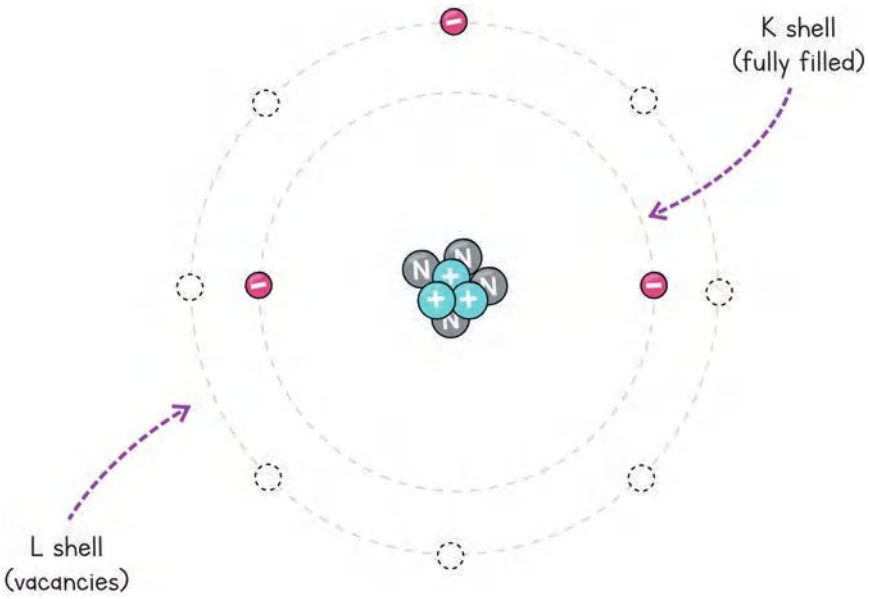


Figure 1-1: The Bohr model of lithium: three protons, four neutrons, and three electrons

Today, it's no longer believed that electrons zoom around the nucleus in circles. Rather, they appear to settle into oddly shaped orbitals that represent solutions to an underlying wave function. These patterns don't correspond to the behavior of classical, discrete particles.

If this sounds cryptic, that's because we don't have philosophically satisfying insights into the universe at subatomic scales. Instead, we have quantum mechanics: a set of equations that are good at predicting the behavior of elementary particles but that don't line up with our intuition about the macroscopic world. We don't know whether these equations are crude approximations of some deeper, yet-unseen reality or if the fabric of the universe is just weird math; the harder we try to come up with relatable analogies, the more off base we get. In any case, the shape of an orbital is of relatively little significance in electronics. It matters in chemistry because it helps model chemical reactions and predict the properties of materials, but for our purposes it suffices to say that each atom or molecule has a certain number of electrons and that these electrons are organized into a certain number of shells.

In the same vein, we don't need to dwell on the structure of the nucleus. In brief, the nucleus is held together by a powerful but extremely short-range field known as the *residual strong force*. This force is thought to be a consequence of the structure of protons and neutrons, each of which appears to be made out of three more-exotic elementary particles known as *quarks*. You don't need to pay attention to this detail: It's typically enough to say that most nuclei on Earth were formed through nuclear fusion in stars and won't undergo any change on the timescales of interest to electronics . . . or to terrestrial life.

A more pertinent observation is that the structure of the atom is also shaped by a weaker but far-reaching *electrostatic force*. This force acts only on certain subatomic particles, and those particles are said to be *charged*. In this context, the "charge" is an intrinsic property of a given type of particle and not something they can gain or lose: Protons are always positive, electrons are always negative, and neutrons don't play along. According to the electrostatic force, similar charges repel and dissimilar charges attract, so protons in the nucleus can pull in an equal number of electrons.

These electrons are then captured—*bound*—by the electrostatic field extending from the nucleus. The result is a stable atom with no net charge.

Again, the fabric of the subatomic world is largely opaque to us, so we have no intuitive explanation of what a bound electron actually does. Despite what some YouTube educators might claim, it's not orbiting the nucleus or spinning around its own axis in any conventional sense. Most simply, it just *exists* as a particular distribution of an electrostatic field in space.

Another important insight from quantum mechanics is that the parameters that describe the state of a bound electron—spin number, azimuthal number, and so on—appear to be *quantized*, meaning they only take specific values or change in discrete increments. As usual, we don't have a satisfying explanation why. We know that a specific progression of quantum numbers corresponds to standing-wave solutions of a wave function we thought of some time ago, and that this model works better than any alternatives. Regardless, the bottom line is that there are specific energy levels that our captive electron in an atom can exist at, hopping from one to another, with no option to dwell halfway between. This is why we see a progression of distinct electron shells—the quantum equivalent of Bohr's orbits—from lowest to highest energies.

The remaining conundrum is why all the electrons don't settle in the most energetically favorable, innermost shell. The relevant constraint is known as the *Pauli exclusion principle*. As noted earlier, the state of a bound electron is fully described by a handful of quantum parameters, and it appears that the electrons must remain distinct; that is, their parameters can't all be the same. This means that each shell has a limited seating capacity. Once all the combinations of "lesser" quantum numbers are exhausted, the next electron needs to bump up the remaining variable, the principal quantum number that corresponds to the shell it occupies.

The Physics of Conduction

The phenomenon of electrical conduction hinges on the behavior of *valence electrons*, the electrons in an atom's outermost shells. While the electrons that populate the interior shells do relatively little (they have low energy, are bound tightly to the nucleus, and tend to be shielded from outside influence), the story is different for the higher-energy electrons on the periphery. For example, chemistry as we know it arises from atoms exchanging or sharing these weakly attached particles. In a *covalent bond*, some electrons from both participating atoms shift to a newly formed, energetically favorable multi-atom orbital. In *ionic bonding*, one or more electrons migrate from one atom to another wholesale. In the latter case, the new molecule is held together by the resulting electrostatic field.

Even without chemical bonding, it's not difficult to temporarily dislodge some valence electrons. The most familiar example is the *triboelectric effect* that's responsible for the harmless zaps of static electricity we sometimes feel around the home. The effect isn't fully understood, but we know that, depending on the shape and orientation of molecules, different surfaces exhibit variable affinity to electrons. In part because of this, when two dissimilar materials are brought together, a greater number of electrons may randomly cross the boundary in one direction than in another. If the pieces are then quickly pulled apart, some charge carriers end up getting stuck where they don't belong.

It should be noted that the scale of this imbalance is minuscule: The displacement of about 100 billion electrons is enough to give you a good zap. This might seem like a big number. But consider that there are roughly 81 trillion times as many

electrons in a single ounce (about 28 grams) of table salt. In other words, the magnitude of the phenomenon is comparable to taking a bucket of water out of Lake Erie.

NOTE *To make this calculation, we start with the definition of a* mole*, an SI unit for measuring the quantity of a chemical substance. For historical reasons, it's defined as about* $6.022 \cdot 10^{23}$ *molecules. The mass of 1 mole of table salt is roughly 58.5 grams (2.06 ounces), so there are about* $2.9 \cdot 10^{23}$ *molecules in an ounce. Each molecule consists of a sodium atom (11 electrons) and a chlorine atom (17 electrons), netting a total of roughly* $8.1 \cdot 10^{24}$ *electrons per ounce.*

Insulators vs. Conductors

The behavior of valence electrons separates substances into two categories: *insulators* and *conductors*. In solid-state insulators, such as glass or plastic, valence electrons can still be pulled off or deposited onto the surface with relative ease, but the resulting charges stay in place and can't propagate through the bulk of the substance. This is often because there are no suitable vacancies in the shells of nearby molecules for the electron to skate across. To move around, the electron would need to be knocked into a higher energy state. We say that in an insulator, the range of electron energies associated with conduction—the so-called *conduction band*—is unoccupied; putting a sufficient number of electrons in that band is possible, but it takes so much work that the material usually catches on fire beforehand. In practical applications, such as in the sheathing for household wiring, we choose the insulating material so that this kind of electrical breakdown isn't a concern in normal use.

The behavior of charged particles is different in conductors. In metals, for example, the atoms form a dense, homogeneous lattice with sparsely populated orbitals that essentially merge together, extending throughout the material and no longer exhibiting clearly quantized energy levels that are characteristic of electrons bound to a single nucleus. In this environment, valence electrons are free to move around at will. The electrostatic interaction with protons still confines the bulk of the mobile charge carriers to the conductor, but in most other respects, the behavior can be modeled as that of a free-flowing electron gas occupying a range of energy levels that together form the conduction band.

Conduction in a Metal Wire

The key property of a conductor is that, in contrast to an insulator, any static charges deposited on the surface can quickly equalize. Consider the scenario shown in Figure 1-2, where a single electron is plucked off one end of a metal wire by an external actor and then injected back on the other end. The remaining electrons promptly scamper away due to electrostatic repulsion until the forces are equalized. If the electron-plucking process continues, we can observe a steady drift of electrons along the length of the conductor.

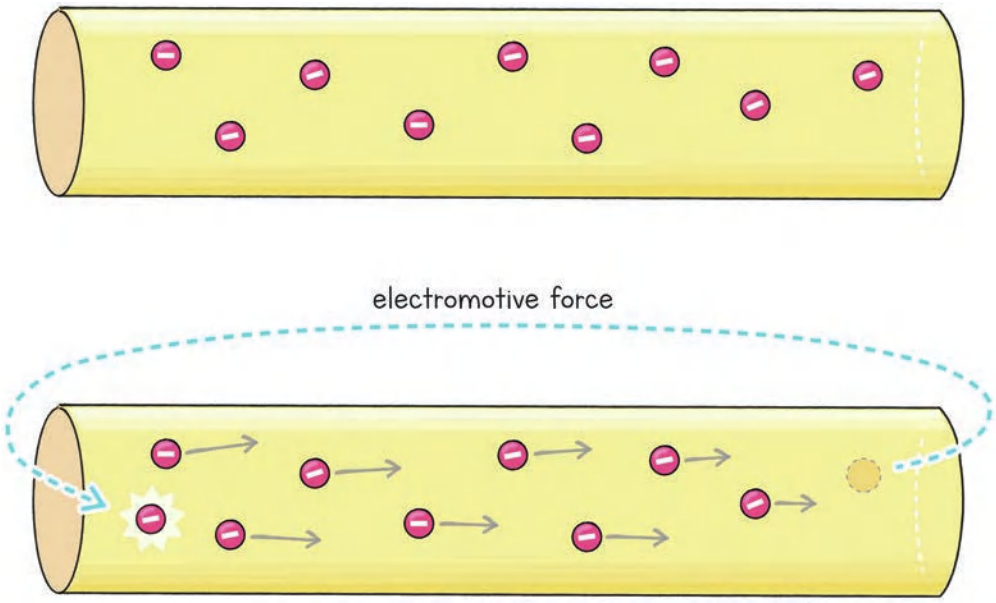


Figure 1-2: The flow of electricity in a metal wire as a consequence of an external force (the dashed line) that plucks electrons off one end and shuttles them to the other end

This motion of charges is the flow of electricity. The movement of electrons in a conductor, mediated through electrostatic fields that extend from charged particles into the surrounding space, can be harnessed to operate electronic components and perform other useful work.

It's important to note that while the charge-equalization process is fast, the drift of individual electrons is not. The electrostatic field propagates at close to the speed of light in a vacuum (about 300,000 km/s); individual electrons in a copper wire typically slither at net speeds measured in centimeters per hour or less. A reasonable analogy is the travel of sound waves in air: If you yell at someone, they'll hear you long before any single air molecule makes it from your mouth to their ear.

Other Forms of Conduction

Electrons moving through metals or metal-like substances are the dominant mode of conduction in modern electronic circuits. However, for the sake of completeness, there are several other mechanisms we should discuss in brief.

Thermionic Emission

In the mid-19th century, several experimenters noticed a phenomenon known as *thermionic emission*, whereby a material heated to a glow would lose any negative electrostatic charge imparted beforehand. The same wasn't happening if the material had a positive charge (that is, was missing some number of electrons). We now know that upon heating, electrons can absorb thermal photons or otherwise get knocked into a higher energy state. Given this modest push, excess electrons deposited on the material but not held in place by protons can free themselves and simply fly off into space, making the charge disappear.

NOTE Photons *are massless particles that carry the energy of electromagnetic radiation; they're behind radio waves, thermal (far infrared) radiation, and visible light. We'll talk about electromagnetic radiation in more detail in Chapter 6. For now, it suffices to say that when an electron moves to a higher energy level, it's often through the absorption of a photon. In the same vein, a photon is emitted when an electron returns to a lower energy state.*

The thermionic emission effect was harnessed in one of the earliest electronic components: vacuum tubes. Figure 1-3 shows the simplest arrangement, with two metal surfaces (*electrodes*) separated by a vacuum. This setup allows electrons to flow through the vacuum, but only from the heated electrode to the cold one, and not back.

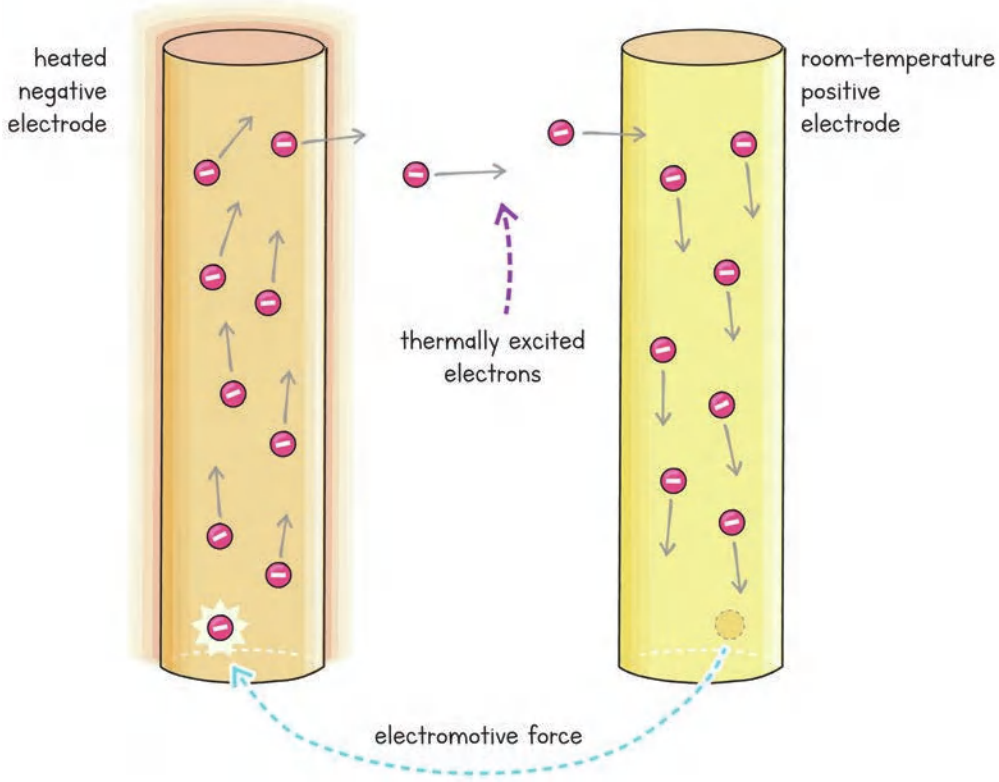


Figure 1-3: Thermionic emission in a vacuum, facilitated by an external electromotive force (the dashed line) that creates a surplus of electrons on the left-hand electrode and a deficit on the right

Thermionic emission devices aren't commonly encountered today, but they're quite interesting, so we'll take a brief stroll down that memory lane in Chapter 10.

Conduction in Plasma

A related mode of conduction is fairly prominent in gases. Gas molecules are spaced too far apart to allow electrons to hop from one atom to another, but in certain conditions—for example, at very high temperatures—electrons can be knocked into higher energy states, momentarily parting ways with their atoms and saddling those atoms with a net positive charge. These charged atoms are called *ions*.

The resulting state of matter, known as *plasma*, can also form at room temperature if a sufficiently strong external electrostatic field is acting on the gas. This external field may come, for example, from some nearby static electricity. There's always some small number of electrons that are momentarily bumped into higher energy

states due to random thermal effects; a strong field can sweep them away and accelerate them sufficiently to knock other electrons out of their valence shells in a cascade effect. This mechanism is responsible both for static zaps that jump through air and for atmospheric lightning. It's also harnessed in neon signs, which exploit the tendency for excited electrons to emit photons of visible light as they fall into lower energy states (and then become excited again by the continued application of external energy).

No matter how they're formed, plasma particles are no longer electrically neutral, so they move in response to any externally applied field. Liberated electrons drift toward the positive electrode, where they can enter the metal substrate and continue their journey in a more conventional way. Positive ions, meanwhile, travel toward the negative electrode, accept surplus electrons, and revert back to a neutral gas (Figure 1-4).

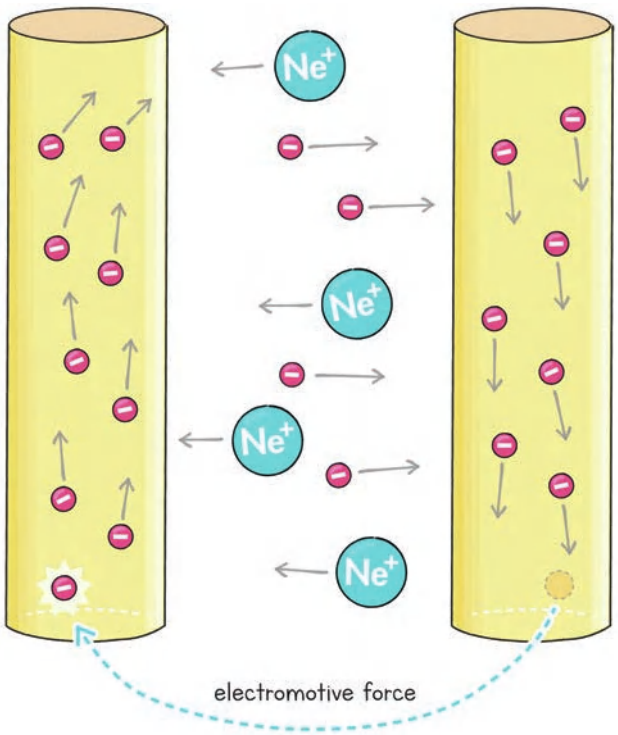


Figure 1-4: Conduction in plasma, using the example of a neon gas

Although the plasma is seemingly spent in the process, it can continually re-form from neutral gas as long as the conditions remain the same as what triggered the separation of electrons in the first place. In other words, the ionized gas can conduct indefinitely: You can keep pushing electrons onto the negative electrode and watch them emerge on the positive one.

As hinted earlier, this effect isn't limited to gases. The same kind of *electrical breakdown* into negative electrons and positive ions can be observed in nominally nonconducting solids if they're subjected to an electric field greater than what the material can handle. The process is usually quite destructive. It's what happens when trees, electronics, humans, or other objects are struck by lightning. Avoid it if you can.

Conduction in Polar Solvents

Water is a peculiar substance. Although its molecules are electrically neutral, the shape of the molecule and the geometry of its molecular orbitals make one side markedly more negative than the other. The property is seen in quite a few other substances, but the effect in water is exceptionally strong. It's part of what makes water a *universal solvent*, able to dissolve an impressively wide range of substances. In particular, the property makes water interact in an interesting way with a number of substances that contain ionic bonds, notably including a variety of salts.

As discussed earlier, ionic bonds are formed when one substance donates electrons to another, producing two nominally charged halves held together by an electrostatic field. Water molecules, being small and strongly polar, can squeeze in between and cleave ionic substances into electrically charged ions, surrounding each ion to substantially offset its electric field. A somewhat simplified diagram of the process for a molecule of copper(II) chloride ($CuCl_2$) is shown in Figure 1-5.

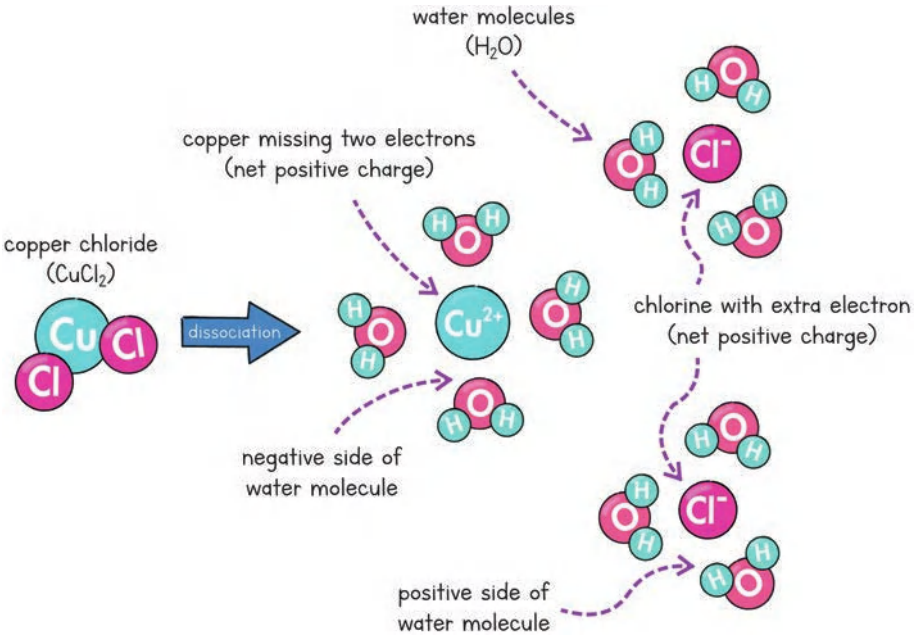


Figure 1-5: A simplified dissociation of copper(II) chloride in water

Analogously to the case of plasma, the ions are independently mobile and, still having some net charge, respond to external fields. In our example, copper ions will be attracted toward the negative electrode, where they can accept two electrons and precipitate as inert metal. Chloride ions, on the other hand, will travel toward the positive electrode, part ways with extra electrons, and turn into chlorine gas.

Because the salt is spent, the process can't continue indefinitely. However, as long as there are any remaining mobile ions, the solution remains conductive. This mode of conduction is why electricity, even at energy levels that don't support the formation of plasma, can be dangerous to living things. The liquids in our bodies are fairly conductive, and life depends on subtle electrochemical reactions that can be disrupted by an electric shock.

The motion of charges in metals and ionic liquids is essential to modern electronics, but the processes don't account for the operation of modern integrated chips. To explore the physics of these devices, we need to introduce a third category of conductive materials, coming up in Chapter 2.

2

THE CURIOUS CASE OF SEMICONDUCTORS



In Chapter 1, we identified two fundamental types of materials: insulators and conductors. In conductors, there is an abundance of mobile charge carriers that can move around in response to electrostatic fields; in insulators, there are none. What I didn't mention is that in between these extremes, there exists a third category of substances: *semiconductors*. As we'll see in this chapter, their properties are more remarkable than the name implies.

Properties of Elemental Semiconductors

Semiconducting elements can be found on the right side of the periodic table, clustered between metals such as aluminum and tin and nonmetals such as sulfur and iodine. By far the best-known example of a semiconductor is silicon. At first glance, silicon resembles metal. It has a lustrous, silver-gray appearance and exhibits good thermal conductivity. At the same time, the substance is less dense and more brittle than most true metals. Just as important, in its pure form, silicon doesn't conduct electricity particularly well.

Recall that in insulators, valence electrons—the electrons with energies associated with the material's *valence band*—can't easily move throughout the bulk of the material. This is often because there are no easily reachable vacancies in the valence shells of nearby atoms. There's an unpopulated, higher-energy conduction band that would allow electrons to move around with ease, but the gap in energy levels is big enough that it can be cleared only at high temperatures or in the presence of very powerful

electrostatic fields. Such conditions can be enough to destroy the insulator or permanently alter its properties.

In pure semiconductors, there's a similar band gap, but the difference in energy levels between valence and conduction bands is small. In fact, the delta is modest enough that, at any given time, thermal interactions at room temperature are enough to knock a fair number of electrons into the conduction band. After a brief moment—typically nanoseconds to milliseconds—the excited electron returns to a lower energy state. However, before this happens, the charged particle can drift a short distance, so a form of stop-and-go conduction can occur. Just as important, because the momentary excitation leaves a vacancy (a *hole*) in the valence shell, the added slack makes it possible for lower-energy valence electrons to also slither around. Still, the magnitude of these effects is fairly weak. In pure silicon at ambient temperature, the conductivity is about 100 billion times worse than in a chunk of copper of the same size.

Due to their poor electrical properties, "vanilla" (*intrinsic*) semiconductors are fairly uninteresting, except for two minor details. First, because charge carriers can be generated by the absorption of external photons falling onto the surface, semiconductors in transparent packaging can be used to measure incident light. Second, because the generation of charge carriers is also driven by thermal effects, the conductivity of a semiconductor sensor will increase in proportion to temperature. For completeness, we should note that, in metals, the opposite is true: Thermal vibrations of the atomic lattice can hinder conduction. But for this to be noticeable, the conductor must be quite hot.

Metals are just metals. For semiconductors, the story changes with the addition of impurities known as *dopants* that alter the performance of these materials in remarkable ways.

Doped Semiconductors

There are two types of dopants that can be added to semiconductors: n-type and p-type. An *n-type* dopant, such as phosphorus, contributes additional conduction band electrons to the semiconductor lattice. Crucially, these electrons depart from shell positions associated with energy levels higher than in the base material, so they don't leave energetically favorable vacancies behind; they're bound to roam for life. More precisely, they can still settle into the short-lived, thermally induced vacancies in the base semiconductor, but this strands another electron in the conduction band—an atomic game of musical chairs. Whichever electron loses, you end up with an abundance of long-lived mobile charge carriers, so electricity can flow with ease.

In contrast, a *p-type* dopant, such as boron, has a higher number of energetically favorable valence-shell vacancies than the base semiconductor. This creates extra slack and allows other valence electrons to hop from one atom to another without needing to be first knocked into a higher energy state. As the sea of electrons flows in one direction in response to some external force, the resulting vacancies seemingly migrate in the opposite direction (Figure 2-1). Because of this apparent motion, we sometimes talk about semiconductor holes as if they were actual, positively charged particles.

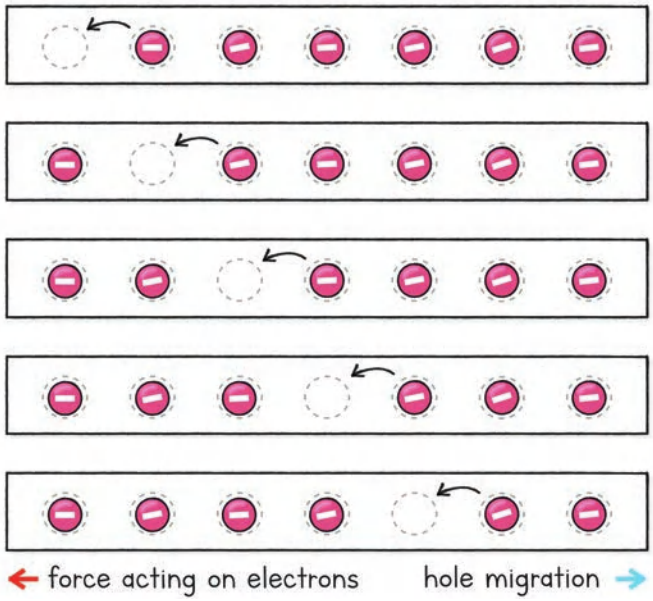


Figure 2-1: The apparent motion of holes (in the direction opposite the drift of valence-band electrons)

However you approach it, both n-type and p-type materials conduct much better than pure silicon. The plot further thickens when you stack these differently doped semiconductors to construct a single electronic device.

The p–n Junction

When a p-doped semiconductor is brought into contact with an n-doped one, an interesting phenomenon takes place. In the immediate vicinity of the junction, higher-energy conduction electrons from the n-side randomly diffuse across the boundary and then promptly fall into the adjacent, abundant valence holes on the p-side. This creates what's known as a *depletion region*: an exceedingly thin layer that has virtually all the holes plugged on the p-side and few conduction band electrons on the n-side (Figure 2-2).

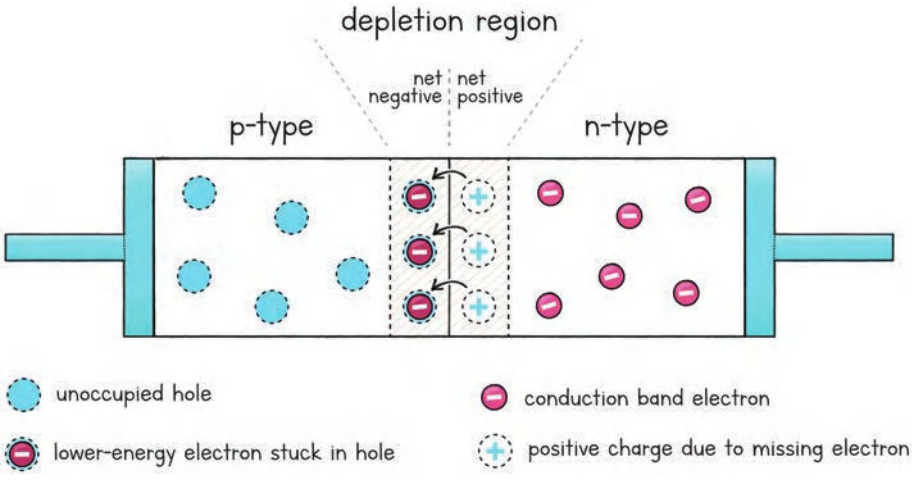


Figure 2-2: A p–n semiconductor junction

Internally, the depletion region is no longer electrically neutral, with a net negative charge on the p-side (due to extra electrons) and a net positive charge on the n-side (due to orphaned, electron-deficient atoms). This charge separation produces an electrostatic field that's wholly confined to the depleted region. The field moves any mobile, negatively charged particles within its reach from left to right and any free-roaming, positively charged particles from right to left. This action makes the junction largely impassable to any additional *majority charge carriers*—namely, electrons coming from the n-side and holes (treated as positive particles) trying to diffuse from the p-side. In both cases, the particles are swiftly ejected back to where they came from.

In this model, it's still possible for the odd *minority carrier* to cross the boundary. For example, just like in an intrinsic semiconductor, a conduction band electron can be thermally generated in a p-doped material. If this minority carrier then wanders into the depletion region before falling back into the valence shell, the field will toss it over to the n-side. Even so, because the density of minority charge carriers is comparatively low and their lifetimes in doped material are short, the effect is more or less negligible.

If we take another look at Figure 2-2, it should be evident that removing electrons from the right terminal of the device and adding them to the left terminal would not accomplish much. It would make the p-side more negative and the n-side more positive, strengthening the existing electrostatic field. Without a way for the majority carriers to cross the p–n boundary, more electrons would get stuck near the junction vis-à-vis positively charged atoms on the other side. In effect, the width of the depletion layer would grow.

The situation changes if we start moving electrons the other way around. Making the n-side more negative and the p-side more positive gradually counters the p–n junction's internal field, eventually offsetting it enough to get the p-side electrons unstuck and allow more n-side electrons to pour in. Past a certain point, our p–n junction, known as a *diode*, starts conducting electricity remarkably well, but only in one direction.

A Sneak Peek at Field-Effect Transistors

The basic p–n junction has some uses in electronic circuits, but it's more valuable as a building block for more complex components, such as transistors. We'll take a more thorough look at transistors and other semiconducting devices in Chapter 10, but it's good to tease the adventure ahead while the knowledge about p–n junctions is still fresh.

Let's start with a *junction field-effect transistor (JFET)*. The component, shown in Figure 2-3, has three electrical terminals. The conductive path through the device is formed by a homogeneous n-type semiconductor that stretches between two primary terminals (left and right). The third terminal is connected to a p-type "wedge" in the middle. The interaction between p-type and n-type materials forms a depletion region surrounding the wedge, so there's normally no conduction between the primary terminals and the wedge.

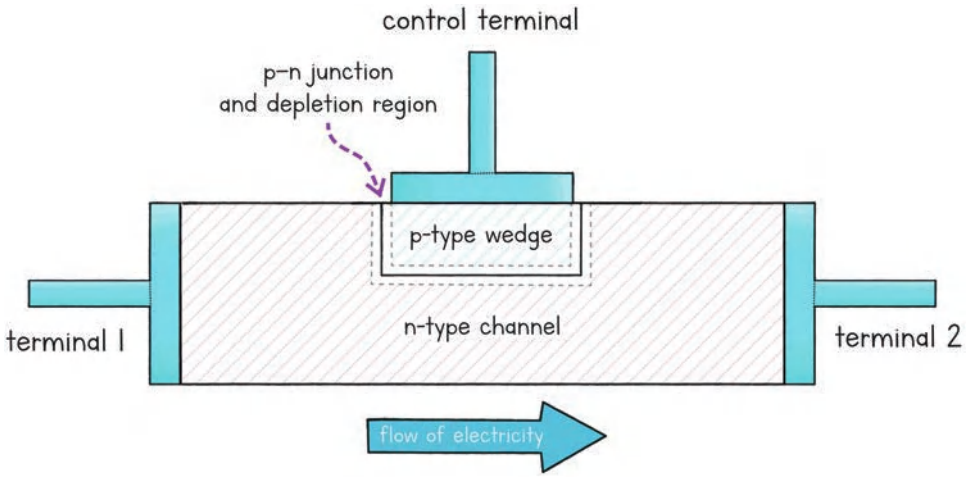


Figure 2-3: A simplified illustration of a JFET

We could shuffle some electrons around to make the p-type wedge positive in relation to the n-type body, squishing the depletion layer and letting some electricity flow to the side. But that's not how JFETs are used. Instead, the idea is to make the wedge more negative, thus widening the depletion layer to constrict (or choke off) the flow of electricity through the main path (Figure 2-4). In effect, a JFET is an electrostatically controlled gate.

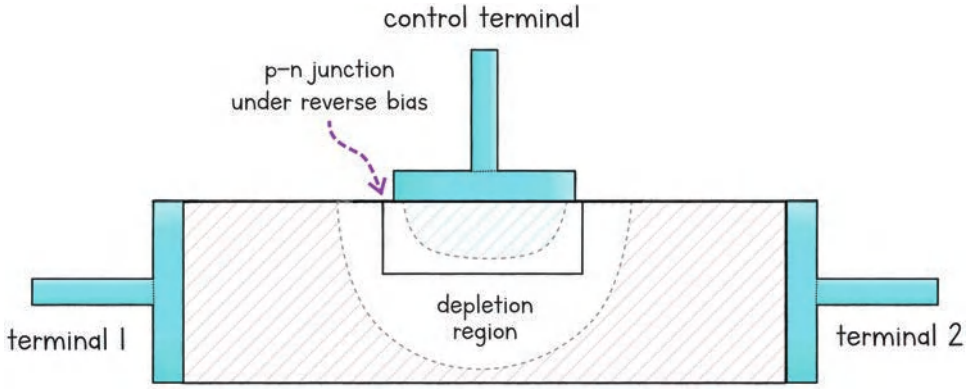


Figure 2-4: A JFET with a negatively charged control terminal, resulting in a wide depletion region that nearly cuts off the primary conductive path

A similarly creative use of p–n junctions is the *metal-oxide-semiconductor field-effect transistor (MOSFET)*. The "oxide" in the name refers to any insulating material; a common choice is glass (silicon dioxide). The overall structure of a simple MOSFET is shown in Figure 2-5.

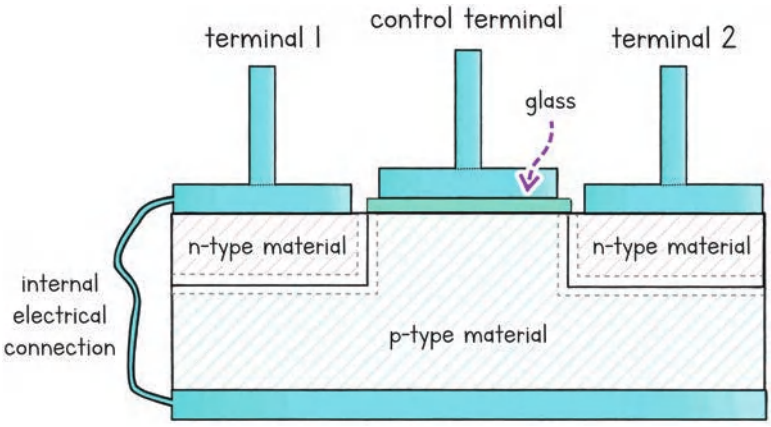


Figure 2-5: A simplified layout of a MOSFET

Similar to JFETs, MOSFETs are designed to modulate the flow of electricity between two primary terminals. Unlike in a JFET, the transistor is normally off. More specifically, electrons can't flow from the terminal on the left to the terminal on the right. The p–n junction on the left doesn't get in the way; it's bypassed altogether through a separate electrical connection to the substrate. But even with that junction out of the picture, you're still putting electrons on the p-side of the second junction near the right-hand terminal. Similar to the case of a diode, this action only serves to widen the depletion region and prevents majority charge carriers from ever getting across.

The trick that makes MOSFETs work is that we can pull some electrons off the control terminal. The resulting electrostatic field strongly attracts any mobile electrons on the other side of the insulator. They can't get across to the control terminal, so they get smushed against the glass. If the field is strong enough, this produces an n-like, electron-rich conductive channel that bridges the n-sides and completes the circuit across the device (Figure 2-6).

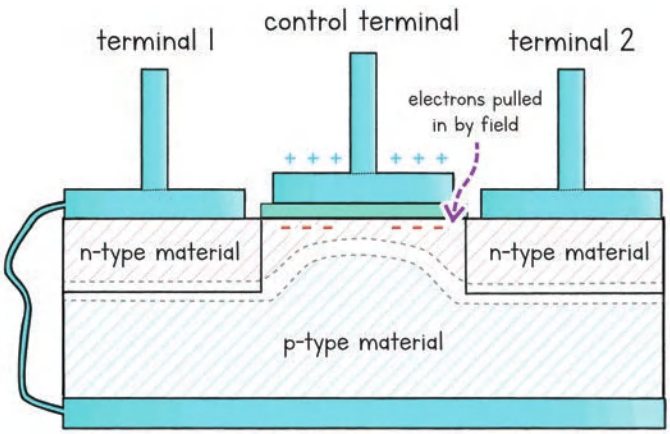


Figure 2-6: A MOSFET with a conductive inversion layer formed vis-à-vis the control terminal

Again, we'll have a closer look at transistors and their applications down the line. For now, the important point is that p–n junctions can be manipulated by electrostatic fields, making them incredibly useful in electronics—and nearly impossible to explain in terms of hydraulic analogies.

Now that we've covered the primary physical phenomena that make electronic circuits work, we're ready to look at the ways to measure the magnitude of these effects.

3

TIME-INVARIANT CIRCUIT CHARACTERISTICS



The physics of conduction is one of the most significant humps in learning electronics. The good news is that we're mostly done with it. We still have some loose ends to tie up in Chapter 6, but in the meantime, we can discuss some of the simple terminology used to describe what's going on in electronic circuits on a macro scale.

We'll start in this chapter with some fundamental parameters that can be described as *time-invariant*—that is, they aren't affected by the passage of time. That's not to say the quantities outlined here can never change. They will if you flip a switch or spin a dial. The basic distinction is that their effects aren't fundamentally dependent on how much time has passed since we turned the circuit on.

Charge

Charge (Q) is the term used to measure the quantity of charge carriers, usually electrons, moving through a wire or lurking on a surface to give you a static zap. It's fairly unwieldy to speak in terms of individual electrons, so we measure the magnitude of charge in *coulombs (C)*. The coulomb isn't a particularly exciting unit; it's just a measure of quantity, with 1 C defined as the equivalent of roughly $6.24 \cdot 10^{18}$ (6.24 quintillion) electrons.

Coulombs aren't a common sight in discussions of practical electronic circuits. Instead, they're used as a stepping stone to define several other units that we rely on day to day. Still, the underlying concept is worth knowing because it lets us decipher certain less obvious relationships—for example, between the current supplied to a capacitor and the resulting voltage.

You should be aware of how the unit of charge relates to other electronic quantities, but you don't need to memorize the underlying number ($6.24 \cdot 10^{18}$). It's just one of several "fudge factors" in the International System of Units. It allows concepts such as volts and amperes to map more elegantly to other SI quantities, such as meters or kilograms.

Current

Current (I) is the measure of the flow of electricity over time. Its unit, the *ampere (A)*, is defined as 1 coulomb's worth of charged particles passing per second through a given spot. We can formalize this relationship as the following formula:

$$I = \frac{Q}{t}$$

The meaning of "a given spot" in the definition may seem a bit hazy in the context of a three-dimensional conductor. A simple mental model is to imagine a measurement plane intersecting the wire, as shown in Figure 3-1.

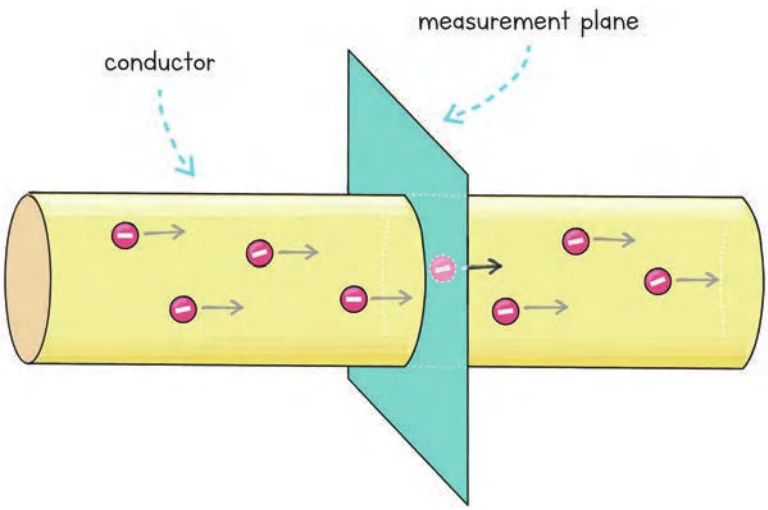


Figure 3-1: Counting charges in a three-dimensional wire

In practice, slicing conductors to look for electrons isn't feasible. Instead, we measure the motion of charges by observing a variety of indirect effects, some of which will be highlighted later in the chapter. Still, as a conceptual approximation of the measuring process, the plane method will do.

Sooner or later, you'll undoubtedly encounter a similar-sounding unit: the *ampere-hour (Ah)*. This is not the unit of current! One ampere-hour is just a way to describe the amount of charge delivered by the flow of 1 ampere over the course of 1 hour (3,600 seconds). Rather unsurprisingly, this works out to 3,600 coulombs. The unit is used, for example, to measure the capacity of batteries. I'd wager it exists mostly because no one remembers what a coulomb is.

Of course, you're better than that. You know that 1 coulomb corresponds to $6.24 \cdot 10^{18}$ electrons. Being a well-read individual, you can even calculate the "coulomb charge" of a single electron:

$$e \approx \frac{1\,\mathrm{C}}{6.24 \cdot 10^{18}} \approx 1.6 \cdot 10^{-19}\,\mathrm{C}$$

With this trick up your sleeve, you can impress your friends by pointing out that moving one electron per second equals a current of 160 zeptoamps. If they ask, explain that *zepto-* is an official SI prefix for "not a whole lot." More practically, Table 3-1 lists some useful SI prefixes that we'll be using in the book.

Table 3-1: Useful SI Prefixes

Full prefix	Short prefix	Multiplier	Plain-language description
Tera-	T	$\times 10^{12}$	Trillion
Giga-	G	$\times 10^{9}$	Billion
Mega-	M	$\times 10^{6}$ (1,000,000)	Million
Kilo-	k	$\times 10^{3}$ (1,000)	Thousand
Milli-	m	$\times 10^{-3}$ (0.001)	One thousandth
Micro-	μ	$\times 10^{-6}$ (0.000001)	One millionth
Nano-	n	$\times 10^{-9}$	One billionth
Pico-	p	$\times 10^{-12}$	One trillionth
Femto-	f	$\times 10^{-15}$	One quadrillionth

A note for US readers: The *nano-* prefix is shunned by some electrical engineers in the country, leading to oddball practices such as describing a 47 nF capacitor as 0.047 μF or 47,000 pF. You can follow suit, but there's no good reason to.

Energy and Power

In physics, *energy (E)* can be thought of as the measure of work. The unit of energy, the *joule (J)*, is roughly equivalent to the effort of lifting a 100-gram (3.5-ounce) object to the height of 1 meter (3.3 feet). Equivalently, there are about 5 megajoules (MJ) of stored energy in a large pizza, and somewhere in the vicinity of 120 MJ of heat will be released if you burn a gallon of gas.

The reason I'm describing the joule in pizza terms is that the precise definition of the unit usually isn't important in electronics. We need the joule to define voltage, but the only insight we need day to day is that a joule represents a fixed and modest amount of work.

A more common energy-related concept is *power (P)*, which is measured in *watts (W)*. Power is the rate at which work is performed, somewhat akin to how speed is the rate at which you travel the distance from point *A* to point *B*. Power is calculated by dividing the amount of expended energy by the elapsed time:

$$P = \frac{E}{t}$$

A car moving at 60 mph will travel a shorter distance in 1 hour than it will in 6 hours. However, in both cases, the rate of travel—the speed—remains the same during the trip. In the same vein, an electric space heater that delivers 1,000 watts will use less energy if plugged in for an hour than if you leave it running for a day. Yet, in both cases, it's delivering the same heating power while powered on.

Similar to the case of ampere-hours, you might also encounter a seemingly power-related unit known as the *kilowatt-hour (kWh)*; it's particularly common on utility bills. Once again, the unit isn't what it seems. Instead of measuring the rate of work, it describes the total amount of delivered energy. It's the equivalent of a 1 kW space heater that's plugged in for 1 hour (3,600 seconds). In other words, it's the same as 3.6 MJ.

Voltage

In the absence of an external field, a lone electron in a wire may randomly wander around, but it's under no compulsion to go anywhere in particular. The addition of an electrostatic field upsets this balance, pushing it along a particular path. *Voltage (V)* is a way to measure this phenomenon.

In Chapter 1, we talked about an invisible being that plucked electrons off one end of a wire and moved them to the other end. In this gremlin-based model of conduction, voltage is the maximum amount of energy that the gremlin is willing to expend to move a single electron. Alternatively, when dealing with static charges, it's the pressure difference in the electron gas that would cause current to flow if we (or the gremlin?) dropped a plumbing wrench across two points in the circuit and caused an electrical short. Different voltages—that is, different electromotive forces—may appear between different locations, so we need to be unambiguous about how we're performing the measurement.

As a matter of practicality, we usually define the unit of voltage—the *volt (V)*—not in terms of individual electrons but as the total expended energy (E) in joules in proportion to the amount of moved charge (Q) in coulombs:

$$V = \frac{E}{Q}$$

To illustrate one of the finer points about voltage, imagine two metal plates, one big and one small, each with 8 extra electrons of static electricity. On both sides, the charge is the same, but when you bridge the plates, there will be an electromotive force (a voltage) that causes electrons to flow from the small plate to the big one. This is because the electrons on the small side are cramped more tightly, so the forces of electrostatic repulsion are stronger. The equilibrium—the point of zero voltage—is an equal charge *pressure* across both plates, not an equal *amount* of charge per se (Figure 3-2). This observation underscores why the measurement always must be made in relation to another well-defined point.

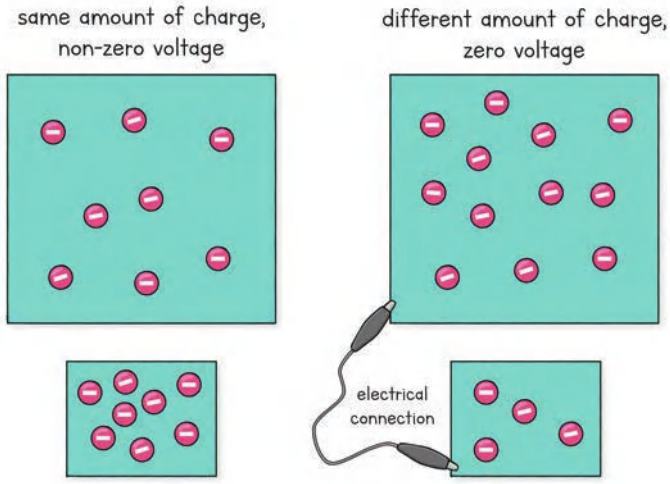


Figure 3-2: The equalization of electron gas pressure across conductive plates

Again, all that voltage is measuring is the magnitude of push given to each charged particle. At high voltages, the particles are more motivated to leave, so they find it easier to get over any humps in the road, such as insulators, human bodies, and so forth. Although the volt is defined in terms of a joule, the precise relationship is seldom important. Most of the time, it suffices to remember that 10 volts make a tiny spark, while 10 kilovolts make a decent-sized one.

To be fair, sometimes we want to calculate how much power is dissipated by an electrical device that's powered by a given voltage and has a known current flowing through. In this case, the relationship between volts and joules is useful to know. Let's start by recalling and rearranging the fundamental formulas for voltage and current:

$$\text{Definition of voltage}: \quad V = \frac{E}{Q}$$

$$\text{Solving for energy } (E): \quad E = V \cdot Q$$

$$\text{Definition of current}: \quad I = \frac{Q}{t}$$

$$\text{Solving for charge } (Q): \quad Q = I \cdot t$$

We can now plug the second equation into the first one. The result is a formula that relates energy consumption to voltage, current, and time:

$$E = V \cdot I \cdot t$$

Further, recall that the unit of power, 1 watt, is equal to 1 joule of energy per second. If we plug this energy calculation into the formula for power, we get the rate of energy consumption independent of time:

$$\text{Power equation}: \quad P = \frac{E}{t}$$

$$\text{Substituting } E: \quad P = \frac{V \cdot I \cdot \cancel{t}}{\cancel{t}} = V \cdot I$$

The final equation is $P = V \cdot I$. It tells us that the power delivered by electricity to any kind of an appliance is a function of the supplied voltage, the current admitted by the device, and nothing else. The formula is the same no matter if the energy is converted into motion, turned into light, or used to mine cryptocurrency.

Resistance

Resistance (R) is the steady, time-invariant opposition to the flow of current. In all common materials, the motion of charge carriers is impeded to some extent. The magnitude of the effect depends on the material's structure, cross-section, and length. In all cases, the wasted energy is radiated away as heat.

The resistance of reasonably sized copper conductors is typically small enough that we can neglect the effect, although some caution needs to be exercised when dealing with very thin wires, very long cabling, or very high currents. When we actually need to impede currents in a controlled way, we turn to specialized components known as *resistors*. These devices are typically made from carbon or from an exceedingly thin metal film.

The unit of resistance is the *ohm* (Ω). It can be defined as the behavior of a resistor that, when subjected to an electromotive force of 1 volt across its terminals, allows 1 ampere to trickle through. This definition is worth dissecting a bit more.

Ohm's Law

The aforementioned relationship between resistance, the applied voltage, and the resulting current through a resistor is given by a formula known as *Ohm's law*:

$$R = \frac{V}{I}$$

In most materials, resistance remains constant throughout a wide range of operating conditions. Still, as hinted in Chapter 2, it's possible to make electronic devices in which the parameter varies depending on factors such as temperature, incident illumination, or applied voltage. These devices are not called resistors, but they can be described as having a (situation-dependent) resistance.

The Ohm's law equation can be rearranged in a couple of useful ways. For example, if we have a well-behaved resistor of a known value, and if we can quantify the voltage across its terminals, we can tell what current is flowing through:

$$I = \frac{V}{R}$$

This trick is useful for circuit measurements. As discussed earlier, it's hard to count individual electrons, but it's easy to route electricity through a small resistor, probe the resulting voltage across its terminals, and then solve for I.

The third form of Ohm's law helps us gain a bit more intuition about volts. It tells us that if we have a current of 1 ampere flowing through a 1 Ω resistor, the voltage across the terminals will be 1 V:

$$V = I \cdot R$$

The three arrangements of Ohm's law are the only formulas in this chapter that are worth committing to long-term memory.

Joule's Law

As noted earlier, resistors dissipate energy as heat. If we're not paying attention, they might even overheat and let the magic smoke out.

To calculate the rate of heat dissipation, we can combine the equation for electrical power ($P = V \cdot I$) with Ohm's law as solved for V or I. For example, if we use Ohm's law to substitute I, we get the following result:

$$\text{Formula for power : } P = V \cdot I$$

$$\text{From Ohm's law : } I = \frac{V}{R}$$

$$\text{Combined result : } P = V \cdot \frac{V}{R} = \frac{V^2}{R}$$

This is known as *Joule's law*. In addition to predicting heat output without needing to know the current flowing through the resistor, this formula offers another informal way to think about the unit of voltage: 1 volt, if applied to a 1 Ω resistor, would dissipate 1 watt of heat.

The equation can also be solved by using Ohm's law to substitute V:

$$\text{Formula for power : } P = V \cdot I$$

$$\text{From Ohm's law : } V = I \cdot R$$

$$\text{Combined result : } P = (I \cdot R) \cdot I = I^2 \cdot R$$

In both of the resulting power formulas, one of the terms (V or I) is squared. This is significant. It means that a small increase in voltage or current flowing through a resistance can produce a steep increase in temperature. In fact, if you have an assortment of standard resistors and a lab voltage supply, there's a neat experiment you can perform. Find a 10 Ω resistor and connect it across the terminals of your power supply. Dial in a voltage of 2.5 V, set the maximum supply current to 3 A or more, then enable output. The current will read 250 mA, just as foretold by Ohm's law, and the resistor will get slightly warm to the touch.

Next, crank up the voltage to 5 V. The current will increase to 500 mA—again, in line with the formula—but now the resistor should be sizzling hot. Turn the dial all the way to 10 V, and the component will probably start glowing red (Figure 3-3).



Figure 3-3: The glowing-resistor experiment

When done, be sure to toss out the resistor. Even if it survived, it's probably damaged and badly out of spec.

The Question of Signs

Before we wrap up the chapter, we should touch on the mildly confusing matter of the signs we put in front of numbers representing current or voltage. The approach we take boils down to convention more than to any steadfast rule.

Let's start with voltage. As emphasized earlier, the quantity is always measured between two points in the circuit. If one of the endpoints isn't specified, it's usually assumed to be some common reference point, often the negative terminal of the power supply. This common reference may also be designated as "0 V" or "ground."

A device used to measure voltages, a *voltmeter*, has two leads: a black one for the reference point and a red one for the relative measurement. The device will display a positive number if it detects an electromotive force that tries to move electrons from the reference side toward the red lead. If the electromotive force is acting in the opposite direction, the meter will report a negative value.

As a practical example, if you connect the black lead to the "minus" side of an AA battery and the red lead to the "plus" side, you'll get a reading of roughly +1.5 V. If you reverse the leads, the meter will show −1.5 V.

The situation is more interesting for current. For historical reasons, we describe the behavior of electronic circuits in terms of an imagined *conventional current*, as if the charges flowed from the positive side to the negative side. This is precisely the opposite of the actual motion of electrons in a metal wire. We guessed the direction wrong about 150 years ago, and since then, no one has wanted to rock the boat.

Thankfully, we usually write currents without a sign. It's just the net number of charges moving per unit of time. If it's important to clearly indicate where the charges are going, we put an arrow on the schematic indicating the direction of conventional current. A negative current reading might still be used in the narrative for the circuit, but only if we want to clearly indicate that the direction of flow has changed compared to some earlier situation discussed moments before.

But then, a digital current meter, an *ammeter*, displays readings with a sign! It will report a positive reading if it's inserted into the circuit so that conventional current is flowing through the device from the red lead to the black one. It will give a negative reading if the electricity is flowing the other way around. This can be informative, but

as noted earlier, when describing the measurements to other humans, you'd normally omit the sign.

Figure 3-4 shows the correct way of using a multimeter in the voltage measurement mode (left) and the current measurement mode (right). Note that, in the latter case, the meter must be placed in the path of the current. For this measurement mode, its internal circuitry is switched so as not to hinder the flow of electricity, and the red lead might need to be moved to another port. It follows that if the current is not controlled in some way and is allowed to exceed the meter's specifications, it's possible to blow an internal fuse.

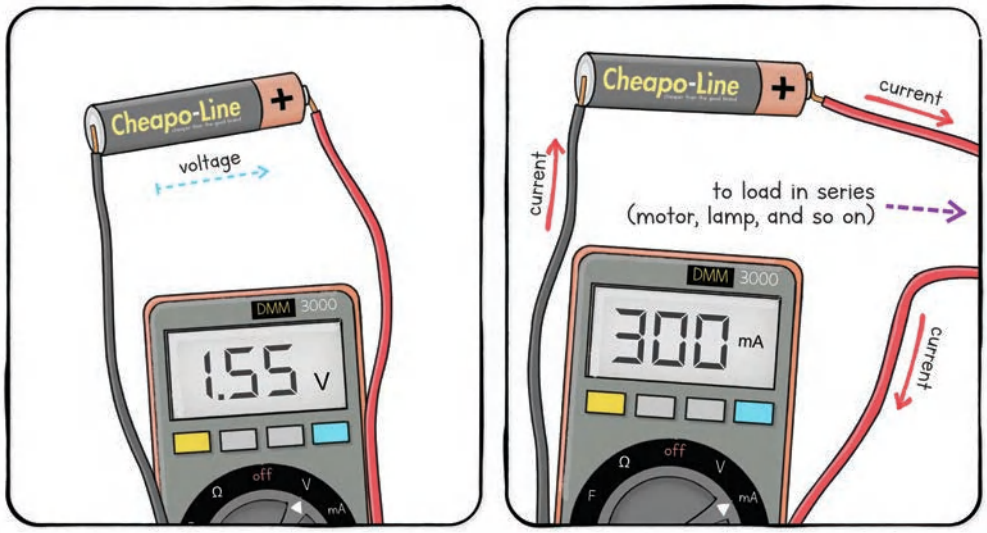


Figure 3-4: Measuring voltage and current in an electronic circuit

If you have a multimeter, see if you can make some measurements of batteries. If you don't and need advice on workshop gear, flip over to the appendix at the end of the book. And if you're all set, strap in for a discussion of circuit dynamics in Chapter 4. It will be an intense journey, but the effort we need to put in will begin to pay dividends in no time.

4

SIGNALS THAT CHANGE OVER TIME



In Chapter 3, we established a couple of methods for thinking about electronics in their steady state. Now it's time to lay the groundwork for analyzing change. Many modern-day electronic devices are in a state of constant flux. The change can come in the form of AC power, an audio signal picked up by a microphone, or a rapid-fire data transmission between two digital chips. In this chapter, we'll explore how to think about these sorts of time-varying signals and develop mathematical tools for predicting circuit behavior down the line.

In mathematics, the study of change is known as *calculus*. The subset of calculus that's useful for electronics isn't hard, but the formulas take years to master and just months to forget. Worse, they're seldom derived from first principles, leaving students with a toolkit that works when they need it but that doesn't meaningfully explain reality.

In this book, we'll buck the trend and talk about circuit dynamics without leaning on any prior college coursework. We'll be taking a slightly longer but more scenic route. We won't be able to escape math entirely, and we must front-load the discussion with some theory, but all you need to follow along is basic algebra and a smidge of trigonometry. First, though, let's talk about what an electronic signal actually is.

What Is a Signal?

When we talk about *signals* in electronic circuits, we usually mean encoding information by varying voltages. As noted in Chapter 3, voltage must be measured between two points in the circuit. For most signals, the reference point will be the circuit's so-called ground rail. This is typically the negative terminal of a battery or the negative lead of an external power supply.

A simple voltage-based signal, a square wave, is shown in Figure 4-1.

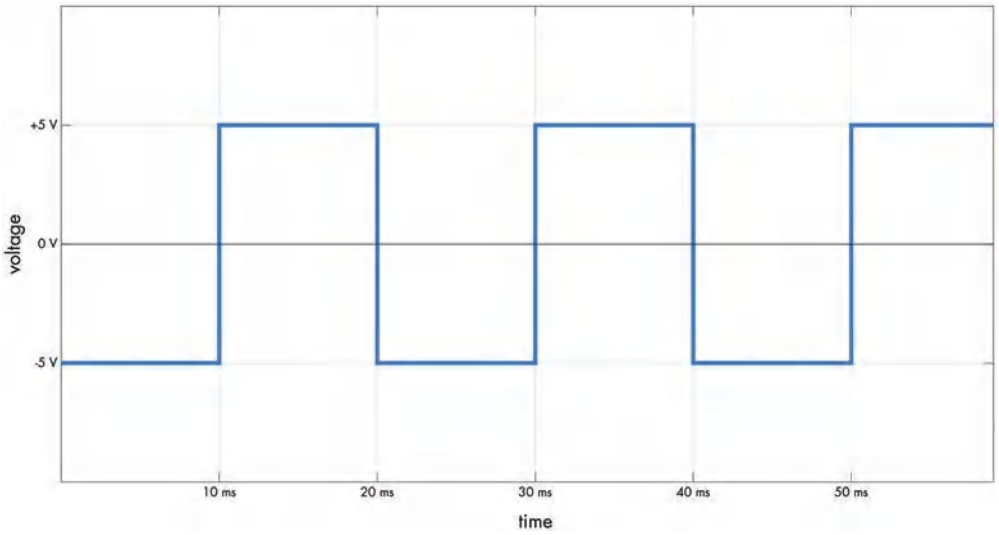


Figure 4-1: An example of a square wave signal centered around 0 V

Naturally, voltages and currents are intertwined. If there's some sort of resistive load on the receiving side, the voltage will rise only if a sufficient current is supplied to the load. Even in an open circuit, some electrons need to be shuffled to adjust the voltage seen on the other end of a wire or a printed circuit board (PCB) trace. Still, when it comes to contemporary electronics, we speak in terms of voltages because they're easier to measure and control with semiconductor devices. I'll talk about the reasons for this in Chapter 10. For now, assume that unless indicated otherwise, signals involve precisely trimmed voltage levels and then supplying just enough current to get that reading where it needs to be. Exceptions happen—for example, some industrial systems may use current-based signaling—but this is sporadic and almost always spelled out.

When referring to voltages rather than currents, we still tend to describe changing signals as *alternating current (AC)* and steady signals as *direct current (DC)*. This is just one of several charming little throwbacks to the early days of electrical engineering. If you want to emphasize that you're talking specifically about voltage-based signaling, *AC voltage* and *DC voltage* are acceptable. "Alternating voltage" is clear too, but no one says "direct voltage."

Many of the analytical tools I'll introduce in this chapter are meant for periodic waveforms that consist of a short pattern repeating at constant intervals. For the square wave in Figure 4-1, the repeated unit is a sequence of two voltages: −5 V and +5 V. Real-world signals usually aren't perfectly periodic. For example, computer transmissions that encode each bit as one of two voltage levels will typically contain irregular runs of zeros and ones, depending on the data being sent:



Even though a periodic signal isn't a drop-in substitute for this irregular waveform, it serves as a simplified test case for analyzing the circuitry. It represents the highest rate of change: a densely packed, alternating pattern of zeros and ones. If the device can handle that, it should be able to deal with transmissions that contain fewer signal edges too.

Frequency and Amplitude Measurements

To describe a periodic waveform, the first parameter we usually look at is the frequency with which the unit pattern repeats. We express this in *hertz (Hz)*. One hertz is one repetition per second, 2 hertz is two cycles per second, and so on. Alternatively, we can specify the waveform's *period*, the duration of a single cycle. Frequency and period are reciprocal quantities. The waveform shown in Figure 4-2 has a period of 20 milliseconds (0.02 second), so the frequency is 1 / (0.02 s) = 50 Hz.

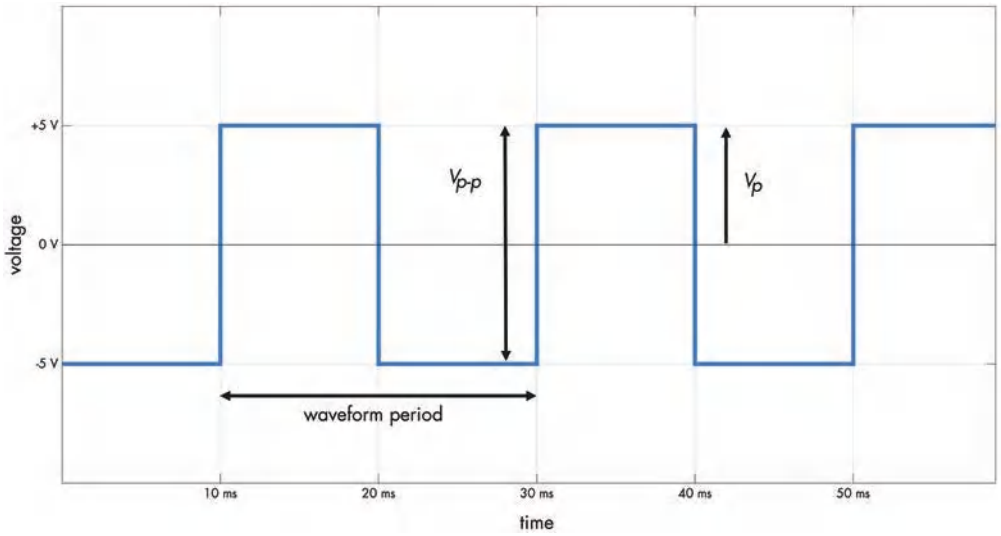


Figure 4-2: A square wave annotated with key parameters

Another important parameter is the waveform's vertical swing. The most straightforward approach is to measure the difference between the lowest and highest voltages recorded, yielding a number that's known as *peak-to-peak amplitude*. This is marked in Figure 4-2 as $V_{p\text{-}p}$, and in this instance it's equal to 10 V.

In some applications, such as audio processing, it's more common to specify *peak amplitude*. Despite sounding similar, the term has a different meaning. It's the swing from the signal's midpoint to the highest positive reading, marked as V_p in Figure 4-2. The definition of peak amplitude is unambiguous only if the waveform is symmetrical. Otherwise, reasonable people may disagree about where the midpoint is. My best advice is to use the term sparingly.

There's also a third way to express signal amplitude, known as the *root-mean-square voltage*, denoted V_{RMS}. The term may sound esoteric, but you probably encounter it every day. The 120 V specification of the electrical outlets in US homes is given as an RMS voltage. To explain the concept properly, we need to take a brief detour.

Root-Mean-Square Voltage

Root-mean-square voltage gives us a way to relate AC and DC voltages in certain applications. It allows us to easily calculate power consumption in AC circuits, determine appropriate component ratings, and perform other power-related math.

In Chapter 3, we noted that electricity flowing through a resistor produces heat. We also discussed that it can be useful to know how much power will be dissipated as heat if a voltage-based signal is supplied to a known resistance, so we don't damage components by supplying too high a voltage. Experiments with glowing resistors aside, a good practical example where this matters would be designing an audio amplifier for a loudspeaker. Speaker coils have a specific resistance and are rated for a certain number of watts. If we exceed the rating, the speaker might overheat.

Earlier, we derived the following formula for heat dissipation in a resistive load:

$$P = \frac{V^2}{R}$$

Alas, the equation works only if the supplied voltage is constant. What do we do if the applied signal changes over time?

To develop intuition about this scenario, we can consider a rudimentary example of a 10 Ω load and an input signal that changes in three discrete steps of equal duration: 1 V, 10 V, and then back to 1 V. This is shown in Figure 4-3.

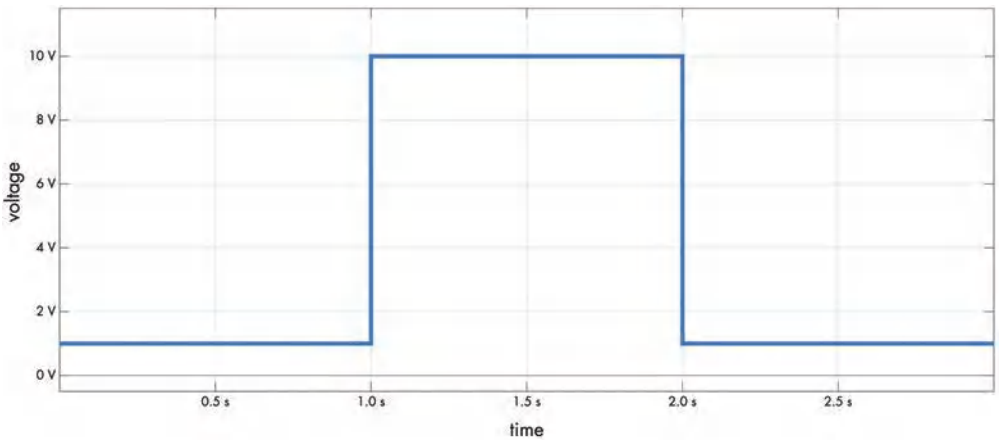


Figure 4-3: A simple three-step signal

Because the steps are discrete, we can treat this as a progression of three steady states with independent, separately calculated power dissipation figures, as summarized in Table 4-1.

Table 4-1: Power Dissipation at Discrete Time Steps

Time	Input voltage	Calculated power
0 s . . . 1 s	$V(0) = 1\ \text{V}$	$P(0) = (1\ \text{V})^2 / 10\ \Omega = 0.1\ \text{W}$
1 s . . . 2 s	$V(1) = 10\ \text{V}$	$P(1) = (10\ \text{V})^2 / 10\ \Omega = 10\ \text{W}$
2 s . . . 3 s	$V(2) = 1\ \text{V}$	$P(2) = (1\ \text{V})^2 / 10\ \Omega = 0.1\ \text{W}$

Common sense tells us that the average power dissipated in the process is a simple arithmetic mean of the individual results:

$$P_{avg} = \frac{P(0) + P(1) + P(2)}{3} = \frac{0.1\,\text{W} + 10\,\text{W} + 0.1\,\text{W}}{3} = 3.4\,\text{W}$$

The result is correct, but this method of calculating the average isn't always ideal. For example, to understand the heat dissipated when playing back Rick Astley's "Never Gonna Give You Up," do we need to compute a power value for every beat of the song? And if someone turns up the volume and thus changes all the momentary voltages, do we really need to redo the entire calculation? There has got to be a simpler way!

A more elegant solution would be to find an equivalent DC voltage that dissipates the same amount of power: a single number that represents the power footprint of an arbitrarily complex AC waveform. This way, we could model the relationship between audio tracks, volume, and speaker performance in a more straightforward way.

The first instinct might be to average the voltages in each time slice, but the resulting power dissipation figure doesn't match the earlier result:

$$V_{avg} = \frac{V(0) + V(1) + V(2)}{3} = \frac{1\,\text{V} + 10\,\text{V} + 1\,\text{V}}{3} = 4\,\text{V}$$

$$P = \frac{(V_{avg})^2}{R} = \frac{(4\,\text{V})^2}{10\,\Omega} = \cancel{1.6\,\text{W}} \leftarrow \text{wrong!}$$

The issue is the V^2 term in the power equation: A sum of squares isn't the same as the square of a sum. To make progress, we can instead solve the power equation symbolically:

$$\text{Power equation for each time slice}: \quad P(0) = \frac{V^2(0)}{R},\ P(1) = \frac{V^2(1)}{R},\ P(2) = \frac{V^2(2)}{R}$$

$$\text{Average power}: \quad P_{avg} = \frac{P(0) + P(1) + P(2)}{3}$$

$$\text{Substituted form}: \quad P_{avg} = \left(\frac{V^2(0)}{R} + \frac{V^2(1)}{R} + \frac{V^2(2)}{R} \right) \Big/ 3$$

The R in the denominator comes from the three $V^2(n)/R$ fractions that make up the individual formulas for $P(0)$, $P(1)$, and $P(2)$. The final divisor, 3, is carried over from the calculation of the arithmetic mean of the sum. The order of division doesn't matter, so we can restate the formula as follows:

$$P_{avg} = \left(\frac{V^2(0) + V^2(1) + V^2(2)}{3} \right) \Big/ R$$

This reordered notation is interesting. The first half of the expression is the arithmetic mean of the squared version of the input waveform, sitting right in place of the V^2 part in the constant-voltage $P = V^2/R$ formula that we started with. Let's denote this square-mean as x and confirm that the math checks out:

$$x = \frac{(1\,\text{V})^2 + (10\,\text{V})^2 + (1\,\text{V})^2}{3} = 34\,\text{V}^2$$

$$P = \frac{x}{R} = \frac{34\,\text{V}^2}{10\,\Omega} = 3.4\,\text{W} \;\leftarrow\; \text{correct!}$$

The lingering question is the nature of x in physical terms. It's not a substitute for voltage; remember that it took the place of the *square* of voltage (V^2) in the original power expression. To find the power-equivalent voltage we're after, we actually need to take the square root of x.

And that's what V_{RMS} is. It's the square root (R) of the arithmetic mean (M) of the squared waveform (S). In this particular case, $V_{RMS} = \sqrt{34} \approx 5.831$ V. This is the constant DC voltage that would dissipate the same amount of heat as our signal that changes from 1 V to 10 V and back to 1 V.

Once we have the RMS formula for a class of waveforms, we can more easily model the effect of turning up the volume on Rick Astley. To see how the power delivered to the speaker changes when the signal amplitude is doubled, simply multiply V_{RMS} by 2. There's no need to adjust the entire waveform and then recalculate $P(0)$, $P(1)$, $P(2)$, and so on.

The more practical case of a square wave that alternates between two voltages, V_{lo} and V_{hi}, can be solved analogously to the toy scenario outlined earlier, except now we only need to calculate the mean of two points instead of three:

$$V_{RMS} = \sqrt{\frac{{V_{lo}}^2 + {V_{hi}}^2}{2}}$$

Returning to your wall outlets, that 120 V is actually the V_{RMS} of a signal alternating between roughly +170 V and −170 V, with a peak-to-peak swing in the vicinity of 340 V (ouch!). The complication is that the signal doesn't have the abrupt transitions of a square wave but rather is a continuously changing sinusoidal waveform. That's where we'll turn next.

Properties of Sinusoids

The undulating shape of a sinusoid is probably familiar to most readers, but there's nuance to how the waveform is constructed, why it crops up all over the place, and what rules govern its curvature. On the next couple of pages, we'll talk about sines as a mathematical construct. After that, we'll come back to the question of the equivalent V_{RMS} for these waveforms. As hinted near the end of Chapter 3, it will be a bit of a journey, but along the way, you'll learn a couple of remarkably useful mathematical tricks.

Sine Wave Basics

The two most fundamental trigonometric functions, sine and cosine, are usually introduced in high school with the help of a right-angled triangle. One of the two acute angles of the triangle is designated as α. At any scale, there's a fixed, angle-specific

ratio of lengths between the vertical edge across from α (the *opposite*) and the long diagonal edge (the *hypotenuse*). This ratio is known as the *sine* of α. Analogously, the ratio of the horizontal edge (the *adjacent*) to the hypotenuse is called the *cosine*. A visual representation of these definitions is shown in Figure 4-4.

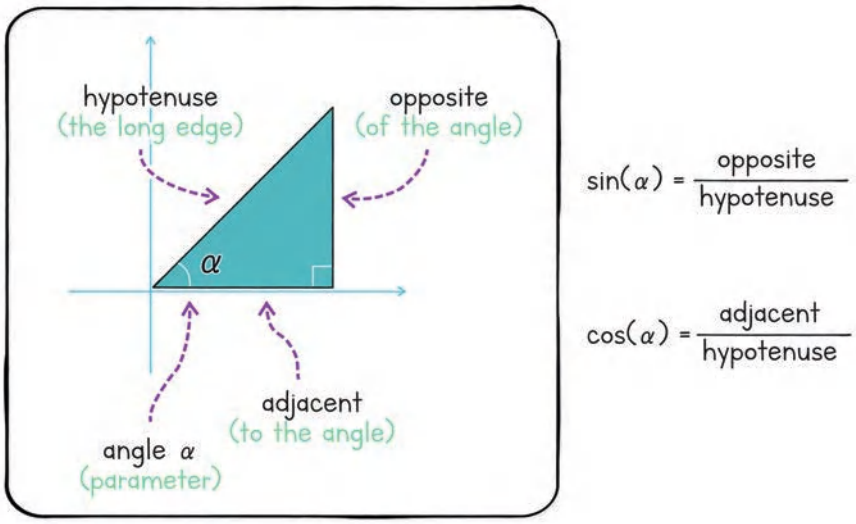


Figure 4-4: The basic anatomy of sine and cosine

It's easy to see that if α is $0°$, the triangle becomes a pancake, and the opposite has a length of 0; the value $\sin(\alpha)$ must be 0 too. Conversely, if the angle is $90°$, the triangle is squished into a vertical line; the opposite is the same length as the hypotenuse, so the value of $\sin(\alpha)$ is 1.

Specific intermediate values can also be calculated. For example, we can easily solve the case of $45°$ if we make the observation that the function plot of $y = x$ is a diagonal line slanted at $45°$. For a triangle anchored to the center of the coordinate system, a hypotenuse extending at a $45°$ angle from that center fully overlaps with the aforementioned diagonal, so it's necessarily made out of points that satisfy the same condition: $y = x$.

One of these points is the far end of the hypotenuse; let's christen that as point A. Given that specific triangle anchored to (0, 0), the length of the opposite is equal to the height (the y-coordinate) of point A; the adjacent has a length equal to the point's x-coordinate. Because the coordinates of A necessarily obey the $y = x$ constraint, we can conclude that the adjacent and the opposite of our $45°$ triangle have equal lengths. From there, we apply the Pythagorean theorem to calculate the hypotenuse and then solve for $\sin(45°)$:

$$\text{hypotenuse}^2 = \text{adjacent}^2 + \text{opposite}^2 = 2 \cdot \text{opposite}^2$$

$$\text{hypotenuse} = \sqrt{2 \cdot \text{opposite}^2} = \sqrt{2} \cdot \text{opposite}$$

$$\sin(45°) = \frac{\text{opposite}}{\text{hypotenuse}} = \frac{\cancel{\text{opposite}}}{\sqrt{2} \cdot \cancel{\text{opposite}}} = \frac{1}{\sqrt{2}} \approx 0.707$$

NOTE *We're going to lean on the Pythagorean theorem several times throughout this chapter. For readers who are iffy on the concept or were taught the rule without seeing a proof of the underlying* $a^2 + b^2 = c^2$ *equality, I recommend reviewing the relevant Wikipedia article. It's surprisingly accessible and offers a number of simple visual proofs.*

Although we can solve $\sin(\alpha)$ for certain angles, there's no universal, finite arithmetic formula that works for every case. Because of this, the sine function is usually treated as one of the fundamental building blocks of practical calculations, somewhat akin to the square root operator or the value of pi (π).

A common simplification for working with sines is to make the length of the hypotenuse equal to 1 and to permanently anchor the figure to the center of the coordinate system. If we make it so, the far end of the hypotenuse (our point A) will draw a circle with a radius of 1 as angle α advances from 0° to 360° (Figure 4-5). Further, because the divisor for calculating the value of $\sin(\alpha)$ is fixed at 1, the value of the function is just equal to the vertical coordinate of point A. Similarly, the value of $\cos(\alpha)$ is the horizontal coordinate.

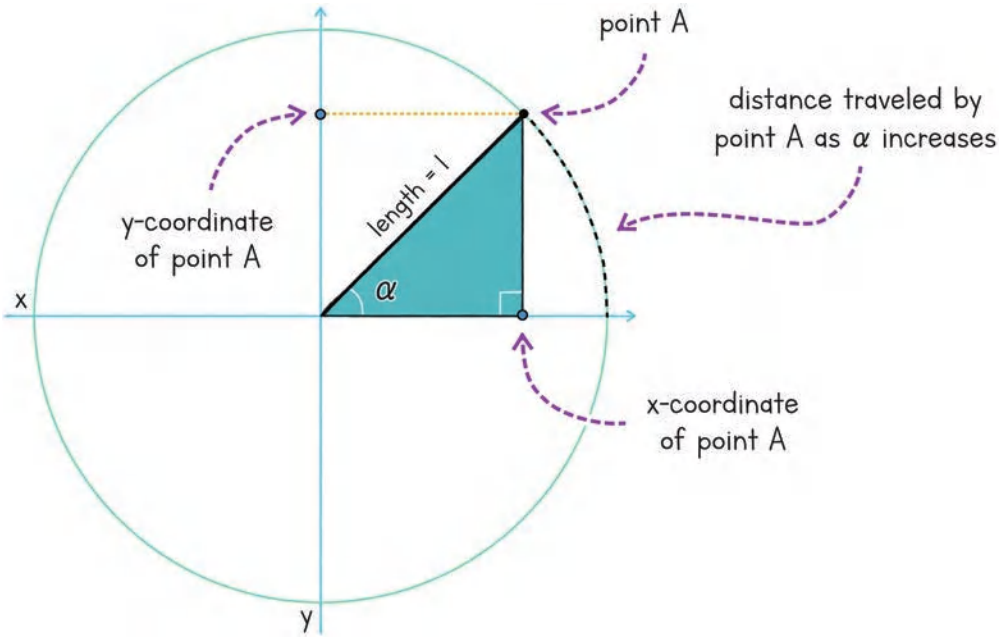


Figure 4-5: Another way of looking at sines and cosines

This model lends itself to another way of specifying the angular parameter of trigonometric functions. In college-level math and in computing, the angle is typically expressed not as degrees but as the distance traveled by point A along the circumference of the unit circle, starting at 3 o'clock and moving counterclockwise. This is known as *radians*. The circumference of a circle with a radius of 1 is 2π, so it takes 2π radians to complete a full turn.

The plots of two cycles of $y = \sin(x)$ and $y = \cos(x)$ are provided in Figure 4-6. The bottom scale is annotated with radians. The top scale shows more traditional values in degrees.

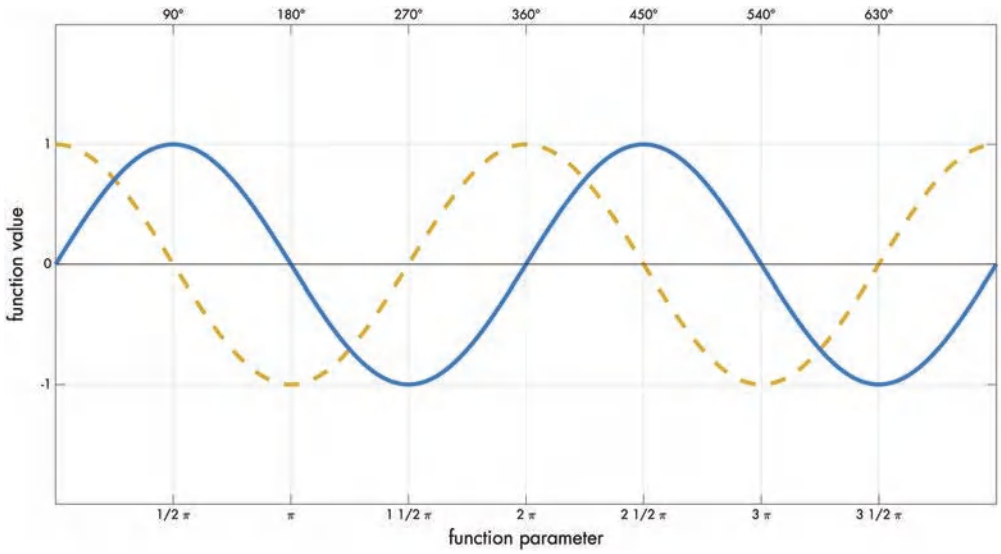


Figure 4-6: Plots of sine (solid line) and cosine (dashed line) for angles from 0 to 4π radians (0° to 720°)

Notice that the cosine function looks identical to the sine, except it's shifted left by one-fourth of a cycle (90°, or $\pi/2$ radians). In effect, the functions are interchangeable with the application of the appropriate shift. One of the many ways to express this relationship is the following formula:

$$\cos(x) = \sin(x + \frac{\pi}{2})$$

When using these functions to model signals in electronic circuits, it's important to remember that incrementing the parameter by 1 doesn't correspond to a full 360° turn. In other words, to build a voltage-based waveform that completes one cycle per second, we can't just write this:

$$V(t) = \sin(t) \quad \leftarrow \text{ wrong!}$$

If t refers to time in seconds, this function would need roughly 6.28 seconds to complete a cycle, netting a frequency of around 0.16 Hz. To get 1 Hz, we need to "speed up" (multiply) t so that every second corresponds to an increment of 2π. Further, if we want the waveform to run at some chosen multiple of 1 Hz—that is, some arbitrary frequency f—we need to throw in a second multiplier:

$$V(t) = \sin(2\pi f \cdot t)$$

NOTE *Many electronics textbooks replace the 2πf part with a new symbol: ω. It stands for* angular frequency, *or hertz multiplied by 2π. Since it's an oddball measure with no relation to how we normally talk about motion or time, and its only significance is that it saves the effort of typing two characters, we won't be using this convention in the book. If you come across the symbol elsewhere, you can treat it the same as 2π*f.

The values we get out of the sine function range from −1 to 1, since in a right-angled triangle the opposite can't get longer than the hypotenuse. To model a voltage-based signal of a certain amplitude, we can multiply the result by a constant. To shift the whole waveform up or down, we can add or subtract a fixed value. For example, the formula for a sine signal that swings from 0 V to 5 V can be written like so:

$$V(t) = 2.5\,\text{V} \cdot \sin(\ldots) + 2.5\,\text{V}$$

Multiplying by 2.5 V gives us a sine wave oscillating from −2.5 V to +2.5 V. Adding 2.5 V then shifts the range up to 0 V to 5 V.

The Example of Simple Harmonic Motion

So far, we have a model of sine signals, but we still lack an explanation of why sinusoids seem to be hiding under every rock. The pattern can be observed on the surface of water, in electromagnetic waves, in the motion of a pendulum . . . and none of these phenomena seem to involve circles or triangles.

At the root of this pattern lies a particular relationship between the internal energy and the easily observed parameters of many physical systems. The relationship holds true for the voltage of a capacitor as much as it holds true for the speed of a bullet fired from a gun. We can explain the relationship in electronic terms, but that would require getting ahead of ourselves. Instead, we'll first lean on an example from classical physics: the *simple harmonic motion* of a mass dangling on a spring. The model seemingly has little to do with electronics, but it lets us practice essential math skills while working with well-known concepts such as position and velocity. Before long, we'll also discover that the inertia of a mass is just like the electronic concept of inductance, and that springs and capacitors have a lot in common too.

The basic setup for our physical model is shown in Figure 4-7. A block moves back and forth as the spring it's attached to alternates between maximum compression and maximum extension.

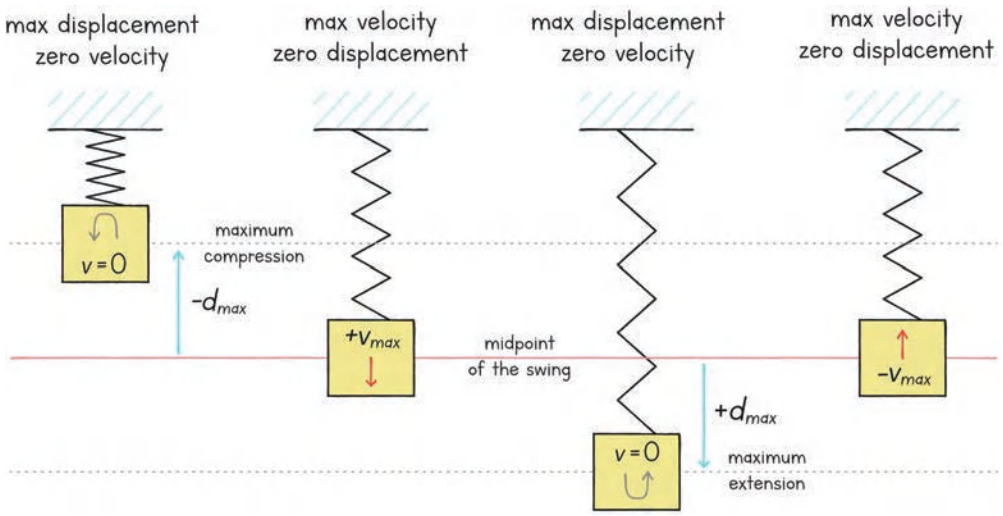


Figure 4-7: The oscillating motion of a block on a spring

For simplicity, let's assume that the spring is perfectly elastic and there's no air drag. In other words, if you give it a push, it keeps bouncing for all eternity. We can also add "in space" to the description of the experiment; this lets us ignore the effects of gravity (although it's also possible to show that gravity is of no special consequence

for the result). To foreshadow the conclusion, if we plot the displacement of the spring or the speed of the moving mass over time, we'll get sinusoids. The adventure ahead is just figuring out why.

To model the behavior of this system, we need to turn to a well-known concept in physics: the conservation of energy. The rule tells us that in a closed system that's isolated from the outside world, energy can change form, but its sum must remain constant over time. In this particular experiment, some of the energy of the system is stored in the spring, and some of it lies in the moving block. At any point, the following must therefore hold true:

$$E_{total} = E_{spring} + E_{block} = \text{const}$$

Alas, in this experiment, much like in an electronic circuit, energy isn't something we can directly observe. What we *can* measure is spring deflection and block velocity. To move forward, we need to turn these "observables" into deeper insights about system state.

Potential Energy of a Spring

The empirical model of a spring, known as *Hooke's law*, is quite simple. Within certain bounds, the spring is just a linear mechanism that pushes back proportionally to how much it's already compressed. More precisely, the instantaneous pushback force (F) you need to overcome to keep compressing a spring is a function of its current displacement d (the distance from the at-rest position) and a constant k that describes the spring's stiffness:

$$F = k \cdot d$$

The constant depends on the material the spring is made from and the device's exact shape, but this is unimportant for our discussion. Regardless of the value of k, if we plot F in relation to d, we get a straight line (Figure 4-8).

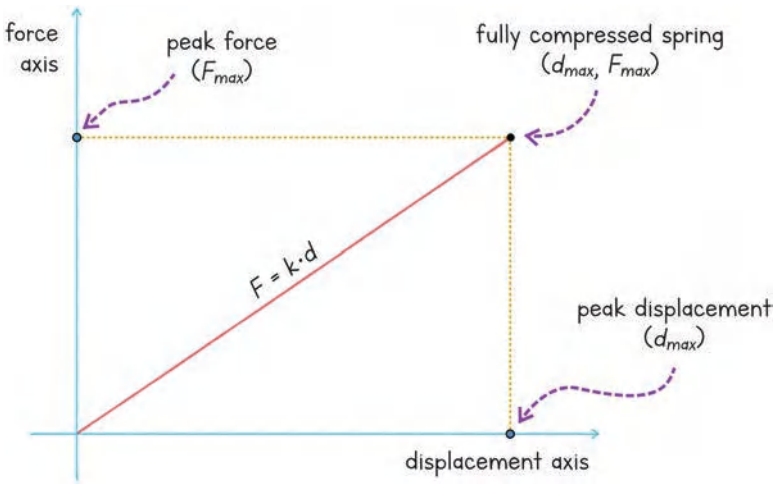


Figure 4-8: Force versus displacement in an idealized spring

Again, lest this discussion feels too abstract, we find similar relationships in electronics. For example, the effort (the voltage) that needs to be applied to deposit additional charge on a metal plate depends on the material's physical properties, along with the repulsion force stemming from the amount of charge already stored on the plate.

In any case, let's stick to springs and consider what happens if we compress the spring from position $d_0 = 0$ to some chosen point d_{max}. From the plot, we can tell the force we need to fight will ramp up linearly from $F_0 = 0$ to $F_{max} = k \cdot d_{max}$. With such a simple linear ramp, we can also intuit that the average force during the movement is equal to the midpoint between F_0 and F_{max}:

$$F_{avg} = \frac{F_{max} - F_0}{2} = \frac{k \cdot d_{max}}{2}$$

Next, we need to relate this force to the energy stored in the spring. Energy is closely related to work: If you're leaning against a stationary lawnmower, you're exerting force, but no energy is being transferred, and no work is getting done. The situation changes if you start pushing the mower up a hill by applying a constant force to the handle. In classical mechanics, the amount of physical work performed—the amount of energy (E) you expended—is equal to the force on the handle times the distance traveled up that hill (d):

$$E = F \cdot d$$

The same logic also works for pushing a spring. We applied an average force of F_{avg} to compress the spring by some distance d_{max}. The stored energy can therefore be calculated as:

$$E_{spring} = F_{avg} \cdot d_{max} = \frac{k \cdot (d_{max})^2}{2}$$

We have half the system sorted out. Next, we need to figure out the kinetic energy of the moving block.

Energy of a Moving Mass

The simplest way to reason about the energy "stored" in a moving object is to consider how much effort is expended to accelerate an identical mass to the same speed. We can accelerate it any way we like. Two identical objects traveling at the same velocity should have the same kinetic energy, no matter if they were shot out of a cannon or thrown by an angry orangutan.

To keep things simple, let's instruct the orangutan to ramp up the speed of the object linearly from $v_0 = 0$ to v_{max} over a period of t seconds. Acceleration is just a measure of the gain in velocity over time, so in this case, it works out to the following:

$$a = \frac{v_{max} - v_0}{t} = \frac{v_{max}}{t}$$

Because the speed increased linearly from v_0 to v_{max}, we can also assert that the average speed in this time window is given by this formula:

$$v_{avg} = \frac{v_{max} - v_0}{2} = \frac{v_{max}}{2}$$

Next, if we know the average speed, we can trivially calculate the distance traveled by the mass while being pushed. We know that the push lasted t seconds, so the distance is just speed multiplied by time:

$$d = v_{avg} \cdot t = \frac{v_{max}}{2} \cdot t$$

In Newtonian physics, the acceleration experienced by an object is proportional to the motive force and inversely proportional to the object's mass ($a = F/m$). A simple analogy is that in space, a rocket with a constant thrust will be gaining speed at a constant rate for as long as the fuel lasts. Rearranging this equation and substituting the earlier value of a, we get the formula for how hard the block must've been pushed to reach v_{max} in time t:

$$F = m \cdot a = m \cdot \frac{v_{max}}{t}$$

Lastly, to figure out the work done and thus the kinetic energy imparted on the object, we plug this force and the previously calculated displacement into the "lawnmower" formula for work ($E = F \cdot d$):

$$E_{block} = F \cdot d = m \cdot \frac{v_{max}}{\cancel{t}} \cdot \frac{v_{max}}{2} \cdot \cancel{t} = \frac{m \cdot {v_{max}}^2}{2}$$

In other words, kinetic energy depends on the mass of the moving object and its speed. It has no dependency on any of the intermediate variables that described the specifics of our thought experiment.

The Complete Model

With the formulas for kinetic and potential energy sorted out, we can finally see how block speed (v) and spring displacement (d) relate to overall system energy:

$$\begin{aligned} E_{total}\,(\text{const}) &= E_{spring} + E_{block} \\ &= \frac{k \cdot d^2}{2} + \frac{m \cdot v^2}{2} \\ &= \frac{k}{2} \cdot d^2 + \frac{m}{2} \cdot v^2 \\ &= \left(\sqrt{\frac{k}{2}} \cdot d\right)^2 + \left(\sqrt{\frac{m}{2}} \cdot v\right)^2 \end{aligned}$$

I rearranged the k and m terms this way to illustrate that they're just material-specific multipliers that scale the observables. Nothing we're about to do depends on these values, so to keep the math tidy, let's assume that by some stroke of luck, the scaling factors are the same ($k = m$). This allows us to write the following observation about the general relationship between the numerical values of v and d:

$$d^2 + v^2 = \text{const}$$

This is an important result, so let's keep it in mind for a while.

To gain additional insights, we can plot the observed state of the system, a specific combination of v and d, as a single point on a Cartesian plane. We can choose the horizontal coordinate to correspond to velocity and the vertical coordinate to correspond to displacement, netting us what's shown in Figure 4-9.

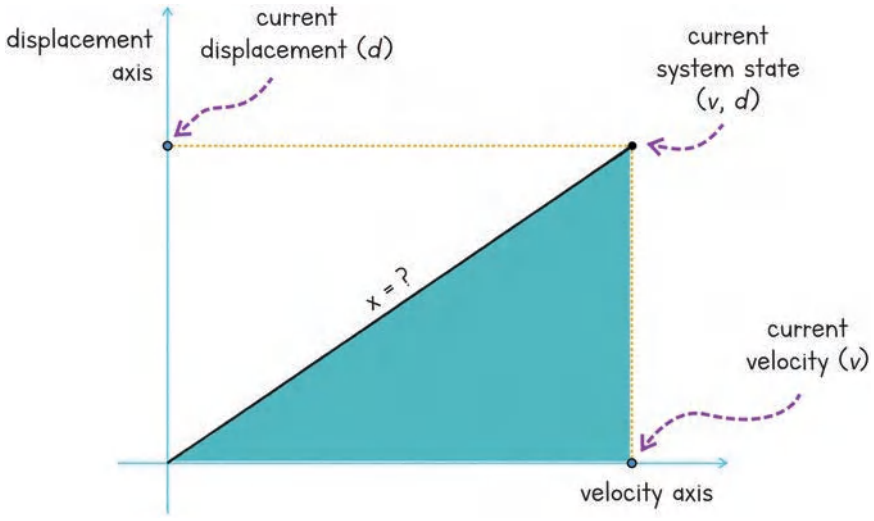


Figure 4-9: A two-dimensional representation of block speed and spring displacement

In that plot, I've also included a segment labeled x. This segment is the distance from the origin of the coordinate system, (0, 0), to the point representing system state. The shaded area of the plot is a right-angled triangle, so the value of x is governed by the trusty old Pythagorean theorem:

$$x^2 = v^2 + d^2$$

But hold on. Note the striking similarity of this expression to the earlier, simplified rule for the conservation of energy in the system:

$$d^2 + v^2 = \text{const}$$

The criterion for conservation of energy tells us that this sum of squares ($d^2 + v^2$) must be constant; if so, then per the Pythagorean equation, x^2 must be constant too. And if x^2 is constant, the same can also be said about just plain x. For our purposes, the values of the constants don't matter. All that matters is that the distance x does not change over time. Every point corresponding to a pair of (v, d) values must be equidistant from the center of the coordinate system—that is, it must lie on a circle (Figure 4-10).

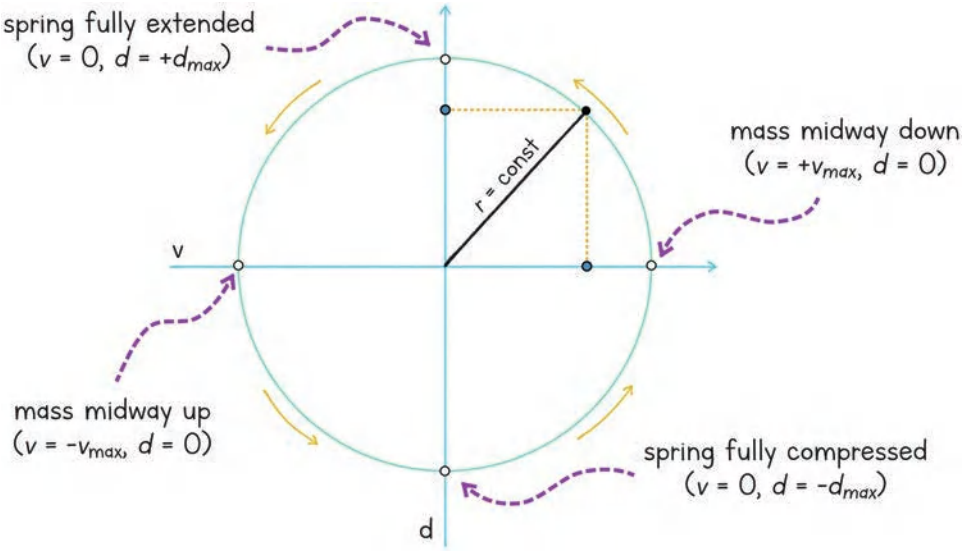


Figure 4-10: The dynamics of observables for a mass on a spring

Again, the state of the system can only progress along this circular path, and no other (v, d) combinations are possible because under the conservation of energy, the distance from (0, 0) to (v, d) can't change. Any point inside the circle has lower energy; reaching any point outside would require an energy gain.

With that, let's plug in what we know from trigonometry. At any given time, spring displacement (the *opposite* in the right triangle tucked under the constant-length segment in Figure 4-9) is described by the sine of some given angle, while the speed of the block (the *adjacent*) is determined by the cosine. This is the genesis of the sinusoidal motion we observe over time.

The same relationship between observable parameters and stored energy is present in many other physical systems. For example, as noted earlier, it describes the deposition of charges on a metal plate—the basic behavior of a capacitor. Another electronic component, the inductor, deals with the "kinetic" energy of moving charges, with strong parallels to the equations for a moving mass.

The Lingering Question of Angular Speed

Before we move on, it's important to acknowledge that our proof is still missing an important piece. We've found the evidence of circular motion, but we don't really know that the state of the system is traveling along that "state circle" at a uniform speed. If the motion is choppy, the plots of v and d over time might not look like sines at all. Uniform rotation *feels* like the right answer, but if we're doing math, it pays to do it properly.

This isn't just a matter of pedantry. To get to the root of the matter, we'll need to introduce a clever calculus trick that will soon pay dividends. We'll be looking at a process under humongous magnification and through the prism of a vanishingly small increment in time. For now, we'll talk about these concepts informally, but don't be alarmed by the seemingly sloppy phrasing. We'll firm up the mathematical reasoning down the line.

To get going, let's denote the tiny time increment as *delta-t*, or Δt. In that time window, the system advances by some microscopic distance Δs along the state circle. Because the distance is tiny, the magnified section of the circle shown in Figure 4-11 is essentially a straight line.

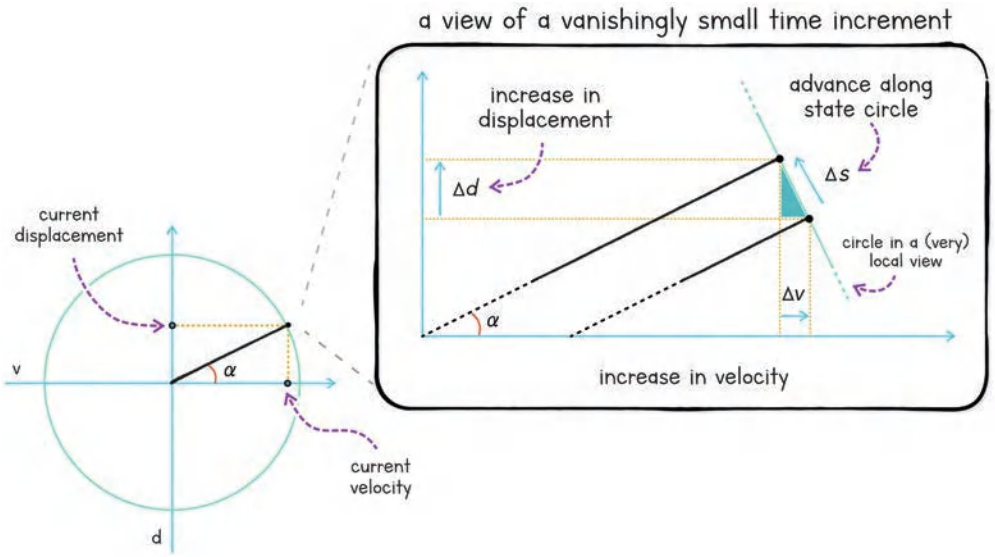


Figure 4-11: A geometric analysis of the system's rate of movement along the circle

Our objective is to describe Δs in relation to the passage of time, finding the speed with which the system is moving along the state circle. If the speed is constant, the plots of v and d must be sinusoids. If it's variable—for example, if the transfer of energy speeds up or down over time—the resulting plots of observables might look different and all our hard work might be for naught.

A good starting point for the analysis is the small, right-angled triangle highlighted in the inset of Figure 4-11. We know the lengths of all three of the triangle's edges, so we can wheel out the Pythagorean theorem once more and obtain the following equality:

$$(\Delta s)^2 = (\Delta v)^2 + (\Delta d)^2$$

Next, let's ponder what Δd actually is. It's the increase in displacement in a given unit of time (Δt), which is just another name for speed. If the block is moving at a particular speed (v), the displacement of the spring will necessarily change by that speed multiplied by the time span we're looking at:

$$\Delta d = v \cdot \Delta t$$

This equation might seem suspect because we know that v is continuously changing. However, we can informally assert that in a vanishingly short time window, the increment in velocity (Δv) is inconsequential in relation to the underlying value of v. In other words, when using this analysis method, we'd be forgiven to treat velocity as constant.

At this stage, we have a formula for Δd that relates it to one of the observables (v). Doing the same for Δv is a bit more complicated. From basic physics, it should be clear that the parameter—the change in velocity—is proportional to acceleration multiplied by time:

$$\Delta v = a \cdot \Delta t$$

The remaining task is to figure out what that acceleration might be. Luckily, we can lean on the earlier analysis: The acceleration of the block is a function of the force applied by the spring ($a = F/m$). Further, the pushback force of the spring is

proportional to displacement times the spring constant ($F = k \cdot d$). Combining the formulas, we get:

$$a = \frac{F}{m} = \frac{k \cdot d}{m}$$
$$\Delta v = a \cdot \Delta t = \frac{k \cdot d}{m} \cdot \Delta t$$

Once again, k and m are just scaling factors that squish or stretch the state circle without doing much else. For a well-proportioned circle ($k = m$), the formula can be boiled down to:

$$\Delta v = d \cdot \Delta t$$

This equation tells us that instantaneous change in velocity is proportional to current spring displacement. Moments earlier, we also established that instantaneous change in displacement is proportional to current block velocity. With these findings in tow, we can go back to the earlier Pythagorean formula describing the triangle in Figure 4-11, plug in these equations, and then solve for Δs:

$$\text{Original equation}: \ (\Delta s)^2 = (\Delta v)^2 + (\Delta d)^2$$

$$\text{Substituting } \Delta v \text{ and } \Delta d: \ (\Delta s)^2 = (v \cdot \Delta t)^2 + (d \cdot \Delta t)^2 = (v^2 + d^2) \cdot (\Delta t)^2$$

$$\text{Simplified}: \ \Delta s = \sqrt{v^2 + d^2} \cdot \Delta t$$

As the final step, the formula for Δs can be divided by the duration of the time slice (Δt) to calculate the instantaneous speed at which the system advances along the state circle. To avoid mix-ups with the variables describing the motion of the block, I'm going to call this u:

$$u = \frac{\Delta s}{\Delta t} = \frac{\sqrt{v^2 + d^2} \cdot \cancel{\Delta t}}{\cancel{\Delta t}} = \sqrt{v^2 + d^2}$$

At first glance, the speed appears to vary depending on the momentary values of v and d. That would be bad news. But recall that the main thrust of the earlier proof for "circularity" was that $v^2 + d^2$ is a constant. Despite appearances, we can therefore assert the following:

$$u = \sqrt{\text{const}}$$

For any combination of v and d, we can rest assured that the angular speed along the state circle remains the same. The day is saved: Springs and simple electronic oscillators alike are free to naturally produce sine waves.

We still have one elephant in the room: Any mathematical argument constructed using terms such as *humongous* or *microscopic* feels more like a mushy approximation than a rigorous proof. Not so! Without bringing up all the formalisms of mathematical analysis, the result becomes exact as the time slice "tends to" zero—that is, as it becomes infinitesimal (infinitely small). This turns Δd and Δv into infinitesimals too, validating our assertion that they exert no measurable influence on the values v and d within any single time step. In the same vein, the state circle must be magnified infinitely to show Δs, so the step distance traveled along it is truly indistinguishable from a straight line.

At this point, we're done with bouncy blocks, but there's one more mathematical property of sines that's essential to the discussion of electronic circuits: the function's rate of change. We don't need physical props to analyze it, but we'll lean on the same infinitesimal Δt trick that helped us before.

Sine Rate of Change

To develop the theory of electronic components such as capacitors and inductors, we'll need to zero in on specific boundary conditions of the sine wave and think about how its value is changing in relation to the function's parameter.

Looking at Figure 4-12, it seems clear that the points where the amplitude of $y = \sin(\alpha)$ is the highest are also the points where the function's rate of change is the lowest. These are the positive and negative "humps" at $\alpha = \frac{1}{2}\,\pi$ (90°) and $\frac{3}{2}\,\pi$ (270°). In the same vein, the highest rate of change corresponds to the function crossing the zero line at $\alpha = 0$ and π (180°). If we draw the axes with a 1:1 aspect ratio, the angle of the slope at these zero crossings appears to be +/−45°.

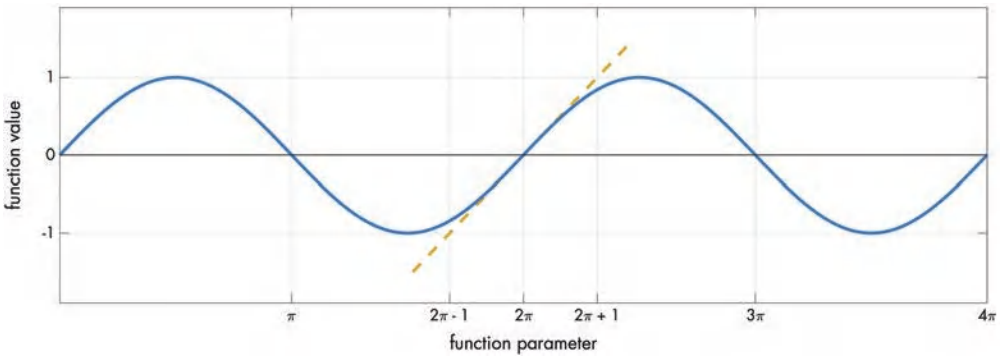


Figure 4-12: The maximum slope of sin(α) corresponds to zero crossings.

An informal proof that validates this observation is shown in Figure 4-13. As we've done before, we visualize the behavior of sin(α) using a point that's traveling along a unit circle. The value of the function is equal to the vertical coordinate of that point.

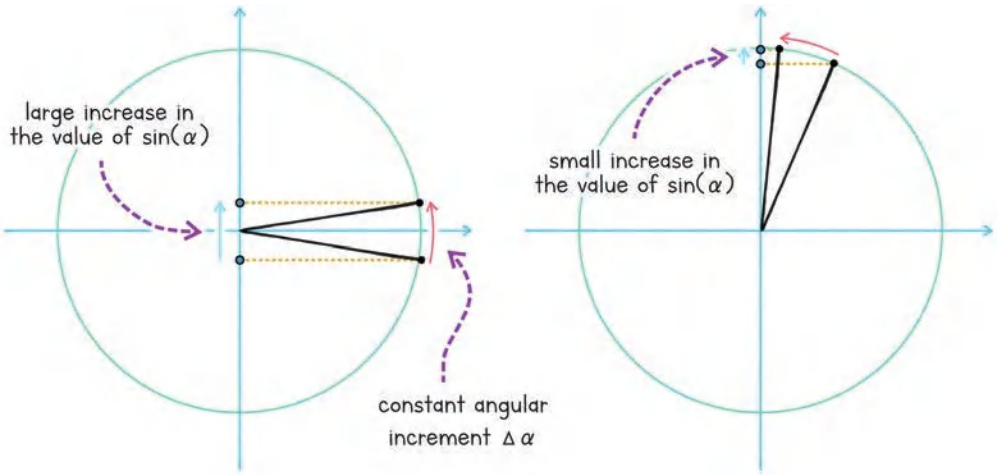


Figure 4-13: Eyeballing the rate of change of the sine function

In the illustration, the increment in function parameter is represented as a step along the unit circle ($\Delta\alpha$), while the change in the value of the function is pictured as a projection onto the vertical axis. If the increment along the circle is always the same, it's clear that the projected vertical step should be the highest at 3 o'clock and 9 o'clock, or when α is equal to 0° or 180°. In other positions, the projected shadow is shorter, approaching zero at 12 o'clock and 6 o'clock.

We can take this a bit further. At the 3 o'clock and 9 o'clock peaks, the motion along the circle is essentially vertical, so there should be a 1:1 correspondence between an infinitesimal increment in the function parameter and in function value. In other words, at these zero-crossing points, the function must be briefly behaving just like $y = +/-\alpha$, copying its input parameter to output. This explains the $+/-45°$ diagonal line shown on the plot in Figure 4-12.

This explanation may still feel a bit hand-wavy, so Figure 4-14 shows a more rigorous analysis of the rate of change.

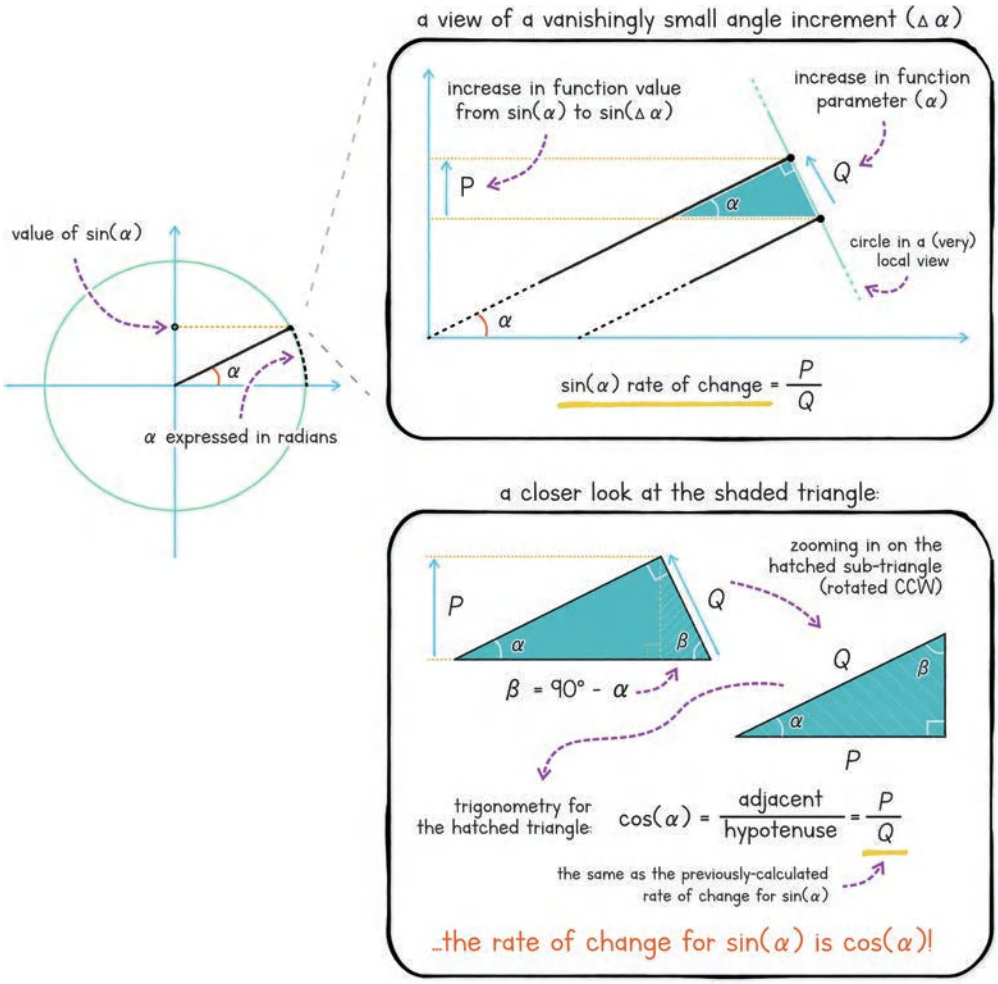


Figure 4-14: A visual analysis of the rate of change for sin(α)

As before, we start with a unit circle and then take a look at a vanishing increment of the parameter of a sine function. We denote this increment as $Q = \Delta\alpha$; the corresponding increment in function value along the y-axis is marked as P. In other words, the instantaneous speed at which the value of sin(α) is changing in proportion to its parameter is P/Q. We would expect this speed to be equal to 1 at $\alpha = 0°$ and to 0 at $\alpha = 90°$, but we want to know what's happening in between.

To figure out the answer to this riddle, we need to take a closer look at the right-angled triangle highlighted in the first panel of the proof. The triangle has one angle equal to α (in radians) and a second angle equal to $90°$ (duh). One edge has a length of Q, and in its original "upside-down" orientation, the triangle has a height of P. The

sum of all angles in any triangle is 180°, so we can also infer that the third angle (β) is equal to $90° - \alpha$.

The next step is to slice this triangle into two halves (second panel, left). The values of the three angles of the smaller, dashed triangle are β, 90°, and $(90° - \beta)$, the last of which works out back to α. That angle α is sandwiched between an adjacent with the length of P and a hypotenuse with the length Q. The ratio of the adjacent to the hypotenuse is given by the cosine function, so we can say that $\cos(\alpha) = P/Q$.

But hold on a second! Didn't we just say that P/Q is the instantaneous rate of change of $\sin(\alpha)$? Now we're also saying that P/Q is equal to $\cos(\alpha)$?

This isn't a glitch. Instead, it's a cool, if counterintuitive, property of trigonometric functions. The instantaneous rate of change for $\sin(x)$ is given by the cosine of the same parameter, and $\cos(x)$ is exactly the same as $\sin(x + 90°)$. If we keep going, the rate of change for $\cos(x)$, or $\sin(x + 90°)$ if you prefer, works out to $\sin(x + 180°)$, aka $-\sin(x)$. In turn, the rate of change for that function is $\sin(x + 270°)$, aka $-\cos(x)$. After that, we're back to $\sin(x)$. It's turtles all the way down!

This is quite convenient because, in circuit design, we often have to calculate the rate of change of a signal to understand how an electronic component may respond to it. Other times, we observe the rate of change and want to figure out the original signal that produced it. For sine waveforms, both of these tasks are as simple as moving up or down the aforementioned four-step sequence of time-shifted sinusoids.

Pythagorean Identity and Sine-Wave RMS

Now that we're nearly done with the theory of sine waves, we can revisit the question of V_{RMS} for this particular wave shape. Among other things, this will allow us to go back and forth between RMS figures and peak-to-peak amplitude for mains voltages, or for audio tracks less jarring than square waves.

As a reminder, to find the RMS of an arbitrary waveform, we need to calculate the square root (R) of the arithmetic mean (M) of the squared waveform (S). We'll start with the last part: squaring the waveform. The general equation for a sine wave electrical signal of a given peak amplitude V_p is as follows:

$$V(t) = V_p \cdot \sin(2\pi f \cdot t)$$

Squaring the right-hand side yields:

$$\left(V_p \cdot \sin(2\pi f \cdot t)\right)^2 = {V_p}^2 \cdot \sin^2(2\pi f \cdot t)$$

The only part of the expression that's changing over time is the $\sin^2(\ldots)$ function. Let's introduce a new shorthand symbol, Sm, to represent the mean value of this component. This allows us to write the V_{RMS} expression as:

$$V_{RMS} = \sqrt{{V_p}^2 \cdot Sm} = V_p \cdot \sqrt{S_m}$$

Now all we have to do is find Sm. To find the mean of any periodic waveform, we only need to consider a single cycle. After all, the average of any repeated sequence is the same as the average of the basic repeating unit:

$$\frac{1 + 2 + 3}{3} = \frac{(1 + 2 + 3) + (1 + 2 + 3) + (1 + 2 + 3)}{9}$$

The period of $\sin^2(\ldots)$ can't be longer than the period of the sine function before squaring. In other words, we can safely limit the analysis to function parameters between 0° and 360°. Alas, this still nets an infinite number of continuously changing function values to average.

Luckily, the indispensable Pythagorean theorem comes to the rescue once again. Let's step away from the specific V_{RMS} scenario and consider an ad hoc right-angled triangle where the hypotenuse has a length n and one of the angles is α. By now, it should be clear that the other two sides can be calculated using trigonometric functions:

$$\text{opposite} = \sin(\alpha) \cdot n$$
$$\text{adjacent} = \cos(\alpha) \cdot n$$

The shape being a right triangle, we can also describe the relationship between the lengths of its sides using—you guessed it—the Pythagorean theorem:

$$\text{hypotenuse}^2 = \text{opposite}^2 + \text{adjacent}^2$$

Combining these three formulas, we can prove that the sum of squares of sine and cosine of the same angle is always equal to 1. This is known as the *Pythagorean identity*:

$$\text{hypotenuse}^2 = \text{opposite}^2 + \text{adjacent}^2$$

$$n^2 = \big(\sin(\alpha) \cdot n\big)^2 + \big(\cos(\alpha) \cdot n\big)^2$$

$$\cancel{n^2} = \cancel{n^2} \cdot \Big(\sin^2(\alpha) + \cos^2(\alpha)\Big)$$

$$1 = \sin^2(\alpha) + \cos^2(\alpha)$$

Now let's get back to the RMS conundrum. To explain the next step, imagine that you only put $5 bills in a piggy bank. The average value of all the bills in the piggy bank must be $5; the constraint placed on the constituent values carries over to the arithmetic mean. In the same vein, if the value of $\sin^2(\alpha) + \cos^2(\alpha)$ is always equal to 1, then the average value of these sums calculated across any range of α must also be 1.

As a matter of basic arithmetic, any long fraction representing the mean can be split into two shorter fractions that group the elements any way we please:

$$\frac{(a_1 + b_1) + (a_2 + b_2)}{2} = \frac{a_1 + a_2}{2} + \frac{b_1 + b_2}{2}$$

We can therefore split out the mean of any number of $\sin^2(\alpha) + \cos^2(\alpha)$ expressions into a sum of two independent means of $\sin^2(\alpha)$ and $\cos^2(\alpha)$ values, and the expression will still be equal to 1. We previously introduced *Sm* to denote the full-period arithmetic mean of $\sin^2(\ldots)$. If we complement it with *Cm* for $\cos^2(\ldots)$, we can express the new finding as:

$$Sm + Cm = 1$$

This equation works for any range of α, but our plan is specifically to average function values for parameters between 0° and 360°. This is important, because both the sine function and the cosine function complete one cycle in that span; they produce two identical waveforms that are offset in time but can be chopped up and reassembled into the same shape. To put it differently, for every $\sin(\ldots) = 0.5$, there will be exactly one corresponding point where $\cos(\ldots) = 0.5$, and vice versa.

Because the curves are effectively made of the same values, the full-period average of $\sin(\alpha)$ is necessarily equal to the full-period average of $\cos(\alpha)$. Simultaneously squaring the values of $\sin(\alpha)$ and $\cos(\alpha)$ doesn't alter this relationship. We end up with new curves, but they still look alike, and there's a continued 1:1 correspondence between any two points on each. The full-period arithmetic mean of $\sin^2(\alpha)$ must be the same as the arithmetic mean of $\cos^2(\alpha)$. This key observation finally allows us to figure out the value of Sm:

$$\text{New finding}: \quad Sm = Cm$$

$$\text{Earlier Pythagorean identity}: \quad 1 = Sm + Cm$$

$$\text{Combined equations}: \quad 1 = Sm + Sm = 2 \cdot Sm$$

$$\text{Final result}: \quad Sm = \frac{1}{2}$$

In other words, we've established that the arithmetic mean of the $\sin^2(\ldots)$ waveform between 0° and 360° is ½. Plugging this into the earlier V_{RMS} equation, we finally get the solution for a 0 V-centered sine wave with a peak amplitude of V_p:

$$V_{RMS} = V_p \cdot \sqrt{Sm} = V_p \cdot \sqrt{\frac{1}{2}} = \frac{V_p}{\sqrt{2}}$$

At this point, we can circle back to some of the earlier remarks about line (mains) voltage, which is one of the few places where you encounter RMS voltages without any indication that this is going on. In the United States, mains voltage of "120 V" is actually a sine-wave RMS figure. We can now fill in the math and calculate the value of V_p as $V_{RMS} \cdot \sqrt{2} \approx 170$ V. The peak-to-peak amplitude is double that, a whopping 340 V.

Working with Non-Sine Signals

So far, we focused on sinusoids because they're mathematically elegant and serve as useful building blocks for understanding circuit behavior. In real-world electronics, however, we encounter many other waveforms: square and triangle waves, single pulses, and so on. When these signals interact with capacitors, inductors, and other electronic components, how do we predict what will happen?

Well, in the days before computers, any circuit designer worth their salt needed to be something of a math whiz, as answering this question often required serious calculus. Today, though, we have more accessible approaches that let you blaze through most day-to-day problems without subjecting yourself to the pain of graduate-level math. In this section, I'll introduce two numerical approximation methods that can help you understand the behavior of real-world signals without breaking a sweat: discrete-time models and sum-of-sines approximations. Both of these methods can introduce small errors, but mathematical models of electronic components also take shortcuts, and in any case, you'll never find a resistor or a capacitor that's *perfectly* true to its factory spec.

Discrete-Time Models

By far the most powerful signal and circuit analysis technique boils down to simulating the circuit in discrete time steps. Circuit parameters are sampled at the beginning

of each tick. If internal voltages or currents need to be recomputed, we store temporary results and update the underlying variables only at the next tick.

As a rudimentary example, let's say we want to calculate the RMS voltage of a triangular waveform. There's a fairly simple symbolic solution to this problem. But suppose we're in a rush and the internet is out. Instead of fussing with trigonometry, we can divide the waveform into 20 discrete steps. At the beginning of each time interval, we sample the input signal and write the number down (Figure 4-15). In this particular experiment, there's no circuit state to recalculate, so we just keep recording measurements until we get to the end.

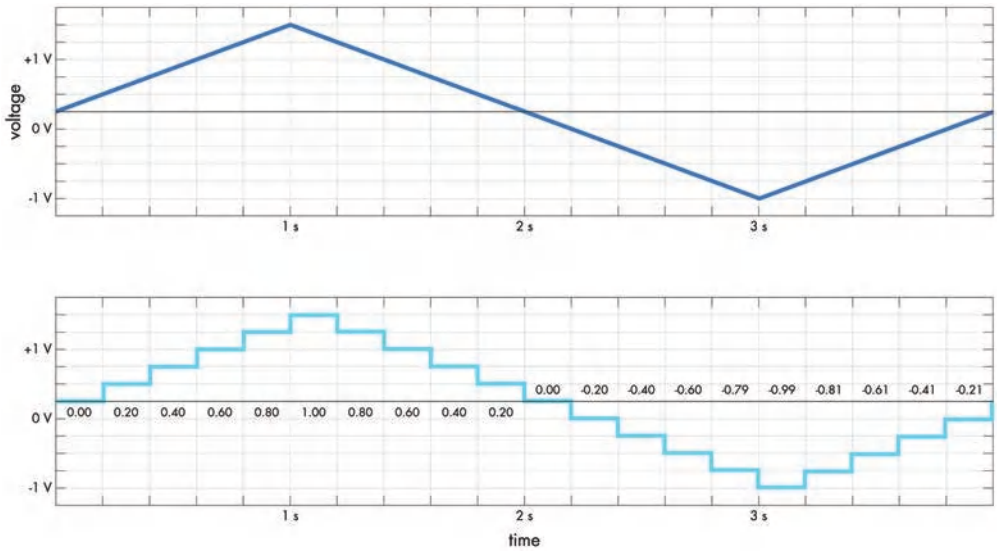


Figure 4-15: A triangle waveform and its 20-step discrete-time approximation. Sampled voltage readings are shown along the center axis.

Applying the techniques discussed when we first introduced the concept of RMS voltage, it's easy to find the V_{RMS} of the sampled staircase waveform. We take the square root of the arithmetic mean of the squares of the 20 measurements. The calculation works out to about 0.583 V. Meanwhile, the theoretical solution for the continuously changing triangle wave is about 0.577 V. In other words, despite picking a comically low time resolution for this demonstration, the error from the discrete-time model is still only about 1 percent.

We'll put this technique of replacing continuous change with a series of discrete time steps to good use multiple times throughout the book, starting with the discussion of capacitors and inductors that's coming up in the next chapter.

To be fair, the method isn't a cure-all. It works well if the solution we're after has an unambiguous representation on a time-based plot, but this isn't always the case. For example, what's the frequency produced by mixing a signal running at f_1 Hz with another waveform running at f_2 Hz? The concept is important for radio receivers, and there's an elegant trigonometric solution for sines (see Chapter 6), but trying to eyeball the formula by squinting at discrete-time approximations isn't fun.

In the end, some problems are best solved symbolically, and the solutions you can most easily derive or look up are for sine waves. This is where our second approximation technique comes in.

Sum-of-Sines Approximation

As it turns out, any complex signal can be expressed as a sum of overlapping sinusoids running at different speeds. For example, Figure 4-16 shows a short series of sine signals that add up to something resembling a square wave.

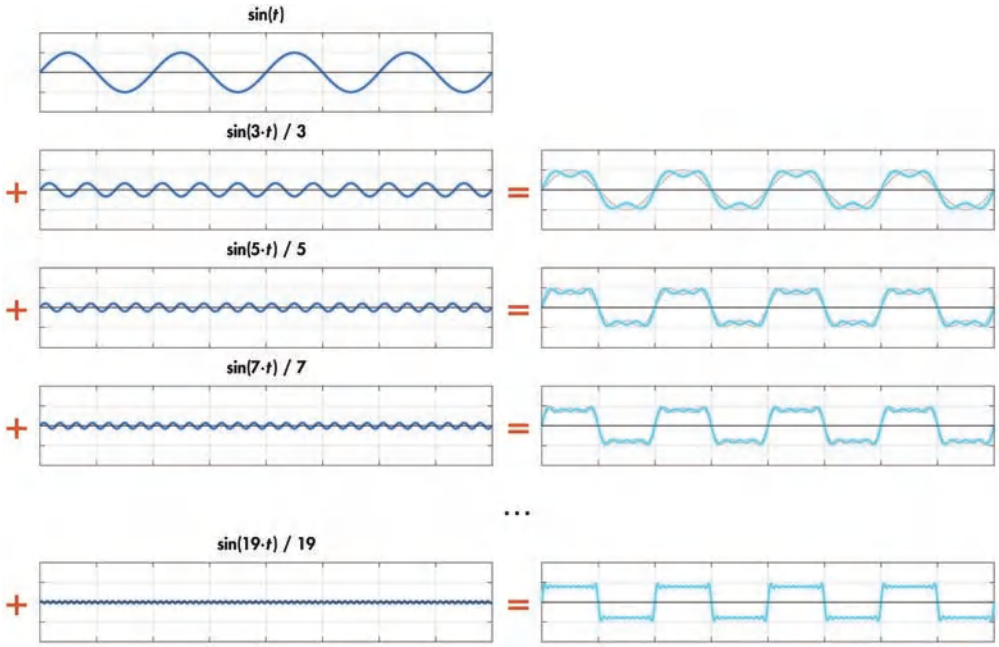


Figure 4-16: Approximating a square wave with a sum of sines

To get a sense of why this technique works, have a look at the second row of the figure. We can observe that zero-crossings of the original sin(t) waveform are aligned with the new 3t signal, so in these regions, the patterns add constructively and produce a steeper slope. At the same time, the 3t component has the opposite polarity near the peaks of sin(t), resulting in destructive interference that flattens the humps.

It is not just square waves that can be approximated this way. There is a general algorithm, known as the *Fourier transform*, which allows arbitrary waveforms to be represented as sums of sines. We won't be covering it in detail here, but if you're interested in how it works or how to implement it on a computer, I maintain an in-depth blog post explaining the idea: *https://lcamtuf.coredump.cx/blog/fourier*.

For our purposes, the aforementioned approximation of a square wave is the most useful application of this technique. Square waves generated by integrated circuits are about as ubiquitous as sinusoids, and this trick gives us a straightforward way to relate the two. The formula combines a sine running at some base frequency f with a series of sines running at the odd multiples (odd *harmonics*) of f, each scaled to have a proportionately reduced amplitude. A complete expression for a square wave with a peak amplitude of 1 and a frequency of f Hz is:

$$\frac{4}{\pi} \cdot \left[\frac{\sin(2\pi f \cdot t)}{1} + \frac{\sin(2\pi f \cdot 3t)}{3} + \frac{\sin(2\pi f \cdot 5t)}{5} + \frac{\sin(2\pi f \cdot 7t)}{7} + \ldots \right]$$

Although the sum should be infinite for a flawless approximation, for most intents and purposes the waveform looks close enough by adding two more terms for 9t and 11t. At that point, mean error is less than 10 percent and the waveform's rising and falling edges are within 3 degrees of vertical. This empirical criterion lends itself

to a useful rule of thumb: If you want to get a sense of how an analog circuit might handle square waves at some frequency f, you can often learn enough by examining its response to sine waves with frequencies of f and $11f$.

Now that we have a solid grasp of signals that change over time, along with a toolkit for modeling sine waves and arbitrary waveforms alike, we're finally ready to investigate the behavior of two types of electronic components that interact with these signals in ways more complex than a resistor would. That's where we'll turn in the next chapter.

5

IDEAL INDUCTORS AND CAPACITORS



If Chapter 4 felt abstract, we're now due for a small payoff. In this chapter, we'll take a step back from springs and pure math and focus on two fundamental, time-dependent behaviors that govern the performance of most electronic circuits: capacitance and inductance.

Although the concepts I'm about to introduce apply more broadly, the formulas in this chapter offer practical insights into how capacitors and inductors work. The models we'll use are idealized, but they're often good enough to reason through circuit design tasks. In Chapter 8, we'll tighten everything down by focusing on how real components tend to diverge from theory.

Capacitance

Capacitance (C) measures the ability of an electronic component, a *capacitor*, to store potential energy in an internal electrostatic field. This energy is stored in response to an applied electromotive force known as a voltage. Although the underlying physics are different, charging a capacitor is conceptually akin to the earlier example of squeezing a spring. The concept of capacitance, meanwhile, is functionally similar to the spring's stiffness coefficient, discussed in Chapter 4.

In more precise terms, the process of charging a capacitor involves applying an external electromotive force that shuffles electrons between two conductive plates separated by a thin insulating film. This makes one of the plates positively charged and the other one negatively charged. Due to the proximity of the plates and the carefully chosen properties of the insulator, the resulting electrostatic field is almost entirely contained within the very narrow gap between the plates, and it largely cancels

out from the vantage point of more distant electrons on the wire. Because of this, the inherent force that counters the separation of charges is diminished, and a fairly substantial amount of energy can be stored by applying a modest voltage (Figure 5-1). For this effect to work, we must maintain symmetry: The amount of surplus charge pushed onto one plate must match the deficit created on the other plate.

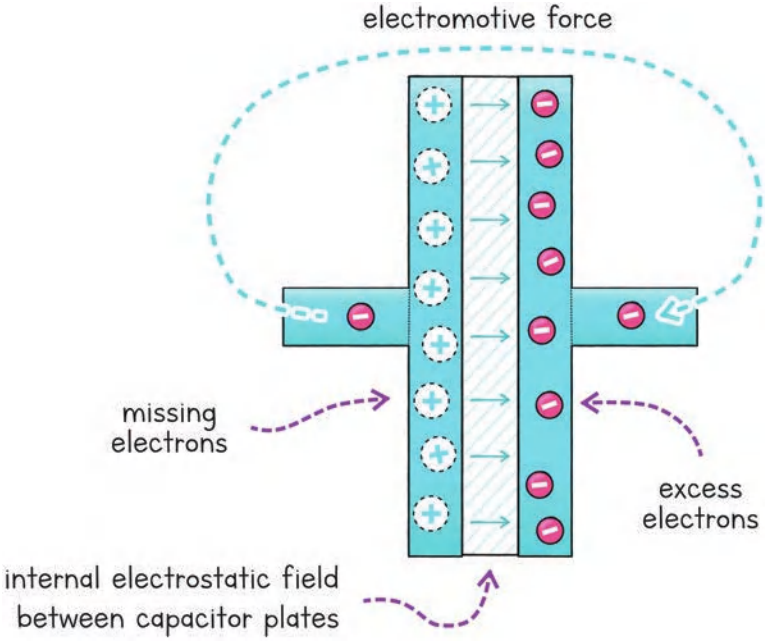


Figure 5-1: The basic structure of a capacitor

Even so, the internal pushback voltage is merely diminished, not eliminated. It gradually builds across the capacitor's terminals in proportion to the amount of charge stored. Once its magnitude matches the externally applied voltage, the net force experienced by electrons on the wire drops to zero and the charging process stops. How quickly that equilibrium is reached depends on how much current you're willing to supply. If the answer is "as much as it takes," the equalization between the supplied voltage and the capacitor's charge level is essentially immediate.

Capacitance is a measure of how much charge can be stored in a given capacitor as a function of the applied voltage. The parameter depends on the surface area of the plates, the thickness of the insulator, and the material the film is made of. A capacitor that stores 1 coulomb when subjected to 1 volt has a rating of 1 farad (F); more coulombs means higher capacitance. The generalized formula is:

$$C = \frac{Q}{V}$$

The farad is a big unit; the capacitors you're most likely to encounter will range from picofarads (pF) to microfarads (µF). Again, we'll talk more about the classification and performance of real-world capacitors in Chapter 8, but for now, let's dig deeper into the behavior of these components in mathematical terms.

Capacitor Voltage vs. Current

In Chapter 3, we defined 1 ampere as the flow of 1 coulomb of charge per second. This gives us a more practical way to think about capacitance: A 1 F capacitor, when supplied with a constant 1 A current for 1 second, will develop a voltage of 1 V across its terminals. The derivation of this rule is shown here:

$$\text{Definition of current}: \; I = \frac{Q}{t} \;\Rightarrow Q = I \cdot t$$

$$\text{Definition of capacitance}: \; C = \frac{Q}{V} \;\Rightarrow V = \frac{Q}{C}$$

$$\text{Combined form}: \; V_{cap} = \frac{I \cdot t}{C}$$

As discussed in Chapter 4, we usually work with voltage-based signals. Because of this, it can be useful to rearrange the equation in a way that gives us the mean charging current that must be supplied over time t for the capacitor to reach a chosen voltage:

$$I = C \cdot \frac{V_{cap}}{t}$$

It doesn't matter how we ramp the current up or down. As long as the average value matches what's given by the formula, the correct amount of charge will be moved onto the capacitor's plates.

The equation works if we start with a partly charged capacitor too. Regardless of the starting state, if we wish to increase the voltage by Δv in some time slice Δt, we can calculate the average additional charging current (i) that must be supplied to reach the goal:

$$i = C \cdot \frac{\Delta v}{\Delta t}$$

This gives us precisely what we need to model the behavior of a capacitor over time.

NOTE *The use of lowercase* v *and* i *in electrical engineering isn't standardized, but the symbols are often used to label small, incremental voltages and currents. The idea is that these parameters can be examined in a simplified way without needing to choose a precise reference point or to account for the non-perfect "large-signal" behaviors of the underlying components.*

A Discrete-Time Capacitor

Capacitors are frequently used to process sinusoidal signals, especially to filter out certain frequencies while passing others through. But what happens to a capacitor when it's subjected to a sinusoidal AC voltage? To find out without too much effort, we can build a discrete-time approximation of the device.

Let's start by generating a sine wave that runs at an arbitrary frequency of f = 50 Hz and has a peak amplitude of V_{peak} = 5 V. The formula for this signal is:

$$V(t) = V_{peak} \cdot \sin(2\pi f \cdot t) = 5 \cdot \sin(100\pi \cdot t)$$

Next, we can set up a simulation that divides each cycle of the waveform into 20 ticks, resulting in a time resolution of 1 millisecond. In each step, we calculate Δv by subtracting the current value of the sine waveform from the capacitor voltage recorded in the previous time slice. This increment determines the momentary current that must flow to reach the target voltage across the capacitor's terminals.

For the purpose of the simulation, we can choose an arbitrary capacitance value C for our capacitor—say, 100 µF—and then calculate the mean charging current (i) for every time slice using the formula derived before ($i = C \cdot \Delta v / \Delta t$). The result of the simulation is shown in Figure 5-2.

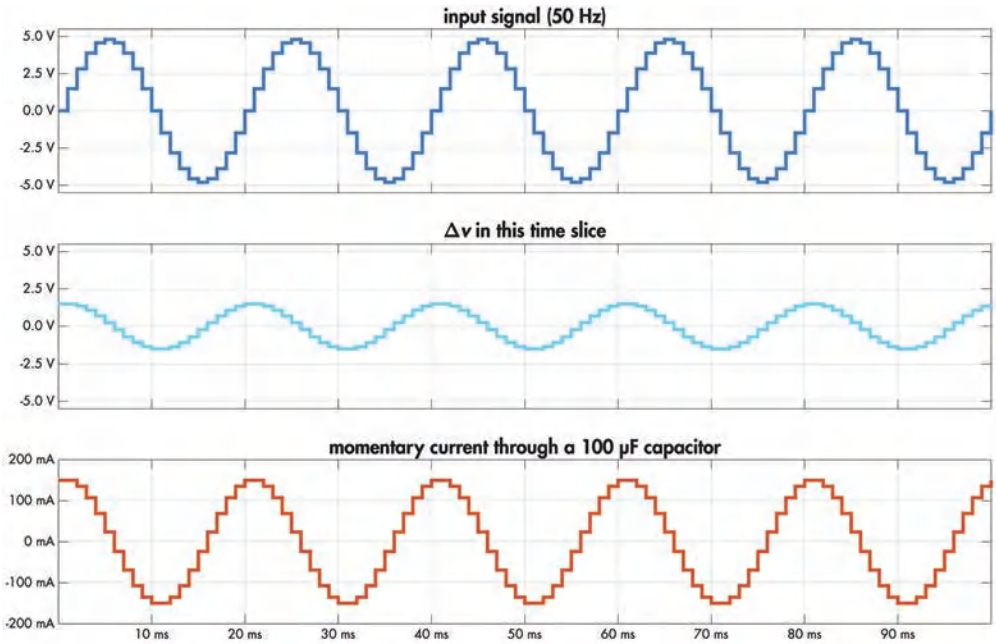


Figure 5-2: A discrete-time model of a capacitor, with 20 samples per cycle of the sine wave

The model shows that there's a sustained back-and-forth motion of charges in response to the applied alternating voltage. This flow of electricity on both sides of the capacitor—a sequence of currents that repeatedly charge and discharge the capacitor—can perform meaningful work. In this sense, the capacitor is a marvelous device that allows alternating currents to seemingly pass through, even though no electrons can hop across the nonconductive layer inside.

This point is worth underscoring. In Chapter 1, we noted that conduction has less to do with the travels of individual electrons and more to do with the propagation and equalization of electric fields. Conductors serve as an efficient medium for this process, but a capacitor is a prime example that electrons don't need to physically make it from point A to point B to get stuff done, and it's precisely what makes these components useful.

Circling back to the simulation, the charging current we're observing is evidently a sinusoid too, except that it's shifted left by one-fourth of a cycle (90°) in relation to the applied voltage. In other words, it appears to be described by a cosine instead of

a sine. This makes sense because in the $i = C \cdot \Delta v / \Delta t$ formula, $\Delta v/\Delta t$ describes the rate of change of the input waveform. Recall from Chapter 4 that the rate of change of $\sin(t)$ is $\cos(t)$.

On the bottom of Figure 5-2, we can peg the peak current at around 150 mA. If we rerun the simulation with a slower-running input signal (f = 10 Hz), the current drops to about 30 mA (Figure 5-3). This is also fairly intuitive. For a smooth sine wave, the rate of change for the input signal decreases at lower frequencies because the waveform gets stretched in time.

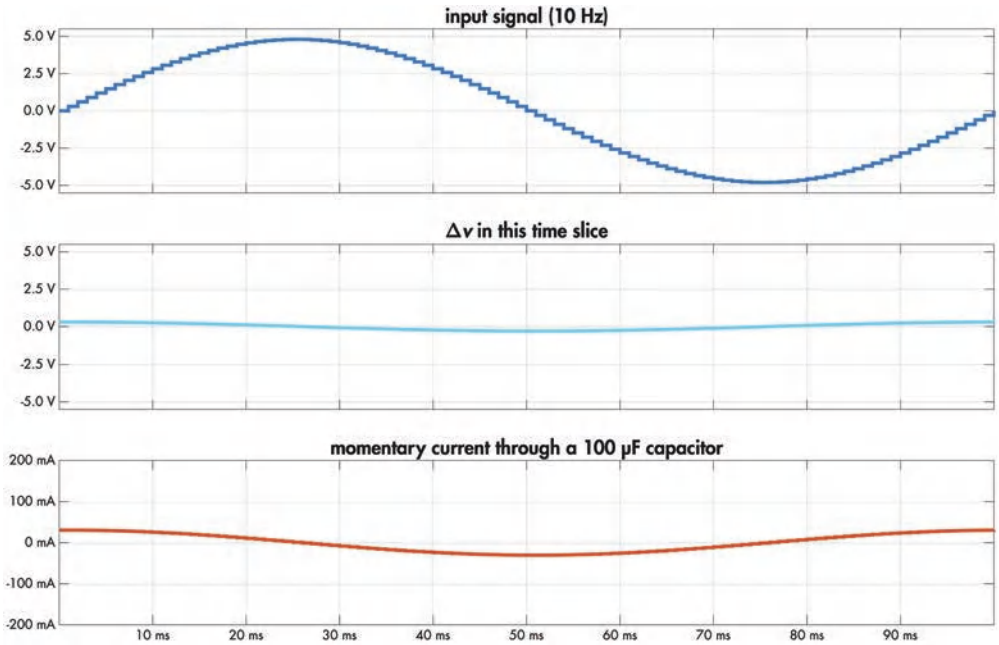


Figure 5-3: The same situation but with a 10 Hz input signal

The main insight from the simulation is that a capacitor, if placed in series with a signal, admits alternating currents but limits their magnitude in a frequency-dependent way. This effect warrants a name and more precise analysis, so let's examine it now.

Capacitive Reactance

Capacitive reactance (X_C) describes the relationship between the amplitude of the applied sine wave and the current that's allowed to flow through a capacitor:

$$X_C = \frac{V_{peak}}{I_{peak}}$$

The quantity is quite similar to resistance ($R = V/I$) and is expressed in the same unit, ohms (Ω). Even so, it's a distinct phenomenon. Unlike resistance, I_{peak} and V_{peak} aren't aligned in phase; as noted earlier, they're one-fourth of a period apart. To put it another way, the two peak values are never observed at the same time.

Although the formula for X_C describes a simple relationship, it's not particularly useful as stated. We typically know the value of V_{peak} for a given input signal, but without running a simulation, it's not immediately clear what the value of I_{peak} may be. Luckily, in Chapter 4 we established that the rate of change of a sine function is highest when its value crosses zero. Because the rate of change of the input voltage decides the charging current, this is where we can solve for that unknown value of I_{peak}.

One zero crossing happens at $t = 0$, so leaning on the familiar calculus trick of assuming infinitesimal Δt, we can solve for I_{peak} as the momentary current in the time slice between $t = 0$ and $t = 0 + \Delta t$:

$$\text{Starting voltage at } t = 0: \quad V(0) = 0$$

$$\text{Ending voltage at } t = 0 + \Delta t: \quad V(0 + \Delta t) = V_{peak} \cdot \sin(2\pi f \cdot \Delta t)$$

$$\text{Voltage delta}: \quad \Delta v = V(0 + \Delta t) - V(0) = V_{peak} \cdot \sin(2\pi f \cdot \Delta t)$$

$$\text{Resulting current}: \quad i = C \cdot \frac{\Delta v}{\Delta t} = C \cdot \frac{V_{peak} \cdot \sin(2\pi f \cdot \Delta t)}{\Delta t}$$

Alas, the formula appears to have some sort of messy dependency on Δt. To get rid of it, recall another trick we leaned on in Chapter 4. At zero crossings, the slope of the sine function is exactly 45°, making $y = \sin(x)$ (very briefly) equivalent to the equation for a 45° diagonal line, $y = x$. Of course, the curvature of $\sin(x)$ changes over time. But if Δt is infinitesimal, then at the exact moment we're interested in, we're free to consider $\sin(x)$ the same as x:

$$i \rightarrow C \cdot \frac{V_p \cdot \cancel{\sin}(2\pi f \cdot \Delta t)}{\Delta t} = C \cdot \frac{V_p \cdot 2\pi f \cdot \cancel{\Delta t}}{\cancel{\Delta t}} = V_p \cdot 2\pi f \cdot C \text{ (as } \Delta t \rightarrow 0)$$

This gives us a clean solution for I_{peak} that's no longer stuck in the domain of infinitesimal time increments. We can plug it into the earlier "conceptual" formula for X_C to get an eminently useful result:

$$X_C = \frac{V_{peak}}{I_{peak}} = \frac{\cancel{V_{peak}}}{\cancel{V_{peak}} \cdot 2\pi f \cdot C} = \frac{1}{2\pi f \cdot C}$$

This tells us that the extent to which a capacitor attenuates currents induced by a sine-wave signal is inversely proportional to signal frequency and the component's rated value (C). The formula describes one of the most important functions of a capacitor, so it's one of the few equations worth committing to memory.

Whenever you use this formula, remember the earlier warning: For a given f and C, capacitive reactance walks and quacks like resistance, but they're not the same. In a resistor, the current flowing through the device stays in lockstep with the applied voltage. In a capacitor, as we've shown, the current peaks one-fourth of a cycle ahead of the voltage sinusoid. This adds some nuance to calculating the cumulative effects of R and X_C. For example, if you have a signal applied to a resistor in series with a capacitor, a simple sum of these two values won't do.

Inductance

Inductance (L) measures the ability of a component—an *inductor*—to store the "kinetic" energy of the current flowing through it. The energy is stored in an internal magnetic field. An inductor exhibits an inertia-like effect that opposes changes to I; the behavior is conceptually similar to the effort needed to accelerate or slow a moving object. Higher inductance is akin to dealing with an object of greater mass.

Inductors are usually made from a length of wire wrapped around a ferromagnetic core. The flow of current is inseparably linked to a magnetic field, a phenomenon we'll discuss in Chapter 6. In ferromagnetic materials, this field couples to microscopic, randomly oriented *magnetic domains*—atomic-scale magnets that make up the material and can be rotated by applying an external force. The coupling prevents the field from ramping up or down until some energy is expended to change the alignment of a number of magnetic domains. This manifests as a pushback electromotive force proportional to the rate of current change ($\Delta i/\Delta t$) and the component's rated inductance (L):

$$V = L \cdot \frac{\Delta i}{\Delta t}$$

The unit of inductance, 1 henry (H), corresponds to the behavior of a device that, when subjected to a current rising at a rate of 1 A per second, opposes this change by developing 1 V across its terminals. As with the farad, the henry is large; common inductors range from nanohenrys (nH) to millihenrys (mH).

To better understand how an inductor responds to voltage-based signals, we can reorder the formula to solve for the increment of current (Δi):

$$\Delta i = \frac{V \cdot \Delta t}{L}$$

This tells us that the current flowing through an inductor increases in proportion to the applied voltage and time. The implication is that if we apply a steady voltage to the terminals of an inductor, the current will keep climbing indefinitely. In practice, something would have to give!

A Discrete-Time Inductor

The $\Delta i = V \cdot \Delta t \,/\, L$ equation allows us to build a discrete-time simulation of an inductor in a manner similar to what we did for a capacitor. First, we generate a stair-step approximation of a sine wave with a peak amplitude of V_{peak}:

$$V(t) = V_{peak} \cdot \sin(2\pi f \cdot t)$$

Next, we calculate the increment in current (Δi) for each time slice. Finally, to convert from these increments to the actual current flowing through the inductor's windings, we add the consecutive deltas to a running sum. For L = 100 µH and f = 50 kHz, the simulation results in the plot shown in Figure 5-4.

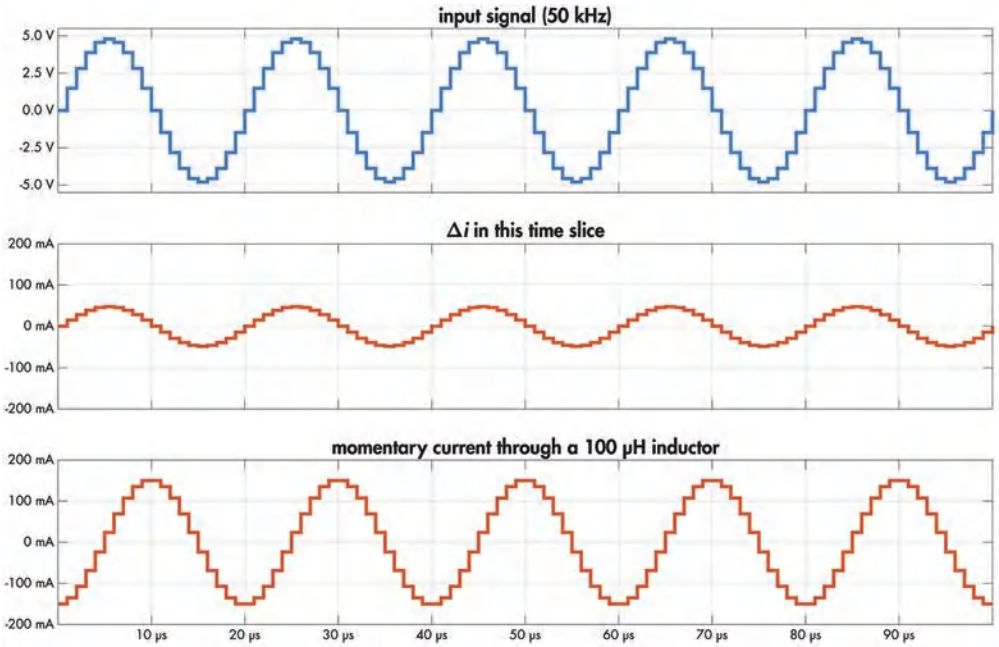


Figure 5-4: A discrete-time model of an inductor, calculated with the resolution of 20 steps per cycle of the input waveform

The running sum appears to be a sinusoid, albeit in contrast to the capacitor, and the current waveform is shifted to the right. This has to do with the observation we made in Chapter 4: The rate of change of $\sin(x)$ is described by $\sin(x + 90°)$.

According to this rule, if we start with a current waveform, we know that its rate of change—the Δi signal—will be shifted one-fourth of a cycle to the left. In this instance, however, the fixed reference point is the Δi signal itself: From the fundamental inductor equation, we know that it must stay in lockstep with the input voltage. What we're after is the underlying current waveform that these Δi increments describe. Since we're working back from the rate of change, it follows that the summed "whole current" waveform must be given by $\sin(x - 90°)$.

Inductive Reactance

Analogously to a capacitor, an inductor passes through sinusoidal signals, but it limits the current in a frequency-dependent way. *Inductive reactance* (X_L) describes the relationship between the amplitude of the applied sine wave and the admitted current. It's conceptually similar to capacitive reactance and is governed by the same high-level formula:

$$X_L = \frac{V_{peak}}{I_{peak}}$$

To express reactance in more useful terms, we can follow an approach similar to what we did for capacitors. We start by modeling a sinusoidal input current:

$$I(t) = I_{peak} \cdot \sin(2\pi f \cdot t)$$

The current flowing through an inductor is phase-shifted in respect to voltage, but to keep the math tidy, let's just say that the underlying voltage waveform happens to be shifted in a way that allows the current to be described as a pure sine.

We know that the sin(x) function has a zero crossing at $t = 0$, so this is where the current is changing the fastest. We can find this maximum rate of change, which I'm going to shorten to m, by employing the familiar "infinitesimal Δt" method:

$$I(0) = 0$$

$$I(0 + \Delta t) = I_{peak} \cdot \sin(2\pi f \cdot \Delta t)$$

$$m = \frac{I(0 + \Delta t) - I(0)}{\Delta t} = \frac{I_{peak} \cdot \sin(2\pi f \cdot \Delta t)}{\Delta t}$$

As before, we can also lean on the tried-and-true $y = x$ simplification that lets us "cancel" the sine. After this, we can cancel the Δt term that appears both in the numerator and the denominator. The result is:

$$m = I_{peak} \cdot 2\pi f$$

With this result, we circle back to the fundamental inductor equation, which tells us that the voltage that develops across the inductor is proportional to inductance (L) and the rate of change in current ($\Delta i/\Delta t$):

$$V = L \cdot \frac{\Delta i}{\Delta t}$$

Since we know the peak rate of change of current (m), the formula trivially nets us the peak voltage:

$$V_{peak} = L \cdot m = L \cdot I_{peak} \cdot 2\pi f$$

We're after the formula for reactance, which is equal to $V_{peak} \,/\, I_{peak}$. Substituting V_{peak} from the preceding equation, we get the final answer:

$$X_L = \frac{V_{peak}}{I_{peak}} = \frac{L \cdot \cancel{I_{peak}} \cdot 2\pi f}{\cancel{I_{peak}}} = 2\pi f L$$

Similar to the formula for X_C, this is an important result. Along with capacitors, inductors are essential in signal processing, and the equation lets us understand the extent to which a sine-wave signal of a particular frequency will be attenuated or amplified by various arrangements of analog parts.

Impedance

Impedance (Z) represents the combined effects of resistance and reactance (capacitive and inductive) for a given arrangement of components. Like all the constituent quantities, the value is expressed in ohms and describes the same relationship between the applied voltage and the current flowing through.

Much of the time, the overall effect is dominated by one of the three parameters; the next time you encounter the term outside this book, it might be a stand-in for R, X_C, or X_L. As long as you're working with just one of these, the math is pretty simple. For example, if you string together three resistors in series, the current must push through this entire obstacle course, so the overall impedance is a simple sum of the per-component resistance values:

$$Z_{total} = R_{total} = R_1 + R_2 + R_3$$

The same logic is applicable to three capacitors or three inductors in series:

$$Z_{total} = X_{C1} + X_{C2} + X_{C3}$$

$$Z_{total} = X_{L1} + X_{L2} + X_{L3}$$

The situation gets more complicated if you have a mix of different effects at play. To explain the rules (and their origins), let's have a quick peek at three basic combinations of components connected in series: an inductor plus a capacitor (L-C), a resistor and an inductor (R-L), and a resistor connected to a capacitor (R-C). We'll lay the groundwork now and flesh out some of the finer points in Chapter 12.

L-C in Series

Let's start with the case of an inductor connected in series with a capacitor. Earlier in the chapter, we observed that the current flowing through a capacitor is shifted to the left in relation to the applied sinusoidal voltage; the offset amounted to adding +90° to the function's timing expression. For an inductor, the offset worked out to −90°. How do we reconcile that?

Well, in this instance, we don't! If the components are connected in series, head-to-tail, the current flowing through both of them must be the same. What comes in must come out—there's no place for electrons to chill. The expected phase shift still occurs, but it's the voltages across the components' terminals that get out of sync in relation to the common current waveform. The voltage across the inductor will be one-fourth of a period ahead of the applied current. The voltage across the capacitor will be one-fourth of a period behind. Their sum will need to match the supply, but there's nothing that requires the individual values to be in lockstep with each other.

Let's describe the sinusoidal input current using the following simplified formula:

$$I(x) = I_{peak} \cdot \sin(x)$$

We don't know the value of I_{peak}, but it's not relevant to the solution. Neither is the $2\pi f$ part of the timing expression, omitted here for brevity.

As discussed earlier in the chapter, capacitive and inductive reactance both describe the same relationship between voltage and current: V_{peak} / I_{peak}. It follows that reactance multiplied by peak current yields the amplitude of the voltage waveform, the shape of which is given by an appropriately phase-shifted sine. Keeping this in mind, we can construct a pair of waveforms for the phase-shifted voltages that are bound to appear across the terminals of the inductor and the capacitor:

$$V_L(x) = X_L \cdot I_{peak} \cdot \sin(x + 90°)$$

$$V_C(x) = X_C \cdot I_{peak} \cdot \sin(x - 90°)$$

Because $\sin(x + 90°)$ is the same as $\cos(x)$ and $\sin(x - 90°)$ is the same as $-\cos(x)$, we can rewrite these as follows:

$$V_L(x) = +X_L \cdot I_{peak} \cdot \cos(x)$$

$$V_C(x) = -X_C \cdot I_{peak} \cdot \cos(x)$$

Voltages in series sum in a straightforward way; two 1.5 V batteries connected end-to-end give us 3 V. That is to say, the overall voltage manifesting itself across the endpoints of a series L-C circuit is:

$$V_{LC\,(\text{series})} = V_L(x) + V_C(x) = (X_L - X_C) \cdot I_{peak} \cdot \cos(x)$$

This tells us that in a series L-C circuit, the effects of X_L and X_C don't add constructively; instead, they subtract! To get the combined impedance, we can just pluck out the $X_L - X_C$ expression verbatim:

$$Z_{LC\,(\text{series})} = X_L - X_C$$

When we plot any functions that describe the attenuation or amplification of signals, we usually rely on a logarithmic scale on both axes, known as a *log-log plot*. The scale is uniquely suited for the task: A 10× increase is always represented by the same distance on the plot, no matter if the starting value is 1, 10, or 100. A sample log-log plot of the absolute value of $Z_{LC\,(\text{series})}$ across a range of signal frequencies is shown in Figure 5-5.

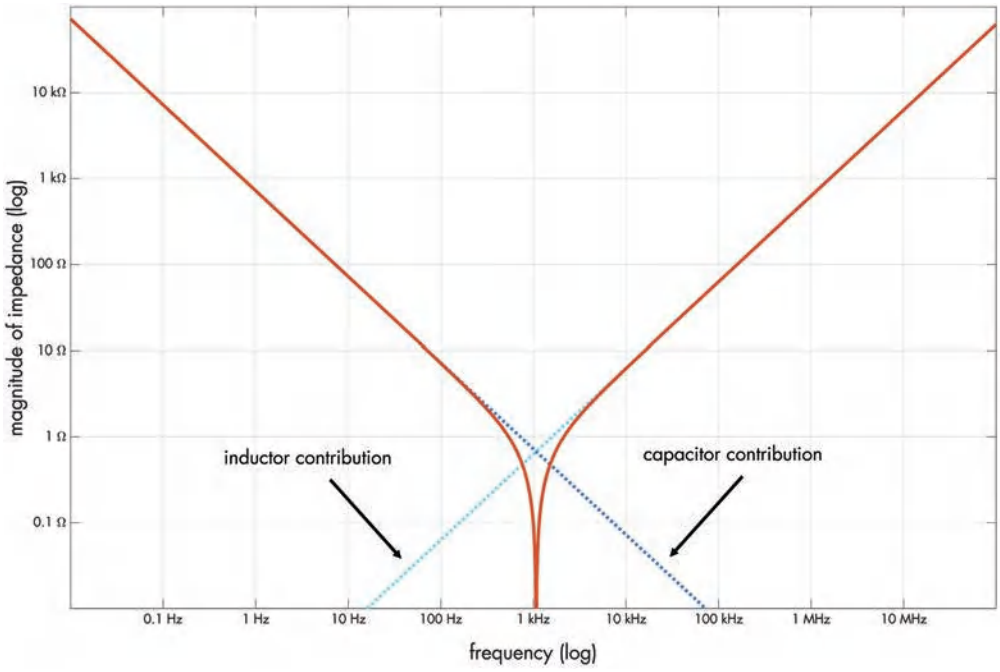


Figure 5-5: The absolute magnitude of impedance of a 220 µF capacitor and a 100 µH inductor in series (log scale)

On this graph, impedance is dominated by the effects of the capacitor all the way to several hundred hertz. We can see that the curve initially clings to the dashed line representing $X_C = 1/(2\pi fC)$. Later, at several kilohertz, inductive reactance takes over. From that point on, the curve follows the $X_L = 2\pi fL$ line and the capacitor has almost no say.

But that's not all. Around 1 kHz, there's a pronounced dip where the value of $Z_{LC\,(\text{series})}$ apparently drops to zero. This is where the value of X_L is equal to the value of X_C; the voltage developing across the capacitor is the same as the voltage across the inductor, except it has the opposite sign. Whenever the inductor tries to impede the flow of current by developing a voltage across its terminals, the voltage across the capacitor swings in the opposite direction by exactly the same amount, and in the end, the flow of current theoretically isn't impeded in any way. The point where this happens is known as the *resonant frequency* of the circuit, and we can calculate it as follows:

$$\text{Assumption}: \; X_L = X_C$$

$$\text{Formulas expanded}: \; 2\pi fL = \frac{1}{2\pi fC}$$

$$\text{Moving } f \text{ to the left}: \; f^2 = \frac{1}{4\pi^2 \cdot C \cdot L}$$

$$\text{Final expression}: \; f_{dip} = \frac{1}{2\pi\sqrt{LC}}$$

If you have a signal generator, an oscilloscope, and some capacitors and inductors at hand, you can observe this effect in real life. Use the formula for f_{dip} to screen for practical component values. For example, a 10 µF capacitor in series with a 10 µH inductor should have a dip around 16 kHz. You can then program a signal generator to output a sine at a particular frequency, with an inductor and a capacitor connected in series between the device's terminals. The voltage across the terminals of the signal generator can be monitored with an oscilloscope. Near the magic frequency, the line should get more or less flat.

If you decide to perform the experiment and plot signal amplitudes across different frequencies, it might feel more natural to do so on a linear scale (Figure 5-6).

Plotted linearly, the curves will appear somewhat distorted and harder to read, which is precisely why log-log plots are preferred for the task. Still, the effect is recognizable either way.

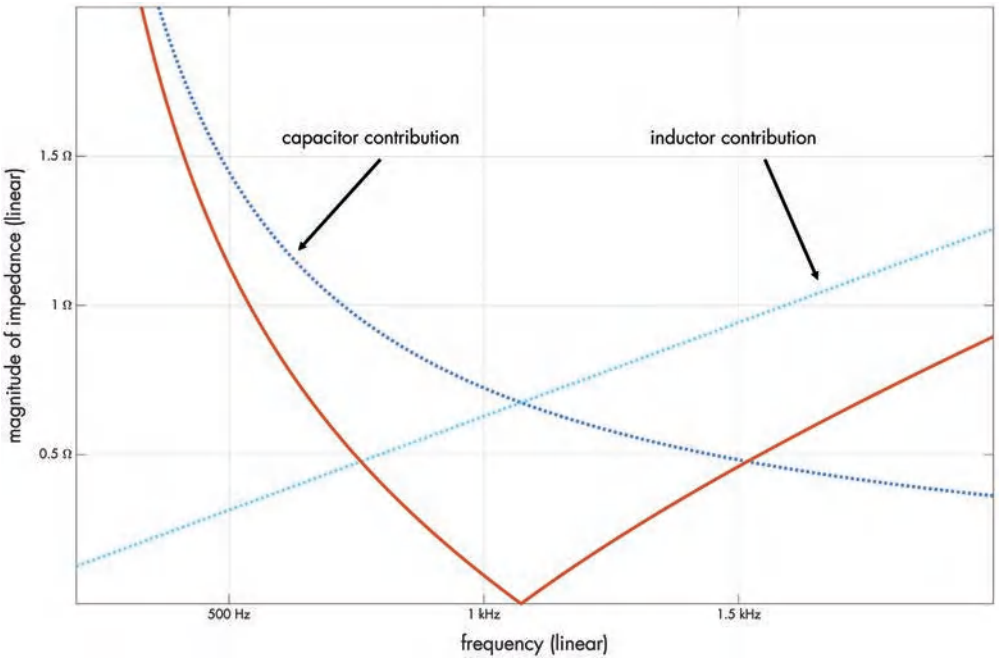


Figure 5-6: Series R-L impedance curve, zoomed in and rendered on a linear scale

R-L and R-C in Series

In the previous section, we developed a simple formula for the impedance of a circuit where the reactances are 180° out of phase. The math gets a bit more complicated if we throw a resistor into the mix. We'll take a series R-L circuit as an example, but the reasoning is precisely the same for series R-C circuits.

The complication is that in contrast to an inductor, the current flowing through a resistor has no phase shift, so what comes in as a $\sin(x)$ also comes out as a $\sin(x)$. The model of voltages for a series R-L circuit is therefore:

$$V_L(x) = X_L \cdot I_{peak} \cdot \cos(x)$$

$$V_R(x) = R \cdot I_{peak} \cdot \sin(x)$$

Analogously to the L-C case, we can try to solve for the sum of voltages manifesting across the terminals, but this time we won't get far:

$$V_{RL\,(\text{series})} = V_L(x) + V_R(x) = I_{peak} \cdot \left(R \cdot \sin(x) + X_L \cdot \cos(x)\right)$$

No matter how we slice and dice it, we have no obvious way to extricate R and X_L to build an equation for impedance; each of them is multiplied by a different waveform. Don't despair, though. A clever solution is shown in Figure 5-7.

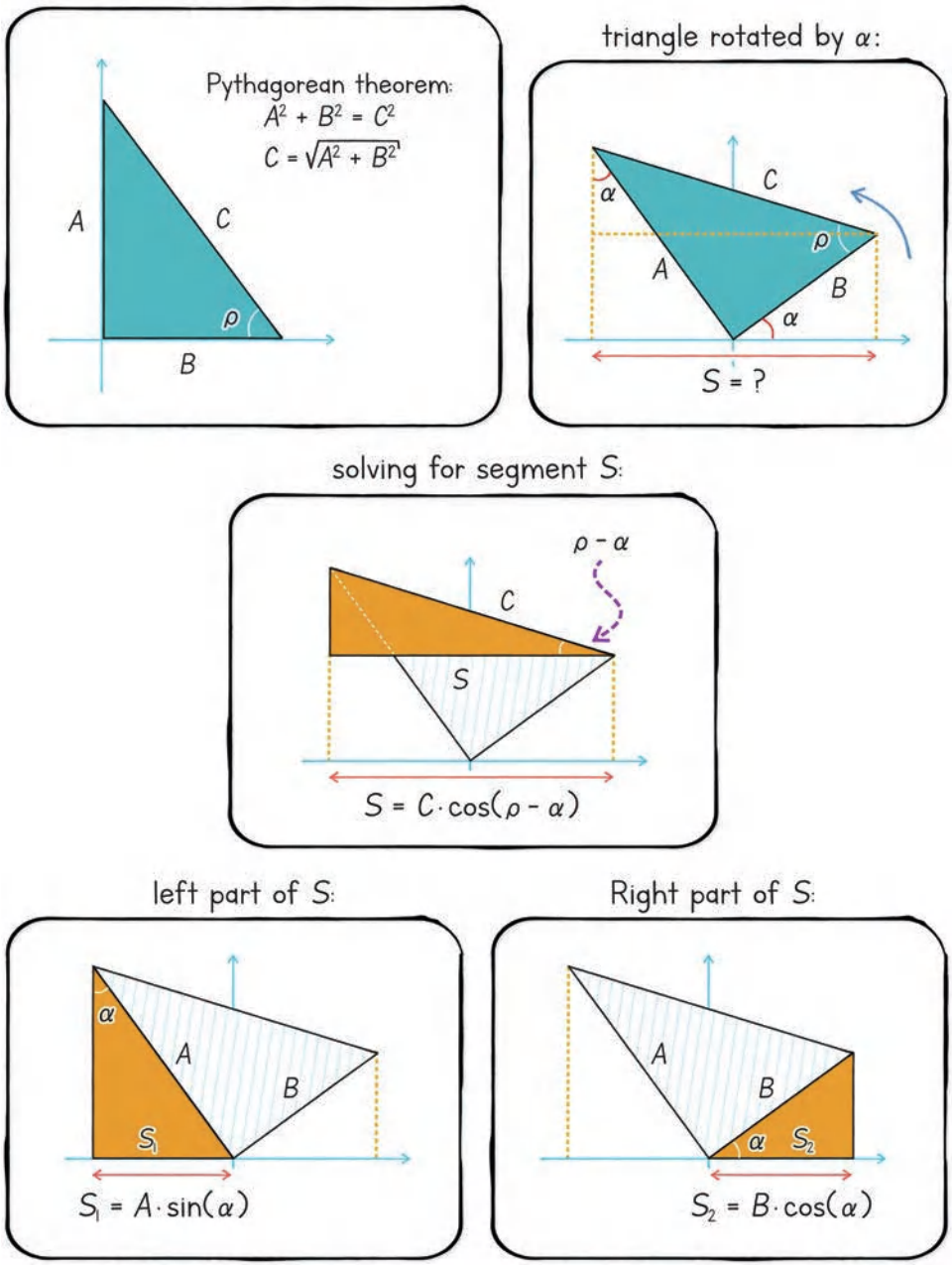


Figure 5-7: Elements of a visual solution to A · *sin(x)* + B · *cos(x)*

Predictably, we start by drawing a right-angled triangle. The triangle has one angle equal to ρ and three sides labeled A, B, and C, the last one being the hypotenuse. Next, we rotate the triangle counterclockwise by α and then draw a projection of the now-tilted hypotenuse onto the horizontal axis. We call this projected segment S. We

can easily calculate its length using the helper triangle added in the middle panel. The length works out to $C \cdot \cos(\rho - \alpha)$.

In the last two panels, we calculate the length of S in an alternative way: as a sum of two subsegments, S_1 and S_2. With the aid of two more helper triangles, we can easily establish that S_1 equals $A \cdot \sin(\alpha)$ and that S_2 comes out to $B \cdot \cos(\alpha)$. This lets us write the following equality:

$$S_1 + S_2 = S$$

$$A \cdot \sin(\alpha) + B \cdot \cos(\alpha) = C \cdot \cos(\rho - \alpha)$$

Further, by applying the Pythagorean theorem to the starting triangle, we can obviously establish the following:

$$C = \sqrt{A^2 + B^2}$$

Therefore, another way to express the solution is:

$$A \cdot \sin(\alpha) + B \cdot \cos(\alpha) = \sqrt{A^2 + B^2} \cdot \cos(\rho - \alpha)$$

This is a remarkable result, because it tells us that a sum of a sine and a cosine of different amplitudes (A and B) but with the same timing variable (α) can be expressed as another sinusoid of the same frequency but with some constant shift ρ. We could try to find ρ, but that's unnecessary at the moment. We have all we need to rewrite the voltage equation into a "good enough" form:

$$\text{Old finding : } V_{RL\,(\text{series})} = I_{peak} \cdot \left(R \cdot \sin(x) + X_L \cdot \cos(x)\right)$$

$$\text{New finding : } A \cdot \sin(\alpha) + B \cdot \cos(\alpha) = \sqrt{A^2 + B^2} \cdot \cos(\rho - \alpha)$$

$$\text{Result : } V_{RL\,(\text{series})} = I_{peak} \cdot \sqrt{R^2 + {X_L}^2} \cdot \cos(\rho - \alpha)$$

Again, we don't care about the input to the cosine function; it's enough to know that it's a well-behaved sinusoid. With that sorted out, we can pluck out the square root part to write the impedance solution for either a series R-L or R-C circuit:

$$Z_{RL,RC\,(\text{series})} = \sqrt{R^2 + {X_{L,C}}^2}$$

It gets better. Note that this is essentially our old friend and savior, the Pythagorean theorem. This tells us that the system plays by the rules of Cartesian geometry. We can choose to interpret Z as the length of a hypotenuse of a triangle on a two-dimensional plane, with R and X corresponding to dimensions along the axes (Figure 5-8). This is a very intuitive way of thinking about impedance—as long as you understand why it works.

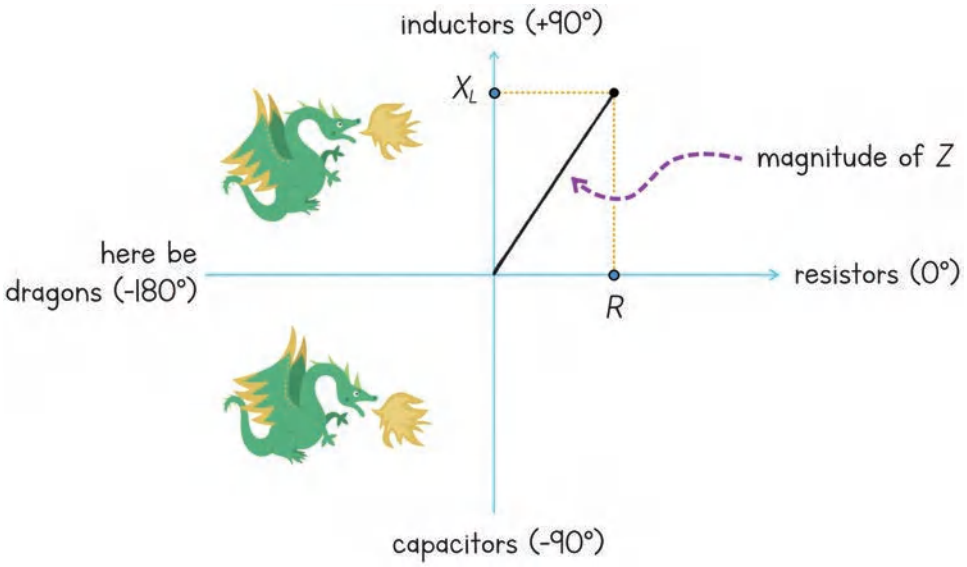


Figure 5-8: Modeling impedance in two dimensions. There are no electronic components with true negative resistance, so only the right half of the plane is being used.

The interpretation also gives us an obvious way to determine the phase shift (θ) between current and voltage. The ratio of the opposite (X) to the adjacent (R) is given by the tangent function: $\tan(\theta) = X/R$. In this instance, we know X and R, but we want to determine θ, so we need the inverse trigonometric function, arctangent:

$$\theta_{RC,RL\,(\text{series})} = \arctan\left(\frac{X}{R}\right)$$

Most calculator apps will give a result in radians by default. If you wish to convert it to degrees, multiply it by $180°/\pi$, or about $57.296°$.

For a sample plot of the impedance of a series R-C circuit, refer to Figure 5-9. Unlike in the L-C case, there's no sudden dip associated with resonance. However, we once again see one portion of the plot that's dominated by reactance and another portion that's dominated by resistive effects.

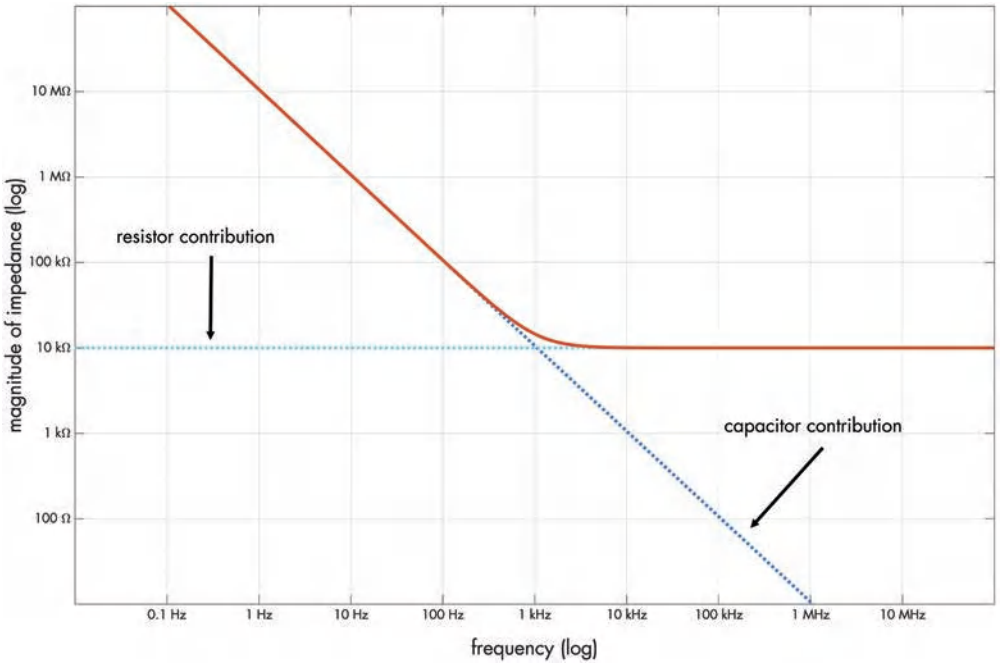


Figure 5-9: The impedance of a 15 nF capacitor and a 10 kΩ resistor in series (log scale)

The knee of the plot—the crossover frequency where the line flattens out—can be found analogously to the L-C case:

$$\text{Assumption}: \quad R = X_C$$

$$\text{Formula expanded}: \quad R = \frac{1}{2\pi f C}$$

$$\text{Reordered}: \quad f_{knee} = \frac{1}{2\pi R C}$$

Again, we'll revisit this approach in Chapter 12. In particular, we'll take a look at how to deal with components connected in parallel, or with more elaborate combinations of R, L, and C.

We're now done with the bulk of the dry theory in the book. We still need to talk about magnetism and radio waves before wrapping up Part I, but after that, we're going to focus on more hands-on work.

6

ELECTRONICS AT THE SPEED OF LIGHT



In Chapter 1, we asserted that electronic signals travel through conductors at nearly the speed of light in a vacuum, in the ballpark of c = 300,000 km/s. We often write this off as "close enough to instantaneous," but the phenomenon can't be wholly dismissed in hobby electronics work. The speed of light is a fundamental constraint on *causality* that affects circuits in ways that extend beyond the introduction of tiny signal delays.

This constraint means we can't send information or perform actions faster than it takes photons to get from here to there, and that holds true in every frame of reference. These limitations create paradoxes for Newtonian physics; Einstein's relativity resolves them by bending the nature of space and time. In this chapter, we'll try to untangle some of the connections between the speed of light, relativity, and electronics. We'll consider the real-world implications these constraints have on circuit design and show how the resulting effects can be harnessed, for example, to transmit signals over radio waves.

Measuring the Speed of Electricity

As noted earlier, signal velocity in electronic circuits is so high that we usually don't bother keeping track of it on tiny circuit boards. That said, there are exceptions to this rule. A straightforward example is a CPU of a modern computer. The CPU talks to memory chips at speeds that exceed 1 GHz, and the period of a 1 GHz waveform is a single nanosecond. In that time, electronic signals on a PCB travel about

18 cm (7 inches). It follows that if one of the signal traces is a couple centimeters longer than the rest, it's enough for signals to get badly out of sync. The fix is usually simple: We add squiggles to shorter PCB traces until all lengths are the same (Figure 6-1).

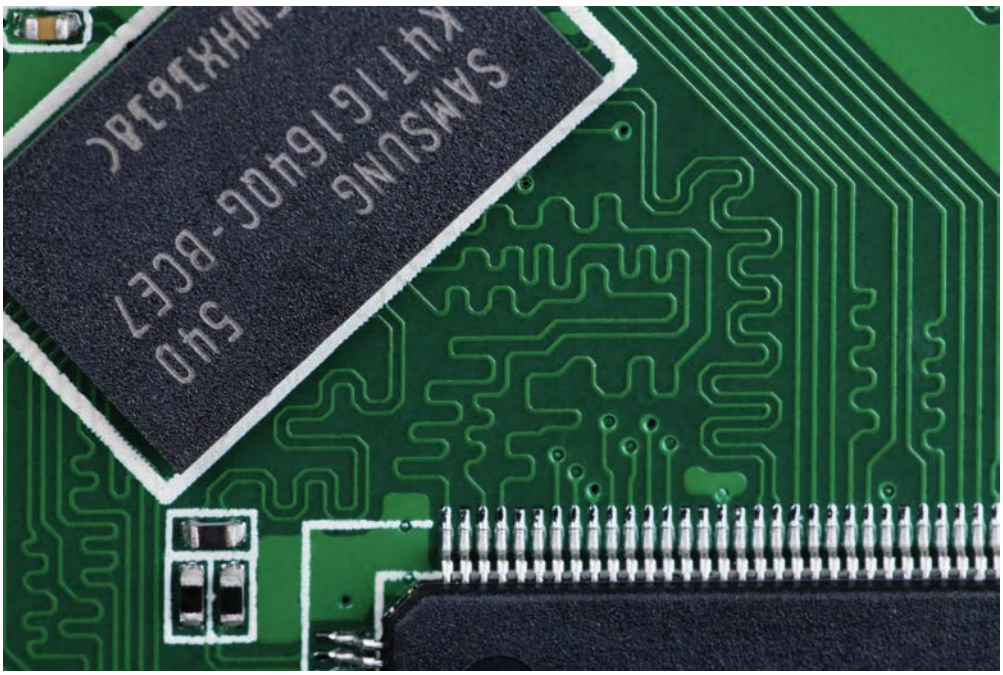


Figure 6-1: Squiggly signal traces in the vicinity of a digital chip

But why trust a book or a Wikipedia article that says electricity travels at close to the speed of light? Let's try a simple experiment to measure how quickly an actual signal propagates through a conductor. If you have an oscilloscope and a signal generator, you can follow along.

For the experiment, you'll need a cable that's about 15 to 30 meters (50–100 feet) long. To minimize crosstalk and radio interference, it's best to use shielded coaxial cable known as RG-58 or RG-8X. As of this writing, you can buy a 100-foot spool on Amazon for as little as $12; look for budget brands such as Monoprice or Cables-Online.

Pick one end as the feed point. On this end, connect the cable's inner conductor to the output of the signal generator and to the oscilloscope's channel A probe; hook up the outer conductor (shielding) to the devices' ground clips. On the far end, place a resistor with a resistance around 50 Ω between the conductors. Next, connect the inner conductor on this side to oscilloscope channel B, and tie the shielding to the channel's ground clip:



We'll get to the purpose of the resistor soon. For now, configure the signal generator to output a 1 MHz square wave with an amplitude of 5 V. You should end up with an oscilloscope trace similar to the one shown in Figure 6-2.

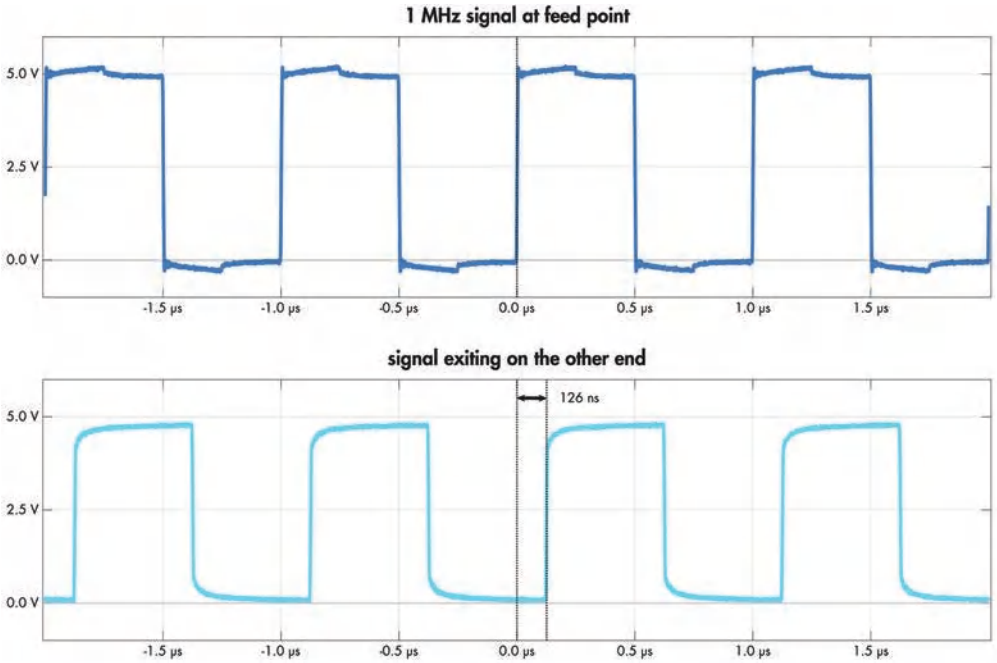


Figure 6-2: A signal entering (top) and exiting (bottom) a 100-foot run of coax

Ignoring a bit of distortion, we're looking at images of the same square wave, except the signal exiting the cable is delayed by 126 nanoseconds compared to the waveform that entered it on the other side. This is evidently how long it took for the electric field to travel from A to B. Based on this observation we can calculate signal velocity:

$$v_s = \frac{30.5\,\text{m}}{126\,\text{ns}} \approx 242{,}100\,\text{km/s}$$

This is roughly 81 percent of the speed of light in a vacuum. The result is fairly typical for coax, although results will vary depending on the insulating material. Solid polyethylene core may yield velocity factors as low as 65 percent, while foamed plastics can approach 90 percent. If we could test a very long PCB trace, the number would be closer to 60 percent.

But hold on. The signal we're sending down the cable is just an electromagnetic wave, not unlike light. Why does this wave propagate more slowly, and why does the speed depend more on the insulating material than the conductor itself?

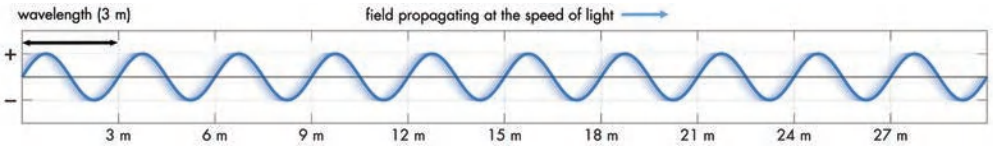
How Signals Interact with Matter

To understand why the electrical signal is slowed, we should first note that light does the same thing. It slows down in substances such as glass. Some online commentators believe this is because light particles collide with atoms and bounce around. Others claim that light is absorbed and then emitted back with a delay. As far as we know, both of these theories are wrong.

There's a set of well-known formulas developed in the late 19th century that mathematically model this and other behaviors of electromagnetic waves. However, the formulas tend to deal in the abstract. It's easy to assert that something must be happening because Maxwell's equations or Fresnel equations say so. It's harder to pinpoint the tangible mechanism behind the result. For the most part, if the equations satisfactorily predict reality, we leave the "why" to philosophers.

For the philosophically inclined reader, I don't have a perfect answer. But the most compelling physical model I've come across is the *Ewald–Oseen extinction theorem*. The concept is quite obscure and has its roots in optics, but it gives a coherent explanation of why electromagnetic waves can be slowed down by insulators as they pass through.

To get the ball rolling, let's assume that all electromagnetic fields always propagate at the speed of light in a vacuum, *c*. When a simple (planar) waveform is moving through space, we can take a snapshot along the direction of its travel and visualize the peaks and valleys of the electric field the following way:



The changing electric field can interact with nearby charged particles. In a nearby insulator, bound electrons can't move far, but many molecules behave somewhat like squish toys. They act as *electric dipoles* that exhibit polarity and can be elastically realigned in response to an external electric field.

To analyze these interactions, let's consider a signal that passes through a relatively thin slab of insulating material. The location of the sheet is marked by the thick vertical line in Figure 6-3. Note that in contrast to earlier time-based plots, the illustration shows a snapshot of a wave propagating through space, left to right.

The electric field associated with the passing waveform aligns the dipoles in the slab. These coherently aligned dipoles in turn produce a secondary, weaker electric field, an echo that radiates away from the surface at the speed of light. It is reasonable to assume that this secondary wave is delayed in phase compared to the primary signal (Figure 6-3, top). The delay can happen simply because the dipoles don't react very quickly. A rigorous mathematical solution is complicated, but we don't need a precise model. It's enough to assert that there's *some* delay, created by *some* conceivable mechanism, and that this delayed signal is then superimposed on top of the original sinusoid.

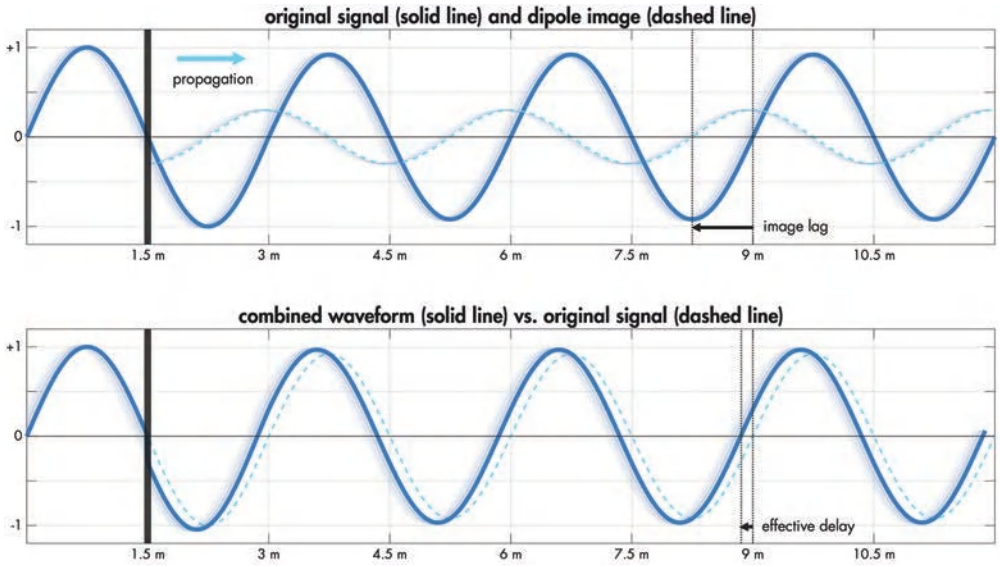


Figure 6-3: A superposition of the original signal and a weaker image produces a new sinusoid.

Toward the end of Chapter 5, we established that summing two sinusoids of different amplitudes and phases produces yet another sine wave with an in-between amplitude and phase. This is what's shown on the bottom of Figure 6-3. The sum of the original waveform and the echo looks just like the starting signal, and it's still moving at the speed of light in a vacuum. But in this view, when the waveform crossed the slab, it evidently acquired a small delay.

Next, let's have a look at a scenario with multiple slabs placed along the path of the signal, essentially a stepwise approximation of a solid. Every time the signal passes through one of the slabs, the interference with the secondary wave adds another delay. If we take a snapshot, the result is a squished sinusoid (Figure 6-4).

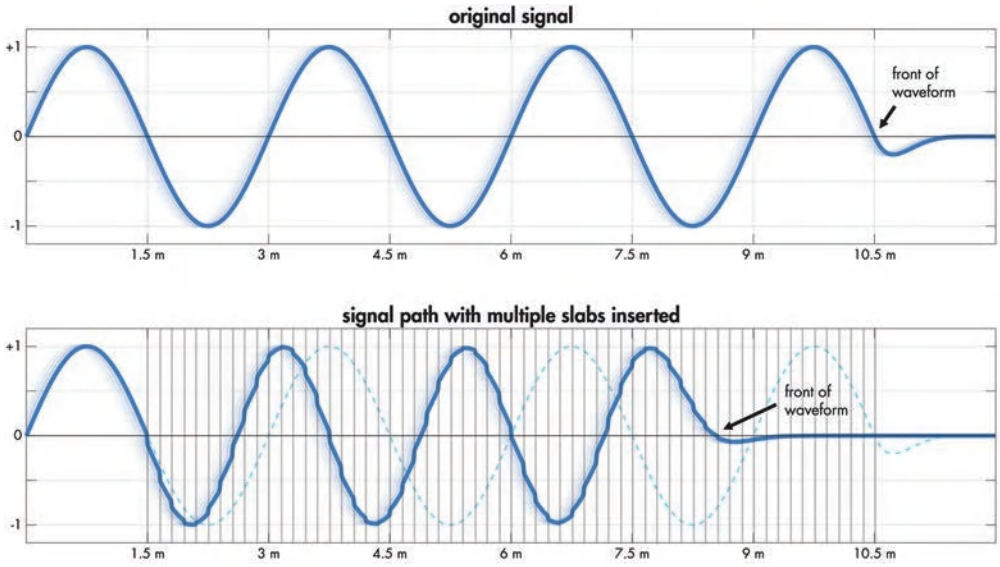


Figure 6-4: Inserting a larger number of thin dielectric (electrically polarizable) layers along the path

The frequency of the squished waveform hasn't changed. The peaks of the signal are entering the solid at the same rate as before, and they exit at the same rate too. Even so, in the instantaneous snapshot we're looking at, the distance between the peaks—the wavelength—is evidently reduced. This may sound paradoxical because wavelength is just speed divided by frequency. If the frequency is the same as before, how come the wavelength has changed? To answer that question, notice that the front of the waveform in Figure 6-4 hasn't propagated as far as we would have expected. For all intents and purposes, the signal appears to be traversing the material at a reduced speed!

This is the essence of the Ewald–Oseen extinction theorem. "Extinction" refers to the original wave still theoretically being there but effectively getting replaced with a composite signal that appears to be moving at a speed lower than c. The slowdown is sustained through continued interactions with dipoles, so it goes on only while the wave is moving through the dielectric. The moment the signal exits the medium, it can pick up the pace but can't make up for the time already lost.

Again, the original model deals with optics, plane waves, and homogeneous dielectrics. I don't know if it seamlessly generalizes to coaxial cables, although at reasonably low frequencies and with normal (transverse) waves, I imagine it could. Either way, it's an interesting and reasonably intuitive explanation of why dielectric materials can profoundly change signal propagation speed.

How Signal Velocity Relates to Inductance and Capacitance

The dipole-based explanation for signal velocity (v_s) also offers an intuitive basis for the classical but mildly puzzling physics equation that ties signal propagation velocity to a pair of parameters known as magnetic permeability (μ) and electric permittivity (ϵ):

$$v_s = \frac{1}{\sqrt{\mu \cdot \epsilon}}$$

Without getting too deep into the woods, these physical qualities are affected by the presence of electric and magnetic dipoles and thus are directly tied to the electronic concepts of capacitance (C) and inductance (L) expressed per unit length (d). In fact, for an idealized (long) conductor, we can write the same equation the following way:

$$v_s = \frac{1}{\sqrt{(L/d) \cdot (C/d)}}$$

We typically talk of inductance and capacitance as properties of specific electronic components, but every sufficiently long wire can be described as having a series inductance that opposes the change in current, along with a conductor-to-conductor capacitance that requires a nonzero current to flow before the voltage goes up. Without these effects, signal propagation would always be effortless and instantaneous. That would be incompatible not only with common sense but also with the theory of relativity.

An interesting corollary of this last remark is that although the actual values of L and C are most commonly associated with solid matter, a vacuum necessarily has a nonzero inductance and capacitance too. The numbers, when plugged into the aforementioned formula, work out to a signal propagation speed of $v_s = c$.

For now, we can set mind-bending tie-ins between electronics and cosmic constants aside. Instead, let's relate this formula to coaxial cable. If you look at the datasheet, good coax might have an inductance of around 210 nH/m and a capacitance of 80 pF/m. After a bit of unit-wrangling, the equation yields a value similar to the earlier experimental result:

$$
\begin{aligned}
v_s &= \frac{1}{\sqrt{210 \cdot 10^{-9}\,\frac{\mathrm{H}}{\mathrm{m}} \cdot 80 \cdot 10^{-12}\,\frac{\mathrm{F}}{\mathrm{m}}}} \\
&= \frac{1}{\sqrt{16{,}800 \cdot 10^{-21}\,\frac{\mathrm{H \cdot F}}{\mathrm{m}^2}}} \\
&= \frac{1}{\sqrt{16{,}800 \cdot 10^{-21}\,\frac{\mathrm{s}^2}{\mathrm{m}^2}}} \\
&\approx \frac{1}{4.1 \cdot 10^{-9}\,\frac{\mathrm{s}}{\mathrm{m}}} \approx 244{,}000\,\mathrm{km/s}
\end{aligned}
$$

From the second line to the third, we exploit the fact that the unit of capacitance multiplied by inductance ($H \cdot F$) works out to s^2. To figure out why this is, flip back to the definitions in Chapter 5. You can avoid these kinds of head-scratchers by using a unit-aware calculator app. I recommend Qalculate: *https://qalculate.github.io*.

Signal Reflections

So far, we've established beyond any doubt that electronic signals propagate at close to the speed of light. This is interesting, but it could've been stated far more succinctly. The reason we went through all these hoops is to set the stage for an exploration of the interesting phenomenon of signal reflections. To take the plunge, all we need to do is replace the 50 Ω resistor originally placed at the far end of the line when we were measuring the speed of electricity in a length of coax.

On the face of it, the resistor is just a test load, a way to ensure that there's some healthy current flowing through the rig. It's also chosen to roughly match the output capability of the signal generator. These devices can maintain the programmed output voltage only if the current is kept in check, and they're typically rated for driving no more than 50 Ω.

Let's probe what would happen if we switch the termination to a lower resistance, for example, 10 Ω. Again, based on the earlier remark about signal generators, the device won't be able to keep up. The current that used to be sufficient to produce an amplitude of 5 V across the 50 Ω resistor will no longer get us anywhere near. But while you might expect to see an ordinary square wave with lower peaks, the reality is more interesting (Figure 6-5)!

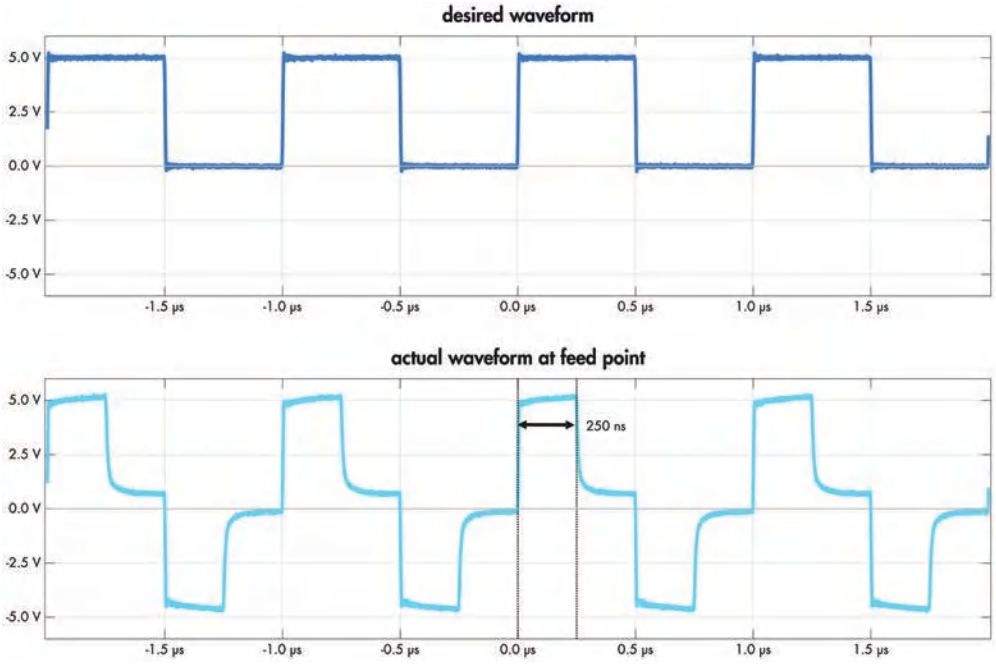


Figure 6-5: Signal distortion when driving a low-impedance load over long coax

The top trace is a reference signal. It's coming out of the second channel of the generator, synchronized to the primary waveform but not connected to any load. The bottom trace is the voltage at the feed point of the coax line. Surprisingly, the signal reaches +5 V for the first 250 nanoseconds of every cycle. After that, it plunges precipitously, at one point dropping down to nearly −5 V.

The explanation is that the generator has no way to just know what it's connected to on the other end of a long cable. It can only "look" down the conductor by sending out a signal and waiting to see what happens. It would outright violate causality if the device could tell what the resistor connected on the other end is going to do before the electromagnetic wave makes it there and back. For the duration of the round trip—about 125 ns there and 125 ns back—pushing electrons into the coax cable necessarily takes the same amount of effort whether it's an open circuit or a short. Only after that does the reality of driving a low-impedance load suddenly set in.

If this sounds unconvincing, we can repeat the experiment with shorter pulses to isolate the effect more clearly. The relation between the "on" time and the "off" time of a waveform is known as a *duty cycle*. Figure 6-6 shows the trace produced for a 1 MHz square wave with a duty cycle of 10 percent.

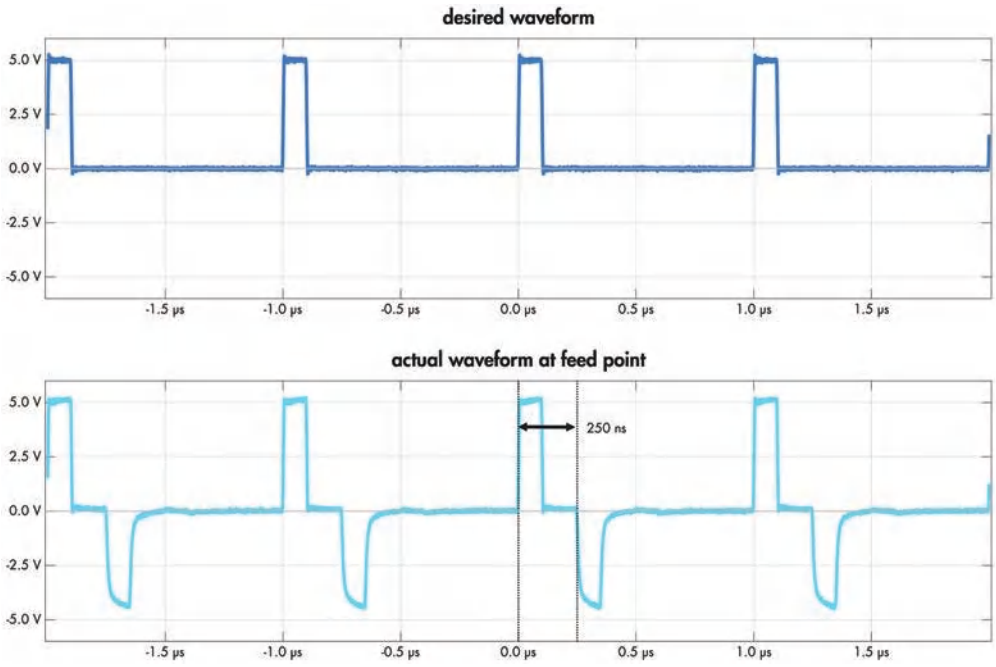


Figure 6-6: Another view of signal reflections for short pulses (10 percent duty cycle)

The echo we're looking at is called a *signal reflection*. The behavior is best understood as a relativistic consequence of prior actions, essentially a temporal decoupling of cause and effect. In this instance, the voltage becomes negative to make up for the unexpectedly high current that has flowed through the far-end resistor some time before.

For completeness, we can also repeat the experiment with a high-impedance load. This can be a 1 kΩ resistor or simply an open circuit at the end of the coax. The resulting traces are shown in Figure 6-7. This time, the voltage shoots up, essentially telling us "the current you sent down the wire had nowhere to go, so here's your energy back."

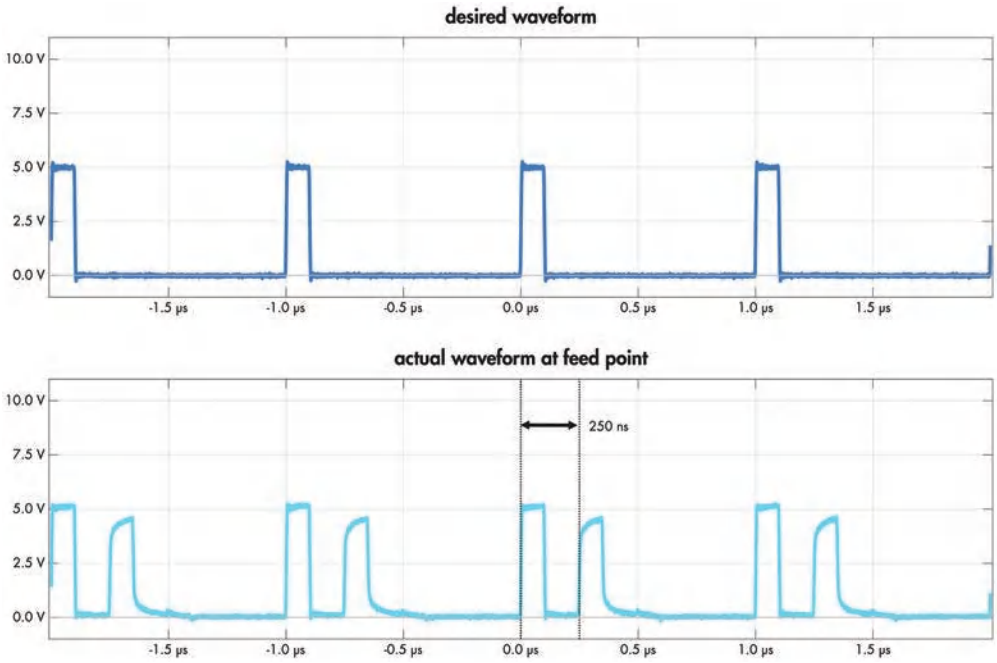


Figure 6-7: Driving a high-impedance load over 100 feet of coax

Reflections may seem perplexing. They come out of nowhere and seemingly cause chaos. Heck, in extreme cases, the unexpected voltage swings could even damage sensitive components. Even so, the phenomenon only matters if the conductor is long in relation to signal wavelength. In a nod to the era of the telegraph, we refer to such long wires as *transmission lines*. If the connection is comparatively short, then the voltage across the entire conductor equalizes much faster than the signal can change. In that case, the reflections are imperceptible and don't linger around for long.

In practical terms, on a typical circuit board, there's no special reason to ponder reflections when working with sine frequencies below 200 MHz. This frequency translates to a wavelength of around 1 meter at PCB signal velocities. As discussed in Chapter 4, digital square waves need to be analyzed as a sum of higher-frequency sines. In that case, some caution is advised above 50 MHz, although the amount of distortion should be fairly modest until you cross 100 MHz. Past that point, the issue can be kept in check by using consistent, matched impedances along the signal path. We'll briefly circle back to this topic in Chapter 30, but in the simplest case, sprinkling some resistors on the PCB could do.

In the discussion of the experimental setup used in this chapter, we've focused on the characteristics of the source (the signal generator) and the load (a 50 Ω resistor). In doing so, we sidestepped the behavior of the cable itself. We can tell the line wasn't acting like an electrical short. In the experiments, the generator was initially able to create the desired voltage across its terminals, and it kept it up until the electromagnetic wave traveled there and back. The coax also wasn't an open circuit. We evidently succeeded at transmitting energy toward the load, which came back and caused the voltage to shoot up in the final experiment. But then, if it was neither of these two things, how did the conductor actually behave? The parameter we're missing is the characteristic impedance.

Characteristic Impedance

The *characteristic impedance* (Z_0) of a conductor describes the ratio between the applied voltage delta and the momentary current that must flow to propagate that voltage to the other end of the wire. (At this point, you'd be right to complain that *impedance* is an overloaded term.) The value of Z_0 plays no role for steady currents or steady voltages. It accounts only for the brief moment after the voltage is adjusted on one end, but before the signal comes out on the other side.

The simplest way to derive the formula for characteristic impedance is to imagine a step voltage increment (V_{step}) that has already traveled x meters down the transmission line. We don't need to choose the value of V_{step}; we just assume that it's some known quantity.

Earlier, we introduced the formula for signal velocity (v_s) in a well-behaved conductor. By dividing distance by signal velocity, we can determine the time t that has elapsed for the wavefront to reach the x meter mark:

$$t = \frac{x}{v_s} = x \cdot \sqrt{L/d \cdot C/d}$$

As before, L/d and C/d correspond to the transmission line's inherent inductance and capacitance per some unit of length. The value of d doesn't need to have any relation to x; in fact, both terms will separately cancel out down the line.

From Chapter 5, we can dust off the formula that relates the voltage applied to a capacitor to the current that must have flowed in a unit of time:

$$I = C \cdot \frac{V}{t}$$

In our model, the propagating step voltage V_{step} is currently present across x meters of the conductor. The capacitance of this section can be trivially calculated from the conductor's inherent value of C/d, multiplied by x:

$$C_{step} = \frac{C}{d} \cdot x$$

Equipped with the capacitance of this length of the conductor, we can apply the earlier formula for charging current to figure out what it must've taken to charge said capacitance to V_{step}:

$$I_{step} = \frac{C_{step} \cdot V_{step}}{t} = \frac{C/d \cdot x \cdot V_{step}}{t} = \frac{C/d \cdot \cancel{x} \cdot V_{step}}{\cancel{x} \cdot \sqrt{L/d \cdot C/d}}$$

$$= V_{step} \cdot \sqrt{\frac{C/\cancel{d}}{L/\cancel{d}}} = V_{step} \cdot \sqrt{\frac{C}{L}}$$

Impedance of any kind is always just the ratio between voltage and current. The applied voltage is given (V_{step}) and we've just come up with a formula for the resulting current, I_{step}. In other words, we have all we need to calculate Z_0:

$$Z_0 = \frac{V_{step}}{I_{step}} = \frac{\cancel{V_{step}}}{\cancel{V_{step}} \cdot \sqrt{\frac{C}{L}}} = \sqrt{\frac{L}{C}}$$

Note that in this idealized scenario, the value of Z_0 doesn't depend on signal frequency, waveform type, distance, or time. In particular, well-made coax should have the same characteristic impedance whether you're buying 10 feet or 100 feet.

Ideally, the characteristic impedance of the transmission line should be matched to the transmitter. This was the case for the RG-58 or RG-8X cables used in the earlier experiment, both designed to have a Z_0 value of about 50 Ω. A mismatch between the source and the conductor is less dramatic than a mismatch at the destination. If the characteristic impedance of the cable is too low for the source, the transmitter won't be able to build up the desired voltage right away. Conversely, if the characteristic impedance of the line is too high, the initial current through the load will be lower than it could be.

The coax used in the earlier experiment had an impedance around 50 Ω, so it happened to match both the source and the load. On a printed circuit board, the situation can be a bit more hand-wavy. The usual Z_0 values hover at around 100 to 150 Ω, although this can vary depending on board material and trace geometry. If you ever need better answers, there are online tools that can help. For example, a good selection of online PCB impedance calculators is available from DigiKey: *https://www.digikey.com/en/resources/online-conversion-calculators*.

Quarter-Wavelength and Half-Wavelength Lines

As discussed earlier, when working with long conductors, we usually seek to minimize reflections by matching the capabilities of the signal source, the characteristics of the conductor, and the impedance of the load. If we neglect this step, there's a good chance of ending up with a mess of voltage swings that get in the way of what we're trying to do.

Even so, there's a special case where the effect can work to our benefit. Consider a constant-frequency sine wave that's feeding into a conductor exactly one-fourth the signal's wavelength. Further, let's assume that the line is terminated with a comparatively high impedance on the far end. In this situation, there's a positive reflection coming back from the far end, an image of the previous one-fourth of the sinusoid neatly "folded back" onto the transmission line. A reflection produced this way always adds constructively on the far end of the cable and adds destructively near the feed point.

Figure 6-8 shows how this effect plays out over three time steps, when the first, second, and third quarters of the original waveform are aligned with the length of the cable. Take note of the dotted line that represents the sum of the driving signal and the reflection. Due to destructive interference between the two signals, the resulting summed voltage is always close to zero volts near the feed point on the left. At the same time, we can observe unexpectedly high voltage swings on the far end (right), where the signals add constructively.

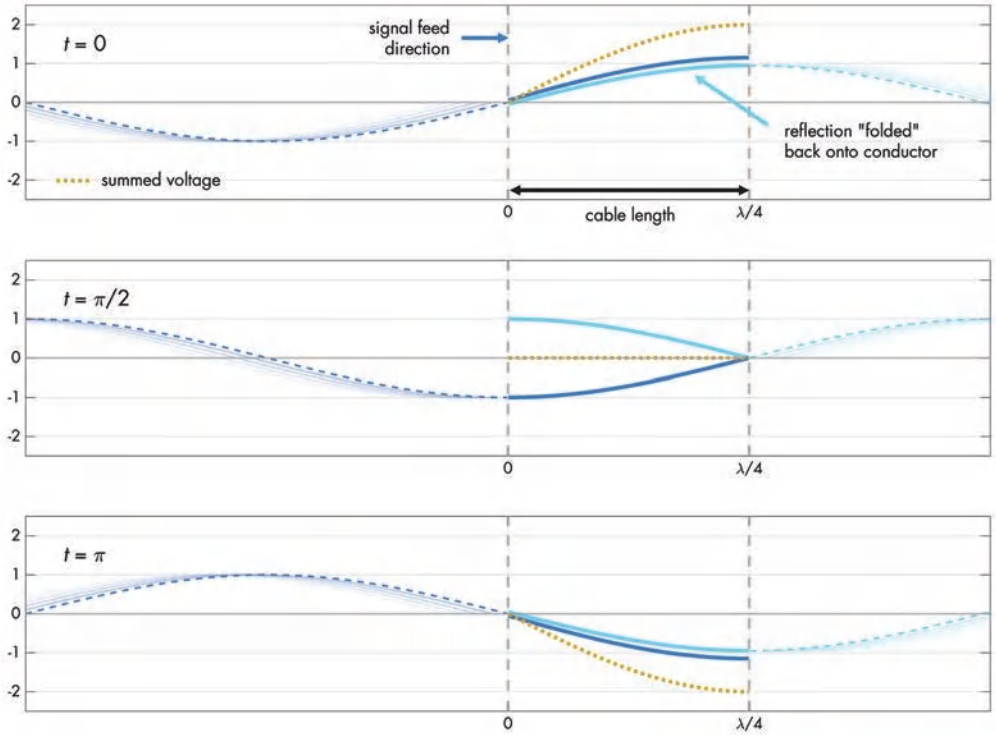


Figure 6-8: The interaction between a sine-wave signal (fed from the left) and its reflection (coming back from the right) in a quarter-wavelength line

The combined signal forms what's known as a *standing wave* that oscillates up and down on the far end of the conductor but doesn't appear to be moving left or right. Because of the consistently low voltage at the feed point, substantial currents can be pumped into the conductor with ease. These currents then produce large voltage swings on the other side. In effect, the conductor is a peculiar form of voltage amplifier.

A different situation can be observed when the cable is twice as long (one-half of the wavelength). This model also leads to the formation of a standing wave, but the low-voltage region is stuck in the middle of the conductor (dashed line in Figure 6-9), with alternating peak voltages at both ends. In contrast to the quarter-wavelength scenario, this conductor will be quite difficult to drive.

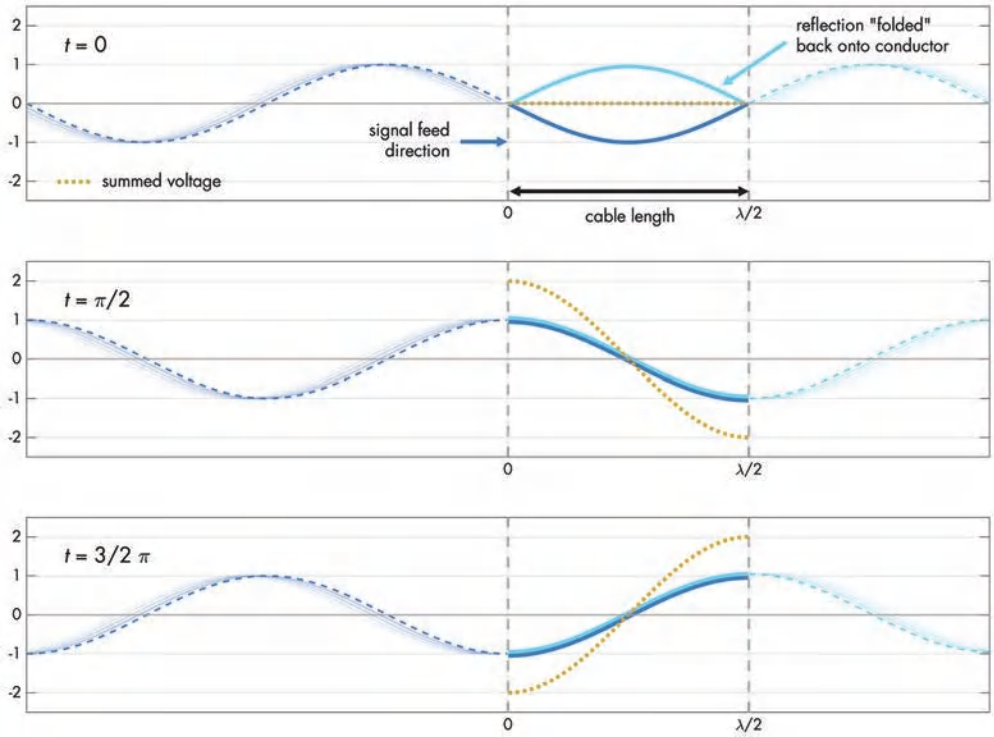


Figure 6-9: A sine-wave signal versus its reflection in a half-wavelength line

On the next couple of pages, we'll talk about the most important application of this concept: the process of sending and receiving transmissions through the void of space.

Radio Waves

Up to this point, our discussion of electronics has revolved around the behavior of conductors and conductor-adjacent materials. The reason for this is simple. A conveyance mechanism for charged particles makes it a lot easier to get work done. Electrostatic fields produced by immobile charges can propagate through wood or plastic just fine, but in the absence of a mechanism to guide or amplify the forces, the effect can be hard to measure, let alone to harness in any way. You can wave around a battery all you want, but it won't make an LED light up from afar.

Of course, there's a whole branch of electronics devoted to sending information and energy through space: radio-frequency (RF) engineering. The concept of radio transmissions is closely coupled to relativity, and at this point we're ready to take a closer look.

A Bad Antenna (and a Good One Too)

Conceptually, the simplest way to produce a coherent electrostatic field might be to take a charged capacitor and pull the plates apart. This act of electronic vandalism might require some serious muscle—the forces of electrostatic attraction are surprisingly powerful—but it should ultimately cause the capacitor's internal field to spill into the surrounding space in a manner loosely illustrated in Figure 6-10.

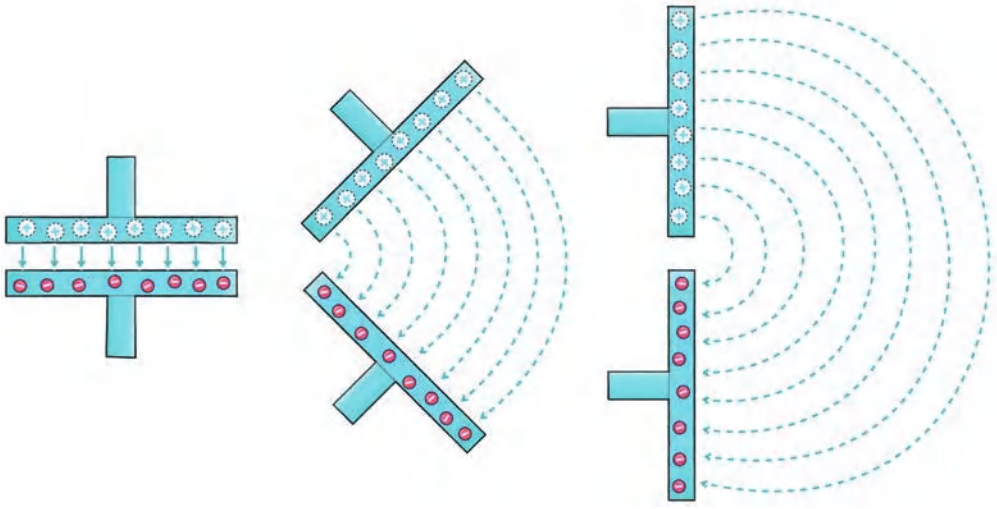


Figure 6-10: Constructing a terrible antenna

Electric fields are commonly visualized by plotting the paths of hypothetical positively charged particles subjected to the field. For our ex-capacitor, we'd be seeing arc-shaped lines from the positive plate to the negative one. Strictly speaking, the lines should be present on both sides of the plates and should extend as far as the eye can see.

An unchanging electric field isn't very useful. If it's just sitting there for all eternity, it doesn't convey information nor perform meaningful work. If we push a charge against the direction of the field and then let it loose, the particle will fly away. But that's just a matter of getting back the energy we expended on pushing it into position in the first place.

The situation changes if we start moving charges back and forth between the plates. This produces an interesting effect: a ripple-like pattern of alternating fields that propagate away from the ex-capacitor at the speed of light (Figure 6-11).

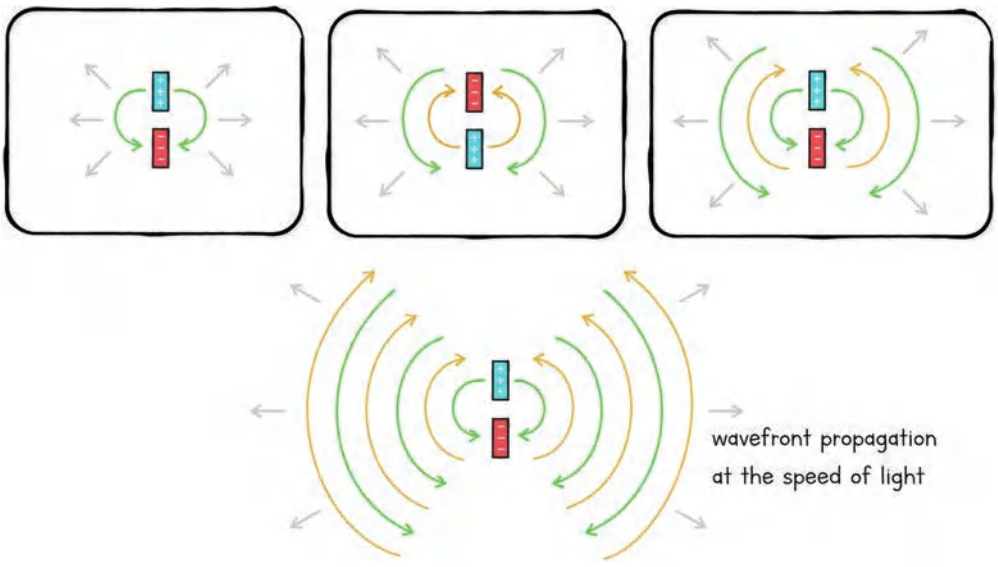


Figure 6-11: A conceptual sketch of an electric field pattern around a simple antenna

Each change in field direction (or intensity) sneaks away with some amount of energy. The radiated energy increases not only with the number of charges shuttled back and forth between the plates but also with the number of transitions per second—that is, with the signal's frequency. Due to relativity, we can no longer recover the radiated energy directly, not in the way we could get it back from a steady-state capacitor charged to a given voltage. Even so, it can be transferred to downstream charges that begin to move back and forth as they "swim" through the gradients of the field. This motion might induce currents in a receiving antenna placed some distance away.

Time for a reality check, though. If we go back to our dismantled capacitor and hook it up to a signal generator, it won't actually do much. When we pulled the plates apart, we greatly reduced the device's capacitance, so we're essentially looking at an open circuit. A very high voltage would be needed to shuffle a decent number of electrons back and forth. Without the motion of a sufficient number of charge carriers, the antenna just doesn't perform.

The most elegant solution to this problem is known as a *half-wavelength ("half-wave") dipole*: two rods mounted along a common axis, driven by a sinusoidal signal at the center, each rod exactly one-quarter wavelength long. From the earlier discussion of standing waves in a transmission line, we know why this works. The quarter-wavelength elements produce large voltage swings at the far ends and thus admit comparatively large alternating currents, without requiring a high-voltage signal source.

At least in theory, we now know how to transmit a steady waveform. But how do we take it up a notch and broadcast some catchy tunes?

Modulation Schemes

Radio signals are typically constructed by starting with a sinusoid at a specific, fixed *carrier* frequency and then subtly changing (*modulating*) that sinusoid to encode a comparatively slower data stream. A fairly typical example would be transmitting analog audio sampled at about 11 kHz by modulating a carrier frequency of 144.450 MHz (one of the amateur radio bands in the United States).

The use of a fixed-frequency carrier comes down in part to the physics at play in the antenna. A half-wave dipole antenna has a specific resonant frequency, dictated by its length, and will be less efficient if the signal diverges from this baseline. Further, for resonance to occur, the reflection of the previous quarter-cycle must be a faithful complement of the current cycle, hence the preference for sinusoids. With a different waveform, the interference pattern within the antenna could get out of whack.

Figure 6-12 shows two common modulation schemes: *amplitude modulation (AM)* and *frequency modulation (FM)*. The effects of both are exaggerated for readability. In real life, the carrier signal's frequency is many orders of magnitude higher than that of the encoded transmission, so the changes from cycle to cycle are nearly nil.

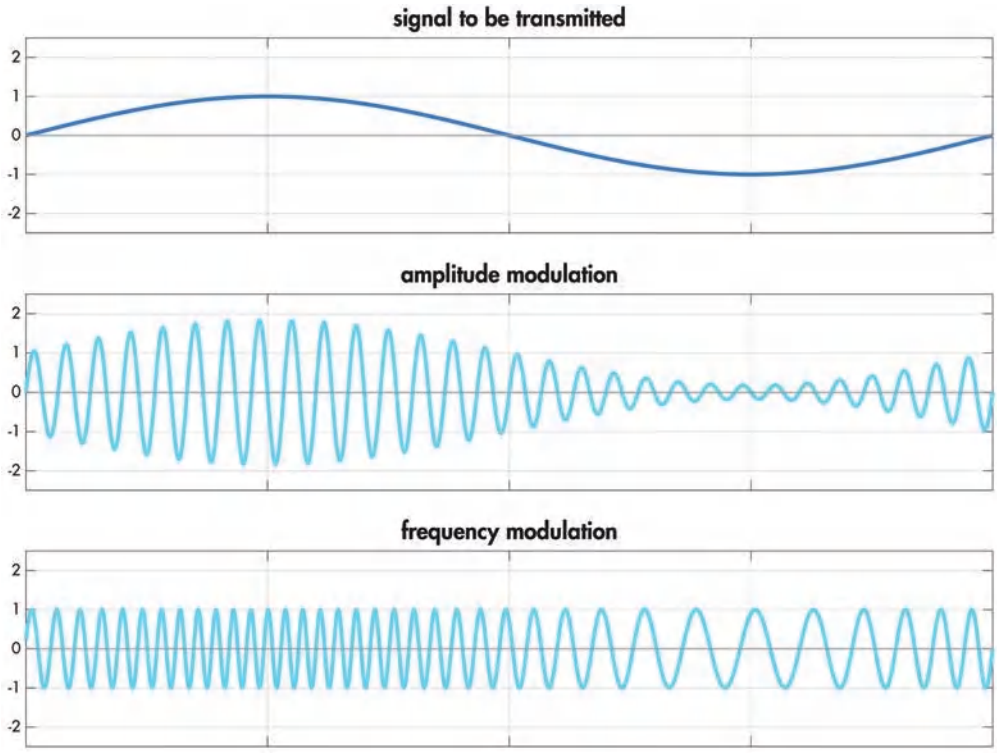


Figure 6-12: AM and FM signals

With amplitude modulation, the voltage swing of the carrier is modified in proportion to the level of the encoded signal. With frequency modulation, the carrier's frequency is tweaked instead.

It's fairly obvious that FM transmissions take up *bandwidth*—that is, they occupy a slice of the spectrum at least between the lowest and the highest modulated frequency. In contrast, AM might feel like a clever, zero-bandwidth hack. After all, we're just changing the amplitude of a fixed-frequency sine wave, so what's to stop us from sending out any number of AM transmissions a fraction of a hertz apart?

Alas, in practice, all modulation is frequency modulation. The difference is the construction method, not the end result. Receivers operate by deconstructing incoming signals into constituent sine waves. In this view, any deviations from a steady sine wave—even changes in amplitude—inevitably produce halos at other frequencies. The scale of the artifacts is proportional to the rate of change in the encoded signal. AM isn't special and takes up bandwidth too.

A full mathematical analysis of the problem would take a while, but we can demonstrate how AM "pollutes" the nearby spectrum with a simple thought experiment. Let's consider a 40 Hz carrier modulated with a sine wave running at 2 Hz. This situation, shown in Figure 6-13, is quite similar to what we discussed moments ago.

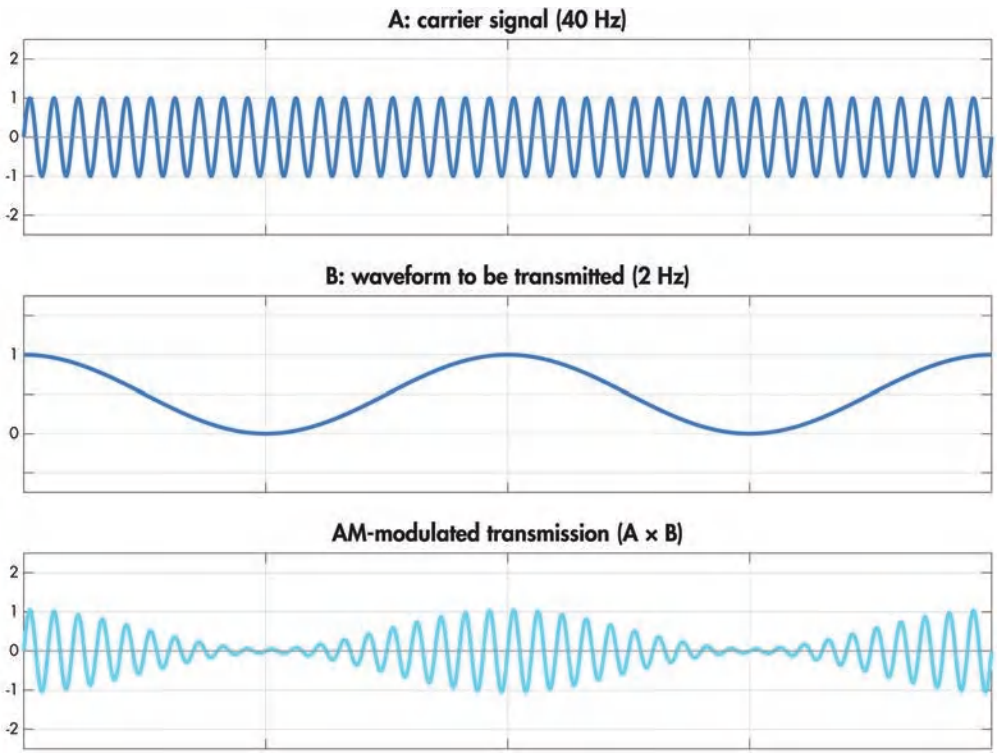


Figure 6-13: Modulating a 40 Hz carrier with a 2 Hz sine wave

Next, let's have a look at a seemingly unrelated case: the sum of two separate sine waves running at 39 Hz and 41 Hz. The first two panels of Figure 6-14 show these constituent waveforms; the bottom panel shows the superposed transmission, similar to what would be seen on the receiving end.

The combined signal shows an undulating pattern because the constituent waveforms are running at slightly different frequencies, precisely 2 Hz apart. This causes the peaks to repeatedly snap in and out of phase alignment at that exact frequency. Half of the time, the values are adding constructively. The other half of the time, the peaks have the opposite signs.

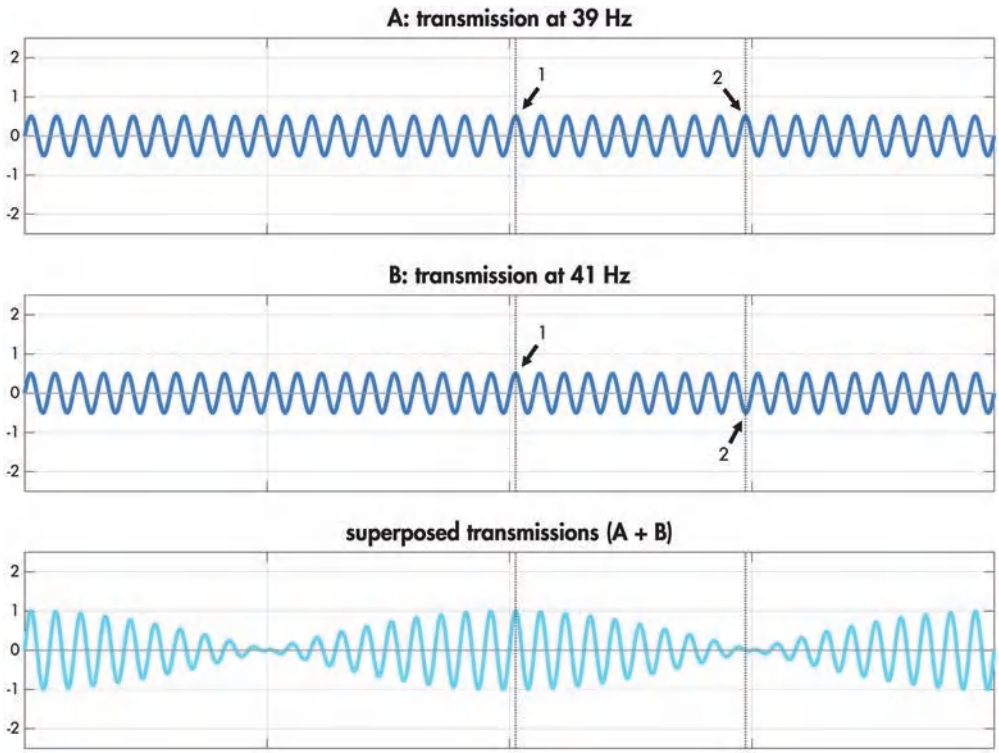


Figure 6-14: A superposition of 39 Hz and 41 Hz waveforms. At the locations marked as "1," both waveforms are in phase and are adding constructively. The point marked as "2" corresponds to a 180° difference in phase and thus destructive interference.

If you compare the bottom waveform in Figure 6-13 (AM modulation) to the bottom of Figure 6-14 (superposed transmissions), it should be clear that the results are nearly the same. This means that on the receiving end, there's no good way to distinguish a sinusoidal AM signal from unrelated sine-wave transmissions happening in the nearby frequency bands.

With this out of the way, let's have a look at how receivers actually work.

Beat Frequencies

The basic operation of almost every conventional radio receiver boils down to mixing (multiplying) the incoming signal with a steady sine wave of a chosen frequency. The mixing of two different sine waves produces a low-frequency "beat" that's equal to the absolute difference between the mixed frequencies, $|f_1 - f_2|$, as well as a second (and less important) component at $f_1 + f_2$. A visual demonstration of the effect is shown in Figure 6-15.

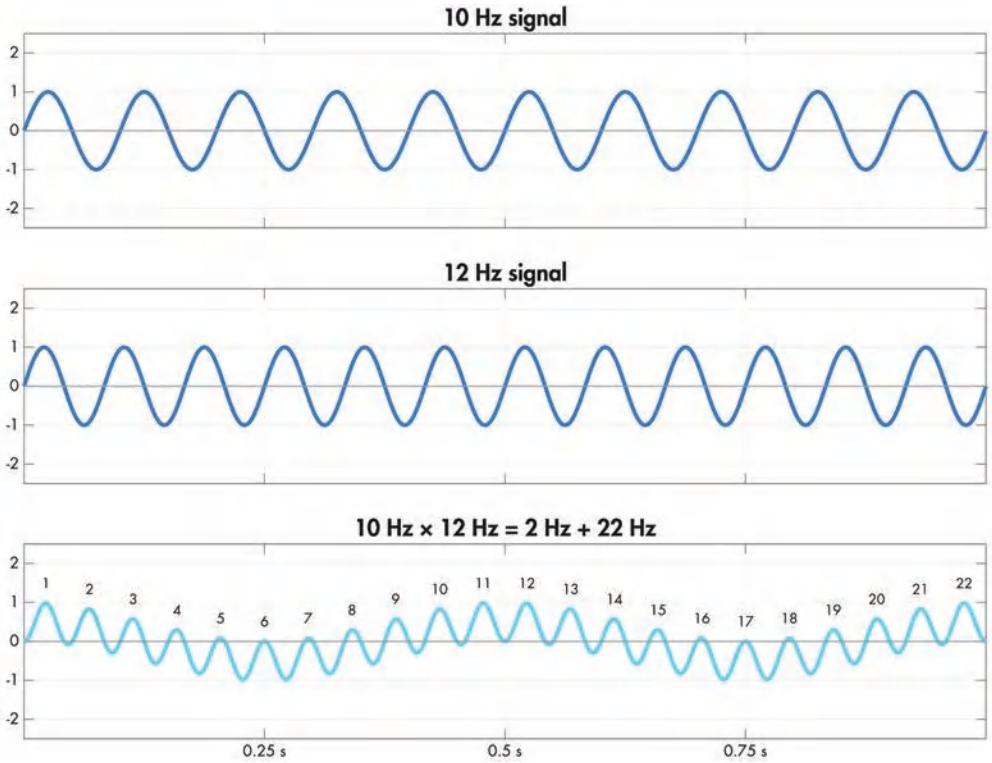


Figure 6-15: The beat frequency evident after mixing a 10 Hz and a 12 Hz sine-wave signal

The illustration shows the result of mixing a 10 Hz sine wave with a 12 Hz signal. The result is a fast-changing sinusoid with 10 + 12 = 22 peaks per second (22 Hz), seemingly overlaid on top of a much slower, rolling waveform of 12 − 10 = 2 Hz.

Even with the benefit of the earlier discussion of the pattern produced by summing two sine-wave transmissions, this behavior might seem perplexing. Still, it's just a matter of trigonometry, and to show that radio receivers aren't just voodoo, we ought to construct a cute proof. For that, we need to lean on the *cosine angle sum identity* (Figure 6-16). This identity asks how we can express the cosine of the sum of two angles in terms of those two angles' separate sines and cosines. At first blush, it might seem like a completely orthogonal geometry problem, but it all comes together in the end.

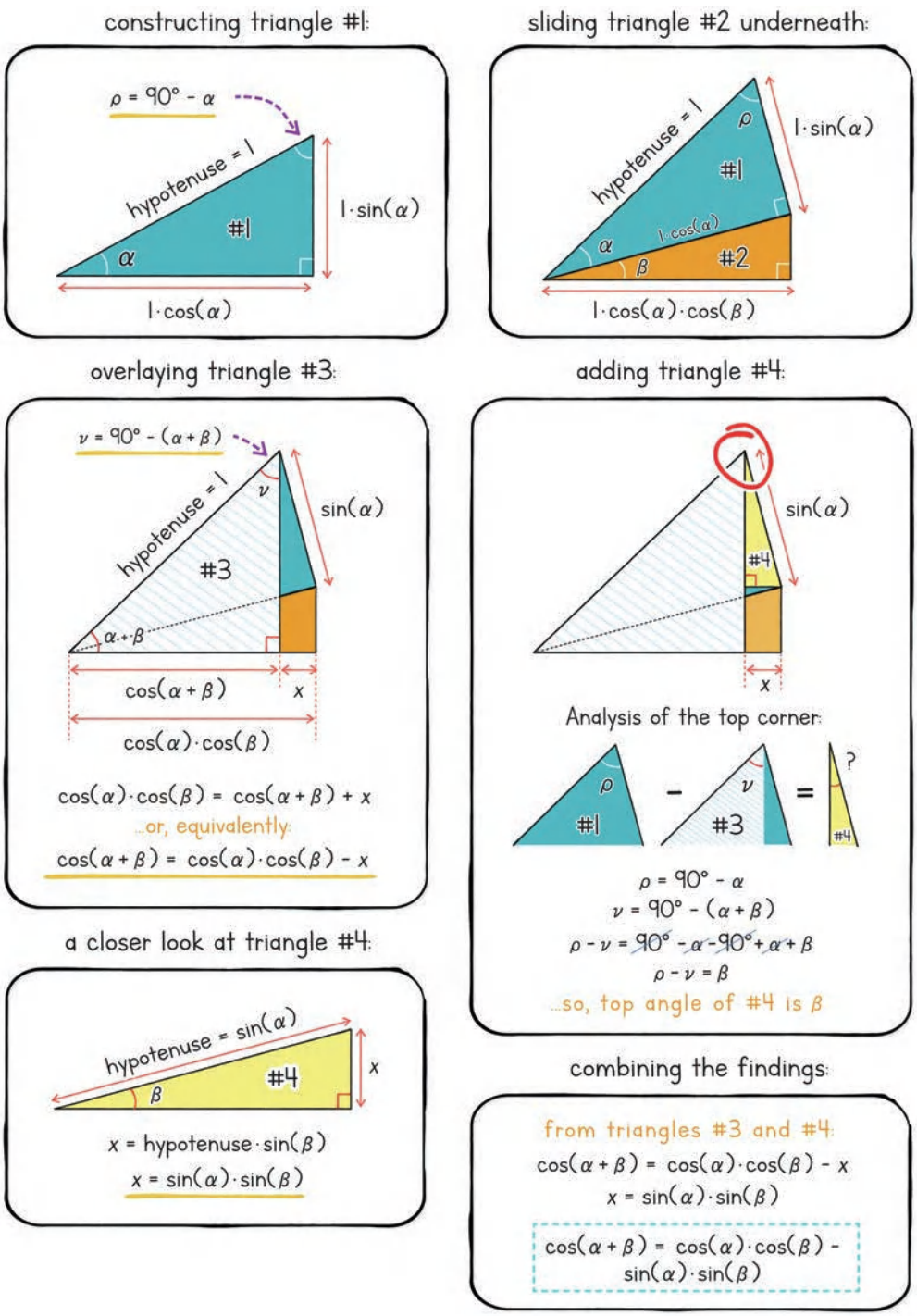


Figure 6-16: A visual proof for the cosine angle sum identity

The derivation of the identity starts with two given angles, α and β. We first construct a right-angle triangle (#1) that has an angle of α and a hypotenuse equal to 1. Next, we calculate the length of the adjacent of that triangle using basic trigonometry: $1 \cdot \cos(\alpha)$. After that, we slide a second triangle underneath. The new triangle

(#2, second panel) has an angle of β and a hypotenuse of the same length as the adjacent of triangle #1, $\cos(\alpha)$. This makes the adjacent of #2 equal to $\cos(\alpha) \cdot \cos(\beta)$.

Next, we add triangle #3, placing it on top of the other triangles (third panel). The new triangle inherits a hypotenuse of 1 from triangle #1, and it has a combined angle of $\alpha + \beta$ as a consequence of being partly inscribed inside #1 and #2. It follows that its adjacent is $1 \cdot \cos(\alpha + \beta)$.

The bottom edge of the combined figure has a length equal to the known adjacent of triangle #2: $\cos(\alpha) \cdot \cos(\beta)$. Alternatively, the length can be expressed as the adjacent of triangle #3, $\cos(\alpha + \beta)$, plus some unknown segment, x. We solve for that segment by looking at a small helper triangle (see the last three panels). A quick analysis shows that x is equal to $\sin(\alpha) \cdot \sin(\beta)$. Putting it all together, we obtain the following angle sum identity:

$$\cos(\alpha + \beta) = \cos(\alpha) \cdot \cos(\beta) - \sin(\alpha) \cdot \sin(\beta)$$

Again, this might not seem to be particularly relevant to the operation of radio receivers, but now that we have an expansion for $\cos(\alpha + \beta)$, we can trivially extend it to $\cos(\alpha - \beta)$:

$$\cos(\alpha - \beta) = \cos(\alpha + (-\beta)) = \cos(\alpha) \cdot \cos(-\beta) - \sin(\alpha) \cdot \sin(-\beta)$$

Applying simple trigonometric substitutions, $\cos(-\beta) = \cos(\beta)$ and $\sin(\beta) = -\sin(\beta)$, produces the following:

$$\cos(\alpha - \beta) = \cos(\alpha) \cdot \cos(\beta) + \sin(\alpha) \cdot \sin(\beta)$$

Together, the equations for $\cos(\alpha + \beta)$ and $(\alpha - \beta)$ let us show that:

$$\begin{aligned}\cos(\alpha - \beta) + \cos(\alpha + \beta) &= \cos(\alpha) \cdot \cos(\beta) + \cancel{\sin(\alpha) \cdot \sin(\beta)} \\ &\quad + \cos(\alpha) \cdot \cos(\beta) - \cancel{\sin(\alpha) \cdot \sin(\beta)} \\ &= 2 \cdot \cos(\alpha) \cdot \cos(\beta)\end{aligned}$$

This is our moment of trigonometric Zen. If we divide both sides by 2 and flip the expression around, we end up with a formula that equates the product of two sine frequencies to an average of two independent cosines running at $|f_1 - f_2|$ and $f_1 + f_2$:

$$\cos(\alpha) \cdot \cos(\beta) = \frac{\cos(\alpha - \beta) + \cos(\alpha + \beta)}{2}$$

In other words, we have proof that mixing two steady sine waves always produces these two components and nothing else.

This is also a serendipitous way to formalize the earlier observation about AM modulation. The modulation process takes a carrier frequency α and multiplies it by a slower-running sine at a frequency β. The result is the same as the sum of two symmetrical signals offset from α by $+/- \beta$, so in the realm of sine waves, AM signals take up bandwidth just the same as any other modulation scheme.

But enough with triangles: We were supposed to talk about radio gear.

Superheterodyne Receivers

Given what we know about beat frequencies, one straightforward way to receive a transmission centered around 10 MHz would be to mix the received signal with an internally generated 10 MHz sine wave. According to our formulas, this should downconvert 10.01 MHz to a 10 kHz beat, turn 10.02 MHz into 20 kHz, and so forth. With

the mixing done, the next step would be to apply some sort of filter to the output, only keeping the low frequencies that are part of the modulation scheme and getting rid of everything else, including unrelated transmissions at other frequencies and the unwanted $f_1 + f_2$ images. (We'll talk about the design of analog filters in Chapter 12, but for now it suffices to say that isolating sine-wave frequencies is a relatively simple task.)

The folly of this method is the modulo operator hidden in the $|f_1 - f_2|$ formula, a consequence of $\cos(x)$ being the same as $\cos(-x)$. This means that beat frequencies exhibit symmetry on the output side. In the aforementioned 10 MHz scenario, the input component at 9.99 MHz produces an image at 10 kHz too—precisely where 10.01 MHz was supposed to go.

To avoid this mirroring behavior, practical receivers mix the RF input with a frequency lower than the signal of interest, shifting the transmission to a still relatively high but constant intermediate frequency (IF). Figure 6-17 illustrates the principle.

After the mixing, the receivers use slightly more complex filtering to pluck out the relevant frequency band. In this design—devised by Edwin Armstrong around 1919 and dubbed *superheterodyne*—the fundamental mirroring behavior is still present, but the point of symmetry can be placed far away. With this trick up our sleeve, accidental mirror images of unrelated transmissions become easier to manage, especially if combined with tuned antennas or radio frequency pre-filters in front of the mixer.

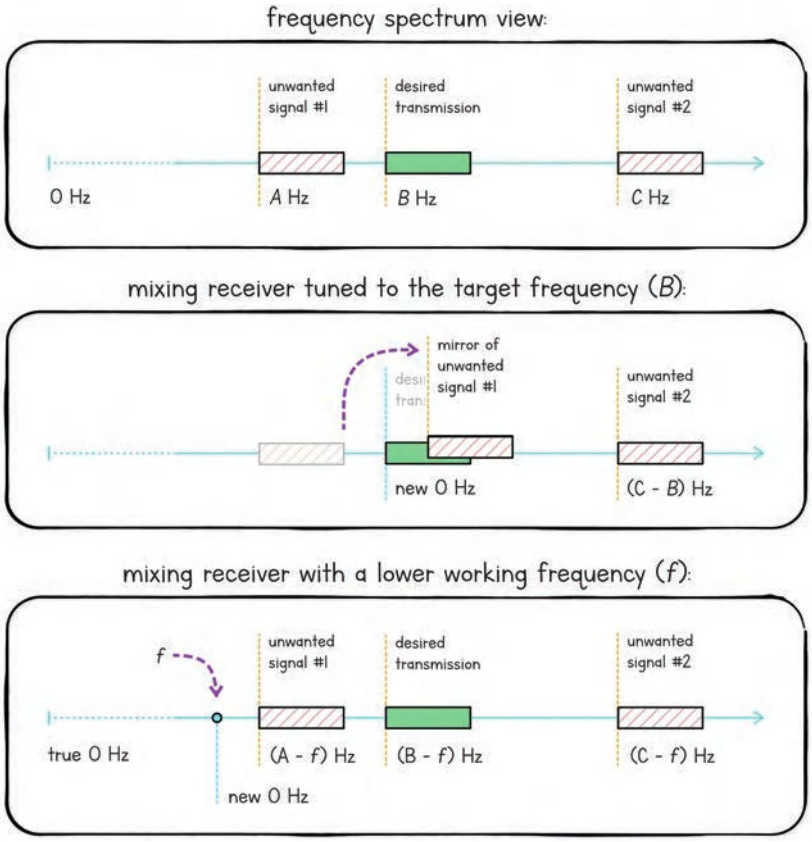


Figure 6-17: Avoiding mirrored signals by using lower frequency

This concludes our quick romp through radio theory. Coming up next: Let's spin the dial and tune in to a classic song by the Insane Clown Posse.

Magnets—How Do They Work?

For decades, magnetic devices known as *transformers* featured prominently in audio and radio gear, many of them constructed by hand or engineered for a specific use. Because of this, a detailed discussion of magnetic fields found its way into almost every introduction to electronics. The explanations inevitably leaned on *classical magnetism*—a collection of equations and rules that work well but don't have an obvious connection to the physical realm. Electric and magnetic fields were notionally introduced as two sides of the same coin but then, somewhat confusingly, were treated as separate animals governed by their own rules. In the end, readers were left with a mathematical model of a mysterious force but no real explanation of where it comes from.

I'm not sure the continued emphasis on classical magnetism in electronics still makes sense. For hobbyists outside a handful of niches, the knowledge of "H fields" and "B fields" matters less than it used to. Gone are the days of beefy hand-wound transformers and bespoke ferrite antennas that cropped up in every other circuit of note. Simple, small inductors are now ubiquitous, but while their behavior comes down to storing energy in a magnetic field, their performance is more usefully modeled using pure electronic concepts outlined in manufacturers' specs. We'll cover these parameters in Chapter 8.

Because of this, I'm inclined to skip the rote repetition of all that abstract theory. Instead, we're going to take the less common but more interesting route of trying to derive magnetism from first principles, even if just to satisfy curiosity. We'll wrap up with a couple of useful rules of thumb that don't require you to memorize any formulas, and then we'll circle back to real-world inductors and transformers later on.

All Aboard the Relativity Train

To properly understand magnetism, we need to come back to the speed of light. As noted early in the chapter, in the most basic sense, c is a fundamental constraint on causality. In other words, nothing can exert influence faster than the time it takes light to travel that distance.

This creates an interesting problem for Newtonian physics. Let's imagine that a guy—we'll call him Finn—is sitting near the front of a railroad car that's 20 meters long and passing by at 50 percent of the speed of light. Just as the rear of Finn's car passes a streetlamp, the lamp suddenly turns on (Figure 6-18).

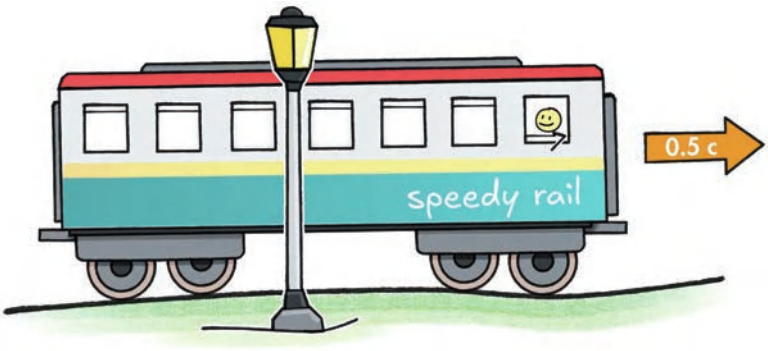


Figure 6-18: Speedy Rail always arrives on time.

Classically, the lamp's frame of reference is as good as Finn's. In fact, Finn might argue that he's stationary and that the lamp is moving away. There should be nothing unusual about physics on board his car. From Finn's perspective, it stands to reason that once the lamp turns on, the light should be reaching him at roughly $t + 67$ nanoseconds—that's how long it takes for photons traveling at the speed of light to clear the length of the car.

But for us, standing in a nearby meadow as the train is passing by, this ain't right. After 67 nanoseconds, the photons have traveled 20 meters, but the front of the car has moved 10 meters forward as well. Unless Finn's photons are a lot faster than ours, more time needs to pass before he sees the light.

An early attempt to address this issue was the concept of *luminiferous aether*, a cosmic medium through which photons supposedly propagate at a constant speed. This theory implied the existence of a special, aether-anchored frame of reference. If your own frame happened to be anchored to the aether, then Finn, tearing through the aether at breakneck speeds, would see the physics on his railroad car start to behave oddly.

Alas, no proof for the existence of luminiferous aether was ever found. Instead, the true explanation appears to be special relativity. The theory posits that, yes, all inertial frames of reference are equivalent. The trick is that when the frames are in relative motion, they experience space and time in different ways. In particular, from our frame of reference in the meadow, instantaneous measurements of Finn's car would indicate it's physically shorter than expected, and the time on board would appear to pass more slowly than it does from where we're standing.

From Electrostatic Fields to Magnetism

For a basic physical explanation of magnetism, we only need to lean on relativistic length contraction—the apparent reduction in the size and distance between moving objects along the direction of their travel. The effect is tiny unless the objects are moving at very high ("relativistic") speeds, but electrons are a special case. The drift speed of electrons in a wire usually doesn't exceed several centimeters per minute, but their numbers are great and, as noted earlier, the forces involved are exceptionally strong. The force of repulsion between two 1 C charges placed 1 meter apart is equivalent to roughly 900,000 metric tons.

To understand where relativity comes into play, let's consider a section of copper wire at rest. In the metal, there's a certain number of mobile electrons in the conduction band, skating across a lattice that contains a matching density of immobile, positively charged copper ions. Internal electric fields require the electrons to stay in the conductor, but as discussed in Chapter 1, they're not kept on a particularly tight leash.

Next, let's imagine a couple of charged particles some distance from a copper wire; one of the particles is sitting still, while another is slowly drifting through space. Because the positive and negative charges in the conductor are in balance, there's no net electric field acting on either of these errant particles (Figure 6-19).

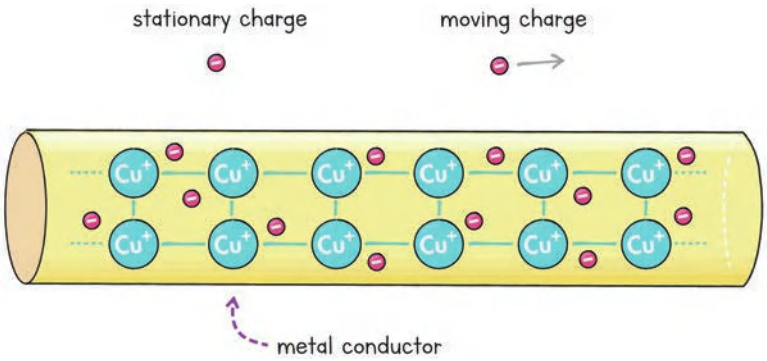


Figure 6-19: Interactions between a conductor and nearby charged particles

To take this thought experiment further, let's fashion the conductor into a circuit, a 1 meter loop of wire with 100 mobile electrons inside. It's a simple mathematical necessity that in the stationary ("lab") frame of reference, the average distance between these mobile electrons remains constant—one-hundredth of a meter (or 1 centimeter)—no matter if they're staying put or circulating around the loop. The only way to change their spacing would be to add electrons or take some away. Otherwise, it's always 100 elementary particles spread throughout a meter of wire.

At the same time, we asserted that electrons in motion are supposed to exhibit a tiny bit of length contraction. Once they start moving in relation to the observer, their apparent spacing in the direction of travel ought to be less than the true distance between the particles in their own frame of reference. The only way to square this is to realize that electrons in motion must be spreading out within their frame of reference so that our (length-contracted) perception of their spacing remains the same as before. After all, relativity doesn't require that the electrons' perception of the length or shape of the conductive loop stays the same as ours.

With this in mind, let's revisit the circuit, this time with some current flowing through. As we established earlier, in the lab frame of reference, the conductor isn't getting any shorter or any longer, so the average density of electrons and copper ions stays in balance. Because of this, there's no net electric field acting on a nearby stationary charge (Figure 6-20).

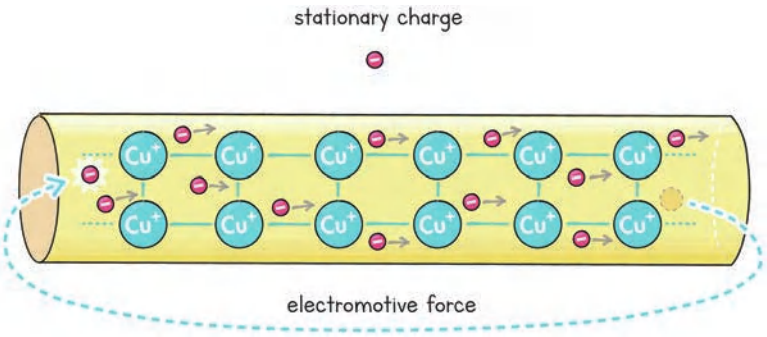


Figure 6-20: Electric current in the lab frame of reference

The situation isn't the same for a stray charge that's traveling in the same direction as the electrons. From its perspective, the conductor consists of a bunch of non-moving electrons that are spaced farther apart than the length-contracted lattice of positive ions moving the other way around (Figure 6-21).

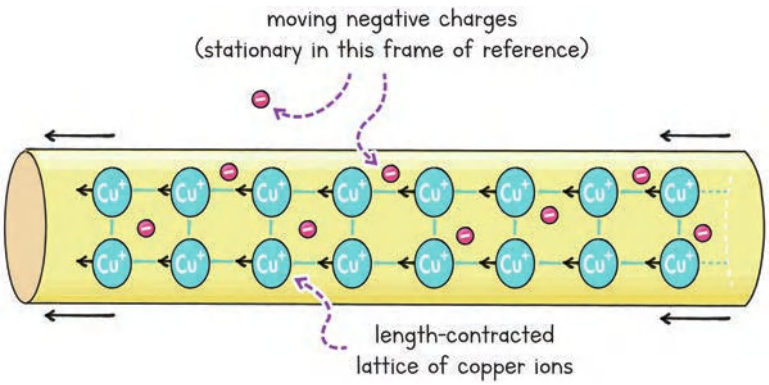


Figure 6-21: Current in a wire from the perspective of a nearby, moving negative charge

Because of the apparent higher density of positive charges, the moving charge, in its frame of reference, would experience a net electric field that's pulling it toward the current-carrying conductor. But again, this only works if the charge is moving along the wire in the first place; a stationary charge in the same position wouldn't be experiencing any pull.

This is, in essence, the origin of magnetic fields. More precisely, it's a way to show that they arise as a natural consequence of electrostatic interactions. We're not giving either of the phenomena a claim to primacy. We could just as well start with magnetic interactions in moving frames of reference and, from that, derive electrostatic fields.

Again, the reason I like this explanation is that it satisfies curiosity better than the classical approach. At the same time, it's not always useful to analyze the effect this way. The model gets less intuitive for accelerating charge carriers and interactions with permanent magnets or ferromagnetic metals. In the first case, common-sense solutions exist, but they take more effort to derive; in the second, the conceptual problem is that the relevant quantum parameter—electron spin—doesn't have a clear-cut analog in the macroscopic world. One could claim that electrons are rotating and that this motion is the basis of magnetic attraction, but the explanation is deficient in numerous ways. In the end, the relativistic basis of magnetism is worth knowing, but some rules of thumb from classical magnetism are still good to have at hand.

Back to the Classics

Some aspects of magnetism are harder to model within the context of special relativity, so a bit of memorization is hard to avoid. The first principle to keep in mind is that currents changing over time—in other words, charges that are accelerating or decelerating—induce sympathetic voltages in nearby parallel conductors. DC doesn't cut it; only AC produces this effect.

The magnetic coupling between parallel conductors is typically weak unless the current flowing through one of the wires is very high. However, the effect is greatly amplified if the wires are wrapped multiple times around a common ferromagnetic core. If only one winding (one run of wire) is used, we get a conventional inductor where the winding develops a pushback voltage proportional to the rate of change in current flowing through it (an effect explored in Chapter 5). If two or more separate windings are present, the induced voltages can manifest in a different circuit that straddles the same core. This makes a transformer, and we'll revisit transformers and their uses in Chapter 8.

The other concept worth knowing is the spatial relationship between electric fields and the customary way of visualizing magnetism. Magnetic fields are portrayed

using arrows that essentially track the direction of a compass needle traveling through the field. Figure 6-22 shows field lines inside a coil through which conventional current is flowing in a counterclockwise fashion. In effect, the magnetic field has a north pole on the top and a south pole on the bottom, and it interacts with permanent magnets accordingly.

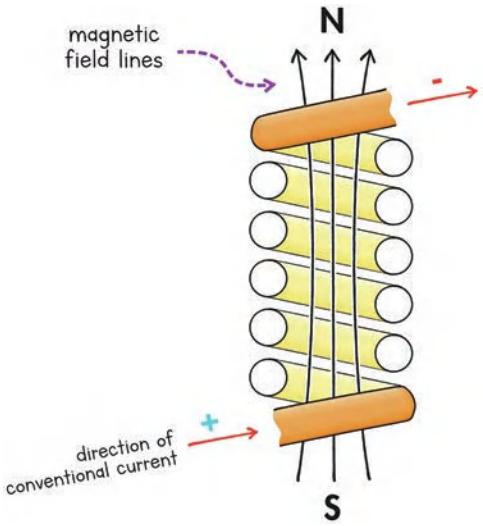


Figure 6-22: A cross-section of a coil showing internal magnetic field lines

A useful mnemonic for this is a variation of the so-called *right-hand rule*. Imagine giving a thumbs-up with your right hand; the curvature of your fingers shows the direction of conventional current, while your thumb points north.

This wraps up the discussion of the last major bit of physics that's needed to understand electronic circuits. Starting in the next chapter, we'll dive into the properties of common components—and before long, we'll be connecting them together to bend machines to our will.

PART II

CIRCUIT BUILDING BLOCKS

7

VOLTAGE AND CURRENT SOURCES



Now that we're done with the bulk of the high-level theory, let's switch gears and take a look at the building blocks of modern electronic devices. In this chapter, we'll start the journey at the source . . . the power source, that is.

In Chapter 3, we jokingly introduced an electron-pushing gremlin that keeps electric currents moving in a circuit. In practice, gremlins are fickle, so we typically leave the job to some sort of conventional power supply.

When you look closely, many power supplies are electronic devices in their own right. There can be a dozen or more components in a typical "wall wart" AC adapter, and even a rechargeable lithium-polymer battery may contain a microchip (a *battery management system*) to prevent the cell from being overcharged or discharged past a certain point. We'll talk about some of these circuits in Part III of the book. Most of the time, however, we abstract away this complexity and treat a power supply as a single component described by a couple of parameters. This is akin to how we don't need to map out every transistor in an integrated chip; we only care about what the chip does.

We can begin with several idealized models of power supplies. These models don't truly align with the real world, but they're an important stepping stone toward more precise approximations down the line.

Ideal Power Supplies

The most practical model of a power supply is a device that maintains a constant electromotive force (voltage) across its terminals. This allows the circuit to draw as much current as needed for a given task.

Such sources are important because many electronic components, from motors to integrated circuits, are designed for specific and well-controlled input voltages and may malfunction if the parameter fluctuates too much. Further, as noted in Chapter 4, electronic signaling is usually voltage-based; steady reference points are essential for transmitting information and interpreting the data correctly.

Alas, even in the realm of abstract electronic theory, a perfect supply of this type can't exist. Ohm's law states that $V = IR$, and if the terminals are shorted by an ideal conductor ($R = 0\ \Omega$), even an infinite current won't suffice. To avoid mathematical singularities, a more realistic model needs to have a current limit baked in. Simple wall warts may simply shut down if overloaded. DC lab power supplies used in electronic workshops are more sophisticated—they start in constant-voltage (CV) mode but switch to constant-current (CC) once a configured maximum current threshold is reached.

A variation of the DC voltage supply is an AC source that outputs an alternating signal of some frequency, amplitude, and shape. AC power sources are uncommon in portable electronics, but it's what comes out of the transformer that supplies electricity to your home. AC sources also serve as conceptual stand-ins for more complex input signals in computer simulations and workshop experiments. A benchtop signal generator can be thought of as a practical realization of a low-power AC voltage supply.

The final theoretical category is constant-current supplies. As the name implies, these devices try to maintain a constant flow of electricity through the load. They do so by adjusting the output voltage, increasing it if the load has high impedance and reducing it if it doesn't. Similar to constant-voltage sources, the concept suffers from a singularity: An ideal open circuit ($R = \infty\ \Omega$) would demand an infinite output voltage. This is why high-quality lab power supplies, if programmed to output constant current, will exit this mode once a preset maximum voltage is reached.

A degenerate form of constant-current source is a device that doesn't move any electrons on its own but moderates the flow of externally generated current when it's placed in series between a power source and a load. The device dissipates energy but doesn't generate it. It can be thought of as a variable resistor, and it's usually implemented with semiconductors. Calling this current limiter a "source" is debatable, but the terminology is well established in electronics, so it's best to be prepared.

Batteries

Moving away from theoretical models, batteries are among the most common ways to power small electronic devices. Each battery is made up of one or more sealed reaction vessels, known as *cells*, which convert the chemical energy of a reactive electrode into the motion of electrons through a wire.

The basic principle of an electrochemical cell is illustrated by a classic school experiment—a lemon battery (Figure 7-1). The experiment involves jamming two dissimilar metals into the titular lemon; the electrodes are typically copper and zinc. If you connect a voltmeter to the terminals, the lemon should read about 1 V. Stringing two or three lemons in series is enough to light up a small LED.

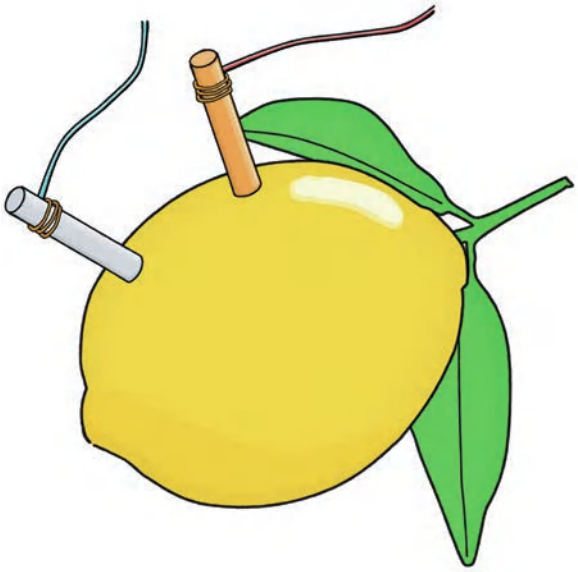


Figure 7-1: A scientific diagram of a lemon battery

Although this cell seems simple, the underlying chemistry is fairly complex and often glossed over in most textbooks. To start, lemon juice is more or less a 5 percent solution of citric acid in water. Water is a small, strongly polar molecule, so, as discussed toward the end of Chapter 2, it can cleave the ionic bonds of many other substances, producing mobile, electrically charged ions. These ions live independent lives in the solution but coalesce back into the original substance if the solvent is removed. In the case of citric acid, we end up with a modest number of positively charged hydrogen (hydronium) cations that are one electron short of an electrically neutral atom, along with negatively charged citrate anions that have some extra electrons in tow.

The overall reaction inside the battery involves zinc donating its electrons to hydronium ions, a reaction that turns the ions into an electrically neutral hydrogen gas and produces zinc cations that dissolve into the solution:

$$Zn\,(s) + 2H^{+}\,(aq) \rightarrow H_2\uparrow + Zn^{2+}\,(aq)$$

That said, this equation doesn't tell the whole story. When the electrodes are connected by a conductor, zinc ions are formed on the zinc electrode and enter the solution there, while hydrogen gas is liberated on the copper side. This process evidently requires the electrons to travel through the conductor, producing electric current along the way. At first blush, it's mystifying what force might be compelling the electrons to take such a circuitous path.

Without turning the chapter into a chemistry textbook, the reaction unfolds in this fashion because the conversion of hydronium ions to hydrogen molecules is a multistep process involving the absorption of atomic hydrogen into the electrode. This step occurs more readily in copper than in zinc. In other words, it's easier for a hydronium ion to snag an electron off copper. On the flip side, copper atoms are not that eager to enter the solution as ions, so the energetically favorable result is for the requisite electrons to be sourced from the zinc side via a conductive path.

More generally, these electron transfers occur because of the quantum-mechanical properties of molecules and solids. Such exchanges can be energetically favorable even when they produce electric fields. A familiar example of this phenomenon is the

p–n junction (Chapter 2), where electrons belonging to one material settle into vacancies in another. These interactions usually don't produce macroscopic voltages or currents, but in a battery, lower-energy products are continually removed, and higher-energy reagents are exposed. This allows for the continuous motion of charges until the material is spent.

Current and Voltage Characteristics

The reaction inside a battery slows down if the electrons can't travel easily between electrodes. The buildup of charges eventually produces an electric field strong enough to prevent any more electrons from trading places. It follows that an ideal battery could be thought of as a constant-voltage source: There is a certain chemistry-specific voltage that the reaction naturally tries to converge on.

In practice, ions are fairly sluggish and can't keep up with arbitrarily high currents, so the voltage drops off in proportion to current drain (Figure 7-2). The magnitude of this effect depends in large part on the physical dimensions of the cell and the surface area of the electrodes. This explains why you can't crank a car with a couple of AA batteries.

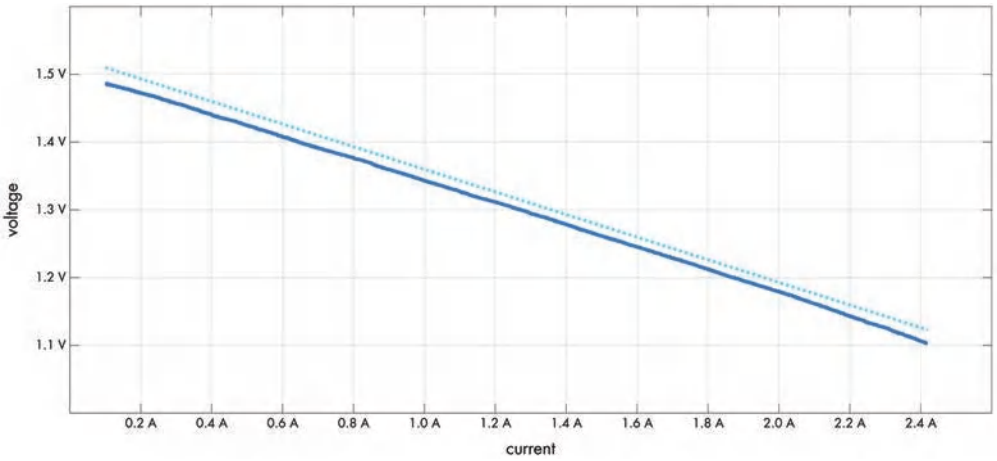


Figure 7-2: Current drain versus terminal voltage for an alkaline AA cell

In Figure 7-2, the solid line shows the measurements I've made for a brand-name alkaline AA battery. The slightly offset dashed line is the Ohm's law plot for the voltage drop we'd expect if we were pushing the same current through an estimated resistance of 160 mΩ.

In effect, a battery can be modeled as an ideal voltage source in series with a small resistor. The resistor's value ranges from a few milliohms for car batteries to tens of ohms for button cells. We call this parameter the battery's *internal resistance*. Another term used for a wider range of electronic components is *equivalent series resistance (ESR)*.

The resistance-based model is useful but not worth getting overly attached to, because the voltage also changes in a far less elegant way as the battery is drained. Figure 7-3 shows the voltage across the terminals of an alkaline 9 V battery being discharged at a constant rate of 200 mA.

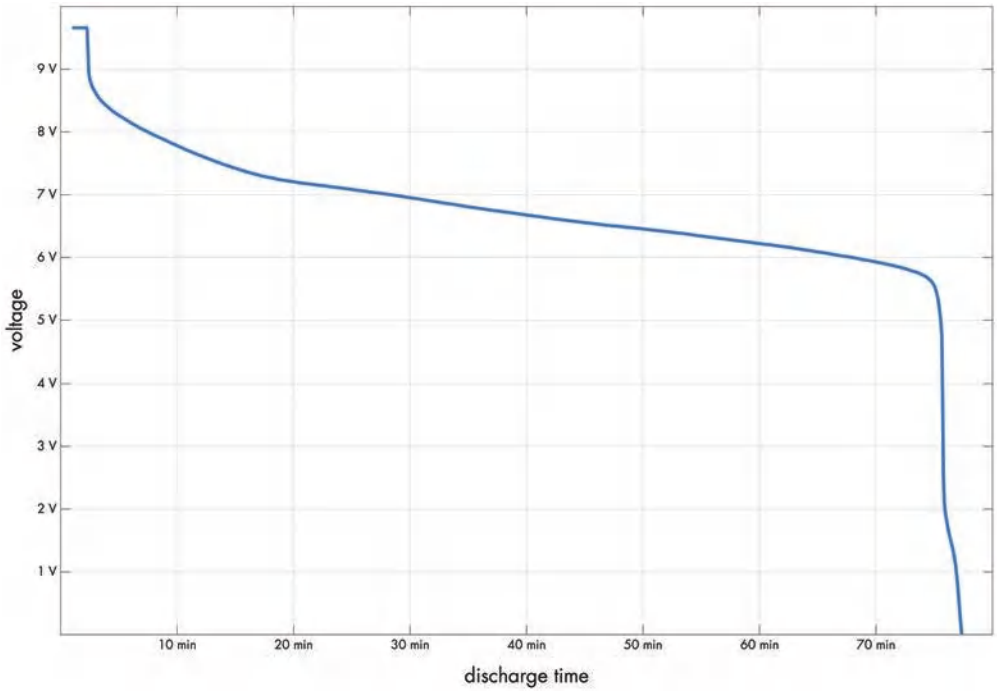


Figure 7-3: Constant-current discharge of a 9V battery (6LR61)

The plot will look different for other chemistries. For example, lithium-polymer batteries have a flatter discharge curve, which complicates the task of estimating their charge state. The apparent capacity of most batteries will also change with temperature (warmer is better) and with current drain (lower currents mean less waste).

A battery's capacity is typically specified in ampere-hours. As discussed in Chapter 3, 1 ampere-hour is simply equivalent to 3,600 coulombs. Table 7-1 provides some common figures.

Table 7-1: Battery Properties by Type

Battery type	Chemistry	Nominal voltage	Typical capacity
377	Silver oxide	1.55 V	25 mAh
CR2032	Lithium	3 V	200 mAh
CR123	Lithium	3 V	1,500 mAh
AAA (L03)	Alkaline	1.5 V	1,000 mAh
AA (LR6)	Alkaline	1.5 V	2,700 mAh
"Nine-volt" (6LR61)	Alkaline	9 V	600 mAh
Mid-size car battery	Lead-acid	12 V	60 Ah

The numbers are given for optimal temperatures and discharge currents. For example, for a typical 6LR61 battery, the manufacturer might test with a current of 2 mA. Earlier, we saw that the same battery was dead within 75 minutes when subjected to a 200 mA load. This means that, under such conditions, we get less than half the rated capacity.

Rechargeable Cells

In many batteries, the direction of the reaction can be reversed if we apply a higher voltage to the terminals. In "non-rechargeable" batteries, the results are almost always disappointing, but if the cell is engineered properly, it might be possible to achieve hundreds of complete charge and discharge cycles before the chemistry deteriorates to the point of no return.

For rechargeable batteries, the optimal charging process depends on battery chemistry, but a common approach is to start with a constant-current stage that replenishes the charge as quickly as possible. The maximum safe current is typically proportional to battery size. The cells can be damaged if the applied voltage is higher than a certain threshold, so the final stages of the charging process use a constant-voltage method. The safety limit is about 4.2 V per cell for lithium-polymer batteries and 2.4 V for lead-acid (keeping in mind that a typical car battery has six cells).

For lithium batteries specifically, the charging process is somewhat perilous—the lithium metal and its complexes are highly reactive and can catch fire if the battery is overcharged or mechanically damaged. Draining the cells below 3 V causes degradation too.

AC Adapters

Along with batteries, inexpensive wall wart adapters are a common way to power electronic devices. These units convert mains AC voltage to more manageable DC levels through a series of steps that will be covered in Part III of the book. For safety, the devices are almost always designed so that there's no DC pathway between any of the output wires and AC mains; this is known as *galvanic isolation*.

Almost all compact AC adapters made today are *switching regulators*. This means they contain an integrated circuit that monitors the state of an output capacitor and replenishes the charge at a high speed if the voltage begins to sag. This active control mechanism allows the adapters to be remarkably compact and energy-efficient. At the same time, the primary design goal is to make them dirt cheap rather than cater to refined tastes. Wall warts don't have the fastest response times, the tightest regulation, or the best noise filtering attainable, so in many applications some form of additional filtering is necessary.

Figure 7-4 shows the output of a typical no-name 5 V adapter from my drawer. The top oscilloscope trace is the no-load output; the voltage is about 7 percent higher than advertised and has high-frequency spikes spanning +/−50 mV. The bottom trace shows the same supply connected to a substantial load. The voltage is noticeably lower and has characteristic valleys due to the control circuitry not quite keeping up with demand. These glitches—together known as *ripple*—are substantial enough to potentially destabilize precision analog instruments or high-speed digital chips.

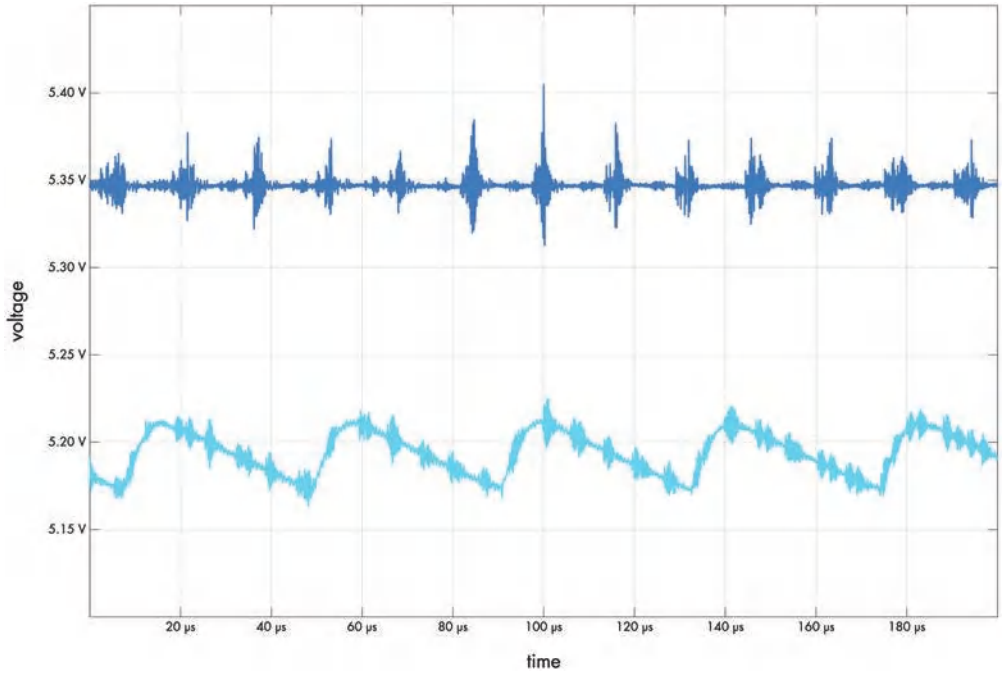


Figure 7-4: AC adapter ripple and voltage drop: no load (top) and 1 A current (bottom)

Imperfect regulation and noise aside, the adapters respond fairly slowly to sudden changes in demand. Figure 7-5 shows the response when a load is being switched on and off at a frequency of about 2 kHz. Every time, it takes about 125 microseconds for the voltage to stabilize.

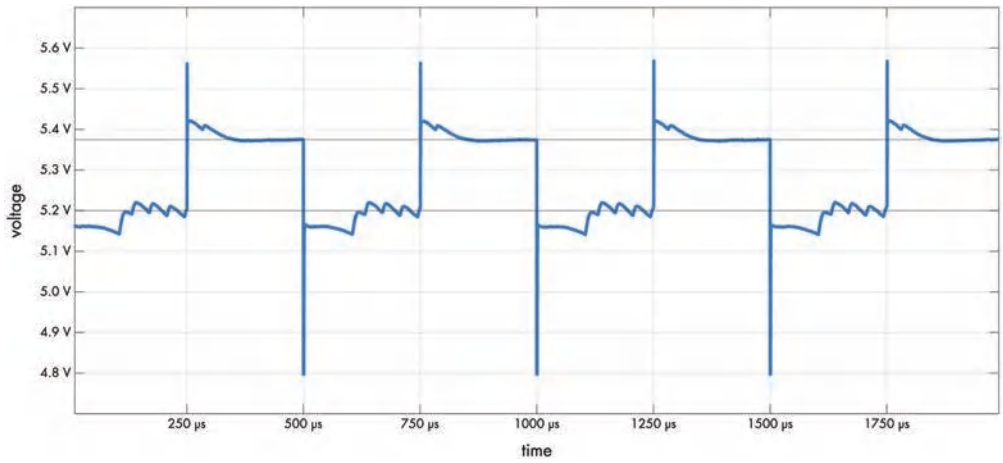


Figure 7-5: The response of an AC adapter to a load being switched on and off at a frequency of 2 kHz

Some supplies can be better, and a handful can be worse. If you're buying them on the cheap or salvaging from existing electronics, there's no way to know what you're getting. As a rule of thumb, voltage regulation within 10 percent, peak-to-peak ripple in the vicinity of 100 mV, and transient response times near 100 µs shouldn't come as a surprise.

If you have an oscilloscope and a spare wall wart, it can be a good exercise to examine its performance under no-load conditions and with a dummy load. A reasonable load that doesn't contribute its own noise is a 10 Ω ceramic resistor rated for 10 watts. From Ohm's law ($I = V/R$), a 10 Ω resistance will admit a current of 500 mA when connected to a 5 V supply. In this scenario, heat dissipation will be $P = I^2R = 2.5$ W.

Solar Cells

Small solar-powered electronics aren't ubiquitous, but they're common enough. The technology is used for environmental sensors, safety and security systems, outdoor accent lighting, old-fashioned calculators, toys, and more. At their core, these devices are just p–n junctions that happen to be exposed to light. Although there are many ways to describe or construct them, one relatively simple structure is shown in Figure 7-6.

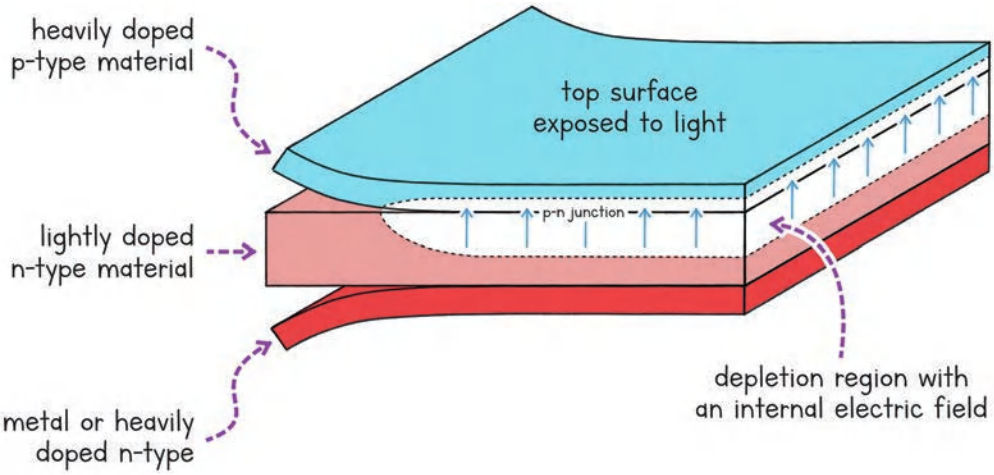


Figure 7-6: Possible structure of a very simple solar cell

The functional part of this rudimentary solar cell is a p-type layer stacked on top of an n-type material. Where the two layers meet, a depletion region forms. The top p-type layer is heavily doped to keep it highly conductive yet as thin and transparent as possible. In contrast, the buried n-type layer is thicker and doped very lightly to ensure that the depletion region extends deeper into the material, improving the odds of photons being absorbed in that area.

Recall from Chapter 2 that the depletion region of a semiconductor junction is largely devoid of mobile charge carriers but is characterized by an internal electric field. If a photon of incident light excites an electron in this area, the electron is swept away toward the n-side, while the hole it leaves behind is pushed toward the p-side. This charge separation process produces a voltage difference between the top and bottom terminals of a solar cell. In the presence of a conductive return path, a sustained current can flow. The open-circuit voltage of a solar cell is somewhat nonlinear and difficult to predict, whereas the generated current is directly proportional

to the rate of charge-carrier generation in the depletion region—that is, to the intensity of light.

Odds and Ends

There are many other ways to generate electricity, but at this stage, they aren't worth an in-depth review. The bulk of the electricity delivered to our homes is produced by moving copper coils through magnetic fields, but this is rare in self-contained electronic devices—except for cheap wind-up flashlights and similar novelties.

In the same vein, thermocouples can produce electricity from temperature gradients, but the efficiency of the process is currently very low, which precludes applications outside of fireplaces and plutonium-powered thermal reactors aboard deep-space probes.

Power produced by conventional means is sometimes delivered wirelessly. This is typically done by bringing two copper coils close together, forming an impromptu transformer. The technology is used for RFID tags and to charge electric toothbrushes and some smartphones. The efficiency is typically poor, especially if precise alignment between the coils can't be guaranteed. Again, we'll talk a bit more about transformers in Chapter 8.

Schematic Symbols for Power Supplies

Schematics are the way we document electronic designs. The drawings represent the structure of the circuit, emphasizing function rather than the physical placement of components. Every component is given an unambiguous graphical symbol, and the connections between symbols are marked with straight lines, usually laid out horizontally and vertically.

We'll discuss the best practices for schematics in Chapter 11, but in the meantime, we can gradually introduce some of the symbols you're likely to see. For power sources, the most common choices are shown in Figure 7-7.

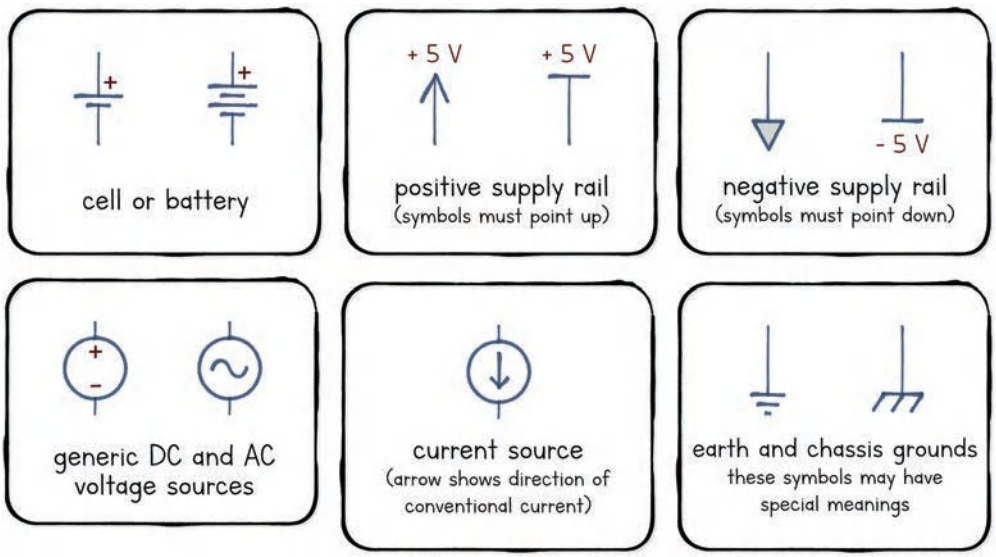


Figure 7-7: Common symbols associated with power supplies

The column on the left represents power supplies as single, two-terminal components. This is the traditional approach, but for DC voltage sources, it's now more common to see split-terminal symbols, as shown on the right. The benefit of this style is that it avoids the mess of long-winded lines needed to connect power to every chip. Instead, we just place a positive or negative rail symbol next to the appropriate pin. It is implied that all of these points are electrically connected.

Note the two special ground symbols in the last panel. The first one, *earth ground*, is supposed to be tied to the grounding pin of the wall outlet. The second, *chassis ground*, connects to the metal enclosure of an appliance. These grounding practices offer operator protection for devices powered directly from mains—for example, if a loose "hot" wire makes contact with other metal parts. These symbols wouldn't be out of place on the schematics of a dishwasher or coffee maker. Less commonly, the approach may be used to protect sensitive circuits from electrostatic discharge or radio interference. Nonetheless, it's common for hobbyists to use these symbols interchangeably with standard ground, which just means a return path to the power supply. It's important to exercise caution and read contextual hints.

Now that we know how to power electronic circuits, let's have a look at some of the components that harness this energy to process information or perform other useful work.

8

COMMON PASSIVE COMPONENTS



Passive components are an important category of devices that includes resistors, capacitors, inductors, and transformers. They're found in virtually every electronic circuit and perform a variety of signal processing tasks. Taxonomically speaking, semiconductor junctions aren't welcome in this club, and most electromechanical parts, such as switches and speakers, are usually considered a separate kingdom as well.

Some introductory texts purport to give crisp definitions of what makes a component "passive," but these explanations seldom survive scrutiny. The most rigorous answer is that passives are described by simple linear equations, while active components, such as transistors, can be governed by more sophisticated math. However, this is an oversimplification. Consider that in the 1950s a number of researchers in Japan exploited the nonlinearity of transformers to fashion them into digital logic circuits known as *parametrons* and built working computers based on this principle.

However we choose to define the term, the assortment of passive devices covered in this chapter should look familiar. We introduced idealized models for most of them in Part I of the book. What remains is a discussion of the practical implementation of these devices, including how the components' behavior can diverge from basic theory.

Resistors

We introduced the concept of a *resistor* in Chapter 3: It's a device that impedes the flow of electric current in proportion to the component's value and in inverse proportion to

the applied voltage ($I = V/R$). In contrast to capacitors and inductors, an ideal resistor has the same effect for any signal frequency. Figure 8-1 shows the typical resistor schematic symbols.

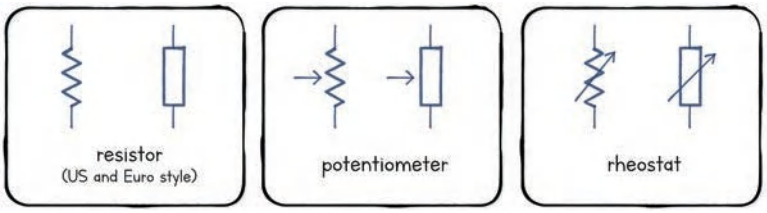


Figure 8-1: Schematic symbols for fixed-value and adjustable resistors

Resistors are among the simplest electronic components. Nevertheless, they come in a variety of flavors and exhibit some subtle physical properties, so it's prudent to give them a closer look.

Fixed Resistance

Most resistors are manufactured to hold a fixed value, typically between 10 Ω and 1 MΩ. As a matter of convention, the values are distributed on a roughly logarithmic scale that divides each *decade* (every multiple of 10) into a fixed number of steps. The two most common scales, E6 and E12, have the following progression of per-decade values:

E6 1.0, 1.5, 2.2, 3.3, 4.7, 6.8

E12 1.0, 1.2, 1.5, 1.8, 2.2, 2.7, 3.3, 3.9, 4.7, 5.6, 6.8, 8.2

In practical terms, this means it's common to encounter 2.2 kΩ, 22 kΩ, and 220 kΩ resistors. In contrast, values such as 2.5 kΩ, 25 kΩ, and 250 kΩ will be harder to find and might cost more.

The scales might look odd, but they play nice with manufacturing tolerances. For example, the E6 scale ensures almost no overlap between component values if parts are made with an accuracy of +/−20 percent. This tolerance level was common when resistors were made from carbon powder held together by a binder. While somewhat of an anachronism today, the convention endures.

Strictly speaking, an improved version of the carbon process still exists, but most modern resistors are made from metal films of precisely controlled composition and thickness. For this manufacturing method, tolerances of 5 percent are standard, and 1 percent accuracy costs only a tiny bit more. Parts accurate to 0.1 percent are also available, but they aren't worth the price unless you have a very specific itch to scratch.

In high-power or high-precision applications, it's also possible to encounter wire-wound resistors. As the name implies, these components are made by wrapping a run of resistance wire along a nonconductive core. This construction method offers better heat dissipation, tolerances down to 0.01 percent, and exceptionally stable performance across a wide range of temperatures. However, wire-wound resistors are costlier and bulkier than metal film; moreover, depending on the winding method, they can exhibit nontrivial inductance.

Variable Resistance

Mechanically operated adjustable resistors known as *potentiometers* are commonly used for analog input and continuous position sensing. They can be found behind volume

knobs, inside the thumb-stick assemblies of gaming controllers, and in servo motors of remote-controlled toys. Smaller, board-mounted *trim pots* are sometimes employed in precision instruments for one-time calibration after assembly. In some applications, adjustable resistors are being supplanted by buttons, touchscreens, or digital encoders. Even still, the original component is nowhere near extinct.

A standard potentiometer consists of a resistive track connected to a pair of electrical terminals, while a sliding wiper forms the third terminal (Figure 8-2). In the simplest case, a signal of interest can be supplied to one end of the track. Turning the knob changes the distance that electricity must travel before reaching the wiper, thus altering the component's resistance.

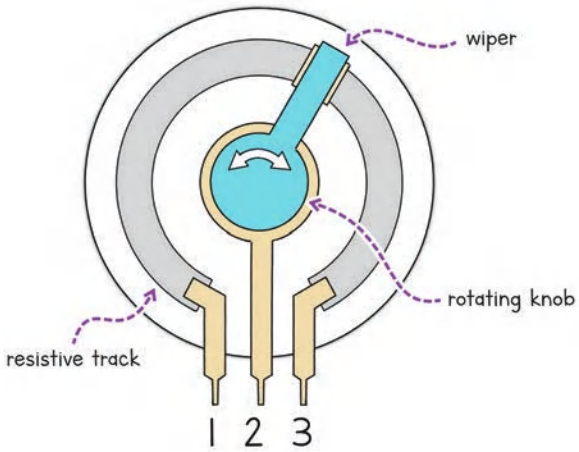


Figure 8-2: The internals of a potentiometer

When a potentiometer is used as a two-terminal device, it's sometimes called a *rheostat*, a throwback to a distinct type of adjustable resistor that's now largely extinct. In this mode, the device simply presents itself as a variable resistor. More advanced setups are also possible. For example, if two different audio signals are provided to terminals 1 and 3, the signal on the wiper terminal is an adjustable mix of the two.

Potentiometers can be tailored in a number of ways. The resistive track can be shaped to achieve a nonlinear relationship between knob position and resistance; for instance, logarithmic tapers are used in audio applications to track perceived loudness more naturally. You can also buy precision potentiometers with a built-in worm drive mechanism, multi-gang devices where a single knob controls a number of separate circuits, and more.

In addition to mechanically operated potentiometers, it's also possible to construct sensors that vary their resistance in response to other phenomena. For example, in Chapter 2, we noted that the conductivity of homogeneous semiconductors can change in response to illumination; this is the basis of *photoresistors* (also known as *light-dependent resistors*, or *LDRs*). In the same vein, temperature-dependent changes in conductivity in semiconductors and metals form the basis of *thermistors* and *resistance temperature detectors (RTDs)*. You might not encounter them often, but they're worth keeping in the back of your head.

Resistor Power Ratings

In Chapter 3, we noted that resistors dissipate energy as heat in a fashion governed by Joule's law ($P = V^2/R$). If the heat can't be removed as quickly as it builds up, the component will eventually overheat and fail. The failure usually manifests as an open

circuit, but it can also cause the resistance to get out of whack. This is why every real-world resistor has a *power rating*. In low-voltage circuits, these ratings matter only for resistances below 1 kΩ or so; from Joule's law, it would take about 100 volts to dissipate a noteworthy amount of heat across a resistance of 100 kΩ.

Standard through-hole resistors are typically rated for 250 mW, while their surface-mount counterparts might range from 100 to 250 mW, depending on size (we'll circle back to component form factors in Chapter 28). Power resistors are available with ratings well over 1 kW. At that point, though, it might be cheaper to buy a space heater.

As for potentiometers, although specialized high-power models are available, standard varieties are fairly fragile. Heat aside, substantial currents or voltages can cause sparking and premature wear of the resistive element. Many low-cost potentiometers are rated for 100 mW or less.

Component Parasitics

All real-world components exhibit some unwanted characteristics known as *parasitics*. To be fair, standard metal-film resistors are pretty close to the theoretical model we introduced in Chapter 3. Power dissipation limits aside, their performance in everyday circuits doesn't change appreciably in response to temperature, signal frequency, current, or voltage.

Nevertheless, even these simple components have a few surprises up their sleeves. For instance, although a resistor isn't meant to be an inductor, a tiny amount of inductance will appear simply because the current flowing through the device produces magnetic fields. We can model this parasitic inductance by imagining an ideal inductor connected in series with an ideal resistor. For film resistors, the inductance is typically on the order of 1 to 2 nH, about the same as a standard PCB trace of a similar size. For wire-wound resistors, it can be much higher.

In Chapter 5, we derived the following formula to describe the resistance-like tendency of inductors to attenuate alternating currents:

$$X_L = 2\pi fL$$

In the formula, L is inductance and f is frequency. The value increases in proportion to f, so it stands to reason that at a sufficiently high frequency, parasitic inductance could make a resistor impede the propagation of signals more than implied by the component's rated value. But because resistors are meant to have a fairly high impedance to begin with, the effect is typically of little consequence.

Figure 8-3 shows the frequency response curve for R = 10 Ω and L = 1 nH. The distortion caused by the parasitic inductor becomes significant only as we approach 1 GHz. For higher values of R, the crossover point would be even farther to the right.

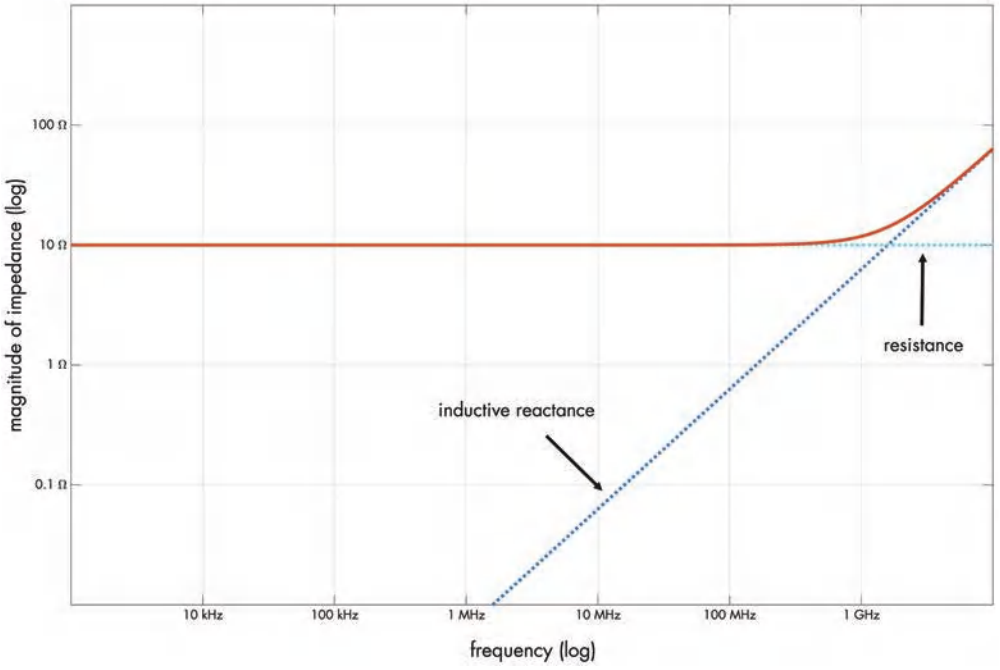
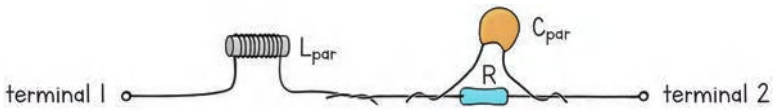


Figure 8-3: A model of an ideal 10 Ω resistor in series with a 1 nH inductor

Alas, this R-L model tells only part of the story when it comes to real-world resistor behavior. There are several ways to model the remaining parasitics, but the most important element we're missing is parasitic capacitance:



The capacitance is present because the resistor is nothing more than a pair of conductive terminals with a slice of poorly conductive material sandwiched in between them—in other words, it's a poorly designed capacitor. The value of C_{par} is typically minuscule, around 100 to 500 fF (femtofarads), but it's important because it's a pathway that allows higher-frequency currents to bypass the resistive substrate altogether.

Again from Chapter 5, recall the following formula for capacitive reactance:

$$X_C = \frac{1}{2\pi fC}$$

If we crunch the numbers for C = 300 fF and f = 5 MHz, X_C works out to about 106 kΩ. This is noteworthy—the figure is comparable to a plausible value of R. At this signal frequency, we might not see much of an issue with a 10 kΩ resistor, but the impedance of a 330 kΩ resistor will unexpectedly take a nosedive (Figure 8-4).

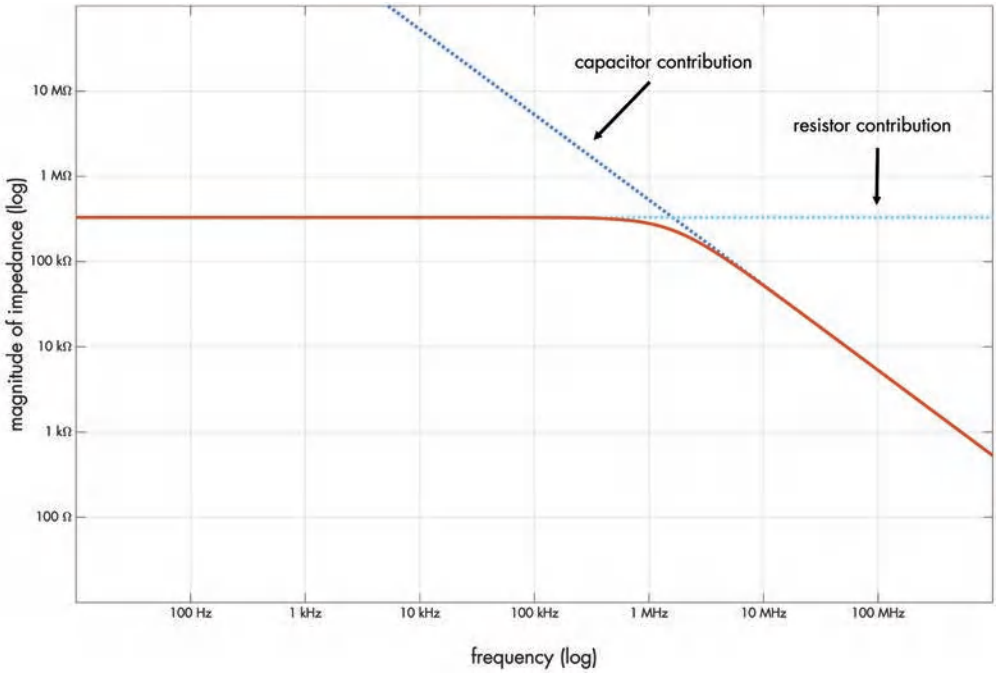


Figure 8-4: Frequency response of a 330 kΩ resistor with a 300 fF parasitic capacitance

The combination of rapidly changing signals and high resistances isn't unheard of in electronics; for example, it comes up in the design of amplifier feedback loops (Chapter 15). The topic of signal amplification also brings us to another real-world gotcha: a resistor's tendency to produce noise.

Johnson–Nyquist Noise

In any conductor, random thermal fluctuations can produce momentary shifts in the distribution of charge carriers. In classical terms, we could say that the atoms vibrate and occasionally toss some of the free electrons around. In good conductors, the process doesn't result in noteworthy voltage fluctuations because the resulting electrostatic fields are equalized nearly instantly and can't build up over time. In resistive materials, on the other hand, the equalization is hindered. This means we can observe a degree of inherent voltage noise. The phenomenon, known as *Johnson–Nyquist noise*, grows in proportion to resistance and temperature.

In most cases, the magnitude of the effect is too small to matter. The voltages are barely measurable with standard lab instruments and are usually eclipsed by other sources of interference, such as power supply ripple. However, in precision amplifiers designed to work with faint signals, large resistances placed in the signal path can severely compromise circuit performance. Resistor noise is an issue in radio receivers, microphone amplifiers, and more.

Thermally induced voltage fluctuations appear uniform at all frequencies—they look the same whether we examine the spectrum near 10 kHz or 1 MHz. In effect, we can model the phenomenon as a series of independent, "fractional" noise generators operating in different frequency bands, each pumping out a signal with the same (narrow) bandwidth and amplitude. Depending on the range of frequencies allowed through by the circuit, we end up with a varying number of these fractional noise sources that add up to a voltage observable on the oscilloscope screen.

The behavior of this sum can be analyzed with different degrees of mathematical rigor. That said, to build basic intuition, we can imagine a pair of uncorrelated, uniform noise signals that take only two values: $-a$ or $+a$. When we add the signals together, there are four possible scenarios, each occurring with equal probability, as outlined in Table 8-1.

Table 8-1: A Toy Model of Summed Noise

Signal 1	Signal 2	Sum
$-a$	$-a$	$-2a$
$-a$	$+a$	0
$+a$	$-a$	0
$+a$	$+a$	$+2a$

How do we describe the overall effect of the summed voltages? The most obvious answer is the arithmetic mean, but the arithmetic mean of any uniform noise is zero. Indeed, we learn nothing useful if we calculate the average of the four possible values in Table 8-1:

$$\frac{(-2a) + 0 + 0 + (2a)}{4} = 0$$

The peak value might seem more promising, but the folly of this approach is evident if we consider a model with a higher number of summed signals (or a greater variety of possible voltage levels). For example, if we extend Table 8-1 to 10 input signals, the number of rows in the table would balloon to 1,024, and yet the peak values would appear only in the two outlying rows, which correspond to the component signals being all negative or all positive. As we move away from discrete-value models toward continuous phenomena in the real world, the probability of hitting the theoretical peak becomes vanishingly low. Consequently, the value becomes meaningless.

In Chapter 4, we introduced the notion of root-mean-square voltage. This parameter is calculated by taking the square root (R) of the arithmetic mean (M) of the squared waveform (S). On the surface, RMS isn't well suited for measuring voltage noise. In most applications, we aren't interested in the actual power delivered by these thermally induced flukes. Instead, we want to know the amount of distortion superimposed on top of the voltage signal we care about.

Nevertheless, RMS does have three useful properties. First, the value is nonzero for symmetrical waveforms. Second, the parameter it represents, power, plays nicely with other equations in physics. Most important, the way we calculate RMS has a serendipitous similarity to standard deviation in statistics, which describes the probability that a measurement of a random process would fall within some distance of the arithmetic mean (here, zero volts). The standard deviation is computed by taking the square root of the mean of squared "errors" from the baseline. In effect, RMS is *exactly equal* to the standard deviation for ideal thermal noise.

But let's not get ahead of ourselves. In the equal-probability scenario outlined in Table 8-1, regardless of the order in which the summed values appear in the waveform, the RMS will be:

$$V_{RMS} = \sqrt{\frac{(-2a)^2 + 0^2 + 0^2 + (2a)^2}{4}} = \sqrt{\frac{8 \cdot a^2}{4}} = \sqrt{2} \cdot a$$

We can also try this for a three-signal scenario (Table 8-2).

Table 8-2: A Three-Signal Model of Summed Noise

Signal 1	Signal 2	Signal 3	Sum
–a	*–a*	*–a*	*–3a*
–a	*–a*	*+a*	*–1a*
–a	*+a*	*–a*	*–1a*
–a	*+a*	*+a*	*+1a*
+a	*–a*	*–a*	*–1a*
+a	*–a*	*+a*	*+1a*
+a	*+a*	*–a*	*+1a*
+a	*+a*	*+a*	*+3a*

In this case, the resulting multiplier is $\sqrt{3}$. In effect, for any number n of independent signal sources, we arrive at an RMS value equal to the amplitude of the individual component signal (a) multiplied by $\sqrt{n}$.

The component noise contributed by a given resistor at a given temperature can be expressed using the concept of *noise amplitude spectral density* (e_n). The parameter corresponds to the voltage swing contributed by a 1 Hz slice of the spectrum. It's akin to a single "fractional" generator in the model introduced earlier. The unit of e_n is $\mathrm{V} / \sqrt{\mathrm{Hz}}$. Per the earlier discussion, to get the total RMS voltage noise for a particular slice of the frequency spectrum—a specific bandwidth Δf expressed in hertz—we simply multiply e_n by the square root of Δf.

Without getting too deep into thermodynamics, for a given resistance R and temperature T_K in kelvins, the formula for Johnson–Nyquist noise density is pretty straightforward:

$$e_n = \sqrt{4 \cdot k_B \cdot T_K \cdot R}$$

In the equation, k_B is the physical Boltzmann constant that relates energy to temperature. It's equal to about 1.381×10^{-23} joules per kelvin. At room temperature, if we're willing to play fast and loose with units, the formula can be simplified to:

$$e_n \approx 0.129 \cdot \sqrt{R} \quad (\text{units: } \mathrm{nV}/\sqrt{\mathrm{Hz}})$$

As a practical example, for a 10 kΩ resistor, thermal noise density is around $13\ \mathrm{nV} / \sqrt{\mathrm{Hz}}$. In a microphone amplifier designed to pass a bandwidth of 20 kHz, the input noise contributed by that resistor to the signal will be in the vicinity of 1.3 µV.

Keep in mind that the model deals with RMS values and RMS voltage isn't something we can see directly on an oscilloscope screen. As noted earlier, however, for well-behaved thermal noise, V_{RMS} happens to be equal to the notion of standard deviation in statistics. Further, basic statistics tells us that for a random process of this nature, more than 99.7 percent of measurements should fall within three standard deviations (3σ, or $3 \cdot V_{RMS}$) of 0 V. In other words, the highest peak-to-peak voltage noise you're likely to see on the oscilloscope should be less than $6 \cdot V_{RMS}$.

The formula discussed in this section represents the theoretical noise floor. In practice, depending on their construction, resistors also exhibit some amount of excess noise above this baseline. That noise might have a different physical basis and isn't distributed uniformly across all frequencies. In practical terms, carbon resistors perform worse than wire-wound or metal film types. Notably, thermal noise also crops up in parasitic resistances of non-resistor components.

Capacitors

We introduced the ideal capacitor in Chapter 5. It's a device that allows a certain amount of charge (Q) to be stored in an internal electric field through the application of an external voltage. Another way to look at it is that the component admits a current proportional to the step change in the applied voltage. In sine-wave applications, a capacitor exhibits an impedance that decreases with signal frequency.

Like resistors, preferred capacitor values lie on a logarithmic scale. Standard manufacturing tolerances hover around 10 to 20 percent, so the six-step E6 scale (introduced earlier in the chapter for resistors) is the most common choice. The components usually range from 1 pF to 1 mF, although comparatively costly *supercapacitors* reach thousands of farads, providing a specialized alternative to rechargeable batteries. The common schematic symbols for capacitors are shown in Figure 8-5.

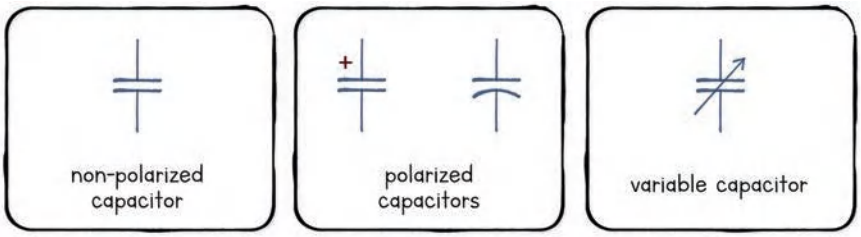


Figure 8-5: Common symbols for capacitors

While capacitors are conceptually simple, the pursuit of miniaturization makes the manufacturing process quite involved. The goal is to maximize the surface area of the plates and make the insulating layer as thin as possible. This results in elaborate three-dimensional structures with some interesting performance trade-offs.

Types and Ratings

For decades, general-purpose capacitors were made using a wide range of processes tailored to different needs. Insulators ranged from paper to mica, Teflon, and polypropylene film. Plates were sometimes made from exotic materials such as niobium and tantalum. Today, however, most of that variety has disappeared. A typical PCB will often feature just two types of capacitors: can-shaped aluminum electrolytics and clay-colored multilayer ceramic caps (MLCCs).

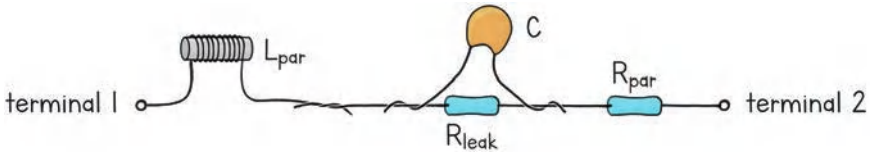
To understand electrolytic capacitors, it's important to note that aluminum is a reactive metal. When exposed to air, it readily forms a thin oxide layer that's nonconductive and chemically similar to rust—except that it stays in place. To make an electrolytic capacitor, aluminum foil is etched to create pits that maximize its surface area. It is then chemically treated to produce a uniform oxide film on the surface. This layer acts as the capacitor's dielectric, with the underlying aluminum serving as one plate. The other plate consists of a conductive electrolyte—an ionic liquid that seeps into every nook and cranny of the etched surface. Because of the nonconductive layer on the aluminum, direct currents can't flow between the capacitor's terminals, but charge carriers on both sides can get much closer than would be possible with a pair of solid electrodes.

Electrolytic capacitors offer remarkably high capacitance in a compact package, but they have two disadvantages. First, the electrolyte has a measurable resistance whenever the capacitor is charged or discharged, so the capacitor can't supply high currents. Second, the component must be used with the correct polarity. A negative voltage applied to the aluminum foil can destroy the insulating layer.

In contrast to electrolytics, MLCCs are made in a more traditional way. They consist of alternating layers of metal and ceramic. Historically, the technology offered limited capacitance, but improvements in manufacturing processes and in the choice of ceramic material have made MLCCs the go-to option for capacitances up to 10 µF, with higher ratings on the horizon. Beyond that point, electrolytics still reign supreme.

Component Parasitics

Similar to what we've done for resistors, we can model the behavior of a real-world capacitor by combining several idealized passive components:



The component's series resistance, R_{par}, is the most straightforward part of the model. It limits the maximum current flowing through the component and dissipates heat. As discussed earlier, the value is fairly high for traditional electrolytic capacitors, typically ranging from 100 mΩ to several ohms. Modern aluminum-polymer chemistry performs better, with values between 10 and 50 mΩ commonplace. For MLCCs, the resistance is typically low enough to ignore.

The other resistor included in the model, R_{leak}, represents the inherent conductivity of the insulator between the capacitor's plates. This phenomenon allows a tiny direct current to flow through the device. For a standard MLCC, R_{leak} may be in the vicinity of 100 MΩ; for electrolytics, it might read closer to 5 to 10 MΩ. The most practical implication is that a low-voltage 100 µF electrolytic capacitor, if left to its own devices, will typically self-discharge within an hour or so. High-voltage capacitors in mains-powered appliances and industrial machinery have beefier insulating layers, so they hold dangerous charge longer.

The only remaining parasitic parameter is the series inductance, L_{par}. As with resistors, this represents the potential for storing energy in the magnetic field surrounding the component. The value is small—often around 1 to 4 nH for miniature capacitors—but because capacitors are often expected to supply substantial momentary currents, it remains a concern.

To illustrate the issue, have a look at Figure 8-6, which shows the behavior of a series L-C circuit with a fixed L and three different values of C.

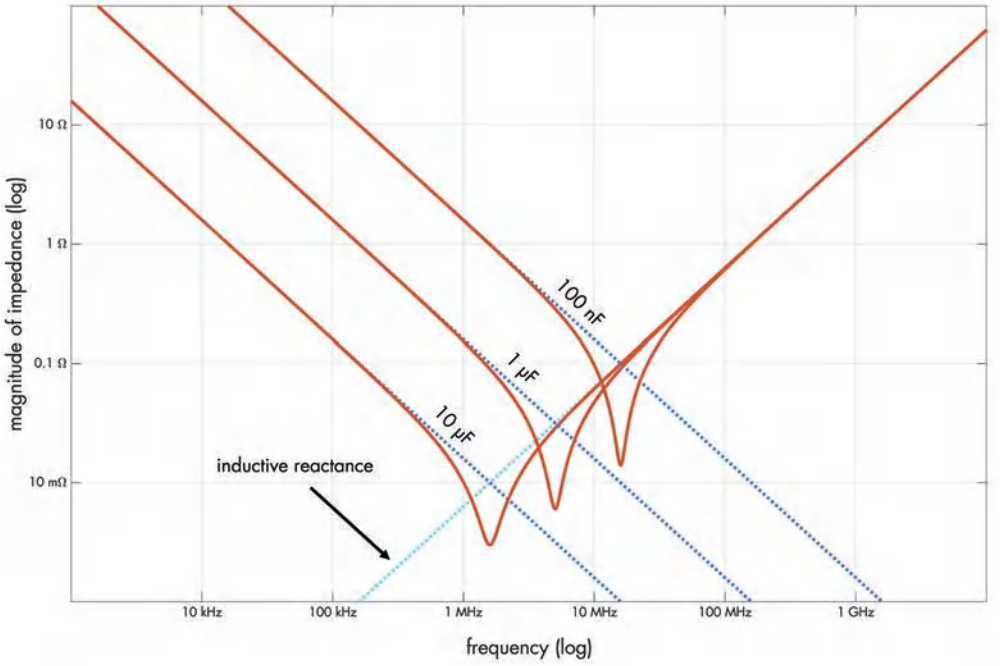


Figure 8-6: A simulation of MLCC frequency response for various capacitance values, showing the interaction of the component's intended capacitive reactance and parasitic inductive reactance

The shape of this curve should be reminiscent of the examples discussed in Chapter 5. It tells us that a capacitor behaves as expected only up to a certain frequency. Past that point, inductive effects take over, and the overall impedance starts climbing. Common MLCCs suffer from markedly degraded performance with signals above 200 to 300 MHz. Specialized low-impedance designs are available, but they cost more.

Effects of Temperature

Electrolytic capacitors are affected by low temperatures; below −20°C (−4°F), the mobility of ions in the electrolyte is markedly reduced, causing the resistance to increase and the capacitance to drop. Around −40°C (−40°F), the component's series resistance (ESR) might be higher by a factor of 10, while the capacitance may be down by about 30 to 40 percent. At elevated temperatures, an inverse effect can be observed, although on a more modest scale.

The response of MLCCs to temperature is more complex. Capacitors made with certain types of ceramic materials lose capacitance at both cold and hot temperatures. Class III ceramics, denoted by codes such as Z5U and Y5V, are of special concern. Y5V may lose up to 80 percent of the rated capacitance if operated at 80°C.

The same behavior is present in class II ceramics, but the losses aren't as pronounced. X5R and X7R capacitors are stable within +/−15 percent, making them good enough for most indoor and outdoor purposes. If better stability is desired, class I materials (C0G) can be used, but this comes at the cost of lower attainable capacitance for a given component size.

Dielectric Saturation

A related property of class II and III materials is that their apparent remaining capacitance drops as the voltage across the terminals approaches the rated maximum. In

an ideal capacitor charged by a constant-current power supply, we would expect the voltage to ramp up linearly over time, and this is what we see in a typical electrolytic design. In an MLCC, however, we see the voltage spiking as the charging process continues. The spike occurs because, past a certain point, it suddenly takes less current—fewer electrons pushed onto the plates—to produce the same Δv (Figure 8-7). In effect, the component's capacitance doesn't stay constant. Instead, it decreases in relation to charge state.

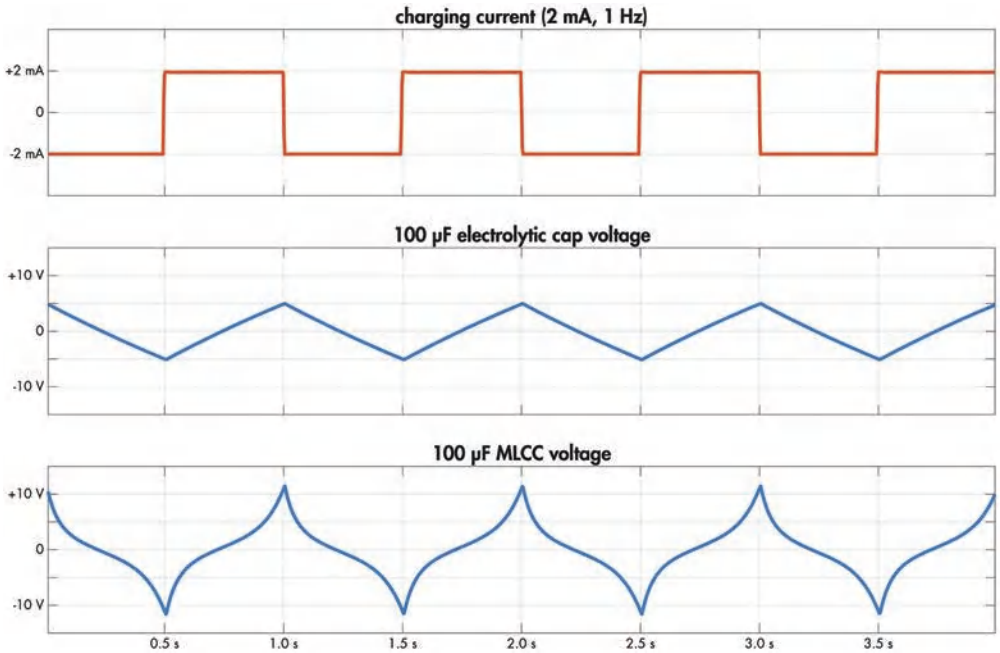


Figure 8-7: An oscilloscope trace showing the nonlinearity of an MLCC near its rated voltage

The reduction in capacitance manifests as a distortion of AC signals with a large voltage swing. It also changes the response to low-amplitude signals with a substantial DC offset, essentially giving you a capacitor with less C for your buck.

In class III ceramics, the loss of capacitance near the rated maximum voltage might approach 80 percent; for X5R and X7R, the losses may be around 50 percent. In physical terms, the effect arises from the presence of permanent electric dipoles in the crystal lattice of the ceramic material. These dipoles elastically reorient themselves in response to an external field, then bounce back when the field disappears. This allows the storage of additional energy in the insulator, similar to how the performance of inductors is enhanced by a ferromagnetic core. Even so, the number of permanent dipoles in the ceramic is limited, and their response is strongest just as the field flips (at around 0 V). Once most of the dipoles are rotated, the capacitor's continued performance isn't quite as advertised.

In practice, this effect typically isn't worth modeling mathematically. Instead, if a ceramic capacitor is to be operated with a DC bias or in the presence of high-amplitude signals, it's best to opt for a larger capacitance. Alternatively, if the circuit depends on a specific value of C, we can also pick a capacitor rated for a higher voltage or shop for a class I dielectric.

Age-Related Considerations

Temperature and DC offsets aside, both electrolytic and ceramic capacitors are prone to aging. In electrolytics, the deterioration of the electrolyte results in a gradual increase in ESR, along with some loss of capacitance. Many electrolytics keep working for decades, but if operated at elevated temperatures and subjected to electrical stress, they might fail sooner.

In ceramics, aging results in a loss of capacitance due to the relaxation of internal stresses in the crystal lattice, which reduces the number of available permanent dipoles. The losses can amount to 30 percent or more. On the flip side, they're easily reversible by briefly heating the capacitor to about 150°C.

These aging-related phenomena mean that it's unwise to design circuits in a way that depends on capacitance remaining static in the long haul. Capacitors are indispensable, but they're not precision instruments.

Inductors and Chokes

As discussed in Chapter 5, inductors develop a voltage across their terminals that's proportional to the rate of change in the applied current. In sine-wave applications, an inductor has an impedance that increases with signal frequency. Figure 8-8 shows the schematic symbols for inductors and related components.

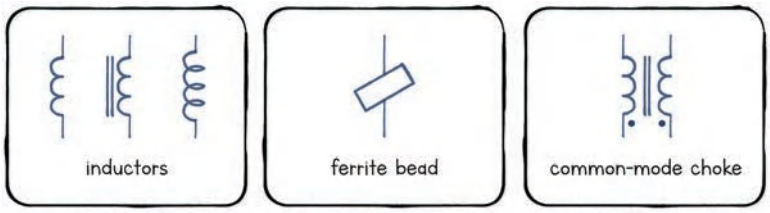
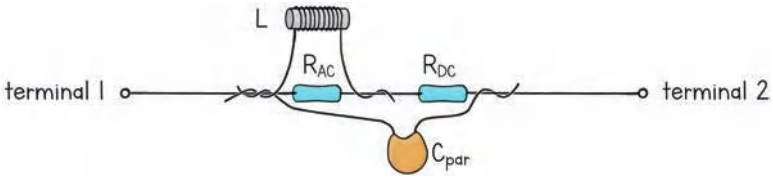


Figure 8-8: The symbols for inductors and chokes

Compared to capacitors, inductors aren't necessarily more nuanced, but they tend to be messier. Magnetic energy is harder to store and contain, and the requisite long runs of conductor tend to have fairly substantial resistance, especially in devices optimized for size and cost.

Component Parasitics

A reasonable approximation of a real-world inductor is shown here:



The series resistance, R_{DC}, represents the losses in the windings. For small PCB-mounted inductors, the resistance is nothing to sneeze at; values ranging from 100 mΩ to tens of ohms are a common sight. This resistance must be taken into account when choosing an inductor for a specific use.

The parallel resistor straddling the windings, R_{AC}, is a more curious beast that, at least in theory, models energy losses in the inductor's core. An ideal inductor would

have $R_{AC} = \infty$. In practice, the value is finite, but it's also not exactly a well-behaved resistance; the parameter varies with frequency, is seldom disclosed by the manufacturer, and can be tricky to compute.

In the absence of a better-differentiated model of energy losses, we instead lean on the *Q factor*, a more widely reported value that represents the "all-causes" ratio of energy stored in an inductor to the energy dissipated in every cycle of a sine wave that's running at some benchmark frequency.

To play nice with other equations that use the familiar approach to sine-wave timing expressions, the ratio is customarily scaled by 2π:

$$Q = 2\pi \frac{E_{stored}}{E_{dissipated}}$$

This means that an inductor with 50 percent efficiency can be described as having a Q of 2π (≈ 6.3). A more sensible value of $Q = 20$ corresponds to an efficiency of about 76 percent; a factor of 40 gets us close to 87 percent.

The remaining component in the equivalent circuit is the parallel parasitic capacitance that straddles the inductor. This is a bypass route for AC signals that arises from having two good conductors (the component leads) with a higher-resistance section (R_{DC}) sandwiched between them. In a small PCB-mounted inductor, the effects of this are usually negligible up to 500 MHz or more, but they markedly affect the performance of the component as you approach GHz frequencies. Figure 8-9 shows a model for a couple of well-behaved inductors.

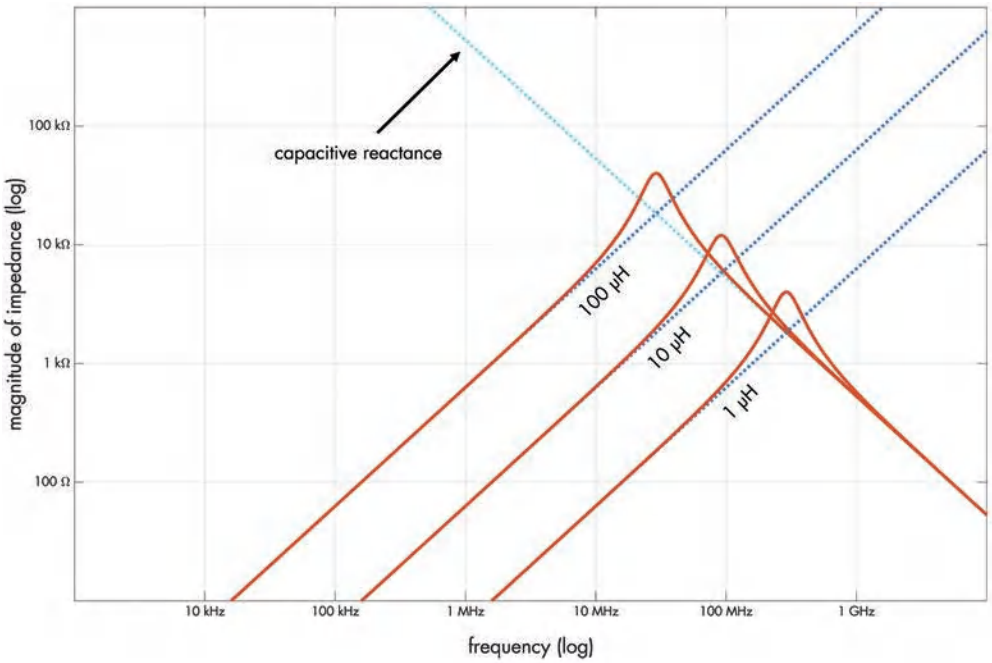


Figure 8-9: Simulated frequency response for chip inductors

As you can see, the plot of the circuit's frequency response is essentially the inverse of the curve for capacitors.

Core Saturation and Hysteresis

Earlier in the book, we noted that the storage of energy in a ferromagnetic core arises from the elastic reorientation of microscopic magnetic domains in the material. Similar to electric dipoles in ceramic capacitors, the number of these domains is limited, and their response is strongest as the field begins to ramp up. As a consequence, there's a certain field intensity—a certain current—past which the core becomes saturated and can't soak up any more energy. If this happens, further increases in magnetic field intensity can occur with relatively little effort, and the device's performance is momentarily compromised. Figure 8-10 shows the oscilloscope capture of a small inductor driven by a square-wave input voltage.

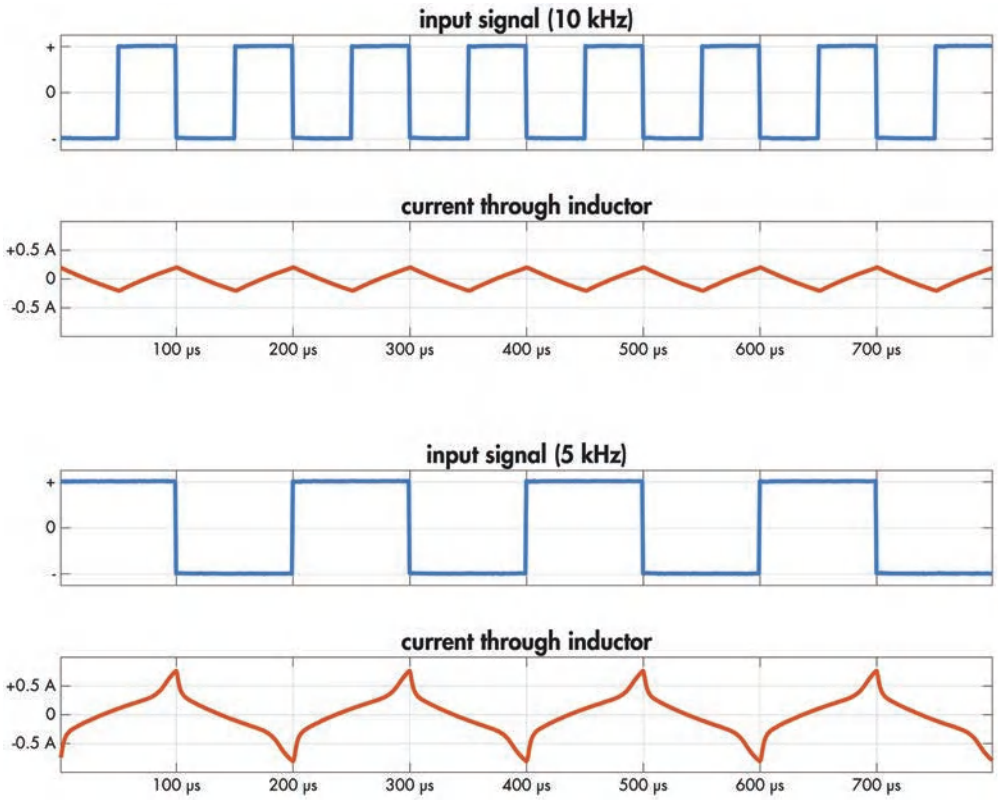


Figure 8-10: Normal inductor behavior (top) and core saturation (bottom)

In the plot, every time the polarity of the driving voltage changes, the current starts ramping up in the opposite direction at a rate dictated by the component's inductance. In an ideal inductor, this ramp should be linear, producing a triangular wave pattern. This happens up to a point, but if the signal frequency is lowered, the ramp eventually crosses the saturation current of about 300 mA. Past that threshold, further changes in current take less work, resulting in a sudden spike.

Because of this effect, most inductors have two current limits. One is the absolute safety limit set by the ability to dissipate heat, and the other arises from the saturation characteristics of the core. Many small inductors are designed so that both values are roughly the same, but this isn't a universal rule.

Another interesting observation about Figure 8-10 is that the bottom plot of current is not perfectly symmetric—the downward ramp isn't a mirror reflection of the upward ramp. A close-up of this behavior appears in Figure 8-11.

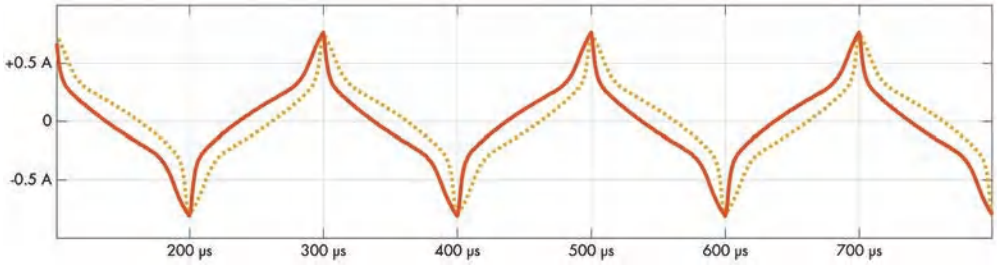


Figure 8-11: A portion of the inductor current plot when subjected to a square wave signal. The dotted line is the mirror image of the waveform.

This occurs because the response of magnetic dipoles isn't as strong as that of the electric dipoles in a capacitor. Instead of bouncing back elastically, some remain in a magnetized state. It takes energy to rotate them back and forth, but that energy is never returned to the circuit. This phenomenon, known as *magnetic hysteresis*, is a significant cause of losses in the core, although not the only one.

Ferrite Beads and Common-Mode Chokes

In addition to general-purpose inductors, there is a related category of devices called *ferrite beads*, designed specifically for noise suppression. High-frequency noise is a significant issue in modern electronic circuits, and inductors seem like a suitable solution. Their impedance increases with frequency, helping to confine rapid voltage spikes to one side of the circuit. Equally important, their frequency characteristics look better than the performance curves of standard MLCCs.

At the same time, when dealing with noise, we don't need the energy to be stored and returned to the circuit down the line. Ferrite beads address this by having deliberately high losses in the core, meaning they dissipate the bulk of the energy captured in their magnetic field as heat. These components are described by their DC resistance and their AC resistance at a chosen target frequency. A run-of-the-mill ferrite bead used in digital circuits may have R_{DC} = 50 mΩ and R_{AC} = 100 Ω at 100 MHz.

A clever variation on the theme is a two-conductor device called a *common-mode choke*. Its most basic form is a tubular ferrite element that's commonly found on computer cables. A more compact version uses a smaller ferrite toroid, with each conductor passing through the central hole of the toroid front to back a couple of times (Figure 8-12).

If there's a common-mode current—that is, if there's a current flowing through both wires in the same direction—the result is a coherent magnetic field that magnetizes the ferrite. In this mode, the device behaves like a typical inductor, passing through DC but aggressively attenuating higher frequencies.

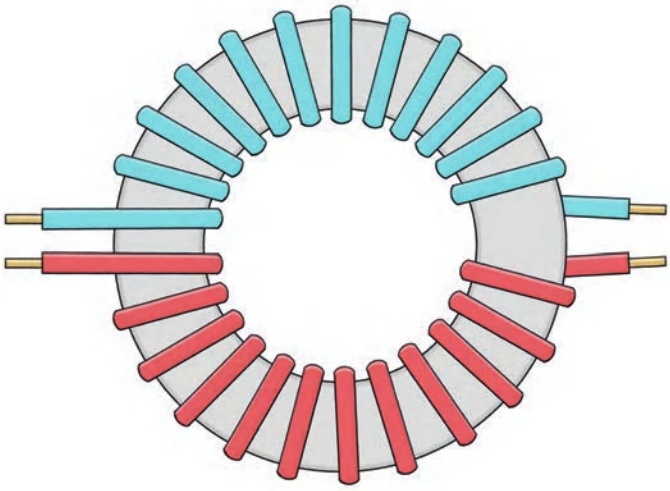


Figure 8-12: A simple common-mode choke

Conversely, if complementary currents flow in opposite directions—as would be the case for "+" and "−" wires connecting to an AC adapter or for certain types of digital signaling—the fields produced by the wires cancel out, resulting in no net field acting on the core. This precludes magnetization. With no energy soaked up by the ferrite, the inductance stays low, and high-frequency signals can pass through.

One practical application of such chokes is the suppression of radio noise picked up by long cables. Radio waves produce the same electromotive force in all bundled conductors, so the resulting currents are common-mode and can be blocked by this device.

Transformers

A *transformer* is an inductor with multiple windings. The simplest design resembles the common-mode choke shown in Figure 8-12. However, unlike a choke, a transformer is designed to transfer energy rather than dissipate it as heat.

In a transformer, an alternating current in the primary winding produces changing magnetization of the core. This changing field, in turn, induces an alternating electromotive force in the secondary winding. If the driving signal is a sinusoid and the coupling between the conductors and the core is sufficient, the induced voltage is proportional to the input signal multiplied by the turn ratio of the coils. Fewer turns in the secondary coil result in a reduced voltage; more turns produce a higher amplitude. There's no power gain in a transformer, so if the voltage increases, the maximum current that can be delivered to the load decreases (and vice versa).

Real-world transformers share the same limitations as inductors. Core saturation is especially noteworthy: To deliver the same amount of power to the secondary while operating at a lower frequency, the current in the primary must ramp up to a higher value during every cycle. Not only does this call for bulkier windings, but the core needs to be upgraded too, resulting in a transformer that weighs more and takes up more space.

In earlier days, many AC adapters used massive, iron-core transformers made for line frequencies of 50 or 60 Hz. A phone charger made in the 1980s would have been a real toe-smasher. Luckily, we figured out how to electronically switch much smaller transformers at frequencies measured in the tens of kilohertz. You see far fewer transformers now too: When galvanic isolation isn't necessary, voltage conversions are accomplished with clever combinations of inductors and capacitors (see Chapter 16).

Figure 8-13 shows the symbols for several common transformer architectures.

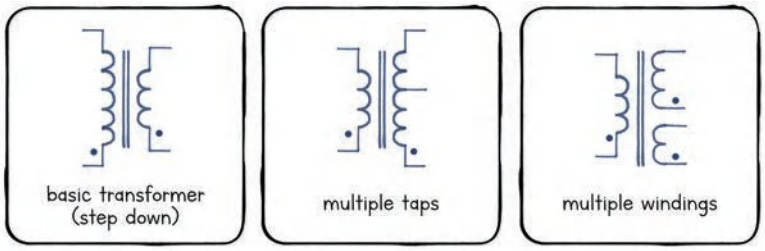


Figure 8-13: Symbols for various transformer architectures

The first panel shows the basic two-winding design. The dots indicate the orientation of the windings, telling us that a top-to-bottom current in the primary should induce a similarly oriented current in the secondary. The center panel shows a variant with a center tap on one of the windings, allowing a single transformer to produce multiple voltages. The design on the right includes three separate windings. This is fairly similar to multiple taps, but the design provides galvanic isolation between each winding, which could be beneficial in some applications.

That's a wrap for passives. Next on our list are the devices that turn electricity into motion and sound.

9

BASIC ELECTROMECHANICAL DEVICES



In this chapter, we'll take a look at a sampling of *electromechanical devices*—components that translate electronic signals into motion and vice versa. These devices are instrumental for tasks such as automatically switching other circuits on and off, playing and capturing sound, and spinning the wheels of an RC car.

As with passive components, the electromechanical category is not especially well defined. It is generally understood to include switches and motors, but knob-operated potentiometers are more commonly classified as passives (see Chapter 8). It's best not to overthink it. The categories matter only insofar as they help you pass exams and browse online catalogs.

As a matter of practicality, we'll focus on common electromechanical components that are fundamental to a wide range of electronic devices. More specialized sensors or actuators also exist, but they generally operate on the same principles. As for complex "computerized" electromechanical sensors, such as accelerometers and gyroscopes, we'll consider them in Chapter 17.

Connectors

Connectors need little introduction. They're simple mechanical devices that provide temporary coupling between circuits. Compared with permanently soldered wires, connectors make electronics easier to transport, reconfigure, and repair.

A taxonomy of every connector on the market would fill hundreds of pages—there's an endless variety. Some connectors are designed to be compact, others to be rugged, and still others to safely handle high currents and voltages. A picture is worth

a thousand words, so for an engaging exploration of common designs, I recommend the highly visual book *Open Circuits* (No Starch Press, 2022). Another way to brush up on connector types is to browse the catalogs of the electronics retailers discussed in the appendix at the end of this book.

Despite the wide range of options, we tend to stick to the classics. The most common small connector in use today is probably USB-C; it supports flexible power delivery and high-speed data transfers between computers and peripherals. Another common type is the tubular barrel plug, still widely used to supply fixed DC voltages to low-power household appliances. Many readers will also recognize the various sizes of audio jacks for wired headphones, speakers, and microphones.

Inside electronic devices, we frequently encounter so-called *headers*—rows of metal pins on a 1.27 mm or 2.54 mm grid that mate with similarly shaped plastic receptacles. It's also common to see *wire-to-board* connectors that clip or crimp onto cables. Finally, compact electronics also use delicate, amber-colored *flexible printed circuit (FPC)* strips providing internal connections to displays, touchscreens, keypads, and so forth.

On the workbench, you'll find BNC connectors for signal generators and oscilloscope probes, along with single-conductor *banana plugs* used with multimeters and lab power supplies.

The side of a connector with exposed prongs or that slides into a receptacle is commonly called *male*; the other side is termed *female*. The words *plug* and *socket* refer to cable-side and device-side connections, respectively. Finally, *jack* is somewhat ambiguous, with different meanings in different parts of the world. Figure 9-1 shows several ways of representing sockets and plugs on circuit schematics.

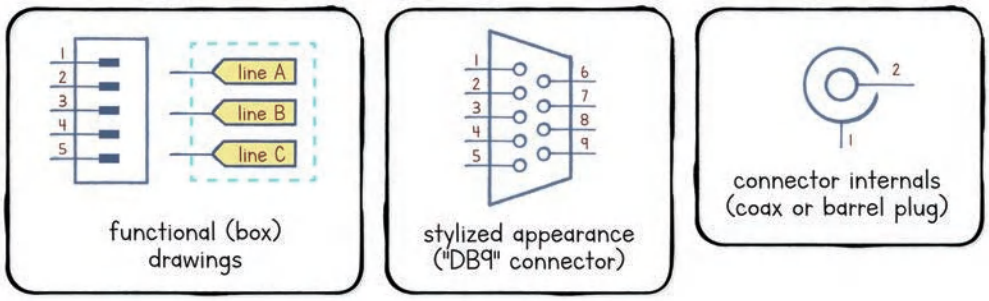


Figure 9-1: Some of the ways to draw connectors

The styles range from simple boxes to more elaborate visual metaphors. You can choose any approach you like, but make sure your drawings are unambiguous: Always include pin numbers or labels, and make the connector appear as a distinct, logical whole.

Switches

Switches should be familiar to any reader who has ever turned on a light or used a vending machine. These devices open or close electrical connections in response to a mechanical action, such as flipping a rocker, pushing a button, or turning a knob.

As with connectors, we could spend hours discussing the many form factors and styles of switches. There are miniature tactile (dome) switches that can be mounted directly onto a printed circuit board to function as reset buttons, simple keypads, and so on. There are also elevator-style illuminated pushbuttons, slide switches, rocker switches, paddle switches, and many others. Instead of rattling off all the options, let's focus on what can't be grokked from catalog photos alone.

Electrical Ratings and Architecture

Every switch is rated for a particular current and voltage. If the current limit is significantly exceeded for more than a split second, the contacts may overheat and possibly weld together. Exceeding the voltage limit usually has less dramatic consequences, but excessive sparking—which can occur briefly even in low-voltage circuits when disconnecting a large inductive load—can cause premature wear. At sufficiently high voltages, sustained electrical arcing could also become a concern.

In practice, miniature tactile switches may be rated for as little as 20 to 50 mA at 12 V; small slide switches are commonly designed to handle at least 300 mA; and larger panel-mounted rocker switches can usually carry between 5 and 20 A at 250 V. When the component's dimensions allow, these ratings are typically stamped on the side of the switch.

Current and voltage limits aside, switches are also distinguished by their mechanical action. A switch is said to have a *momentary* action if it stays in position only while an external force is applied. In contrast, *latching* or *sustained* action means that the mechanism stays in position until toggled back. A doorbell is a typical example of a momentary switch, while a wall-mounted light switch is a common sustained type. Most momentary switches are normally open (NO), but normally closed (NC) variants are also available from some manufacturers for specialized applications.

Another distinguishing feature of a switch is the number of *poles*. Each pole is a separate electrical circuit operated by a common mechanical assembly—essentially multiple units ganged together. Most switches have a single pole (SP), although double-pole (DP) mechanisms are sometimes used in outdoor lighting, motor control, and similar applications. Higher pole counts (3P, 4P, 5P, and so on) are fairly rare, partly because few applications require toggling five completely separate circuits at once.

The number of poles is often cited together with the number of *throws*. A throw refers to the set of electrical paths through which the mechanism cycles. A single-throw (ST) switch still has two positions—since a one-position switch would be useless—but it provides a single on/off path between two terminals, possibly repeated for each switch unit in a multi-pole assembly. A double-throw (DT) mechanism can toggle each pole between two electrical paths; for example, in a single-pole double-throw switch with three terminals labeled A, B, and C, the connection might alternate between A-B and A-C.

Switches with three or more throws—and thus three or more positions—are uncommon, but they crop up in some household appliances. For example, an old-fashioned fan may use a 3T switch to select among three motor speeds (low, medium, and high). Figure 9-2 shows some of the most common schematic symbols for switches.

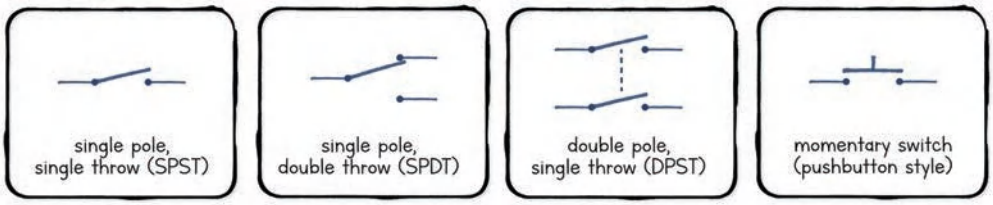


Figure 9-2: Schematic symbols for switches

In the third panel (DPST), note the use of a dashed line to indicate a mechanical linkage between the two electrically separate paths in a single package.

Switch Bounce

If you pry open a typical switch, you'll probably find that its internals are quite complex. Inside, there might be a small spring-loaded roller that travels over a tiny hump or an intricately shaped metal leaf that "pops" when pressed firmly enough. The purpose of these mechanisms is to keep the switch firmly in place until sufficient force is applied, at which point it snaps immediately into the new position. This nonlinear behavior provides satisfying tactile feedback, reduces sensitivity to vibration and other accidental inputs, and prevents the switch from resting in any state other than "fully off" or "fully on."

The flip side of this approach is that once the mechanism gives in, things happen quickly: The contacts are pulled apart or slammed together at a considerable speed. Because metal is elastic, this motion produces some vibration that manifests as electrical glitches lasting from tens of microseconds to a few milliseconds (Figure 9-3). The effect is known as *switch bounce*.

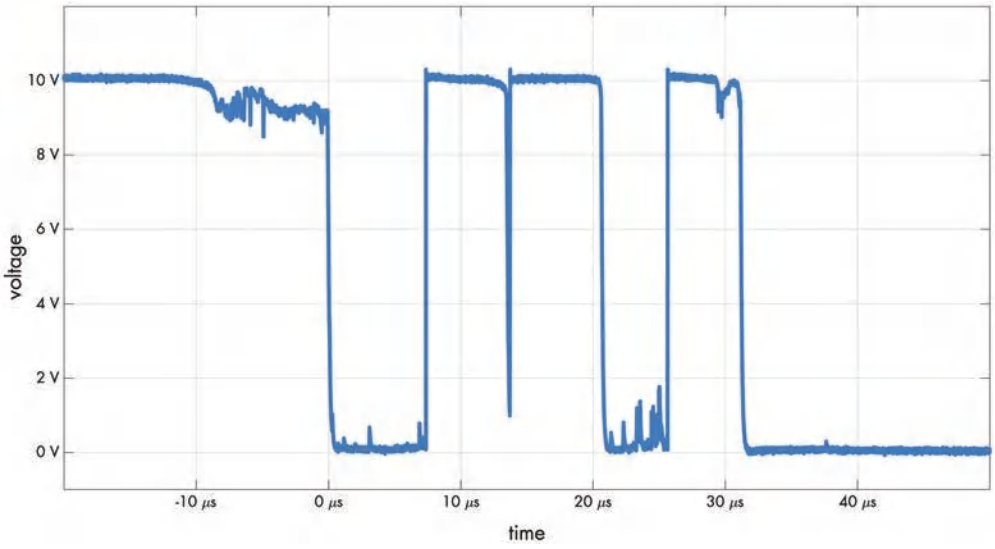


Figure 9-3: A fairly typical pattern of switch bounce when a switch goes from on to off

The glitches are imperceptible to humans and inconsequential in most analog applications, but they last an eternity from the perspective of a digital chip. If the bounce pattern isn't taken into account, embedded computers can misread a single input as a series of independent key presses, producing confusing results and sending novice circuit designers on a wild-goose chase.

In earlier days, we used to incorporate electronic filters to suppress high-frequency switch noise before it reached a high-speed chip. Nowadays, the problem is more commonly handled in software. In most cases, requiring around 10 milliseconds to pass between successive inputs is enough to filter out switch bounce without adding any human-perceptible lag.

Switch-Based Sensors

Switches are a common building block for simple sensors. For example, reed switches, used in home security systems, contain a permanent magnet and are activated by bringing another magnet nearby. Similarly, mechanical thermostats use a bimetallic strip that deflects in response to changes in temperature, bringing contacts together or moving them apart.

Another interesting type of sensor in this category is the *incremental rotary encoder*. As the name implies, rotary encoders convert rotation into electrical signals; their applications range from car radio volume knobs to position sensors in industrial machinery. In such uses, they serve as longer-lasting alternatives to delicate potentiometers. The downside is that encoders output pulses, not analog readings, so the signal needs to be interpreted by an embedded computer to convert the pulses into angular position.

Rotary encoders can be constructed in a number of ways, but the simplest version consists of two stationary pins that glide across a rotating, rosette-shaped conductive disk. Because the pins are slightly offset, the order in which they contact the disk can be used to infer the direction of rotation (Figure 9-4). Meanwhile, the frequency of the generated pulses represents the rotation speed.

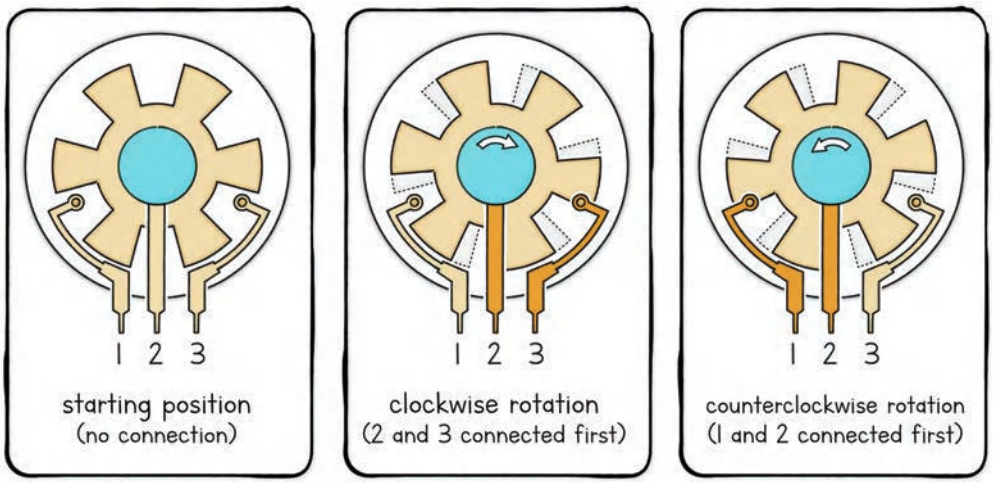


Figure 9-4: A conceptual illustration of a simple rotary encoder

There's no well-established schematic symbol for this class of devices. Components that don't have dedicated symbols are often illustrated as blank boxes with a descriptive label on the side. That said, some authors may opt to include additional visual cues of their choice.

Simple Magnetic Actuators

In Chapter 6, we noted that the flow of electricity through a coil of wire can produce a coherent magnetic field. This field can attract or repel permanent magnets, and it can also more weakly attract ferromagnetic materials that don't produce coherent magnetic fields of their own but contain microscopic, reorientable magnetic domains in the crystal lattice. The materials that exhibit this property include iron, cobalt, nickel, and some grades of steel.

A coil of wire used to attract metals is known as an *electromagnet*. An electromagnet can be used on its own—for example, to sort scrap metal—but it's more common to combine it with a captive plunger that's pushed or pulled by the field. This contraption, usually referred to as a *solenoid*, can be found in some electronically controlled door locks, hydraulic valves, and vintage pinball machines.

The coil of a typical electromagnet consists of very thin wire wrapped around the electromagnet hundreds or thousands of times. This construction serves two purposes. First, it maximizes the strength of the resulting magnetic field. Second, the

length of the conductor gives the coil substantial resistance, limiting the current to a safe maximum at the rated supply voltage. Most electromagnets can be operated at higher voltages to deliver more punch, but the duty cycle must be short to allow the coils to cool down.

Solenoids are a niche product, but they share their operating principle with two additional common electronic components: speakers and relays.

Speakers

A speaker is a solenoid with a twist: Instead of a fixed coil that actuates a plunger, there's a permanent magnet affixed to the frame and an electromagnet coil mounted on a lightweight diaphragm made of paper or plastic. Because of the great difference in mass, the coil doesn't move the magnet. Instead, it moves itself back and forth in response to the applied current. This motion also moves the diaphragm in a similar fashion. Because the diaphragm has a fairly large surface area, it scoops up a decent volume of air and produces audible sound waves (Figure 9-5).

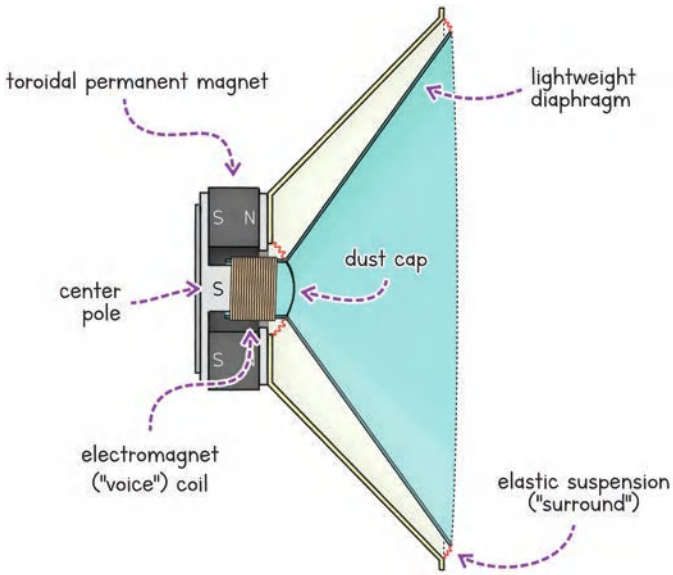


Figure 9-5: The internals of a simple speaker

Like solenoids, speakers are designed with a nominal coil resistance, usually 8 Ω, although 4 Ω and 16 Ω coils aren't unheard of. Another key parameter is the maximum continuous power rating. For small speakers, this usually falls between 250 mW and 2 W; home audio components may be rated for 50 W or more. As expected, exceeding the rating risks overheating the coil.

Acoustically, speakers are complicated beasts. Size matters: Small diaphragms have less inertia, and small coils have less inductance, so compact units reproduce high-frequency sounds more effectively. On the flip side, tiny speakers can't move a whole lot of air, so they can't deliver good low-frequency rumble. This problem can be mitigated to some extent by electronically boosting some audio frequencies and attenuating others (see Chapter 12). Still, the hi-fi solution is to spend a bit more money and have multiple speakers of different designs working together.

To get a sense of how speakers work and how well they handle different frequencies, you can try hooking up a small speaker to a function generator. Crank up the voltage to 5 to 10 V, and then try outputting sine waves at different frequencies;

around 1 kHz is a sensible starting point. Note how the apparent loudness changes with frequency. Some of this reflects the nature of human hearing, but you will also notice marked differences between speakers of different sizes. Next, check out square waves. They should sound much louder due to their fast-rising and fast-falling edges, but the sound will also have a much harsher, tinnier quality.

You can also connect the speaker to an oscilloscope and see what happens if you speak into it or push the diaphragm with your finger. You should see AC voltages across the terminals. We'll discuss why this happens shortly.

Relays

A *relay* is a solenoid-operated switch. Like normal switches, relays can have a varying number of poles and throws. Most relays feature momentary switching action, though several varieties of latching mechanisms are also available. The devices range from miniature models used in precision electronic instruments to fist-sized *contactors* found in high-voltage appliances, such as whole-house generators and air conditioners. Figure 9-6 shows some common schematic symbols for relays and speakers.

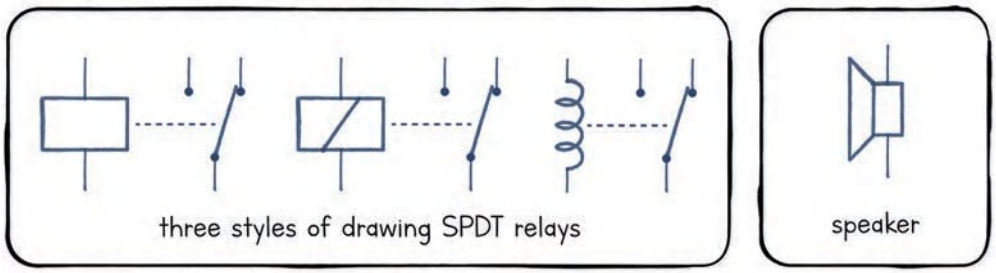


Figure 9-6: Symbols for relays and speakers

If you have a miniature non-latching relay and a capacitor in the vicinity of 100 µF, there's an interesting experiment you can conduct (Figure 9-7).

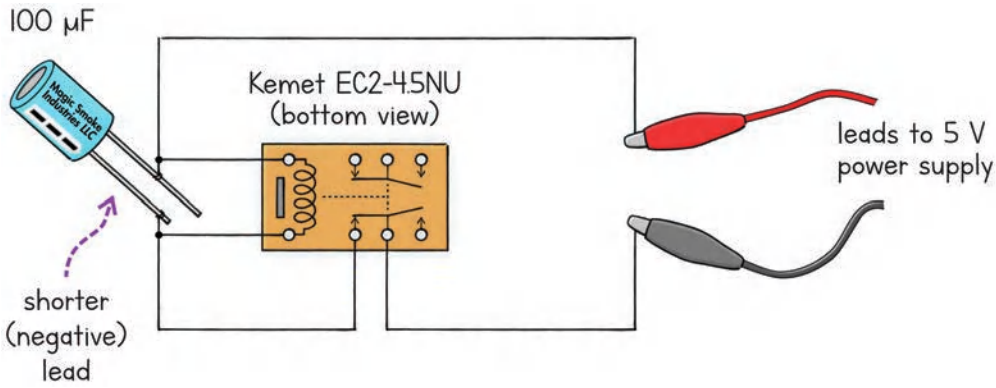


Figure 9-7: An experiment with a miniature relay (bottom view)

The circuit uses a DPDT relay made by Kemet and labeled EC2-4.5NU. Other relays might have a different pinout, but most of them should work fine if the necessary wiring adjustments are made.

Once the circuit is connected to a 4 to 5 V supply at 500 mA, the relay should start clicking at a frequency of several hertz. See if you can figure out why—connecting an oscilloscope to the leads of the capacitor may help.

If you're stumped, note that when first powered on, there's a connection between the negative supply, the capacitor, and the bottom relay control pin. The capacitor quickly charges, and the relay is energized. After a brief moment for the mechanism to move, this interrupts the connection and opens the circuit. The relay doesn't turn off immediately because of the energy stored in the capacitor. Only after the capacitor sufficiently discharges does the electromagnet release, and the cycle restarts. We've constructed a very simple electromechanical oscillator.

Relays are among the oldest electronic components that can be harnessed for digital computation. They've been used to construct some of the earliest computers and telephony systems in the world. Still, the devices suffer from three significant limitations. First, their switching speed is limited by the time it takes for a mass to travel from one position to another. Second, the components are fairly power-hungry—even the most efficient miniature relays require around 30 mA just to hold in place. Above all, relays are prone to mechanical wear and fail after far fewer cycles than a transistor would. They keep working for decades in devices such as wall thermostats, where they're actuated only several times a day, but if we upped this to several times per second, the relay would probably be toast within a month.

Motors

Whereas solenoids produce single nudges, motors are designed to convert electric energy into continuous motion. This can be achieved in a couple of ways, but the most practical approach uses an arrangement of electromagnets to spin a metal shaft. The rotating shaft can then be coupled to gears, timing belts, hydraulic pumps, rack-and-pinion mechanisms, and much more.

The usual design of a small DC motor features a combination of permanent magnets and electromagnet coils. If the electromagnets are built into the rotor, as in Figure 9-8, the permanent magnets are integrated with the motor frame (the *stator*). It's also possible to reverse this arrangement.

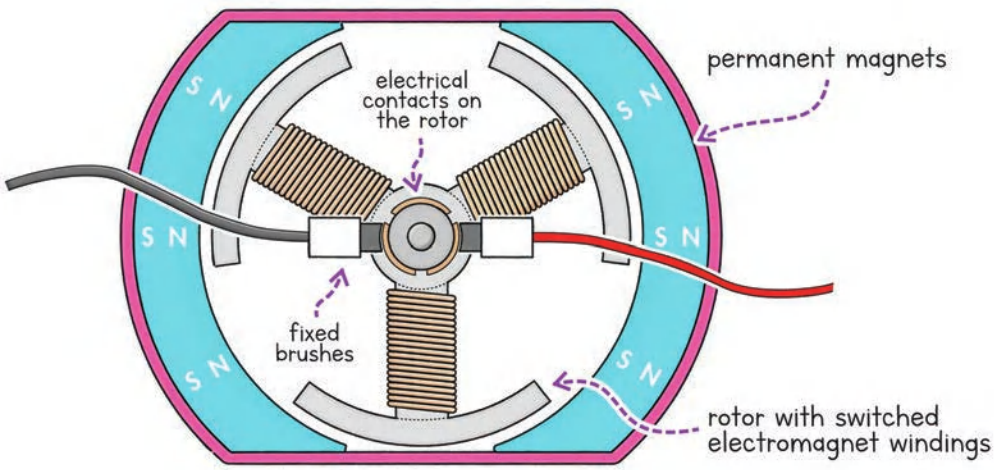


Figure 9-8: The structure of a brushed DC motor

Regardless of the layout, the idea is to energize the electromagnets in a way that pulls the rotor toward one of the stationary magnets. Then, just as the poles start to align, we break the circuit and send the current to another coil, making the rotor

want to move to another configuration that requires turning farther along. We keep doing this to achieve the electromagnetic version of a dog forever chasing its own tail.

Brushed DC Motors

In a typical *brushed motor* like the one shown in Figure 9-8, the switching of coils on the rotor is accomplished by a mechanical device known as a *commutator*. The commutator consists of a circular pattern of copper contacts mounted on the rotor and two smooth, conductive brushes affixed to the motor frame. This energizes different coils depending on the angular position of the rotor assembly.

High-speed DC motors exhibit an interesting property: They don't use more power than necessary. From Chapter 6, recall that if the current flowing through an inductor (a coil) is changing, some of the energy is soaked up by a magnetic field. This manifests as a momentary pushback voltage that's proportional to the rate of change in the supplied current.

Importantly, the effect works both ways. If the magnetic field is changing for some external reason—for example, because the coil is spinning near a permanent magnet—electrons in the coil are pushed around and a voltage develops. This is what we observed with a manually actuated speaker, and it's what happens in a motor. The faster the rotor turns, the greater the rate of change of the magnetic field acting on the coils and thus the higher the pushback voltage. This phenomenon, known in motor design as *back electromotive force (back EMF)*, eventually reaches an equilibrium with the supply voltage. At that point, the rotor stops accelerating, and the device admits only as much current as needed to make up for friction and other energy losses. Yet, as soon as the rotor is connected to a load and starts spinning more slowly, back EMF is reduced, allowing more current to flow.

Because of this desirable property, there's some nuance to parsing the specs of brushed motors. Motors are advertised based on their RPM and torque, but you can never get the maximum of both. Without load, the motor spins fastest, but the available torque is zero—it can muster only enough energy to keep the rotor itself running at that speed. To get maximum torque, you must make back EMF disappear—and that happens only at stall. Between these extremes lies a roughly linear relationship among speed, torque, and current. For example, if you want to spin something at 80 percent of the motor's no-load RPM, you must settle for about 20 percent of the advertised stall torque. On the upside, you only need to supply roughly 20 percent of the stall current given in the motor spec.

Brushless DC Motors

Unlike their brushed counterparts, *brushless DC motors (BLDCs)* don't rely on mechanical commutation to switch current to the coils; they use electronic switching instead. Brushless designs are meant to overcome the disadvantages of brushed motors, such as relatively noisy operation and the tendency for brushes to wear. BLDCs can also be made lighter and more compact by making the control circuitry sense overload and kill power in a matter of milliseconds. In a brushed motor, the windings must be made bulky enough to survive stall currents indefinitely—or at least long enough for the operator to react or a fuse to blow.

Figure 9-9 shows some typical schematic symbols for motors. The left panel of the figure shows the usual symbol for a two-terminal motor. In a DC circuit, this motor would be brushed. There's no consistent symbol for BLDCs, but some common choices are shown on the right.

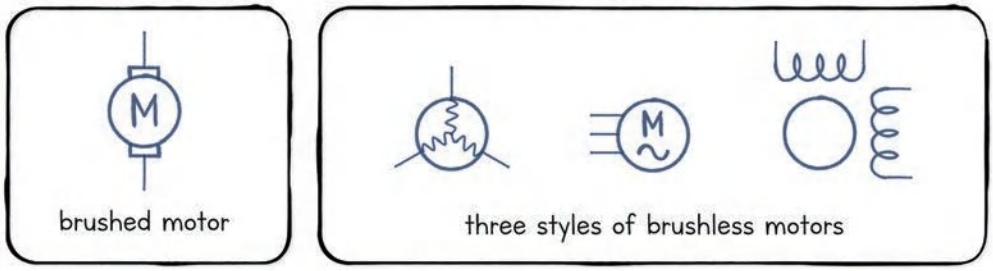


Figure 9-9: Common schematic symbols for motors

Constructing the control circuit for an electronically switched motor isn't trivial. Without knowing the position of the rotor, we might end up switching the coils too slowly or too quickly for the load. The former would result in jerky motion and excessive power consumption, and the latter would make the rotor lose steps and helplessly flail around.

Stepper motors were an early solution to this problem—or more precisely, an electronically simple solution to a related use case of actuating precision mechanisms at lower speeds. These motors have a high number of coils and magnet poles, producing relatively smooth motion without requiring perfectly timed coil switching. Further, they are meant to be operated at low RPM so that there is no need to adjust switching speed for loads that stay within the motor's spec.

Stepper motors usually run no faster than several hundred rotations per minute—about two orders of magnitude slower than a brushed motor of a comparable size. The benefits of back EMF largely disappear, so steppers are essentially constant-current devices, power-hungry even when not doing any useful work. Still, their simplicity, near silence, and precise angular positioning—due to the high number of coils—make them valuable in mains-powered equipment. For example, most hobby 3D printers use them for positioning.

A refinement of this concept is the modern brushless DC motor, which incorporates some form of position sensing. A reliable method uses a magnetic *Hall effect sensor* that detects small disturbances in the flow of current through a semiconductor substrate when a magnet passes nearby. A cheaper but more complex alternative is to monitor the current flowing through the motor coils; the amount of back EMF is proportional to rotor speed, so rudimentary coil timing corrections can be done by monitoring how successful the power supply is at pushing electrons through the device.

A BLDC motor has at least three terminals for the coils. Because these coils need to be energized in a specific, carefully timed sequence, an external digital controller is almost always necessary. However, as noted earlier, the motors themselves are lighter, quieter, and more durable than their brushed counterparts, so they find use in performance gear, ranging from high-end drones to premium battery-operated workshop tools.

Piezoelectric Crystals

Piezoelectric crystals are electronic components made from substances that change dimensions in response to an applied electric field. This might sound somewhat mystifying, so to understand this phenomenon, let's start by having another look at a more familiar component: the capacitor.

Consider a simple capacitor consisting of two metal plates and an inert insulator sandwiched in between. In Chapter 1, we noted that dissimilar charges attract, so in a charged capacitor, there must be an attractive force between the plates that's proportional to the applied voltage. This force is quite strong; it squishes the insulator layer, changing the dimensions of the device to some microscopic extent.

This relationship goes both ways. If we put a charged capacitor in a vise and apply external mechanical force, this brings the plates closer together, increasing capacitance (C). In Chapter 5, we established that capacitance is equal to charge (Q) divided by voltage (V); if C increases and Q remains the same, the voltage across the terminals of a mechanically squished capacitor must drop.

In real capacitors, this effect is usually negligible, but a special class of insulators—piezoelectrics—amplifies it through permanently aligned electric dipoles. These crystals, when squeezed, can produce sufficient voltage swings to give off sparks, so they're used as igniters in cigarette lighters and some gas appliances. In the same vein, although their dimensions change too little for general-purpose speakers, the crystals can generate high-intensity ultrasound or high-pitched beeps in smoke alarms. This is the operating principle of *piezoelectric transducers*.

Beyond fire-starting and noise-making, the most common use of these materials is as timing devices in modern electronics, from wall clocks to radios to computers. The resulting timing component is simply called a *crystal* or *quartz* (hence "quartz watch"). To explain the design, we can lean on the analogy of a playground swing. You'd be right to treat such analogies with suspicion, but in this instance, we're dealing with mechanical oscillations, so the comparison is apt.

A swing has a natural resonant frequency: If you give it a gentle push, it moves away a tiny bit and then comes back after a while. If you time the next push correctly, the amplitude of motion increases; if you time it incorrectly, it doesn't. The same is true for a piezoelectric crystal. We can set it in motion with electrical nudges, and if the frequency of the nudges matches the physics of the crystal, the amplitude of the oscillations will grow. It's possible to design a circuit that zeros in on maximum signal amplitude and thus naturally settles at the crystal's resonant frequency. This works because, as noted earlier, the phenomenon is bidirectional: Mechanical oscillations induced by small nudges eventually begin to manifest as measurable voltage swings.

Figure 9-10 shows the response of a crystal trimmed to a natural resonant frequency of 2 MHz. In the top portion of the plot, we can see that as the nudges from a signal generator approach the resonant frequency of the crystal, the number of electrons moving back and forth increases. This tells us that mechanical oscillations are increasing in amplitude, sympathetically pushing charges back and forth. At precisely 2 MHz, we notice a dip in impedance, meaning the admitted back-and-forth current is at its peak in relation to the constant-amplitude driving voltage.

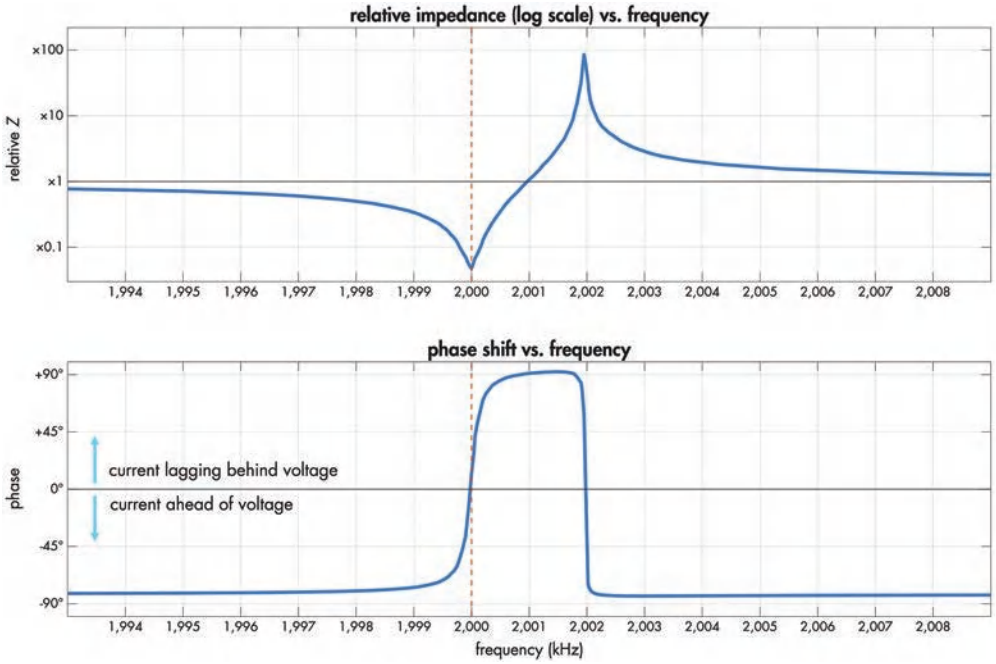


Figure 9-10: Real-world behavior of a 2 MHz crystal

In the same plot, we see that if the driving frequency is slightly higher than 2 MHz, the impedance briefly spikes. This is the anti-resonant frequency. The spike is most easily explained with the swing analogy: If you push too early—while the swing is still moving toward you—you're not helping at all. Instead, you're removing energy from the system and forcing the swing to stop.

One difference between a swing and a piezoelectric crystal is that in electronic circuits, we usually don't drive the crystal with discrete pulses. Instead, we use a continuous sine wave. At the resonant frequency, the driving signal and the motion of the crystal are perfectly in sync, so there's no phase shift between the input voltage and the resulting current (Figure 9-10, bottom half). At most other frequencies, the crystal isn't resonating, so we're looking at two pieces of metal with an insulator in between—a run-of-the-mill capacitor with its associated −90° phase lag. However, in the narrow range of frequencies—the "active braking" region between resonance and anti-resonance—the phase shift is flipped around. I'll spare you the math that explains it, since the phenomenon is relevant to just one circuit design: a particularly minimalistic oscillator we'll review in Chapter 23. If you're curious, though, exploring a discrete-time model should offer good insights.

In this day and age, it might seem odd that we're still using mechanical devices as timing sources. The reason is simple: They work exceptionally well. The specific resonant frequency of a crystal can be laser-trimmed during the manufacturing process with an accuracy measured in parts per million. Equally important, the frequency remains stable over time. Even better performance can be achieved by controlling for temperature, either with bundled sensors or with a miniature PCB-mounted "oven" housing the crystal.

Crystals are commonly available for frequencies between 2 and 60 MHz or so. A separate range of slow crystals, tuned to 32.768 kHz, serves applications such as low-power real-time clocks. Figure 9-11 shows symbols for piezoelectric elements and our next subject, microphones.

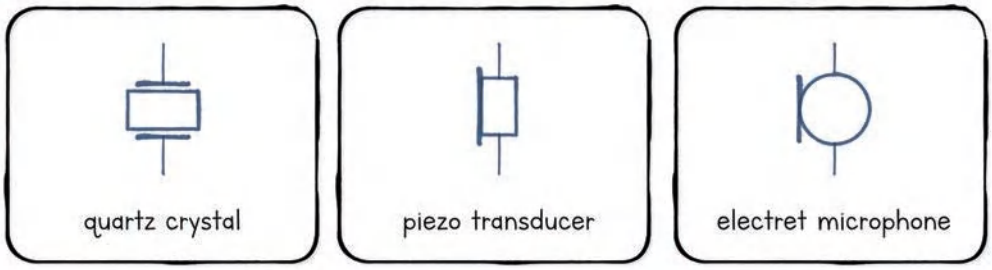


Figure 9-11: Symbols for piezoelectric elements and microphones

Apart from crystals, there's also a competing range of *micro-electromechanical system (MEMS)* devices that are manufactured by etching the resonant structure directly onto silicon, alongside the other components needed for an electronic oscillator. The advantage of this relatively new process is space savings, so MEMS oscillators can be found in smartphones and other highly miniaturized electronics where every square millimeter counts. Still, it's just a spin on the same theme, and when size isn't the overriding concern, piezoelectric crystals currently perform equally well or better at lower cost.

There's not much we can do with a crystal just yet, but as noted earlier, we'll revisit practical timing circuits in Chapter 23.

Microphones

In this chapter, we've discussed two ways of making noise: speakers and piezoelectric transducers. In principle, both can be used in reverse to capture sound waves, but they aren't particularly good at that. Conventional piezoelectric elements don't have sufficient movement range to capture low frequencies. A speaker performs better—the coil produces back EMF voltage if moved by air pressure across a wide range of frequencies—but the component is needlessly bulky and fragile for this use.

Instead, to record audio, we typically use a device called a *condenser* or *electret microphone*. The design consists of a permanently charged membrane—the electret—made of plastic in which an electrostatic charge is trapped during the manufacturing process. This membrane forms one plate of a capacitor; when it vibrates in response to sound, its motion relative to the other plate induces the movement of electrons. This resembles our earlier discussion of how squishing or stretching a ceramic capacitor produces swings in the voltage across its terminals.

In Chapter 2, we introduced field-effect transistors as voltage-controlled electronic valves in which the voltage applied to the gate moderates the flow of current between the two remaining terminals. In a condenser microphone, the second plate of the capacitor is connected to the FET gate. The induced microscopic currents change the gate voltage, so the current admitted through the transistor's main conduction channel changes accordingly.

As with piezoelectric crystals, the traditional design is facing some competition from the newfangled MEMS manufacturing process. MEMS microphones use active circuitry to detect the motion of a micromachined structure—typically a suspended membrane-shaped capacitor. As of this writing, MEMS microphones lack exceptional acoustic specs, but their small size and compatibility with industrial reflow soldering make them attractive for applications such as smartphones and hearing aids. For everyone else, electrets are still fine.

The electrical ratings of electret microphones are usually straightforward. The component can often tolerate a wide range of voltages across its terminals. For MEMS

models, the active circuitry requires a more precise supply regulation, but the electrical side remains uncomplicated. The more important consideration is acoustic performance. Every microphone has a specific spatial pickup pattern, which can be plotted by spinning a sound source around the device under test. Like speakers, microphones also have a frequency response curve that should ideally remain flat across the audible spectrum, from 20 Hz to 15 kHz. Electrets typically come close to this ideal, but for hi-fi applications, it's always wise to check.

So far, we have covered the easy stuff. However, the most important acoustic parameters are sensitivity and noise performance, and to make sense of these specs, we regrettably need to talk about decibels.

Deciphering Decibels

In electronics and some adjacent disciplines, we often compare magnitudes. Differences in magnitude can become large, so we need shorthand to avoid spelling out that a quantity has changed by a factor of, say, 1,000,000.

A simple solution is to use exponents: We can write 1,000,000 as 10^6, take the base of the multiplier (10) as a given, and then use a keyword to indicate that we're talking only about the exponent of the multiplier. For example, we could call 10^6 an increase of 6 "clerts." In the same vein, a decrease by a factor of 10,000 could henceforth be known as −4 clerts. This notation also simplifies some circuit math. For example, if we have a long cable that attenuates signals by x clerts and another by y clerts, the combined clert effect is a simple sum, because $10^x \cdot 10^y = 10^{x+y}$.

Alas, the clert is made-up; the *decibel (dB)* is real. More precisely, the original pseudo-unit is the *bel*. The bel works much like our clert: +1 bel is a tenfold increase, while −2 bels is a decrease by a factor of 100. So far so good. But there's a twist: The bel was originally designed specifically for measuring power. Recall from Chapter 3 that power sometimes has a quadratic relationship to applied voltage. Specifically, for purely resistive loads, we have Joule's law: $P = V^2 / R$.

Seeing this, someone decided that 1 bel should always represent a hypothetical tenfold increase in power, even when applied to another base unit and even without resistive loads present. This means that if you're talking about watts, +1 bel is an increase of 10×, but if you're talking about volts, you're supposed to interpret +1 bel as an increase of $\sqrt{10}\times$ (circa 3.1622×). This ensures that squaring the voltage later yields the correct power via Joule's law. For other units, where the relationship to power is less obvious than for volts, figuring out which kind of bel you're supposed to be using can be a major headache.

The weirdness doesn't end there. At some point, we decided that the bel is too big to use—but only slightly so. Instead of putting up with the occasional fraction or switching to another base for the exponent, we divided the bel into 10 steps, hence *deci*bel. In effect, we started raising 10 to fractional powers, producing irrational per-decibel multipliers. For power, 1 decibel is a multiplier of $10^{1/10} \approx 1.258925$; for voltage, it's $\sqrt{10}^{1/10} = 10^{1/20} \approx 1.122018$. In all cases, 0 dB means a multiplier of 1—signifying no change.

The original bel is gone; we now use decibels exclusively.

Quantifying Microphone Sensitivity and Noise

Knowing the meaning of the decibel isn't enough. When someone uses the term, we also need to know the base unit and the reference point they have in mind. In acoustics, decibels are used to measure sound level; 0 dB is the loudness of a 1 kHz acoustic

wave that's calibrated to exert 20 micropascals of pressure. This is roughly the threshold of human hearing.

From this 0 dB origin, we can derive several acoustic scales. One popular choice measures the absolute sound pressure level with no regard to frequency, often denoted as dB SPL. Other scales can be weighted to mimic human hearing, typically peaking at 3 kHz and then tapering sharply below 200 Hz and above 10 kHz. The hearing-weighted loudness scale is typically denoted as dBA.

In a microphone spec sheet, the sensitivity might be given as −45 dB. This isn't an acoustic decibel: Here, the base unit is volts (dBV). The zero point on the scale is a theoretical microphone that would output 1 V_{RMS} in response to a reference acoustic wave. If your head is spinning already, it gets worse: The reference sound isn't 0 dBA but a 94 dB squeal at 1 kHz, roughly the loudness of a gas-powered lawnmower. If a microphone has a sensitivity of −45 dBV, it means the lawnmower-induced swing is 45 (voltage) decibels lower: $1\ V_{RMS} \cdot 10^{-45/20} \approx 6\ mV_{RMS}$.

Needless to say, you don't need to memorize all this detail. I'm just giving you a quick overview in case you need to figure out how the numbers in the spec translate into the voltages on the oscilloscope screen.

Sensitivity aside, another important characteristic of a microphone is its noise floor. There are two ways to measure it, but the more common approach is the signal-to-noise ratio (SNR). If the SNR is given as 60 dB, it means there's 60 decibels of headroom between the noise floor and the aforementioned 94 dB lawnmower reference point. In more logical terms, this means that the internal "hiss" contributed by the electronics of the microphone is equivalent to a 94 dB − 60 dB = 34 dBA source of static noise placed in the vicinity of an ideal microphone. This is known as *equivalent input noise (EIN)* and is a more reasonable, if less common, way to report microphone performance. A positive EIN means the hiss is louder than the threshold of human hearing; 34 dBA is enough to drown out whispering or nature sounds.

An average electret microphone might have −45 dBV sensitivity and an SNR of 60 dBA. Spending $2 more will get you a top-of-the-line module with −25 dBV sensitivity and an SNR of 80 dBA. For low-fidelity applications, such as gaming headsets, this doesn't matter. For everything else, a good microphone makes a dramatic difference. If you're curious, you can listen to a quick demo at the following URL: *http://lcamtuf.coredump.cx/slc/mic-compare*.

NOTE *The astute reader may have picked up that I'm not a fan of decibels. Although exponent notation is useful, this particular flavor of it is a source of needless pain. Suffixes help, but their landscape is chaotic too. In communications, for example, dBm references 1 milliwatt, but dBμ denotes 1 microvolt; in other disciplines, the same acronyms may mean something else. We can't escape decibels entirely, but apart from component specs, we won't be leaning on them throughout the book.*

Circuit Protection Components

Circuit protection devices (Figure 9-12) are electromechanical only in the loosest sense: They're meant to "pop" before a more expensive portion of the circuit catches fire. Still, we need a place to discuss them briefly. If you squint your eyes, the blowing of a fuse can be seen as an act of electromechanical sacrifice.

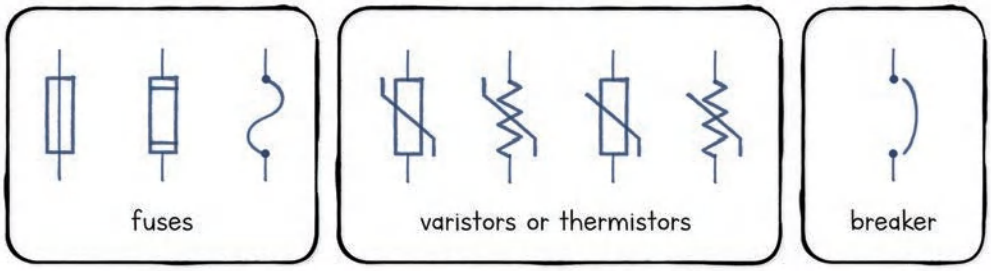


Figure 9-12: Symbols for circuit protection devices

Fundamentally, a *fuse* is a current-protection device. It serves two functions. First, it shields equipment from transient overloads (for example, an unexpected motor stall). Equally important, it protects surroundings from electrical malfunctions caused by internal short circuits. An unchecked short could ignite flammable plastics if the device overheats.

Basic fuses consist of a strip of (slightly) resistive material and essentially exploit the squared term in one form of Joule's law: $P = I^2R$. They generate manageable heat up to a certain current (I), but the material melts quickly—interrupting the circuit—if that threshold is exceeded.

A more user-friendly variant of this approach is the *breaker*, a latching switch that releases automatically if the current exceeds a certain threshold. Small pushbutton breakers, found on some appliances and power strips, are usually thermal and work like thermostats. Larger breakers use a solenoid that pulls a linkage if the current flowing through the breaker is higher than it's supposed to be.

Simple fuses are necessarily somewhat approximate. They trip quickly if the current is completely out of whack, but they might tolerate as much as 150 percent of their rated current for minutes or even hours. This can be a curse or a blessing. Some fuses are deliberately designed to have a higher thermal mass to withstand current surges when, for example, a large motor is spooling up. Reputable manufacturers provide detailed response curves for their products, so you can pick the right fuse for your needs.

In addition to traditional fuses and breakers, we have *resettable fuses*, also known as *polyfuses* or (more generically) *thermistors*. Once again, these devices are designed to be heated by the flow of current through a resistive material. A *positive temperature coefficient (PTC)* thermistor consists of conductive particles embedded in a nonconductive substrate that expands with temperature and eventually pulls the conductive particles apart. This typically reduces—but does not interrupt—current flow until the fuse is allowed to cool.

The last circuit protection devices on our list are *metal-oxide varistors (MOVs)*. Unlike fuses, they're meant to protect the device not against current but against voltage spikes that may follow lightning strikes, downed power lines, and so on. MOVs are made of ceramic materials with a carefully controlled breakdown voltage. The components usually straddle mains supply lines and are supposed to act as a crowbar if the voltage gets too high. If tripped, they tend to explode rather spectacularly. The hope is just that the MOV's fiery death gives enough time for the fault to clear or a standard fuse to take notice and trip.

Leaving passives and electromechanical devices in the rearview mirror, we're now ready to enter the kingdom of semiconducting devices.

10

DISCRETE SEMICONDUCTORS



In Part I of the book, we outlined the physics of semiconductor junctions inside field-effect transistors and diodes. At the time, we spoke in generalities. By now, we've covered enough electronic theory that, in this chapter, we can take a more systematic look at how these components respond to currents and voltages. These effects are what make modern electronics possible, from your trusty Juicero to every Bluetooth-enabled cat litter box.

For the time being, we'll still focus on the behavior of the individual, discrete electronic components. Once we wrap up Part II of the book, we'll start arranging these building blocks into increasingly complex designs. At that point, we'll also properly introduce integrated circuits, which combine a variety of miniaturized components inside hermetically sealed packages.

But first things first. Let's start with the diode.

Conventional Diodes

In Chapter 2, we noted that pure silicon is a poor conductor of electricity because it lacks long-lived, mobile charge carriers. In this material, some conduction is still possible through the short-lived thermal excitation of valence electrons, but the effect is very modest. A typical PCB trace made of silicon instead of copper would have a resistance measured in the high megaohms.

The material's conductivity can be improved by adding dopants. Certain dopants produce long-lived electrons that occupy higher energy levels in the crystal lattice,

creating what's called an *n-type semiconductor*. Other additives create a substrate with an excess of valence band vacancies, allowing lower-energy electrons to slither around more easily.

When an n-type material is brought into contact with a p-type semiconductor, a curious phenomenon takes place. Higher-energy electrons from the n-side randomly diffuse to the p-side and then promptly fall into the abundant lower-energy holes. (For a visual, refer back to Figure 2-2.) This process produces an internal electric field with a positive charge on the n-side and a negative charge on the p-side. The field jettisons any majority charge carriers that wander into the region back to where they came from, resulting in a thin, poorly conductive depletion region at the boundary where the materials meet.

As noted in Chapter 2, such a p–n junction responds to applied voltages in an asymmetric way. In a condition known as *reverse bias*, additional electrons are taken from the n-side and moved to the p-side, which only widens the depletion region and keeps the junction nonconductive. Under *forward bias*, electrons are removed from the p-side and added to the n-side, offsetting the internal electric field and eventually allowing the majority charge carriers to cross. In the case of silicon, the junction becomes conductive once the forward voltage reaches about 600 mV.

A p–n junction—or diode—can be thought of as a rudimentary voltage-controlled gate. When the component is forward-biased, but the voltage remains below a certain threshold, the diode acts like a very high resistance, typically in excess of 100 kΩ, so only a negligible, microscopic current can flow. As the voltage rises, the current increases exponentially and crosses into the milliamp range near the aforementioned threshold voltage. This growth doesn't continue indefinitely, because the substrate's resistance begins to dominate and the V-I curve becomes more linear.

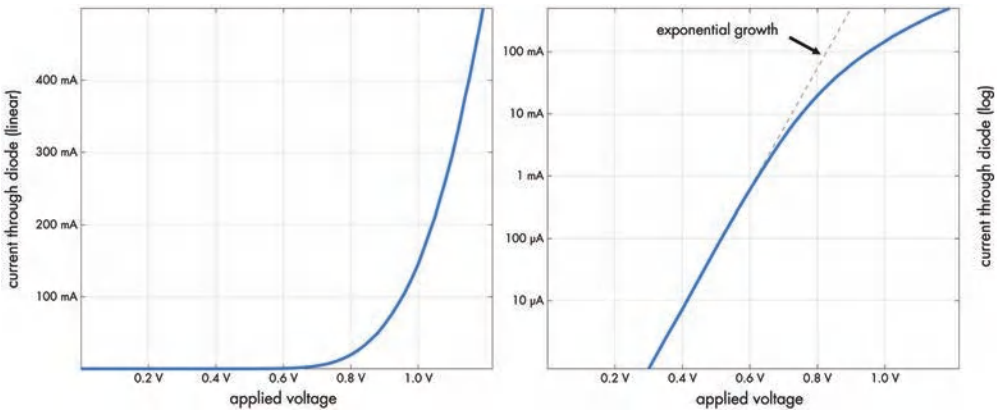


Figure 10-1: Forward-bias voltage versus current for a small, general-purpose diode (1N4148), plotted on a linear scale (left) and on a log scale for current (right)

In Figure 10-1, the plot on the left shows the V-I curve on a linear scale. The version on the right uses a logarithmic y-axis, so every doubling of the current is represented by the same vertical distance. In this view, exponential growth should produce a straight line. Yet, for the tested diode, the line bends noticeably to the right past around 10 mA due to substrate resistance.

Whether the current is limited by the dynamics of the p–n junction or by the substrate's properties, the effect is resistive, and the diode dissipates energy as heat. In keeping with the fundamental power equation discussed in Chapter 3, the rate of heat generation is proportional to the applied electromotive force multiplied by the resulting current: $P = IV$. Looking at the upper end of the curve in Figure 10-1, we

see that when the voltage across the terminals reaches 1.2 V, the current flowing through the diode peaks at about 500 mA. At that point, the component is dissipating $P = 500\ \text{mA} \cdot 1.2\ \text{V} \approx 600\ \text{mW}$.

Like a resistor, every diode has a power limit. If the limit is exceeded, the device might release the "magic smoke." In semiconductors specifically, thermal stresses are concentrated in the junction region, so the device can get cooked and become permanently conductive even without any outward signs of thermal damage.

The most rudimentary application of a diode is as a *rectifier* that converts alternating current to direct current. Consider the setup shown in the top half of Figure 10-2—a capacitor driven by a sinusoidal waveform. In this configuration, the capacitor is constantly charged and discharged in sync with the driving signal, so the circuit accomplishes nothing useful.

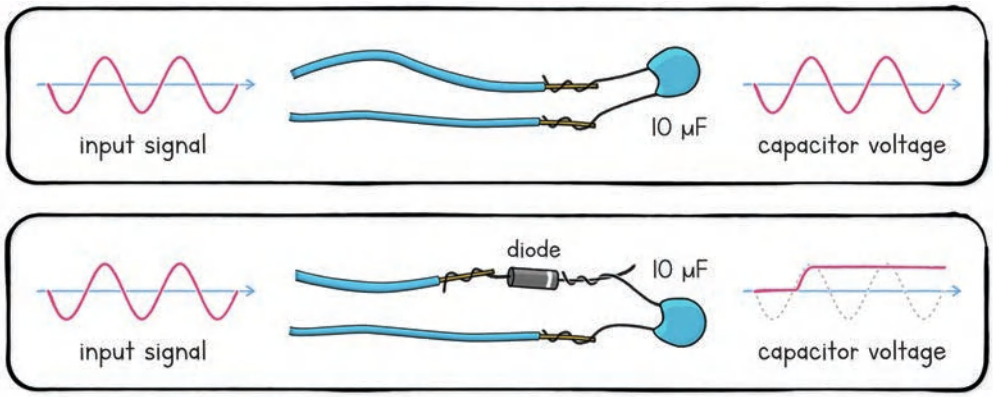


Figure 10-2: Charging a capacitor with an AC signal, with and without a diode

Next, let's modify the circuit by inserting a series diode that conducts well only if the input voltage is about 0.6 V higher than the capacitor's current charge state. During the first positive cycle of the sinusoid, the capacitor charges to V_{peak} − 0.6 V. After that, the input voltage starts to drop, which reverse-biases the diode and prevents the capacitor from discharging. In effect, we have a way to rectify AC into DC, with losses limited mainly to the capacitor's gradual self-discharge and the minuscule leakage current of the reverse-biased diode.

If you have a µF-range capacitor, a diode, a signal generator, and an oscilloscope on hand, I encourage you to give this arrangement of components a try.

Reverse-Bias Operation

Earlier we asserted that a reverse-biased diode is largely nonconductive. That is correct, but only up to a point. If the reverse voltage becomes high enough, the electric field across the terminals grows so strong that charge carriers barrel into the depletion zone with enough energy to knock other electrons into the conjunction band before being ejected. The newly freed charge carriers are also accelerated by the field, creating an avalanche-like effect. At that stage, the junction becomes conductive again.

Most diodes are designed to keep the reverse breakdown voltage well outside the device's intended operating range, with typical limits for discrete components falling between −100 and −1,000 volts. Once the threshold is exceeded, however, a reverse-biased diode starts conducting just fine. Figure 10-3 shows the effect.

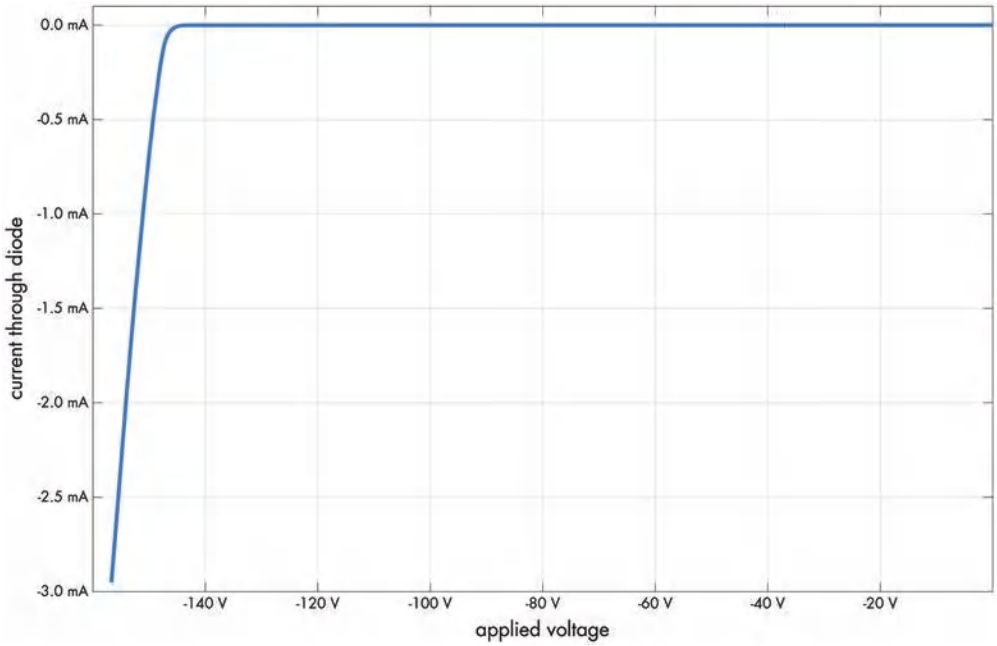


Figure 10-3: Reverse-bias voltage versus current for a 1N4148 diode. Currents are labeled negative for consistency with the forward-bias case shown in Figure 10-1.

As before, a diode operating in this regime dissipates heat equal to $P = IV$. In Figure 10-3, the applied voltage reaches nearly −160 V, so a current of only about 3 mA is enough to approach the component's ballpark power limit of 500 mW. This is why the plot stops at that point.

Other Diode Behaviors

Although a diode is often thought of as an on-off switch, it's good to remember that microamp-range currents can flow through the junction even in its "off" state. In most cases, the effect is negligible, but it can occasionally cause surprises. For example, if you tried the rectification experiment from Figure 10-2, you might have noticed the capacitor charging to a voltage higher than advertised. This happens because a common diode, such as the 1N4814, will let through about 1 μA when subjected to as little as 300 mV. If no load is connected to the capacitor, these tiny leakage currents accumulate over time.

Another important property of diodes is that when they're in a nonconducting state, their internals resemble a capacitor: They contain two conductive regions separated by a very thin insulating layer. For small diodes, the resulting parasitic capacitance is usually only a few picofarads, but that can be enough for some high-speed signals to leak through. The problem is analogous to the parasitic capacitance of the nonideal resistors discussed in Chapter 8.

For macroscopic conduction to occur in a diode, the depletion region at the p–n junction needs to be disrupted by an externally applied field. Once the input voltage drops, the layer takes a brief moment to re-form. During this interval, known as the *reverse recovery time*, it's possible for reverse currents to flow. For low-power diodes, the recovery time is usually measured in single nanoseconds, while larger diodes used in power supplies might need a microsecond or more.

Specialized Variants

As mentioned earlier, most diodes are designed to withstand reverse voltages in excess of 100 V. However, a special category of products, *Zener diodes*, is manufactured to undergo breakdown at much lower voltages. When forward-biased, a Zener diode behaves like a conventional diode, becoming conductive at around 600 mV. When reverse-biased, it starts conducting at a voltage specified by the manufacturer; common low-voltage options range from 1.8 V to 30 V.

From Figure 10-3, we know that the V-I curve of a diode in reverse breakdown is rather steep: A small change in the applied voltage can produce a comparatively large swing in the current flowing through. We can also turn the diagram sideways and see that if the current through the diode is limited in some way, fluctuations in that current will have a comparatively minor effect on the voltage that develops across the diode's terminals.

This observation makes Zener diodes useful as *voltage references*. Using an unregulated power source—for example, a battery—we can, in the simplest configuration, use a resistor to roughly limit the current flowing through the diode. From Ohm's law, the current admitted by the resistor will change in proportion to the supply voltage, but as shown in Figure 10-4, these current fluctuations affect the diode voltage to a much lower extent.

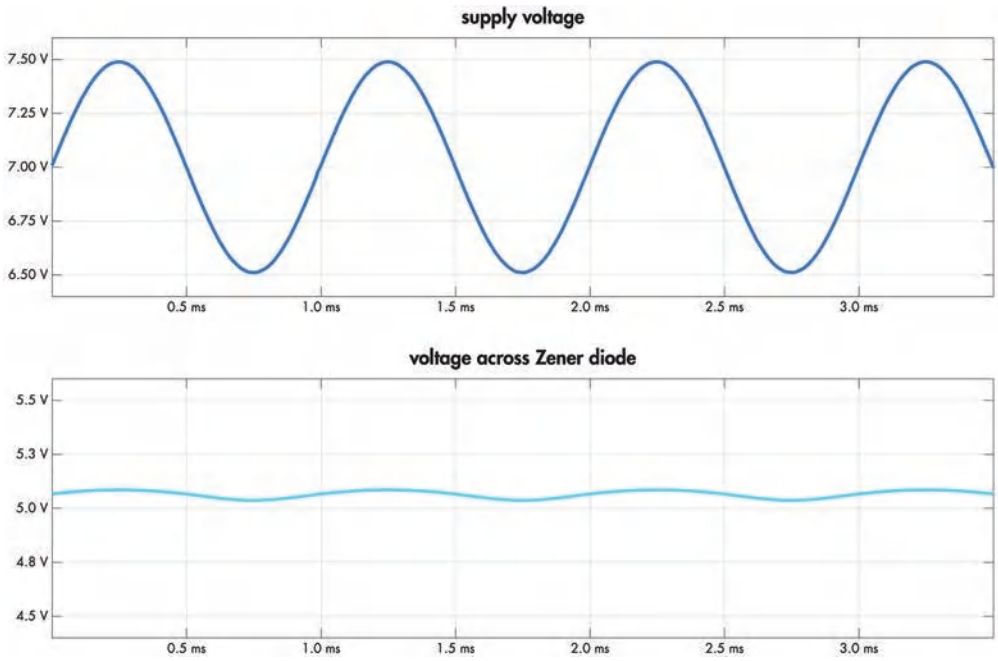


Figure 10-4: Fluctuating supply voltage versus the voltage measured across a resistor-limited 1N4733 Zener diode

Another popular two-terminal semiconductor is the Schottky diode. This device uses a modified junction formed between a doped semiconductor and an exotic metal. It operates much like a standard diode but offers faster recovery times and much lower threshold voltages—often around 200 mV. The trade-off is higher leakage current and noticeably poorer thermal performance.

In addition to these devices, you might also encounter *transient voltage suppressors* (TVSes). A TVS is simply a fast, reverse-biased diode placed between a protected signal line and ground. If the voltage stays below a certain threshold, the diode does nothing;

once the limit is exceeded, the component shunts the excess energy. The primary purpose of a TVS is to protect sensitive digital circuits from electrostatic discharge, especially near external connectors such as audio jacks or USB ports. In this respect, the component functions similarly to a metal-oxide varistor (see Chapter 9), though it's designed to guard against less energetic events—and survive them.

A range of more specialized diodes can be found in vintage circuits or in niche applications such as radar and microwave communications. These include tunnel diodes, Gunn diodes, IMPATT diodes, germanium diodes, and others. Some of these components exhibit unusual V-I curves with a saddle-shaped region in the middle where the current briefly drops as the voltage rises—a property that can be used to build unconventional amplifiers and oscillators. However, such devices are almost never encountered in hobbyist projects, and some are no longer available commercially.

Standard symbols for diodes are shown in Figure 10-5. The triangle at the center of each symbol is meant to resemble an arrow, indicating the direction of conventional current (from plus to minus) that flows when the diode is forward-biased.

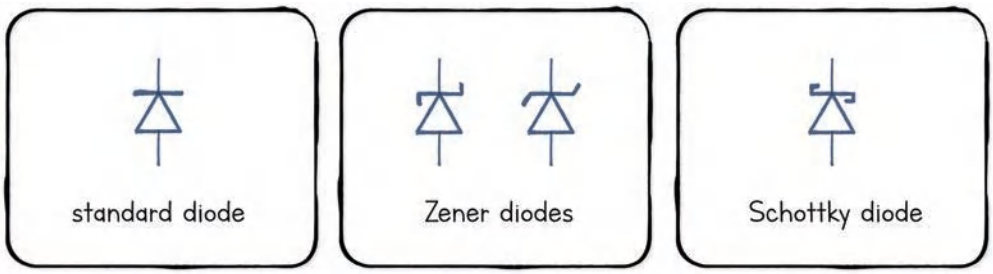


Figure 10-5: Common electronic symbols of diodes

A diode's polarity is typically marked on the component itself with a black or white band. The band corresponds to the horizontal bar in the schematic symbol, indicating the terminal must be negative to achieve forward bias.

Light-Emitting and Light-Sensing Diodes

In Chapter 2, we noted that mobile electrons in an n-type semiconductor occupy an energy level known as the conduction band. When these electrons cross into a p-type material, they fall into vacancies (holes) in the lower-energy valence band. If a sufficient forward-bias voltage is applied to the junction, the electrons continue their journey on the p-side, making room for additional n-side electrons to recombine with holes. Each recombination event releases energy determined by the difference between the conduction and valence energy levels. In a *light-emitting diode (LED)*, this bandgap is chosen so that the released energy is radiated away and appears as photons of visible, infrared, or ultraviolet light.

LEDs are typically made from alloys of gallium, indium, arsenic, and related compounds; as a result, they have higher threshold voltages than silicon. For red LEDs, the conduction threshold is usually around 1.9 V. Yellow and green diodes may require about 2.2 V, while blue LEDs might need as much as 3.4 V.

Similar to traditional diodes, the V-I curve of an LED is fairly steep: Small fluctuations in supply voltage produce large jumps in the current passing through the diode. To avoid accidentally destroying the junction, it's better to limit LED current than to fiddle with voltages. While household lighting and other high-power LEDs rely on more sophisticated current-regulating circuits, small diodes used as indicator lights typically need just 20 to 30 mA. In such cases, a standard resistor can be connected before the LED to set the current without producing a lot of waste heat. To find the correct value for this resistor, let's start with the circuit schematic shown in Figure 10-6.

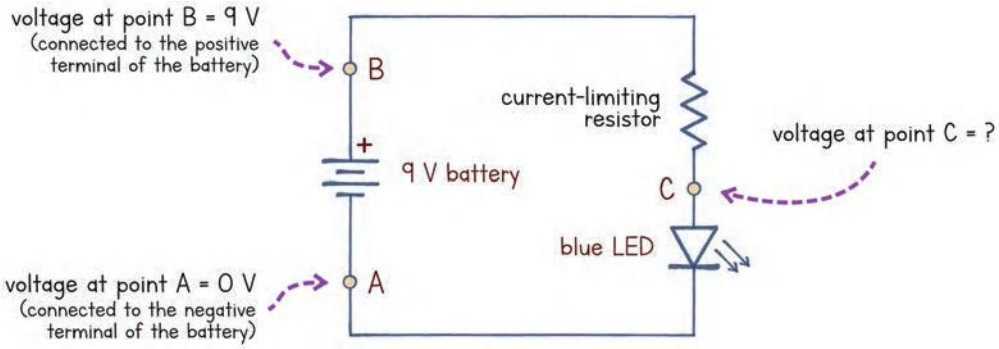


Figure 10-6: An analysis of voltages in an LED circuit

From Ohm's law, the current flowing through a resistor equals the voltage across its terminals divided by the component's resistance: $R = V / I$. Suppose we have a 9 V battery and wish to supply 20 mA to a blue LED. It's tempting to calculate the required resistance as $R = 9\ \text{V} / 20\ \text{mA} = 450\ \Omega$. However, that result isn't entirely correct: A diode will admit 20 mA only if a certain voltage is maintained across its terminals. In Figure 10-6, point C will measure around 3.4 V during operation. The resistor that limits LED current straddles points B and C, so it's subjected only to the voltage difference between them: 9 V − 3.4 V = 5.6 V. The optimum resistor value to limit the current to 20 mA is therefore 5.6 V / 20 mA = 280 Ω.

In practice, nothing bad will happen if we use a larger resistor. In fact, if you're designing electronic devices with standby lights, I urge you to dial the brightness down. A slightly smaller resistance is also acceptable, as long as current stays within certain bounds. If we connect the LED directly to a 9 V battery, hundreds of milliamps will flow through the junction, destroying the component in the blink of an eye.

Miniature LEDs made from novel organic semiconductors (OLEDs) are common in modern display panels. A typical screen contains hundreds of thousands—or even millions—of individual pixels, making it impractical to control each one independently. Instead, we use connection patterns similar to the matrix shown in Figure 10-7.

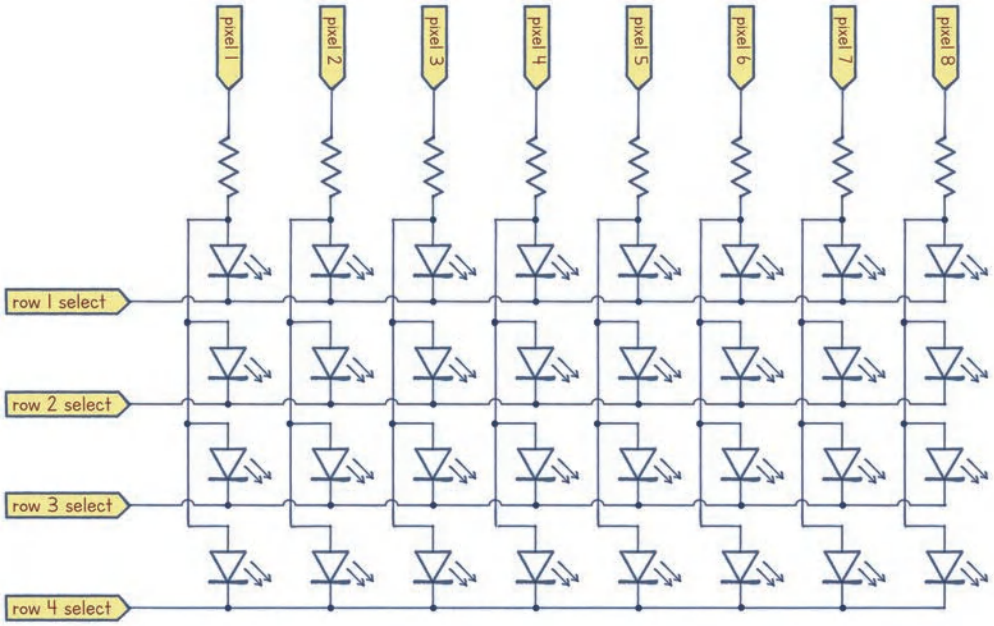


Figure 10-7: An 8×4 matrix of LEDs with current-limiting resistors

In this configuration, an external display controller connects one of the four rows to ground while leaving the remaining "row-select" lines floating. The circuitry then pushes data onto the eight pixel lines. Although the voltages applied to the pixel lines propagate through every row, only the diodes in the active row have a path to ground, so only those pixels will light up. If the controller cycles through the rows quickly enough, our eyes don't perceive the flicker, and the entire image appears steady. This phenomenon, known as the *flicker fusion threshold*, has been used at least since the days of silent films.

In Chapter 7, we discussed the physics of solar cells, in which a specially engineered p–n junction is exposed to light. Photons absorbed in the depletion region of the junction produce excited electron-hole pairs that are then driven apart by the region's internal electric field. The resulting effect produces a photocurrent proportional to the incident illumination. This induced current flows in the direction opposite to the forward current in a standard diode.

A *photodiode* is a light sensor that operates on the same principle as a solar cell. The main difference is that a photodiode is much smaller—often about the size of a standard indicator LED. As a result, it produces only tiny currents that require amplification before use. Even so, photodiodes are highly linear, respond quickly, and offer a huge dynamic range—from daylight to candlelight. Because of these advantages, they have largely replaced cadmium-based photoresistors (see Chapter 8) in consumer gear. The symbols for LEDs and photodiodes are shown in Figure 10-8.

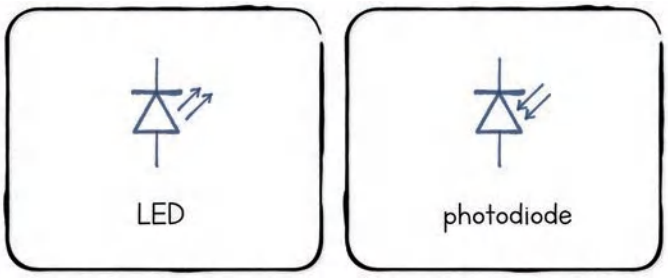


Figure 10-8: Symbols for the light-emitting diode and photodiode

Physical LEDs and photodiodes have polarity markings that vary by form factor. We'll cover component packaging later in the book. For now, it suffices to note that in through-hole components, the longer lead indicates the positive terminal. Surface-mount variants might use a dot, an arrow, or a clipped corner to mark the negative side.

Transistors

Compared with diodes, transistors form a more diverse and versatile class of semiconductors. They're used to switch and amplify electronic signals, and this can be achieved in a number of ways.

JFETs

We introduced junction field-effect transistors (JFETs) in Chapter 2. To recap briefly, the transistor's internal structure consists of a conductive channel that connects two terminals labeled source (S) and drain (D). Along the length of this channel lies a wedge connected to the gate (G) terminal. The conductive channel can be made from either an n-type or p-type semiconductor, and the wedge is always of the complementary material—so, in an n-channel transistor, it's p-type.

At the interface between the p-type and n-type materials, a depletion region forms. This keeps the gate isolated from the main conductive path as long as the voltage applied to that terminal stays within certain bounds. In an n-channel JFET, the gate must be kept below 600 mV relative to the conductive channel for this isolation to hold. For simplicity, we usually don't track the gate-to-channel voltage directly, as that parameter is hard to measure and depends on the transistor's geometry. Instead, we measure the voltage between the gate and source terminals and denote it as V_{GS}.

Response Characteristics

To operate an n-channel JFET, we can apply a relatively negative voltage to the gate terminal. This reverse-biases the p–n junction, causing the depletion layer to expand and narrowing the main conduction path from source to drain. Figure 10-9 shows the relationship between the gate-to-source voltage and the resulting channel current (customarily labeled I_D) for a fairly typical JFET. At around −3.5 V, the depletion region begins to straddle the entire conduction channel, and beyond that point virtually no current can flow.

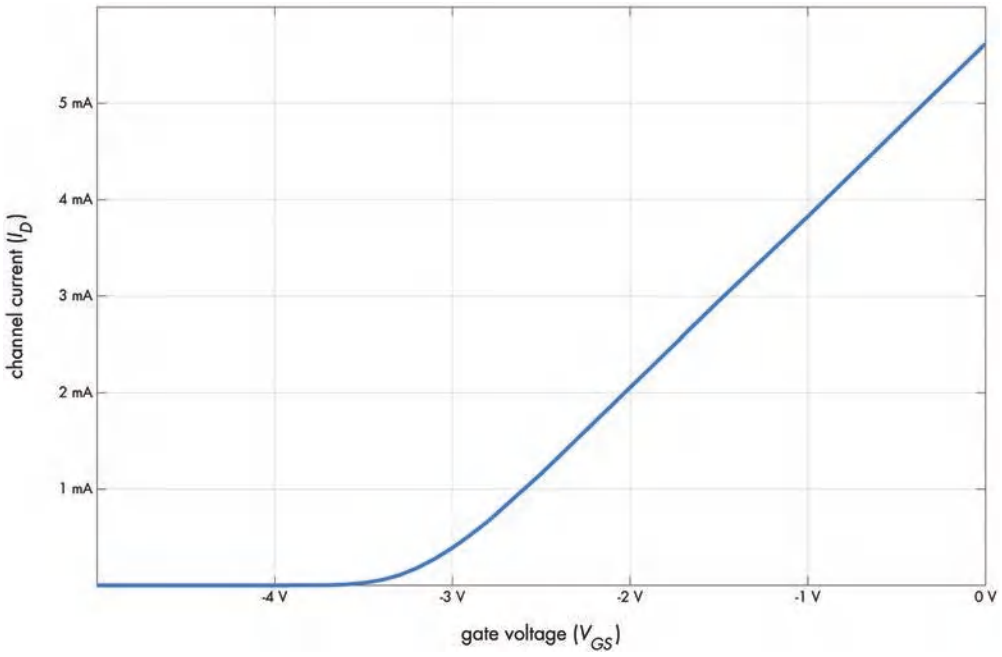


Figure 10-9: Gate-source voltage versus channel current in a 2N4416 n-channel JFET. The data was captured for a fixed drain-to-source voltage of V_{DS} *= 1 V.*

This curve is the transistor's *raison d'être*: It shows that if we can shuffle a comparatively modest number of electrons onto the JFET's gate and back, we can command respectable currents flowing between the other two terminals. It's the fundamental operating principle of amplifiers, a topic we'll explore in detail in Part III of the book.

Based on the visualization in Figure 10-9, it might be tempting to think of the transistor as an adjustable resistor, with the gate voltage acting as an electronic knob. This isn't quite correct. In a resistor, we would expect the current to be linearly proportional to the voltage that pushes electrons through the channel. If we plot the relationship between the drain-to-source voltage and the resulting current for some chosen V_{GS}, however, we get a nonlinear curve, as shown in Figure 10-10.

The initial, steep portion of the plot, corresponding to relatively low drain-to-source voltages (up to around 2.5 V), is sometimes called the *ohmic region* because it could pass for a resistor's V-I curve, especially if your eyesight isn't good. The gentle slope that follows, from about 4 V onward, can't possibly be approximated by Ohm's law ($I = V/R$) because the line doesn't point toward the center of the coordinate system; this is known as the *saturation region*. In this section, the transistor behaves like a nearly constant-current device. The current limit is set by V_{GS} and varies relatively little with voltage: In the plot, a fourfold increase in drain-to-source voltage—from 5 V to 20 V—causes I_D to rise by only about 15 percent.

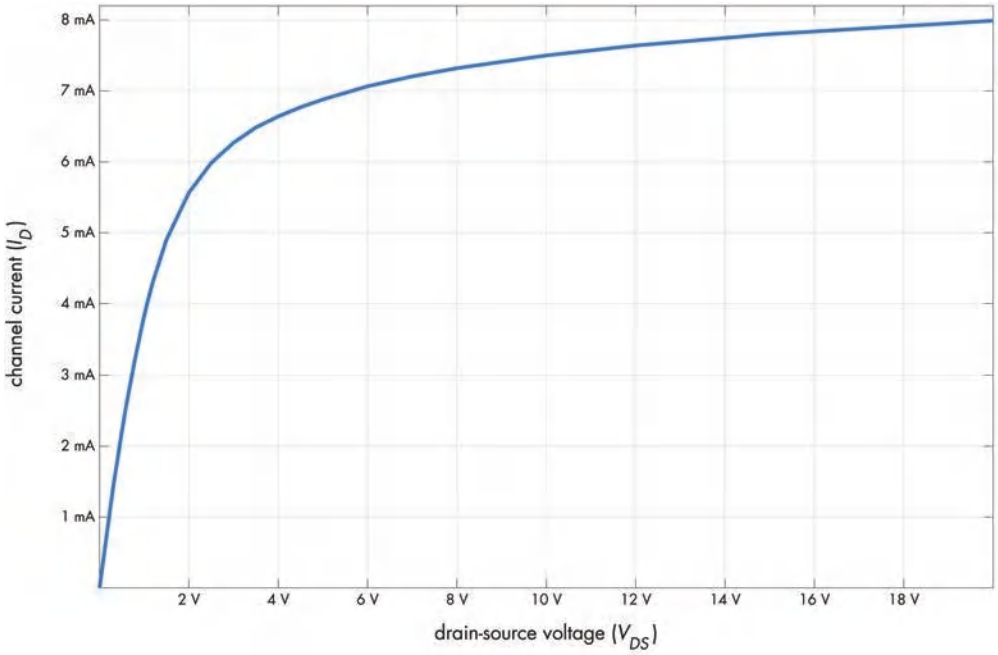


Figure 10-10: Drain-source voltage versus channel current in a 2N4416 transistor (gate voltage fixed at V_{GS} = –1 V)

The two-part shape of the curve in Figure 10-10 may seem puzzling, but it stems from our simplified use of the gate-to-source voltage, V_{GS}. The abstraction is helpful in circuit design, yet inside the transistor we're dealing with an interplay of electric fields contributed by all three terminals; the drain voltage on the drain terminal makes the channel more positive too. If we instead calculate the channel voltage as the average of the drain and source voltages and use that as the horizontal axis, the JFET plot will look much more like a straight line.

Although the V_{GS} versus I_D and V_{DS} versus I_D curves are important, their exact shapes shouldn't be relied upon. The readings fluctuate with temperature—including self-heating effects within the junction itself—and to capture the plots for this chapter, I had to submerge the transistors in oil and pace myself. Just as important, characteristics vary from one component batch to another. When we need precision, additional circuitry to compensate for transistor behavior is therefore a must. We'll discuss ways to accomplish this in Chapter 14 and Chapter 15.

Varieties, Symbols, and Secondary Specs

So far, we focused on n-channel JFETs. The transistor can also be made with a p-type channel and an n-type gate. In that configuration, the p–n junction is reversed, so it becomes conductive when the gate voltage is about 600 mV lower than the channel. The transistor is therefore normally operated with a positive V_{GS}. Such p-channel JFETs suit situations where we want to switch or amplify signals that swing toward the positive supply rail. However, because n-type materials conduct electricity better, we use the p-channel variety sparingly.

Field-effect transistors come with a variety of factory specs, most of which should be familiar by now. These include the maximum safe current for the main channel and gate pathway, along with maximum voltages that may appear across the terminals without risking electrical breakdown. Other specs of note include the cutoff gate voltage and the approximate "on" resistance measured at V_{GS} = 0 V and very low V_{DS} (that is, in the ohmic portion of the transistor's response curve).

The junction between the gate terminal and the channel acts as a diode, so it has some parasitic capacitance. This parasitic capacitance limits the transistor's response speed and allows sufficiently high-frequency alternating currents to flow between the gate terminal and the channel. This is known as *input capacitance*; for small JFETs, it usually ranges from 5 to 30 pF.

For roughly the same reason, when the transistor is in the "off" state, some nontrivial capacitance can also be measured between the drain and source terminals. The value is lower, typically under 5 pF, but it might provide a bypass route for alternating currents. In Chapter 5, we derived the concept of resistor-like capacitive reactance:

$$X_C = \frac{1}{2\pi f C}$$

For a 5 pF capacitance, this equation works out to a parasitic 318 Ω manifesting for sine-wave signal frequencies of 100 MHz.

The symbols of JFETs are shown in Figure 10-11. The distinction between n-channel and p-channel boils down to the direction of the arrow.

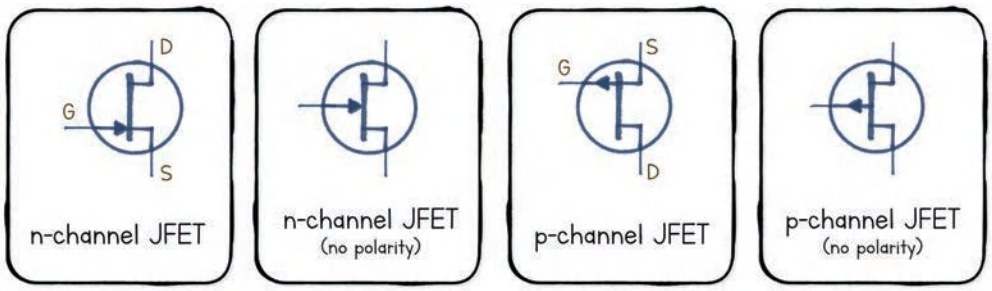


Figure 10-11: Schematic symbols of junction field-effect transistors. The circles are sometimes omitted.

Most of the time, drain and source terminals are functionally interchangeable for this type of transistor, but for some channel geometries, which terminal is which may matter. The source terminal is the one we reference the gate voltage to. In n-channel devices, the source terminal is also usually kept at a lower voltage than the drain (and vice versa for p-channel FETs).

There are no reliable rules of thumb for guessing the physical pinout of transistors: It varies even for products in the same form factor and made by the same company. Always check the datasheet.

Metal-Oxide-Semiconductor FETs

Metal-oxide-semiconductor field-effect transistors (MOSFETs), also introduced in Chapter 2, resemble junction field-effect transistors in that a voltage applied to the gate terminal controls current flow in the drain-source path. Unlike JFETs, however, a standard MOSFET is off by default due to a reverse-biased p–n junction in the conduction path. The device begins conducting only when the gate voltage reaches a certain threshold, $V_{GS(th)}$; for discrete n-channel transistors, this is usually around 2 V.

The threshold exists because mobile charge carriers must be pulled from the channel toward the gate terminal and smushed against the oxide layer to create a conductive path. (For a glimpse of the mechanics, flip back to Figures 2-5 and 2-6 in Chapter 2.)

Response Characteristics

The relationship between V_{GS} and channel current for a typical n-channel transistor of this type is shown in Figure 10-12.

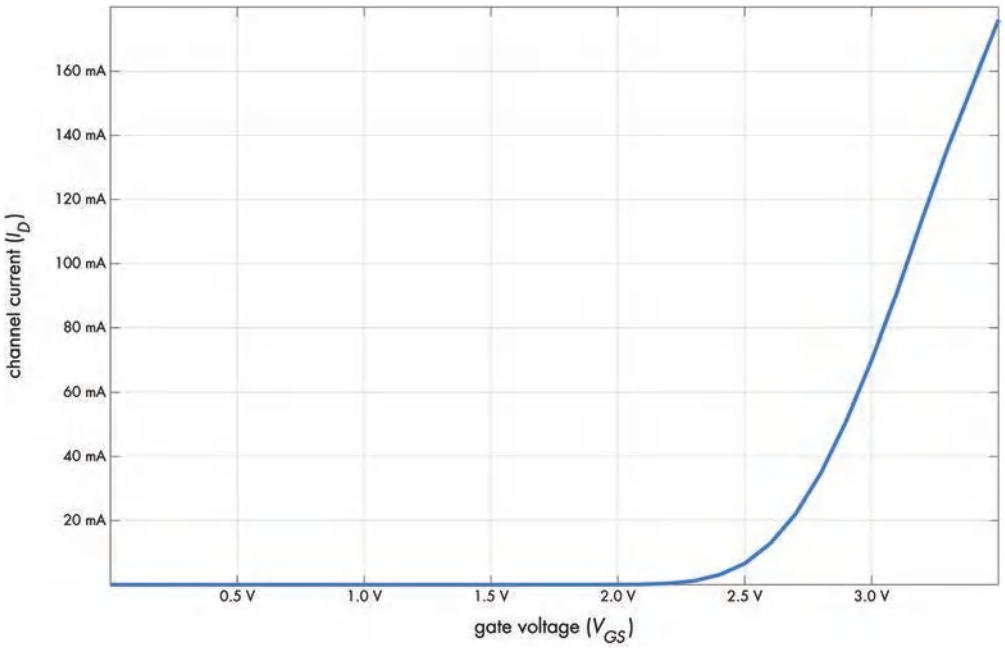


Figure 10-12: Gate voltage versus channel current in a BS170 n-channel MOSFET (captured at V_{DS} = 1 V)

Note that the business end of the curve is much steeper than for JFETs, so these transistors can achieve higher signal amplification ratios (albeit with poorer linearity). A related advantage of the architecture is that the component can control much higher drain-source currents than a JFET of comparable size.

Even more than a JFET, a MOSFET shouldn't be thought of as a variable resistor controlled by a specific gate-to-source voltage. In a resistor, the relationship between the voltage applied to the conductive path and the resulting current is linear. In a MOSFET, however, we end up with the sharply nonlinear curve shown in Figure 10-13 In the plot, the ohmic portion is quite narrow, appearing only below about 500 mV, while the constant-current (saturation) region is comparatively wide.

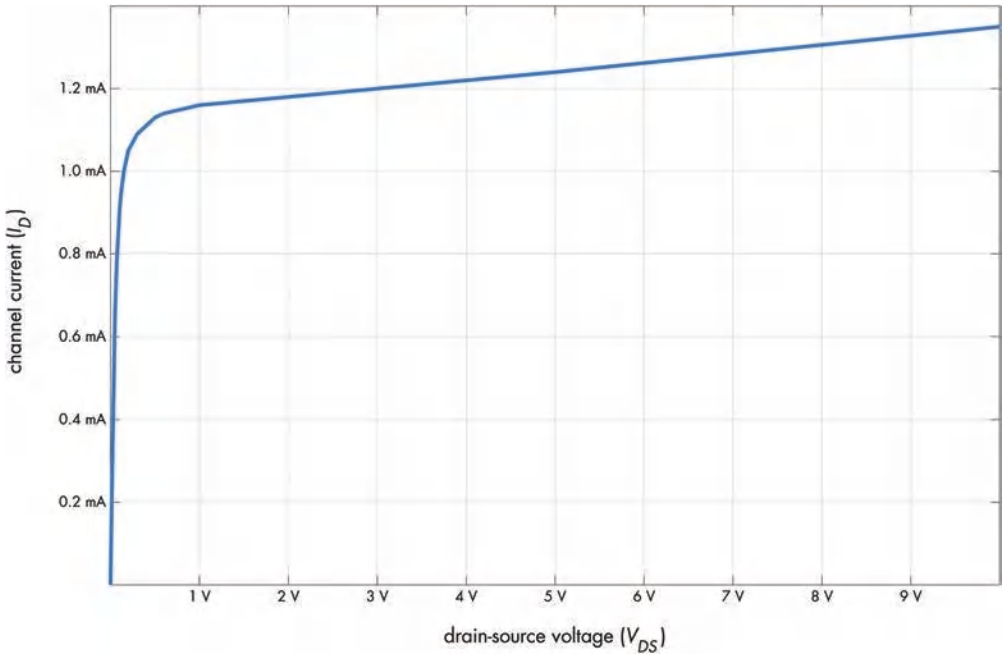


Figure 10-13: Drain-source voltage versus current in a BS170 transistor (captured at V_{GS} = 2.2 V)

When describing transistors, we sometimes refer to an abstraction called *dynamic resistance*. This is the apparent resistance of a transistor at specific values of V_{DS} and V_{GS}. However, if the applied voltages change, so does this parameter. Therefore, the concept can be misleading, and the terminology must be used with care.

The oxide-insulated gate in a MOSFET is both a blessing and a curse. On the upside, unlike JFETs, there's no risk of DC flowing through the gate terminal if we accidentally forward-bias it relative to the conductive channel. On the flip side, the gate acts as a capacitor that can store charge for a long time. If the terminal is left floating, the transistor may randomly turn on, turn off, or even pick up radio interference. This behavior enables some exciting applications such as nonvolatile computer memory, though it occasionally stumps novice circuit designers.

Another trade-off is that the insulating layer, which is just nanometers thick, is exceptionally vulnerable to electrostatic discharge; even a tiny spark can create a big hole. This is a major reason why discrete transistors and integrated circuits ship in special packaging and why commercial electronics workshops usually invest in gear such as humidifiers, dissipative work mats, and wrist straps. In a hobby lab this is usually unnecessary, but it might be worth looking into if you live in a dry climate or get frequent static shocks at home.

Varieties, Symbols, and Secondary Specs

Most MOSFET parameters are analogous to those of a JFET. About the most notable difference is that MOSFETs used to drive motors and other high-current devices tend to have physically large gates and thus input capacitances well into the nanofarad range. This can make the transistor harder to switch on and off rapidly.

Unlike JFETs, discrete MOSFETs are polarized: They remain nonconductive in the "off" state due to a single reverse-biased p–n junction (*body diode*). Another junction appears in diagrams (see Figure 2-5 in Chapter 2), but it's electrically bypassed by an internal connection that enables the gate-substrate interface to function as an easily charged capacitor. In an n-channel MOSFET, the source terminal must therefore

be kept at a voltage similar to or lower than the drain terminal. In a p-channel transistor, the inverse applies. Reversing the polarity turns the drain-source pathway into a forward-biased diode.

In addition to standard (*enhancement mode*) MOSFETs, you might encounter *depletion mode* devices. These transistors are on by default and turn off only when sufficient gate voltage is applied. They are more or less novelties: A few exist on the market and appear in textbooks, but it's unusual to encounter one in the wild.

The usual symbols for MOSFETs are shown in Figure 10-14. As with JFETs, n-channel and p-channel variants are distinguished by the direction of an arrow. It's also a good practice to draw the symbols so that the terminal which must remain more positive points upward. Unfortunately, you'll encounter plenty of schematics where symbols are rotated or flipped in random ways.

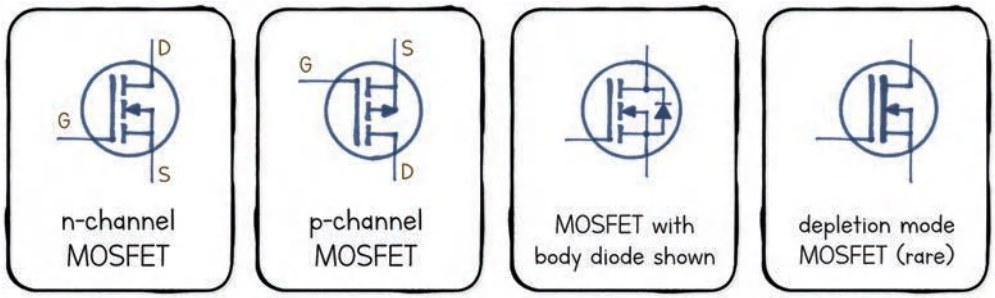


Figure 10-14: Typical symbols of metal-oxide-semiconductor field effect transistors

Note also the second panel from the right: Sometimes the internal parasitic diode inside the transistor is included in the drawing. If you must draw the transistor in an unusual orientation, including this element makes the circuit easier to understand at a glance.

A Simple MOSFET Experiment

If you're feeling overwhelmed by theory, Figure 10-15 should offer some respite. It outlines a simple MOSFET circuit you can build and play with on your own terms.

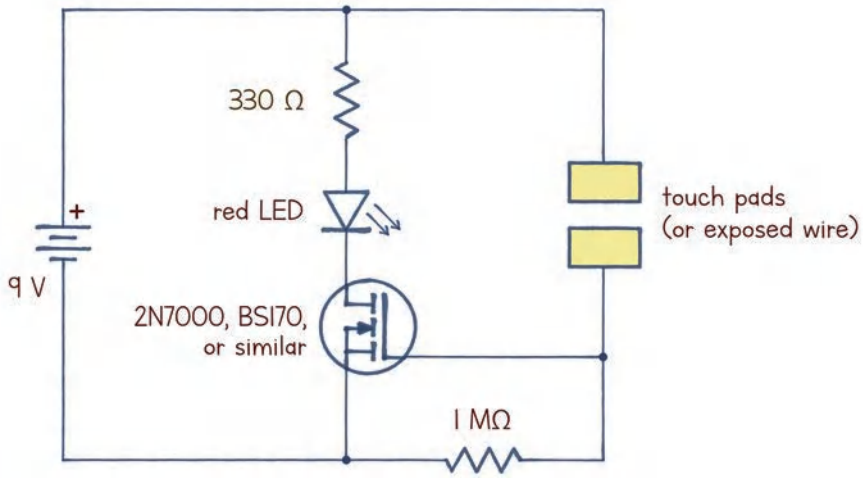


Figure 10-15: A touch-sensing circuit using a MOSFET and an LED

The centerpiece of the design is a field-effect transistor that controls an LED connected through a current-limiting resistor. Almost any n-channel MOSFET should work; the 2N7000 and BS170 are two good, inexpensive choices. The gate of the transistor connects to a touch pad, which can simply be a piece of stripped wire. Another touch pad connects to the positive terminal of a 9 V battery.

When you touch both pads with your fingers, the microamp-range current flowing through your body should be enough to increase the gate voltage above the 2 V threshold, lighting the LED. When you let go, the gate discharges to ground through the 1 MΩ resistor at the bottom, turning the transistor off. If you remove that resistor, you might be able to "latch" the transistor with your touch or make the LED blink erratically from radio frequency interference picked up by the wire.

Use this circuit as a springboard for experimentation. Does it light up if you submerge the leads in water, alcohol, or cooking oil? What happens if you reverse the transistor's polarity? Can you build a similar circuit for a JFET, if you happen to have one around?

Bipolar Junction Transistors

A bipolar junction transistor (BJT) consists of three semiconducting layers. Figure 10-16 shows the common NPN variant of this device. As the name implies, it consists of two n-type layers sandwiching a thin p-type layer in between.

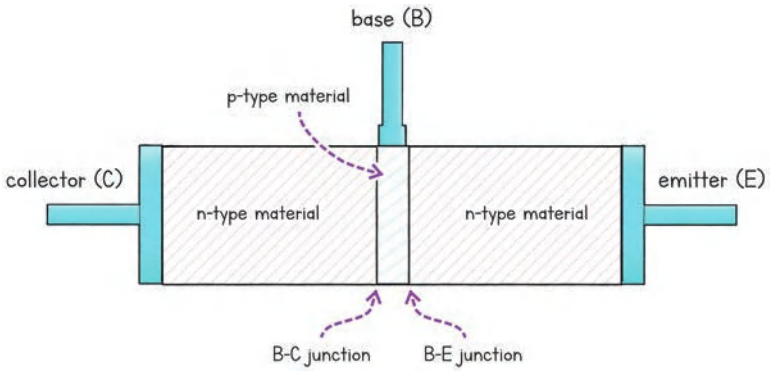


Figure 10-16: The internal structure of an NPN bipolar junction transistor

The primary conduction path in the transistor extends between the collector (C) and emitter (E) terminals. Two p–n junctions lie along the way: One depletion region forms between the p-type base and n-type collector (B-C), and another—oriented oppositely—lies between the base and n-type emitter (B-E). In this setup, it's seemingly not possible for any collector-emitter current to flow, because if one of the junctions is forward-biased, the other ends up under reverse bias. However, where there's a will, there's a way.

Conduction Model

To understand how BJTs actually operate, recall that the depletion region of a p–n junction isn't electrically neutral. Some n-side electrons fall into p-side holes, leaving a positive charge on the n-side and a negative charge on the p-side. The resulting electrostatic field acts as a sorting mechanism for stray charge carriers that wander into the region: Free electrons are jettisoned to the n-side, and holes are pushed to the p-side. This makes the junction largely impassable to majority carriers—that is, to electrons coming from the n-side where they're normally abundant, and to holes diffusing

from the p-side where that entity is dominant. At the same time, if the odd carrier wanders in from the "wrong" direction—for example, if an excited electron somehow shows up on the p-side—the field will chuck it across the junction. This allows some microscopic, momentary currents to flow, but the effect is typically negligible.

In a BJT, two junctions lie in the collector-emitter path, but only one separates the base terminal from the emitter. If the B-E voltage exceeds roughly 0.6 V, the right junction in Figure 10-16 begins conducting. One of the two depletion layers is disrupted, and electrons from the emitter area start pouring into the center region.

In the NPN layout shown earlier, this middle layer is exceedingly thin. Many charge carriers thus find themselves within reach of the internal electrostatic field from the still-intact B-C junction on the left. These electrons are on the "wrong" side of the field, so—as we've discussed—they get swept over en masse to the collector side. In effect, a small base-emitter current opens the floodgates for much higher collector-emitter current across the device.

Regrettably, this isn't the entire story for BJTs: The model holds only if the base voltage remains lower than the collector voltage ($V_{BE} < V_{CE}$). If this condition is reversed, the B-C diode (left) becomes forward-biased and starts conducting too, which throws a wrench in the works. We'll have a look at this phenomenon shortly.

Response Characteristics

In their normal ($V_{BE} < V_{CE}$) operating regime, BJTs are most usefully modeled as current-controlled devices: I_B sets I_C. Of course, as in any diode, the current that flows through the B-E junction is related to the applied voltage. Within a narrow and somewhat unpredictable range, this type of transistor can therefore be treated as voltage-controlled: Once we exceed the V_{BE} threshold of about 0.6 V, further base-voltage increases drive I_C higher, as shown in the response curve in Figure 10-17.

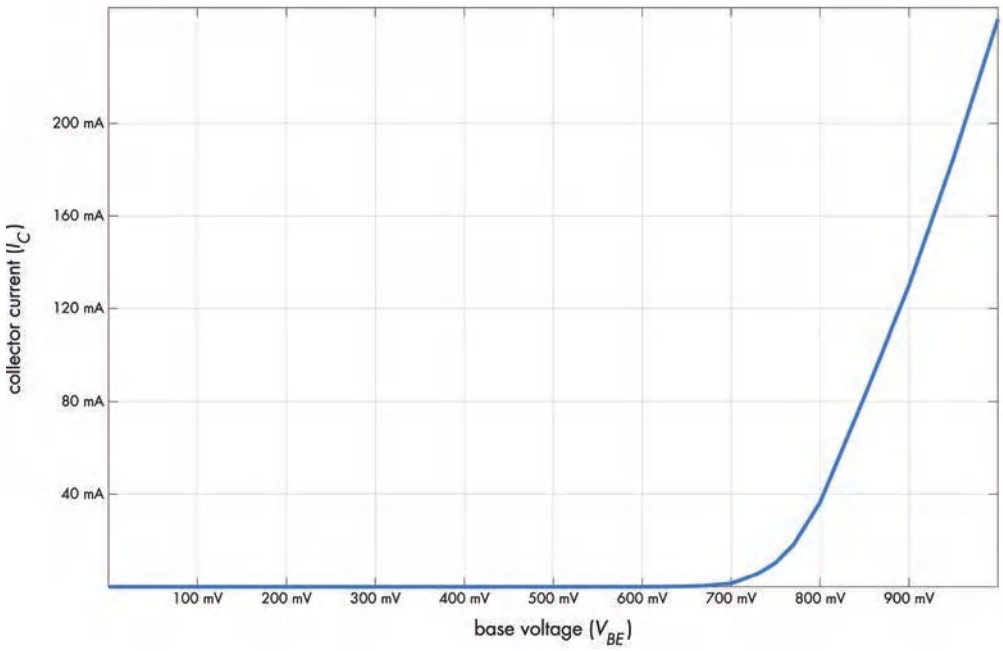


Figure 10-17: Base voltage versus collector current in a 2N3904 NPN transistor (captured for V_{CE} = 1 V)

Still, except for simple switching applications, the performance of a BJT is best understood through the relationship between base current and collector current.

Figure 10-18 shows the plot of I_B to I_C. This plot is linear enough to suggest a specific current gain: $I_C \approx I_B \cdot \text{const}$. This amplification factor can be measured with varying degrees of rigor; the most common terms are transistor beta (β) and h_{fe}.

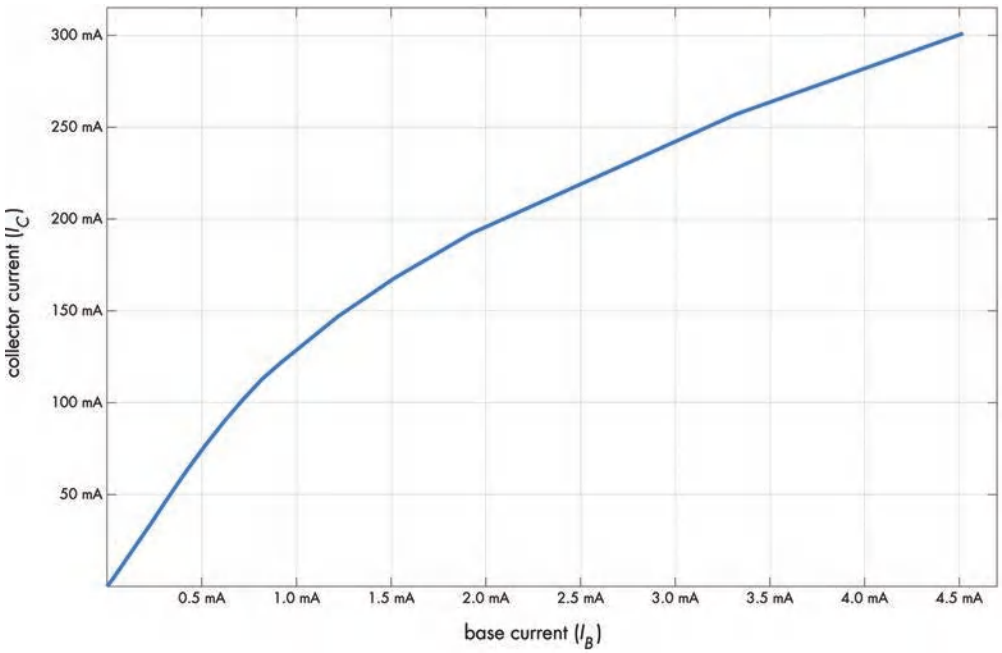


Figure 10-18: Base current versus collector current in a PN2222 transistor, showing current gain in the general vicinity of 100 (captured for V_{CE} = 1 V)

Like MOSFETs and JFETs, a BJT isn't simply an electronically controlled resistance. For a given control current I_B, the current admitted through the main conduction path varies nonlinearly in response to the collector-emitter voltage (Figure 10-19). The plot looks nearly identical to those for FETs (Figures 10-10 and 10-13), though the underlying mechanism is different.

In essence, the transistor acts as a roughly constant-current source in the $V_{BE} < V_{CE}$ region of operation. The behavior is different in the aforementioned case of $V_{BE} > V_{CE}$ (the steep slope on the left of the plot). At any point, to drive meaningful base current, V_{BE} must exceed 0.6 V. At the same time, if the base terminal voltage rises above the collector voltage, the B-C junction becomes forward-biased, resulting in a pair of bog-standard conducting diodes in the main current path. These diodes make I_C sharply dependent on V_{CE}.

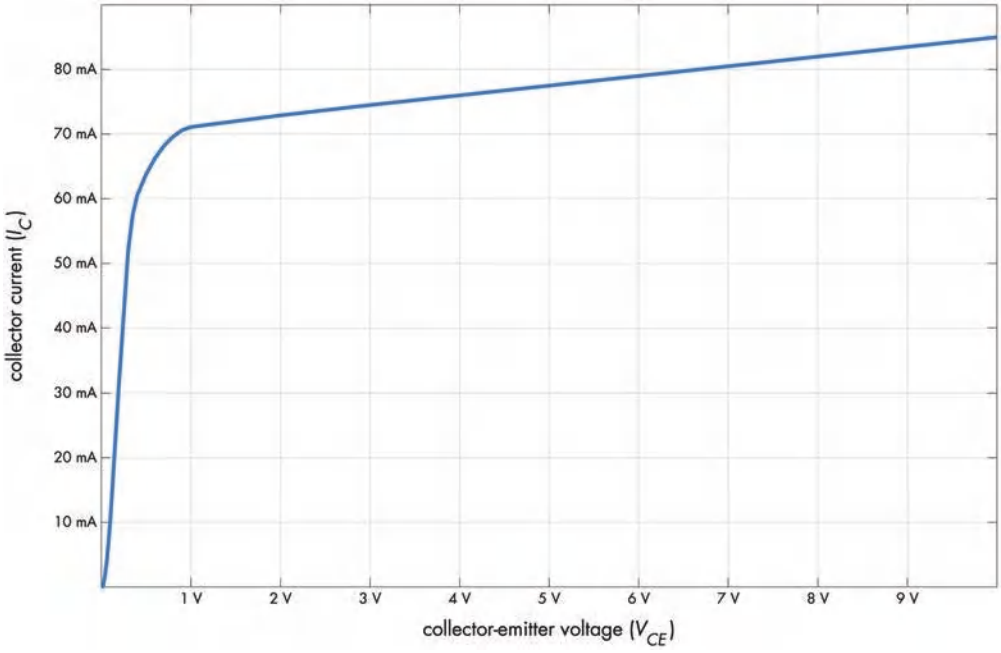


Figure 10-19: Collector-emitter voltage versus collector current in a PN2222 transistor (captured for I_B = 500 µA)

As you might recall, in field-effect transistor nomenclature the constant-current portion of the main-path voltage-to-current plot is called the saturation region, since the channel voltage no longer significantly affects the current. Confusingly, BJT terminology reverses this: The saturation region refers to the diode-like $V_{BE} > V_{CE}$ conduction regime, where the control signal (I_B) has little impact on I_C. This difference in terminology creates chaos, but we seem to be stuck with it.

Similar to FETs, the terminals of a BJT have parasitic capacitances. The datasheet typically specifies input capacitance (C_{ibo}) and output capacitance (C_{obo}). The first value roughly corresponds to the capacitance of the B-E junction in reverse bias (that is, when the transistor is off). The second approximates the capacitance of the reverse-biased B-C junction. Both capacitances limit the transistor's response speed.

Symbols and Varieties

In addition to the NPN architecture we've focused on, BJTs are also made in a PNP layout. The relationship is similar to that between n-channel and p-channel FETs: In a PNP device, the emitter is usually held at a higher voltage, and the base begins conducting when its voltage is about 0.6 V lower.

The base terminal of a BJT can be replaced with a photodiode. The result is a *phototransistor*, in which the current generated by the photodiode is internally amplified from microamps to milliamps. This saves you the effort of adding a separate amplifier. On the flip side, because BJT current gain is only roughly linear and temperature-dependent, the resulting measurements aren't particularly precise.

The usual schematic symbols for these transistors are shown in Figure 10-20.

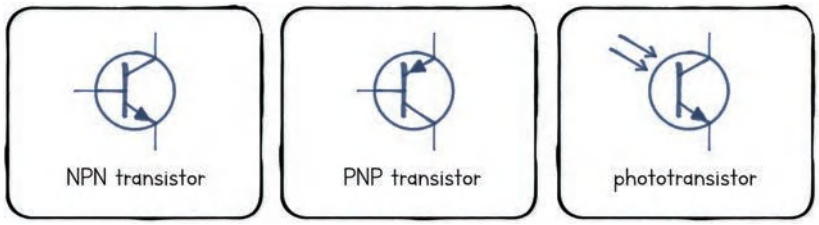


Figure 10-20: Symbols of bipolar junction transistors

Like MOSFETs, BJTs have polarity. In an NPN device, the emitter region is more heavily doped and should be kept at a lower voltage than the collector, serving as the initial source of electrons that pour into the base area. In PNP devices, the relationship is reversed. When wired the other way around, the transistor will still function, but performance falls short of specifications.

You might also encounter composite devices that combine two BJTs in a single package—an intermediate form between discrete components and integrated circuits. The two most important examples are *Darlington pairs*, in which two transistors cascade to achieve higher gain; and *thyristors*, also known as *silicon-controlled rectifiers (SCRs)*, which latch on in response to a pulse supplied to the control terminal. These are used in AC power circuits to implement light dimming or motor speed control.

Remembering the Vacuum Tube

Unless you're an eccentric collector or an audiophile, you probably don't have many vacuum-tube appliances in your home. Nevertheless, this now-obsolete technology merits reminiscence. Vacuum tubes kickstarted modern electronic theory and paved the way for its many applications, from space exploration to magnetic resonance imaging. Some examples of these components are shown in Figure 10-21.



Figure 10-21: Vacuum tubes big and small

To see the parallels between old and new technology, consider the simplest vacuum tube. Just like its modern-day counterpart, it's known as a *diode*. The device consists of two electrodes: an electrically heated cathode that serves as the negative ter-

minal and an unheated anode (plate). Thermionic emission (see Chapter 2) allows electrons to escape from the red-hot electrode into the surrounding vacuum. In the presence of an electrostatic field, they then drift toward the plate. Because only one electrode is hot, current flows in this direction only.

Another riff on the theme is a *triode*, which places a mesh-like electrode in between the other two. This terminal, known as the *grid*, has a low cross-section, so relatively few electrons end up hitting it. Instead, it's a source of an electrostatic field that either cancels the plate's pull or reinforces it when voltage is applied, accelerating electrons and helping them reach the plate. This allows the current to be modulated in a manner essentially identical to what we see in field-effect transistors, though with much better linearity (Figure 10-22).

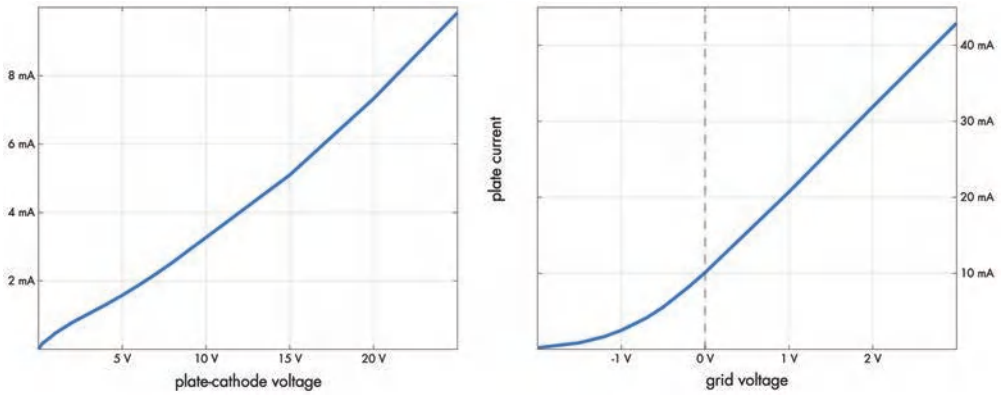


Figure 10-22: GE 6GM8 vacuum tube response curves. Left: diode mode, plotting plate voltage versus plate current (grid fixed at 0 V). Right: triode mode, plotting grid voltage versus plate current (plate fixed at 25 V).

In the end, what doomed the vacuum tube wasn't that the transistor performed better or offered any new tricks. It was enough that the new tech didn't require an always-on heater coil—one that made tube computers consume as much electricity as a small town. The tube was already dead in the water by the time integrated circuits arrived.

Even before this brief trip down memory lane, we had everything we needed to know about transistors and other basic electronic components. In the next chapter, we'll take a look at the conventions for capturing schematics. After that, we'll dive right into the theory and practice of circuit design.

11

CREATING CIRCUIT SCHEMATICS



Throughout Part II of this book, I gradually introduced the elements of circuit schematics. We started with component symbols in Chapter 7, sketched lifelike equivalent circuits in Chapter 8 and Chapter 9, and wrapped up with streamlined diagrams in Chapter 10. In this chapter, we'll cap everything off with a closer look at the conventions for creating schematics. Being able to plan out a circuit in a schematic diagram is essential for keeping your experiments organized and for communicating your ideas to others—or to your future self.

While we'll discuss best practices, for the most part there are no strict rules you need to obey. Simply put, a good circuit diagram is one that's unambiguous and easy to follow. Inventing your own symbols for common components would be a bridge too far, but you can approach the finer points in any way that suits your style and clearly communicates your intent.

There are a fair number of free programs for creating schematics (the industry term is *schematic capture*), but most take time to master because they're part of sprawling PCB design packages. For example, they throw a hissy fit if you just want to draw a generic integrated circuit; they expect you to have the component's detailed specs, down to a physical model for eventual PCB placement. My best advice as you're learning is to sketch your first schematics by hand. We'll circle back to circuit board design software in Part VI, but it's not a hill you need to climb on day one.

Representing Connections

Each electronic component has a certain number of terminals. On a circuit diagram, we join these terminals with lines (*wires*) representing electrical connections (Figure 11-1, left). We typically use horizontal and vertical segments aligned to a uniform grid. Nothing bad will happen if you violate this convention, but arbitrary angles and inconsistent distances usually make the drawing much harder to read.

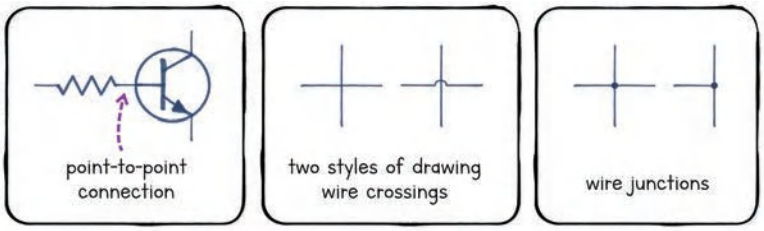


Figure 11-1: Wires, wire crossings, and junctions in circuit schematics

Wires can cross in a diagram without forming an electrical connection (Figure 11-1, center). To distinguish such crossings from actual connections, we mark wire junctions and branches with a heavy dot (Figure 11-1, right). A collection of all wires and component terminals that are electrically tied together is called a *net*. A typical circuit may have from two to several dozen nets.

Digital circuits (covered in Part IV) frequently use multiple signal lines to transfer a single unit of data—for example, several bits transmitted in parallel, accompanied by a clock. Such a collection of logical signals is called a *bus*. If we aren't careful, these groups of connections can quickly clutter a schematic, crisscrossing needlessly and obscuring text. For example, consider the data bus between two integrated circuits shown in Figure 11-2. It would take a team of forensic wire-tracers to make sense of what's going on.

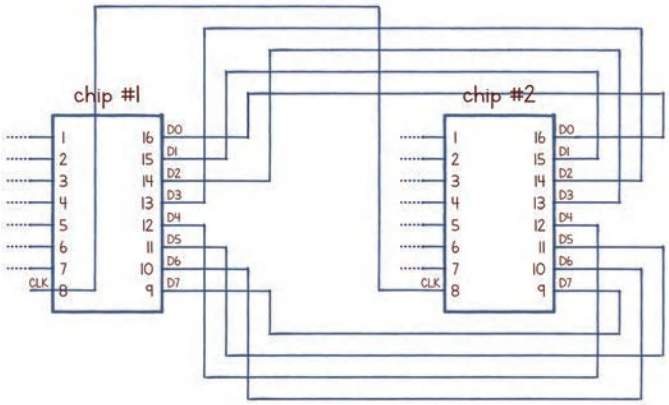


Figure 11-2: A bad example of wire routing and text placement

We can reduce clutter and improve readability by representing the data bus as a single thick line (Figure 11-3). This notation doesn't mean the constituent wires are shorted together. Instead, think of it as a cable with multiple conductors. Whenever you use this approach, ensure a clear correspondence between the wires going in and those coming out.

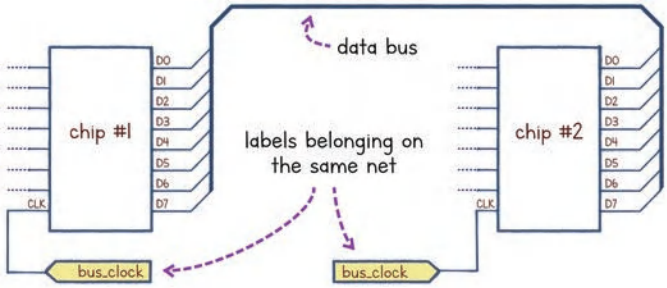


Figure 11-3: The use of bus connections and labels to reduce clutter

In this drawing, we're saying there is a connection between the D0 line on the one and on the right, but there's no electrical short between, say, D0 and D5.

Another common technique for reducing clutter, also illustrated in Figure 11-3, is to terminate a wire with a label, shown here as a dangling tag but often just as text near the wire end. The understanding is that all labels with the same designation belong to the same net, indicating electrical connection without drawing long lines. This approach helps but should be used sparingly. A well-routed wire is easy to follow; hunting for matching labels on a complex schematic can be a chore.

Component Drawing Styles

Electronic components can be drawn in two styles: as nondescript boxes or as symbols that hint at the component's function (Figure 11-4). While symbols always include identifying information such as a manufacturer's part number, you should use self-documenting visuals whenever possible. Overusing box drawings makes the diagram less readable, even if you happen to know every part number by heart.

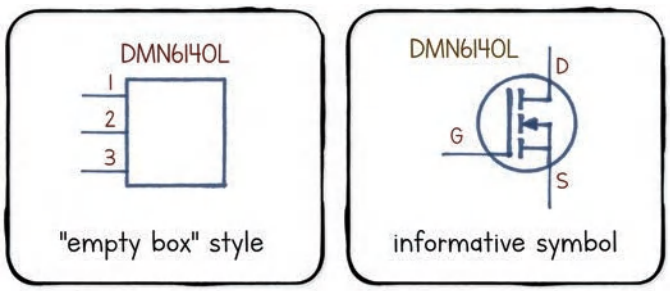


Figure 11-4: Two ways of representing the same n-channel MOSFET

For commodity parts such as resistors and capacitors, specify values instead of part numbers. If you write "GRM155R61A106ME11D" next to a capacitor, no one will know what you mean; "10 µF" is much easier to parse. If the choice needs to be narrowed down—for example, a ceramic capacitor with a particular voltage rating—spell that out too.

Some designers use shorthands for component values. This is particularly common for resistors, where values often omit the unit symbol: "10 kΩ" becomes "10k." Another shorthand replaces the decimal dot with an SI prefix, so "1.5 kΩ" morphs into "1k5." To extend this notation to resistances lower than a kiloohm, we can write "6R8" instead of "6.8 Ω." Such hijinks would be out of place for capacitors or inductors, although you will encounter "u" in lieu of "µ."

For integrated circuits, we resort to box drawings. The only exception is chips that contain a manageable number of standard functional blocks, such as amplifiers or standalone logic gates. In that case, we can draw the symbols of those functional blocks, perhaps with an annotation suggesting that they belong to the same chip.

There are two schools of thought on how to draw the pins of an integrated chip. You can follow the physical layout on the package; to me, this can make it easier to wire the device and troubleshoot it down the line, but others might disagree. Alternatively, you can reorder the pins by function, with no regard for how they're organized in the real world. Do what feels more logical. In all cases, ensure that pin numbers and their functions are clearly marked on the diagram.

Logical Flow

It's good to lay out the circuit so that it reads naturally or resembles how similar schematics are customarily drawn. In particular, if the circuit has inputs and outputs, it's beneficial to place the inputs on the left and the outputs on the right and then organize the rest so that you don't need to jump back and forth to trace the signal path. This principle is illustrated in Figure 11-5.

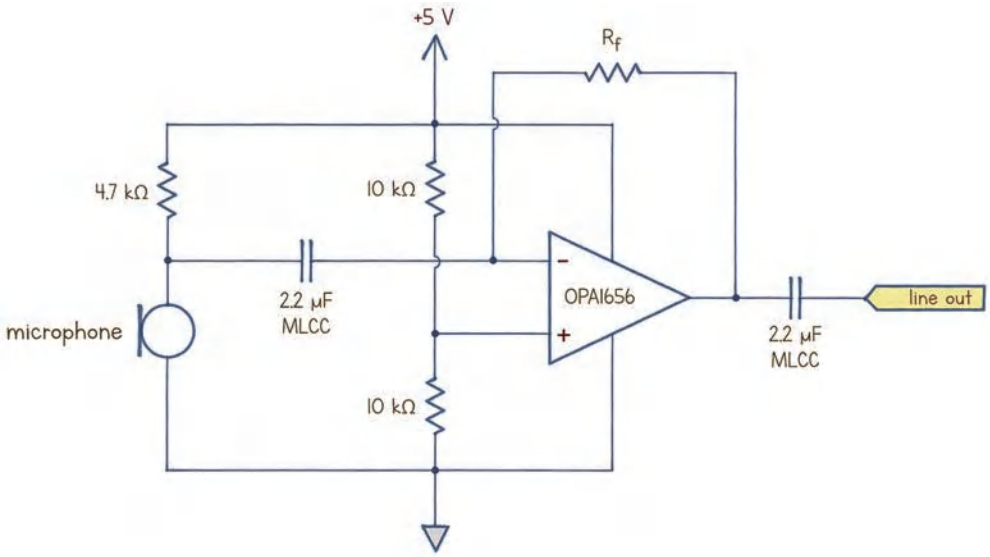


Figure 11-5: A rudimentary microphone amplifier circuit in a customary left-to-right layout

Even though you might be unfamiliar with the triangular component in the middle (coming up in Chapter 15), the arrangement of the circuit on the page provides some hints. It's a reasonable guess that the device takes input from a microphone, runs it through the mystery component, and then outputs it for some use down the line. The part number on the triangle tells us that it's an amplifier, and that dispels any remaining mystery.

Contrast this with Figure 11-6. In this diagram, the same circuit is organized in a more arbitrary way. At a glance, it's hard to say what the device does and why. Also note the chaos caused by ground and positive supply symbols that are drawn upside down or sideways. In Chapter 7, I advised against the practice; this is why.

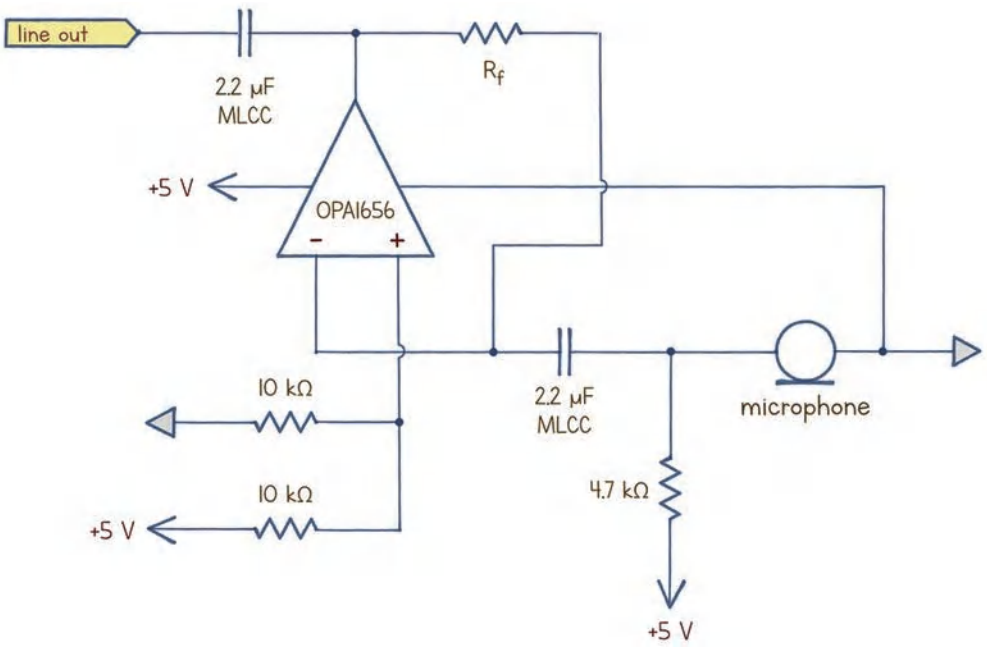


Figure 11-6: The same amplifier circuit as a jumble of wires

Sometimes it's good to break down complex circuits into smaller building blocks. The least invasive way to do it is to surround a section of the circuit with a dashed line and provide an annotation explaining what the section does. A more radical solution is to divide the circuit into separate schematics drawn on separate sheets. This approach should be used judiciously; clutter is bad, but so are schematics where every component occupies its own page, connected only by labels with incomprehensible names.

PART III

EXPLORING ANALOG CIRCUITS

12

ARRANGEMENTS OF PASSIVE COMPONENTS



So far, we've covered the operation of individual electronic components. Next, let's take a closer look at what happens when they're combined in more complex circuits.

We've already done some preparation for this in Part II of the book. In Chapter 5, we derived formulas for the impedance of resistors, capacitors, and inductors connected in series. In Chapter 10, we discussed using resistors to limit the current supplied to diodes. Now it's time to start analyzing circuits in a more systematic way.

In this chapter, we'll start small. We'll outline some useful circuit abstractions and then examine basic combinations of resistors and other passive components. In later chapters, we'll add more variety with diodes, transistors, and integrated chips of all kinds.

A Brief Review of Circuit Theory

Before we start experimenting with practical analog circuits, it's good to recap some of the underlying circuit theory. Rest assured, we won't be introducing any brand-new physics or math. Instead, we'll formalize a couple of intuitive observations about circuit dynamics and give them proper names. We'll also discuss a certain wacky abstraction that isn't necessary to progress through the book but that you'll sooner or later encounter when perusing Wikipedia articles, Stack Exchange threads, or other texts.

Kirchhoff's Circuit Laws

Of all the rules of electronic circuits, Kirchhoff's laws are among the simplest yet most powerful. To introduce the concept, let's circle back to something I've noted on several earlier occasions—that in idealized electronic circuits, what comes in must come out. In other words, electrons don't disappear; they just move back and forth. At any chosen point (*node*) in a circuit, the current flowing in must be equal to the current flowing out.

As a practical illustration, have a look at Figure 12-1. The circuit pictured there consists of a battery supplying some current (I_{supply}) to three parallel branches containing some mystery electronic components. I've also marked a measurement point A that precedes these branches.

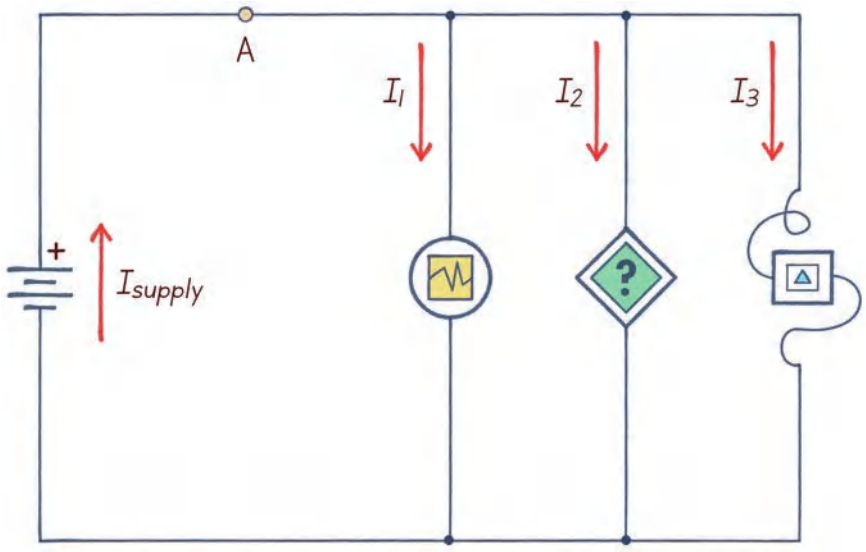


Figure 12-1: The analysis of currents in a mystery circuit

We don't know what currents might be flowing through each of the three branches, but we know that what comes into node A can't be different from what comes out: $I_{supply} = I_1 + I_2 + I_3$. If we move all terms of this equation to the same side, we can rewrite the formula as $I_{supply} - I_1 - I_2 - I_3 = 0$ A. Alternatively, if we give each current a sign depending on its direction (in or out), the equation could be written as $I_{supply} + I_1 + I_2 + I_3 = 0$ A. Whichever convention we follow, the point is the same—the balance of currents flowing into and out of point A must be zero. This is known as *Kirchhoff's current law (KCL)*.

An analogous rule can be formulated for voltages. Figure 12-2 shows another circuit with three mystery components, this time connected in series. I've also included four measurement nodes marked A, B, C, and D, along with voltage differentials between these points.

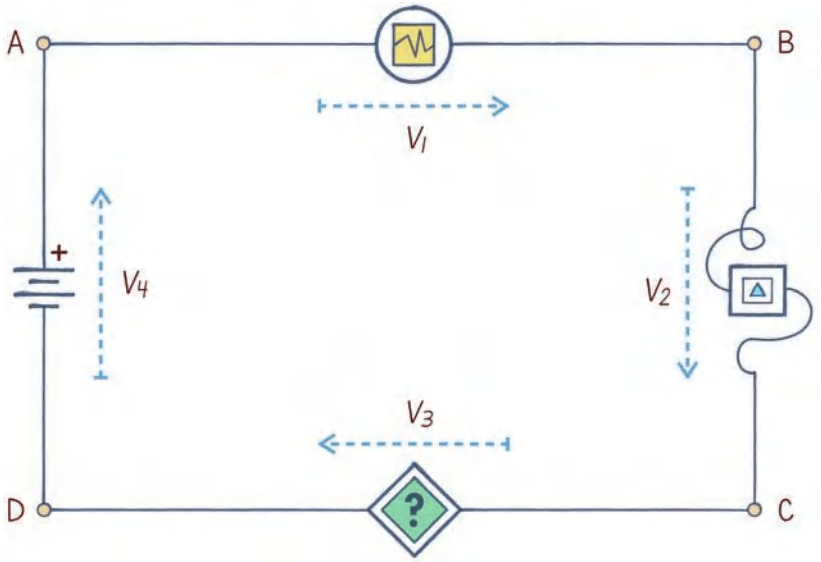


Figure 12-2: Voltages in a circuit with three mystery components

In this circuit, if we connect a voltmeter across the top component—between points A and B—we obtain a measurement of V_1. From there, we can move clockwise and make an incremental measurement of V_2 that represents the electromotive force between points B and C. Based on these two measurements, we can infer the overall electromotive force between points A and C without measuring it explicitly. It's a simple sum: $V_{A\text{-}to\text{-}C} = V_1 + V_2$.

Next, we take another measurement across the bottom component, V_3. As before, we can use this to calculate the voltage between A and D as a sum of all the incremental measurements: $V_{A\text{-}to\text{-}D} = V_1 + V_2 + V_3$. Finally, we measure V_4 and update the sum: $V_{A\text{-}to\text{-}A} = V_1 + V_2 + V_3 + V_4$.

By now, we've come full circle, from point A to point A. The point we're at can't be at two different voltages at the same time, so the voltage difference we calculated, $V_{A\text{-}to\text{-}A}$, is always zero. We must conclude that:

$$V_1 + V_2 + V_3 + V_4 = 0\,\text{V}$$

This is *Kirchhoff's voltage law (KVL)*. It states that the directed sum of voltages in a closed loop is zero. A *closed loop* is any continuous circuit path that gets you from point A back to point A; if the circuit has several parallel branches, you can take any branch you like. A *directed sum* is what we get by traversing nodes in a consistent direction—in Figure 12-2, it means we can't jump from measuring the voltage between A and B (clockwise) to taking a measurement between D and C (counterclockwise).

If either of these laws seems unconvincing, I encourage you to pause and experiment. You can use a 5 to 12 V power supply, and resistors in the 220 Ω to 4.7 kΩ range make good stand-ins for the mystery components. You can also toss a diode into the mix.

Impedance in Series

In Chapter 5, we noted that any number of resistors in series can be replaced by a single equivalent resistance. Each resistor is an obstacle the electrons need to clear, and the effects add up in a straightforward way:

$$R_{total\,(series)} = R_1 + R_2 + R_3 + \ldots$$

In the same chapter, we introduced the closely related concept of reactance (X). Reactance is the tendency for inductors and capacitors to impede the flow of sine-wave currents in a frequency-dependent way. It's measured in ohms and sums the same way as R. For example, for inductors in series, we get:

$$X_{total\,(series)} = X_{L1} + X_{L2} + X_{L3} + \ldots$$

When you reach for an inductor, it won't be labeled by reactance; instead, the component value (L) is specified in henrys (H), with 1 H describing a component that develops 1 V across its terminals when subjected to a current ramping up at 1 A per second. Earlier, we derived a formula that relates these quantities: $X_L = 2\pi fL$. This allows us to rewrite the earlier sum as:

$$2\pi fL_{total\,(series)} = 2\pi fL_1 + 2\pi fL_2 + 2\pi fL_3 + \ldots$$

The constant $2\pi f$ term cancels out, showing that a replacement for several inductors in series is simply a larger inductor:

$$L_{total\,(series)} = L_1 + L_2 + L_3 + \ldots$$

Capacitor reactances sum similarly to inductor reactances, but we've previously established that X_C is inversely proportional to the component's value C (in farads):

$$X_C = \frac{1}{2\pi fC}$$

This means that to replace some number of capacitors in series with a single component, the math is a bit more contrived:

$$\text{Total reactance}: \quad X_{total\,(series)} = X_{C1} + X_{C2} + X_{C3} + \ldots$$

$$\text{Terms substituted}: \quad \frac{1}{\cancel{2\pi f}\,C_{total\,(series)}} = \frac{1}{\cancel{2\pi f}\,C_1} + \frac{1}{\cancel{2\pi f}\,C_2} + \frac{1}{\cancel{2\pi f}\,C_3} + \ldots$$

$$\text{Replacement capacitance}: \quad C_{total\,(series)} = \frac{1}{\frac{1}{C_1} + \frac{1}{C_2} + \frac{1}{C_3} + \ldots}$$

In other words, the equivalent capacitor value is the reciprocal of the sum of reciprocals. Because writing such complex, stacked fractions is tedious, electrical engineers often use a special operator, $\|$ ("parallel sum"), defined this way:

$$a \parallel b \parallel c = \frac{1}{\frac{1}{a} + \frac{1}{b} + \frac{1}{c}}$$

In the case of two identical series capacitors, each with a value C, the calculation works out to $\frac{1}{2}C$.

At first blush, it might be surprising that as we sandwich more capacitors together, the overall capacitance drops. To make sense of this result, keep in mind that in such

a stacked structure, only the outer plates count—that's where the charges flow in and out. Beyond that, we're effectively just adding more dielectric in between. A thicker dielectric layer lowers capacitance, although it improves the ability of this composite part to withstand high voltages. Once again, if the idea gives you pause, it's a good opportunity to grab a multimeter and some capacitors and experiment!

Impedance in Parallel

Now that we know the impedance of passive components in series, let's ponder what happens if we connect the same parts in parallel.

We can start with resistors. In a parallel circuit, each resistor is subjected to the same voltage across its terminals. We can label this voltage V_{supply} and tap into Ohm's law ($I = V / R$) to calculate the current flowing through a chosen resistor branch n:

$$I_n = \frac{V_{supply}}{R_n} = V_{supply} \cdot \frac{1}{R_n}$$

If we factor out the common V_{supply} term, we can write the sum of all branch currents as:

$$I_{total\,(parallel)} = V_{supply} \cdot \left(\frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \dots \right)$$

With this sorted out, we can apply another form of Ohm's law ($R = V / I$) to solve for the equivalent resistance that would admit the same current as our parallel resistors:

$$R_{total\,(parallel)} = \frac{V_{supply}}{I_{total\,(parallel)}} = \frac{\cancel{V_{supply}}}{\cancel{V_{supply}} \cdot \left(\frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \dots \right)} = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \dots}$$

This is another case of the reciprocal of a sum of reciprocals. We can restate the formula for the equivalent resistance of resistors in parallel using the $\|$ operator introduced earlier in this chapter:

$$R_{total\,(parallel)} = R_1 \parallel R_2 \parallel R_3 \parallel \dots$$

In particular, for two identical resistor values R, the equivalent resistance is $\frac{1}{2} \cdot R$.

As before, the reasoning generalizes to parallel reactances. For inductors, we've already established that reactance is directly proportional to component value, so the value for a replacement inductor can be calculated as:

$$L_{total\,(parallel)} = L_1 \parallel L_2 \parallel L_3 \parallel \dots$$

For capacitors, the math is once again reversed because reactance is inversely proportional to component value: $X_C = 1/(2\pi fC)$. In this instance, it works in our favor, as component value substitutions can be calculated as a normal sum:

$$C_{total\,(parallel)} = C_1 + C_2 + C_3 + \dots$$

In line with what we discussed in the preceding section, there's a physical basis for this additive behavior: Multiple capacitors in parallel are electrically equivalent to a single capacitor with a proportionally larger plate surface area and, thus, a greater ability to store charge.

Mixed Impedances

In Chapter 5, we analyzed the behavior of sine-wave signals passing through L-C, R-C, and R-L circuits. For the series L-C case—an inductor and a capacitor—we showed that the resulting impedance equals the difference of the underlying component reactances:

$$Z_{LC\,(series)} = X_L - X_C$$

Some textbooks define the reactance of a capacitor as a negative number: $X_C = -1/(2\pi fC)$. This convention bakes the minus sign into the second operand, so in these contexts the equation for $Z_{LC\,(series)}$ would be written as a straightforward sum.

We'll skip a rigorous derivation of the formula for the parallel L-C circuit; the unsurprising result is just a parallel sum:

$$Z_{LC\,(parallel)} = X_L \parallel (-X_C)$$

Toward the end of Chapter 5, we also looked at the behavior of series R-C and R-L circuits. These cases are more complicated because the voltages that develop across the terminals of each component are 90° out of phase in relation to each other. The signal getting through is still a sine wave, but its amplitude and phase shift require more effort to calculate.

In the aforementioned chapter, we showed that if one variable describes the amplitude of a sine wave with a 0° phase shift and another describes a wave with a 90° shift, the superposition of these signals follows the rules of two-dimensional Cartesian geometry. For a visual reference, see Figure 5-8. In a nutshell, the magnitude of impedance (Z) of the combined waveform can be calculated from the Pythagorean theorem, while the phase shift (θ) is given by a reverse trigonometric function known as arctangent. The geometric approach applies to sine-wave voltage- and current-based signals, as well as to their ratios (impedances).

For a resistance and a reactance connected in series, we established the following equalities:

$$Z_{RC,RL\,(series)} = \sqrt{R^2 + X^2}$$

$$\theta_{RC,RL\,(series)} = \arctan\left(\frac{X}{R}\right)$$

We won't redo the calculations for parallel circuits, but the result is analogous to the cases discussed earlier:

$$Z_{RC,RL\,(parallel)} = \sqrt{\frac{1}{\frac{1}{R^2} + \frac{1}{X^2}}} = \sqrt{(R^2) \parallel (X^2)}$$

$$\theta_{RC,RL\,(parallel)} = \arctan\left(\frac{\frac{1}{X}}{\frac{1}{R}}\right) = \arctan\left(\frac{R}{X}\right)$$

These solutions work for two-component R-C and L-C circuits. In a circuit containing additional passive components, it's preferable to work with independent 0° and 90° sine-wave components for as long as possible, calculating amplitudes and phase shifts only at the end. For example, regardless of the order of the components, the impedance of a series R_1-L_1-R_2-L_2 circuit can be represented as two separate sums: $R = R_1 + R_2$ and $X = X_{L1} + X_{L2}$.



These sums can then be used to find the overall Z and θ:

$$Z_{total} = \sqrt{(R_1 + R_2)^2 + (X_{L1} + X_{L2})^2}$$

$$\theta_{total} = \arctan\left(\frac{X_{L1} + X_{L2}}{R_1 + R_2}\right)$$

The reason this approach works is simple and is illustrated in Figure 12-3.

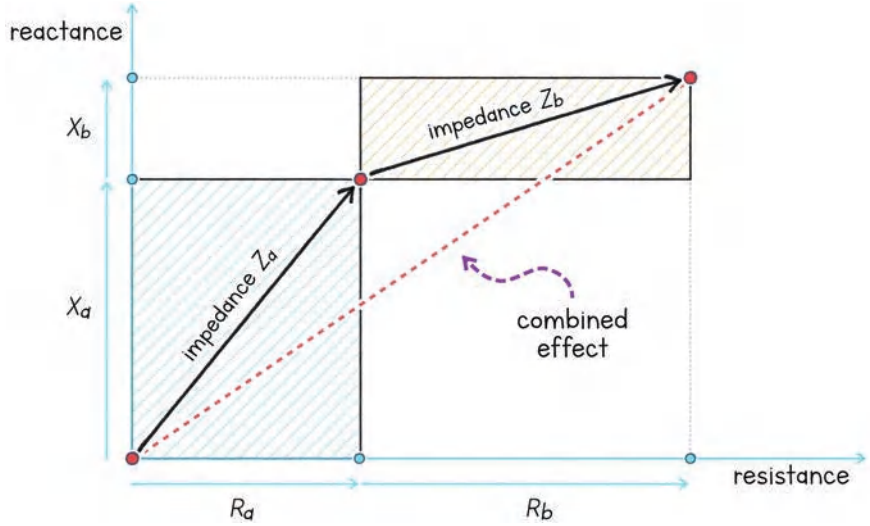


Figure 12-3: A visual explanation for summing impedances

To explain this visual: We've previously established that the impedance of a series circuit with resistance R and reactance X is represented by the diagonal of an R-by-X rectangle on a Cartesian plane. If we're summing two impedances, $Z_a = R_a + X_a$ and $Z_b = R_b + X_b$ (Figure 12-3, arrows), the outcome is the same as the diagonal of a single rectangle with dimensions $R = R_a + R_b$ and $X = X_a + X_b$ (dashed line).

Complex Currents, Voltages, and Impedances

Our discussion of sine-wave dynamics could end here, but let's entertain a wacky thought: What if, instead of tracking the 0° and 90° components separately, we had a way to roll them into a single composite variable? Perhaps we could represent one value as a regular number and couple the other with a made-up symbol, such as 😺. The cat would act as a separator, preventing us from accidentally commingling the cat-free and cat-bearing parts while still letting us perform calculations such as this:

$$\overbrace{(5 + 3😺)}^{\text{operand 1}} + \overbrace{(2 + 1😺)}^{\text{operand 2}} = \overbrace{7 + 4😺}^{\text{result}}$$

With this ace up our sleeve, we could define a cat-composite alternating voltage (symbol: $\tilde{V}$) that describes a superposition of two sinusoidal waveforms offset in phase

by 90°. To give a specific example, a cat-voltage of (3 V) + 😺(4 V) would correspond to a waveform described by the following time-domain formula:

$$V(t) = (3\,\text{V}) \cdot \sin(t) + (4\,\text{V}) \cdot \sin(t + 90°)$$

The rules we developed in Chapter 5 tell us that this signal is equivalent to a single sinusoid with an amplitude of 5 V and a phase shift of about 53°. However, as noted in the previous section, this verbose 0° + 90° representation makes life simpler when doing incremental circuit math.

For impedance, resistive (0°) and reactive (90°) effects already have their own names, so we can write the new cat-composite formula as $\tilde{Z} = R + 😺X$. In mathematics, structures that follow this template are known as *complex numbers*. The cat-free component is called the *real part*, and the cat-coupled number is the *imaginary part*. Somewhat regrettably, electrical engineers don't use the cat symbol—they've settled for a less exciting token called j. This convention means that complex impedance is written as $\tilde{Z} = R + j \cdot X$. Similarly, complex sine-wave voltage can be written as $\tilde{V} = V_{0°} + j \cdot V_{90°}$, while complex sine-wave current can be expressed as $\tilde{I} = I_{0°} + j \cdot I_{90°}$.

NOTE *I'm using tildes to distinguish complex impedances, voltages, and currents from their conventional (scalar) counterparts. Be aware that not all authors make this distinction. Those who do may instead set scalar values apart by writing them as* $|\text{Z}|$, $|\text{V}|$, *and* $|\text{I}|$.

If your head is spinning, don't worry—you don't need complex numbers to understand the rest of the book. That said, the beauty of this notation is that it lets us infer multiplication and division rules that automatically handle phase shifts, allowing us to treat all impedances in the same way, whether they belong to a resistor, an inductor, or a capacitor. In other words, the approach trades some up-front learning effort for reduced mental toil when dealing with analog circuit math. You might find this handy in your adventures down the line.

To figure out the multiplication rules, let's consider what happens if we take a real number represented on the horizontal axis of a Cartesian coordinate plane and multiply it by j. The operation turns the number into a j-coupled imaginary value that belongs on the vertical axis. This means that this multiplication can be geometrically explained as a counterclockwise rotation of 90°. It follows that multiplying by j twice should produce 180° rotation, returning the value to its original axis but reversing its sign. We can express this rule as $x \cdot j^2 = -x$. Equivalently, we can say that $j^2 = -1$ or that $j = \sqrt{-1}$. If this sounds familiar from your earlier mathematical education, that's because the concept of j is the same as the mathematical constant i. Electrical engineers renamed it to avoid confusion with the notation for incremental current, which makes an appearance every now and then.

If multiplication by j results in a counterclockwise rotation, then division by j ought to undo this action by rotating a point clockwise. In a two-dimensional Cartesian view, taking −1 on the horizontal axis and rotating it 90° counterclockwise ($-1 \cdot j$) leads us to the same spot as if we started with +1 and rotated it 90° clockwise ($1 / j$). This gives us another important rule of complex number algebra: $-j = 1 / j$.

NOTE *If you're interested in a more thorough, illustrated introduction to complex numbers, I have a companion article here:* https://lcamtuf.coredump.cx/blog/complex.

We'll do an exercise to show the utility of complex numbers toward the end of the chapter; in the meantime, let's talk about a convenient shorthand associated with this analytical technique.

The idea is that when analyzing circuit behavior, we usually prefer to discuss component values (C, L) rather than their frequency-specific reactances (X_C, X_L). We can

take the conventional formulas developed in Chapter 5 and rewrite them accordingly, but the results are rather unwieldy. For example, for a series R-C circuit, substituting X_C with $1/(2\pi fC)$ produces the following result:

$$Z_{RC\,(series)} = \sqrt{R^2 + X_C^2} = \sqrt{R^2 + \frac{1}{4\pi^2 f^2 C^2}}$$

$$\theta_{RC\,(series)} = \arctan\left(\frac{-X_C}{R}\right) = \arctan\left(\frac{1}{-2\pi fCR}\right)$$

The equivalent complex-number notation for series R-C impedance is still somewhat verbose, but at least it's free of trigonometric functions, exponentiation, and square roots:

$$\tilde{Z}_{RC\,(series)} = R - j \cdot X_C = R + \overbrace{(-j)}^{\text{same as } 1/j} \cdot \frac{1}{2\pi fC} = R + \frac{1}{j \cdot 2\pi fC}$$

The substitution of $-j$ with $1/j$ in this formula follows from the algebraic rules discussed earlier in the section.

This brings us to the promised trick: The notation can be further simplified if we replace the constant $j \cdot 2\pi f$ part with a new symbol, s. The mathematical meaning of this symbol is more complicated than I'm letting on, but for steady sine waves, this substitution is correct. It reduces the earlier formula to:

$$\tilde{Z}_{RC\,(series)} = R + \frac{1}{sC}$$

For a series R-L circuit, the same simplification method nets us:

$$\tilde{Z}_{RL\,(series)} = R + sL$$

Formulas written in this manner are much easier to work with than the unholy messes of their noncomplex equivalents.

Although complex number notation might seem alien, it isn't really saying anything new. It's simply a way of representing data: We can always take complex $\tilde{Z}$, $\tilde{V}$, or $\tilde{I}$, switch back from s to $j \cdot 2\pi f$, and then pluck out the real and imaginary parts. Once that's done, we're free to harness the Pythagorean theorem and the arctangent function to calculate scalar amplitudes and phase shifts the old-fashioned way.

Uses of Resistors

Up to this point, we've discussed circuits in fairly abstract terms. We've devoted many pages to the relationships among voltage, current, resistance, reactance, and impedance. We even covered the whimsical "cat math" and complex numbers. What we haven't talked about yet is what all this theory is good for in real life.

We can now switch gears and finally talk shop. We'll start the journey with some resistor-based circuit building blocks. The resistor is about the simplest electronic component we work with, but it's still a pretty capable multifunction tool.

Current Limiters

Ohm's law tells us that the current flowing through a resistor is proportional to the applied voltage and inversely proportional to the component's resistance: $I = V/R$.

This formula suggests that if we have a standard power supply delivering a constant voltage and want to limit the current supplied to an electrical load, a resistor placed in series with the load ought to do the trick.

At first blush, it seems that the appropriate resistor value can be calculated as:

$$R = \frac{V_{supply}}{I_{limit}}$$

This isn't quite correct, though, and we've already seen why. In Chapter 11, we used a resistor to limit the current flowing through a light-emitting diode. At the time, we also identified a complication with this naive approach to calculating the appropriate resistance: Some loads, including diodes, don't conduct until a nontrivial voltage develops across their terminals.

From KCL, we know that the directed sum of voltages in a closed loop must be zero; in other words, $V_{load} + V_{resistor}$ must equal V_{supply}. It follows that the higher the voltage across the terminals of a load, the lower the electromotive force acting on the resistor, resulting in less current flowing through. To account for this effect, we need to amend the equation to read:

$$R = \frac{V_{supply} - V_{load}}{I_{limit}}$$

If you have a multimeter at your disposal, this behavior is worth examining. You can construct a simple LED circuit with a resistor, measure the voltages across the terminals of each component, and then measure the current by placing the meter in series with the power supply.

Another, more significant limitation of the resistor-based approach is that some loads might develop terminal voltages that change over time. A classic example is a charging capacitor: The voltage across this component gradually increases in proportion to its charge state. It follows that a resistor cannot be used to charge a capacitor at a constant rate. A similar issue arises if the load voltage is constant, but the supply is not. In effect, a resistor can be thought of more as a constant-voltage-to-constant-current converter than as a true constant-current source.

The final issue with resistive current limiters arises from Joule's law, $P = I^2R$. The amount of heat dissipated by a resistor increases in proportion to the square of the admitted current, which makes resistors practical only when the current is relatively low—or when the resistance is tiny enough to drown out the squared term. As discussed in Chapter 10, a resistor used to set the current for an indicator LED is typically no big deal; in a 5 V circuit, it shouldn't dissipate more than about 60 mW. On the flip side, using the same approach to control the speed of a motor is probably a bad idea.

Current Sensors

In the previous section, we talked about using resistors to convert voltages to currents. We can also go the other way around; the most common application of this principle is *current sensing*, for example, to detect motor stalls. If a resistor is placed in the path of an unknown current I, it develops a voltage equal to that current times the component's resistance ($V = IR$). If the current spikes—say, because the motor is stuck—the voltage across the resistor's terminals rises accordingly.

For this technique to work, the resistance must be small enough not to interfere with the operation of the monitored current path. As a practical example, a 10 kΩ current-sensing resistor placed in series with a motor running on a 12 V supply isn't

going to work: The maximum current the supply can push through such a resistor is 1.2 mA, which isn't enough for the motor to move. A better choice would be a 100 mΩ resistor, which can handle several amps without breaking a sweat.

The trade-off is that small resistances make precise measurements more challenging. A current of 1 mA results in a healthy 1 V reading across the terminals of a 1 kΩ resistor, but only an anemic 100 µV across a 100 mΩ one. Precision or current capacity—with this approach, it's hard to have both.

Voltage Dividers

When two resistors are placed in series, their equivalent resistance is $R_{total} = R_1 + R_2$. If this series circuit is then connected to the terminals of a voltage supply, the resulting current will be:

$$I = \frac{V_{supply}}{R_{total}} = \frac{V_{supply}}{R_1 + R_2}$$

The same current flows through R_1 and R_2; across the terminals of each resistor, it induces a voltage proportional to the component's value. We can designate these voltages as V_{R1} and V_{R2}. Solving for the voltage induced by this current as it flows across R_2, we get:

$$V_{R2} = I \cdot R_2 = V_{supply} \cdot \frac{R_2}{R_1 + R_2}$$

A visual representation of this series circuit is shown in Figure 12-4. Note that the bottom terminal of R_2 is tied to ground, so the voltage at the labeled output node between the resistors must be equal to 0 V + V_{R2}. Because of this, the circuit functions as a voltage divider: V_{out} is equal to V_{supply} times the ratio of R_2 to $R_1 + R_2$. In particular, if $R_1 = R_2$, the output works out to $\frac{1}{2} \cdot V_{supply}$.

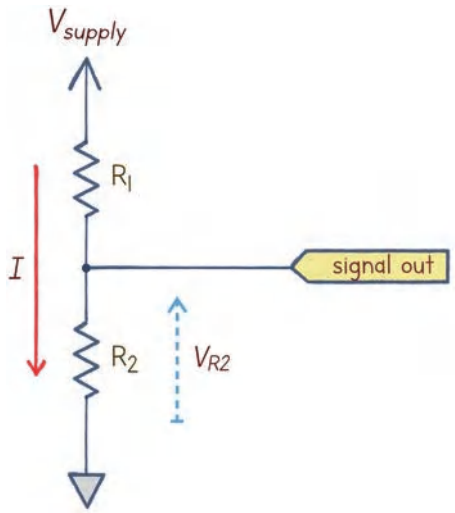


Figure 12-4: A simple voltage divider

Voltage divider circuits are a common way to produce reasonably stable reference signals for a variety of applications. For example, the output from a divider can be compared to the output from a light sensor to decide if it's night or day. Alternatively, a resistive light sensor (LDR; Chapter 8) can be integrated right into a voltage divider, taking the place of R_1 or R_2. This arrangement would produce an output voltage that

changes in response to the amount of incident light. It's a good experiment to try if you have an LDR on hand.

As with the resistor-based current limiter, the approach is simple but has some caveats. First, the output voltage is always relative to the supply voltage; if the supply fluctuates, the output of the divider will move in unison. Another constraint is that for the circuit to work correctly, there must be no significant current flowing through the output terminal. If this assumption is violated, we end up with a circuit that might look like what's shown in Figure 12-5.

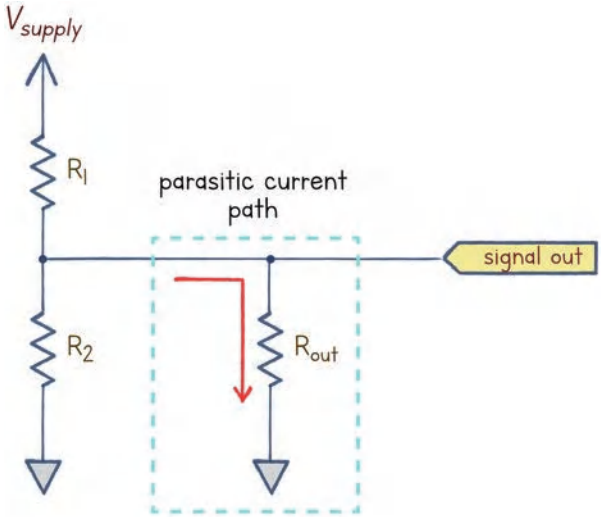


Figure 12-5: A voltage divider with a low-impedance load modeled as a parasitic current path to ground

This revised circuit is also a voltage divider—albeit not the one we might have intended to build. The example shows two parallel resistances to ground, so in place of R_2, we need to consider an equivalent resistance $R_{equiv} = R_2 \parallel R_{out}$.

In practical applications, the circuit is typically constructed with resistances between 1 kΩ and 100 kΩ to keep the waste R_1-R_2 current low. On the flip side, there's a limit to how far we can go: To make the impedance connected to the output leg safe to ignore, it must be about two orders of magnitude higher than $R_1 + R_2$.

Analog Voltage Shifters

The voltage divider approach can be generalized to situations where a pair of resistors is connected to a signal other than the supply rail. To analyze this situation, let's examine the circuit shown in Figure 12-6. It consists of one resistor in series with an input signal, followed by another resistor connected between the signal path and a constant voltage of our choice. Again, it can be thought of as a generalization of the earlier voltage divider—we've simply replaced 0 V on the bottom leg of R_2 with an arbitrary waveform.

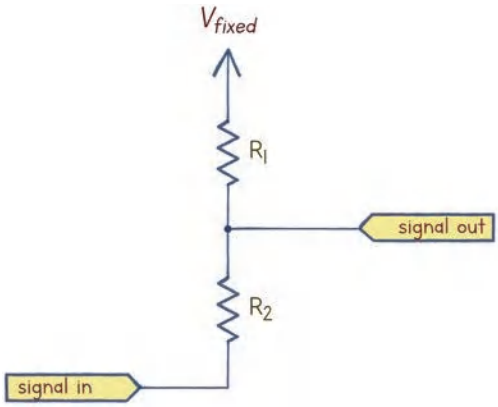


Figure 12-6: A signal-shifting circuit

Analogously to what we've discussed before, the current flowing through the R_1-R_2 path depends on the voltage difference between the input signal and the fixed reference point. The input signal may change over time, but if we take a snapshot, the momentary current equation can be written as:

$$I = \frac{V_{fixed} - V_{signal}}{R_1 + R_2}$$

From this, we can calculate the voltage that appears across resistor R_2:

$$V_{R2} = I \cdot R_2 = \frac{(V_{fixed} - V_{signal}) \cdot R_2}{R_1 + R_2} = \frac{V_{fixed} \cdot R_2 - V_{signal} \cdot R_2}{R_1 + R_2}$$

Resistor R_2 straddles the input and output terminals, so this formula gives only the relative voltage between these two points. To get a value referenced to the circuit's ground, we need to take V_{signal} and add V_{R2}:

$$V_{out} = V_{signal} + V_{R2} = V_{signal} + \frac{V_{fixed} \cdot R_2 - V_{signal} \cdot R_2}{R_1 + R_2}$$

The first term, V_{signal}, can be rewritten as a fraction that uses the same $R_1 + R_2$ denominator as the second part:

$$V_{out} = \underbrace{\frac{V_{signal} \cdot (R_1 + R_2)}{R_1 + R_2}}_{= V_{signal}} + \frac{V_{fixed} \cdot R_2 - V_{signal} \cdot R_2}{R_1 + R_2}$$

This substitution allows us to cancel two terms in the resulting sum:

$$V_{out} = \frac{V_{signal} \cdot R_1 + \cancel{V_{signal} \cdot R_2}}{R_1 + R_2} + \frac{V_{fixed} \cdot R_2 - \cancel{V_{signal} \cdot R_2}}{R_1 + R_2}$$

$$= \frac{V_{signal} \cdot R_1}{R_1 + R_2} + \frac{V_{fixed} \cdot R_2}{R_1 + R_2}$$

The equation further generalizes to any two signals fed into such a divider. The general rule can be stated as:

$$\text{<output>} = \frac{\text{<}R_1\text{-side input>} \cdot R_2}{R_1 + R_2} + \frac{\text{<}R_2\text{-side input>} \cdot R_1}{R_1 + R_2}$$

In particular, if $R_1 = R_2$, the output is just a simple average of the two input voltages.

Averaging works such that if we take, for example, a signal that swings between −1 V and +1 V and combine it with a fixed reference at 5 V, we'll get a new signal with a lower bound of (−1 V + 5 V) / 2 = 2 V and an upper bound of (1 V + 5 V) / 2 = 3 V. The signal is shifted into positive voltages, which is usually what we want. This is why we think of this type of circuit as a *voltage shifter*.

On the other hand, the amplitude of the voltage swing in our example is reduced from 2 V to 1 V, which is usually something we don't want. Indeed, the more we skew the average toward the fixed supply voltage—that is, the higher the value of R_2 in relation to R_1—the more we attenuate the swings of the original waveform. The signal can be amplified down the line to make up for the losses, but in the analog realm the process is never free: We sacrifice some linearity and introduce noise.

Pull-Ups and Pull-Downs

Some types of input devices pose an interesting challenge to field-effect transistors: Components such as switches and relays provide a low-impedance electrical path in one operating position but leave the circuit open in another. Pull-up and pull-down resistors are a simple way to address this challenge.

To illustrate the potential problem, consider the circuit shown in the left panel of Figure 12-7. If the switch is closed, the gate of the MOSFET is connected to the positive supply rail; the gate becomes positively charged, and the transistor begins to conduct. If we open the switch, the gate doesn't get discharged. Instead, it's left floating in an indeterminate state. The transistor might continue to conduct for a while due to the gate capacitance, as in the touch-sensing circuit in Chapter 10.

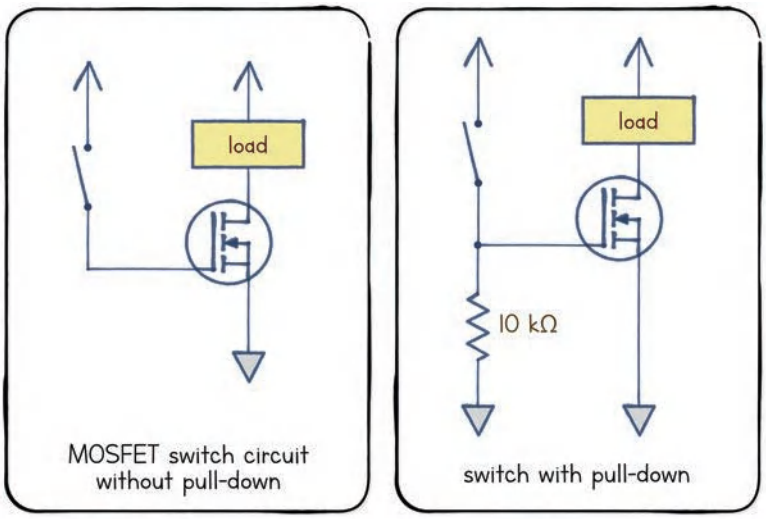


Figure 12-7: Using an n-channel MOSFET to switch a load. Left: no pull-down. Right: a better approach with a pull-down resistor.

To fix this problem, we can add a *pull-down resistor* (Figure 12-7, right). In the "on" position of the switch, the resistor has no appreciable effect on the gate voltage because its impedance is much higher than that of a closed switch. This is essentially a degenerate form of a voltage divider where R_1 (the switch) ≈ 0 Ω and R_2 = 10 kΩ, so per our earlier analysis, $V_{out} \approx V_{supply}$. When the switch is in the "off" position, in contrast, the resistor becomes the lowest-impedance path; the accumulated charge bleeds off to ground, and the transistor turns off properly.

A complementary arrangement, in which the resistor is connected to the positive supply rail and the switch is wired to ground, is called a *pull-up*. It's used whenever we want the "off" voltage to be high.

Uses of Capacitors

Compared to resistor-based circuits, clever arrangements of capacitors alone are relatively rare. In theory, resistors and capacitors can be used for signal processing in nearly the same way: A capacitor, when subjected to a sinusoidal waveform, exhibits frequency-specific resistance-like behavior—reactance—so in some circuits, a C_1-C_2 voltage divider could work similarly to its more conventional R_1-R_2 counterpart. That said, we usually don't go there; resistors work better and are more predictable.

It's far more common to see a capacitor combined with a resistor or an inductor to form a circuit that attenuates signals in a frequency-dependent way. This is particularly useful for constructing filters. We modeled the fundamentals of series R-C, L-C, and R-L arrangements in Chapter 5, and we'll circle back to their practicalities soon.

Outside of signal filtering, the primary use of capacitors is energy storage. In this role, a capacitor can smooth out power supply fluctuations or provide short-term backup power. More intriguingly, the component's ability to hold charge can be exploited to build simple timers and oscillators. In Chapter 5, we stated that the voltage across the terminals of a capacitor changes in relation to the charging current and time ($\Delta v = I \cdot t/C$). It follows that if C is known and I is controlled in some fashion, the capacitor's voltage keeps track of time.

If we start with a discharged capacitor and supply a constant current, the math is trivial. We can calculate the elapsed charging time simply by measuring the voltage across the component's terminals, multiplying it by the component's value C, and dividing by I. However, in most cases we don't have the luxury of a true constant-current supply. Let's figure out what happens in the more common case: a voltage supply connected through a resistance.

The Dark Art of Charging a Capacitor

Earlier in the chapter, I noted that resistors in tandem with voltage sources can be used as current limiters, but that charging a capacitor is a rare exception to the rule. This is because the voltage across the terminals of a capacitor rises in proportion to its charge state; from Kirchhoff's and Ohm's laws, this necessarily causes the resistor current to taper off over time.

Capacitors charged this way are ubiquitous, so it's good to have a model of the behavior at hand. We can start with a discrete-time simulation, illustrated in Figure 12-8. For simplicity, let's assume C = 1 F and R = 1 Ω, and choose a supply voltage of 1 V. These values are somewhat atypical but not impossible; I'm choosing them to reduce the amount of unit-wrangling down the line.

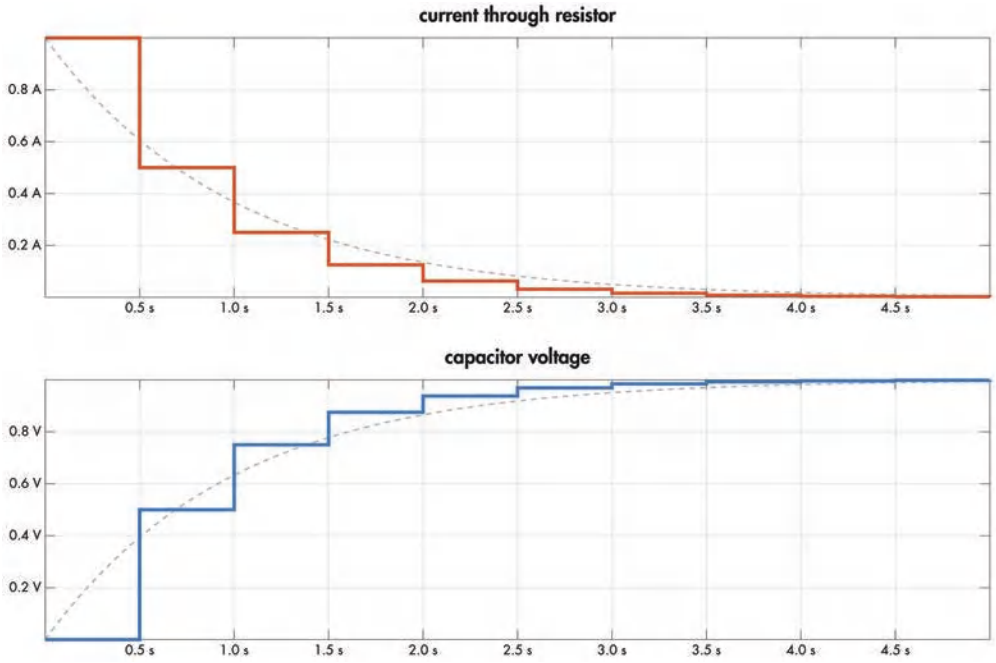


Figure 12-8: A discrete-time simulation of a charging capacitor. The dashed lines in the background show a more precise solution calculated with a finer time step.

In Figure 12-8, I've run the simulation with a relatively large time step of 500 milliseconds; this makes the results somewhat imprecise, but it helps illustrate an important point. At the beginning of the first discrete time interval, we have a fully discharged capacitor, so the current flowing through the resistor is $I = 1\text{ V} / 1\ \Omega = 1\text{ A}$. From the capacitor equation, this current, if flowing for 500 ms, must increase the capacitor voltage by $\Delta v = 1\text{ A} \cdot 500\text{ ms} / 1\text{ F} = 500\text{ mV}$.

NOTE *Volts appear in the solution as a consequence of the basic definitions of units: An ampere is equal to 1 coulomb of charge per second (Chapter 3); a farad is equal to 1 coulomb per volt (Chapter 5). After these substitutions, every unit except the volt cancels out.*

At the beginning of the next time slice, the capacitor is already charged to 500 mV, so the resistor is exposed to just half the previous voltage differential; from Ohm's law, the resulting current is therefore halved to 500 mA. This charging current increases the capacitor voltage by $\Delta v = 250$ mV, adding up to a total of 750 mV. The process continues: With every passing 500 ms, the charging current and the resulting voltage gain are reduced by 50 percent.

This pattern of repeated halving is known as *exponential decay*. The current flowing through our circuit starts at 1 A, so after the nth halving, it can be described by a very simple formula:

$$I_{cap}(n) = \frac{1}{2^n} = 2^{-n} \quad \text{(units: A)}$$

The capacitor voltage curve is just a mirror image of the current plot, starting at 0 V and then inching toward V_{supply}. In our 1 V circuit, it can be modeled as:

$$V_{cap}(n) = 1 - 2^{-n} \quad \text{(units: V)}$$

The problem with these equations is that they describe what happens at halving intervals but don't tell us how long each halving is supposed to take. From Ohm's law

and the capacitor equation, we can reason that the charging time must be directly proportional to R and C: One reduces the charging current and the other increases capacity. With this in mind, we can start our quest for a better formula by renaming n to t to signify time in seconds and then dividing it by $R \cdot C$. The tentative current equation is:

$$I_{cap}(t) = 2^{-\frac{t}{RC}} \quad \text{(units: A)}$$

That said, in our particular thought experiment, C = 1 F and R = 1 Ω, so the division doesn't change much. A higher-resolution discrete-time simulation (dashed line in Figure 12-8) reveals that the actual current-halving time is about 693 ms. Meanwhile, because our candidate equation uses base 2 for the exponentiation, it predicts halving at 1-second intervals ($2^{-0} = 1$, $2^{-1} = 0.5$, $2^{-2} = 0.25$, and so on). The discrepancy can be fixed by incorporating an ad hoc fudge factor, but it would be better to know where the error is coming from.

To diagnose the problem, we can make a well-proportioned plot that shows a high-resolution discrete-time simulation next to a plot of our candidate function $I_{cap}(t)$ (Figure 12-9). The simulation shows that at the beginning of the properly simulated charging process, the initial downward slope for the current curve is very close to 45°. In contrast, the slope of the candidate function is about 35°.

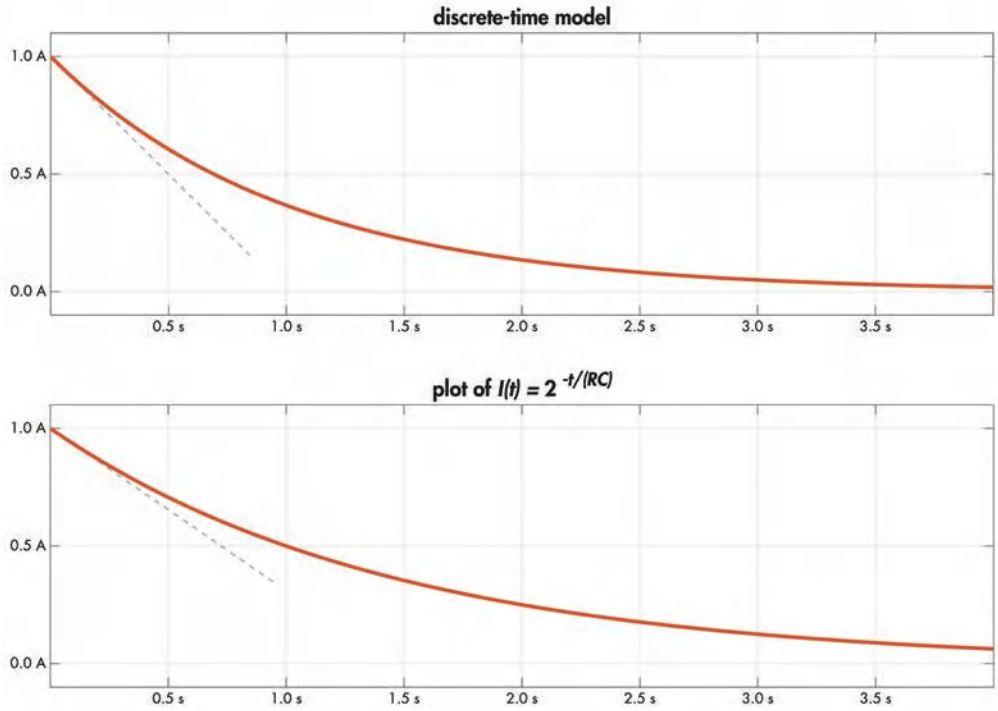


Figure 12-9: Comparing two models of capacitor charging current: a discrete-time simulation (top) and a candidate function that doesn't look quite right (bottom)

It's possible to show that the correct slope must be exactly 45°; to do this, we can use the technique of minuscule time deltas, an approach we perfected in Chapter 4. From Ohm's law, we've already established that in a circuit characterized by R = 1 Ω and V_{supply} = 1 V, the initial charging current must be $I_{cap}(0)$ = 1 A. Further,

by applying the capacitor equation ($V_{cap} = I \cdot t/C$), we can show that within the next microscopic time slice Δt, the current will charge a 1 F capacitor to:

$$V_{cap}(0 + \Delta t) = \frac{I_{cap}(0) \cdot \Delta t}{C} = \frac{1\,\text{A}}{1\,\text{F}} \cdot \Delta t = 1\,\text{V/s} \cdot \Delta t$$

This allows us to calculate the rate of change between $t = 0$ and $t = 0 + \Delta t$. After a bit of tidying up, we get an unsurprising result of 1 volt per second:

$$r(0) = \frac{V_{cap}(0 + \Delta t) - V_{cap}(0)}{\Delta t} = \frac{1\,\text{V/s} \cdot \cancel{\Delta t}}{\cancel{\Delta t}} - \frac{0\,\text{V}}{\cancel{\Delta t}} = 1\,\text{V/s}$$

On a well-proportioned plot, a function that increases by 1 volt per second will exhibit an upward 45° slope. We know that the curve for current is a mirror image of the curve for voltage, so if V is pointing up, then the plot of I should be starting with a matching downward slope, as we saw in the top panel of Figure 12-9.

There's only one exponent base that naturally produces a 45° slope at zero crossing. It's not the base 2 we've chosen for $I_{cap}(t)$. It is, by definition, the mathematical constant $e \approx 2.71828$. This realization allows us to construct a complete current equation that accurately describes gradual decay from the initial inrush current of $I_{cap}(0) = V_{supply}/R$:

$$I_{cap}(t) = \frac{V_{supply}}{R} \cdot e^{-\frac{t}{RC}}$$

In the same vein, we can construct a voltage equation that slowly approaches the supply voltage:

$$V_{cap}(t) = V_{supply} \cdot \left(1 - e^{-\frac{t}{RC}}\right)$$

Note the structure of the timing expression; the relative charge state of the capacitor is dictated only by the ratio of t to $R \cdot C$ (the latter value is sometimes called the *RC time constant*). This lends itself to some useful rules of thumb. At $t = 0.7RC$, the capacitor is roughly 50 percent charged ($1 - e^{-0.7} \approx 0.5034$); at $t = 3RC$, the charge state is about 95 percent ($1 - e^{-3} \approx 0.9502$); and at $t = 5RC$, we hit over 99 percent ($1 - e^{-5} \approx 0.9933$).

These reference points are worth keeping in the back of your head. For example, if you need to figure out how long it would take to charge practically fully a 470 µF capacitor from a 9 V alkaline battery (internal resistance around 1.7 Ω), the answer is $t = 5RC = 5 \cdot 1.7\,\Omega \cdot 470\,\mu\text{F} \approx 4$ milliseconds. Blink and you'll miss it!

R-C Low-Pass and High-Pass Filters

In signal processing, the two most fundamental resistor-capacitor circuits are the low-pass filter and the high-pass filter. Such filters are particularly common in audio and radio circuits, where they're used to pluck out interesting frequencies and attenuate everything else. The simplest implementations of these architectures are shown in Figure 12-10.

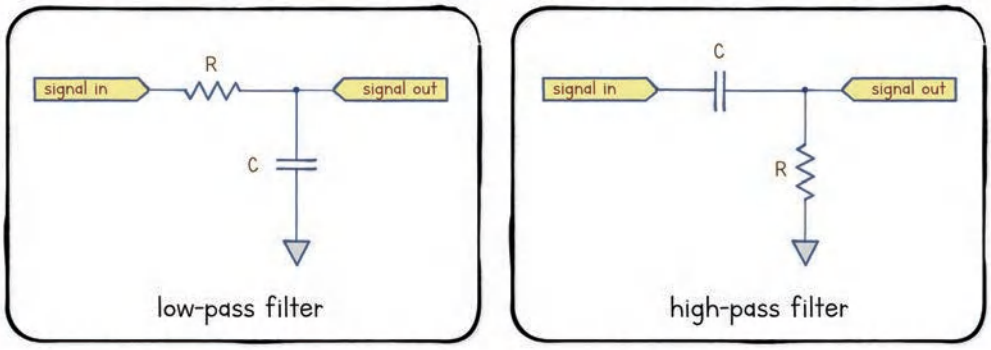


Figure 12-10: Single-stage R-C low-pass and high-pass filters

If we stick to the realm of sine waves, both of these circuits can be considered an extension of a resistor-based voltage divider (Figure 12-4). In a resistor circuit, the output voltage is determined by the ratio of the bottom resistance to the summed resistance along the path to ground:

$$V_{out} = V_{in} \cdot \frac{R_2}{R_1 + R_2}$$

In an R-C filter, one of those resistors is replaced with a capacitor. Let's consider what effect this has, starting with the low-pass case.

Low-Pass Voltage Gain

In an R-C low-pass filter, the input resistor R takes the place of R_1 (connecting to a signal source) and C takes the place of R_2 (connecting to ground). The only complication is that in the conventional real-numbers view, we can't directly sum the impedances of R and C because their effects aren't aligned in phase. That said, from Chapter 5, we already know how to compute the combined impedance of a series R-C signal path:

$$Z_{RC\,(series)} = \sqrt{R^2 + X_C^2} = \sqrt{R^2 + \frac{1}{(2\pi fC)^2}}$$

If we plug this result and the formula for capacitor reactance into the voltage divider equation, we get:

$$V_{out} = V_{in} \cdot \frac{X_C}{Z_{RC\,(series)}} = V_{in} \cdot \frac{\frac{1}{2\pi fC}}{\sqrt{R^2 + \frac{1}{(2\pi fC)^2}}} = V_{in} \cdot \frac{1}{2\pi fC \cdot \sqrt{R^2 + \frac{1}{(2\pi fC)^2}}}$$

To clean this up, we can move the $2\pi fC$ multiplier under the square root. This requires squaring the value beforehand:

$$V_{out} = V_{in} \cdot \frac{1}{\sqrt{(2\pi fC)^2 \cdot \left(R^2 + \frac{1}{(2\pi fC)^2}\right)}} = V_{in} \cdot \frac{1}{\sqrt{(2\pi fRC)^2 + 1}}$$

We can also move V_{in} to the left to get a formula for the ratio of V_{out} / V_{in}. This effectively gives us a nifty way to describe the circuit's gain at some chosen frequency f:

$$G(f) = \frac{V_{out}}{V_{in}} = \frac{1}{\sqrt{(2\pi fRC)^2 + 1}}$$

The log-scale plot of this function across a range of frequencies will depend on the values of R and C, but it will always look similar to the example shown in Figure 12-11.

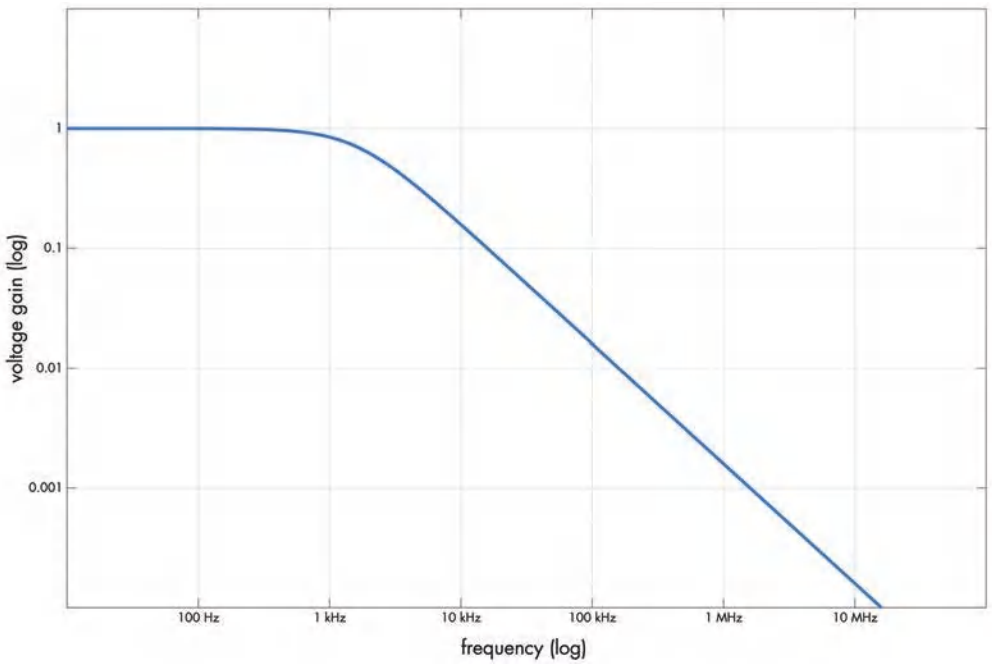


Figure 12-11: R-C low-pass filter gain for C *= 10 nF and* R *= 10 kΩ*

The knee of the plot—the midpoint between the horizontal part of the plot and the diagonal slope—marks the point where R and X_C are in balance. To the left of this frequency, the capacitor has much higher impedance than the resistor, so the current flowing to ground is negligible and the output voltage is close to V_{out}. To the right of the knee, the situation reverses, and the capacitor becomes a comparatively low-impedance path. This produces a downward 45° slope because the current flowing through the input resistor can't keep up and the voltage is pulled ever more strongly to ground.

Once again, we solved for the midpoint $R = X_C$ frequency of a series R-C circuit in Chapter 5:

$$f_{knee} = \frac{1}{2\pi RC}$$

We can also use the gain formula to calculate the relative amplitude of the output signal at this frequency:

$$G(f_{knee}) = \frac{1}{\sqrt{(2\pi f_{knee}RC)^2 + 1}} = \frac{1}{\sqrt{\left(2\pi \frac{1}{2\pi RC} RC\right)^2 + 1}} = \frac{1}{\sqrt{2}} \approx 0.707$$

The midpoint frequency is sometimes also referred to as the center frequency,

the corner frequency, the cutoff frequency, or the −3 dB frequency, which is close enough in decibel math:

$$\left(\sqrt{10}\right)^{-3/10} \approx 0.708$$

Be aware that the latter two naming conventions can be confusing when talking about higher-order filters with steeper response curves, so I recommend using the terminology with care; many online sources mix up these concepts. Steeper-response filters are not hard to build, and we'll look at one example very soon.

High-Pass Voltage Gain

The behavior of a high-pass filter (second panel, Figure 12-10) is analogous to that of a low-pass filter, except it works in reverse: The filter attenuates lower frequencies while passing high-frequency signals through. The gain function is calculated in a very similar way, except that we start with R instead of X_C in the numerator:

$$G(f) = \frac{R}{Z_{RC\,(series)}} = \frac{R}{\sqrt{R^2 + \frac{1}{(2\pi fC)^2}}}$$

Figure 12-12 shows an example plot of this circuit's voltage gain.

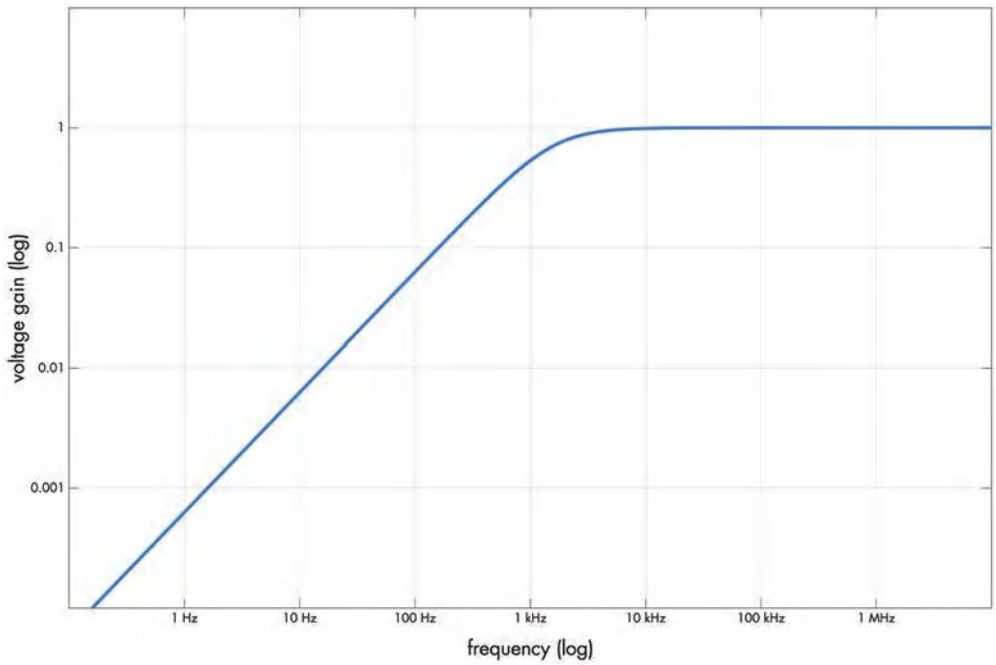


Figure 12-12: R-C high-pass filter gain for C *= 10 nF and* R *= 10 kΩ*

In addition to its obvious applications in signal processing, a degenerate form of an R-C high-pass filter is often used when we want to block steady currents but let AC signals pass. To do this, we place a reasonably large capacitor in series with the signal and let the attached load serve as the R leg of the R-C circuit. This is called *AC coupling* and is particularly common in audio. Speakers produce sound only when they move, so steady currents accomplish nothing except heating the room (and potentially burning out the speaker coil). A series capacitor solves this problem.

It's worth noting that if there's no load whatsoever connected to a series capacitor, the component can't charge or discharge. Instead, it holds a constant voltage

across its terminals—which can be zero but might be something else if it happened to be charged beforehand. In this mode, the capacitor acts as a conduit for electromotive force, not a frequency-dependent filter of any kind.

Signal Phase Shift

The output signal produced by low-pass and high-pass filter circuits is always shifted in phase in relation to the input voltage. In Chapter 5, we talked about the current-versus-voltage shifts in R-C circuits. In a series R-C arrangement, the voltage always lags in relation to the current, akin to having a negative offset in the sin(...) timing expression that describes the voltage waveform. Earlier in the book, we calculated this voltage offset as:

$$\theta_{V\text{-}relative\text{-}to\text{-}I} = \arctan\left(\frac{X}{R}\right) = \arctan\left(\frac{-X_C}{R}\right) = -\arctan\left(\frac{1}{2\pi fRC}\right)$$

In this instance, though, it's more convenient to take the phase of the input voltage as the reference point; in this view, it's the current that's running "ahead" of the input sine. To get the offset for the current timing expression, we just need to flip the sign in the previous formula:

$$\theta_{I\text{-}relative\text{-}to\text{-}V} = -\theta_{V\text{-}relative\text{-}to\text{-}I} = \arctan\left(\frac{1}{2\pi fRC}\right)$$

So far, all we have is a way to assert that in an R-C filter, the common current flowing through the R-C path to ground is offset in phase from the voltage-based input waveform. However, in R-C low-pass and high-pass circuits, we don't take currents as the output signal; instead, we're interested in the voltage on the output pin.

The solution for the high-pass case is simple: The common current, offset in phase from the input sine by $\theta_{I\text{-}relative\text{-}to\text{-}V}$, flows through a resistor that straddles ground and the output pin. The voltage that develops across this resistor is proportional to the current and stays in phase with it. In other words, the formula for the signal phase shift in a high-pass filter is simply:

$$\theta_{RC\,(high\text{-}pass)} = \arctan\left(\frac{1}{2\pi fRC}\right)$$

The case of a low-pass filter is a bit more nuanced because we now have a capacitor in place of the resistor. As discussed in Chapter 5, it's a fundamental property of capacitors that the voltage that develops across their terminals is always one-fourth of a cycle (90°) behind the applied sinusoidal current. This means that the overall result of these shifts—first $\theta_{I\text{-}relative\text{-}to\text{-}V}$ in one direction, and then 90° back—is:

$$\theta_{RC\,(low\text{-}pass)} = \theta_{I\text{-}relative\text{-}to\text{-}V} - 90^\circ = \arctan\left(\frac{1}{2\pi fRC}\right) - 90^\circ$$

From basic trigonometry (or a look-up on Wikipedia), we know that $\arctan(x) - 90^\circ$ is the same as $-\arctan(1/x)$. We can thus simplify the formula to:

$$\theta_{RC\,(low\text{-}pass)} = -\arctan(2\pi fRC)$$

A visual summary of these shifts is shown in Figure 12-13.

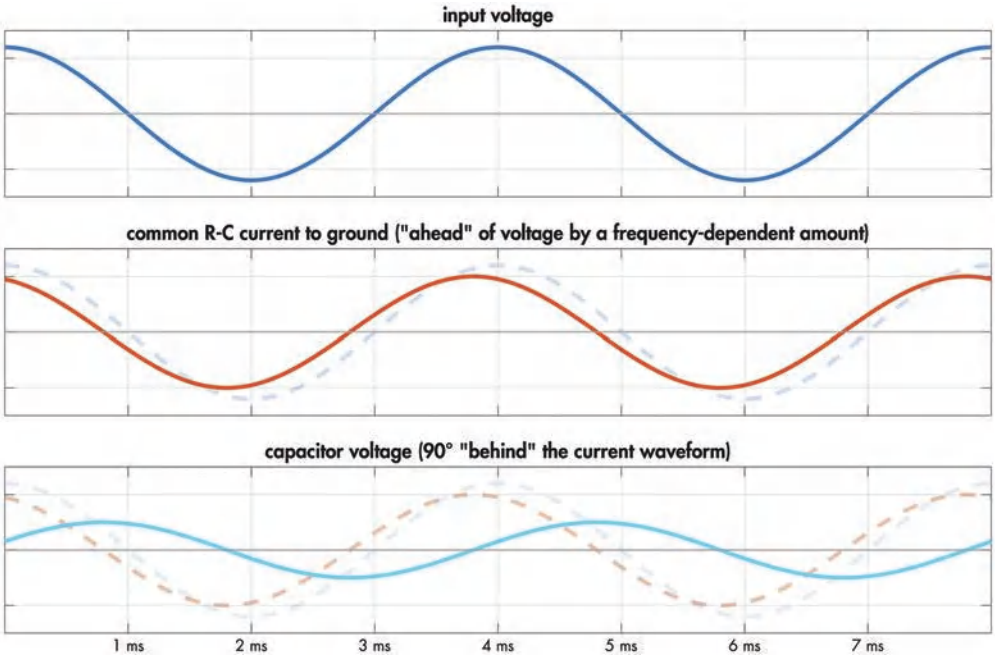


Figure 12-13: A conceptual illustration of R-C low-pass phase offsets

We can also plot the filters' phase shifts vis-à-vis their gain curves (Figure 12-14). In a low-pass filter, the phase of the output signal lags behind the input sinusoid; in a high-pass filter, it appears to be running ahead. In both cases, the shift is relatively modest in the "pass" region of the filter and then gradually increases toward plus or minus 90° as the output voltage drops.

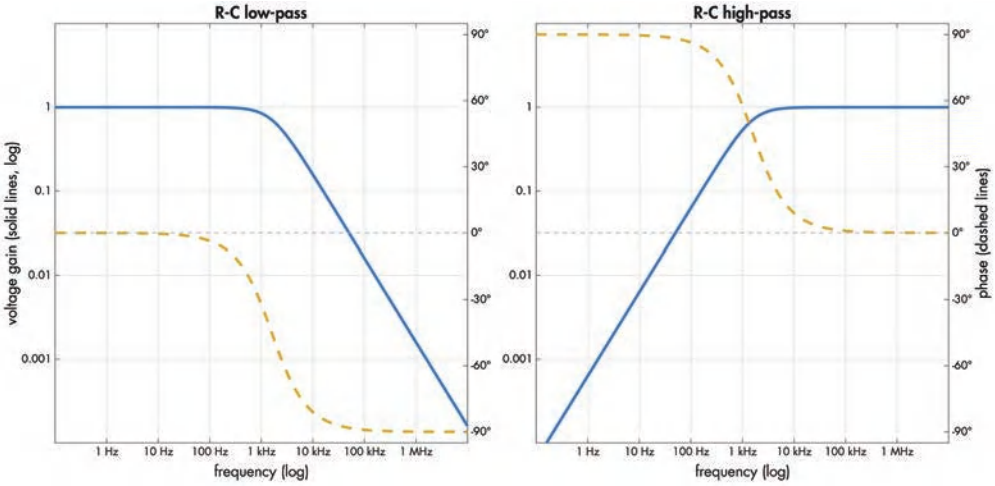


Figure 12-14: Gain and phase shifts in R-C filters, plotted for C = 10 nF, R = 10 kΩ

For the willing reader: The midpoint frequency f_{knee} corresponds to the point where $R = X_C$. Can you show why, at this frequency, the phase offset of the filters is exactly −45° / +45°?

R-C Filters with Non-Sine Waveforms

Before we wrap up, it bears repeating that the abstractions we've built for R-C filters work only for sine waves. Figure 12-15 shows a discrete-time model of what happens if the same filters are supplied with a square wave.

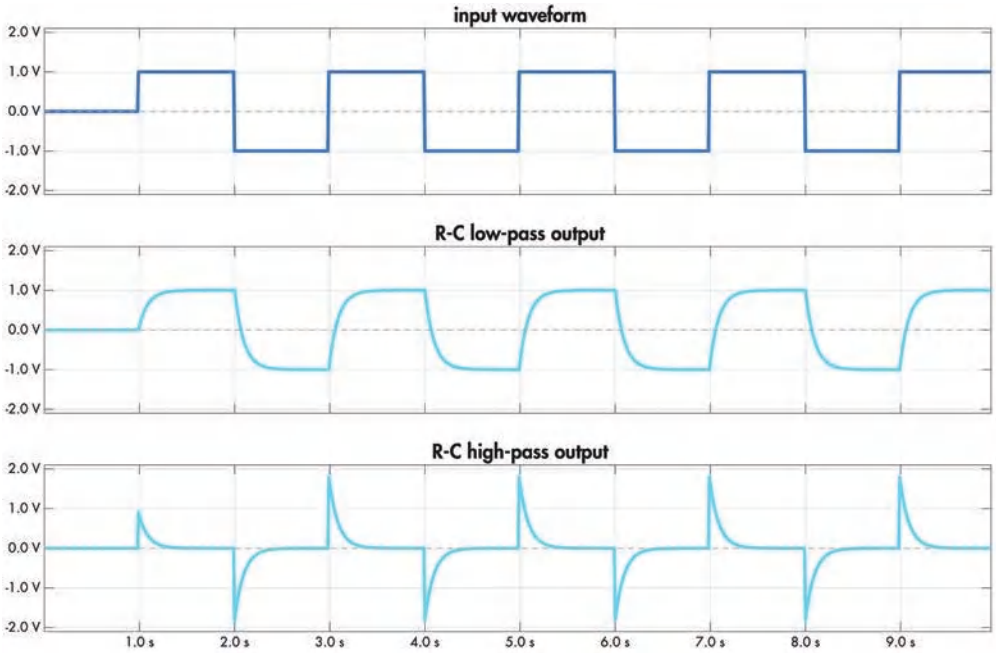


Figure 12-15: A simulation of the behavior of R-C low-pass and high-pass filters for C = 10 µF and R = 10 kΩ

The low-pass filter's behavior mimics the constant-voltage charging and discharging curves that we discussed earlier in the chapter. With a much higher input frequency, the capacitor wouldn't have time to charge fully and reach the per-cycle plateau, so the pattern would be closer to a slightly rounded triangle wave. If you have a signal generator and an oscilloscope at hand, it's interesting to experiment with how the shape changes. For selecting the right components, remember that it takes roughly $t = 5RC$ for a capacitor to reach a 99 percent charge state.

The case of the high-pass filter is more intriguing. A jump in the voltage on one terminal induces a sympathetic voltage on the other plate. The capacitor has no way to charge or discharge as quickly as the input voltage is changing, so for a moment it acts only as a conduit for electromotive force, the voltages across its terminals changing in unison. After that, a more gradual charging or discharging process can happen via the resistor; the combined effect is an interesting pattern of spikes followed by exponential decay. Once again, this is a worthwhile experiment to try at home.

Knowing these patterns by heart is useful, as they often alert you to specific problems in your designs. For example, a broken wire or a bad PCB joint will act as a tiny series capacitor; if you see a pattern of spikes where you ought to be seeing a square wave, that's what might be going on.

Another Visit to the Complex-Verse

We're done with the analysis of two-component R-C filters, but as an optional exercise we can model the behavior of the filter in a different way. In essence, we'll revisit the complex-number approach discussed earlier in the chapter and see if it's as useful as they say.

In the realm of complex numbers, the impedance of a resistor is the same as before: R. The impedance of a capacitor, in contrast, can be represented by an imaginary number, $1/(sC)$. In this notation, the imaginary unit j is rolled into the symbol s ($s = j \cdot 2\pi f$).

Because the imaginary unit keeps the values separated and automatically accounts for the 90° phase shift, we can add impedances without having to calculate squares and square roots immediately. In the complex-verse, capacitors, inductors, and resistors are all the same, so the circuit is nothing but a voltage divider. The complex-number gain of an R-C low-pass filter is the ratio of the impedance of the capacitor to ground (X_C) divided by the summed impedance of the R-C current path:

$$\widetilde{G}(f) = \frac{\frac{1}{sC}}{R + \frac{1}{sC}} = \frac{1}{sRC + 1}$$

This is commonly called the circuit's *transfer function* and may also be denoted as $H(f)$. It's simpler and more elegant than the scalar $G(f)$ formula we derived earlier.

To return from this representation to the real world, we'd normally substitute s with $j \cdot 2\pi f$, gather the j-free terms as the real part (often written in calligraphic form as $\mathfrak{Re}$), and pluck out the j-coupled bits as the imaginary part ($\mathfrak{Im}$). Once we have these values, we can calculate the resulting gain using the Pythagorean theorem:

$$G(f) = \sqrt{\mathfrak{Re}^2 + \mathfrak{Im}^2}$$

With complex numbers, we also get the voltage phase shift for free:

$$\theta_{RC\,(\text{series})} = \arctan\left(\frac{\mathfrak{Im}}{\mathfrak{Re}}\right)$$

There's no need for any additional whodunnits involving currents and voltages.

To be fair, in the $\tilde{G}(f)$ formula we're examining, the complex number is in the denominator of a fraction, so it might not be obvious how to separate $\mathfrak{Re}$ from $\mathfrak{Im}$. The most straightforward technique to remove j from the denominator is to multiply both parts of the fraction by a complementary complex number with the sign of the real part flipped: $(j \cdot a + b) \cdot (j \cdot a - b)$. This produces a real-only number because the j-coupled parts cancel out. However, there's a better way.

The shortcut is that if we take any two complex numbers, $z_1 = j \cdot a + b$ and $z_2 = j \cdot c + d$, we can easily show that the length of the Pythagorean diagonal of their complex-number product $(z_1 \cdot z_2)$ is the same as the product of the separately calculated diagonal lengths of z_1 and z_2. This follows from the analogous property of addition Figure 12-3, but we can also prove it more formally. We start by calculating the length of these separate diagonals, which we'll denote using the operator $|...|$:

$$|z_1| = \sqrt{a^2 + b^2}$$

$$|z_2| = \sqrt{c^2 + d^2}$$

Next, we simply multiply the results:

$$\begin{aligned}|z_1| \cdot |z_2| &= \sqrt{a^2 + b^2} \cdot \sqrt{c^2 + d^2} \\ &= \sqrt{(a^2 + b^2) \cdot (c^2 + d^2)} \\ &= \sqrt{(ac)^2 + (ad)^2 + (bc)^2 + (bd)^2}\end{aligned}$$

We need to show that this is equal to the measure of the diagonal of $z_1 \cdot z_2$. The result of this complex multiplication is:

$$\begin{aligned}z_1 \cdot z_2 &= (j \cdot a + b) \cdot (j \cdot c + d) \\ &= \underbrace{j^2}_{=-1} \cdot ac + j \cdot ad + j \cdot bc + bd \\ &= j \cdot \underbrace{(ad + bc)}_{\mathfrak{Im}} + \underbrace{(bd - ac)}_{\mathfrak{Re}}\end{aligned}$$

In the preceding formula, I've annotated the real and imaginary parts; the length of the Pythagorean diagonal for the resulting complex number is:

$$\begin{aligned}|z_1 \cdot z_2| &= \sqrt{\mathfrak{Re}^2 + \mathfrak{Im}^2} = \sqrt{(bd - ac)^2 + (ad + bc)^2} \\ &= \sqrt{(bd)^2 - \cancel{2abcd} + (ac)^2 + (ad)^2 + \cancel{2abcd} + (bc)^2} \\ &= \sqrt{(ac)^2 + (ad)^2 + (bc)^2 + (bd)^2}\end{aligned}$$

The calculated values of $|z_1 \cdot z_2|$ and $|z_1| \cdot |z_2|$ are the same. The proof applies to both multiplication and division, so if we have a complex fraction, we can assert that $|z_1/z_2| = |z_1|/|z_2|$. This means we can return to the earlier transfer function in the z_1/z_2 form, solve for the diagonals of the numerator and the denominator separately, and then combine the results. Once again, our starting point is:

$$\widetilde{G}(f) = \frac{1}{sRC + 1}$$

For the numerator, $\mathfrak{Re} = 1$ and $\mathfrak{Im} = 0$. The diagonal—also known as the *module*—is:

$$|numerator| = \sqrt{\mathfrak{Re}^2 + \mathfrak{Im}^2} = 1$$

We do the same for the denominator, where $\mathfrak{Re} = 1$ and $\mathfrak{Im} = 2\pi fRC$:

$$|denominator| = \sqrt{1 + (2\pi fRC)^2}$$

The resulting $|numerator| \,/\, |denominator|$ fraction describes the scalar magnitude of the $\tilde{G}(f)$ transfer function, and it turns out to be the same as the scalar gain formula we developed earlier:

$$|\widetilde{G}(f)| = G(f) = \frac{1}{\sqrt{(2\pi fRC)^2 + 1}}$$

It took us a bit of effort to get here, but that's only because we first needed to explain the fraction-based approach. On the upside, we don't need to perform circuit analysis to understand the resulting phase shift—it's already there, in the form of $\arctan(\Im\mathfrak{m} / \Re\mathfrak{e})$. There's another shortcut we can use to calculate this trigonometric function. I'll spare you the proof, but it's easy to show that for any complex fraction, the resulting angle is the difference of the angles calculated separately for the numerator and the denominator. In our case, this resolves to:

$$\theta_{RC\ low\text{-}pass} = \arctan\left(\frac{0}{1}\right) - \arctan\left(\frac{2\pi fRC}{1}\right) = -\arctan(2\pi fRC)$$

If you're fed up with complex numbers at this point, be assured that after this chapter we'll circle back to them just once. We'll do it to diagnose a faulty multistage filter circuit, where the trigonometric approach is a lost cause.

Uses of Inductors

I'm going to cover inductor circuits only in broad strokes, as inductor-based signal processing designs are increasingly uncommon in hobby electronics, owing to inductors' higher prices, larger physical dimensions, and limited advantage over their functional equivalents, which can be constructed using integrated chips. We'll spend a bit more time on inductor-based switching power supplies in Chapter 16. Until then, we'll keep it brief.

Let's start with the use of inductors for power conditioning—that is, for the elimination of external noise and the containment of noisy components within an electronic device. A prominent source of electronic noise is all kinds of switching, from brushed DC motors to the ultrafast transistors inside modern digital chips. In the case of digital ICs, the noise can sometimes harm the chip itself, causing voltage fluctuations sufficient to flip bits or destabilize internal clocks.

To solve the problem, we often employ lossy inductors (ferrite beads or chokes; see Chapter 8). These components augment a slower-acting capacitor near the chip or another noisy or sensitive load. A classic decoupling arrangement for digital chips is shown in Figure 12-16.

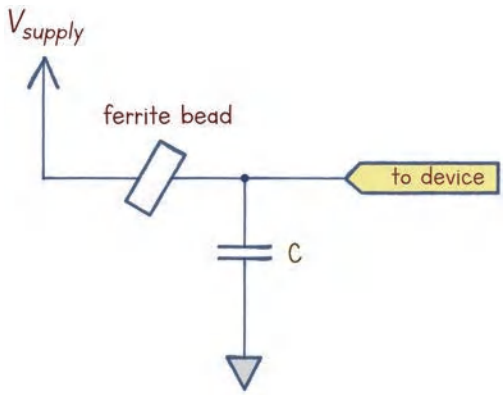


Figure 12-16: A decoupling circuit featuring a ferrite bead and a capacitor

Here, the capacitor provides a reservoir of energy to smooth out spikes in demand, while the choke forms a boundary that prevents high-frequency ripples from propagating to the other side by requiring currents to ramp up or down at a controlled rate.

The circuit in Figure 12-16 can be analyzed in terms of its L-C frequency response characteristics, but we usually don't bother—the goal is to reduce noise as much as possible, not to hit a specific mark. For digital chips, the rule of thumb is a capacitor of about 1 µF and a ferrite bead with an impedance of around 100 Ω at 100 MHz.

In addition to power-related applications, inductors can also be used to build signal filters. An R-L circuit with an inductor in the signal path and a resistor to ground forms a low-pass filter; the inverse configuration acts as a high-pass filter. Both circuits can be modeled similarly to their R-C counterparts. However, R-L filters are comparatively rare in signal-processing tasks. This is not only because of the components' cost or size but also because the layout results in an idle current flowing through the resistor at all times.

A more interesting filter building block is the L-C arrangement. As discussed in Chapter 5, an inductor and a capacitor connected in series will have a specific sine-wave frequency at which the reactance of the capacitor and that of the inductor cancel out, causing the circuit's impedance to drop sharply. We calculated this resonant frequency as:

$$f = \frac{1}{2\pi\sqrt{LC}}$$

This dip in impedance—a point where significant alternating currents can flow in response to modest voltages—can be used as a stepping stone for certain types of oscillators. The phenomenon can also be harnessed to construct R-L-C filters in which the gain can be boosted near the midpoint frequency, resulting in a sharper knee compared with the gradual roll-off seen in the non-resonant R-C architecture.

The usual R-L-C low-pass and high-pass layouts are shown in Figure 12-17.

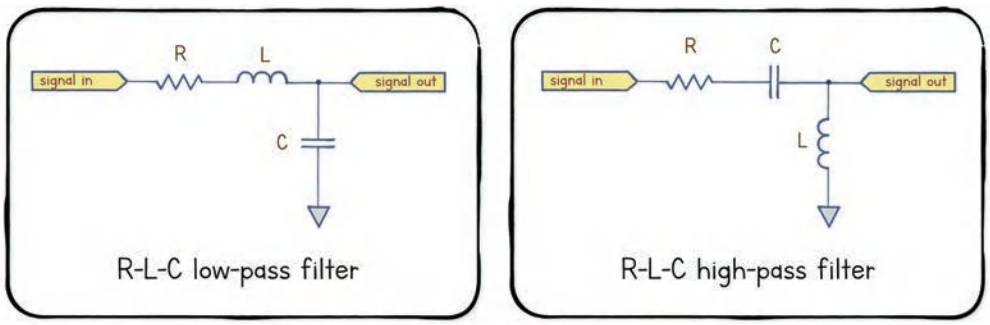


Figure 12-17: Common R-L-C filter layouts

Similar to what we did for their R-C analogs, these circuits can be modeled as three-component voltage dividers—the gain being the ratio of the shunt impedance to ground to the summed impedance of the R-L-C path. Using the low-pass variant as an example, the gain function is:

$$G(f) = \frac{Z_C}{\text{sum}(Z_R, Z_L, Z_C)}$$

This can be solved the old-fashioned way or by using complex numbers. The latter formula is much easier to manipulate because we don't need any trigonometry in the denominator:

$$\widetilde{G}(f) = \frac{\frac{1}{sC}}{R + sL + \frac{1}{sC}} = \frac{1}{sCR + s^2LC + 1}$$

The transfer function can be used to calculate the gain at the L-C resonant frequency, as derived in Chapter 5:

$$f = \frac{1}{2\pi\sqrt{LC}}$$

This frequency is also the center of the filter's knee. That gain at the knee frequency, also known as the filter's Q factor, works out to:

$$Q = \frac{\sqrt{\frac{L}{C}}}{R}$$

Earlier in the chapter, we noted that for a non-resonant R-C filter, the gain at the knee frequency is fixed:

$$\frac{1}{\sqrt{2}} \approx 0.707$$

The equation for an R-L-C circuit hints at a new possibility: We can select another value by keeping L and C constant and varying R. The appropriate resistor to achieve a specific Q factor can be calculated by rearranging the equation slightly:

$$R = \frac{\sqrt{\frac{L}{C}}}{Q}$$

Figure 12-18 shows the effect of various Q factors on the frequency response of the R-L-C low-pass circuit. Note that the Q factor is not synonymous with maximum gain; for example, the $Q = 1$ curve crosses through 1 at f_{knee} but peaks at around 1.15 slightly before that.

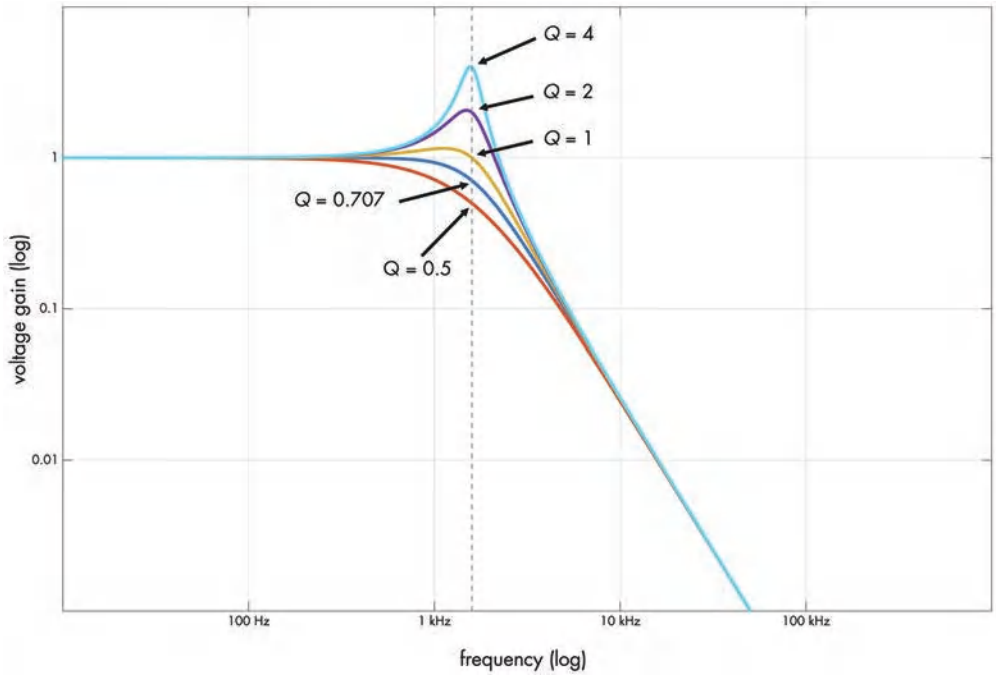


Figure 12-18: R-L-C low-pass gain curves for various Q factors

Another thing to note is that this filter's attenuation slope is sharper than that of its single-stage R-C counterpart (Figure 12-14). In the R-C filter, the amplitude drops

tenfold with every tenfold increase in sine-wave signal frequency. Here, the attenuation rate is 100:10, a consequence of the frequency appearing in the denominator of the transfer function in squared form. We call these f^2 filters *second order*.

In practice, we usually don't tinker with inductor-based filters, even if we want this kind of resonant response. It's more practical to use active circuits that include a cleverly placed amplifier—a good example of this is the Sallen–Key topology discussed in Chapter 15. If the rare need arises, however, entire books have been written about analog filters and oscillators, and you can look up the relevant details online as well. With the analytic toolkit now in place, you'll be in a better position to fight back if a page assaults you with transfer functions and mysterious constants such as s and j.

We're now ready to leave all the trigonometry in the rear-view mirror and dive into the mechanics of semiconductor circuits. In Chapter 13, we begin with another look at diodes.

13

APPLICATIONS OF DIODES



In this chapter, we'll discuss a variety of diode-based circuits. The diode lives in the shadow of its more capable sibling, the transistor. As we'll see, though, the component is nevertheless capable of pulling off quite a few interesting tricks.

Readers who made it through Chapter 12 might be starting to suspect that electronics is just endless math. For them, I have good news: In the remainder of the book, there are virtually no formulas or theorems to memorize. We'll do some calculations once more—in Chapter 15, when discussing operational amplifiers—but beyond that, we'll focus on abstractions that reduce mental toil rather than add to it. Case in point: In this chapter, the most serious math you'll encounter is simply adding and subtracting voltages.

Circuit Protection and Signal Clipping

About the simplest application of diodes is circuit protection. In Chapter 10, we introduced transient voltage suppressors (TVSes)—diodes with a carefully chosen breakdown voltage that are placed in reverse bias between a protected line and ground. The diode remains nonconductive under normal operating conditions, but it experiences electrical breakdown and temporarily starts conducting when the input voltage exceeds a safe limit. In effect, the diode acts as a crowbar that dissipates energy and protects more delicate components downstream. This arrangement is shown in the first panel of Figure 13-1.

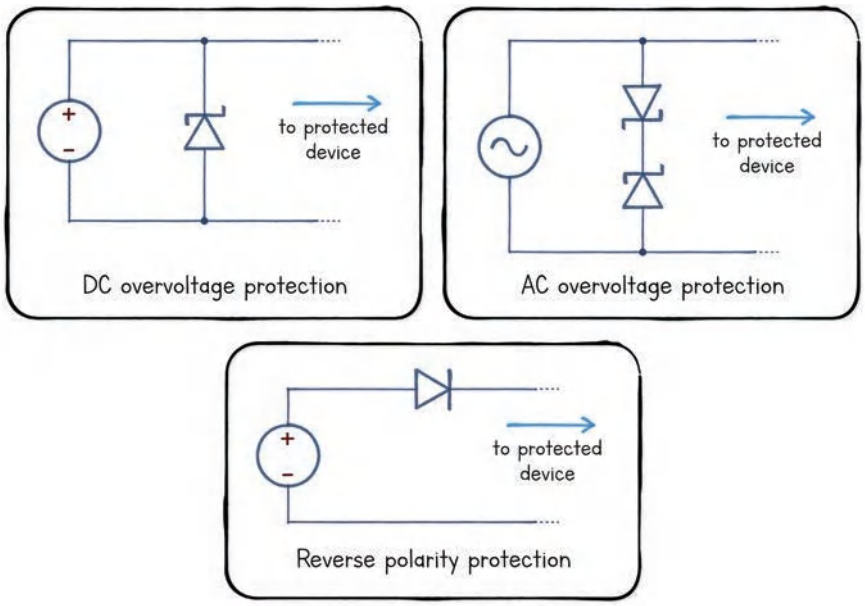


Figure 13-1: Three circuit protection schemes using diodes

A single TVS (or a generic Zener diode) can only be used to protect inputs where the input signal has a defined polarity—that is, where one rail is always more positive than the other. If the polarity is reversed, the diode becomes forward-biased and begins to conduct much sooner than intended.

To provide overvoltage protection for AC signals with alternating polarity, we instead use the two-diode arrangement shown on the right in Figure 13-1. This combination of components is also available in a single package known as a *bidirectional TVS*. In the circuit, regardless of the polarity of the applied voltage, the "crowbar" path always consists of a forward-biased diode in series with a reverse-biased one. It follows that the positive and negative conduction thresholds are the same: They're equal to the diode's reverse breakdown voltage plus roughly 600 mV needed to overcome the field of the other p–n junction in forward bias.

The same approach can be used to clip an input signal to a range of voltages for reasons other than circuit protection. For example, in electronic musical instruments, a sine-wave signal can have its peaks clipped by a pair of Zener diodes to make it resemble a square wave, thus producing a different, harsher sound.

Another useful circuit protection technique is shown at the bottom of Figure 13-1. A regular forward-biased diode with a high reverse breakdown voltage can be placed in series with the supply. This arrangement prevents potential damage to sensitive components if the polarity of the supply is accidentally reversed—for example, if the batteries are inserted the wrong way. The trade-off is the inevitable voltage drop across the p–n junction, along with resistive heating if the current is high. A transistor can be used to work around these issues. We'll circle back to this challenge in Chapter 14.

In signal-protection applications, large Zener diodes and TVS diodes have fairly large "off" capacitance that risks attenuating fast-changing signals. To address this issue, we sometimes use a modified protection circuit, shown in Figure 13-2. In this design, two small, normally reverse-biased diodes (D_1 and D_2) provide a shunt between the input signal and the supply rails. D_1 begins to conduct if the input voltage rises above the positive supply, while D_2 conducts if the signal falls below the negative rail. A separate, beefier TVS (D_3) limits the voltage between the supply rails themselves.

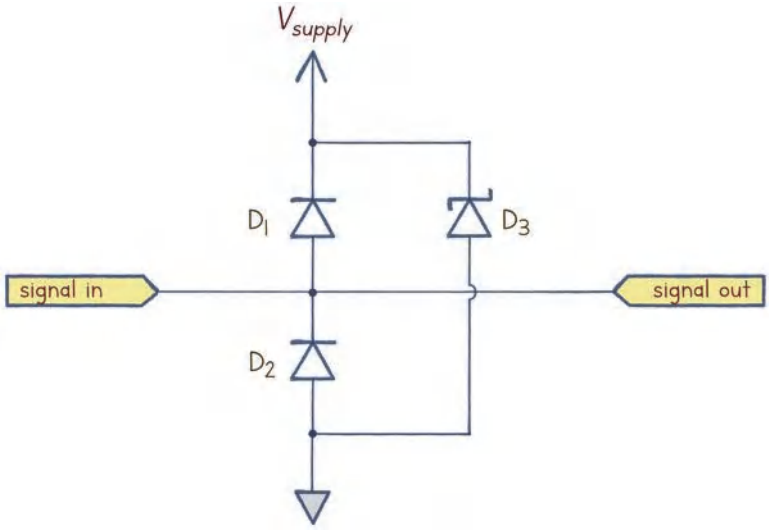


Figure 13-2: Input line protection with steering diodes

In this topology, we refer to D_1 and D_2 as *steering diodes*.

Voltage References

In Chapter 10, we outlined a simple way to build a voltage reference that's relatively insensitive to fluctuations in the supply voltage. The schematic of the circuit described in that chapter is shown in Figure 13-3.

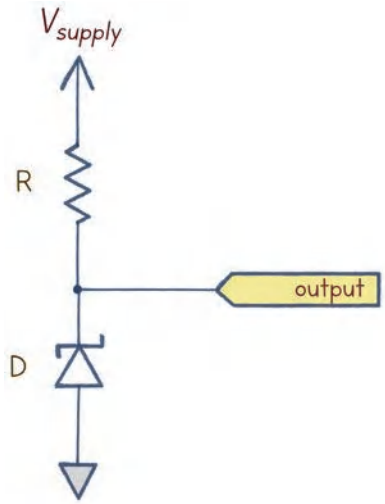


Figure 13-3: Using a Zener diode as a voltage reference

In this circuit, the current flowing through the resistor changes linearly with the applied voltage ($I = V/R$), but the voltage that develops across the terminals of a conducting Zener diode varies relatively little in relation to I. In one instance (Figure 10-4), we observed that the voltage swing on the output leg of the circuit was equal to just 5 percent of the variations in the supply voltage.

The number might not sound impressive, but the circuit can be cascaded: The output of one resistor-diode voltage reference can be used as the supply voltage for

a second one (Figure 13-4). The effects stack up: 5 percent of 5 percent is equal to 0.25 percent—a much more respectable result.

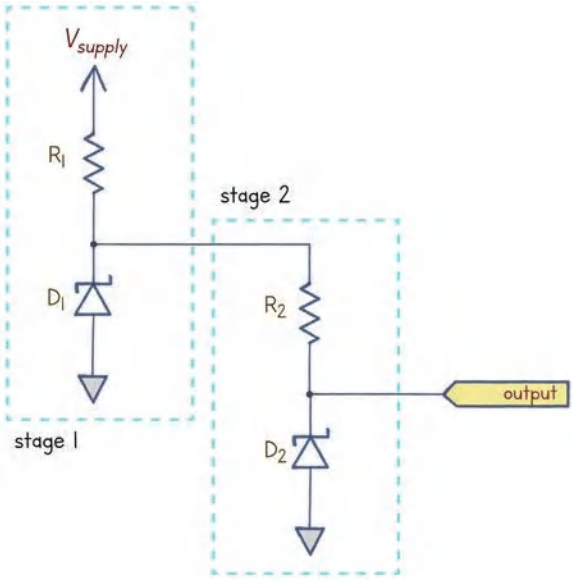


Figure 13-4: Cascaded voltage references

For the cascaded layout to behave correctly, the first diode needs to have a higher Zener voltage than the second one. We also need to ensure that the side current going into the second stage is much lower than the intended current flowing through the first-stage resistor-diode path. This gotcha is similar to the output loading problem of resistor-based voltage dividers, as discussed in Chapter 12. One way to keep the loading minimal is to assign R_1 a value about two orders of magnitude lower than R_2—for example, R_1 = 1 kΩ and R_2 = 100 kΩ. Another solution is to separate the stages with a transistor circuit that mirrors the output voltage of stage 1 for use by stage 2. We'll discuss transistor and chip-based voltage followers soon.

A more significant wrench in the works is that diode voltage changes to some extent in response to ambient temperature. Luckily, while the reverse breakdown voltage of a Zener diode rated for 5 V or more increases with temperature, the forward-bias voltage of a regular diode drops. This means that a combination of forward- and reverse-biased diodes can be used to compensate for the drift.

In practice, we usually don't bother to construct multistage temperature-compensated circuits by hand. When high precision is required, ready-made integrated circuits can be bought for pennies. The two basic architectures are *shunt voltage references*, which are drop-in replacements for Zener diodes (sometimes with a third terminal for adjusting voltage), and *series voltage references*, which are standalone devices that are connected to a power source and output the reference voltage on a separate pin.

An example of a shunt regulator is the LM4040 from Texas Instruments. It's a three-terminal package, but only two terminals are used by the chip. Another variant from the same manufacturer, the TL431, features a third terminal that can be used to adjust the generated voltage—essentially by setting the gain of an internal amplifier. As for series references, you can have a look at the datasheet for Microchip's MCP1525.

Because the amplifiers used in these integrated circuits have a limited response speed, the devices usually can't suppress fast supply fluctuations as well as they can respond to slower drift. We'll talk about these characteristics of amplifiers later, but

a good rule of thumb is that the danger zone spans from around 1 kHz to 100 kHz. Supply ripple at these sine-wave frequencies typically needs to be reduced with capacitors and possibly inductors before the voltage reference can do its job.

Rectification

In Chapter 10, we experimented with a basic diode-based circuit that converted an AC signal to a steady DC voltage by charging a capacitor through a diode (see Figure 10-2). An analysis of this arrangement—a *half-wave rectifier*—is shown in Figure 13-5.

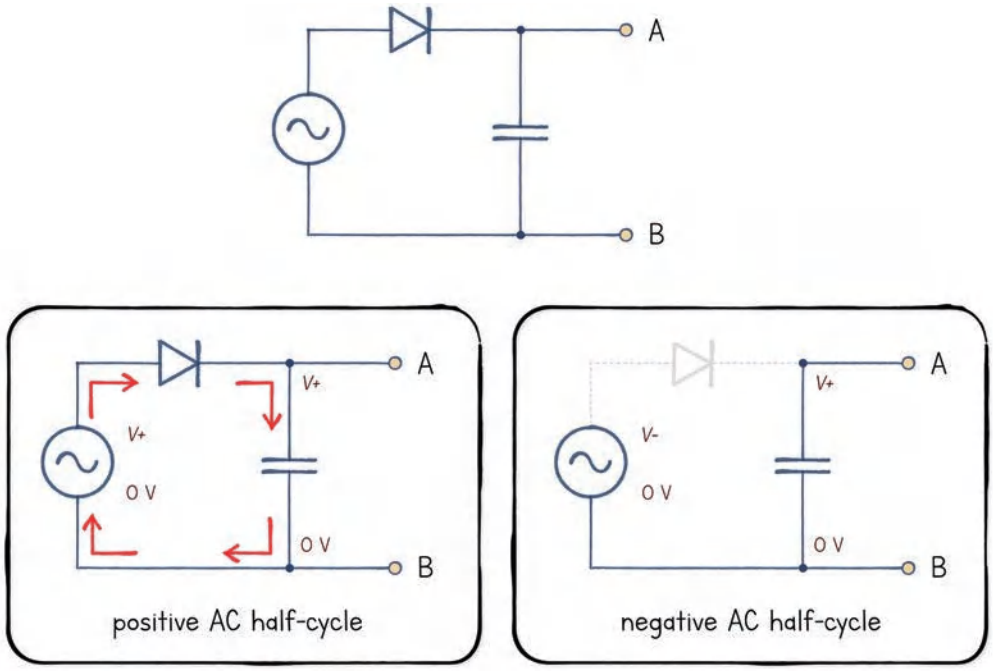


Figure 13-5: A half-wave rectifier (top) and a simplified analysis for the positive and negative AC cycles (neglecting diode voltage drop)

The analysis is easiest if we reference all circuit voltages to the bottom output leg, marked as B. During the positive half-cycle of a sine wave, when the upper terminal of the supply is more positive, the diode is initially forward-biased. This allows the capacitor to charge to a voltage equal to the peak amplitude of the input signal, minus the usual diode voltage drop. When the polarity of the signal reverses—making the upper supply terminal more negative—the diode becomes reverse-biased and doesn't conduct, so the capacitor holds its charge. Figure 13-6 shows a discrete-time simulation of the process, with some modest resistive load added across the output terminals to discharge the capacitor slightly between each positive peak.

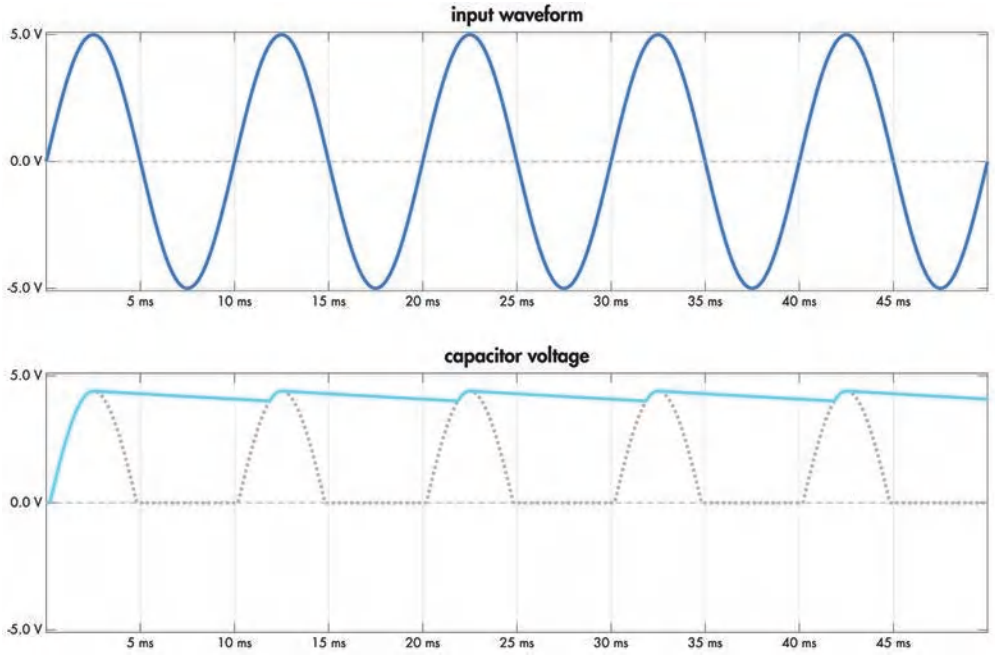


Figure 13-6: A discrete-time simulation of a half-wave rectifier with a modest load. The dotted line on the bottom plot shows the positive half of the input waveform, minus 600 mV.

If the load resistor is made a permanent part of the circuit and is chosen so that the capacitor discharges quickly enough to keep up with the amplitude modulation of a carrier wave, we get a circuit known as an *envelope follower*. The circuit is a simple way to extract the approximate modulating waveform from the carrier signal in AM radio circuits and similar applications (Figure 13-7). Notice how the capacitor voltage tracks the changing peak heights of the input waveform.

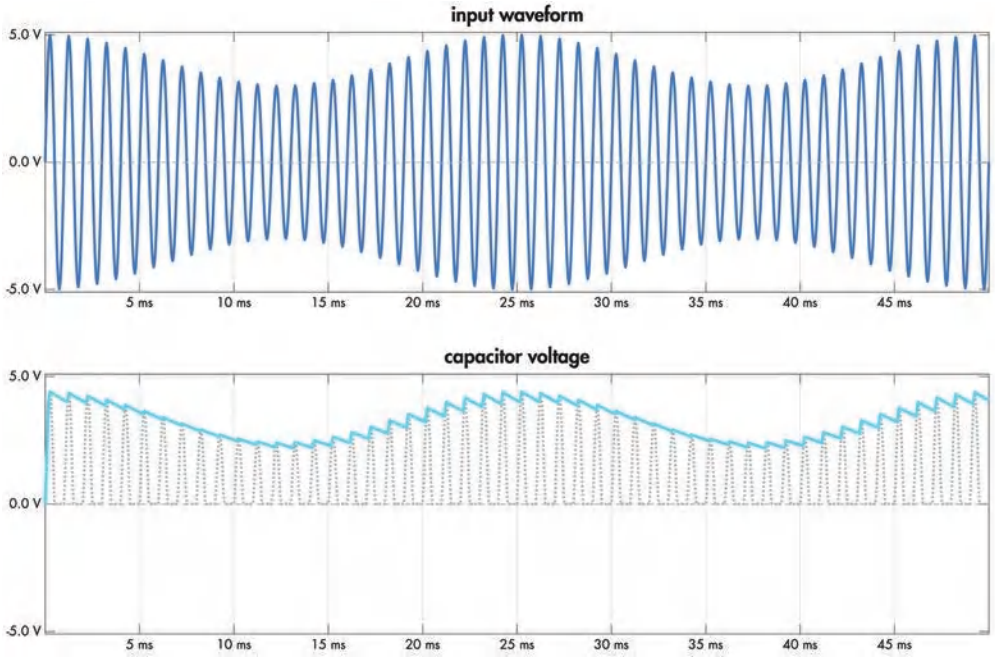


Figure 13-7: A half-wave rectifier as an envelope follower for AM demodulation

The main drawback of the half-wave rectifier is that the capacitor is charged only by the positive half-cycle of the sine wave. This is wasteful if the goal is to maximize the power delivered to a load. The deficiency can be addressed by constructing a *full-wave rectifier*, shown in Figure 13-8.

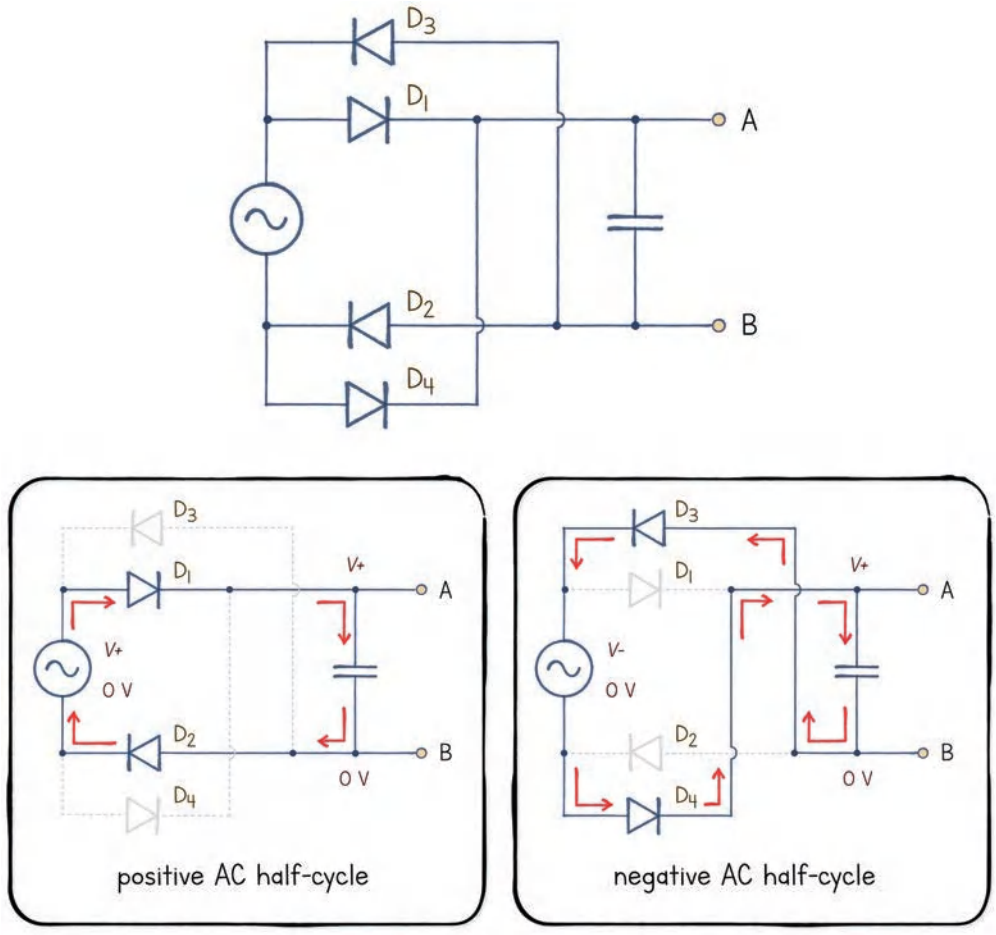


Figure 13-8: A full-wave rectifier (top) and its simplified analysis for positive and negative AC cycles (neglecting diode voltage drops)

Once again, the analysis of the circuit is simplest if we consider output B to be the reference point. During the positive half-cycle, diodes D_1 and D_2 are initially forward-biased. This allows the capacitor to charge to the AC peak voltage (minus the summed 1.2 V forward drop of two diodes). During the negative half-cycle, diodes D_3 and D_4 become forward-biased, which connects output B to the upper supply terminal. Meanwhile, the bottom supply terminal, which is currently at a higher voltage, is connected to point A. This allows the capacitor to continue charging in the same polarity as before. A simulation of the process is shown in Figure 13-9.

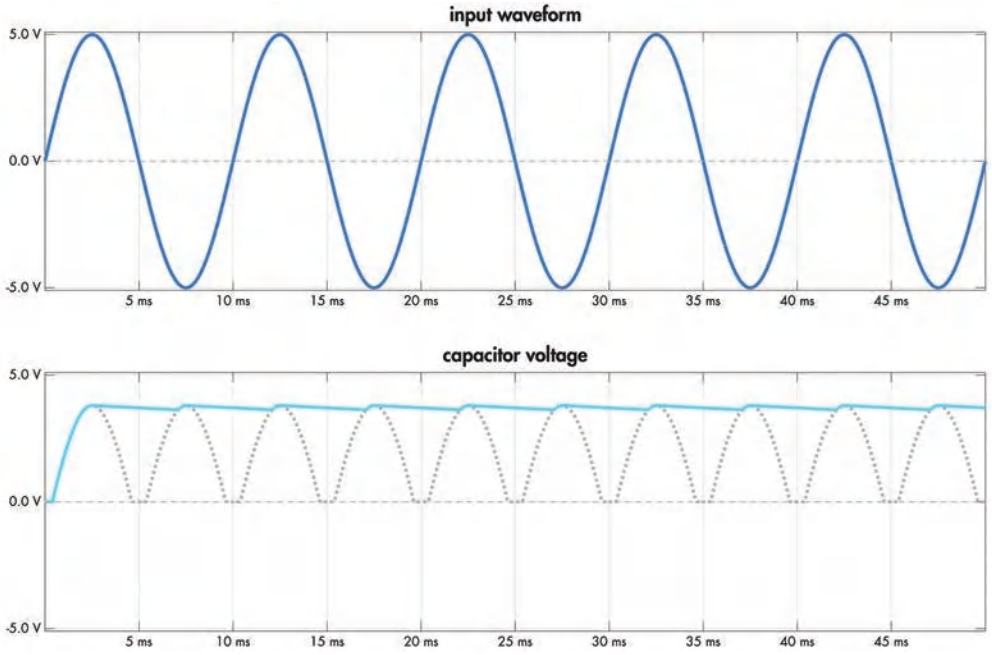


Figure 13-9: A discrete-time simulation of a full-wave rectifier. Note that the capacitor voltage is boosted twice in each cycle of the input waveform, first during the positive peak and then during the negative peak.

Figure 13-10 outlines another way of drawing the full-wave rectifier circuit. This diamond shape is taught in many books, so it's good to be able to recognize it on sight. It's also one of the few situations where no one will complain if you draw diagonal lines in a circuit schematic.

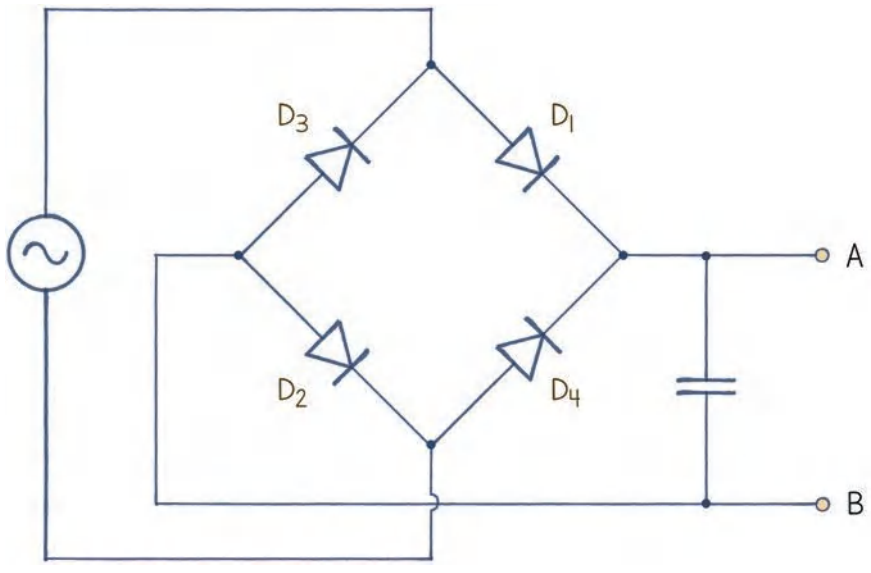


Figure 13-10: An alternative way of drawing a full-wave rectifier

The rectifier circuit is worth a try. The arrangement can be useful even in DC-powered circuits, allowing the output capacitor to charge to a positive voltage regardless of the polarity of the battery or power supply.

Voltage Doubling

The rectifier circuits outlined in the preceding section use diodes as voltage-controlled switches. Another application of the same principle is a circuit known as a *voltage doubler*.

There are many flavors of voltage doublers, but one particularly clean design is shown in Figure 13-11. The circuit outputs a DC voltage that's equal to twice the peak amplitude of the zero-centered input waveform, minus the usual diode voltage drops.

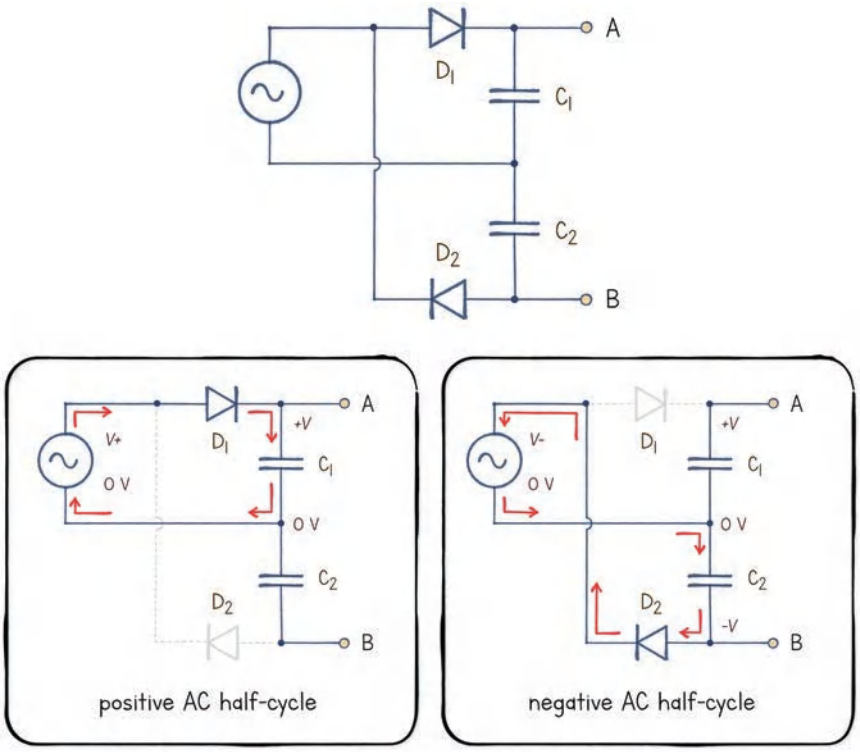


Figure 13-11: A simplified analysis of a voltage doubler circuit (neglecting diode voltage drops)

This time, let's use the midpoint between the two capacitors as the reference point for voltage measurements. During the positive half-cycle of the AC signal (left panel), the upper diode (D_1) can conduct to charge the top capacitor. This puts the output terminal A at a positive voltage relative to the midpoint.

Next, let's examine the negative half-cycle (right). Here, the top diode is always reverse-biased, so it doesn't conduct; C_1 retains charge. At the same time, conduction is possible for the lower diode (D_2), which charges C_2 so that output terminal B ends up at a negative voltage relative to the midpoint. In effect, we store the positive peak voltage of the input waveform in C_1 and the negative maximum in C_2. The total voltage between B and A is $V_{peak} \cdot 2$ (once again, minus the expected diode voltage drops).

The idea of multiplying voltages with switched capacitors lives on, although modern circuits typically use digitally controlled transistors instead of diodes. This avoids voltage drops and eliminates the need for a standalone AC voltage supply. We'll revisit switched-capacitor circuits in Chapter 16.

DC Restoration

Most of the time, alternating waveforms that are centered on 0 V are inconvenient to work with. In particular, it's more challenging to build circuits that run off a single voltage supply but that can discern, amplify, or generate signals that extend below the negative supply rail. We outlined one possible solution in Chapter 12, where we discussed a resistor-based technique for shifting voltages up and down. The downside of the approach was that it attenuated the input waveform proportionally to how strongly we pulled the signal toward one of the supply rails.

This brings us to a somewhat mind-bending alternative approach, shown in the top panel of Figure 13-12. The circuit is known as a *clamper*—or, less cryptically, a *DC restorer*. It takes an AC waveform and shifts it so that the negative peaks are at roughly 0 V, without having an appreciable impact on signal amplitude.

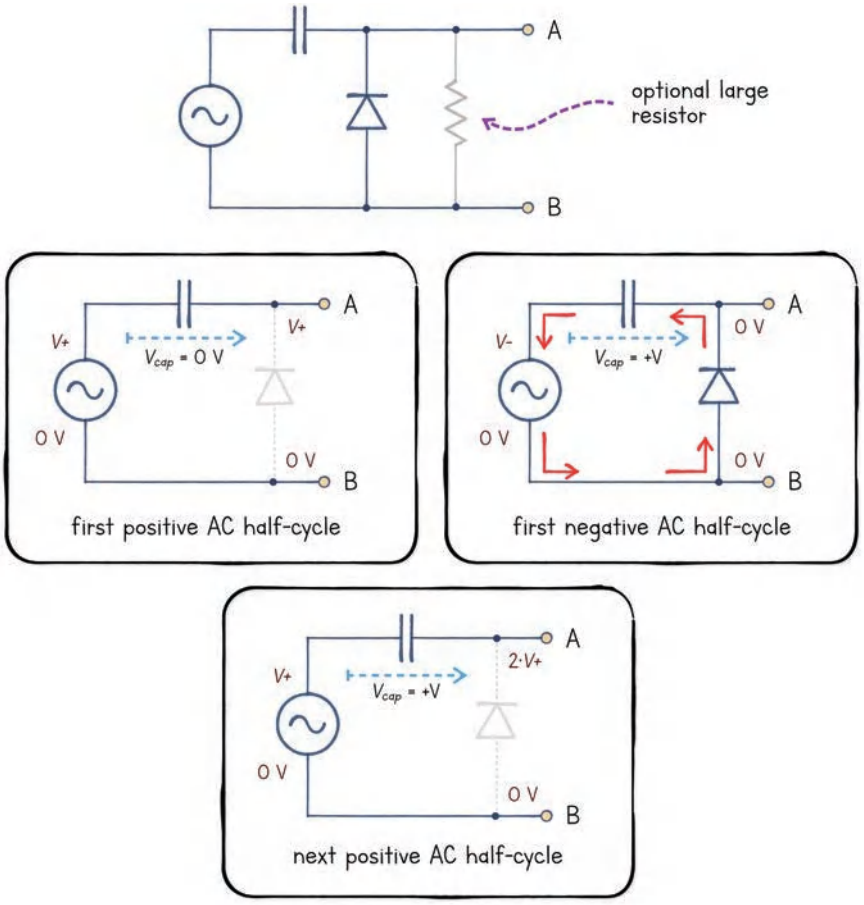


Figure 13-12: A positive clamper circuit and its simplified analysis (neglecting diode drop)

For now, let's ignore the optional resistor. We start with the first positive half-cycle of the input sine wave (center left). Initially, the capacitor isn't charged (V_{cap} = 0 V). Further, there's no current path via the diode, so charging isn't possible. It's the nature of the capacitor that the electromotive force still couples through the dielectric gap: An attempt to push more electrons onto one plate produces a sympathetic force that tries to boot electrons from the other plate. This means that in a capacitor with one free-floating terminal, any voltage applied to one side carries across. We should therefore expect the voltage at point A to track the supply signal 1:1.

In the negative half-cycle, once the voltage at point A reaches about −600 mV, the diode starts to conduct (center right). Because the diode conducts well, this essentially creates a connection between point A and ground, clamping the voltage at that node while simultaneously allowing the capacitor to charge so that its left terminal becomes negative relative to the right terminal. If we measure V_{cap} left to right, the resulting voltage is positive and equals the peak amplitude of the supply signal minus roughly 600 mV.

In the subsequent positive cycle, the diode becomes reverse-biased again, so the capacitor must hold its previous charge. This means that V_{cap} remains unchanged and that the voltage at point A is necessarily offset from the input waveform by an amount equal to the capacitor voltage. If the input waveform runs from $-V_{peak}$ to $+V_{peak}$, the output will now move from −0.6 V to $V_{peak} \cdot 2 - 0.6$ V.

The role of the optional load resistor is to control the circuit's ability to respond to gradual shifts in signal amplitude. Without it, the circuit would theoretically be stuck forever at a voltage offset dictated by the largest encountered single-cycle swing in the input waveform. With a resistor, the capacitor can discharge over time, so the offset can change if the envelope of the waveform changes in some way.

In practice, the diode's leakage current and the capacitor's self-discharge rate are typically enough to make the circuit behave reasonably even without an output resistor. If you wish to experiment with this layout, I recommend using a 10 to 100 µF capacitor and either skipping the resistor or using a large one (1 MΩ).

Diode-Based Logic Gates

A proper dive into digital logic is coming in Part IV of the book. However, some of you are probably already familiar with the concept of logic gates, which lie at the heart of digital computing; if so, you might be interested to know that they can be approximated using simple diodes.

An OR gate is a circuit with two or more inputs that produces a positive voltage when any of the inputs is positive. An AND gate, in contrast, outputs a positive signal only if *all* the inputs are positive. A practical application of an OR gate might be a circuit that raises an alarm if any of the door or window sensors in a house are triggered. An application of an AND gate might be a system that illuminates a "PARKING FULL" sign when all the spots in a parking garage are occupied. Figure 13-13 shows two simple ways to implement such logic with diodes.

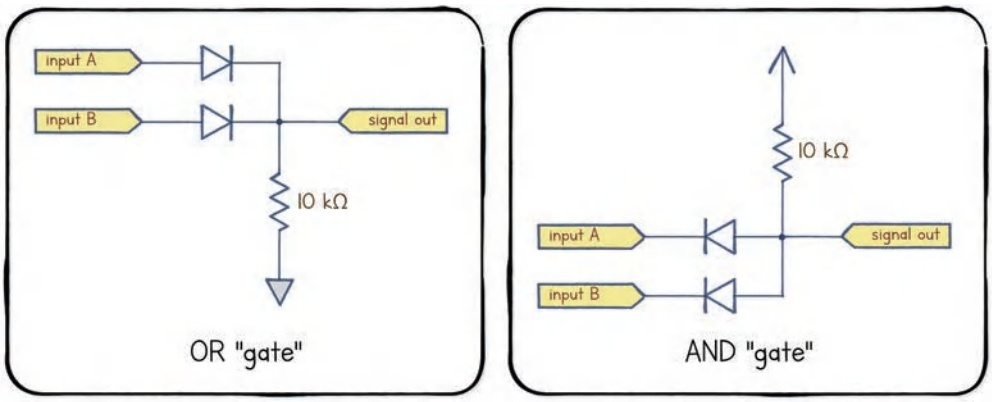


Figure 13-13: Two simple, diode-based logic pseudo-gates

In the case of an OR circuit (left), if the positive supply rail is connected to any of the input terminals, this forward-biases the corresponding diode and causes a current to flow through a resistor. Because the impedance of a forward-biased diode is much lower than that of a 10 kΩ resistor, the output voltage shoots up very close to the upper supply rail. The AND circuit (right) essentially works in reverse: If any of the inputs is connected to ground, this pulls the output voltage close to 0 V, so the output is positive only if both inputs are high (or are left floating).

The reason I put "gate" in scare quotes in Figure 13-13 is that the circuits aren't readily composable to implement more complex digital logic. Each of the gates requires current to flow through the input terminals, but it can't necessarily deliver such currents on its output leg. To illustrate the problem, consider the arrangement of an OR gate and an AND gate shown in Figure 13-14.

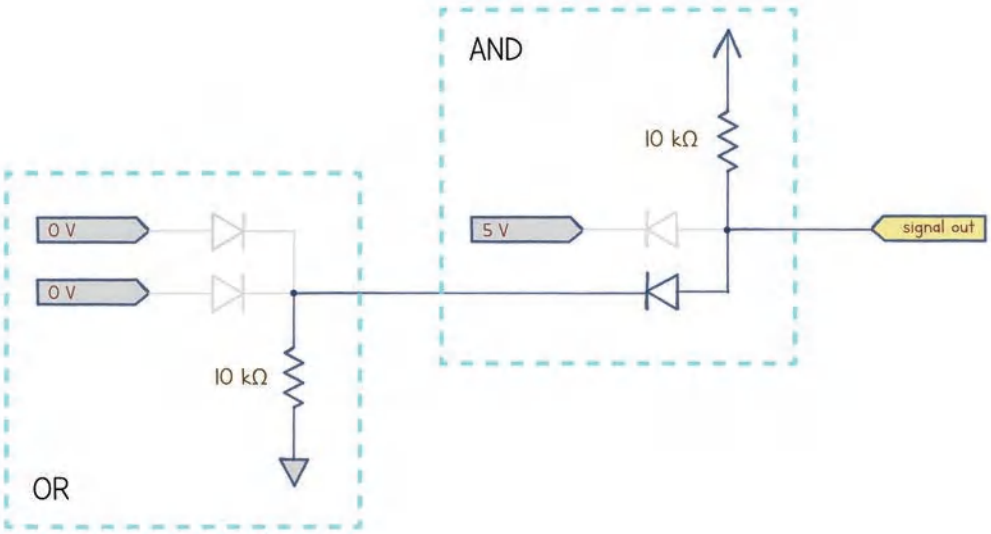


Figure 13-14: The problem with diode-based logic gates

With the input signals as indicated in the drawing, three out of four diodes are reverse-biased. The only current path is the series resistor–diode–resistor connection between the positive supply rail and ground. The circuit is essentially a voltage divider. Instead of providing a binary output, it produces an ambiguous intermediate voltage. To build real computers, we'd need gates that can deliver output currents higher than what they demand as input of the gates upstream. We'll talk about the way to do this in Chapter 18.

Now that we've covered the uses of diodes, we can turn to the celebrity of the semiconductor world. In Chapter 14, we'll cover the applications of transistors.

14

WORKING WITH TRANSISTORS



In the previous chapter, we showed that we can use diodes without having to wade knee-deep through equations and function plots. Now we'll do the same for transistors, exploring some of their applications for switching, signal amplification, and more. The behavior of these components is relatively difficult to model and influenced by factors we can't fully control, so rigorous approaches are usually a dead end. Instead, the goal is to design circuits that behave as intended even if we don't have the transistor's V-I curves mapped out exactly right.

The technique we'll repeatedly lean on is called *negative feedback*. It boils down to looping the output of a circuit back onto its input so that if the output signal starts drifting off target, the relevant voltages are pulled in the opposite direction until the error goes away. This sounds complicated, but it isn't—in the simplest case, it's just a matter of adding a single resistor in the right spot.

In the following pages, we'll focus primarily on circuits constructed using n-channel MOSFETs because these transistors tend to perform better and are easier to model than their p-channel and bipolar junction counterparts. That said, each example in the chapter can be easily tweaked to use transistors of a different type.

We won't be reviewing the entirety of transistor lore; there are hundreds upon hundreds of circuits you can find online, most of them dating back to an era when operational amplifiers (Chapter 15) and programmable digital chips were out of reach for hobbyists. The basics are still a must, but the rest probably isn't worth your time.

Switching Circuits

The most common transistor circuit is an electronically controlled switch. This application can be thought of as the solid-state equivalent of an electromechanical relay (Chapter 9). In addition to its obvious utility for toggling other electronic circuits on and off, the transistor switch plays a pivotal role in digital computing.

Common-Source Switch

Figure 14-1 shows the most common and robust switch layout implemented with n-channel and p-channel MOSFETs. This architecture is known as *common source*; the equivalent BJT topology is called *common emitter*. In this context, the word *common* refers to the terminal that's connected to a common supply rail.

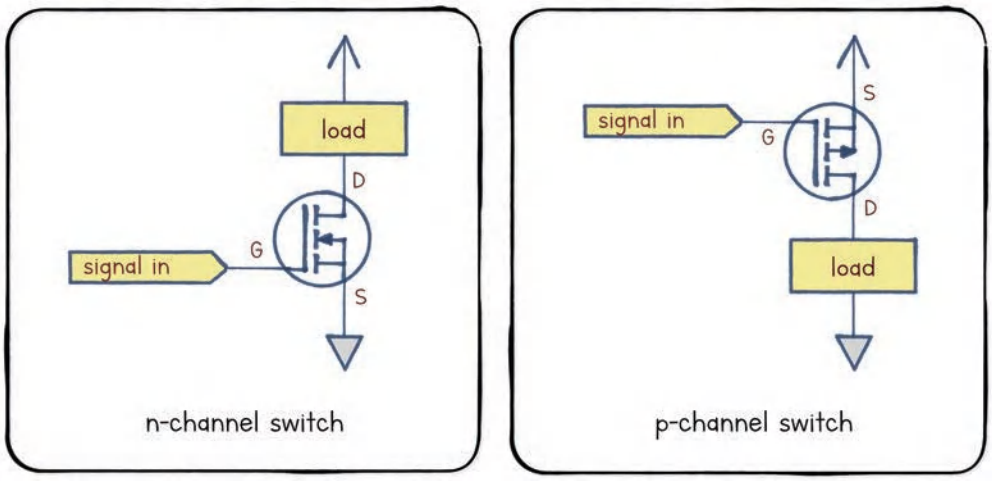


Figure 14-1: Transistor switching circuits, as implemented using n-channel and p-channel MOSFETs

Let's start with the n-channel circuit (left). In an n-channel MOSFET, the voltage on the source terminal can't be higher than the drain; otherwise, we risk forward-biasing the internal p–n junction (body diode) and making the device conduct in its "off" state. This means that if we want to tie the source terminal to a supply rail, it must be the negative supply rail.

To turn the transistor on, we store a positive charge in the internal capacitor formed primarily between the gate and the source. Once the voltage across these terminals (V_{GS}) reaches a certain threshold, some modest current is allowed to flow through the drain–source conduction path. For discrete MOSFETs, the threshold voltage (V_{th} or $V_{GS(th)}$) is usually around 2 V, coinciding with a manufacturer-specified current that may hover around 250µA or 1 mA. To admit higher currents, we'd need to increase V_{GS} further, in line with the transistor's V_{GS} to I_D curve (Figure 10-12 in Chapter 10).

When using a transistor as a switch, the goal isn't to dial in a specific current limit; instead, we increase V_{GS} as far as we can so that the transistor doesn't impede current flow. Many MOSFETs can supply their rated continuous current by the time V_{GS} reaches 5 V, though they might require up to 10 V to meet the remaining specs. The datasheet will contain a combination plot called "output characteristics" or "on-region characteristics," which shows current curves for various gate voltages and allows you to determine how many amps the switch can handle at the V_{GS} available in your circuit.

Of course, you still need to keep the component's power rating in mind, because the transistor dissipates heat proportional to the current flowing through it and the applied voltages. As a convenient abstraction, the datasheet typically provides "on" resistance ($R_{DS(on)}$) curves for various combinations of I_D and V_{GS}. This resistance is not constant, but if you look up $R_{DS(on)}$ for your specific use case, you can calculate the dissipated heat from Joule's law: $P = I^2R$.

The discussion so far covers the n-channel architecture; the second panel in Figure 14-1 also shows the p-channel counterpart. With a p-channel MOSFET, the source terminal needs to be kept at a higher voltage, and the conduction is triggered by making V_{GS} sufficiently negative. In other words, when $V_{GS} = V_{supply}$, the p-channel switch is off; when $V_{GS} = 0$ V, the load is energized.

For completeness, bipolar junction transistors (BJTs) can be used in a similar way. An NPN transistor can take the place of an n-channel MOSFET; a PNP one substitutes a p-channel MOSFET. In both cases, conduction is triggered by supplying a small base current that induces a proportionally larger current in the main conduction path. The exact value of BJT current gain (h_{fe}) shouldn't be relied on, but the spec should provide a safe minimum; for general-purpose transistors, this minimum gain is typically between 50 and 100. We can then use a series resistor to limit the base current to the desired value, taking into account the diode-like 600 mV conduction threshold at the base terminal.

In all transistor-switching circuits, care must be taken when disconnecting inductor-like loads (motors, speakers, relays, and so forth). When the current flowing through an inductor is abruptly stopped, the collapsing magnetic field continues to shuffle electrons around and may produce a brief voltage spike that can damage sensitive components. A simple workaround is to place a TVS diode across the terminals of the load (or across the source and drain terminals of the transistor). The diode remains nonconductive until the voltage exceeds a certain threshold, at which point it acts as a shunt to protect other components.

Polarity-Detecting Switch

In Chapter 13, we discussed the use of a series diode for battery-polarity protection. We also noted a major disadvantage of the solution—the diode produces an unavoidable voltage drop. Figure 14-2 shows a better approach implemented with a MOSFET as an unconventional switch.

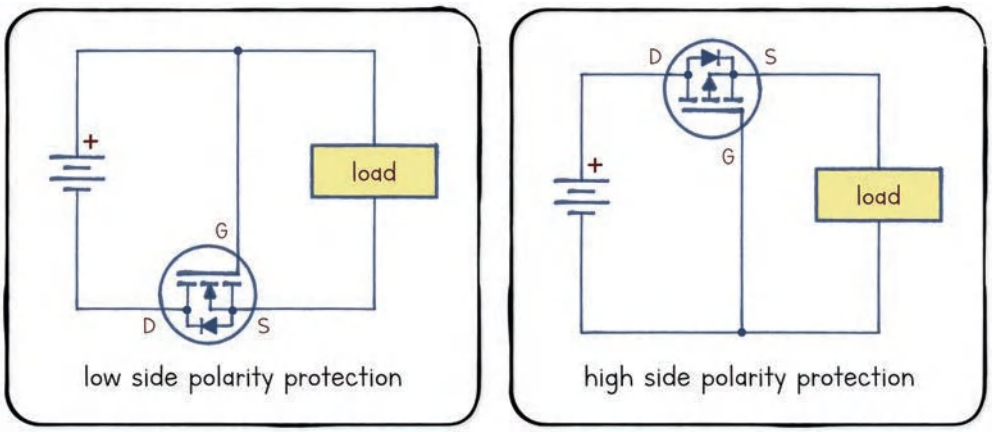


Figure 14-2: A MOSFET as a polarity-protection device: n-channel (left) and p-channel (right)

Note that in both circuits, the transistors are placed the "wrong" way around: The internal diode is poised to always conduct if the supply is connected the right way. This arrangement is intentional and mimics the use of a diode for polarity protection.

To understand the added benefit of MOSFETs in this role, let's focus on the n-channel variant shown on the left. If the gate were left disconnected and discharged, a diode-related voltage drop would appear across the transistor, placing the source terminal at 600 mV in relation to the negative supply rail. But in this circuit, the gate terminal is connected to the positive rail, so $V_{GS} \approx V_{supply} - 600$ mV. This configuration normally puts the transistor firmly in the conduction regime, bridging the drain and source terminals and causing the body diode to disappear. In contrast, if the supply is accidentally connected the wrong way, the diode is reverse-biased and V_{GS} is negative, so the transistor remains off.

The circuit can be built with an n-channel transistor to control the negative supply rail or with a p-channel device to interrupt the positive supply. The latter option may be preferred if the circuit must maintain an uninterrupted ground connection at all times—for reasons such as noise suppression or safety.

The polarity-protection circuit has some failure modes. First, because of the MOSFET conduction threshold, it doesn't work well at very low supply voltages. Second, if the current needed by the load is greater than what the transistor can muster, the voltages can get out of whack. Finally, a supply overvoltage event—for example, a static discharge—may damage the gate terminal unless an additional TVS is put in place. Nevertheless, for circuits running at 3 V and above, and with an appropriately sized transistor, the solution beats just using a diode.

Signal Followers

There are many situations where we want to create a replica of a voltage- or current-based signal without loading its original source to an appreciable extent. This may be because the source can't supply sufficient current to drive the components downstream, or because we want several branches of the circuit to behave exactly the same way without risking any crosstalk. Let's discuss some of these replicating circuits, broadly called *signal followers*.

Voltage Follower

Voltage followers are used to remedy the issues caused by signal sources that can't supply significant currents. Such sources are said to have high impedance; this is analogous to an idealized signal generator that can supply arbitrary currents but is placed in series with a large resistor. To turn a high-impedance signal into a low-impedance one, we can build a circuit that creates a copy of the input signal without significantly loading the original source. The simplest voltage follower is shown in Figure 14-3.

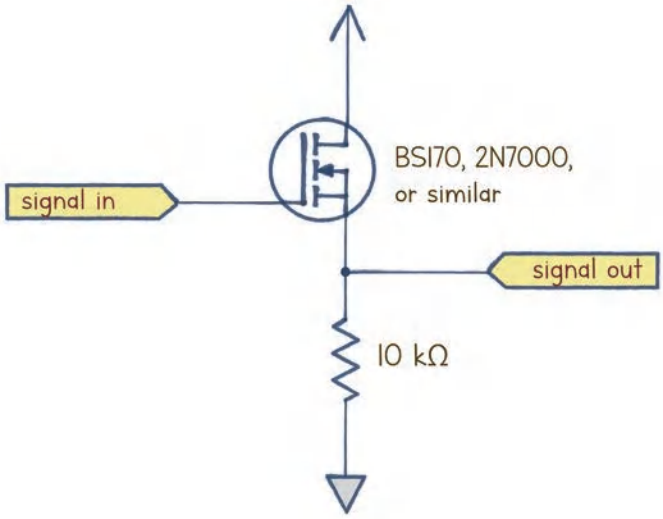


Figure 14-3: A rudimentary voltage follower. The resistor acts as a dummy load and can be omitted if the load offers a reasonable path to ground.

This transistor layout is known as *common drain* (or *common collector* if you're building it with a BJT). In the n-channel case, the drain terminal is connected to the positive supply rail. The circuit behaves roughly the same with a dummy resistor or with a more useful resistive load in its place. In both cases, if there's current flowing through that path, then from Ohm's law, a proportional voltage develops across the resistance: $V_R = IR$.

In contrast to the earlier switching layout, the source terminal isn't tied to ground, so the gate-to-source voltage differential that controls the transistor's behavior isn't dictated solely by the signal supplied on the circuit's input leg. Instead, it equals the difference between the voltage at the gate terminal and the resistor-governed voltage at the source terminal: $V_{GS} = V_{signal} - V_R$.

Let's assume that the input signal (V_{signal}) shoots up from 0 V to 5 V. Initially, the voltage across the resistor is zero, so V_{GS} is effectively equal to 5 V, and the transistor begins to conduct. This causes the current flowing through the 10 kΩ resistor to ramp up; once the current reaches 300 µA, the resulting voltage across the resistor is $V_R = 3$ V. At that point, V_{GS} is brought down to the threshold of conduction ($V_{signal} - V_R = 5\text{ V} - 3\text{ V} = 2\text{ V}$), and no further current (or V_R) gains are possible.

This is a negative feedback loop: Any deviation from the equilibrium is counteracted. The current can't increase because that would put V_{GS} below V_{th} and thus pull the current back down. It also can't decrease because that would reduce V_R, sending V_{GS} higher and letting more electrons through. There is only one possible outcome: V_R must track $V_{signal} - V_{th}$ (Figure 14-4).

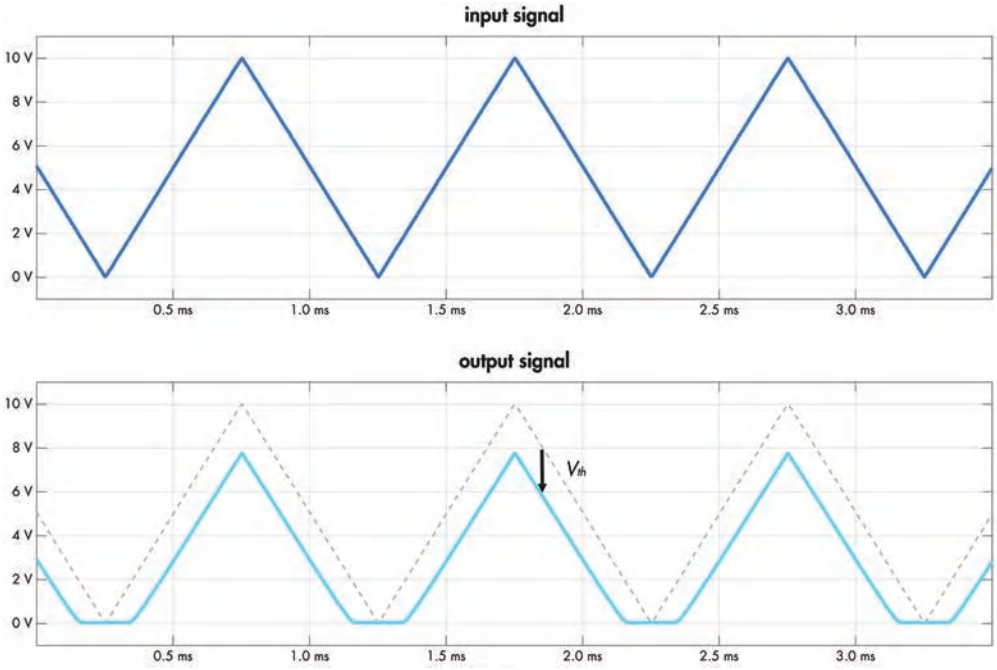


Figure 14-4: The behavior of an n-channel voltage follower, showing the V_{th} *offset between the input and output voltages*

In a MOSFET, the gate is insulated from the conductive path and presents itself as a small series capacitance, so at low frequencies the loading of the input signal is practically nil. On the output side, the current that can be supplied to a ground-connected load is limited only by the physics of the transistor and can be easily scaled to tens or even hundreds of amps.

Yet not all is roses: For high-current applications, the required V_{th} will diverge from the figure in the spec. Recall that V_{GS} is a lever that sets the maximum current; the number given in the datasheet is the lever position needed to admit a current in the vicinity of 1 mA. If we need to supply much more, the lever will have to be pulled harder—a higher offset voltage. For loads with a constant current draw, a different offset can be accounted for with ease, but if the demand fluctuates over time, our voltage follower is bound to become less linear.

In such a case, a circuit constructed with an NPN bipolar junction transistor might be a somewhat better option. These transistors are current-controlled, so in theory there should be no drift in the offset voltage at all. In practice, the transistor's forward-biased internal diode in the B-E path has a voltage drop that depends to some extent on current we're pushing through. However, the diode's V-I curve is fairly steep, so across a wide range of currents the shifts in V_{BE} are comparatively small (compare Figures 10-12 and 10-17 in Chapter 10).

An equivalent BJT circuit will exhibit lower input impedance, although the amount of current flowing into the base terminal is neatly self-limiting. That's because the negative feedback mechanism lowers V_{BE} until just enough current is flowing so that when the value is multiplied by h_{fe}, the resulting collector current produces the right V_R.

I encourage you to set aside some time and experiment with the n-channel and NPN version of this circuit, as well as their p-channel and PNP mirror images. Observe what happens to the voltage offset if you lower the resistance from 10 k Ω to 470 Ω. Also

try replacing the resistor with a good-sized capacitor—say, 100 µF. How does it affect the circuit's behavior and why?

As you play with the circuit, you might identify another flaw: It has an asymmetric ability to supply currents. In the n-channel or NPN case, if the voltage on the output leg is lower than the target, the transistor turns on and can supply a fair amount of juice. But if the voltage on the load side is too high, the transistor remains off and the problem needs to resolve on its own. A dummy resistor to ground can sink some current, but in a practical design it's not much.

For some applications this doesn't matter; for others it might. If the circuit is used to charge a capacitor, for example, the voltage will remain stuck until the charge bleeds off by some other means. If it's used to drive a speaker, it will be able to move the diaphragm with force in only one direction, producing muffled sound. The solution to these problems is our next circuit: a push-pull driver stage.

Push-Pull Stage

A *push-pull* circuit is designed to work similarly to a single-transistor follower, but to drive loads symmetrically. As shown in Figure 14-5, it features two complementary transistors driven by the same input signal. The top transistor is an n-channel voltage follower; the bottom is its p-channel complement.

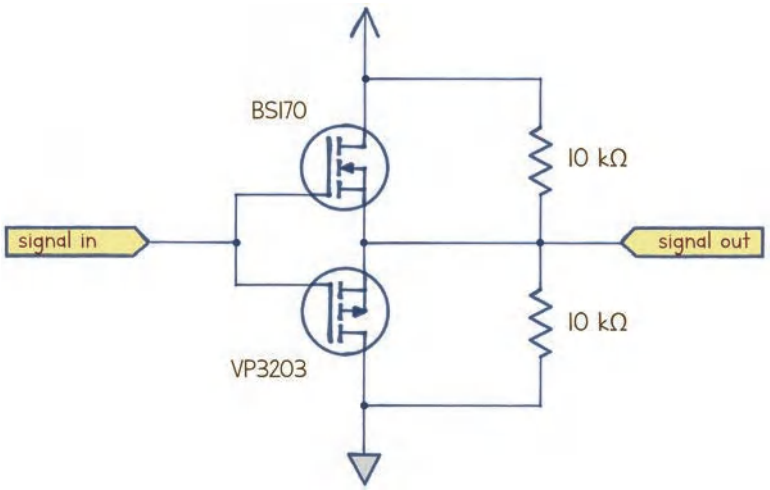


Figure 14-5: A sketch of a push-pull circuit. The resistors are used to keep the output voltage predictable when both transistors are off.

If the input signal exceeds the voltage present on the output leg by more than V_{th}, the upper transistor turns on, connects the output to the positive supply, and pulls the voltage up. If the input signal drops more than V_{th} below the output, the bottom transistor kicks in and connects the output to the ground rail.

The transistors are never turned on at the same time, but when the output voltage is within $+/-V_{th}$ of the input, both remain off. This dead zone produces what's known as *crossover distortion*, shown in Figure 14-6 as the flat line segments. The distortion will look more chaotic if we remove the two 10 kΩ resistors that force the output voltage to sit at a midpoint when the circuit is idle. Without the resistors, the output voltage would be dictated by the small differences in the "off" characteristics of the two transistors—or by other factors beyond our control.

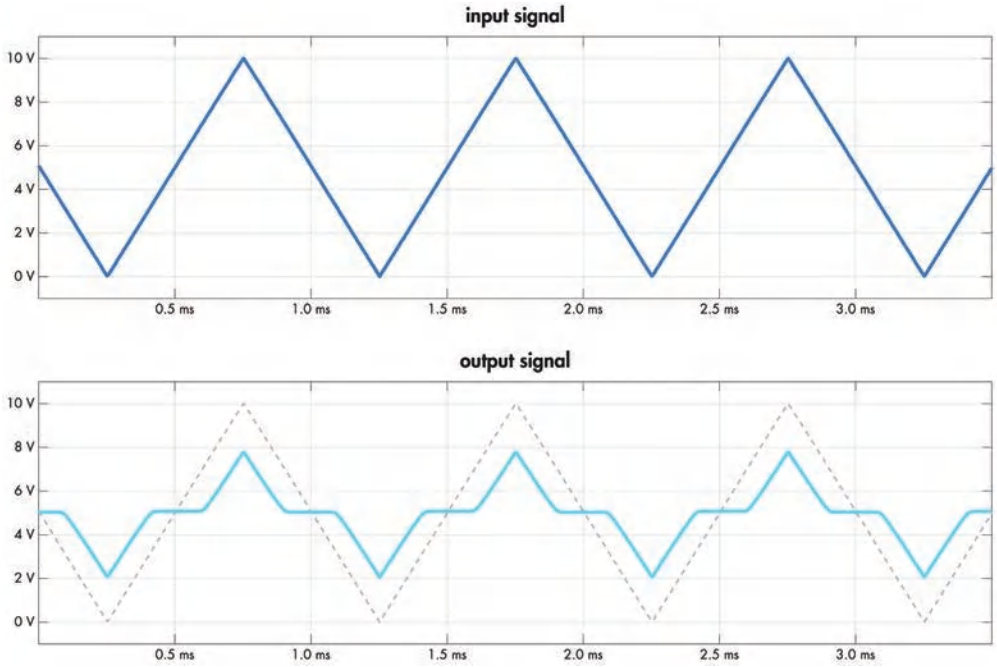


Figure 14-6: Crossover distortion in a MOSFET push-pull stage

To fix this distortion, we would need to split the signal into two parallel paths, one offset by $+V_{th}$ and the other offset by $-V_{th}$.

Current Mirror

A *current mirror* can be thought of as a counterpart to a voltage follower: It allows us to take the current flowing through one branch of a circuit and have it manifest in another branch. More fundamentally, the mirror is also an essential building block for circuits that supply a constant current no matter the impedance of the connected load. As discussed in Chapter 7, these current sources find specialized uses every now and then.

Earlier in the book, I remarked that transistors are best modeled as current limiters; in the case of MOSFETs, we use V_{GS} to set the maximum current that can flow through the channel. In contrast to an adjustable resistor, the admitted current has relatively little dependency on the applied main-path voltage (V_{DS}), at least with moderate currents where the inherent resistance of the semiconductor substrate doesn't play a dominant role.

It follows that if we want to supply a specific current to a load, we could recycle the switch (common-source) architecture and supply the transistor with a precisely trimmed gate voltage. The problem with this approach is that the V_{GS} to I_D curve of a MOSFET is steep and unpredictable. It varies from part to part and changes with temperature, so even if we dial in the voltage perfectly, the device may drift off the target due to internal heating.

The current mirror, shown in Figure 14-7, is a better solution. The circuit has two parallel branches with two identical transistors; for now, let's focus just on the transistor and resistor on the left.

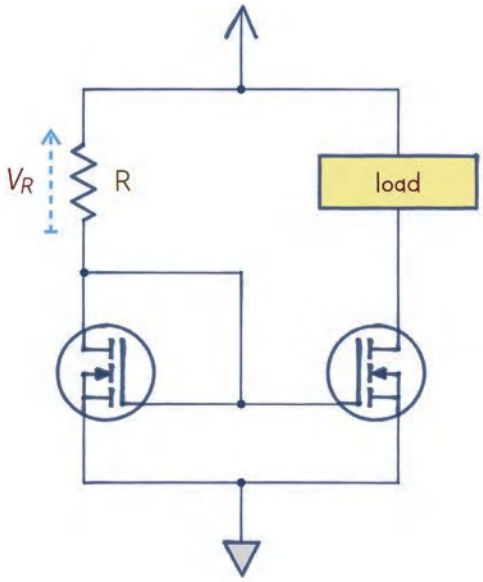


Figure 14-7: A simple current mirror

When the circuit is first powered on, the transistor on the left is in an "off" state. This means that there's no current flowing through the resistor, so V_R = 0 V. The bottom terminal of the resistor is connected to the gate of the transistor, making V_{GS} equal to V_{supply} and prompting the transistor to conduct. As a consequence, V_R begins to rise, yet the voltage across the resistor can't exceed $V_{supply} - V_{th}$ because that would put V_{GS} below the conduction threshold—another form of negative feedback loop.

The equilibrium for the circuit is $V_R \approx V_{supply} - V_{th}$. If we know the value of the resistor, we can calculate the current that must be flowing through the left branch of the circuit:

$$I = \frac{V_R}{R} = \frac{V_{supply} - V_{th}}{R}$$

Note that the solution doesn't depend on the exact shape of the device's V_{GS} to I_D curve. It depends only on the supply voltage, the value of the resistor, and the transistor's ballpark threshold voltage.

The *coup de grâce* is that we also have a second, matched transistor on the right—and it's subjected to the same gate voltage. This causes the right branch to admit the same current as determined by the preceding equation. Self-heating effects cancel out because both transistors will be at roughly the same temperature; it helps if they're physically close too.

Basic Amplifiers

In contrast to followers, amplifiers provide signal gain—that is, they boost the amplitude of an input waveform. Amplification is essential for interpreting the readings from many types of analog sensors, for deciphering faint radio transmissions, and more.

Inverting Amplifier

The simplest single-transistor amplifier circuit is an *inverting amplifier*, shown in Figure 14-8. This topology isn't meant to drive substantial loads; it only serves to boost the amplitude of an input waveform.

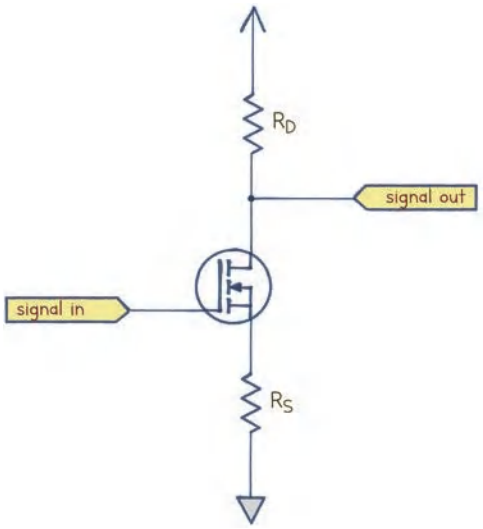


Figure 14-8: A rudimentary voltage amplifier

The layout is a minor riff on the voltage follower architecture. As before, the transistor admits just enough current to produce a voltage in the vicinity of $V_{signal} - V_{th}$ across the bottom resistor (R_S):

$$I = \frac{V_{signal} - V_{th}}{R_S}$$

This time, the same current also flows via the drain-side resistor (R_D), inducing a voltage drop that can be determined from Ohm's law:

$$V_{R_D} = I \cdot R_D = (V_{signal} - V_{th}) \cdot \frac{R_D}{R_S}$$

In other words, the drop is proportional to the input signal, minus V_{th}, multiplied by the ratio of resistances. Because R_D is placed between the positive supply rail and the output leg, the actual output voltage is inverted and can be described by the following formula:

$$V_{out} = V_{supply} - V_{R_D} = V_{supply} - (V_{signal} - V_{th}) \cdot \frac{R_D}{R_S}$$

Figure 14-9 shows a real-world demonstration of the amplifier configured for 2× signal gain ($R_D = 2 \cdot R_S$). The input signal is centered around 3 V; this bias keeps it above V_{th} for the n-channel MOSFET.

The supply voltage is 10 V, so from the aforementioned formula, the circuit produces a signal with a midpoint of 10 V − 2 · (3 V − V_{th}), or about 8 V. As expected, the amplitude is doubled, from 1 V to 2 V peak-to-peak, and the polarity of the waveform is flipped.

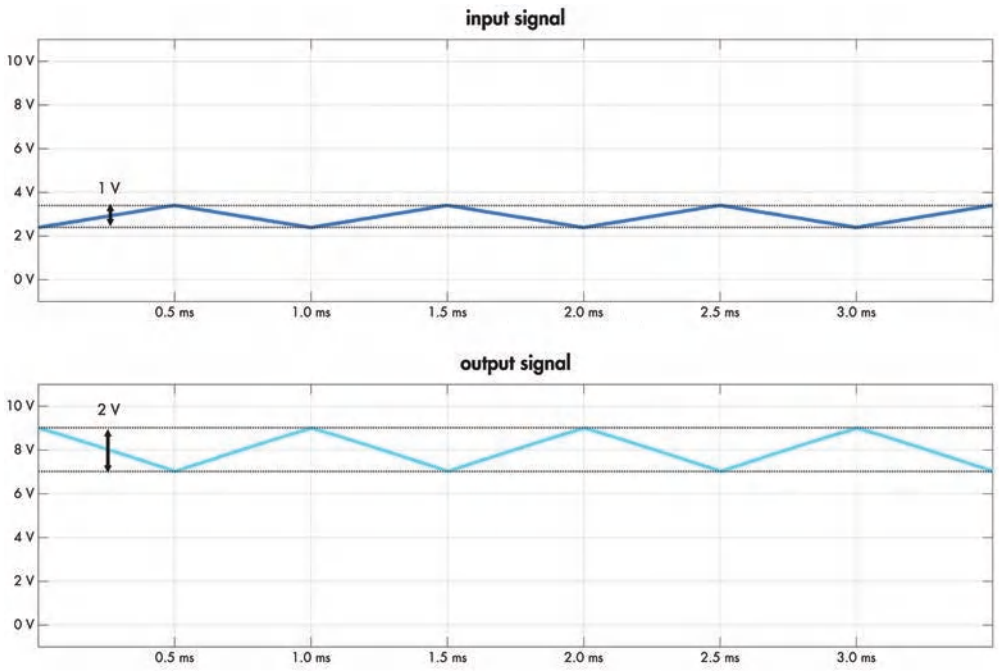


Figure 14-9: The behavior of a simple voltage amplifier using a BS170 transistor; V_{supply} *= 10 V,* R_D *= 100 kΩ,* R_S *= 50 kΩ*

There's an interesting caveat with this architecture. In the resistor–transistor–resistor path of this circuit, voltage drops develop across each of the three components. These drops must all be positive (or at best, zero) because none of the components can act as a source of electromotive force. If we take voltage measurements at four points along this path, we can write the following: $V_{supply} \geq V_D \geq V_S \geq 0$ V.

To understand what this means, note that the voltage on the drain terminal (V_D) is also the output voltage, and the voltage on the source terminal (V_S) equals the drop across the R_S resistor—which is always the input signal minus the transistor's threshold voltage. In other words, the amplifier's output range is limited: If we want to accommodate an input signal that swings to 3 V, then for the voltage follower section to work as intended, the voltage across R_S must be able to rise to 3 V − $V_{th} \approx$ 1 V. This means the circuit's gain must be capped to keep V_{out} above 1 V—limiting the amplifier's output range.

Another limitation of the circuit is that it amplifies its own nonlinearities. The current flowing in the circuit varies depending on the input voltage, shifting V_{th} up and down by a small amount. For a standard voltage follower, the issue is barely perceptible unless we're driving substantial loads. For an amplified variant, the imperfections become evident on their own, especially at gains in excess of 50×. If you have a signal generator, you can readily observe this problem by constructing the circuit and then supplying it with a triangle wave.

Differential Amplifier

We've seen that the concept of negative feedback is quite useful for building well-behaved transistor circuits, but its basic implementation is strangely limiting: We don't get to choose what signal gets compared to what. We can eliminate certain issues but have no way to tackle others—for example, we can't easily compensate for

V_{th}-related voltage drops or post-amplification nonlinearities, because there's no way to make the transistor "look" at anything other than V_{GS}.

This observation provides a good segue into a very important transistor circuit that allows us to build more flexible feedback mechanisms: the *differential amplifier*. The simplest version of this architecture, shown in Figure 14-10, isn't meant to be a particularly good amplifier at all. Instead, it can be thought of as a voltage comparator. It essentially discriminates between three input conditions: $V_A < V_B$, $V_A = V_B$, and $V_A > V_B$.

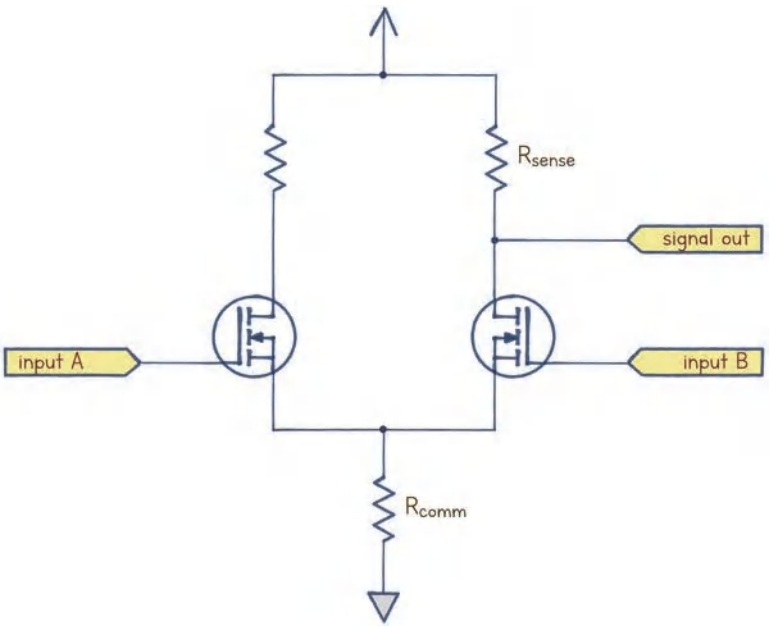


Figure 14-10: The simplest differential amplifier. The resistor in the left branch of the circuit is optional but improves linearity.

The circuit consists of two parallel branches with identical resistors and transistors; each branch acts as a voltage follower, admitting just enough current to develop a voltage across the bottom resistor that's equal to the transistor's input signal minus V_{th}. Crucially, however, the bottom resistor is shared by both branches, even though the input signals are separate. This means that if $V_A > V_B$, the transistor on the left is aiming for a higher resistor voltage than the one on the right, and therefore it remains on after the other cuts off. In that case, the entirety of the current for the bottom resistor is ultimately sourced via the left branch. Similarly, if $V_A < V_B$, the current flows exclusively through the right branch; and if $V_A = V_B$, the currents are split 50-50.

In the first case ($V_A > V_B$), there's no current flowing through the R_{sense} resistor, so the voltage on the output leg equals V_{supply}. In the second case ($V_A < V_B$), the entire R_{comm} current is sourced through this branch, so the output voltage drops to a minimum. Finally, if both branches are chipping in, the current through R_{sense} is one-half of the necessary total, so the output voltage is at a midpoint.

Although the concept seems elegant, the design suffers from a major flaw: The current that must flow through the bottom resistor isn't constant. Instead, it's proportional to the higher of the two input voltages. To illustrate the issue, let's assume that the common resistor is 10 kΩ, that V_A = 2 V, and that V_B = 3 V. In this scenario, the transistor on the right will source the entirety of the current necessary for

1 V (3 V − *Vth*) to appear across R_{comm}; this current works out to:

$$I_{comm} = \frac{3\,\text{V} - V_{th}}{10\,\text{k}\Omega} = 100\,\mu\text{A}$$

Now let's look at another case: $V_A = V_B = 4$ V. In this instance, the current is split equally across the two branches, and the sum that produces the necessary 2 V drop across the resistor works out to:

$$I_{comm} = \frac{4\,\text{V} - V_{th}}{10\,\text{k}\Omega} = 200\,\mu\text{A}$$

In both scenarios, the current flowing through R_{sense} is 100 µA. This makes the output voltage sensitive not only to the difference between the input signals but also to the absolute (common-mode) voltage on the input terminals. Further, we can't reliably distinguish between $V_A < V_B$ and $V_A = V_B$.

The problem can be worked around by restricting the range of input voltages to ensure that the drop across R_{comm} is always high and that the current fluctuates only a tiny bit. That said, a better solution is shown in Figure 14-11.

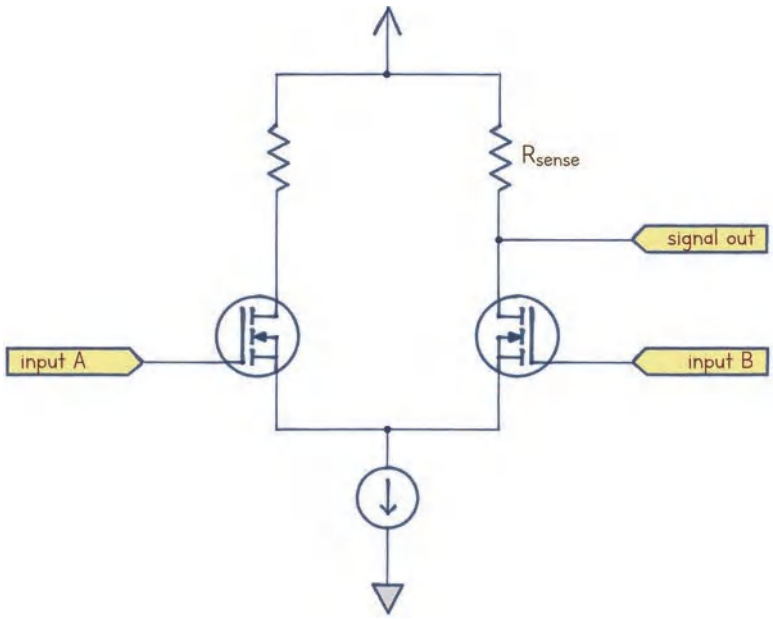


Figure 14-11: An improved differential amplifier with a current source in place of the common resistor

The symbol at the bottom is a current source, similar to the two-transistor current mirror circuit we constructed earlier in the chapter. An ideal current source always admits the same current, no matter the applied voltage; this means that the transistors can set their V_{GS} for free. In practice, ideal current sources don't exist, but we can get pretty close. In a real implementation, the current will fluctuate only a little across a wide range of common-mode voltages.

We'll devote the entire next chapter (and then some) to the applications of differential amplifiers. Before we move on, though, there's one more application of discrete transistors that's worth knowing about.

Relaxation Oscillator

Oscillators are the bread and butter of circuit design: They're used to provide clocks to digital circuits, to generate audio waves in old-school electronic instruments, to transmit and receive radio signals, and more.

Nowadays we usually don't build such circuits with individual transistors—it's a waste of mental energy, PCB space, money, and time. However, on the off chance you'll one day need to design or troubleshoot such a circuit, let's have a quick look at one of the simplest ways to accomplish the task.

To get started, imagine a machine equipped with a light sensor and a robotic arm. If it detects that the lights in the room are turned on, it strolls over to the wall switch and flips it to the "off" position. If it detects darkness, another part of its programming takes over and toggles the switch back on. This machine is destined to be stuck in an endless cycle of switch-flipping at a frequency dictated by how quickly it can process information and react.

At first blush, we should be able to replicate this principle with a transistor switch. Figure 14-12 shows an n-channel switch with a resistor connected between the transistor and the positive supply rail. If the MOSFET is off, there's no current flowing through the circuit, so the output leg—and thus the transistor's V_{GS}—moves toward the positive supply rail. In theory, this should turn the transistor on. Flipping on the switch should in turn connect the bottom terminal of the resistor to ground and thus cause V_{GS} to plunge, turning the circuit off again. Rinse, repeat.

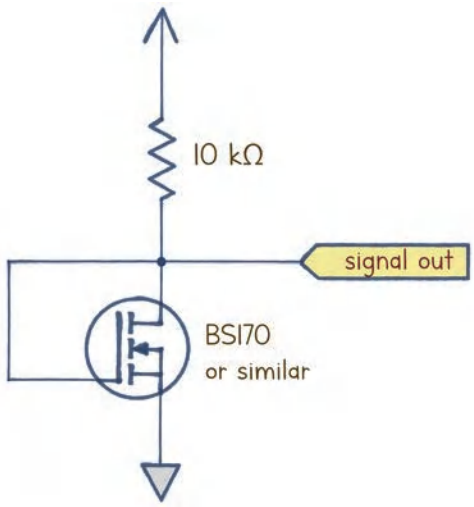


Figure 14-12: An ineffectual design for an oscillator

In practice, though, you're not likely to see oscillating behavior. As we've discussed before, the curve of V_{GS} to I_D is steep and unpredictable, but the MOSFET isn't a toggle switch. The circuit can settle on a stable equilibrium where the transistor admits just enough current to keep $V_{GS} \approx V_{th}$—a scenario akin to the light switch being half on and the room being bathed in dim light. For a 10 kΩ resistor in a 5 V circuit, this equilibrium current is around 300 µA. From there, if V_{GS} moves upward, the increase in resistor current pulls the signal back down; and if it moves down, the decrease in current pulls it up. It's another instance of a negative feedback loop: As long as the analog equilibrium voltage exists, the loop suppresses oscillations instead of driving them.

To fix this issue, we need to build a switch that has no stable equilibrium; this boils down to placing the on-to-off and off-to-on transition voltages some distance apart. This is known as the *Schmitt trigger*, shown in Figure 14-13.

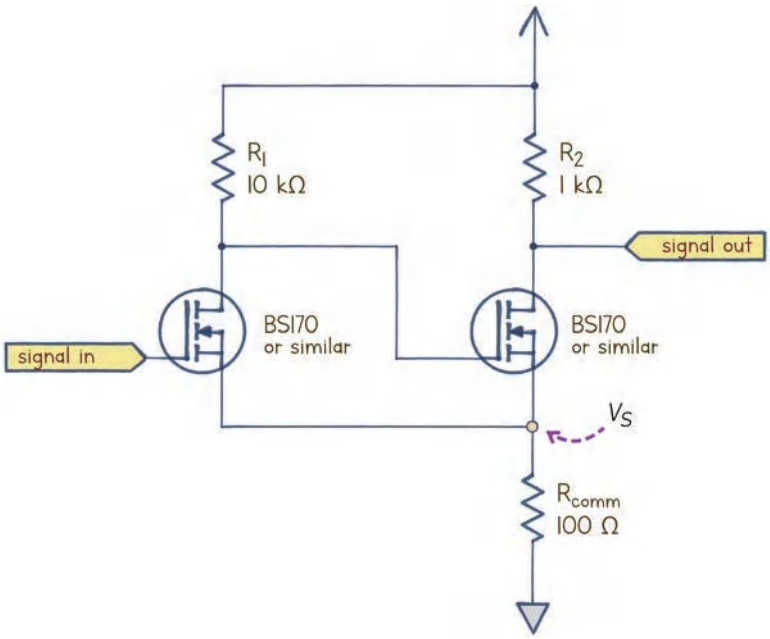


Figure 14-13: A Schmitt trigger circuit

To analyze the design, let's assume the circuit is running off V_{supply} = 5 V. If the input signal is 0 V, the transistor on the left isn't conducting, which sets V_{GS} for the output-side transistor to 5 V. This allows nearly arbitrary currents to flow through the right branch of the circuit, making that current path more or less equivalent to a two-resistor voltage divider. We can calculate the midpoint voltage of the divider as:

$$V_S \approx V_{out} \approx V_{supply} \cdot \frac{R_{comm}}{R_{comm} + R_2} \approx 450\,\text{mV}$$

This voltage is also propagated to the source terminal of the input transistor on the left. The actual V_{th} for the BS170 transistor is about 2.15 V, so for the left transistor to turn on, the input signal will need to exceed $V_S + V_{th} \approx 2.6$ V in reference to ground. When that happens, a large voltage drop appears across R_1, reducing the V_{GS} of the output-side transistor below the threshold of conduction and choking off the current in the right branch.

At this point, there's still current flowing through the common resistor on the bottom, but it's now shifting to the left branch. The left branch forms a new voltage divider; because R_1 has higher resistance than R_2, V_S is gradually reduced—effectively increasing V_{GS} for the input-side transistor and thus knocking it more firmly into conduction: positive feedback with no option to linger in a half-on state. Once the transition is complete, the voltage across the common resistor drops from 450 mV to about 50 mV. This means that although the left transistor first turned on when the input signal crossed 2.6 V in reference to ground, it won't turn off until the voltage drops all the way to 2.2 V—a 400 mV gap.

This circuit lets us build what's known as a *relaxation oscillator*. To do so, we only need to make two small tweaks. First, we need to loop an inverted output signal back

onto the input; the most intuitive way of doing this is to add a transistor in a switch-like configuration, akin to what we've talked about before (Figure 14-14, left). The output leg of this section is pulled toward the positive rail when the input voltage is low or toward ground if it's high.

Next, to set a sensible oscillation speed, we need to add a time delay, which can be accomplished by charging a capacitor through a resistor (Figure 14-14, middle). The resistor just needs to be large enough not to overload the inverter stage.

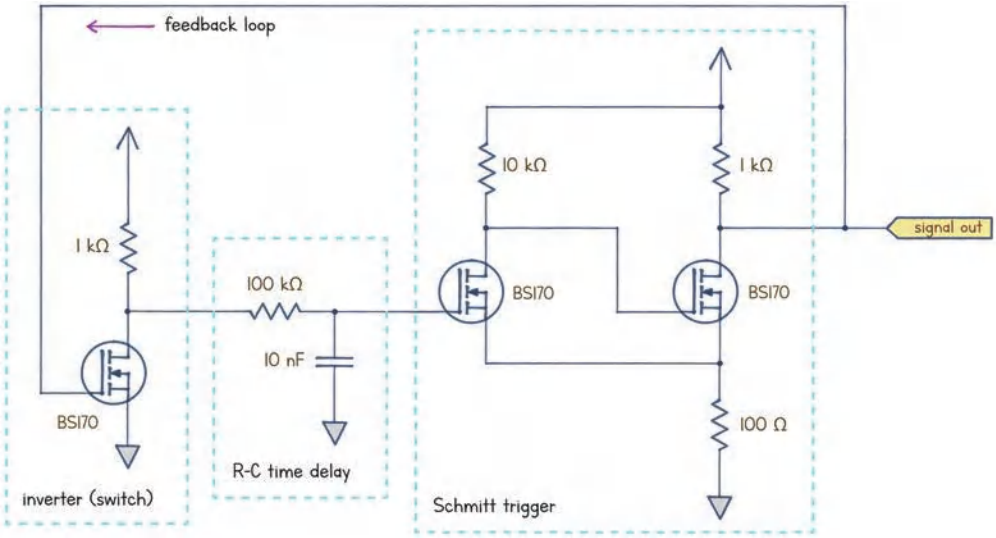


Figure 14-14: Turning a Schmitt trigger into a relaxation oscillator

For the component values shown in the schematic, the circuit should oscillate at a frequency of almost exactly 3 kHz when supplied with 5 V (Figure 14-15). The frequency is governed by how long it takes for the capacitor to move 400 mV between the two Schmitt threshold voltages: 2.2 V and 2.6 V.

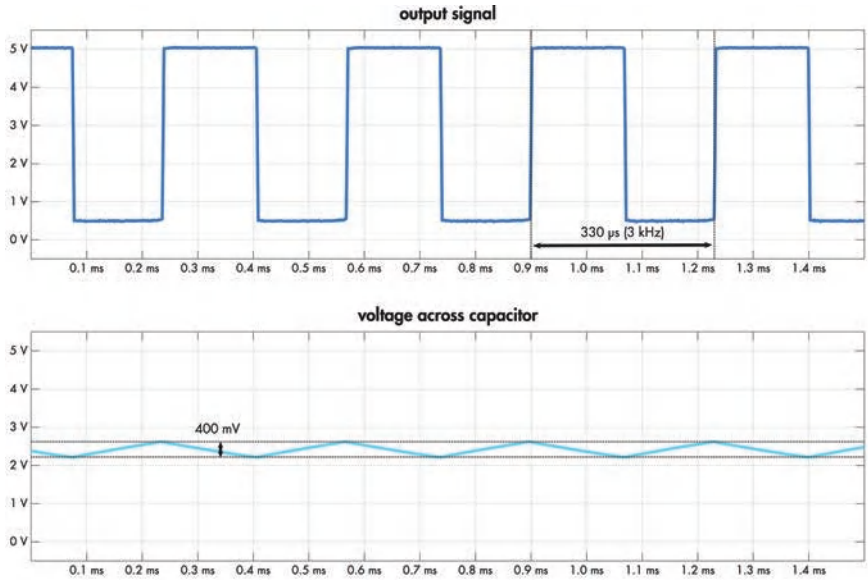


Figure 14-15: An oscilloscope trace for the relaxation oscillator circuit

Because the overall variation in capacitor voltage is small, we can squint our eyes and say that the parameter is nearly constant (about 2.4 V). When the 100 kΩ resistor is connected to the positive rail, we can approximate that $V_R \approx 5\text{ V} - 2.4\text{ V} \approx 2.6\text{ V}$. Conversely, when the resistor is connected to ground, we get $V_R \approx 2.4\text{ V}$. If the voltages across the resistor are nearly constant, so are the resulting capacitor currents:

$$I_{C\,(\text{charging})} \approx \frac{2.6\,\text{V}}{100\,\text{k}\Omega} \approx 26\,\mu\text{A}$$

$$I_{C\,(\text{discharging})} \approx \frac{2.4\,\text{V}}{100\,\text{k}\Omega} \approx 24\,\mu\text{A}$$

From the fundamental capacitor equation ($\Delta v = I \cdot t \,/\, C$), we can solve for the charging time needed to move the voltage by $\Delta v = 400$ mV: The result is about 154 µs for the charging period and 167 µs for the discharging period. The sum is 321 µs, corresponding to a frequency of about 3.1 kHz—pretty close to real life.

This approach isn't the only way to build oscillators. Another method is to take the ill-fated circuit showcased in Figure 14-12 and modify the feedback loop so that at some specific sine-wave frequency it produces a half-wavelength (180°) phase shift, effectively inverting the waveform. This way, at that exact frequency, the looped-back voltage is no longer suppressing departures from the equilibrium; instead, the signal turns into positive feedback that induces oscillations and causes their amplitude to grow.

One way to achieve this frequency-specific signal inversion is to cascade R-C or L-C filters in series. Each filter adds its own phase shift, and at some specific frequency the shifts should add up to 180°. That said, with discrete transistors the circuit is tedious to design and finicky to operate: A fair amount of math is needed to get the voltage offsets and signal gains just right. Instead of subjecting ourselves to unnecessary pain, we'll revisit this class of oscillators once we have a better abstraction in place. The abstraction we're after is the operational amplifier, coming up next.

15

OPERATIONAL AMPLIFIERS



If there's one takeaway from Chapter 14, it's that transistors are capable but tricky to use. Indeed, many fundamental transistor circuits are works of art, requiring a stroke of genius to devise and then a fair amount of effort to explain.

In this chapter, we'll take a look at *operational amplifiers (op-amps)*, a class of standardized integrated circuits that enable engineers to perform the same tasks but in a more straightforward way. A relatively simple integrated circuit, an op-amp consists of several dozen components etched onto a silicon wafer and then sealed inside a plastic package. The chip's internals are worth examining, but the whole idea is that you can use an op-amp as a near-ideal, universal amplifier without needing to think about what's inside.

Since this is our first encounter with an integrated circuit in the book, we'll take it slow. First, we'll briefly look at the chip's internal architecture, pinout, and electrical characteristics. Then, we'll explore a variety of practical circuits that illustrate the versatility of this design. As we progress through the chapter, you might find it useful to grab a datasheet for a sample op-amp and see if you can decipher what it says. The datasheets for most analog chips follow a common layout, so it's good practice for your own projects down the line.

Op-Amps at a Glance

An op-amp consists of three main stages. The first stage is a differential amplifier, functionally similar to the basic design outlined in Chapter 14. The differential stage in a modern op-amp is often constructed with p-channel MOSFETs (or PNP bipolar transistors) to avoid the V_{th}-related dead zone that was present in our n-channel design when the inputs hovered near 0 V. The switch to p-channel transistors doesn't

eliminate the dead zone, but it puts it near the positive supply rail, where it's less likely to interfere with weak signals we wish to amplify.

The design of the differential stage can be further improved by adding a complementary n-channel (or NPN) input stage designed to take over when the primary differential pair cuts off. This eliminates the V_{th} dead zone, although it tends to introduce a slight amplification error near the transition point because the complementary stages can't be matched with perfect accuracy. Another approach is to add an internal capacitor-based charge pump (see Chapter 16) that produces a higher supply voltage for the differential amplifier, eliminating the dead zone with no handover artifacts. Either way, the result is a chip with rail-to-rail input capability—a property usually spelled out in the datasheet or inferred from the chip's stated input voltage range (V_I or V_{CM}).

The output from the differential stage is routed to a simple secondary amplifier that further amplifies the delta between the two input voltages. The resulting end-to-end gain (A_{OL}) is typically between 100,000× and 1,000,000× (that's 100–120 dB in decibel-speak). Finally, the amplified signal is routed to a carefully biased push-pull stage that allows the chip to deliver respectable currents to connected loads. General-purpose op-amps can usually deliver between 20 and 200 mA, but specialized chips are available that can supply 5 A or more.

From this description, it follows that an ideal op-amp performs a simple mathematical operation: It calculates a difference between two input voltages (customarily denoted as V_{in+} and V_{in-}), multiplies that difference by a large factor, and outputs the amplified result so that the zero point lies halfway between the supply voltages (V_{mid}):

$$V_{out} = V_{mid} + \left(V_{in+} - V_{in-}\right) \cdot A_{OL}$$

In a single-supply circuit, V_{mid} equals $V_{supply} / 2$. The amplifier's output voltage also can't swing beyond the supply rails, so in the single-supply scenario V_{out} is in the best case constrained to a range between 0 V and V_{supply}. The actual output range may be a bit less, depending on the design of the output stage.

Because A_{OL} is extremely large, the chip essentially outputs 0 V when the differential input is negative ($V_{in+} < V_{in-}$) and immediately switches to V_{supply} when the differential input is positive ($V_{in+} > V_{in-}$). There is a microvolt-range transition region near $V_{in+} \approx V_{in-}$; in that region, tiny differential inputs are amplified linearly to a full range of intermediate voltages, but the behavior is quite hard to observe with standard tools. Even shorting the op-amp's inputs won't necessarily get you there: Small errors in the differential stage may place the zero point at a slight offset, given as V_{OS} in the spec, which varies from chip to chip and may amount to +/−100 µV or more.

Keeping this in mind, Figure 15-1 shows the usual behavior for V_{in-} fixed at 2.5 V and a V_{in+} signal that moves up and down slightly.

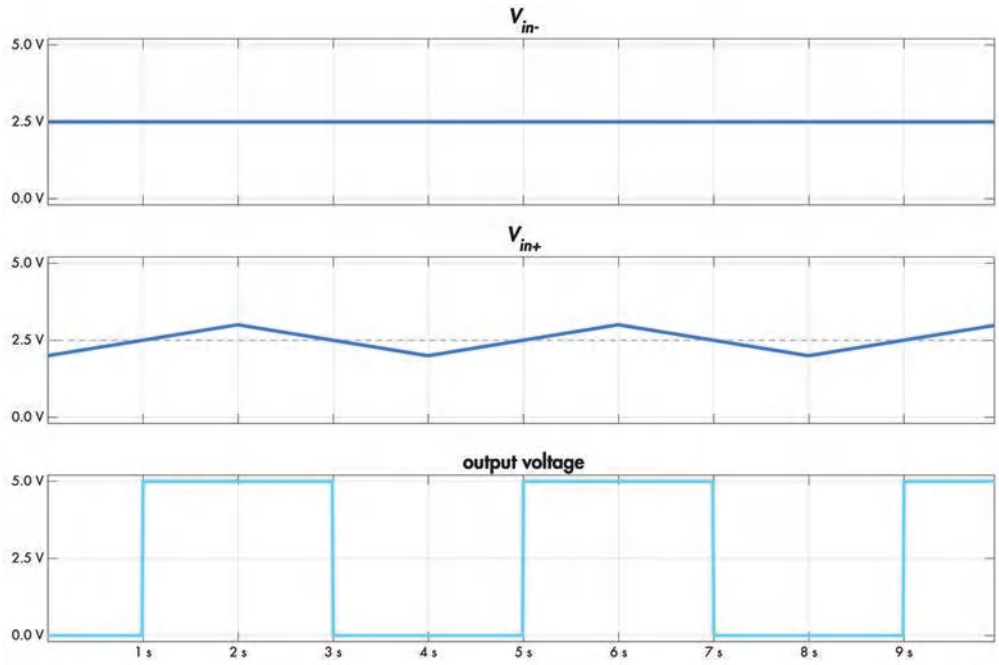


Figure 15-1: A simulation of the intrinsic response of an op-amp powered from a single 5 V supply

We call V_{in+} the *noninverting input* because increasing this voltage ultimately drives V_{out} higher—that is, the two are positively related. By extension, V_{in-} is the *inverting input* because an increase in the voltage on this terminal causes the output of the differential amplifier to decrease.

IC Packages and Symbols

Op-amps come in a variety of form factors, but they're often available in one of two eight-pin packages: SOIC-8 (for surface mounting) or DIP-8 (for through-hole applications). We'll discuss chip packaging further in Chapter 28, but the bottom line is that these two packages are similar except for size and that each contains one or two amplifiers. The pinout might vary, but it will be spelled out prominently in the datasheet and is usually one of the two options shown in Figure 15-2.

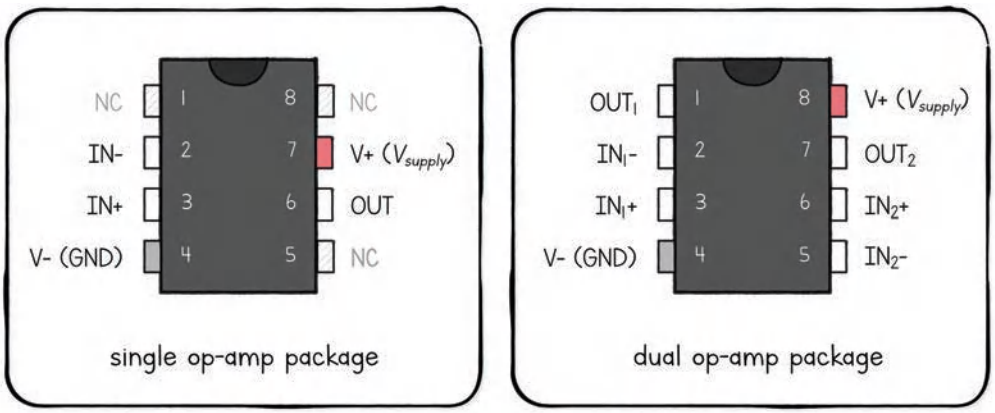


Figure 15-2: Two common op-amp pinouts (as seen from the top, legs pointing down). Pins marked "NC" are not connected.

If only one amplifier is used in a dual-amplifier chip, the inputs of the unused op-amp should be connected to ground—or better yet, the noninverting input should be connected to ground and the inverting input wired to the output pin. This prevents the inputs from picking up and amplifying radio interference, which would waste electricity and create unnecessary electrical noise that could propagate to other parts of the circuit.

Note the pin numbering scheme; it's common to nearly all integrated circuits. We start in the top-left corner and move counterclockwise. The correct orientation of the chip may be marked with a notch on top, a dot or clipped corner near pin 1, or a beveled left edge for surface-mount devices.

We have already discussed the meaning of V_{in-}, V_{in+}, and V_{out} signals. In the datasheet, the corresponding pins should be labeled clearly. As for the power supply, the positive supply pin is most commonly denoted "V+" or "+V," but you may also see conventions such as "Vdd" or "Vcc." The negative rail may be labeled "V−," "−V," "GND," "Vss," or "Vee." These conventions are common to all chips; the historical reasons for the variety are not worth dwelling on—they go back to the early days of integrated circuits and carry no consistent meaning today.

Some amplifier packages may also include pins with special functions. The most common example is a signal that allows the amplifier to enter a low-power sleep mode when not in use. This leg may be labeled "$\overline{\text{SHDN}}$" (shutdown); the overline indicates that the function is activated by applying a low voltage to the pin, so for normal operation, the pin must be held positive.

Although op-amps come in an IC form factor, it's customary to represent each amplifier with a standardized symbol rather than as a simple box. The symbol is shown in Figure 15-3. For brevity, the V+ and V− pins are often omitted in single-supply circuits; it is assumed that they're connected to the supply.

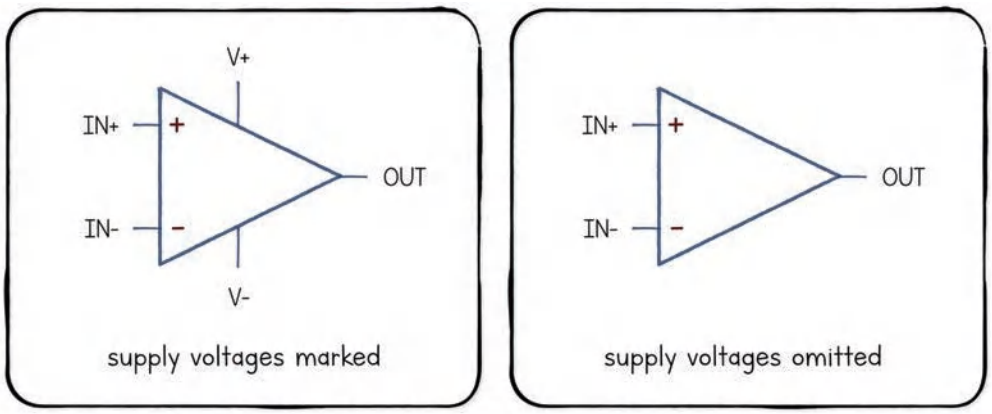


Figure 15-3: A schematic symbol of an operational amplifier, with and without supply pins

Now that we have a sense of what an op-amp looks like and does, let's discuss some of the chip's specs.

DC Characteristics

The most fundamental op-amp spec is the chip's supply voltage. Many modern rail-to-rail amplifiers are designed to operate at low voltages, usually between 2 and 5 V. Higher-voltage variants are available; for example, amplifiers designed for audio applications can usually handle up to 36 V. In the datasheet, the minimum and maximum supply voltage for proper operation is usually given in the "electrical characteristics"

section. This should not be confused with the "absolute maximum ratings" section, which specifies higher voltage limits that, if exceeded, could damage the chip.

Part of the reason higher-voltage chips exist is that early op-amp technology required input and output signals to stay at least 1 to 2 V away from the supply rails. Designers became accustomed to powering op-amps with dual supplies, such as −15 V and +15 V, so even the crotchetiest op-amp could handle signals near 0 V. With modern designs, such range limitations are less common, although they persist in certain ultrafast or high-power chips.

After supply voltage, the next parameter to inspect is the maximum output current. As discussed earlier, this is usually between 20 and 200 mA. The number in the spec describes short-circuit current (I_{SC}); at that point, the amplifier keeps doing its best but no longer controls the voltage on the output pin. Alternatively, the limit can be modeled by imagining the op-amp as a perfect signal source in series with an output resistance; this theoretical resistance is given in the spec as Z_O.

Many modern op-amps can handle input signals that swing slightly below the negative rail. In rail-to-rail input designs, the range also extends slightly past the positive rail. This voltage range is typically labeled V_I in the datasheet. Some chips impose an additional restriction on the maximum differential input voltage—that is, the delta between V_{in+} and V_{in-}. This usually stems from internal, cross-connected protection diodes that straddle the input terminals and conduct if the difference exceeds about 600 mV.

As noted earlier, an op-amp can't produce outputs lower than the negative supply voltage or higher than the positive supply voltage. In devices advertised as having a rail-to-rail output stage, V_{out} can usually approach the rails within 10 to 100 mV. In some older or high-power amplifiers, an output-side gap of 1 to 2 V or more may exist. The amplifier's output voltage swing is labeled V_O.

The inputs of an ideal op-amp would have infinite resistance. Amplifiers with FET input stages come close, with values typically expressed in gigaohms or teraohms. In chips with BJT inputs, the resistance to ground (Z_{ICM}) is also typically high. However, because of the current-driven differential stage, the isolation between the two inputs (Z_{ID}) isn't perfect—usually in the low megaohms range. A small input bias current (I_B) must flow for proper operation and may increase when the input voltages diverge. For faint signals, FET amplifiers with high input resistance can be a safer choice.

AC Characteristics

For most applications, op-amps are selected based on the static DC parameters outlined earlier. When working with fast-changing waveforms, however, frequency-dependent parameters come into play. We'll review them now because they offer interesting insights into the limitations of these chips, though none will be important for the first op-amp circuits you build.

To begin, consider input impedance: In addition to static resistance, capacitive coupling exists between the input pins and between each pin and ground. Op-amp datasheets list these parasitic capacitances as part of the Z_{ICM} and Z_{ID} values. The capacitances are small—usually just a few picofarads—so they have negligible impact on signals in the kilohertz range. A 2 pF input capacitance to ground, however, acts as a fairly substantial 8 kΩ shunt for a 10 MHz sine wave.

In part because of similar parasitic capacitances inside the chip, real-world op-amps can't respond arbitrarily quickly to input signals. This limitation causes a gradual decrease in apparent gain (A_{OL}) as sine-wave frequency increases. The effect is

accompanied by a growing phase shift between the input waveform and the voltage that appears on the amplifier's output pin.

This phase shift poses an interesting problem. As we'll discuss shortly, practical op-amp circuits often rely on negative feedback loops to stabilize the output voltage. This is a more sophisticated version of the single-transistor mechanism introduced in Chapter 14. Toward the end of that chapter, I noted that if the phase shift in a feedback loop approaches 180 degrees, the input waveform becomes a mirror image of the output. This turns intended negative feedback into positive feedback, driving oscillations instead of stabilizing the output voltage or controlling gain.

Because of this hazard, op-amp designers don't leave the A_{OL} curve to chance. Instead, in general-purpose (*unity-stable*) designs, they deliberately cause the gain to roll off to less than unity (1×) before the phase shift reaches the danger zone. Figure 15-4 shows a fairly typical example; the downward slope on the plot corresponds to gain halving every time the frequency doubles. The curve crosses unity gain at a point that still leaves a roughly 60-degree safety margin before complete phase reversal.

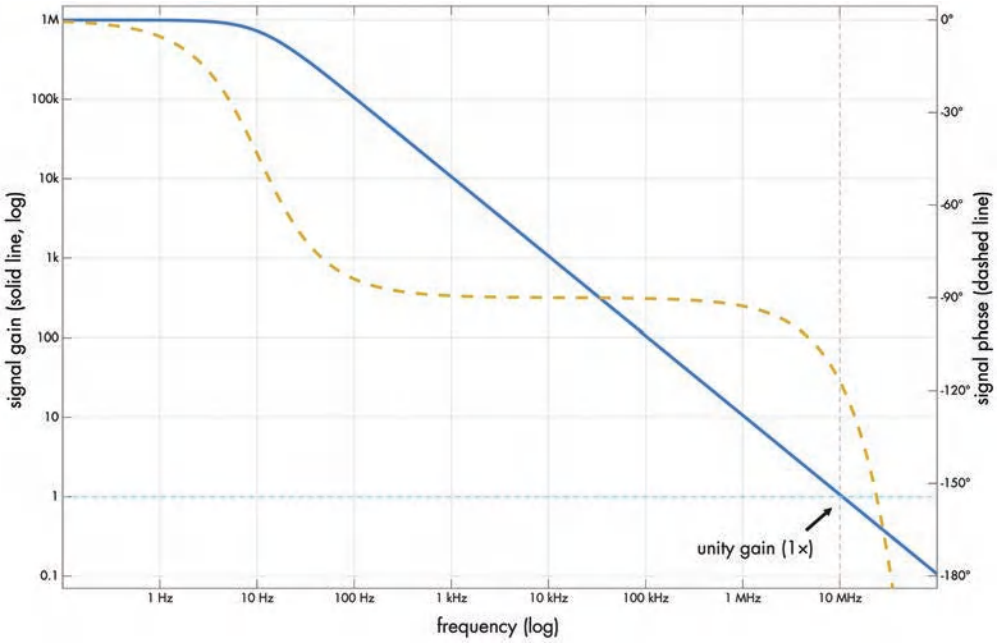


Figure 15-4: A model of an intrinsic frequency response curve for a typical 10 MHz op-amp

This example A_{OL} curve matches the frequency response of a single R-C low-pass filter (Chapter 12, Figure 12-11). The similarity is no accident: The rolloff is accomplished with an internal compensating capacitor. In some high-performance amplifiers, the compensation might be only partial, so the plots show additional kinks, especially toward the tail end.

The datasheet for every op-amp provides a similar frequency plot. In addition, the manufacturer specifies either *unity-gain bandwidth* or *gain-bandwidth product (GBP or GBWP)* as a quick component screening tool. Unity-gain bandwidth is simply the frequency at which A_{OL} drops to 1, describing roughly the highest practically useful operating frequency of the amplifier (10 MHz in Figure 15-4). The other value—GBWP—extrapolates the same characteristic by measuring A_{OL} at some other frequency f and multiplying the two values. If the attenuation curve is straight, both methods yield the same number. For example, in Figure 15-4, A_{OL} = 100 at f = 100 kHz, so the gain-bandwidth product is $A \cdot f$ = 10 MHz.

The frequency curve describes the amplifier's pass-through characteristics, but the chip's output stage also has a limit on how quickly the voltage can rise or drop. This is known as *slew rate* and is expressed in volts per second. To illustrate, a typical 10 MHz chip may have a slew rate of around 10 V/µs. An output stage with this spec can produce a 5 V voltage swing up to 1 MHz; at 10 MHz, the maximum peak-to-peak amplitude is 500 mV.

Another secondary performance constraint is the amplifier's *overload recovery time*: the time required for the output stage to return to linear operation after being driven fully into one of the supply rails (as we repeatedly did in Figure 15-1). Depending on the design, this can be anywhere from three to a hundred times slower than the chip's nominal reaction time. In modern FET amplifiers, it's typically close to the lower bound.

The limitations of an op-amp's response speed also extend to its ability to counter fluctuations in the supply voltage. The spec gives a *power supply rejection ratio (PSRR)*, but the value represents a best-case scenario. More insights can be gleaned from a plot of PSRR versus frequency, typically found in later datasheet sections; it usually shows a marked drop in noise immunity for supply ripple above 100 kHz. For stable operation, the supply voltage to an op-amp needs to be filtered—a single decoupling ceramic capacitor of 1 to 10 µF, placed across the supply rails and physically close to the chip, is typically sufficient.

A related characteristic, the *common-mode rejection ratio (CMRR)*, describes the differential stage's immunity to fluctuations in the common-mode voltage on the input pins (that is, when both input signals move in unison). This parameter is usually better controlled than PSRR.

Op-amps are also sources of wideband thermal noise, similar to resistor noise (Chapter 8) and measured as noise density, e_n, typically expressed in nV / $\sqrt{\text{Hz}}$. The noise model is input-referred, meaning a hypothetical noise source is placed in series with one of the inputs to an idealized, noise-free amplifier. For most modern chips, the thermal noise is tiny—usually comparable to a single 10 kΩ resistor—and thus inconsequential unless the rest of the circuit is designed with great care.

In addition to wideband thermal noise, op-amps exhibit low-frequency current fluctuations in the input stage, manifesting as *flicker noise* that dominates at frequencies below 1 kHz. These fluctuations are hard to eliminate but can be compensated for. The simplest (if crude) solution is to periodically disconnect and internally short the amplifier's inputs to measure the error. For ultraprecise applications, internally compensated "zero-drift" ICs are available, eliminating the need to build your own.

A Note on Old vs. New

The first monolithic op-amps hit the market in the 1960s. These chips were not particularly good. As hinted at earlier, they suffered from significant input and output voltage limitations. Worse, exceeding those limits often led to unexpected behaviors such as output voltage reversal or latch-up that required power-cycling the circuit. The chips were also plagued by crossover artifacts, poor noise performance, and more.

This is also when op-amp circuits were popularized among hobbyists, leading to a considerable body of endlessly copied designs anchored to Cold War–era relics such as the LM741, LM324, or LM358. This status quo is now further enshrined by AI-regurgitated advice filling up the internet.

Op-amps designed in the 21st century are better choices. You wouldn't want to browse the internet on a 1960s computer, and the same principle applies to using stone-age amplifier chips. For circuits powered from 5 V or less, a good dual-amp option is the MCP6022 from Microchip Technology. The chip offers 10 MHz unity-gain

bandwidth and suits most audio and sensing applications. A premium upgrade with twice the bandwidth is the OPA2323 from Texas Instruments. For higher-voltage applications, the OPA1656 is a superb choice.

With this advice out of the way, let's look at how op-amps are used in the real world.

Op-Amp Voltage Amplifiers

The discussion of op-amp dynamics earlier might leave you with the impression that these devices are even messier than transistors. To show that this suspicion is unfounded, consider the single-transistor voltage follower from the beginning of Chapter 14. When I introduced the circuit, I noted two flaws. First, the output voltage was always offset from the input by V_{th}. Second, because the shift between the input and output signals depended on the current we needed to supply to the connected load, the circuit's linearity wasn't great.

With an op-amp, we can make these problems go away.

Voltage Follower

Figure 15-5 shows a voltage follower implemented with a single operational amplifier. The circuit requires no external components other than a power supply and a piece of wire. The trick is simply to loop the output signal into the device's inverting leg.

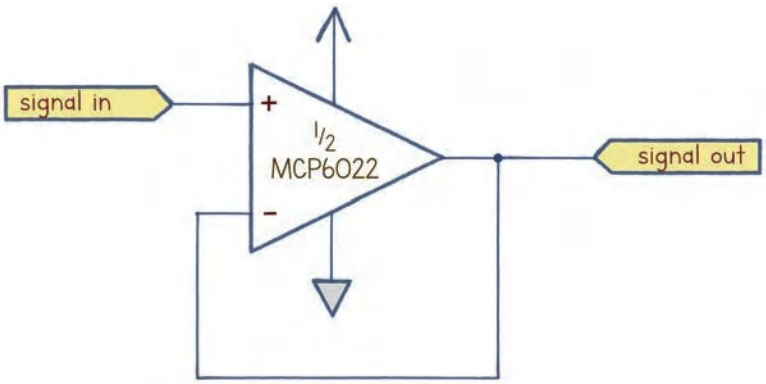


Figure 15-5: Op-amp voltage follower

To understand this architecture, we can examine what happens if the input voltage on the noninverting leg is markedly higher than V_{out}. Because the output signal is looped onto the inverting input of the amplifier, this creates an input condition of $V_{in+} \gg V_{in-}$, causing the output voltage to shoot up. Conversely, if the output voltage exceeds V_{in+}, the reverse occurs ($V_{in+} \ll V_{in-}$), pulling the output voltage down. This is a negative feedback loop. If there is a stable equilibrium point, it must be somewhere near $V_{in+} \approx V_{in-} \approx V_{out}$.

From our earlier discussion, recall that the op-amp calculates the difference between V_{in+} and V_{in-} and multiplies it by A_{OL}. To produce a full range of output voltages, the inputs thus can't be exactly equal; some small offset must exist. To prove that there is a stable solution, we can go back to the fundamental op-amp equation:

$$V_{out} = V_{mid} + \left(V_{in+} - V_{in-}\right) \cdot A_{OL}$$

In this case, because V_{in-} is equal to V_{out} and V_{in+} is the external input signal, the formula simplifies to:

$$\begin{aligned} V_{out} &= V_{mid} + \left(V_{signal} - V_{out}\right) \cdot A_{OL} \\ &= V_{mid} + V_{signal} \cdot A_{OL} - V_{out} \cdot A_{OL} \end{aligned}$$

Grouping all the V_{out} terms on the left and tidying up, we get:

$$V_{out} \cdot (1 + A_{OL}) = V_{mid} + V_{signal} \cdot A_{OL}$$

To solve for the output voltage, we simply need to divide both sides by $1 + A_{OL}$. The result is the following formula:

$$V_{out} = \frac{1}{1 + A_{OL}} \cdot V_{mid} + \frac{A_{OL}}{1 + A_{OL}} \cdot V_{signal}$$

At low signal frequencies, A_{OL} for the MCP6022 chip is about 1 million, so the first fraction is negligibly small and the second is almost exactly one (≈0.999999). In other words, a stable equilibrium exists; the output voltage follows the input with near-perfect fidelity.

Note that the circuit does nothing to the intrinsic gain of the chip; it remains equal to A_{OL}, and we want it to stay high for precision. At the same time, the circuit's signal gain equals 1: Waveforms pass through without changing their amplitude.

The only real trade-off in this design is that at high input frequencies, the chip's A_{OL} drops. If you refer to Figure 15-4, the available gain falls to about 10 for a 1 MHz sine wave. At that point, the differential nudges that piggyback on top of the "ideal" output signal will need to be much larger. For example, in a single-supply 5 V circuit, a signal that swings from 1 V to 4 V would produce the following output voltages:

$$\begin{aligned} V_{\text{lo}} &= \frac{1}{11} \cdot V_{mid} + \frac{10}{11} \cdot 1\,\text{V} \approx 1.14\,\text{V} \\ V_{\text{hi}} &= \frac{1}{11} \cdot V_{mid} + \frac{10}{11} \cdot 4\,\text{V} \approx 3.86\,\text{V} \end{aligned}$$

The resulting signal amplitude is about 10 percent off target. For optimal linearity, you typically want A_{OL} to be higher than 100 at the circuit's operating frequency, although few applications require millivolt-level precision combined with megahertz speeds.

The op-amp follower is an excellent circuit to build and experiment with. If you have a signal generator, you can use the setup to observe the slew rate limitation discussed earlier. Past a certain point, the op-amp's output stage will not be able to keep up, producing an attenuated ramp-up, ramp-down waveform similar to that shown in Figure 15-6.

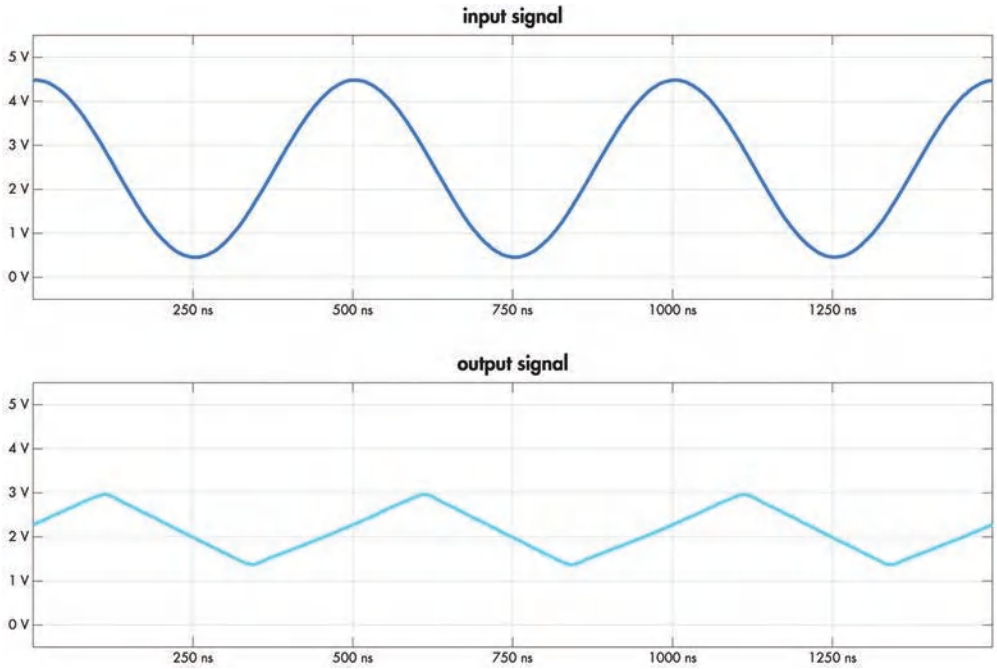


Figure 15-6: Slew rate limitation in an MCP6022 op-amp for a signal with a 4 V peak-to-peak amplitude and a frequency of 2 MHz

Noninverting Amplifier

There's an interesting observation we can make about the op-amp voltage follower: As long as a straightforward relationship exists between the output voltage and the feedback signal, the negative input of the chip doesn't need to be wired *directly* to the output pin. For example, if we tap into the output voltage only after the signal has passed through a forward-biased diode connected to a modest load, the op-amp equilibrium of $V_{in+} \approx V_{in-}$ will require V_{out} to settle about 600 mV higher to make up for the diode's voltage drop.

A variant of this trick is to use a resistor-based voltage divider to make the feedback voltage equal to an attenuated image of the output signal. The approach is outlined in Figure 15-7.

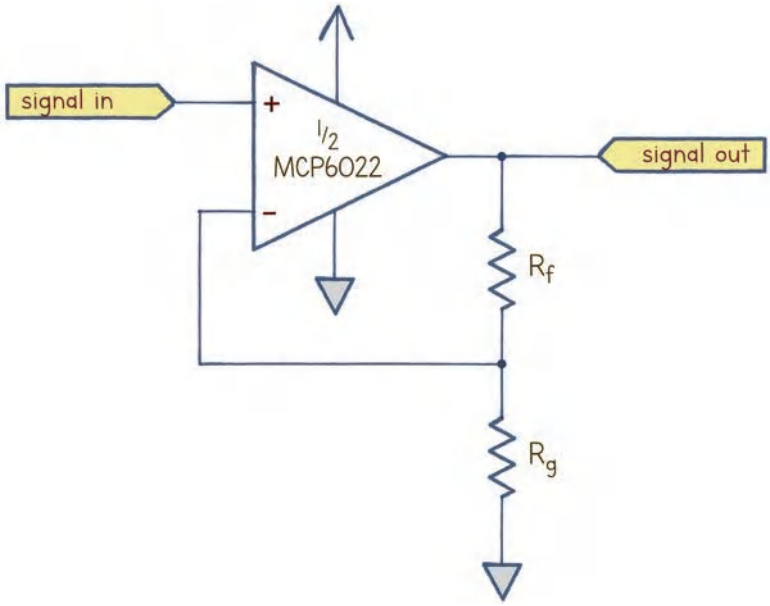


Figure 15-7: A noninverting voltage amplifier

The resistor labeled R_f is called that because it sits on the inner side of the feedback loop. The other resistor is labeled R_g because it connects to ground.

In this circuit, the voltage on the V_{in-} leg is a fraction of the output voltage, given by the standard voltage divider formula (Chapter 12):

$$V_{in-} = \frac{R_g}{R_f + R_g} \cdot V_{out}$$

You probably sense where this is going: In this configuration, the output voltage needs to swing proportionally higher to reach the equilibrium point of $V_{in+} \approx V_{in-}$. In effect, the output voltage becomes a multiple of the signal applied to the noninverting leg.

It helps to crunch the numbers to confirm. The $R_g / (R_f + R_g)$ fraction is always less than 1. Because it's constant for a given circuit, let's shorten it to β to keep the notation tidy. If we plug the resulting $V_{in-} = \beta \cdot V_{out}$ formula into the op-amp equation, we get:

$$\begin{aligned} V_{out} &= V_{mid} + \left(V_{signal} - \beta \cdot V_{out}\right) \cdot A_{OL} \\ V_{out} \cdot (1 + \beta \cdot A_{OL}) &= V_{mid} + V_{signal} \cdot A_{OL} \\ V_{out} &= \frac{1}{1 + \beta \cdot A_{OL}} \cdot V_{mid} + \frac{A_{OL}}{1 + \beta \cdot A_{OL}} \cdot V_{signal} \end{aligned}$$

At low frequencies, the first fraction is (once again) nearly zero, while the second equals roughly $A_{OL} / (\beta \cdot A_{OL}) = 1 / \beta$. This tells us that the output voltage is about the same as the input voltage multiplied by the inverse of β; the circuit's gain is:

$$A_{signal} \approx \frac{R_f + R_g}{R_g} = 1 + \frac{R_f}{R_g}$$

The gain we've implemented isn't quite free, however. Let's say that in the earlier voltage follower circuit, V_{in-} only needed to diverge from V_{in+} by +/−2.5 µV to achieve

a full range of output voltages—for a 5 V circuit at A_{OL} = 1,000,000, this sounds about right. If so, that 2.5 µV was also the magnitude of the unavoidable feedback nudges we'd see on the V_{out} pin.

If we modify the follower circuit by adding a voltage divider that attenuates the feedback signal by 50 percent, the output signal now needs to move +/−5 µV to produce the same +/−2.5 µV swing between V_{in-} and V_{in+}. As far as these feedback nudges are concerned, adding the divider had the same effect as buying a worse amplifier with an A_{OL} figure 50 percent lower.

The apparent reduction in A_{OL} has no direct effect on signal amplification. That parameter is set by the resistors, just as the gain of the voltage follower was set by a piece of wire bridging two of the chip's pins. Instead, what changes is the deviation from ideal output voltage. The ratio between the resulting output swing and the magnitude of feedback nudges is called the circuit's *loop gain*. In a voltage follower, loop gain equals A_{OL}. In this circuit, it's reduced in proportion to the loop attenuation ratio (β), which is the inverse of signal gain:

$$A_{loop} = \beta \cdot A_{OL} = \frac{A_{OL}}{A_{signal}}$$

In the frequency-domain view, this means the point at which the nudges get out of whack is reached proportionately sooner. In the voltage follower circuit, we calculated a 10 percent reduction in signal amplitude at 1 MHz. In a circuit with 2× signal gain, that same point will be reached at around 500 kHz. In essence, we lose the portion of the op-amp's A_{OL} curve that lies below the line representing the circuit's configured signal gain (Figure 15-8).

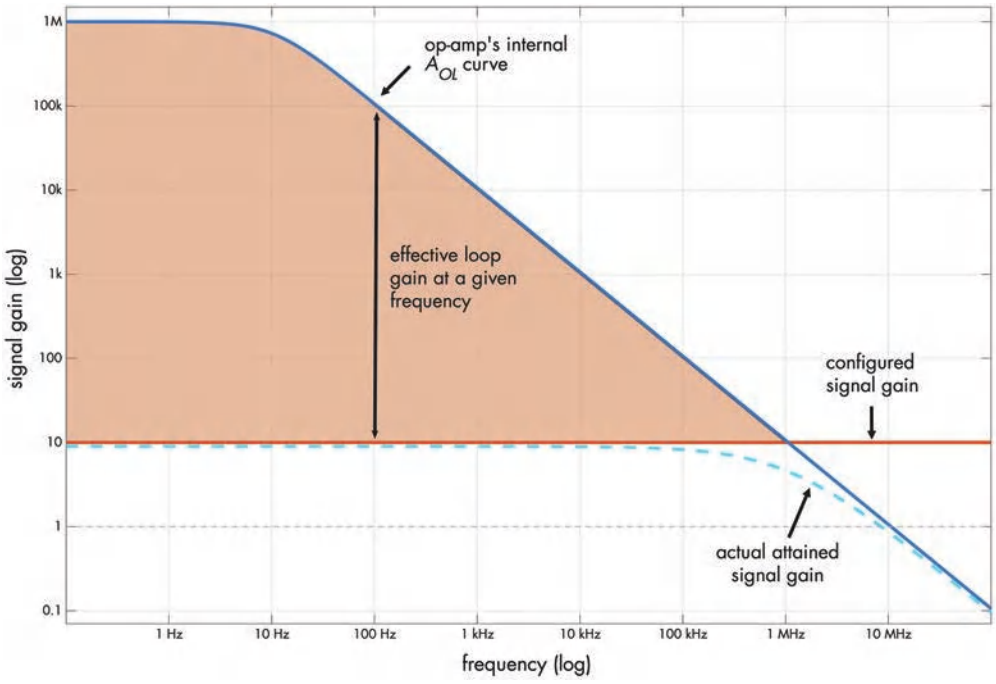


Figure 15-8: The impact of an amplification-setting voltage divider on the circuit's loop gain. The dashed line representing actual gain is slightly offset for readability.

In other words, in this and similar resistor-feedback circuits, maximum attainable signal bandwidth must be reduced proportionately to desired gain.

Inverting Amplifier

The inverting amplifier is another type of op-amp circuit. In this design, shown in Figure 15-9, the positive leg is connected to a fixed supply voltage—perhaps a supply midpoint produced with a resistor-based voltage divider. The negative pin, in contrast, combines the input signal with a looped-back output voltage at a ratio dictated by R_{in} and R_f.

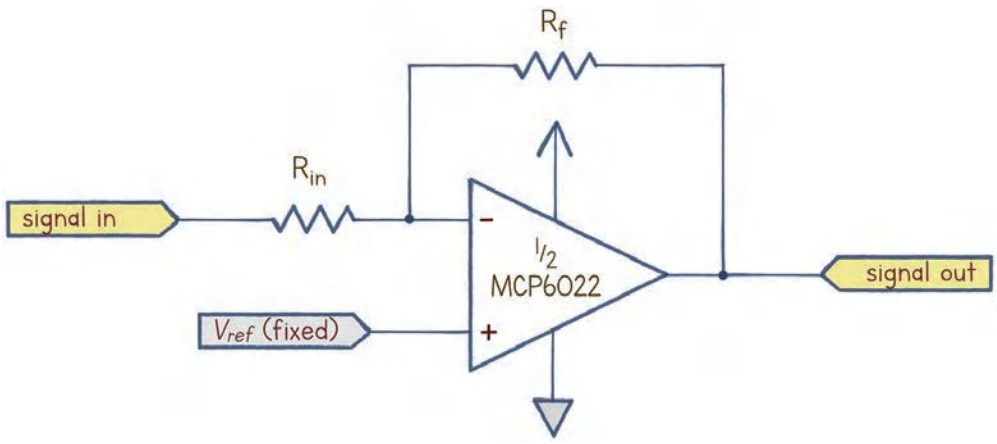


Figure 15-9: A single-supply inverting amplifier

To understand this architecture, let's simplify by assuming that R_f and R_{in} have the same value. In this case, as discussed in Chapter 12, the voltage at the midpoint of the R_f-R_{in} path is simply the average of V_{signal} and V_{out}.

As in all earlier cases, the chip's equilibrium condition is $V_{in+} \approx V_{in-}$. If the circuit is at equilibrium and V_{signal} suddenly increases by 100 mV, the voltage on the inverting leg jumps by 50 mV. This change puts the amplifier firmly in $V_{in+} \ll V_{in-}$ territory, prompting the output voltage to plunge. The process continues until equilibrium is restored—which requires a 100 mV reduction in V_{out} to return the V_{in-} average to its prior level. When $R_f = R_{in}$, we have an inverting voltage follower that mirrors the input signal relative to the reference voltage provided on the noninverting pin.

We can attempt a more rigorous analysis of the behavior by dusting off the generalized two-resistor voltage divider formula from Chapter 12. This gives us the solution for V_{in-}:

$$V_{in-} = \frac{R_f}{R_f + R_{in}} \cdot V_{signal} + \frac{R_{in}}{R_f + R_{in}} \cdot V_{out}$$

To make the math less tedious, let's calculate the output voltage relative to V_{ref} and choose V_{ref} equal to the midpoint of the supply voltage. In this view, $V_{in+} = V_{mid} = 0$ V, so the op-amp formula reduces to:

$$V_{out\,(rel)} = 0\,\text{V} + \left(0\,\text{V} - V_{in-\,(rel)}\right) \cdot A_{OL} = -V_{in-\,(rel)} \cdot A_{OL}$$

Combining the two equations, we obtain:

$$\begin{aligned} V_{out\,(rel)} &= -\left(\frac{R_f}{R_f + R_{in}} \cdot V_{signal\,(rel)} + \frac{R_{in}}{R_f + R_{in}} \cdot V_{out\,(rel)}\right) \cdot A_{OL} \\ &= -\frac{R_f \cdot A_{OL}}{R_f + R_{in}} \cdot V_{signal\,(rel)} - \frac{R_{in} \cdot A_{OL}}{R_f + R_{in}} \cdot V_{out\,(rel)} \end{aligned}$$

To tidy this up, we can multiply both sides by $R_f + R_{in}$:

$$(R_f + R_{in}) \cdot V_{out\,(rel)} = -R_f \cdot A_{OL} \cdot V_{signal\,(rel)} - R_{in} \cdot A_{OL} \cdot V_{out\,(rel)}$$

Next, we can consolidate all the V_{out} terms on the left side and solve for V_{out}:

$$(R_f + R_{in} + R_{in} \cdot A_{OL}) \cdot V_{out\,(rel)} = -R_f \cdot A_{OL} \cdot V_{signal\,(rel)}$$

$$V_{out\,(rel)} = -\frac{R_f \cdot A_{OL}}{R_f + R_{in} + R_{in} \cdot A_{OL}} \cdot V_{signal\,(rel)}$$

In this equation, when A_{OL} is large, the value of the fraction is determined chiefly by the ratio of two big numbers: $R_f \cdot A_{OL}$ in the numerator to $R_{in} \cdot A_{OL}$ in the denominator. In other words, we can neglect the $R_f + R_{in}$ part and write the circuit's gain as:

$$A_{signal} \approx -\frac{R_f}{R_{in}}$$

The main perk of the inverting amplifier is that, as the name implies, it inverts the waveform relative to a chosen point of symmetry (the voltage on the noninverting leg). This is useful as a building block for some clever op-amp signal math we'll discuss down the line. The trade-off is reduced input impedance due to current flowing through the R_f-R_{in} path. The dynamics of loop gain are essentially the same for inverting and noninverting variants: It's reduced proportionally to the amplification ratio configured with R_f and R_{in}.

Bandwidth Control

Up to this point, we've discussed why fast amplifiers can perform better: They handle high-frequency signals with greater fidelity. At the same time, we don't always need chips to run at full throttle. To reduce noise, improve stability, or lower power consumption, it might be preferable to run them at a reduced speed.

In both noninverting and inverting amplifier architectures, a simple trick achieves this: Place a small capacitor in parallel with the "loopback" (feedback) resistor (R_f in Figures 15-7 and 15-9). At low signal frequencies, the capacitor's impedance is much higher than R_f, so it does nothing of note. As frequency increases, however, its impedance drops. In Chapter 5, we calculated the sine-wave frequency at which the two impedances are equal:

$$f = \frac{1}{2\pi \cdot R_f \cdot C}$$

Past that point, the capacitor dominates, providing a low-impedance path that increases the influence of V_{out} on V_{in-}. This gradually changes the circuit from an amplifier back into a voltage follower. The result is precisely controlled, frequency-dependent gain reduction. The formula can be rearranged to find the maximum feedback capacitor that still passes a chosen frequency f_{pass} without substantial attenuation:

$$C_f \leq \frac{1}{2\pi \cdot R_f \cdot f_{pass}}$$

This inline capacitance in the feedback loop works as expected. Adding a traditional low-pass R-C filter, however, may not! Such a filter attenuates fast-changing signals but, as discussed in Chapter 12, also creates a phase shift approaching −90 degrees as the frequency increases. Together with the internal phase shifts in the amplifier, this might be enough to create a −180-degree shift that turns negative feedback into positive feedback, causing oscillation if there's enough remaining loop gain.

When working with op-amp feedback loops, any shunt capacitances—especially on the V_{in-} pin—should be minimized.

Current-to-Voltage Amplifier

So far, we've discussed op-amp circuits that translate voltage-based input signals into (typically larger) output voltages. Another important amplifier architecture is a circuit that converts current to voltage. This design, known as the *transimpedance amplifier* (Figure 15-10), is not nearly as common as its voltage-to-voltage counterparts, but it's useful when working with current-based sensors such as photodiodes and electret microphones.

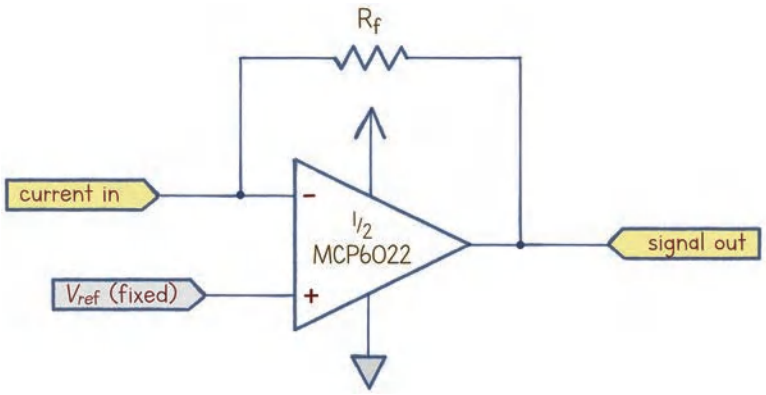


Figure 15-10: A basic transimpedance amplifier

The design is startlingly simple. If a load connected to the V_{in-} leg tries to sink some current by connecting this terminal to ground, the voltage on the inverting input will slump—the amplifier's inputs are high impedance—and this will send V_{out} higher, allowing increased current to flow through R_f. To restore the equilibrium of $V_{in+} \approx V_{in-}$, the current flowing through the feedback path will need to match the demand nearly perfectly on the input leg. From Ohm's law, the necessary output swing will be proportional to the sourced current times the value of R_f. The amplifier's equation is simply:

$$V_{out} = V_{ref} + I \cdot R_f$$

For example, a 10 kΩ resistor will produce a swing of 1 volt per 100 µA. The output voltage will move up from the reference point if current is flowing out, or down if it's flowing in.

Frequency Response

In contrast to voltage-based op-amp circuits, the transimpedance amplifier's gain is theoretically free: If the connected load is an ideal current source (or sink), nothing attenuates the voltage nudges in the feedback loop once currents synchronize. This means that to model the circuit's response, we don't need to divide the chip's A_{OL} by the configured signal gain.

Alas, that's just theory. As discussed earlier, the inverting input of an op-amp will always exhibit some internal parasitic capacitance to ground, with more potentially contributed by the connected signal source; we label the sum of these capacitances C_{in}. The value is typically just a couple of picofarads, but it forms a low-pass R-C filter

with R_f and eventually begins to attenuate signals in the feedback loop. This eats into A_{OL} much like voltage-to-voltage gain did (Figure 15-11).

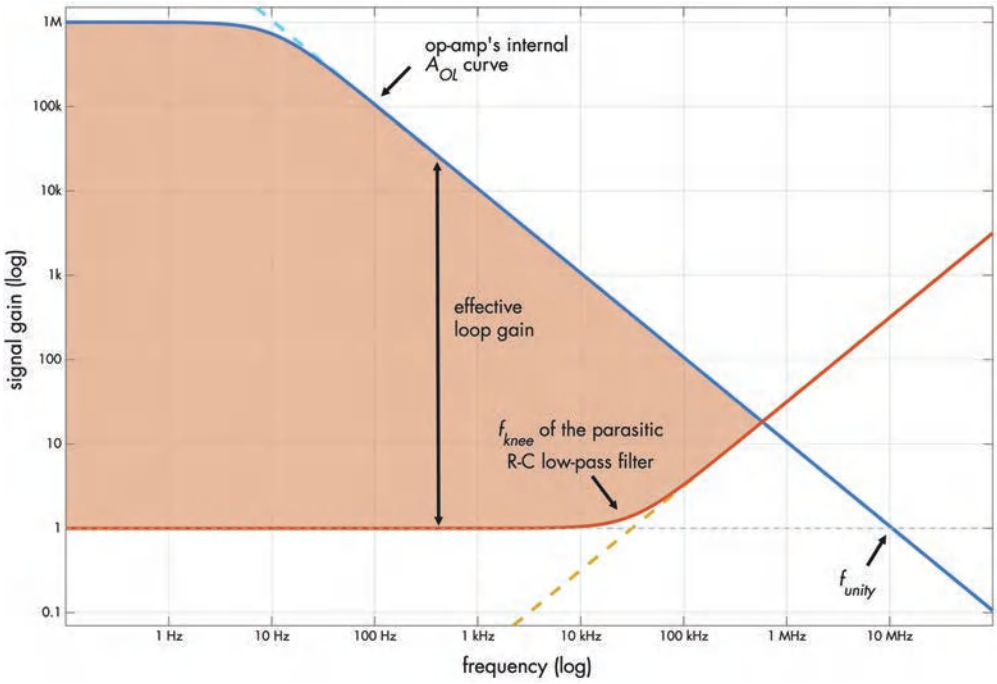


Figure 15-11: The loss of loop gain in a transimpedance amplifier

The point where the curves meet is where loop gain drops to 1. This limit is easy to find. Start by calculating the R-C corner frequency for the feedback resistor and the parasitic capacitance on the input leg. The formula should be familiar by now:

$$f_{knee} = \frac{1}{2\pi \cdot R_f \cdot C_{in}}$$

This allows us to write the equation for a function that represents the dashed, upward 45-degree diagonal in Figure 15-11. In a logarithmic plot, such diagonals are produced by expressions of the form $y = x \cdot const$, with the constant moving the line up and down. Looking again at the figure, we need to push the line down so that when the horizontal coordinate (frequency) is $x = f_{knee}$, the vertical coordinate (gain) is $y = 1$. The formula that fits is $y = x / f_{knee}$.

Next, we want to build a function that describes the downward 45-degree diagonal of the op-amp's inherent A_{OL} curve. These diagonals are given by $y = 1/x \cdot const$. To match the plot, we need the line to cross $y = 1$ when $x = f_{unity}$; once again, there's just one obvious solution: $y = f_{unity} / x$.

All that remains is to find the frequency (the x-coordinate) at which the functions are equal to each other—that's the crossover point. The task boils down to finding x such that $x / f_{knee} = f_{unity} / x$:

$$\frac{x^2}{f_{knee}} = f_{unity} \Rightarrow x^2 = f_{unity} \cdot f_{knee} \Rightarrow x = \sqrt{f_{unity} \cdot f_{knee}}$$

Given this result, we can plug in the earlier formula for f_{knee}. The final formula is:

$$f_{crossover} = \sqrt{\frac{f_{unity}}{2\pi \cdot R_f \cdot C_{in}}}$$

This formula gives us the practical frequency limit for the amplifier. For most op-amps, f_{unity} is functionally the same as the gain-bandwidth product (GBWP), so the values can be used interchangeably.

In line with the earlier discussion of voltage-to-voltage amplifiers, an additional bandwidth-limiting capacitor can be placed in parallel with R_f. The formula is the same as before. For a chosen maximum pass-through frequency f_{pass}, the feedback capacitor must be chosen to obey the following inequality:

$$C_f \leq \frac{1}{2\pi \cdot R_f \cdot f_{pass}}$$

With the theory sorted out, let's put this knowledge to practical use.

Designing a Microphone Amplifier

Earlier, I noted that transimpedance amplifiers are useful for sensors such as microphones and photodiodes. For photodiodes, this is because they generate current proportional to incident illumination. An electret microphone doesn't generate current directly, but the movement of an electrically charged diaphragm modulates current flow between the source and drain terminals of the device's internal FET.

In principle, voltage-to-voltage amplifiers can be used in these situations too. The resulting current can be channeled through a series resistor; the resistor will develop a voltage across its terminals that's proportional to the current flowing through. Such circuits require more components, introduce more noise, and—at the astronomically high amplification rates required for many sensors—suffer severely limited bandwidth, often requiring complex multistage designs.

For photodiodes, the basic transimpedance circuit is simple: Place the diode in reverse bias between ground and the inverting leg of the transimpedance amplifier (this is because the induced photocurrent has a direction opposite to what flows through a forward-biased diode). The architecture of a microphone amplifier is a bit more complex; a complete implementation appears in Figure 15-12.

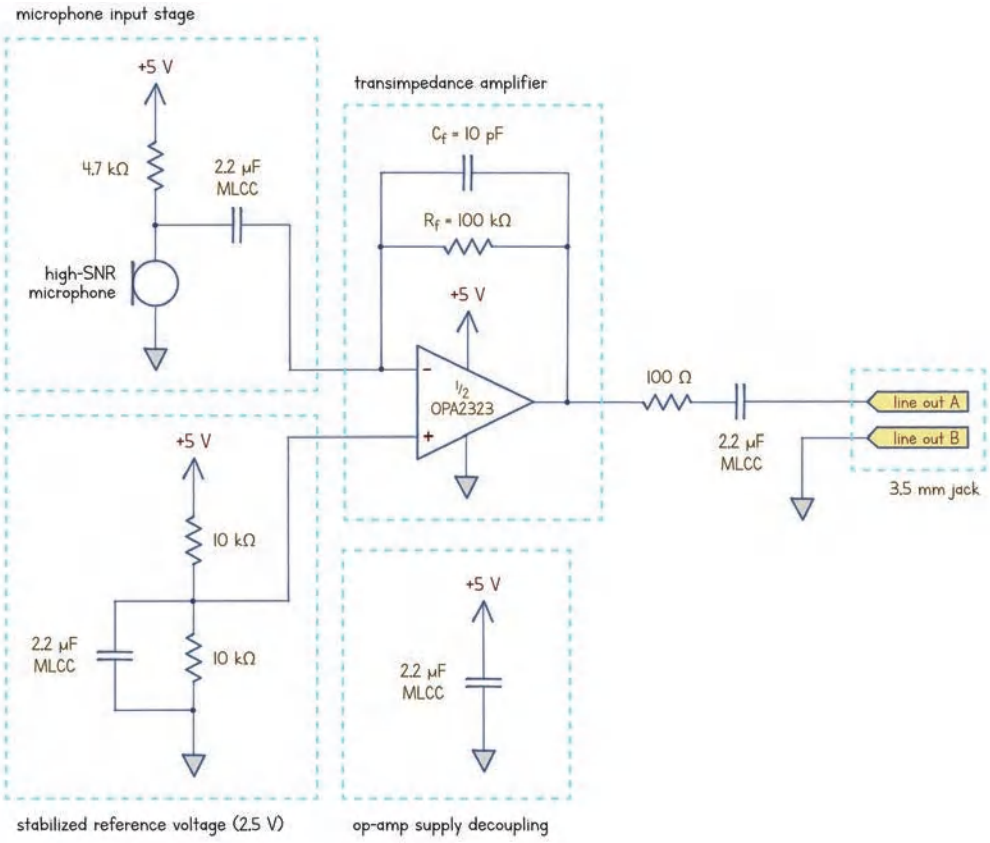


Figure 15-12: A decent microphone amplifier

Let's start with the microphone section (top left). In line with what we've discussed in Chapter 9, an electret microphone is a two-terminal device that admits variable current in response to acoustic waves. The component's terminals are connected to the drain and source of an internal field-effect transistor. As shown in Figure 10-13 in Chapter 10, FETs enter the current-modulation regime (saturation) only when a minimum voltage is applied to the drain-source path. To create this constant bias voltage, we connect the microphone to the supply rail through a resistor.

The value of the resistor can't be too high, as that would cause excessive voltage drop in response to sound-induced currents and knock the transistor out of the desired operating regime. On the flip side, if the resistance is too low, the modulated current will be difficult to measure, as most of it would flow through the resistor rather than the metered path we provide. For a microphone amplifier, the optimal choice is usually 2.2 kΩ in 3 V circuits, 4.7 kΩ for 5 V, and perhaps 10 kΩ for 9 V.

The current flowing through the microphone can be sourced via this biasing resistor, but in our circuit, the path we'll be watching is the 2.2 µF capacitor connecting the component to the transimpedance amplifier. The capacitor doesn't conduct DC, but it has a low impedance to AC signals. The lowest frequency we can hear hovers around 20 Hz; at that point, the impedance of this capacitor is roughly 3.6 kΩ. This allows the amplifier section to sense the currents sourced by the microphone section and convert them to output voltages.

The architecture of the amplifier itself (central section) should be familiar. The amplification ratio, set by R_f = 100 kΩ, might need to be adjusted for a specific microphone (I recommend a high-SNR model such as the PUI Audio AOM-5024L-HD-R).

When tinkering with the gain, please exercise caution: The circuit has no mechanism for limiting volume, so if you overdo it, it'll hurt your ears.

The amplifier section also features a small feedback capacitor, C_f = 10 pF. As discussed earlier, the capacitor attenuates the frequencies above the threshold of hearing, reducing noise and improving stability. In this instance, the capacitor sets the limit at 160 kHz, which is more than needed but leaves room for R_f adjustments. If you build the circuit, experiment by swapping the feedback capacitor for 470 pF. This should make the audio markedly more muffled due to the loss of high frequencies.

Note that the fixed midpoint voltage for the amplifier (bottom left) is further stabilized with a capacitor. This is because any voltage fluctuations of this signal are amplified too. If you switch both of these voltage divider resistors to 100 kΩ and yank out the capacitor, you will likely increase audible thermal noise and radio frequency interference.

The output from the amplifier passes through a small current-limiting resistor and a capacitor. The resistor limits the maximum current draw of the headphones to avoid overloading the amplifier. The capacitor once again blocks DC voltages while passing AC through, avoiding waste current through speaker coils when there's no waveform to reproduce.

Oscillators

In Chapter 14, we introduced a three-transistor relaxation oscillator. At the time, I noted that designing oscillators with discrete transistors becomes frustrating due to careful biasing and current management requirements. Now that we have a better abstraction in place, we can revisit the topic and have a bit more fun.

We won't be reviewing every possible flavor of these circuits—dozens exist—but will discuss the two fundamental modes of operation common to nearly all analog designs.

Relaxation

Relaxation oscillators rely on negative feedback in a circuit lacking stable equilibrium. This forces the system to constantly bounce between two nonequilibrium states. The metaphor we used earlier—a robot programmed to turn lights on when it's dark but off when detecting light—illustrates this well.

To implement this in analog electronics, we needed a switch without a half-on state. This proved challenging; we accomplished it with a Schmitt trigger circuit, which abruptly shifts the V_{GS} voltage of two MOSFETs whenever one begins to turn on.

Figure 15-13 shows a similar circuit constructed using an op-amp. Compared to our earlier design, this approach has fewer moving parts. It's a good project to construct on a breadboard and examine with an oscilloscope.

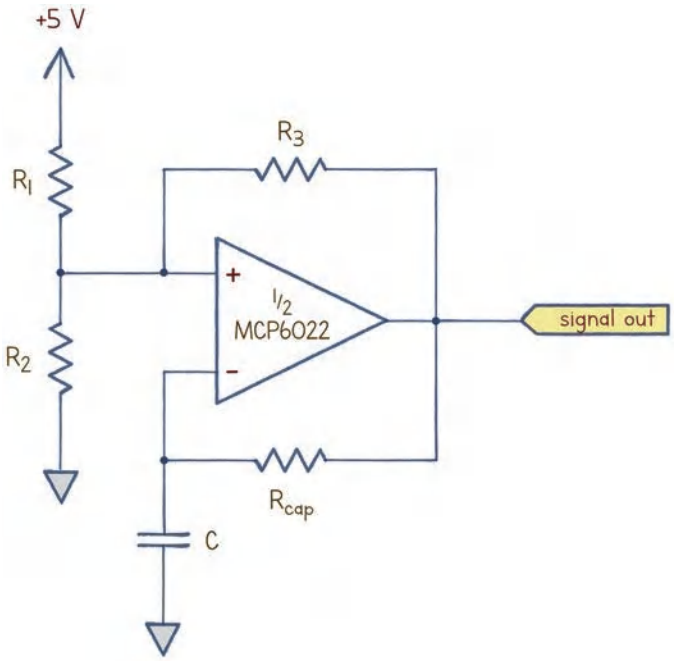


Figure 15-13: An op-amp relaxation oscillator

Let's choose the same values for R_1, R_2, and R_3 and examine the noninverting input of the chip. What we have there is a three-way voltage divider: The signal on the noninverting input is the simple average of three voltages—V_{supply} (5 V), ground (0 V), and V_{out}. We don't know the value of V_{out} just yet, but it can only vary from 0 V to V_{supply}. Thus, we can tell that the V_{in+} signal will always stay between $\frac{1}{3} \cdot V_{supply}$ and $\frac{2}{3} \cdot V_{supply}$.

Next, let's examine the inverting input. When the circuit is first powered on, the capacitor C isn't charged, so V_{in-} sits at 0 V. Since the voltage on the noninverting input can't be lower than $\frac{1}{3} \cdot V_{supply}$, this creates a $V_{in+} \gg V_{in-}$ scenario, sending the output voltage toward the positive rail. When V_{out} shoots up, it also bumps the V_{in+} average to $\frac{2}{3} \cdot V_{supply}$.

Because V_{out} is now high, this starts the process of charging the capacitor through the bottom resistor (R_{cap}). After a while, the capacitor voltage is bound to exceed $\frac{2}{3} \cdot V_{supply}$. The capacitor voltage is also hooked up to the amplifier's inverting input, and at that point, V_{in-} begins to exceed V_{in+}, nudging the output voltage lower. This output voltage drop is reflected in the three-way average present on the V_{in+} leg, pulling it further down and causing the difference between V_{in-} and V_{in+} to widen. This puts the amplifier firmly into $V_{in+} \ll V_{in-}$ territory.

At that point, V_{out} must drop to 0 V, lowering the voltage on the noninverting leg to $\frac{1}{3} \cdot V_{supply}$. With V_{out} low, the capacitor starts discharging through R_{cap}, but it needs to travel from the current charge state of $\frac{2}{3} \cdot V_{supply}$ all the way to $\frac{1}{3} \cdot V_{supply}$ before V_{in-} becomes lower than V_{in+} and the cycle restarts. The repeated charging and discharging of the capacitor between $\frac{1}{3} \cdot V_{supply}$ and $\frac{2}{3} \cdot V_{supply}$ voltages results in periodic oscillation. The circuit produces a square wave signal with a period dictated by the value of C and R_{cap}.

The oscillation frequency can be approximated as with the discrete-transistor version in Chapter 14. In a 5 V circuit with $R_1 = R_2 = R_3$, the capacitor charges and discharges by $\Delta v = 1.67$ V. If $R_{cap} = 10$ kΩ, the quasi-constant capacitor charging current is $I = 2.5$ V / 10 kΩ = 250 µA.

Knowing Δv and I, and assuming C = 1 µF, we can tap into the capacitor equation ($\Delta v = I \cdot t/C$) to solve for t. The result is 6.7 ms. This puts the charge–discharge round trip at 13.4 ms, suggesting a frequency of about 75 Hz.

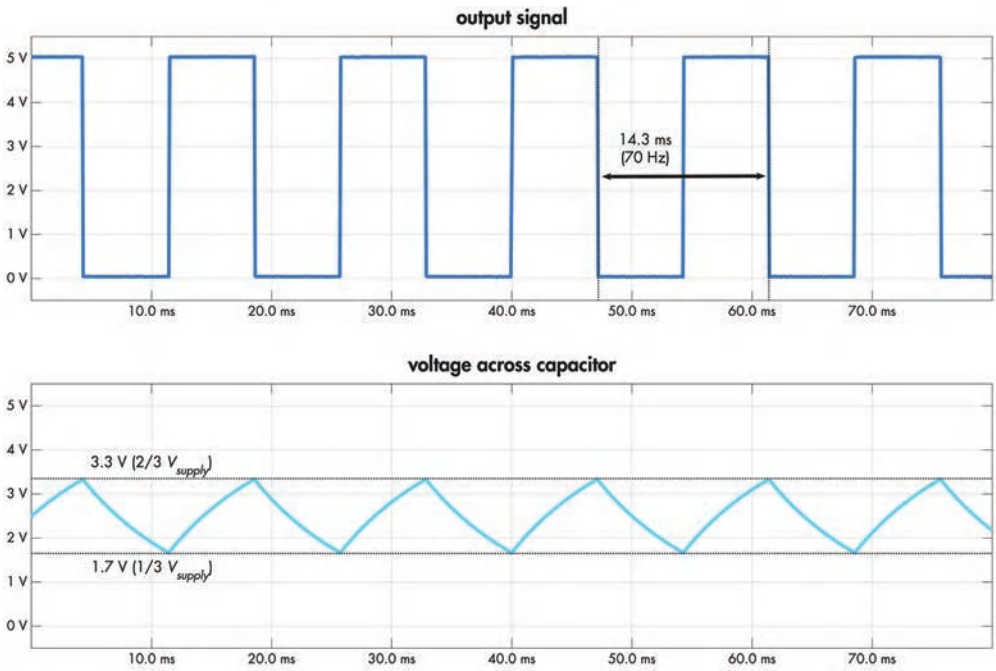


Figure 15-14: The behavior of an op-amp relaxation oscillator with $R_1 = R_2 = R_3 = R_{cap}$ = 10 kΩ and C = 1 µF

In reality, as shown in Figure 15-14, the observed frequency is a bit lower: 70 Hz instead of 75 Hz. This occurs because with Δv = 1.6 V, the constant-current approximation of the capacitor charging process is stretched thin; the segments in the bottom oscilloscope trace deviate noticeably from straight lines.

Short of reducing R_3 to lower Δv and thus reducing the variations in current, a better formula uses the capacitor-charging equation derived in Chapter 5 for constant voltage through a resistor:

$$V_{cap} = V_{in} \cdot \left(1 - e^{-\frac{t}{RC}}\right)$$

To simplify the math, we can use $\frac{1}{3} \cdot V_{supply}$ as the reference point for the calculation. In this view, the new "virtual" supply voltage is $V_{in} = \frac{2}{3} \cdot V_{supply}$ (because we shaved off the bottom third). The capacitor thus charges from the new 0 V. In the old frame of reference, the upper capacitor voltage was equal to two-thirds of the original V_{supply}; in the new view, this can be restated as 50% of V_{in}. To find the charging time, we just need to rearrange the R-C formula for the ratio of $V_{cap} \,/\, V_{in}$ and then solve for t at which the capacitor is 50 percent charged (0.5):

$$0.5 = \frac{V_{cap}}{V_{in}} = 1 - e^{-\frac{t}{RC}}$$

After moving 1 to the left and flipping signs, the equation simplifies to:

$$0.5 = e^{-\frac{t}{RC}}$$

To get rid of exponentiation, we can take the natural logarithm of both sides: $\ln(0.5) = -t/(RC)$. Then, solve for t:

$$t = \underbrace{-\ln(0.5)}_{=\ln(2)} \cdot RC$$

In this case, the charging resistor is called R_{cap}, so the equation should be restated as:

$$t = \ln(2) \cdot R_{cap} \cdot C \approx 0.693 \cdot R_{cap} \cdot C$$

The value of t can be used to find the oscillation frequency, keeping in mind that every period consists of a charging and discharging cycle:

$$f_{osc} = \frac{1}{2t} \approx \frac{1}{1.386 \cdot R_{cap} \cdot C} \approx \frac{0.721}{R_{cap} \cdot C}$$

If we plug 1 µF and 10 kΩ into the equation, the value works out to 72 Hz, which is within 3 percent of the observed behavior and comfortably within the tolerances for standard passive components.

An interesting property of this circuit is that it easily converts to a *voltage-controlled oscillator (VCO)*. We can change the threshold voltages for the charge–discharge cycle by adjusting the voltage supplied on the upper leg of R_1 or the lower leg of R_2. If we bring the thresholds closer together, the circuit will need less time to toggle between the two states, resulting in a higher operating frequency. VCOs are quite useful in electronic musical instruments, radios, and certain types of timing circuits that we'll cover in Chapter 23.

Phase-Shift

As the next challenge, consider the minimal oscillator circuit pictured in Figure 15-15. At first glance, the general layout resembles a voltage follower. A static input voltage (2.5 V) is supplied on the noninverting leg, and a feedback loop containing extra capacitors appears to pipe the output voltage back to the inverting input. Once the capacitors are charged, we could expect the output voltage to settle at 2.5 V.

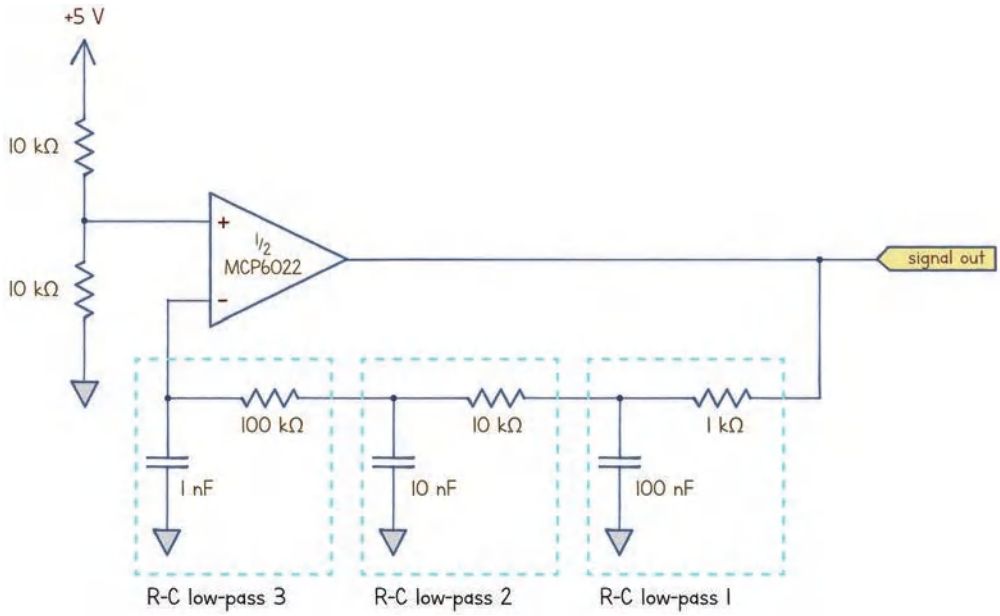


Figure 15-15: An op-amp phase-shift oscillator

Yet that's not what happens! I encourage you to give it a go and see it with your own eyes. To understand the actual behavior, once again recall the discussion about phase-shift oscillators toward the end of Chapter 14: We noted that if a negative feedback loop is constructed so that a 180-degree phase shift can occur at some sine-wave signal frequency, this effectively inverts that waveform and turns negative feedback into positive feedback. Instead of limiting gain, the feedback loop begins driving oscillations. Again, this happens only at the frequency for which the shift is 180 degrees, and only if at that frequency, the circuit still offers sufficient gain.

In this circuit, the feedback loop contains three R-C low-pass filters. As discussed in Chapter 12, each filter shifts the phase of the sine-wave signals from 0 degrees to −90 degrees, depending on frequency. Here, to achieve an overall −180-degree offset, each R-C stage needs to contribute −60 degrees.

Finding that frequency is straightforward. In Chapter 12, we came up with the following formula describing the shift (in degrees or radians) associated with a low-pass filter:

$$\theta = -\arctan(2\pi fRC)$$

Arctangent is the inverse of the tangent function. In a right triangle, the tangent function describes the ratio of opposite to adjacent sides for a given angle; arctangent reverses this, giving us an angle for a ratio. In other words, if $x = \tan(\alpha)$, then $\alpha = \arctan(x)$. This allows us to rewrite the equation as:

$$2\pi fRC = -\tan(\theta)$$

We're trying to solve for f where $\theta = -60°$; the value of $-\tan(-60°)$ is roughly 1.73, so we can plug that into the equation and then move everything except f to the right. Throwing in the component values for the first R-C stage in the schematic, we obtain:

$$f \approx \frac{1.73}{2\pi RC} \approx \frac{1.73}{2\pi \cdot 1\,\text{k}\Omega \cdot 100\,\text{nF}} \approx 2.75\,\text{kHz}$$

You'll notice that the result is the same for the other two stages. They have higher resistances but proportionally lower capacitances, so the denominator of the fraction doesn't change.

Oscilloscope traces for the circuit are shown in Figure 15-16. Because the amplifier's gain isn't constrained in any way, the output waveform is a square wave. Nevertheless, the edges of the signal are rounded by each of the three R-C stages before looping back onto the input, so the sine-wave model holds up. A comparison between an ideal sine-wave R-C model and the waveforms in this circuit is shown in Figure 15-17.

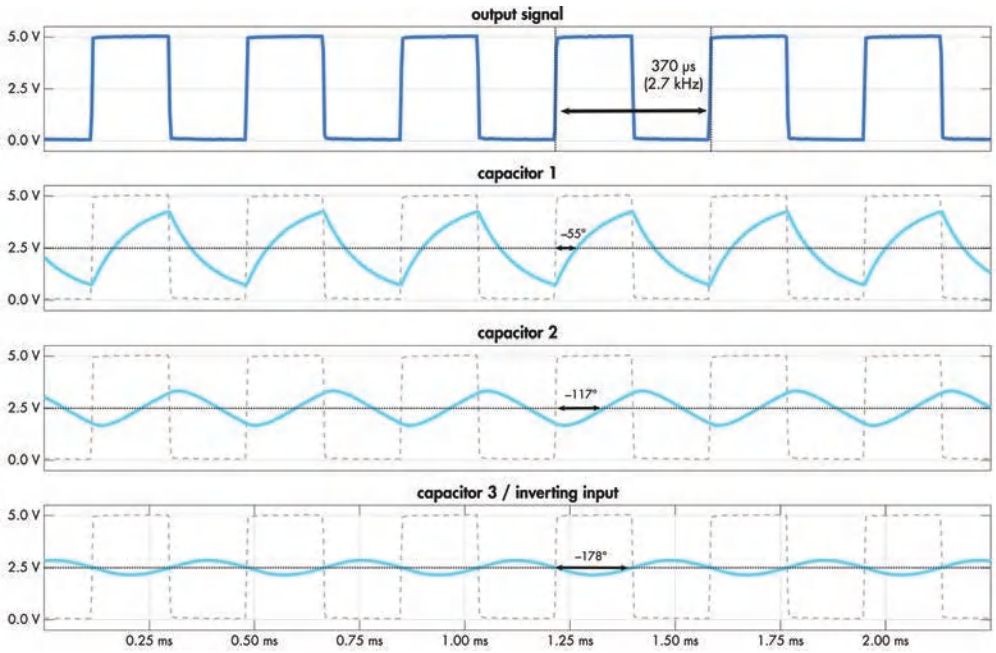


Figure 15-16: Oscilloscope traces for the phase-shift oscillator, annotated to indicate the measured phase shift

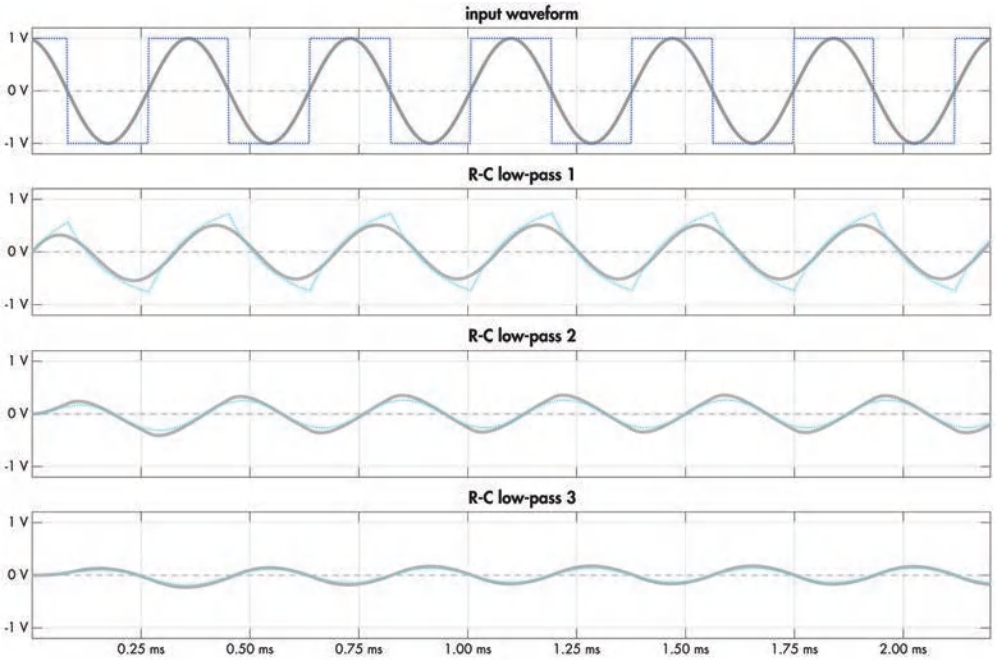


Figure 15-17: A simulation of a square and sine waves passing through three R-C filter stages

If you wish to make the output a sine wave, it's possible to tinker with the feedback loop to lower the circuit's gain, but this is finicky because the gain must still be sufficient to make up for the attenuation produced by the three R-C sections. A simpler trick is to tap into the signal on the noninverting leg (bottom oscilloscope trace) and use the other part of a dual op-amp IC to amplify this signal to your heart's desire.

Astute readers might be wondering why I designed the stages so that each of them has an impedance 10 times larger than the stage before it. This is to prevent the filters from appreciably loading one another. If all the impedances were in the same ballpark, the middle filter could source currents from the left as easily as it could from the right. In that situation, finding the point of −180-degree phase shift with decent accuracy would require calculating the transfer function for the entire six-component Franken-filter—a doable task but, to borrow a mathematical term, rather unpleasant.

Active Filters

Oscillators aside, a more conventional application of an R-C filter is to eliminate unwanted noise and zero in on signals of interest. In an audio circuit, we might use filtering and selective amplification to boost or reduce treble and bass. In a radio receiver, we can use the same approach to pluck out one transmission and disregard the rest.

When used in this capacity, a single-stage R-C filter clearly leaves something to be desired. Its response curve has a gentle 45-degree slope on a logarithmic plot. By contrast, we imagine an ideal filter as having a "brick wall" response—passing through chosen frequencies without attenuation, then completely blocking the rest.

In this section, we'll look at techniques for combining filters, whether to address the problem just mentioned or to realize more complex *bandpass* schemes that admit only a range of frequencies and require two complementary filters to each do its part. First, we'll attempt to make do without op-amps. Afterward, we'll toss some in and see if that helps.

Bending the R-C Slope

In principle, if we have a circuit that attenuates signals past a certain frequency, the simplest way to increase the attenuation rate is to perform the same operation twice—that is, connect two filters in series. Figure 15-18 shows the mathematical outcome of applying the transfer function of a single R-C filter to the same signal a couple of times.

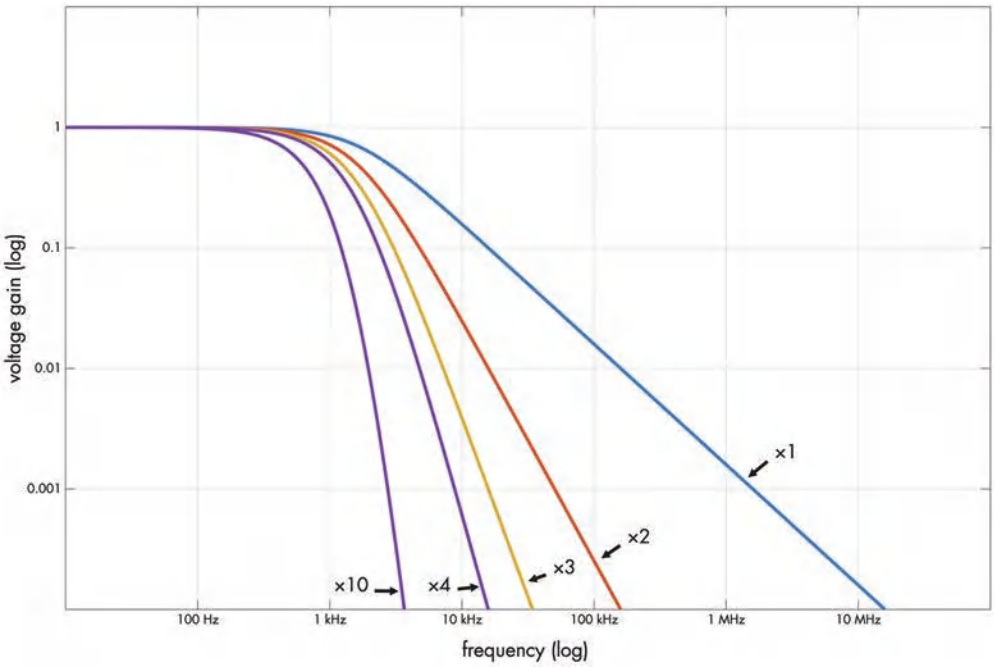


Figure 15-18: The theoretical effect of stacking first-order R-C low-pass filters

In practice, any attempt to construct a proper multistage R-C filter using passive components will run into the stage-loading issue that also came up in the context of the phase-shift oscillator. Unless each filter is made to have an input impedance much higher than the output impedance of the preceding stage, we end up with currents flowing through unexpected paths.

To articulate the problem a bit more precisely, let's consider the circuit displayed in Figure 15-19. The first stage is constructed with R_1 = 10 kΩ and C_1 = 10 nF; the resulting signal then feeds into a second filter constructed with R_2 = 100 Ω and C_2 = 1 µF. Notionally, both of these stages have the same center frequency of 1.59 kHz.

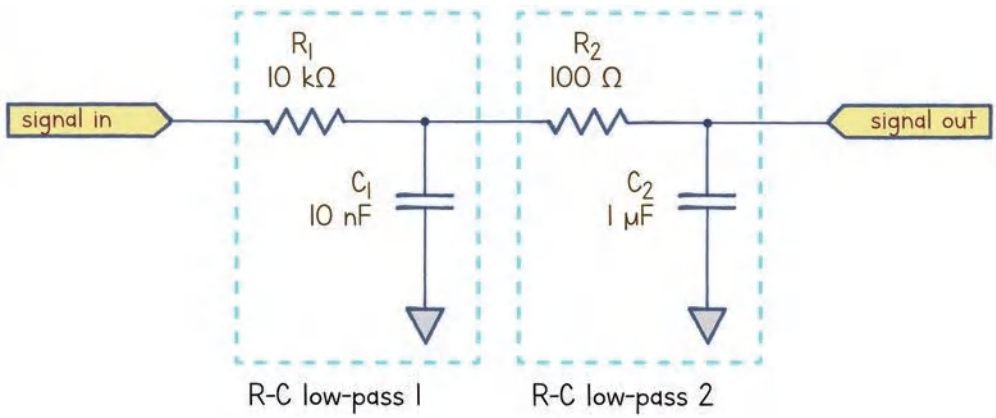


Figure 15-19: A problematic two-stage R-C filter

Of course, something about the design is fishy. The second capacitor is supposed to charge through R_2 = 100 Ω, but its path to the signal source appears to have an overall resistance of R_1 + R_2 = 10.1 kΩ. If we construct the circuit and measure its frequency

response, we're not going to see a curve resembling the standard R-C plot. Instead, we'll get the weird, flattened shape seen in Figure 15-20.

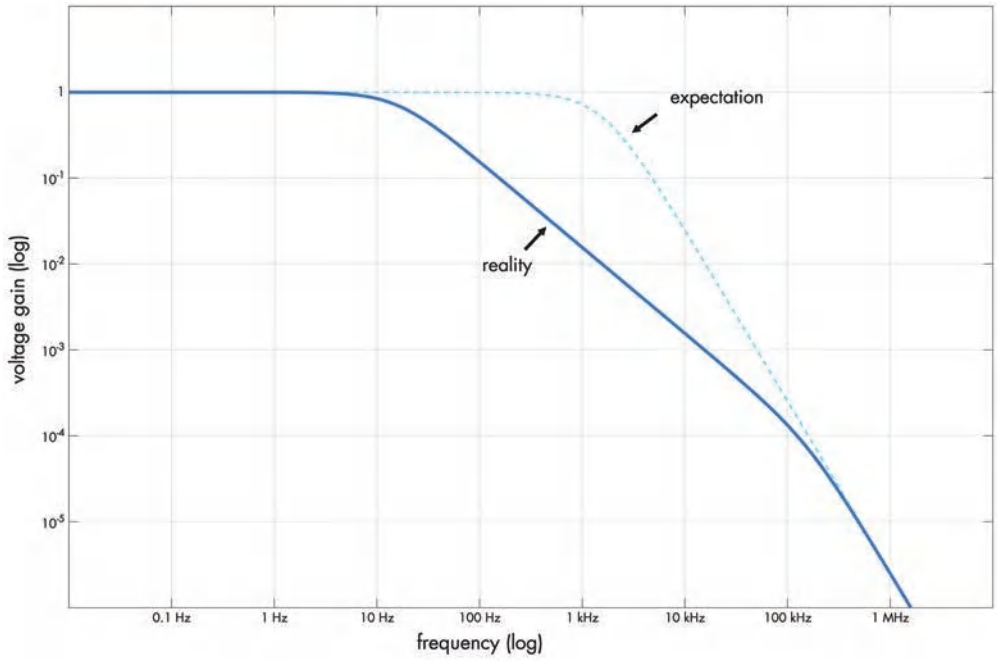


Figure 15-20: The actual behavior of the faulty R-C filter

To get a sense of what went wrong, let's return to our earlier observation about C_2. The component is supposed to charge via R_2 = 100 kΩ, but because the stages are not separated, it's actually charging via the $R_1 + R_2 \approx R_1$ path. This suggests the existence of a parasitic R_1-C_2 filter with a knee frequency of 15.9 Hz, matching the first bend in the plot.

There's a similar problem with capacitor C_1: It's meant to charge via R_1 = 10 kΩ, but at high frequencies, it has a lower-impedance path to ground via R_2 = 100 Ω. This hints at a second parasitic filter with a knee at 159 kHz—just where we see the second bend.

This explanation is a bit hand-wavy, so we can also build a more rigorous model of the circuit using the complex number approach introduced in Chapter 12. The method is exceedingly helpful in this case, but if you're still not enthused, don't fret—this is the last time you'll encounter complex numbers in this book. For readers who are severely allergic, it's also safe to skip ahead.

Anatomy of a Catastrophe

The analysis begins by marking two key junction points in the circuit, point A and point B (Figure 15-21). We'll investigate these nodes using Kirchhoff's current law. The law, once again, boils down to the observation that what comes in must equal what goes out.

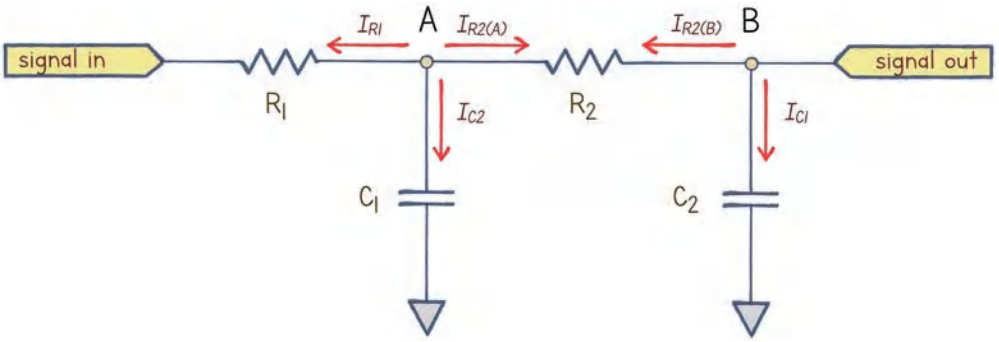


Figure 15-21: Key junction currents in the circuit

Let's start with node B. We assume there's no substantial output loading, so the node has two possible outgoing current paths: via resistor R_2 and via capacitor C_2. For the resistor, the current is proportional to the voltage between A and B, so we can write the following equation for the outgoing complex current from point B:

$$\widetilde{I}_{R2(B)} = \frac{\widetilde{V}_B - \widetilde{V}_A}{R_2} = \frac{\widetilde{V}_{out} - \widetilde{V}_A}{R_2}$$

In the circuit, point B and the output terminal are connected, so $\widetilde{V}_B$ is the same as $\widetilde{V}_{out}$. Our ultimate goal is to solve for $\widetilde{V}_{out}$, so we should cut down on unnecessary variables and make the substitution now.

The other outgoing current from point B flows through C_2; for sine-wave signals, it's proportional to the applied voltage divided by the component's impedance ($\widetilde{I} = \widetilde{V}/\widetilde{Z}$). In the complex-verse, we don't need to manually track phase shifts, so we just tap into the *s*-based capacitor impedance formula from Chapter 12 and plow ahead:

$$\widetilde{I}_{C2} = \frac{\widetilde{V}_{out}}{\widetilde{Z}_{C2}} = \frac{\widetilde{V}_{out}}{\frac{1}{sC_2}} = \widetilde{V}_{out} \cdot sC_2$$

For point B, assuming we didn't mix up the direction of the currents, the following KCL equality must hold:

$$\widetilde{I}_{R2(B)} + \widetilde{I}_{C2} = 0\,\text{A}$$

At this point, we have three equations and we're about halfway there. The next task is to repeat the same analysis for node A, netting the following haul:

$$\widetilde{I}_{R1} = \frac{\widetilde{V}_A - \widetilde{V}_{in}}{R_1}$$

$$\widetilde{I}_{R2(A)} = \frac{\widetilde{V}_A - \widetilde{V}_{out}}{R_2}$$

$$\widetilde{I}_{C1} = \widetilde{V}_A \cdot sC_1$$

The formulas are subject to the KCL equation for point A:

$$\widetilde{I}_{R1} + \widetilde{I}_{R2(A)} + \widetilde{I}_{C1} = 0\,\text{A}$$

We end up with a system of seven equations that can be combined into one fairly grungy formula and then solved for the ratio of $\widetilde{V}_{out}$ to $\widetilde{V}_{in}$. After a bit of equation-wrangling, this gives us a frequency-dependent transfer function $H(f)$:

$$H(f) = \frac{1}{1 + s(R_1 \cdot C_1 + R_2 \cdot C_2 + R_1 \cdot C_2) + s^2 R_1 \cdot R_2 \cdot C_1 \cdot C_2}$$

If we replace s with $2\pi j \cdot f$ and tidy up a bit, we get:

$$H(f) = \frac{1}{\underbrace{1 - 4\pi^2 f^2 \cdot R_1 \cdot R_2 \cdot C_1 \cdot C_2}_{\text{real term}} + j \cdot \underbrace{2\pi \cdot f \cdot (R_1 \cdot C_1 + R_1 \cdot C_2 + R_2 \cdot C_2)}_{\text{imaginary term}}}$$

The function can be plugged into any scientific calculator or graphing software that supports complex numbers to calculate signal amplitude and phase shift for any chosen f. With a bit of basic geometry, it's also possible to solve for the center frequency of the circuit's messy response curve; in brief, it works out to:

$$f_{knee} = \frac{1}{2\pi\sqrt{R_1 \cdot R_2 \cdot C_1 \cdot C_2}}$$

Knowing the center frequency, we can also solve $H(f_{knee})$ and find the corresponding gain:

$$Q = \frac{\sqrt{R_1 \cdot R_2 \cdot C_1 \cdot C_2}}{R_1 \cdot C_1 + R_1 \cdot C_2 + R_2 \cdot C_2}$$

Of course, this kind of math is not something we want to do every day; it exists mainly to torture students and intimidate hobbyists. If your circuit design forces you to go down that path, it's usually a good sign to pause and rethink the approach.

Multistage R-C Filter

A better way to design the circuit is to separate the stages so that stage loading is not a concern. We approximated this for the phase-shift oscillator by making the input impedance of each stage much higher than the output impedance of the stage that preceded it. For coarse isolation, it sufficed to bump every next resistor value by a factor of 10. To get closer to optimum, we'd need to aim for a multiplier of 50 or so.

The need to jack up impedances puts an obvious limit on the scheme. The first resistor can't be too low, as that would risk overloading the source; the last one can't be too high, as that would produce a faint signal that's easily overwhelmed by noise. A two-stage design is typically easy; a four-filter cascade would involve some substantial design trade-offs.

A more flexible approach to the problem is to use operational amplifiers. A simple voltage follower with no additional components can act as a buffer between each stage, as shown in Figure 15-22.

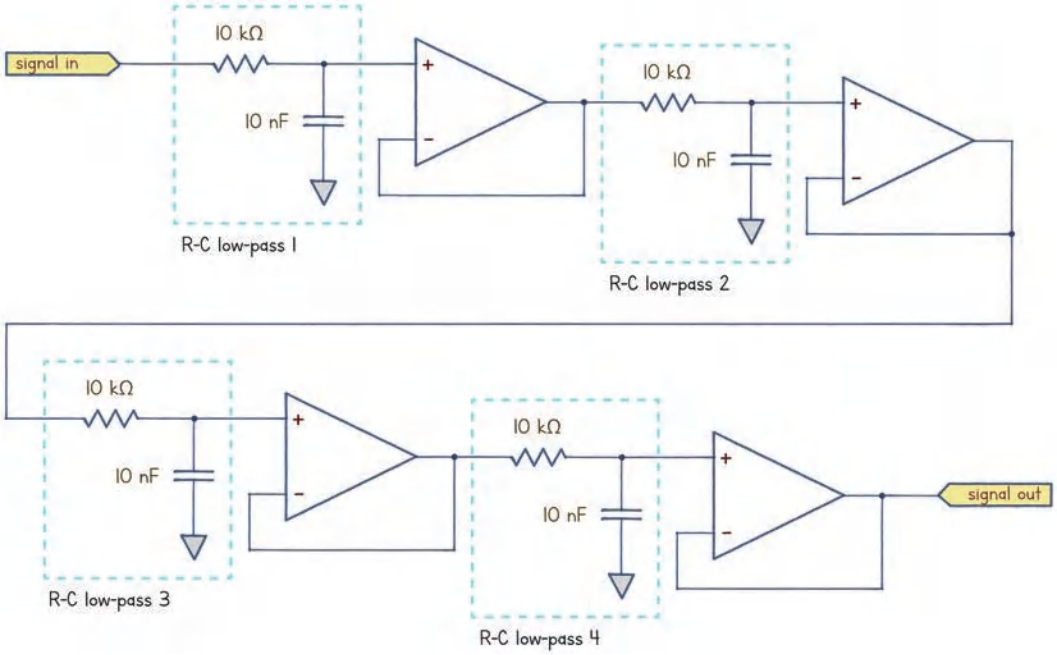


Figure 15-22: A four-stage R-C low-pass filter with op-amp voltage followers

In many cases, the simplest architecture is also the best. A basic op-amp costs less than 10 cents—far less than the time and effort required to design and troubleshoot filter architectures that demand dusting off textbooks or solving serious math.

Sallen–Key Topology

Although basic R-C filters are good enough for most uses, they suffer from weak knees. We can cascade any number of simple R-C stages to increase the slope of the attenuation curve, but each time we do that, we also cut deeper into the "pass" region on the other side of f_{knee} (Figure 15-23).

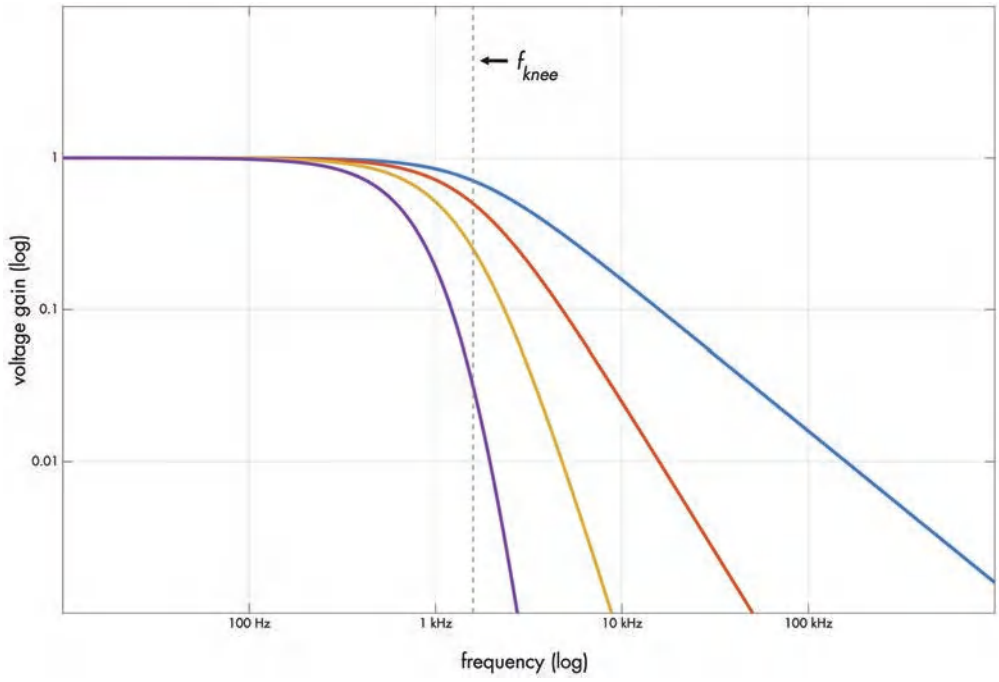


Figure 15-23: Detail of the frequency response curves for first-, second-, fourth-, and tenth-order R-C filters

We had a glimpse of a potential cure toward the end of Chapter 12. While examining the frequency response of a resonant R-L-C filter, we noted that in contrast to the basic R-C topology, the circuit could selectively boost its gain near f_{knee} by choosing the correct value of R. This lever, known as the Q factor, allows us to give our filters' response curves more attractive, sharper knees. A preview of what's possible is shown in Figure 15-24.

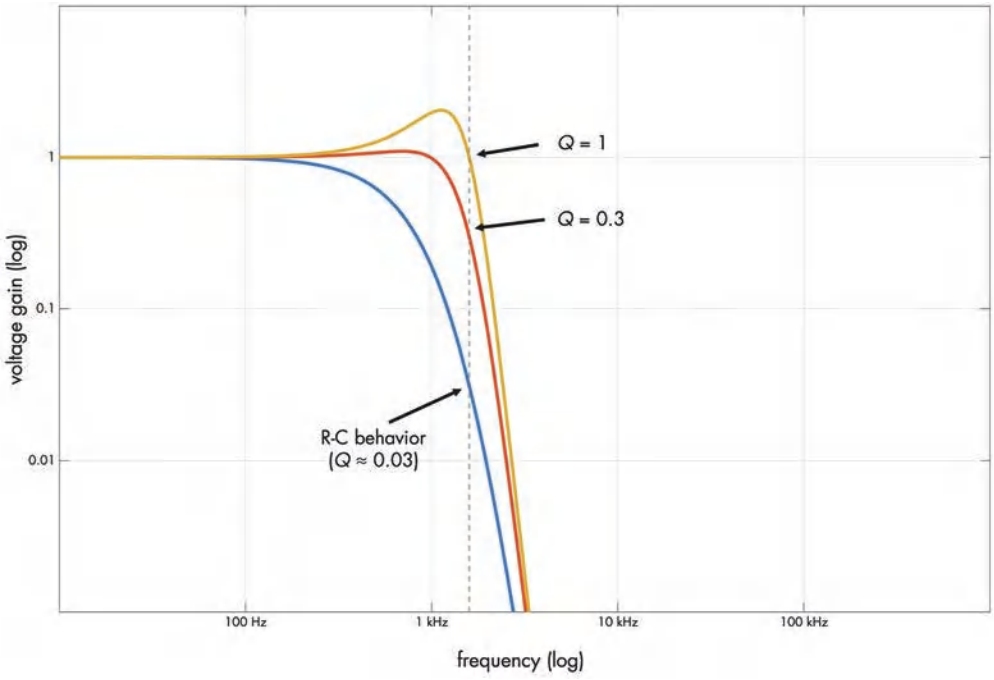


Figure 15-24: Response curve for a 10th-order R-C cascade, next to the equivalent curves for R-L-C cascades designed for an overall Q of 0.3 and 1

The main disadvantage of the R-L-C architecture is the inductor itself—these components are usually bulkier, more expensive, and less efficient than capacitors. This brings us to an interesting revelation: There is a way to build a similar circuit with an op-amp and no inductors at all.

The approach is known as the Sallen–Key topology. The basic architecture can be used in several ways, but we'll stick to low-pass filtering. The low-pass circuit can be thought of as a modification of a bog-standard two-stage R-C filter that's coupled to a voltage follower (Figure 15-25).

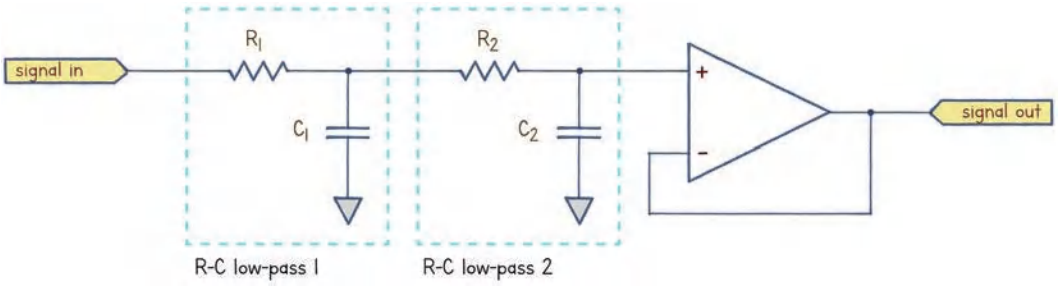


Figure 15-25: An uninteresting R-C low-pass architecture

Of course, that's not the circuit we're after. The key modification is to disconnect the bottom leg of C_1 from ground and connect it to the op-amp's output pin. This gives us the proper Sallen–Key low-pass filter, as shown in Figure 15-26.

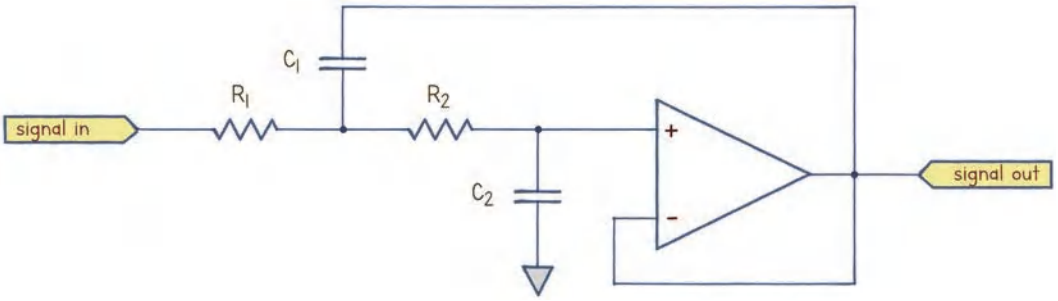


Figure 15-26: The Sallen–Key low-pass filter

The circuit isn't worth analyzing with full mathematical rigor. The analysis involves a lot of grueling equations, and it's not used often enough to justify the time. Still, we can lean on intuition to get a rough idea of what it does.

First, we can note that at low sine-wave frequencies, the impedance of both capacitors is very high. This means that up to a certain point, only the R_1-R_2 signal path matters, and the circuit behaves like a run-of-the-mill voltage follower.

As the frequency increases, the impedance of C_2 drops, eventually falling below the impedance of R_2. This means that past some other threshold, the circuit will exhibit a normal low-pass attenuation slope.

Finally, if component values are chosen correctly, the impedance of C_1 can become comparable to R_1 and R_2 before the low-pass effect of C_2 begins to dominate. In this scenario, we end up with a positive feedback loop: At certain frequencies, a fraction of the output voltage is fed back onto the noninverting input of the amplifier. This produces a frequency-sensitive signal boost.

Again, we'll skip the painstaking calculations, but there are two useful results that are worth jotting down. First, the filter's center frequency is:

$$f_{knee} = \frac{1}{2\pi\sqrt{R_1 \cdot R_2 \cdot C_1 \cdot C_2}}$$

Second, the Q factor is given by:

$$Q = \frac{\sqrt{R_1 \cdot R_2 \cdot C_1 \cdot C_2}}{C_2 \cdot (R_1 + R_2)}$$

In practice, we don't need this many independent variables in filter design. To simplify the formulas, we can choose the same value for both resistors (R) and then pick a midpoint capacitance C. We then define C_1 as $C \cdot n$ and C_2 as C / n. With these tweaks, the first equation reduces to:

$$f_{knee} = \frac{1}{2\pi \cdot R \cdot C}$$

The equation for the Q factor gets even simpler; only the ratio between C_1 and C_2 plays a role:

$$Q = \frac{n}{2}$$

In other words, the usual Sallen–Key design process is to pick R and C for a specific frequency response and input impedance, choose n for the desired boost, and then pick the capacitors such that $C_1 = C \cdot n$ and $C_2 = C / n$.

The caveat is that the frequency boost is created by the circuit feeding on its own output. If the input waveform isn't a well-behaved sine, this can have unexpected effects. In a filter with a very high Q, a sharp input pulse will cause resonant ripples that

take a while to decay (Figure 15-27). Nonresonant filters can handle such signals in a more graceful way.

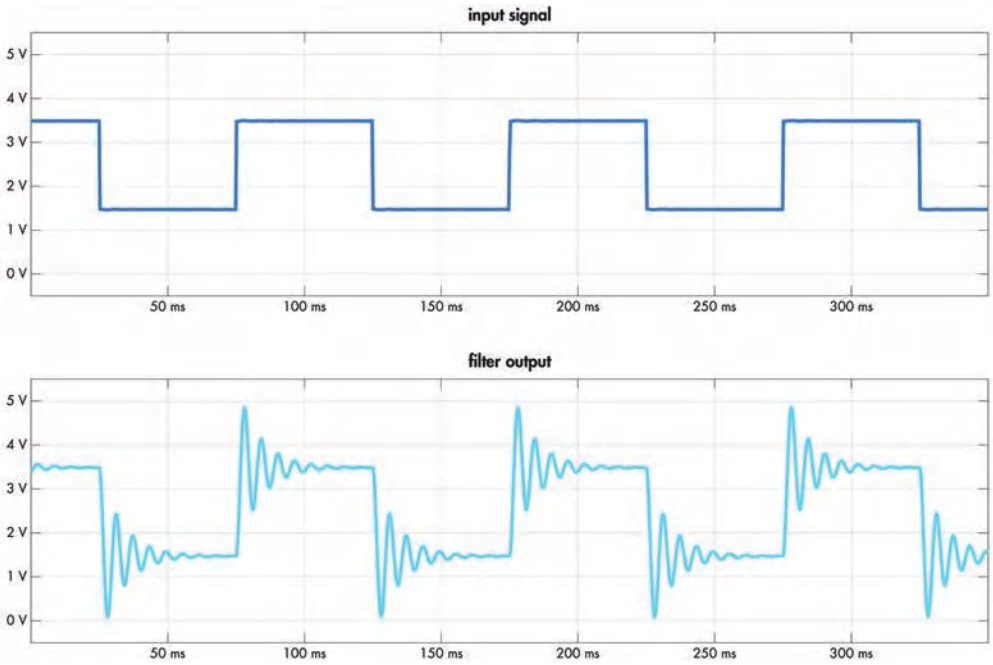


Figure 15-27: Resonance in a Sallen–Key filter designed with Q = 5 and a corner frequency of 159 Hz (R = 10 kΩ, C_1 = 1 µF, C_2 = 10 nF)

True Mathematics

This chapter might have given some readers the impression that the op-amp is a three-trick pony: It can amplify signals, generate waveforms, and turn up the bass when you're playing your favorite song. It's true that these scenarios account for most of what these chips are asked to do. At the same time, there's more to them than meets the eye.

My favorite example of how creative you can get with these devices is the realm of analog computation. Analog computers are wildly impractical, but it's still mind-blowing that an amplifier can solve equations if you ever need it to.

Summing Amplifier

The simplest example of analog computation is the adder circuit (Figure 15-28). At a glance, it's a noninverting amplifier with a number of parallel input paths.

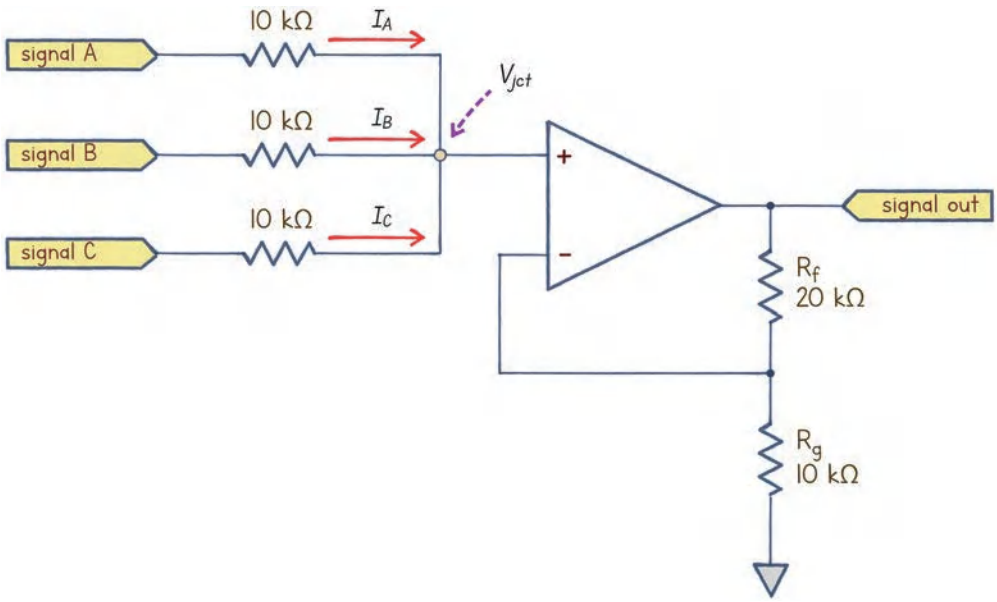


Figure 15-28: An op-amp voltage adder

If all the input resistors are identical and the signal sources can source and sink currents, we can reason that the voltage at the V_{in+} leg is the average of the three input signals:

$$V_{in+} = \frac{V_A + V_B + V_C}{3}$$

This intuition can also be confirmed using Kirchhoff's current law. If we use V_{jct} to denote the voltage at the junction, then from Ohm's law we can write the following current equations for the currents flowing through each resistor branch:

$$I_A = \frac{V_A - V_{jct}}{10\,\text{k}\Omega}$$

$$I_B = \frac{V_B - V_{jct}}{10\,\text{k}\Omega}$$

$$I_C = \frac{V_C - V_{jct}}{10\,\text{k}\Omega}$$

Further, from KCL, we can assert that the currents must balance out: $I_A + I_B + I_C = 0$ A. Combining all these equations and multiplying both sides by 10 kΩ, we get $V_A + V_B + V_C - 3 \cdot V_{jct} = 0$ V. Solving for V_{jct}, we arrive at:

$$V_{jct} = \frac{V_A + V_B + V_C}{3}$$

In other words, the first portion of the circuit averages the three input voltages and presents the result on the noninverting input of the op-amp.

The gain of the amplifier section is set by the R_f and R_g resistors; earlier, we derived the following formula for gain:

$$A \approx 1 + \frac{R_f}{R_g}$$

In this instance, because R_f is 20 kΩ and R_g is 10 kΩ, the gain works out to 3. This allows us to write the following equation for the output voltage:

$$V_{out} \approx V_{in+} \cdot A \approx \frac{V_A + V_B + V_C}{\cancel{3}} \cdot \cancel{3} \approx V_A + V_B + V_C$$

In other words, we just summed the three input voltages.

Difference Amplifier

Next, we can try our hand at subtraction. This can be implemented using the circuit topology shown in Figure 15-29.

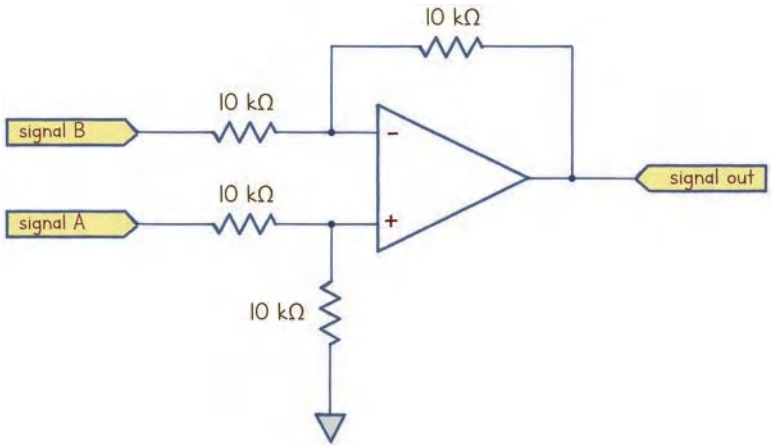


Figure 15-29: A difference amplifier

We can start the analysis with the noninverting input of the amplifier. The signal on this leg is generated by a voltage divider consisting of two identical resistances connected in series between V_A and ground. In other words, the voltage here is $V_{in+} = \frac{1}{2} \cdot V_A$.

The inverting input is a voltage divider too, except it produces a voltage that's halfway between V_B and V_{out}. We can express it as $V_{in-} = \frac{1}{2} \cdot (V_B + V_{out})$. As with any op-amp topology, linear operation can happen only when $V_{in-} \approx V_{in+}$. In other words, we can assert that for the circuit to function, the following must be true:

$$\frac{1}{2} \cdot V_A \approx \frac{1}{2} \cdot (V_B + V_{out})$$

We can cancel out the repeated ½ term on both sides and then reorder the equation to arrive at $V_{out} \approx V_A - V_B$. This is what we want in a circuit that subtracts voltages.

To be fair, the design isn't perfect. In a single-supply circuit, an op-amp can't output negative voltages, so the topology we've just analyzed works only if $V_A > V_B$; otherwise, V_{out} will hit the lower rail and stay there until the input voltages change.

To accommodate use cases where $V_A < V_B$ and the result is negative, we'd need to use a higher output voltage as the virtual "zero" point. Let's denote this as V_{zero}. If we choose V_{zero} = 2.5 V, then a computed difference of −1 V could be represented by $V_{out} = V_{zero} - 1$ V, or 1.5 V in the original frame of reference. In the same scheme, a difference of +2 V ought to correspond to a ground-referenced output voltage of 4.5 V.

To do this, we just need to disconnect the bottom voltage divider from ground

and replace 0 V with the V_{zero} of our choice. This changes the equation for the positive leg to $V_{in+} = \frac{1}{2} \cdot (V_A + V_{zero})$. The overall equilibrium condition becomes:

$$\frac{1}{2} \cdot (V_A + V_{zero}) \approx \frac{1}{2} \cdot (V_B + V_{out})$$

After tidying up and solving for the output signal, we get $V_{out} \approx V_{zero} + (V_A - V_B)$.

Op-Amp Multiplication and Division

The concept of analog computation can also be extended to multiplication and division. The most common and mildly mind-bending approach hinges on the fact that any positive number can be rewritten as a constant base n raised to some power; for example, 8 can be written as 2^3, while 42 is approximately $2^{5.3924}$.

From the basic properties of exponentiation, it's easy to show that $n^a \cdot n^b$ is the same as n^{a+b}. It follows that if we have two numbers represented as exponents of a common base, we can reduce the problem of multiplication to the addition of these exponents.

We already know how to build a summing circuit, so all we're missing is a way to convert a number to an exponent. We don't really care what base we're using, as long as the base remains constant for each operand. This brings us to the design shown in Figure 15-30.

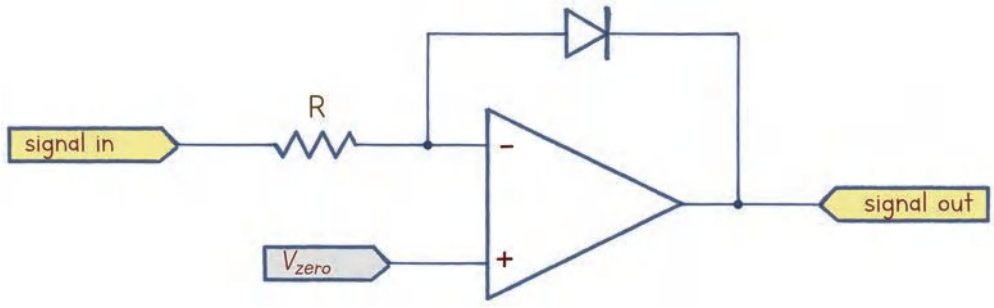


Figure 15-30: A logarithmic converter

As before, the linear equilibrium condition requires $V_{in-} \approx V_{in+}$. Let's assume that the initial input voltage is about equal to the reference signal V_{zero}. In this case, the output settles in the same vicinity.

Next, let's analyze what would happen if the input voltage increased by v_s = 100 mV. In such a scenario, for the op-amp to stay at an equilibrium of $V_{in-} \approx V_{in+}$, we would need a sufficient current to flow through the resistor to create a 100 mV voltage drop:

$$I_R = \frac{v_s}{R}$$

The op-amp has a very high input impedance, so the current must flow through the diode. If it doesn't, the circuit would move toward a condition of $V_{in-} \gg V_{in+}$, which would cause V_{out} to move toward the negative supply rail. That would forward-bias the diode and thus motivate it to conduct better. In other words, the circuit has an automatic mechanism that coerces the diode to admit the current matching I_R, and the amount of convincing is reflected in how much the output voltage has been reduced from the midpoint. We can denote this relative shift as v_o.

From Chapter 10, recall that although the relationship between the applied diode voltage and the resulting current is complicated, there is an initial region where the

component's V-I curve is exponential. For the common 1N4148 diode (refer back to Figure 10-1), this property holds up for currents up to about 1 mA. You might also see the circuit implemented with an unusual arrangement of a BJT with a grounded base; this works similarly but exhibits exponential behavior across a somewhat wider range of currents flowing through the component.

In the case of a diode, if the input resistor is large enough to keep the current in the safe range, we can say that the swing in the output voltage (v_o) will be dictated by the magnitude of an exponent of some constant base n that yields the correct diode current: $I_D = n^{v_o}$.

We also know that the current that must flow through the diode is proportional to the shift in the input signal (v_s) divided by R. This means that we've accomplished the number-to-exponent conversion between v_s and v_o. Or, in mathematical terms, we've calculated a logarithm.

To implement multiplication, we need two logarithmic converters on the input side, a summing amplifier to add the exponents, and an exponential converter that goes from the summed exponent back to a normal value. That last part can be accomplished by switching the location of the diode and the resistor in the log converter circuit we already have.

Integrator

Integration is just a fancy word for summing values over time—a bucket in your backyard "integrates" rainfall over the duration of a storm.

Although integration is important in calculus, analog integrators have more down-to-earth uses. For example, the circuits can convert square waves into triangular shapes that are useful in electronic musical instruments. The circuit's ability to produce very linear up and down slopes also comes in handy in analog-to-digital converters (coming up in Chapter 22).

A rudimentary integrator circuit is shown in Figure 15-31.

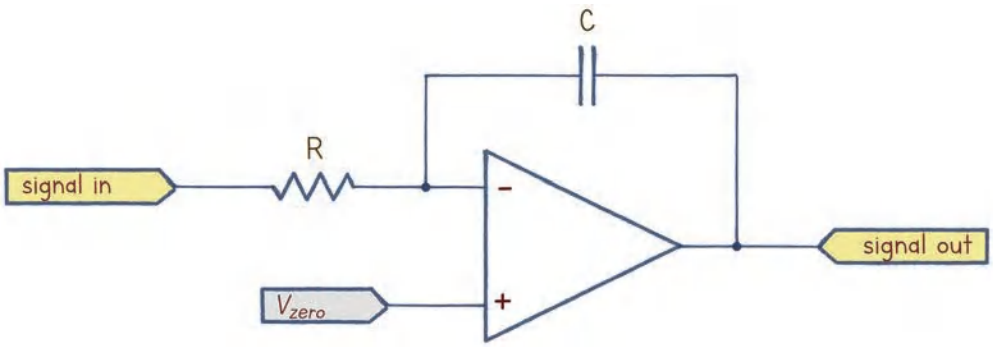


Figure 15-31: The textbook op-amp integrator

Once again, we can note that the linear operating condition is $V_{in-} \approx V_{in+}$. Further, let's assume that the input signal is equal to V_{zero} and the capacitor is discharged, so both the op-amp inputs and the output sit at about the midpoint.

Next, like the analysis of the log amplifier, let's assume that the input signal shifts up by v_s = 100 mV. For the op-amp to stay at equilibrium, we would need a sufficient current to flow through the resistor to create a 100 mV voltage drop: $I_R = v_s / R$.

The only possible path for this current is the capacitor. A capacitor doesn't admit steady currents, but it will allow the movement of charges during the charging process,

which will kick off when the op-amp's output voltage begins to drop, creating a voltage differential across the capacitor's terminals.

From the fundamental capacitor equation (Chapter 5), charging the capacitor with a constant current I_R for a time t will produce the following voltage across its terminals:

$$V_{cap} = \frac{I_R \cdot t}{C}$$

As per the earlier analysis, after substituting $I_R = v_s / R$, we obtain:

$$V_{cap} = \frac{v_s \cdot t}{RC}$$

To keep V_{in-} steady, the voltage to which the capacitor gets charged must be accounted for by a directionally opposite shift of the output voltage (v_o). The shift will persist after V_{signal} returns to the midpoint, because with no charging or discharging current, the capacitor just retains charge. The shift can be undone if v_s swings the other way around.

From the earlier formula for capacitor voltage, it should be clear that the circuit keeps a sum of (midpoint-relative) input voltages over time.

An Improved Integrator

The textbook integrator we've been working on has an inverted output: V_{out} moves down whenever V_{signal} moves up. This makes the design somewhat clunky to use in single-supply circuits. The problem can be addressed in a couple of intuitive ways, but a particularly neat solution is shown in Figure 15-32.

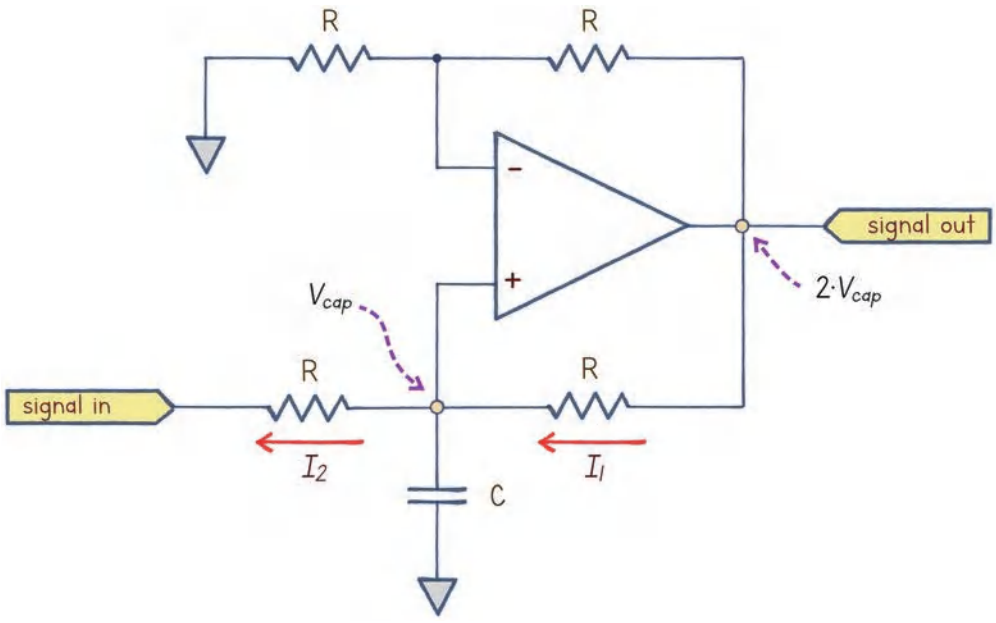


Figure 15-32: An improved noninverting integrator

As in all other cases, the prerequisite for linear operation is $V_{in-} \approx V_{in+}$.

We can start the analysis with the two-resistor divider on the top: It simply ensures that V_{in-} is equal to $\frac{1}{2} \cdot V_{out}$. For the bottom portion of the circuit, the instantaneous voltage on the noninverting input depends on the capacitor's charge state (V_{cap}).

The bottom resistors will influence the charge of the capacitor over time, but if we're living in the moment, we can combine the equations and write the following equilibrium rule:

$$V_{cap} \approx \frac{1}{2} \cdot V_{out}$$

Equivalently, we can write that $V_{out} \approx 2 \cdot V_{cap}$.

If V_{out} is always twice V_{cap}, this creates a voltage difference equal to V_{cap} across the terminals of the resistor on the bottom right. From Ohm's law, the resistor will admit the following current:

$$I_1 = \frac{V_{cap}}{R}$$

If the input voltage is zero, the neighboring resistor to the left is subjected to the same voltage differential, so the current flowing into the junction (I_1) is the same as the current flowing out (I_2). With the currents in balance, the capacitor holds its previous charge, and the output voltage doesn't change.

That said, if the input voltage (V_{signal}) is nonzero, the voltage differential across the terminals of the resistor on the left differs, and the formula for I_2 becomes:

$$I_2 = \frac{V_{cap} - V_{signal}}{R}$$

In this case, there is a nonzero balance of the currents (I_{net}) flowing in and out via the resistors:

$$I_{net} = I_1 - I_2 = \frac{\cancel{V_{cap}} - \cancel{V_{cap}} + V_{signal}}{R} = \frac{V_{signal}}{R}$$

This current is flowing in via the resistor on the right but not flowing out via the resistor on the left, so from KCL, it necessarily must take the third available path and charge the capacitor: $I_{cap} = I_{net}$. Note that this capacitor charging current is entirely independent of V_{cap}; it remains constant as long as the input voltage is constant too.

As before, from the fundamental capacitor equation ($V_{cap} = I_{cap} \cdot t / C$), we can tell that a constant charging current will cause the voltage on the output leg to ramp up in a straight line. We have a noninverting integrator that can be reset by shorting the terminals of the capacitor.

This covers the basics of operational amplifier applications. Many other circuits can be constructed with these chips, including precision current sources, better-behaved approximations of inductors or diodes, fancy signal filters, and so on. If you wish to explore these circuit ideas, many are cataloged in Texas Instruments' AN-31 application report, freely available at *https://www.ti.com/lit/an/snla140d/snla140d.pdf*. Otherwise, we're ready to dive into our next topic: the regulation of supply voltages.

16

SUPPLY REGULATION



In this chapter, we'll look at voltage regulation and conversion in DC circuits. Sometimes the supply voltage you're getting isn't what you need. For batteries, the choice is limited by cell chemistry; for inexpensive wall wart power adapters, 5 V and 12 V options vastly outnumber everything else. And that's if a single voltage suffices in the first place—what works for a motor might not suit a digital chip.

On the next couple of pages, we'll focus on underlying design principles rather than turnkey projects. We'll take this route because inexpensive, well-performing building blocks are readily available from many vendors. You can choose between monolithic chips that need just a couple of external components and miniature *point-of-load (PoL)* modules with all the necessary bits in a single package. Monolithic chips start at about five cents each; basic modules cost a buck or two. In other words, it's good to know how the sausage is made, but there's usually no need to raise your own pigs.

Linear Regulators

Linear regulators are the oldest surviving method of adjusting voltages in DC-powered circuits. These devices monitor the voltage on their output terminal and compare it to a stable internal reference. Depending on the measured error, the regulator adjusts the gate voltage of a built-in output transistor—essentially using it as a dynamic resistance to maintain a constant voltage drop between the positive supply rail and the connected load. This architecture can only produce voltages lower than the supply. A simplified model is shown in Figure 16-1.

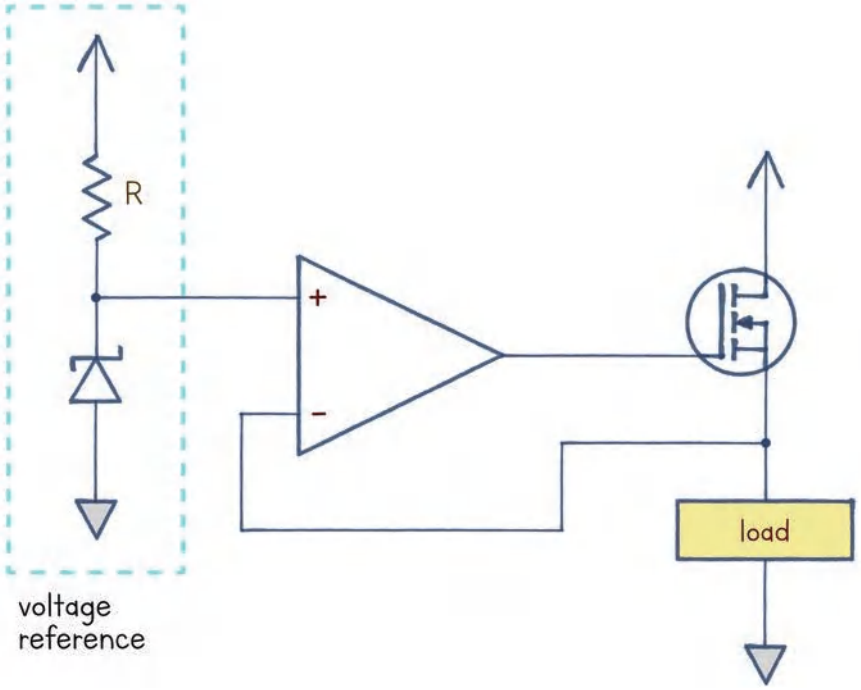


Figure 16-1: A rudimentary linear regulator implemented with an op-amp and a Zener diode

In practice, the internals of a modern *low-dropout (LDO)* regulator are a bit different. First, designers usually rely on a p-channel transistor and an inverting amplifier so that the regulated output voltage can stay very close to the positive rail (with an n-channel transistor, the minimum voltage drop is roughly V_{th}). This is the "low-dropout" part—the ability to work correctly with supply voltages close to the desired output voltage. Second, in an off-the-shelf chip, the internal voltage reference is more sophisticated, with temperature compensation and constant-current drive.

Many linear regulators also include thermal shutdown to prevent damage if the IC overheats. This highlights an important shortcoming: As noted earlier, the output transistor acts as a dynamic resistance. The regulator thus dissipates heat according to the power equation $P = IV$, where I is the load current and V is the required voltage drop. If these numbers are small, the losses are manageable. But past a certain point, waste heat becomes significant, and the chip might need a fan or a bulky heatsink. As a rule of thumb, linear regulators work best if current stays below 20 to 40 mA and voltage drop doesn't exceed a couple of volts. For everything else, the approaches discussed later in the chapter are usually better.

In a typical application, the efficiency of the linear regulator—the ratio of output power to input power—is limited by this resistive process and can't exceed the ratio of output voltage to input voltage. For example, an LDO stepping 12 V down to 5 V can't exceed about 42 percent efficiency.

Although linear regulators are integrated circuits, they're often packaged identically to transistors. The package has one pin for the positive supply rail, one for ground, and one for the regulated output voltage. For stable operation, you need to place a ceramic capacitor between the output and ground, both to give the regulator a predictable load and to buffer rapid changes in demand. The device's specs will typically recommend a capacitance of around 1 µF; it's wise to add a similar capacitor on the input side too.

Linear regulators are fairly simple tools, so you only need to look at a handful of parameters in the datasheet. Pay attention to the input voltage range, the output voltage (which may be adjustable in some designs), regulation accuracy (usually around 1 percent), and maximum current. The plot of power-supply ripple rejection versus frequency might also be worth a look; linear regulators usually keep up with ripple only up to about 1 kHz.

There are plenty of LDOs to choose from, but if you're looking for a modern low-power solution, I like the MCP1700 series from Microchip. The chips are fairly inexpensive, have good specs, and come in both surface-mount and through-hole packages.

Pulse-Width Modulation

As discussed earlier, linear regulation is inherently wasteful. The simplest way to avoid those losses is not to regulate the voltage in the first place. For example, instead of lowering the voltage supplied to a DC motor, we can change its speed by rapidly cycling the power on and off while altering the duty cycle—that is, the per-cycle "on" time of the driving waveform. This control scheme is known as *pulse-width modulation (PWM)* and is illustrated in Figure 16-2.

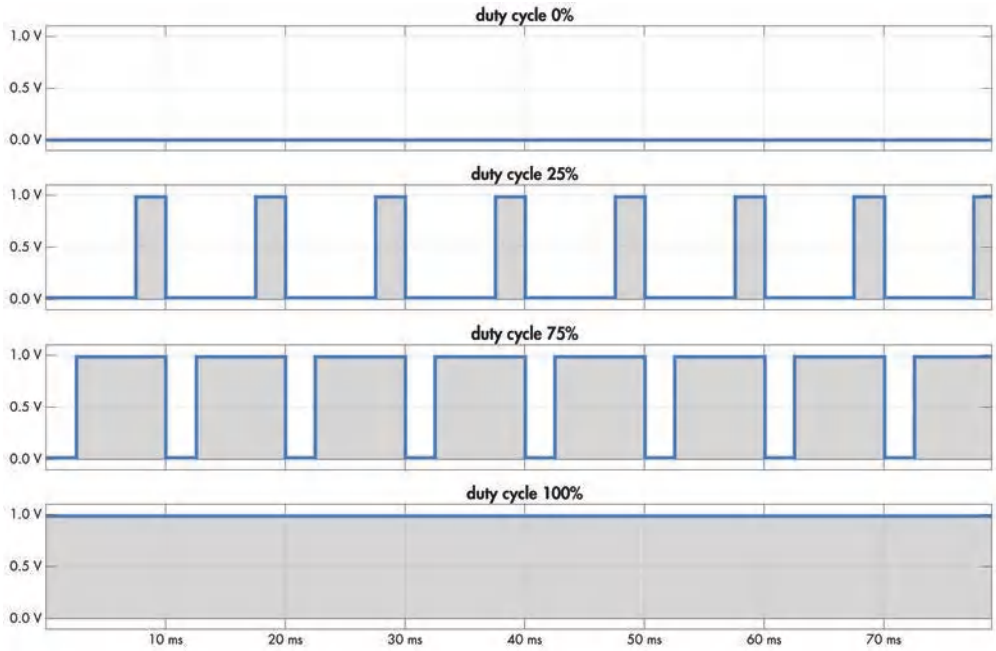


Figure 16-2: Pulse-width modulation with duty cycles of 0, 25, 75, and 100 percent

This rudimentary approach has its limits. We can't run a 5 V digital circuit by connecting it to 110 V for one second and then letting it cool off for half a minute. Still, PWM works well if the supply voltage is safe for the load and we just want to regulate the effective rate of work performed—the speed of a motor, the intensity of an LCD backlight, and so on.

When used for motor speed control, the inductance of the windings and rotor inertia typically ensure smooth motion at PWM frequencies as low as 1 to 2 kHz. For light dimming, the flicker fusion threshold creates the appearance of constant

intermediate brightness at frequencies as low as 50 Hz—although 200 to 400 Hz is needed to avoid motion artifacts and eyestrain.

In theory, PWM can be nearly 100 percent efficient. In practice, it's hard to switch transistors on and off instantly: To change the component's V_{GS}, we must overcome its parasitic gate capacitance. For example, with a MOSFET having 1 nF input capacitance driven by a digital chip outputting at most 20 mA, the fully off to fully on transition takes about 250 ns in a 5 V circuit. We get this estimate from the capacitor equation ($V = I \cdot t / C$), solving for t.

At 10 kHz PWM, the 250 ns overhead matters little: The transistor will be at an intermediate V_{GS} for only about 0.5 percent of each cycle. On the flip side, if the frequency is 1 MHz, the figure balloons to 50 percent. In these half-on and half-off states, the transistor's resistance is higher than expected, and we'll see more energy dissipated as heat.

Pulse-width modulation must be tailored to specific applications, so few if any "universal" PWM chips exist. Instead, the technique is frequently integrated into more specialized products, such as motor controllers (Chapter 17). As for one-off needs, it's possible to tinker with analog oscillators, but the scheme boils down to toggling between "zero" and "one," so it's more conveniently implemented on any programmable digital chip. We'll cover how to do this in Part V of the book. For now, let's look at several conceptually related approaches that allow us to change voltages when simple PWM won't suffice.

Charge Pumps

Charge pumps are switched-capacitor circuits that multiply or divide supply voltages. They are simple, typically boast efficiencies above 90 percent, produce little electrical noise, and can be etched directly in silicon. For these reasons, this type of DC–DC converter is common inside integrated circuits—including some op-amps and many digital chips. A conceptual diagram of a voltage-doubling charge pump is shown in Figure 16-3.

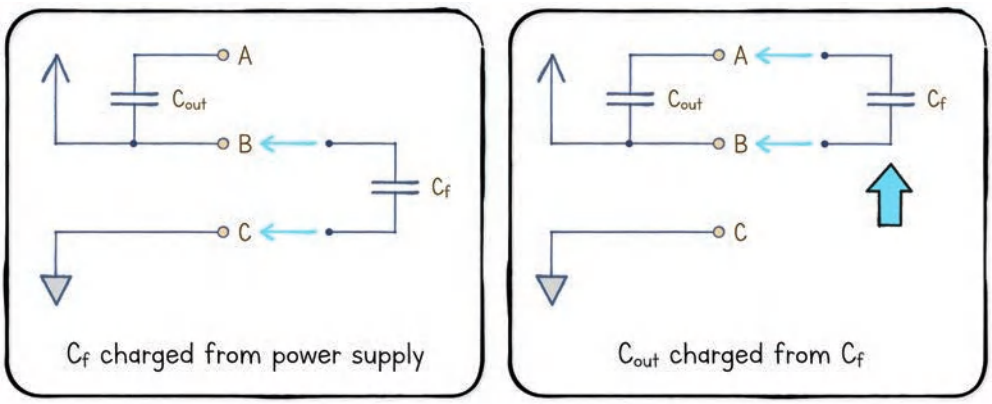


Figure 16-3: A conceptual illustration of a charge pump

In the panel on the left, we see a C_{out} capacitor that sits idly while a "flying" capacitance C_f is charged to the supply voltage. This voltage is internal to the capacitor: We can unplug it, carry it in our pocket, and then hook it up to another circuit to power it for a brief while.

In the second panel, C_f is disconnected from the supply and hooked up to the terminals of C_{out}. If the voltage across C_{out} is lower than across C_f, the process transfers some charge to the output capacitor. After several round trips, the V_{AB} voltage approaches V_{supply}. We know that the voltage between points B and C (V_{BC}) equals V_{supply}, so V_{AC} must equal $V_{AB} + V_{BC} = 2 \cdot V_{supply}$. The circuit is a voltage doubler. The back-and-forth motion of C_f ensures that the charge in C_{out} is continually replenished; the amount of output ripple can be controlled by sizing the capacitors and choosing the circuit's operating frequency to match the intended load.

The design is a bit mind-bending, so it's good to experiment on a breadboard. Two large caps—perhaps 470 µF—are a good starting point; a 100 kΩ resistor can simulate a load.

Of course, practical charge pumps don't actually move capacitors around. Instead, they use transistors operating as switches to alternately connect C_f to the supply and to the output capacitor (Figure 16-4). The transistors can be driven by a simple relaxation oscillator or by a programmable digital chip; the switching frequency is usually tens to hundreds of kilohertz—low enough to make switching easy but high enough to allow reasonably compact capacitors, usually around 1 to 10 µF.

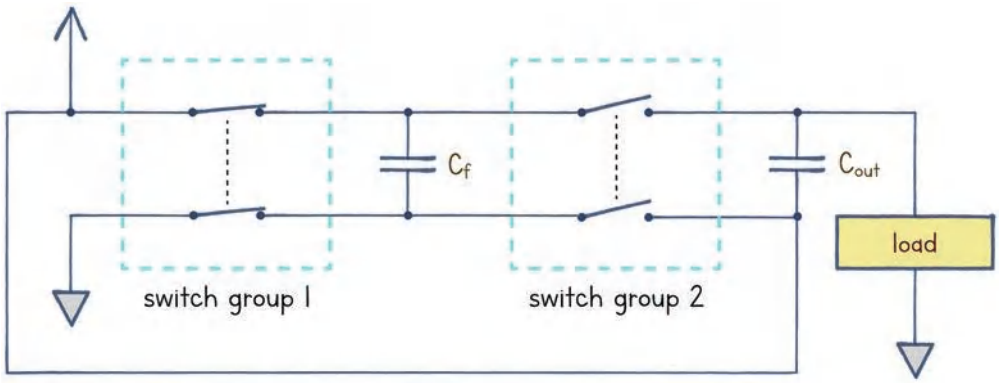


Figure 16-4: A charge pump voltage doubler implemented with pairs of ganged switches

A similar circuit can be used to produce negative voltages: We do this by dangling C_{out} from the negative supply rail instead of orienting it on top of the positive one. This effectively places the capacitor's lower terminal at $-V_{supply}$.

A more perplexing variation of the charge pump circuit is the voltage-halving arrangement shown in Figure 16-5. To grasp the design, let's ignore C_f for the time being and look just at the pair of series capacitors, C_1 and C_2. When the two capacitors are first connected to the power supply, some common charging current flows through both of them. If the value of C_1 equals the value of C_2, that current produces the same charge in each capacitor, so when charging finishes, $V_{AB} \approx V_{BC} \approx V_{supply} / 2$.

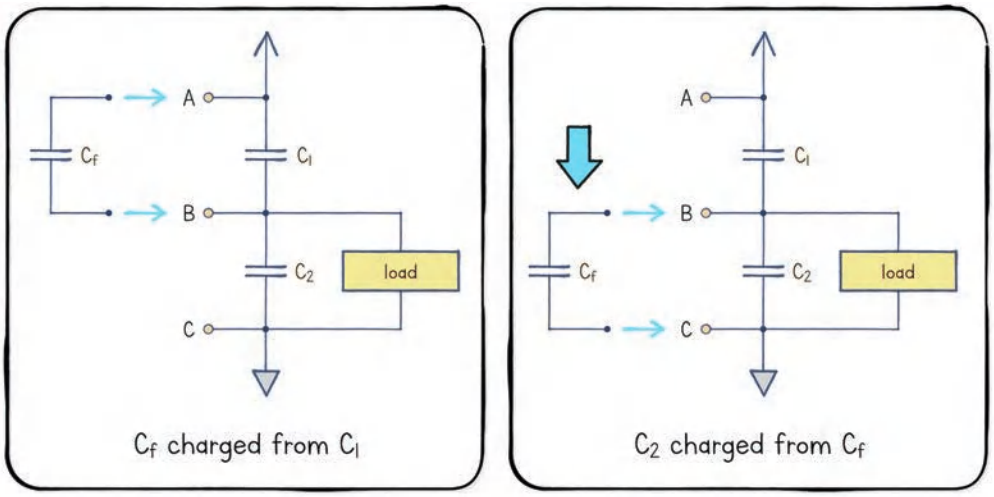


Figure 16-5: A charge pump voltage-halving circuit

This might sound like a voltage divider, but it's not guaranteed to stay that way. Once charging from a DC source stops, nothing forces the charge in C_1 to remain the same as in C_2. In particular, if we connect a load between points B and C—as shown in the figure—the bottom capacitor discharges. Further, because the load creates a path to ground from point B, the top capacitor charges to a higher voltage. Following this action, the end state that obeys KVL in the circuit is $V_{AB} = V_{supply}$ and $V_{BC} = 0$ V.

This is where the flying capacitor C_f comes into play. If it moves back and forth between C_1 and C_2, it charges from the capacitor at higher voltage and discharges into the one at lower voltage. This motion replenishes charge in C_2, allows steady current to flow through the connected load, and maintains the equilibrium of $V_{AB} \approx V_{BC} \approx V_{supply} / 2$. Similarly to the voltage doubler case, the circuit makes for an interesting perfboard experiment.

Charge pumps are easy to build and operate from scratch, but you can also buy ready-made integrated circuits that combine high-frequency switching logic and four drive transistors on a silicon die. These chips usually require just two external capacitors to operate. Run-of-the-mill examples include Texas Instruments' LM2766 (voltage doubler) and LM2776 (negative voltage generator).

Although the architecture is efficient and produces little noise, it can only produce multiples or fractions of the supply voltage. Additional regulation is needed to trim it to a specific absolute value if the supply fluctuates over time. Another drawback is that charge transfer is inherently discontinuous and depends on voltage differentials: The closer the output voltage is to target, the less charge C_f can deliver each cycle. For this reason, the architecture is less suited for high-current applications and rarely serves loads higher than perhaps 200 mA. Beyond that, inductor-based approaches take the lead.

Buck Converters

A *buck converter* is a circuit that uses a switched inductor to deliver charge to a capacitor in carefully measured increments. The basic architecture of the converter is shown in Figure 16-6.

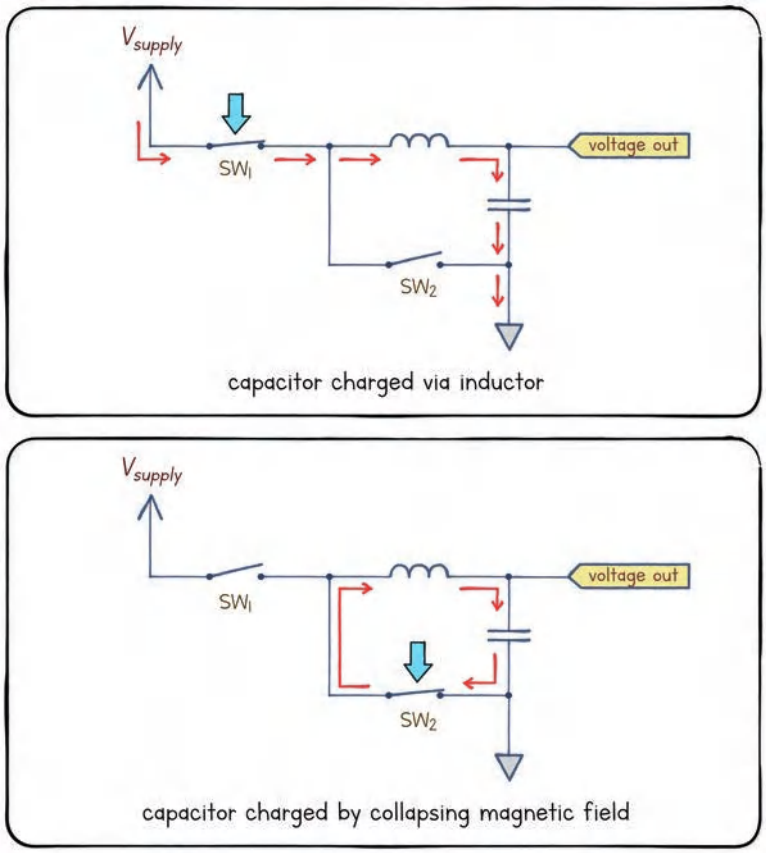


Figure 16-6: The operating cycle of a buck converter

The circuit rapidly cycles between two states. In the first half of the operating cycle, the supply-side switch (SW_1) is closed, and the inductor limits the rate at which charging current ramps up—allowing the capacitor's charge state to increase in an orderly fashion. We could achieve a similar effect with a series resistor, but it would dissipate energy as heat. In an inductor, the supply's energy is stored in an internal magnetic field.

This brings us to the second half of the cycle. To pump this stored energy into the capacitor, the circuit disconnects from the supply by opening SW_1. The collapsing magnetic field will try to sustain a current between the terminals of the inductor, so when SW_2 briefly closes, an additional predictable unit of charge moves to the capacitor plates. For simpler control at the expense of slightly reduced efficiency, SW_2 can be replaced with a reverse-biased diode.

Compared to charge pumps, the control of buck converters is more complicated. The output voltage needs to be monitored, and the "on" time of SW_1 must be adjusted to keep the output voltage on target under all load conditions. The timing of SW_2 needs to be dialed in correctly too: Keeping it closed for too long risks capacitor discharge once the magnetic field is gone. Because of this, switching regulators are usually implemented using specialized controller chips. The devices typically achieve around 90 percent efficiency—but only for certain ranges of output currents. If the output demand is low, power is wasted by the control logic, and efficiency can drop to 50 percent or less.

A major advantage of buck converters is the continuous current path from supply to output and a persistent voltage difference across the terminals of the inductor,

making high-current loads easier to accommodate. In some other respects, performance can suffer: Resistive and magnetic losses in inductors exceed those in capacitors, especially with small, low-cost components optimized for PCB footprint. More energy is also radiated away as RF interference. Historically, most buck converters operated at frequencies of around 50 kHz, where the noise remained relatively easy to control. Today, the push for cheaper, smaller inductors drives frequencies to 2 MHz, requiring careful PCB design to avoid wideband noise.

Buck converters are available as chips requiring an external inductor and capacitor (examples include Texas Instruments' TPS561243 and Diodes Inc.'s AP62200); some may also need an external transistor. Another option is self-contained point-of-load units such as the Recom RFM series, XP Power RBT05W, or Murata NME1. The modules are pretty affordable and easier to use, so I recommend sticking to that option unless you need to keep the design as cheap as possible or solve some other unusual design constraint.

Because these devices usually have some form of sensing and recovery for short-circuit conditions that could overheat the inductor or the output transistors, the IC may limit maximum output-side capacitance to ground (usually 100 uF to 1 mF). A capacitor larger than the limit may take too long to charge, pushing the module into recovery regime. This may involve repeatedly cycling the power on and off until the apparent fault clears—a so-called *hiccup mode*.

Boost Converters

Buck converters work only if the desired output voltage is lower than the input voltage. If the output capacitor is ever charged above V_{supply}, toggling SW_1 would cause the current through the inductor to flow in the opposite direction, discharging the capacitor via the positive supply rail. A *boost converter* is an architecture that overcomes this limitation to produce voltages higher than the supply. The basic layout is shown in Figure 16-7.

In the first stage of operation (top panel), the inductor is connected between the positive supply and ground, storing energy in a magnetic field at a rate independent of the output capacitor's charge state. In the second stage (bottom panel), SW_1 opens and SW_2 closes. At that point, the collapsing magnetic field in the inductor tries to sustain the earlier current. Because the component's left terminal is connected to the positive supply rail, this necessarily puts the right terminal at a relatively higher voltage, pushing additional charge carriers onto the plates of the capacitor.

Boost converters tend to be less efficient than buck converters while sharing the same disadvantages. For low-power applications, charge pumps are typically preferable. Still, the architecture is sometimes employed to generate higher voltages for LCD backlights or OLED displays, or to run digital electronics off a single AA battery. Examples of boost converter ICs include the AP3012 from Diodes Inc. and the MIC2288 from Microchip. Again, the efficiencies of these designs are lower, typically around 80 percent.

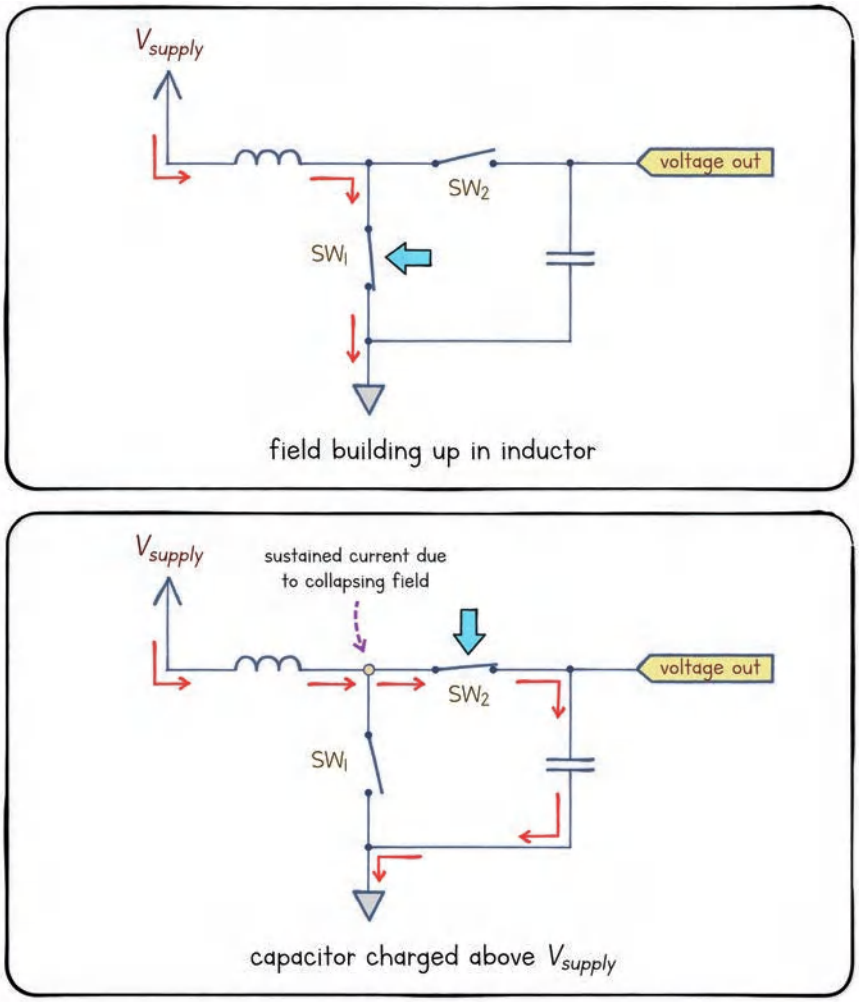


Figure 16-7: The operation of a boost converter, where SW_2 can be a forward-biased diode

Many other DC–DC converter topologies exist, although they're less common now. For example, small transformers operated at high frequencies are sometimes encountered in circuits used to generate high voltages. This used to be popular in CRT TVs and fluorescent LCD backlights, but both of these technologies are now largely extinct; perhaps the most common surviving applications of the technology are engine ignition systems and stun guns. Because of this, we won't dwell on these topologies in this chapter. Instead, we'll tie up a couple of remaining loose ends in the analog realm in Chapter 17 and then step into the exciting world of digital circuitry.

17

OTHER ANALOG ICs



To wrap up Part III of the book, let's look at some less common analog chips you might encounter in your adventures ahead. Operational amplifiers, voltage references, regulators, and DC–DC converters make up the bulk of what's out there, but there's a bit more variety on the tail end of circuit design.

The number of true analog ICs shrinks every year. Had I published this book a decade ago, this chapter would have needed to be much longer. The reason for the trend is that programmable digital electronics are getting faster, cheaper, and more versatile, so in modern consumer products, most analog signal processing is done by converting the signal to bits and bytes and then doing the magic in the digital realm. We'll talk about the tech used to perform these conversions soon—but first, let's look at some of the analog holdovers that are worth knowing about.

Timers and Oscillators

In Chapter 14, we noted that it's relatively hard to clobber together a competent oscillator using discrete semiconductors. Back when op-amps and microcontrollers were still in their infancy, salvation came in the form of the NE555, a versatile chip developed by Signetics (now Philips). The chip contained almost all the necessary biasing, amplification, and trigger logic needed to construct relaxation oscillators, pulse generators, and time-delay circuits. For example, to construct an oscillator, the chip needed just two external resistors, two capacitors, and a couple of wires.

In the following decades, the NE555 gained a cult following among industrial designers and hobbyists. You could buy entire books written about it, such as *The 555 Timer Applications Sourcebook* by Howard Berlin (Sams Publishing, 1978). If you're

interested in its design but don't have time for a vintage book, this web article is a great read: *https://cdn.evilmadscientist.com/KitInstrux/555/555_principles_rev30.pdf*.

By the turn of the century, the 555 party ended. Inexpensive microcontrollers allowed timing signals to be produced more efficiently, more flexibly, and with fewer components on a PCB. Variants of the chip are still made by several companies, and you will undoubtedly encounter them in hobby circuits, but there's little reason to use one in new designs.

For applications needing a variable-frequency square wave without microcontroller code, the 555 will do, but simpler alternatives exist. For example, the LTC1799 from Analog Devices can generate square-wave signals at frequencies from 1 kHz to 33 MHz with just a single external resistor—versus multiple components for a 555. That said, as a niche product, the LTC1799 will set you back about $4; a programmable chip that can do roughly the same costs $0.75 or less.

Interface Chips

An important family of analog chips is designed specifically for driving loads that require substantial currents or voltages or are otherwise challenging to control. Perhaps the simplest example is line drivers designed to switch high-current loads on and off. A classic is the ULN2003 transistor array, which contains seven independent drivers, each capable of handling 50 V and 500 mA. Purpose-built interface chips are also common for implementing higher-voltage or long-range communication protocols—Ethernet, automotive CAN, and so forth.

A more complex example of the same approach is *full-bridge motor drivers*. For a brushed motor, a driver such as Texas Instruments' DRV8231 combines four transistor switches to allow the polarity of the supplied voltage to be reversed and the motor to spin in both directions. In the same vein, for three-wire brushless motors, ICs such as Monolithic Power Systems' MP6637 will have six output transistors to toggle the coils in the right order and polarity, as needed to make the rotor spin in the first place. In both cases, the controllers may offer internal PWM speed control and overload sensing, further simplifying circuit design.

Another unusual but noteworthy example of interface chip is the *optocoupler*: an LED and a phototransistor sealed in a light-tight package. This four-terminal device provides complete electrical isolation between the LED side and the phototransistor side of the circuit, but it still allows analog or digital signals to be relayed using light. This type of coupling can be used for safety reasons or to contain electrical noise in sensitive measurement gear.

Switched-Capacitor Filters

Several manufacturers make high-order low-pass and high-pass chips with a configurable frequency response; examples include the Texas Instruments TLC04CD and Analog Devices MAX7410. The ICs offer an attractive alternative to building multistage filters by hand.

The chips are interesting not only because they simplify circuit design but also because they use a switched capacitor architecture in lieu of the traditional R-C layout. This is in part because manufacturing precise resistances is difficult on a semiconductor die. The basic operation of a switched-capacitor low-pass filter is shown in Figure 17-1.

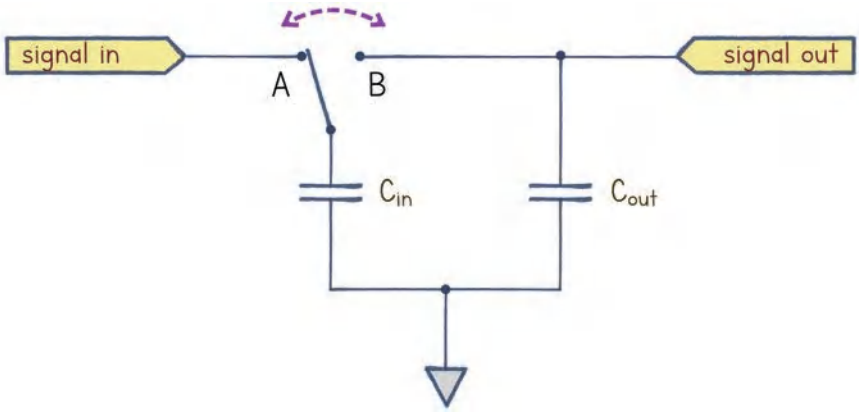


Figure 17-1: A conceptual illustration of a switched-capacitor low-pass filter

The architecture is simple yet mystifying. In the first half of the timing cycle, the switch is in position A. This connects a small input capacitor, C_{in}, to the input terminal. As soon as the connection is made, the capacitor charges to a voltage equal to the momentary level of the input waveform, which we can denote as V_A.

In the second half of the cycle, the two-way switch is moved to position B. This causes the voltages across C_{in} and C_{out} to equalize. We don't need to solve for the relative shifts in the capacitors' terminal voltages; it suffices to say that C_{in} started at V_A and ended at V_B.

With this in mind, we can apply a form of the fundamental capacitor equation ($I = C \cdot \Delta v / t$) to find the average current that must have flowed out of C_{in} to shift its voltage by that unknown but well-specified amount in time t:

$$I_B = \frac{(V_A - V_B) \cdot C_{in}}{t}$$

The period t is the reciprocal of the capacitor switching frequency f_s, so we can also restate this as $I_B = (V_A - V_B) \cdot C_{in} \cdot f_s$.

The formulas tell us that the average current is proportional to the voltage present across the A and B terminals of the switch. The actual current is pulsed, but its magnitude is evidently governed by two constants and the difference between two voltages, so it looks like the ohmic behavior of a series resistor. In fact, from Ohm's law, we can find the equivalent resistance that appears to be feeding C_{out}:

$$R_{in} = \frac{V_{\text{switch}}}{I_{\text{switch}}} = \frac{V_A - V_B}{I_B} = \frac{1}{C_{in} \cdot f_s}$$

If the circuit can be modeled as a series resistor feeding a shunt capacitor, we're essentially looking at a bog-standard low-pass R-C architecture. The knee frequency of this R-C filter can be written as:

$$f_{knee} = \frac{1}{2\pi R_{in} C_{out}} = \frac{C_{in}}{C_{out}} \cdot \frac{f_s}{2\pi}$$

In other words, the frequency depends only on the ratio of two on-die capacitances and the supplied digital clock. These parameters can be controlled with greater precision than the value of a single resistor.

The math might seem improbable, but the circuit works in practice. The oscilloscope traces in Figure 17-2 show a filter constructed with C_{in} = 100nF and C_{out} = 1µF, toggled at f_s = 50 Hz. This corresponds to $f_{knee} \approx 0.8$ Hz—and indeed, we see some attenuation for a 1 Hz input sine and a lot more for a 5 Hz one.

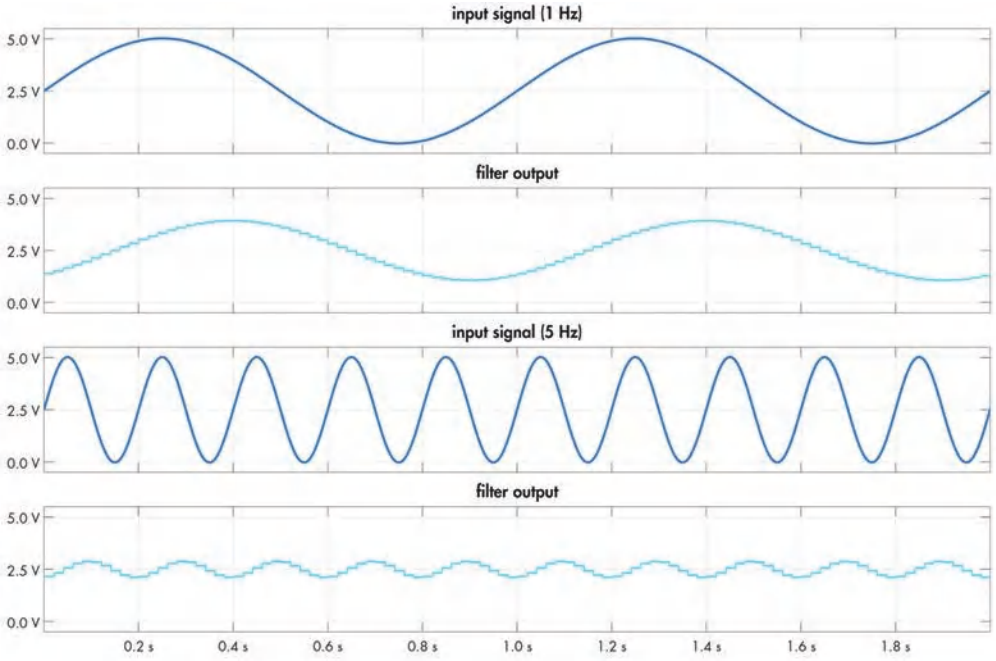


Figure 17-2: Oscilloscope traces showing the behavior of the switched capacitor low-pass filter

The plot offers an intuitive validation of the math. In each clock cycle, only a certain amount of charge can move between the capacitors; this corresponds to the maximum height of a single step in the output waveform, proportional to the relative sizing of the caps (the ratio of C_{in} to C_{out}). The larger the vertical step, the easier it is for the output waveform to track a fast-moving input signal. The switching frequency f_s controls how many charge transfers occur per second, which gives us another method of controlling the filter's center frequency: To shift it, we can speed up or slow down the supplied clock. That's a major boon for digitally controlled analog signal processing.

In practice, to minimize the stair-step pattern, we tend to choose the switching frequency f_s to be about two orders of magnitude higher than the filter's intended center frequency. To achieve this, from the earlier f_{knee} formula, we need to aim for $C_{out} \approx 16 \cdot C_n$.

Sensors

A wide range of sensing ICs exist. We won't dive into their full taxonomy and application notes, but the most intriguing and occasionally useful category includes magnetic field sensors and MEMS accelerometers. Magnetic sensors can be used to detect the position of moving assemblies that contain embedded permanent magnets; if sensitive enough, these sensors can even measure Earth's magnetic field. Similarly, accelerometers, which measure the minute changes in capacitance caused by the deflection of an internal micromachined structure, are popular for inertial navigation and gesture sensing in smartphones, gaming handhelds, and wearables.

The sensors are fundamentally analog, though some chips internally convert the reading to digital signals. For analog accelerometers such as Analog Devices' ADXL325BCPZ, the output is typically an amplified voltage that sits near the midpoint of the supply when the device is not subjected to any acceleration. Analog magnetic sensors are similar; a good example is Texas Instruments' DRV5053.

Odds and Ends

In your journeys, you will sooner or later encounter a couple of other chips that are hybrids of digital and analog designs. For example, *digital potentiometers* use digitally controlled transistors to selectively bypass some resistors that are connected in series between two analog input terminals. This allows convenient, stepwise control of the overall resistance of the signal path. The technique is handy for controlling audio volume or adjusting sensor gain. Another hybrid design is the *analog switch*, in which a pair of complementary transistors can be operated to create a bidirectional current path between select input and output pins.

You will also come across *comparators*: op-amps meant to be used without feedback, simply indicating whether one of the input voltages is higher than the other (for a visual, refer back to Figure 15-1 in Chapter 15). Comparators usually have different design trade-offs than "proper" operational amplifiers: They may be designed to be relatively slow but to have very low offset voltage. They might also lack a push-pull output stage, which helps conserve power in certain uses, such as battery supervisor ICs that cut off power to the device if the voltage drops below a certain threshold. Supervisory circuits may be used to allow the protected device to shut down gracefully or to protect rechargeable batteries from being discharged past the point of no return.

With this, we wrap up the discussion of analog circuitry and embark into the world of bits and bytes. We'll revisit analog-to-digital and digital-to-analog conversions in Chapter 22, but for the most part, from now on, we'll be surfing the cyberwave.

PART IV

THE DIGITAL REALM

18

LOGIC GATES



In this part of the book, we'll focus on digital electronics. Digital circuits seek to transcend the ickiness of analog devices by developing methods to store, replicate, and transform information in a way that's free of the incremental errors—thermal noise, radio interference, amplifier nonlinearities—that inevitably accumulate in the analog realm.

In this chapter, we'll outline the principles and the basic functional blocks for digital computation. In the remainder of Part IV, we'll put these principles to use, showing how to manipulate and store digital information in a variety of useful ways.

Defining Digital Circuits

The fundamental idea behind digital computation is to represent information using a finite alphabet of symbols. This allows analog errors to be eliminated simply by snapping the inputs and outputs to the nearest valid symbol, thus restoring the correct meaning of the signal every step of the way.

As a practical example, if a circuit uses a four-symbol alphabet where 0 V stands for "A," 1 V means "B," 2 V corresponds to "C," and 3 V stands for "D," we could interpret any input voltage between 1.6 and 2.4 V as symbol "C." This would give us a margin of +/−400 mV for transmission losses, radio interference, and other sources of noise. The circuit would then output the correct, adjusted voltage, possibly after performing some additional calculations on the input data first.

Most digital electronics use a two-symbol alphabet known as *binary code*, with the symbols customarily written as "0" and "1." Zero typically corresponds to a voltage close to the negative supply rail, and 1 is signified by a level close to the positive rail. This scheme is dominant because it allows for the simplest circuit design and maximum

noise immunity at the lowest operating voltages—conditions needed to pack the highest number of transistors into the smallest possible physical envelope. Digital circuitry that uses more than two voltage levels is comparatively rare but not unheard of; for example, some solid-state drives allow every cell to have four or more possible states.

A conceptual model for a nearly ideal digital input stage is an op-amp used as a voltage comparator (Figure 18-1). In this circuit, any input signal slightly below 50 percent of the positive supply voltage causes the output to swing toward ground, while any input slightly above the threshold drives the output toward V_{supply}.

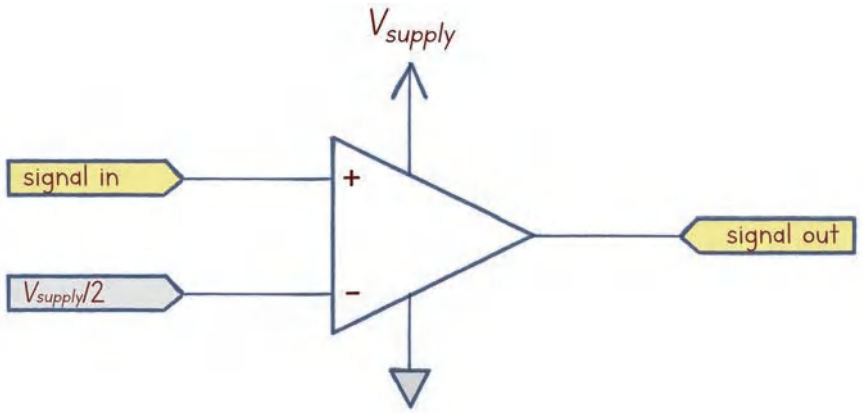


Figure 18-1: A near-ideal binary input buffer

In practice, a full-fledged op-amp would be overkill for digital logic: The circuit uses too many transistors, consumes too much power, and has a limited response speed. Instead, digital chips use much simpler transistor arrangements with lower gain, which places the maximum "0" voltage and the minimum "1" voltage farther apart and leaves a forbidden analog region in between. In the now-dominant MOSFET architecture (*complementary metal-oxide-semiconductor*, or *CMOS*), "zero" typically falls within 0 to 30 percent of V_{supply}, "one" is represented as a voltage between 70 and 100 percent of the positive supply rail, and the 30 to 70 percent territory is no-man's land.

Figure 18-2 shows the relationship between the input voltage and the output signal for a simple CMOS inverter chip, the 74HCU04. The chip is designed to output a logic level opposite the binary value presented at the input—and as shown in the oscilloscope trace, it does so only when the input voltage stays within a certain range. "Illegal" analog input signals in between produce intermediate output voltages.

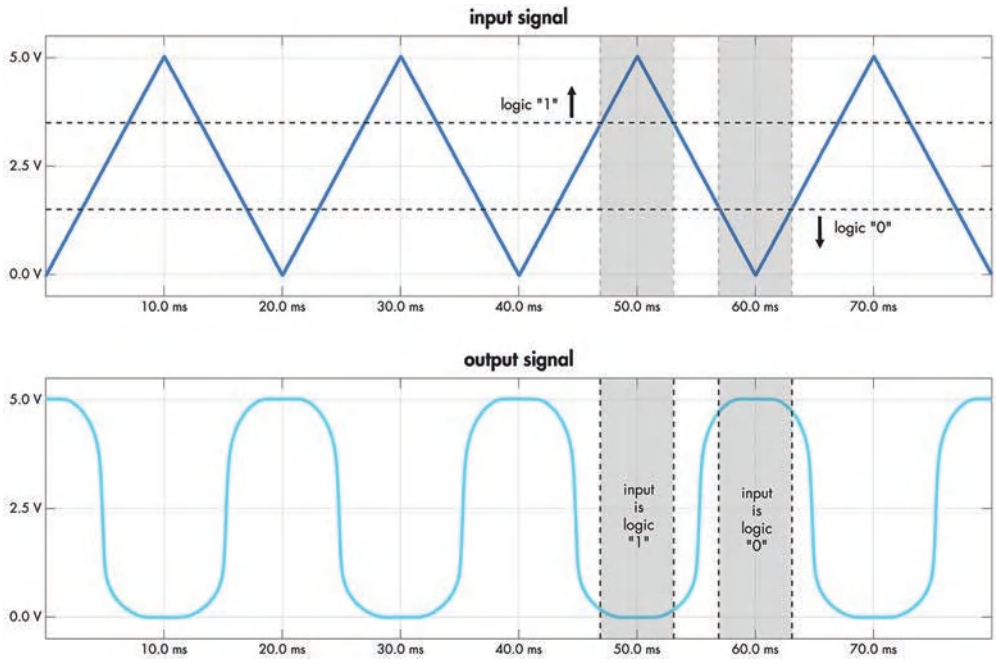


Figure 18-2: Input voltage versus output voltage for a 74HCU04 inverter chip

The analog nature of digital chips can show up in a few other ways. For one, the output stages of these chips typically have a modest maximum current capacity—perhaps 20 to 40 mA—so the voltage will sag if the chip is connected to a power-hungry analog load or to too many digital inputs on other chips. The devices also have a maximum operating speed, typically determined by internal signal propagation delays and transistor switching rates. For example, the popular 74HC series of basic CMOS logic may not operate properly with digital signals faster than around 25 MHz.

Truth Tables

Truth tables are a simple analytical tool for understanding the behavior of digital logic. They are constructed by listing all possible input states and annotating each row with the corresponding output value. For example, for a rudimentary inverter that converts "0" to "1" and "1" to "0," we can write what's shown in Table 18-1.

Table 18-1: Truth Table for an Inverter

Input	Output
0	1
1	0

Table 18-2 shows a slightly more advanced example of the same notation: the states of a two-input architecture that outputs "1" if and only if one of the inputs is "1."

Table 18-2: A More Complex Truth Table for a Two-Input Logic Circuit

Input A	Input B	Output
0	0	0
0	1	1
1	0	1
1	1	0

Truth tables have their limits; you can't realistically build one for a CPU because the number of possible input combinations is astronomical. Even so, we can use this approach to analyze the behavior of individual building blocks in more complex digital architectures—and that's the approach we'll take in the next couple of chapters, culminating in the outline of a general-purpose computing device.

Logic Gate Design

A *logic gate* is a rudimentary, reusable digital circuit that takes one or more digital inputs and performs a basic logic operation on the input data. The language of logic gates is essentially the same as the language of formal logic if we take "1" to mean "true" and "0" to mean "false." For example, an inverter circuit is equivalent to the mathematical negation operator (¬). We need not dwell on the theoretical underpinnings of digital circuitry or introduce a new lexicon of abstract symbols; instead, we'll review a few common designs and then move on to combining these building blocks in useful ways.

Buffer and Inverter

The simplest single-input logic gates are *buffers* and *inverters*. A buffer takes an input signal and passes it through unchanged (Table 18-3). Although it's a digital logic gate, it's essentially a no-op, so when it appears in a circuit, it's often for analog reasons—for example, to introduce a slight signal delay, clean up a noisy signal, or supply additional current to drive an analog load.

Table 18-3: Truth Table for a Buffer

Input	Output
0	0
1	1

A more practically useful single-input logic circuit is an inverter, also known as a *NOT gate*. As discussed earlier, it works like a buffer except that it outputs a flipped version of the input signal: "1" for "0" and "0" for "1" (refer back to Table 18-1).

In the common CMOS architecture, of the two gates, the inverter is easier to build. The typical two-transistor implementation is shown in Figure 18-3. The circuit consists of a pair of complementary MOSFET switches: The top p-channel switch connects the common output line to the positive supply rail when V_{in} is low, and the bottom n-channel transistor connects it to the ground rail when V_{in} is high.

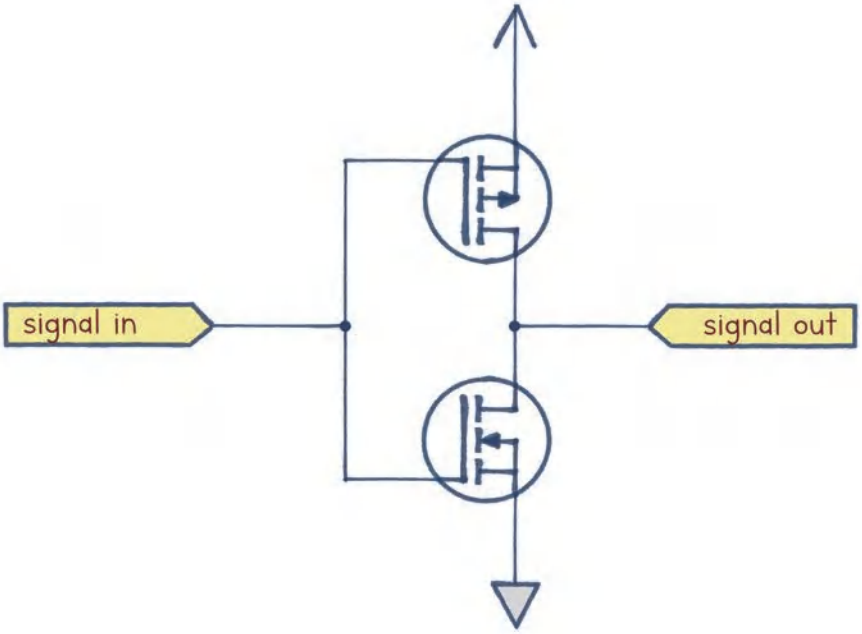


Figure 18-3: A standard CMOS inverter architecture

As noted earlier, intermediate input voltages leave both transistors partly on, producing invalid output voltages. The transistors usually have an "on" resistance high enough that the resulting short-circuit does not damage the chip.

The simplest way to build a buffer is to chain two NOT gates—double negation cancels out. We can express the behavior of the chained circuit as follows:

$$f(a) = \text{NOT}\big(\text{NOT } a\big) = a$$

The standard symbols for buffer and inverter gates are shown in Figure 18-4. In circuit schematics for digital logic, supply rails are almost always omitted; however, if a circuit has multiple supplies, some annotation should be included to clarify which one is reserved for logic gates.

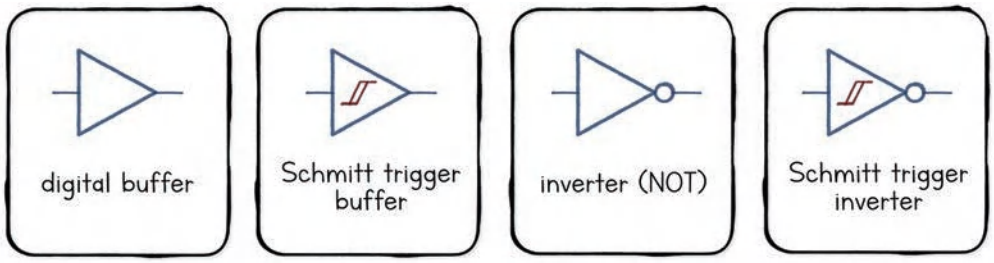


Figure 18-4: Symbols for buffers and inverters

For both buffers and inverters, a specialized version of the circuit can be constructed with a Schmitt trigger input stage, as discussed in Chapter 14 and Chapter 15. This design offers improved noise immunity and eliminates problems associated with intermediate input voltages.

An example of a basic six-inverter chip is the 74HC04; its Schmitt trigger counterpart is the 74HC14. Standard buffer ICs are less common because they nominally don't do anything useful, but specialized variants with added features do exist. For example, the 74HC125 includes an additional $\overline{OE}$ (output enable) terminal for each buffer. The buffer operates normally when $\overline{OE}$ is pulled low but disconnects from the output pin when $\overline{OE}$ is high. Table 18-4 illustrates this behavior.

Table 18-4: Truth Table for the 74HC125

Input	$\overline{OE}$	Output
0	0	0
1	0	1
0	1	Hi-Z
1	1	Hi-Z

This three-state operation can be useful when multiple buffers are connected to a common output line. In such cases, we can use $\overline{OE}$ to select which input signal is routed to the output line while keeping the remaining buffers in high-impedance mode.

OR, AND, NOR, and NAND

An *OR gate* is a circuit with two or more inputs that outputs "1" if any of the input signals is high. The behavior is summarized in Table 18-5.

Table 18-5: Truth Table for a Two-Input OR Gate

Input A	Input B	Output
0	0	0
0	1	1
1	0	1
1	1	1

Although the operation of the gate is simple, its construction is somewhat involved. A six-transistor CMOS implementation is shown in Figure 18-5.

We can start with the first section of the schematic (top left). It consists of two p-type MOSFET switches in series that conduct only if both inputs are low. Some readers might notice that the bottom MOSFET in this section looks unusual: It has four terminals instead of three. As discussed in Chapter 2 and Chapter 14, discrete transistors have the substrate internally tied to the source terminal, allowing conduction to be controlled by applying a specific gate-to-source voltage (V_{GS}). If the source-to-substrate connection isn't made on a semiconductor die, conduction is controlled primarily by the voltage between the gate terminal and the substrate, which is allowed to differ from the voltages on the source and drain legs. In this circuit, the design improves the switching performance of the bottom transistor by making it less sensitive to the voltage drop that may develop across the transistor directly above it. It's a minor implementation detail—on a breadboard, normal three-terminal transistors will work fine.

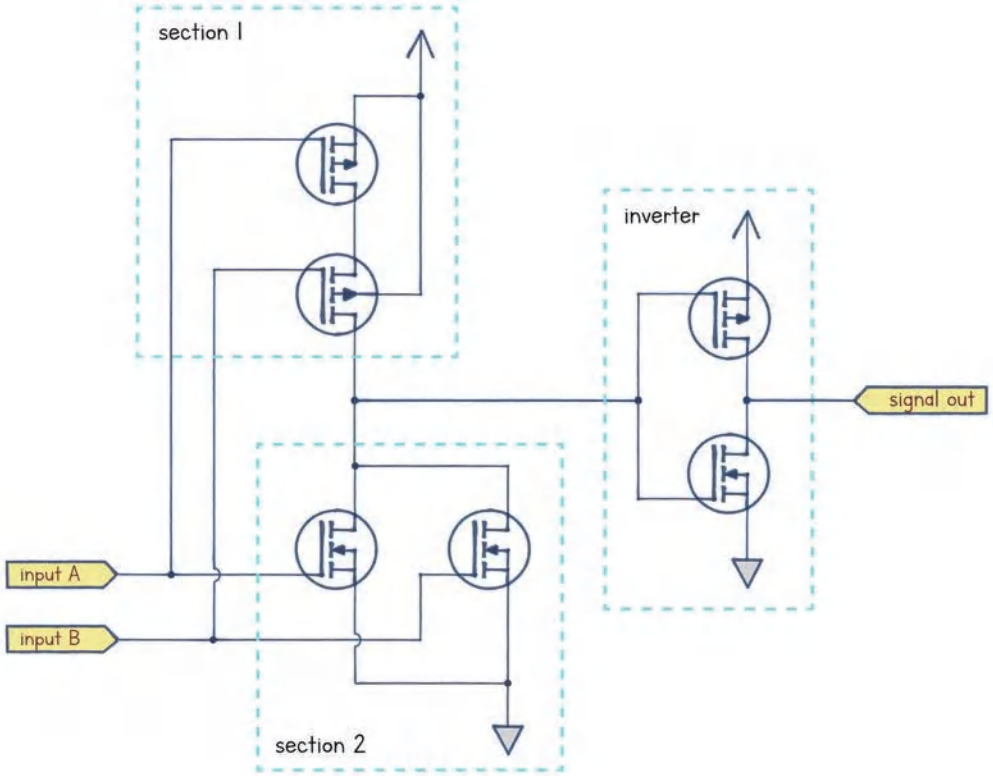


Figure 18-5: Schematic of a CMOS OR gate

With that out of the way, let's turn to the second section at the bottom of Figure 18-5. This consists of a pair of n-channel switches in parallel, without any additional complications. The stage conducts to ground if either of the input voltages is high.

The two stages are tied together, and their combined behavior is represented in Table 18-6.

Table 18-6: Truth Table for the Two Sections on the Left of Figure 18-5

Input A	Input B	Output	Comment
0	0	1	Section 1 on, section 2 off
0	1	0	Section 1 off, section 2 on
1	0	0	Section 1 off, section 2 on
1	1	0	Section 1 off, section 2 on

The behavior of this four-transistor arrangement is the inverse of what we'd expect from an actual OR gate—it outputs "0" if both inputs are zero and "1" otherwise. This inverted gate has its own name: *NOR*. To get the desired OR behavior, we need to add an inverter at the output, as shown on the earlier schematic on the right.

An *AND gate* is a straightforward complement of OR: It outputs "0" unless both inputs are high at the same time. Its behavior is summarized in Table 18-7.

Table 18-7: Truth Table for a Two-Input AND Gate

Input A	Input B	Output
0	0	0
0	1	0
1	0	0
1	1	1

The typical six-transistor implementation of an AND gate is shown in Figure 18-6. It is a simple modification of the OR topology; in this case, the p-channel switches in the first section are in parallel, while the n-channel switches in the second section are in series.

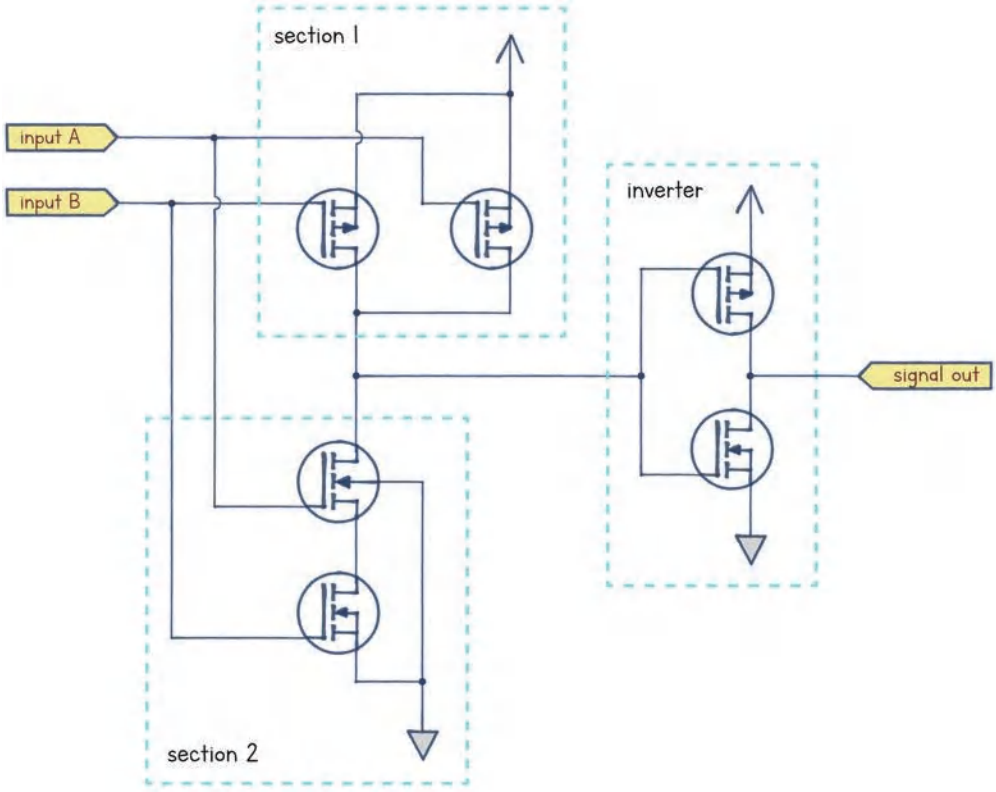


Figure 18-6: CMOS AND gate

An analysis of the two sections on the left nets us the truth table in Table 18-8.

Table 18-8: Truth Table for the Two Sections on the Left of Figure 18-6

Input A	Input B	Output	Comment
0	0	1	Section 1 on, section 2 off
0	1	1	Section 1 on, section 2 off
1	0	1	Section 1 on, section 2 off
1	1	0	Section 1 off, section 2 on

Once again, this is the inverse of what we want, a gate known as *NAND*. To get the variant we're after, we need to add a two-transistor inverter on the output side.

The typical symbols for these gates are shown in Figure 18-7.

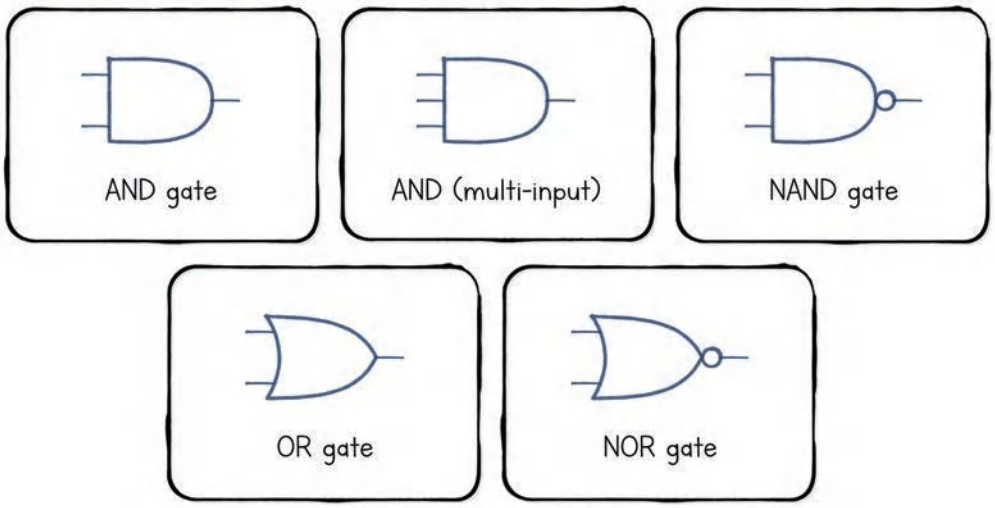


Figure 18-7: Schematic symbols for AND, NAND, OR, and NOR gates

The small circle on the output side of NAND and NOR indicates negation. You can sometimes see the symbol in other contexts—for example, on the input leg of a gate or a more complex digital chip. It simply means that the logic value of the signal is inverted before it enters or leaves the component.

A common 14-pin IC containing four AND gates is the 74HC08; a similar quad-OR package is available as the 74HC32. The NAND and NOR equivalents are the 74HC00 and 74HC02, respectively. If you have any of these chips in your workshop, they can be fun to experiment with. For example, you can build a logic puzzle that causes an output LED to light up only if the correct binary value is entered.

Exclusive OR

The last of the commonly encountered gates is the *exclusive OR (XOR)*. It works similarly to OR, except it allows only one of the input signals to be high. If both inputs are "1," the output is low, as shown in Table 18-9.

Table 18-9: Truth Table for a Two-Input XOR Gate

Input A	Input B	Output
0	0	0
0	1	1
1	0	1
1	1	0

As a warm-up exercise before diving into more advanced digital logic, we can construct the XOR gate from the gates we already know how to build: AND, NAND, and OR (Figure 18-8).

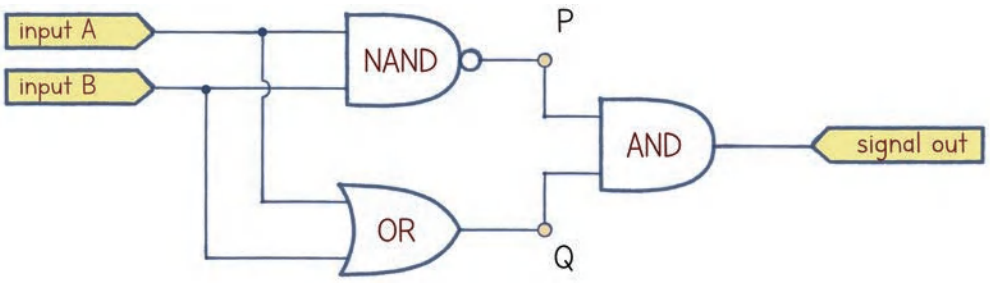


Figure 18-8: An implementation of XOR using other gates

In this design, the bottom OR gate outputs "1" if any of the input signals is high. This is the signal Q that appears at the bottom terminal of the output-side AND gate. That gate can be thought of as a mechanism that passes through signal Q if and only if P is high; if you're unsure why, take another look at Table 18-7 and analyze it from this perspective.

This brings us to P: The signal is the product of a NAND gate that outputs "1" at all times except when both inputs are high (Table 18-8). This means the AND gate will pass through the output of OR except in that final case. There we have it: a working XOR.

The standard symbols for the XOR gate and its inverted equivalent (XNOR) are shown in Figure 18-9. A common quad-XOR package is the 74HC86.

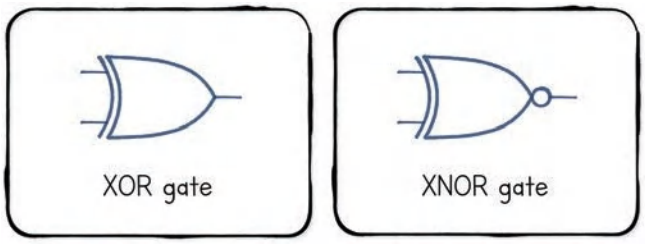


Figure 18-9: Symbols for XOR and XNOR gates

Universal Gates

Earlier in the chapter, I noted that in the CMOS architecture, NAND and NOR gates are easier to construct than their noninverting counterparts. They require just four transistors each because no inverter is needed on the output side.

These two inverting gates have an interesting property: Each one can be used to construct every other type of logic gate. The approach using NAND is shown in Figure 18-10. I encourage you to pause and analyze these configurations with pen and paper—it's a worthwhile exercise.

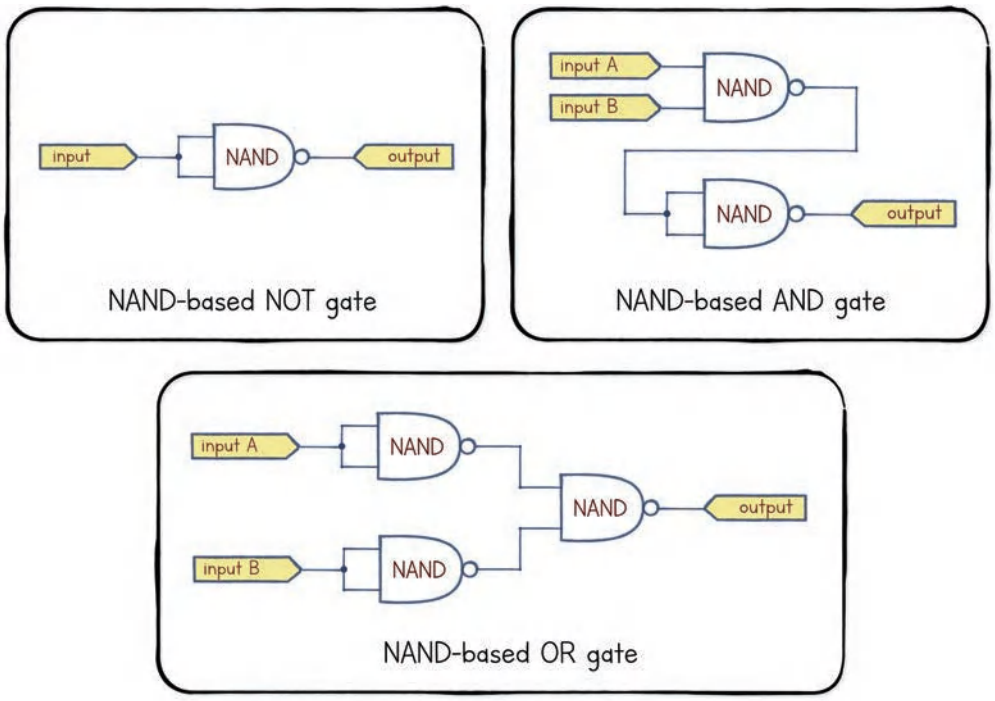


Figure 18-10: NOT, AND, and OR implemented using only NAND gates

The first panel of Figure 18-10 should be easy to decipher: NAND outputs "1" when the inputs are low and "0" when they're high, so if we present the same signal on both input terminals, we get a standard NOT gate.

The second circuit in the same figure is a standard NAND with its output chained to a NAND-based inverter. We can write it using the following notation:

$$f(a, b) = \text{NOT}(a \text{ NAND } b)$$

Since *a NAND b* is by definition the same as *NOT(a AND b)*, we can rewrite the formula as:

$$f(a, b) = \text{NOT}(\text{NOT}(a \text{ AND } b))$$

As before, double negation cancels out, so the output is the same as *a AND b*. Logic equivalencies that follow this pattern are known as *De Morgan's laws*.

The final circuit in the bottom panel of Figure 18-10 is a bit trickier: It's a NAND gate with both inputs inverted. The most expedient way to analyze it is to take the earlier NAND truth table and flip the input values to mimic the NAND-based NOT gates seen in the schematic. This yields Table 18-10.

Table 18-10: Inverting the Inputs of a NAND

Input A	Input B	Output
~~0~~ 1	~~0~~ 1	1
~~0~~ 1	~~1~~ 0	1
~~1~~ 0	~~0~~ 1	1
~~1~~ 0	~~1~~ 0	0

If we re-sort the rows so that 0-0 is on top, we'll see that the truth table is identical to that for an OR gate.

The gates we've discussed in this chapter may seem simple, but they allow for endless creativity in digital circuit design. In the next chapter, we'll begin to unravel the mysteries of multi-gate computing devices.

19

ADDERS AND BINARY MATH



In Chapter 18, I mentioned that logic gates can perform computations. This might seem like a big leap: The circuits outlined in that chapter can detect when two input signals are high simultaneously (AND) or equal to each other (XNOR), but this feels far from the flexibility needed to run a business spreadsheet or play video games.

In this chapter, we'll show that there's more to these circuits than meets the eye. Many types of reasoning, including the foundations of mathematics, can be expressed in the basic language of propositional logic, which uses a handful of simple operators and two possible truth values (true and false). In particular, arithmetic calculations of great complexity can be accomplished with nothing more than two voltage levels and a couple of logic gates. Let's see how we can pull this off.

An Introduction to Binary Arithmetic

To understand how logic gates can perform calculations, we first need to grasp the rules of binary arithmetic. As we've discussed, binary uses just two numbers—0 and 1—but in all other respects it can be used similarly to the decimal numeral system we're accustomed to. The difference is simply that when counting in decimal numbers, we wrap around to zero and carry one to the next column after we reach nine, whereas in binary the same thing happens once we count past one.

The first natural number expressed in binary is the same as in the decimal system: 0. After that comes 1; for the next number, we run out of available digits, so we need to start over and carry. The next value is therefore 10_2 (decimal 3), then 11_2 (4), then 100_2 (5), 101_2 (6), and so on.

NOTE *To avoid confusion between decimal and binary, we'll use the subscript suffix "$_2$" to denote binary numbers. Another common notation you will encounter in computing is a prefix of "0b."*

Both of these systems are positional, so we can assign a multiplier to every digit position. In decimal, the multipliers are the powers of 10; for example, in the number 543, we have five hundreds, four tens, and three ones:

$$543 = 5 \cdot 10^2 + 4 \cdot 10^1 + 3 \cdot 10^0$$

Binary works the same way, except every position corresponds to a successive power of two. In the number 1101_2, we have an eight, a four, no twos, and a one:

$$1101_2 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 13$$

If you spend enough time tinkering with digital circuits or computer software, you're bound to memorize the progression of at least some of the first powers of two (Table 19-1).

Table 19-1: A Collection of Memorable Powers of Two

Expression	Value
2^0	1
2^1	2
2^2	4
2^3	8
2^4	16
2^5	32
2^6	64
2^7	128
2^8	256
2^9	512
2^{10}	1,024
2^{11}	2,048
2^{12}	4,096
2^{13}	8,192
2^{14}	16,384
2^{15}	32,768
2^{16}	65,536

The sequence is useful for quickly converting between binary and decimal. If you know one of the values, you also know its immediate neighbors: You just need to divide or multiply the number by two.

Every pen-and-paper technique for adding, subtracting, multiplying, and dividing decimal numbers works for binary as well. For example, to add 1001_2 (decimal 9) and 0011_2 (decimal 3), we can sum the digits column by column, starting on the right (Figure 19-1).

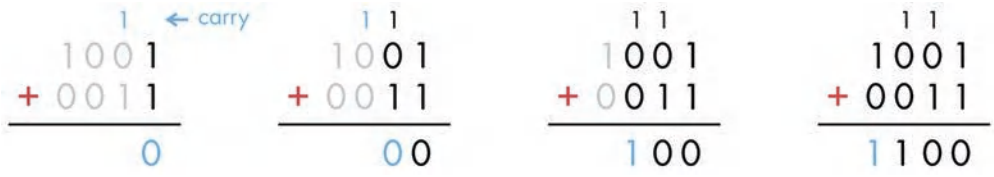


Figure 19-1: Addition in binary

The rightmost column is $1 + 1 = 10_2$, so we write "0" at the bottom and carry 1 to the next column. In the second column, we have 1 + 0, but we also need to include the carried value, so the result is $1 + 0 + 1 = 10_2$. We once again write 0 and carry. Next, we sum 0 + 0 and the carried value, which gives us 1 with no carry. The final (leftmost) column is 1 + 0 = 1. When all is said and done, we end up with 1100_2, or 12 decimal.

In computing, a single binary digit is called a *bit*; bits are usually grouped into larger, fixed-width units that can hold a more useful range of values. The smallest commonly used unit is 1 *byte*; it consists of 8 bits and it's capable of expressing natural numbers from 0 to 255. The next common choice is a *word*—16 bits, counting from 0 to 65,535. After that, we have a *double word* (32 bits, 0 to 4,294,967,295) and a *quadruple word* (64 bits, 0 to 18,446,744,073,709,551,615).

Building an Adding Circuit

Now that we've outlined the principles of binary arithmetic and some of the associated nomenclature, we can try to replicate the rules of pen-and-paper addition with logic gates. As we'll see, the task is easier than it might appear.

Half Adder

In the elementary school addition method outlined earlier, there are just four possible scenarios at the beginning of the process: We can only be adding 0 + 0, 0 + 1, 1 + 0, or 1 + 1. The outcomes are summarized in Table 19-2.

Table 19-2: Addition Table for Two Binary Digits

First operand (A)	Second operand (B)	Output digit (S)	Carry (C)
0	0	0	No
0	1	1	No
1	0	1	No
1	1	0	Yes

That's a truth table if we've ever seen one! Further, the patterns in the output columns should look familiar: The output digit follows the logic of the XOR gate, while the carry flag follows the pattern of the AND gate. We can express the rules as:

$$S = A \text{ XOR } B$$

$$C = A \text{ AND } B$$

The equivalent digital circuit is shown in Figure 19-2.

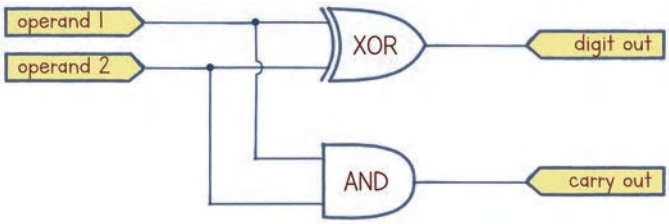


Figure 19-2: A two-digit adder implemented using logic gates

That said, this solution works only for the initial, rightmost column in pen-and-paper addition; for subsequent columns, we also need to accommodate a third parameter: input carry. The arrangement we've constructed is known as the *half adder*; we now need to extend it to build the more complete addition circuitry called a *full adder*.

Full Adder

To build a full adder, we start by constructing a new truth table with an additional input for the carry signal. The first part of the table—when there is no input carry (C_{in})—is identical to the behavior of the half adder, as captured in Table 19-3.

Table 19-3: Full Adder Behavior in the Absence of an Input Carry

A	B	C_{in}	S	C
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1

For completeness, we also need to consider the outcomes when the input carry flag is set; this nets us the pattern shown in Table 19-4.

Table 19-4: Full Adder Output with Input Carry

A	B	C_{in}	S	C
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

We can start constructing the circuit by examining the columns representing the output digit (S) in Tables 19-3 and 19-4. In the presence of input carry, the value is inverted from the usual XOR logic: A sequence of 0-1-1-0 becomes 1-0-0-1. A simple way to express it is the following equation:

$$S = (A \text{ XOR } B) \text{ XOR } C_{in}$$

To understand the formula, the second XOR function can be thought of as a construct that passes through the value of the first operand if the second operand is false (because 0 *XOR* 0 = 0 and 1 *XOR* 0 = 1) or inverts it if the second operand is true (0 *XOR* 1 = 1, 1 *XOR* 1 = 0).

This leaves us with the output carry signal. In the absence of input carry, the value is described by *A AND B*; in the presence of the carry flag, it evidently morphs into *A OR B*. We can write this as:

$$C = (A \text{ AND } B) \text{ OR } \left(C_{in} \text{ AND}(A \text{ XOR } B)\right)$$

The first AND expression produces the usual 0-0-0-1 output pattern. The nested XOR on the right produces 0-1-1-0, but the subsequent AND turns this part of the expression into 0-0-0-0 unless the input carry flag is set. The two patterns are then ORed together, producing 0-0-0-1 if input carry is false or 0-1-1-1 if it's true.

We could build the equivalent digital circuit by combining six individual gates as outlined earlier, though a more streamlined construction is to use two cascaded half adders, as shown in Figure 19-3.

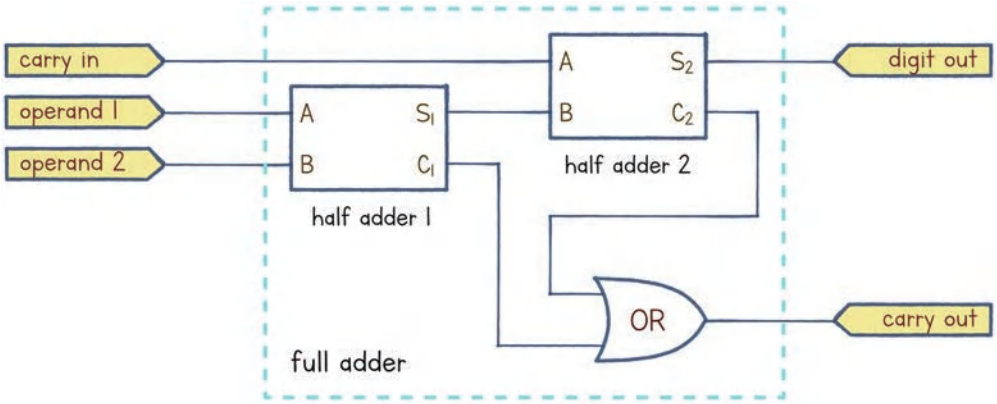


Figure 19-3: A full adder implemented with two half adders and an OR gate

To analyze this layout, recall that the output digit produced by each half adder is just the result of a single XOR operation. The first adder sums the two input values, so the output digit is:

$$S_1 = A \text{ XOR } B$$

The second adder sums this value with the input carry, effectively tacking on another XOR and telling us that the output digit produced by the circuit aligns with the formula we saw earlier:

$$S = S_1 \text{ XOR } C_{in} = (A \text{ XOR } B) \text{ XOR } C_{in}$$

This still leaves us with two disparate output carry signals. The first is the carry from adding the two input operands; for a half adder, we know the result is:

$$C_1 = a \text{ AND } b$$

The second carry signal is the carry after adding the input carry flag and the output digit from the first half adder:

$$C_2 = C_{in} \text{ AND } S_1 = C_{in} \text{ AND}(A \text{ XOR } B)$$

These two carry signals are then combined by passing them through an additional OR gate, so the circuit's output carry flag is:

$$C = C_1 \text{ OR } C_2$$

Substituting the previously calculated expressions for C_1 and C_2, we get:

$$C = \underbrace{(A \text{ AND } B)}_{C_1} \text{ OR } \underbrace{(C_{in} \text{ AND} (A \text{ XOR } B))}_{C_2}$$

This is the same formula for C that we developed earlier from analysis of the truth tables for binary addition with input carry.

Now that we have a working full adder, constructing a circuit that adds two binary numbers of any bit width is as simple as cascading the adders, as shown in Figure 19-4.

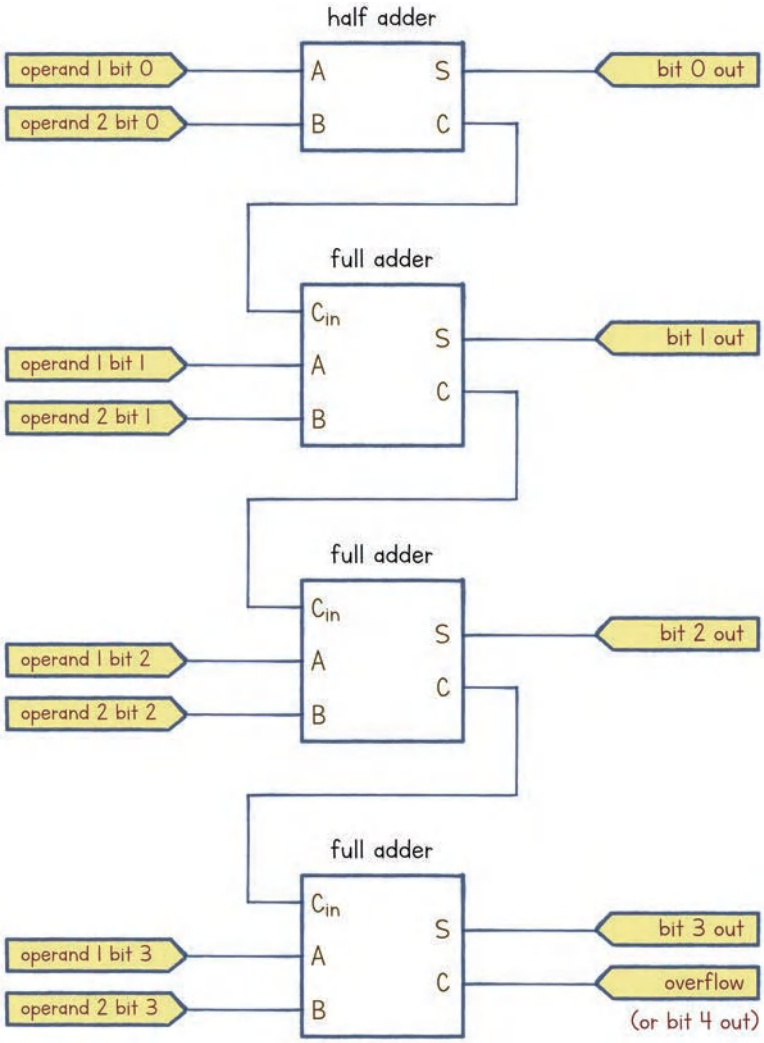


Figure 19-4: A 4-bit adder circuit

This circuit can add two binary numbers ranging from 0000_2 to 1111_2 (0 to 15 decimal). If the result is too large to fit into 4 bits—that is, if there is a carry to the nonexistent fifth column—the carry signal from the bottom adder can be used to trigger an overflow signal that can be interpreted by other components as an error condition.

NOTE *By way of nomenclature, in computing the rightmost digit of a binary number is typically called* bit 0 *because it corresponds to the underlying positional multiplier of* 2^0 *(1). The digit to the left is* bit 1*, and so on. The rightmost value is also sometimes called the* least significant bit (LSB) *because it affects the magnitude of the resulting number to the least extent. Conversely, the leftmost digit is known as the* most significant bit (MSB).

Toward Multiplication and Division

The most straightforward way to implement binary multiplication and division is to follow the pen-and-paper "long multiplication" method you might have been taught in school. Figure 19-5 illustrates the principle for 110_2 (decimal 6) multiplied by 1011_2 (decimal 11).

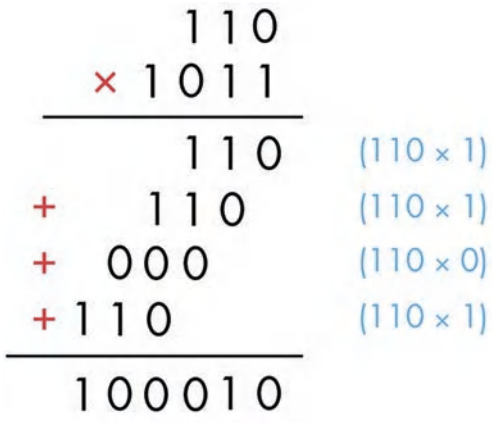


Figure 19-5: Simple binary multiplication

To find the result, we take the rightmost digit of the second operand, multiply the first operand by it, and then write down the result below the first horizontal line. In binary, single-digit multiplication is trivial—the result can only be $110_2 \cdot 0 = 000_2$ or $110_2 \cdot 1 = 110_2$. In this instance, the multiplier is 1, so we write down "110".

With this done, we move to the next digit of the second operand and follow the same procedure, except that we shift the result one column to the right. We repeat the process until we run out of digits. With all the values written down, we then calculate their sum (second horizontal line). The result is 1000010_2 (66).

A circuit needed to implement the algorithm in a single step would be fairly complicated because of the sheer number of adders we'd need to stack. The complexity increases even further if we consider tasks such as exponentiation, square roots, or logarithms. To make digital computation scalable, it's often better to break problems into a sequence of steps that can be executed one after another, with the appropriate subcircuit selected according to a prescribed plan.

To reach that point, we first must address a more basic need: a way to store intermediate results in some form of working memory—and that's where we turn next.

20

MEMORY



In the previous chapter, I noted that to realize their full potential, digital circuits must be able to break down complex tasks into a series of steps. This requires some effort because logic gates are deterministic and stateless: Given exactly the same inputs, they always produce the same result. To do anything other than endlessly repeat a single operation, a computing device needs a way to keep track of its progress and store intermediate results.

To that end, in this chapter we'll explore the techniques that let us move beyond one-shot computation and enable machines to remember what happened before.

Latches

The basic gate-based data storage mechanism is a *latch*. A latch is a two-state (bistable) circuit that can be toggled on or off. Like many mechanical switches, the latch stays in position—that is, it keeps producing the same output voltage—until a specific input signal toggles its state again.

S-R Latch

The design of a *set-reset (S-R) latch* is shown in Figure 20-1. The most important aspect of this circuit is the connection at the top that loops the circuit's output signal back to one of the input legs of an OR gate.

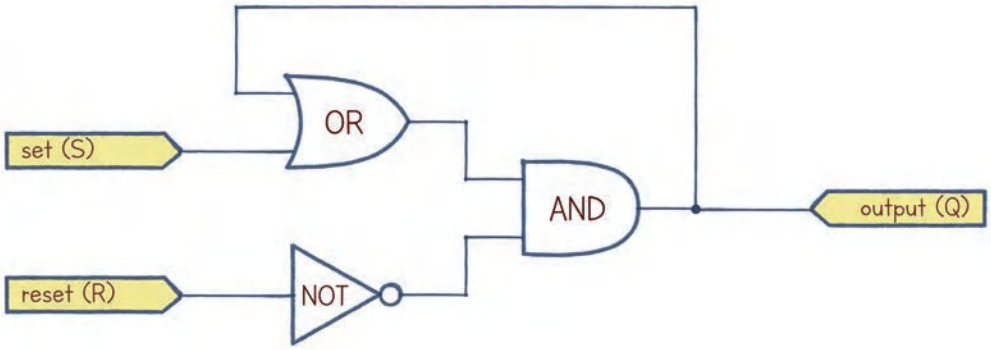


Figure 20-1: An intuitive design for a three-gate S-R latch

To analyze the circuit, let's assume that the "set" signal (S) is high and the "reset" signal (R) is low. In this case, the output of the OR gate is a logical one, regardless of the looped-back signal present on the gate's other terminal. In this configuration, one input of the AND gate is always high, as is the other, because it's an inverted copy of the signal present on R. All in all, when S = 1 and R = 0, both inputs of the AND gate are high, and therefore so is the output of the entire S-R latch (Q).

Next, let's imagine that S transitions to a logical zero. This puts one of the OR inputs at zero volts, but the other remains high because it's the looped-back output signal Q. In this condition, the circuit still outputs the same voltage as before.

The only thing that can break the cycle is pulling the "reset" line high. This causes one of the AND inputs to go low, forcing the output signal to zero and breaking the loop that kept the OR gate latched. From that point on, the output remains low even if R returns to zero volts.

We can construct the truth table shown in Table 20-1 to document the behavior of the latch.

Table 20-1: The Behavior of the Three-Gate S-R Latch

S	R	Previous Q	Current Q
0	0	0	0 (hold)
0	0	1	1 (hold)
0	1	*any value*	0
1	0	*any value*	1
1	1	*any value*	*invalid*

Simultaneous S = 1 and R = 1 inputs are not valid and should be avoided. For some latch architectures, they may produce erratic outputs. In the topology outlined here, the input reliably resets the gate, but a subsequent transition to S = 0 and R = 0 may cause an unexpected bit flip if the falling edge of the "set" signal arrives a split nanosecond later than the falling edge on the "reset" line.

The design is elegant, but a simpler version of the circuit can be built with just two NOR or NAND gates. A NOR variant is shown in Figure 20-2.

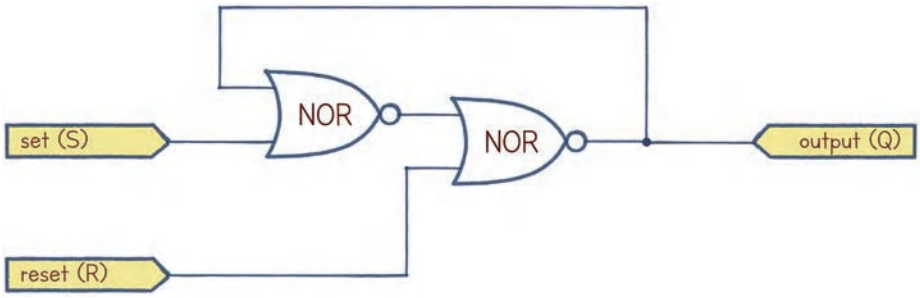


Figure 20-2: An optimized S-R latch architecture with two NOR gates. An equivalent circuit with inverted inputs can also be constructed with a pair of NAND gates.

Once again, let's start with a scenario where S = 1 and R = 0. A NOR gate outputs a logical one only if both inputs are low, so in this case the gate on the left is forced to produce zero regardless of the previous output signal. The gate on the right effectively calculates 0 *NOR* 0, so the output line is pulled high.

Next, if S is pulled low, nothing materially changes: The gate on the left now receives the looped-back logical one, so its output remains the same as before—that is, the circuit is latched. To undo this, R must be pulled high, which produces a 0 *NOR* 1 condition for the gate on the right, resets the output voltage, and breaks the loop. The behavior of the simplified circuit is summarized in Table 20-2.

Table 20-2: The Behavior of the Two-Gate S-R Latch

S	R	Previous Q	Current Q
0	0	0	0 (hold)
0	0	1	1 (hold)
0	1	*any value*	0
1	0	*any value*	1
1	1	*any value*	*invalid*

Gated S-R Latch

The basic S-R latch is interesting but somewhat awkward to use. Instead of storing a single bit of information when commanded to do so, it requires two separate levers that must be pulled depending on the value to be recorded.

To refine the design, we can add a gating mechanism—an "enable" signal (E) that controls when the S and R lines are active. A simple implementation appears in Figure 20-3.

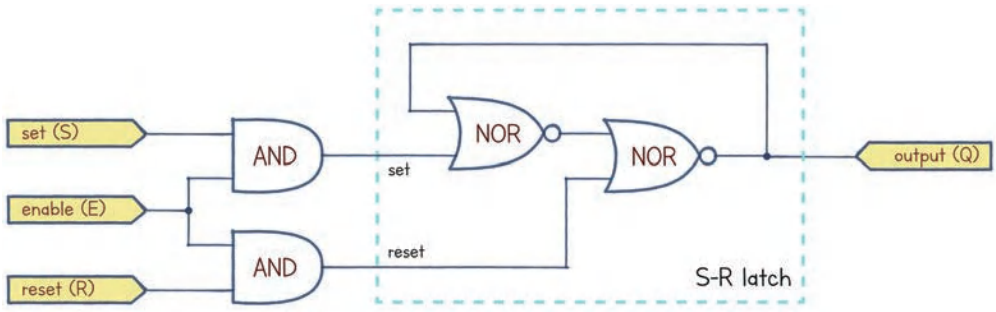


Figure 20-3: The S-R latch gated by an "enable" signal

The two added AND gates pass the S and R signals through only if the "enable" line is high. In all other cases, the outputs of the gates are zero, so the latch can't be toggled and continues to return the previously stored value. The gating mechanism doesn't make the circuit easier to use, but it serves as a stepping stone toward another improvement.

D Latch

The modified S-R latch circuit has an interesting property: Because the "enable" line can disconnect the S and R signals, we can replace those two inputs with a single data line, D, which determines what value is written to the latch whenever E is high (Figure 20-4).

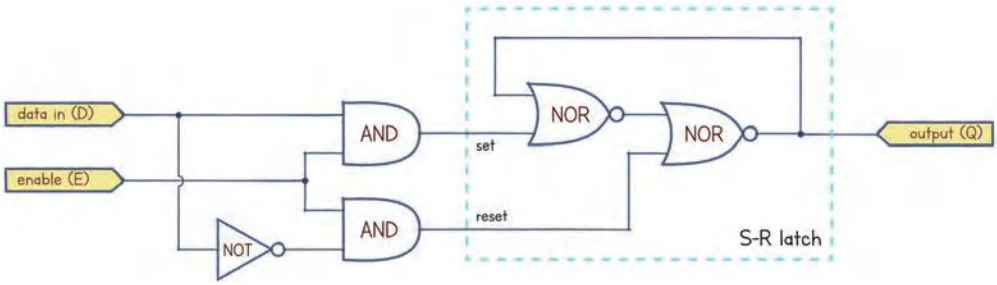


Figure 20-4: The architecture of a D latch

If E = 0, the underlying S-R latch receives an input of S = R = 0. Otherwise, if the "enable" line is high, an input signal of D = 0 is translated to S = 0 and R = 1, while D = 1 is mapped to S = 1 and R = 0.

This combined architecture is known as the *D latch*. The behavior of the circuit is summarized in Table 20-3.

Table 20-3: The States of a D Latch

D	E	Previous Q	Current Q
any value	0	0	0 (hold)
any value	0	1	1 (hold)
0	1	*any value*	0
1	1	*any value*	1

The benefit of the D latch is that we don't need to concern ourselves with whether we're storing "0" or "1"—the procedure is always the same: Supply an input signal on the D line and then strobe the "enable" pin to save the data for later. Another minor perk is that we eliminate the invalid S = 1 and R = 1 input state present in the earlier design.

An example of a chip containing four D latches is the 74HC75. A simple experiment to perform with such a chip is to build an adder that sums two input values but allows them to be entered using a single button, with another switch to select the active input bit.

Flip-Flop Circuits

One remaining weakness of the D latch is that as long as the "enable" line is high, the circuit passes through the input signal in real time. This creates a problem if we want

to use a memory cell to store the result of a computation and then feed the stored value back into the circuit that performed the computation, inadvertently changing the result.

This is fairly common—for example, if we have an adder circuit that we're using to implement long multiplication (Chapter 19), we want to perform successive additions on the same intermediate value. But if the input of an adder is connected to the output of a latch-based memory cell, any attempt to store the result in the latch will cause the new value to instantly propagate to the input of the adder and thus alter the calculation we're trying to preserve.

D Flip-Flop

To solve the issue, we need an architecture known as a *flip-flop*. The nomenclature can be a bit confusing because a latch does its share of flipping and flopping too. The distinction lies elsewhere: Flip-flop circuits use a clocked architecture, as shown in Figure 20-5.

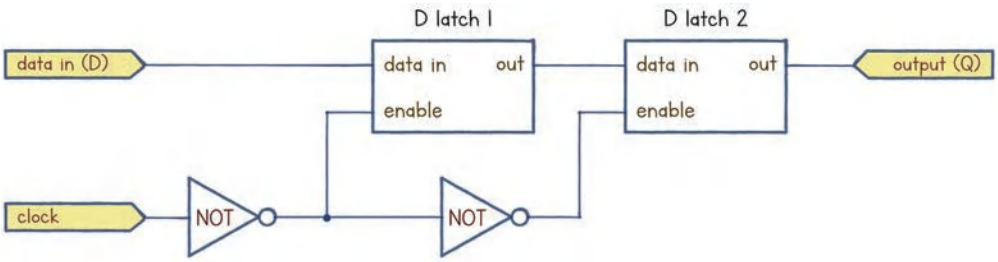


Figure 20-5: A clocked D flip-flop constructed from two D latches

The clock signal is just a binary square wave that may be generated by an R-C relaxation oscillator, sourced from a signal generator, or produced by manually flipping a switch if you're so inclined.

The heart of this flip-flop circuit is two D latches connected in series and configured so that they are alternately enabled by the clock signal and never active at the same time. First, the input-side latch is activated when the clock signal is low; this allows the input data to be stored but not passed through to the output, because the second latch remains inactive.

Sometime later, the clock signal shifts to a logical one. At that point, the first latch is disconnected from the "data in" line and the second latch becomes active, allowing the stored value to propagate to the output pin (Table 20-4).

Table 20-4: The Behavior of a D Flip-Flop

D	Clock signal	Previous Q	Current Q
0	⎍ (0 → 1)	*any value*	0
1	⎍ (0 → 1)	*any value*	1
any value	0, 1, ⎏ (1 → 0)	0	0 (hold)
any value	0, 1, ⎏ (1 → 0)	1	1 (hold)

A common and experimentation-worthy chip containing a pair of D flip-flops is the 74HC74.

The architecture of the D flip-flop is commonly described as *edge-triggered* because the input data appears to be loaded just as the clock transitions from low to high.

However, the nomenclature undersells the underlying principle: The key point is that the update of the stored data happens in two separate steps. There is never a continuous path between the input and the output, so there is no risk of the input data being distorted due to feed-through effects. This property is vital to the circuit we intend to build next.

T Flip-Flop

The *T flip-flop* is a simple extension of the D flip-flop. The arrangement, shown in Figure 20-6, doesn't directly store any input data. Instead, when the "toggle" signal is high, the circuit alternates between two logical values on every rising edge of the clock.

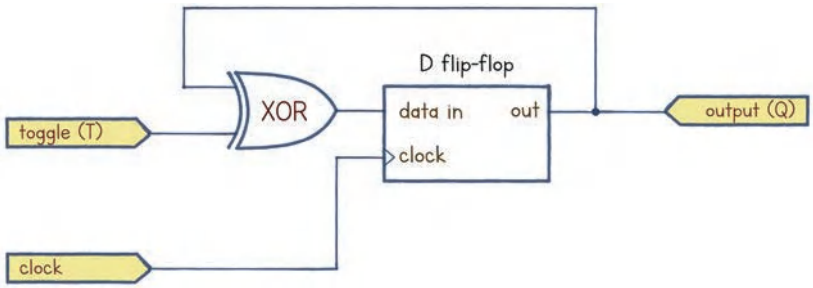


Figure 20-6: The usual architecture of a T flip-flop

The circuit can be further simplified by removing the "toggle" input and replacing the XOR gate with an inverter. In this variant, the toggling happens unconditionally with every clock tick.

Although the design might seem somewhat useless, it has two interesting properties that we'll build on in Chapter 21 and beyond. First, it can be thought of as a one-bit counter (0, 1, 0, 1, 0, 1 . . .). Second, it offers a way to halve the frequency of the input clock, because there's just one rising edge in every full cycle of the clock waveform, but two edges are needed to return the output of the T flip-flop to the starting point—that is, to complete a full cycle of the output waveform.

In addition to D and T flip-flops, you might also encounter J-K flip-flops. These flip-flops hold their state when both J and K inputs are zero; can be set if J = 1 or reset if K = 1; and can be toggled if both J and K are high. They are less useful than the fundamental circuits we examined earlier, but they find creative applications every now and then.

Beyond Logic Gates

The gate-based memory architectures introduced in this chapter are not the only way to store data in digital circuits. In the early days of computing, working memory was implemented in all kinds of unusual ways—from spinning drums full of capacitors to acoustic waves traveling through columns of mercury. Today, latch-style circuits are the preferred solution for fast, energy-efficient memories used on microprocessor dies. In this capacity, the technology is known as *static random-access memory (SRAM)* and is usually implemented using six transistors per cell (a slight simplification from the eight transistors in our designs).

Because of the relatively high number of transistors needed per bit of SRAM, other solutions are sometimes preferred for bulk data storage. In particular, dynamic RAM (DRAM)—used as the main system memory in personal computers and smartphones—is implemented using arrays of capacitors that can store charge for a brief while but

need constant refreshing. This trades space savings for added complexity and a fair amount of wasted energy.

If you're interested in a stroll down *memory* lane, I have a short essay on the history of RAM at *https://lcamtuf.coredump.cx/blog/ram*. Otherwise, our next stop is operation sequencing circuitry.

21

SEQUENTIAL OPERATION



In the last couple of chapters, we outlined several approaches that allow digital circuits to perform calculations and store data. However, to get the most out of these techniques, we're still missing one piece of the puzzle: a way to activate different subsystems and shuffle data between them according to a predefined plan.

To make this goal more tangible, recall the discussion of long multiplication toward the end of Chapter 19. The algorithm we discussed requires cycling through the binary digits of an operand and then summing bit-shifted results. These operations must be performed in a sequence with no repeating or missed steps. In other words, the multiplication circuit needs to know how to count from zero to an architecture-specific bit position *n*. We won't be building a full-fledged hardware multiplier, but we can use this chapter to lay out the conceptual groundwork.

Counters

Counters differ from most other logic circuits in that they usually don't have data inputs per se. Instead, once the counter is initialized, it generates a sequence of numerical outputs on its own, advancing on every clock tick.

Counters are typically constructed using D or T flip-flops (Chapter 20). The flip-flops are cascaded to progressively feed each other. Let's review three common counter architectures you might encounter; they are quite similar but differ in important ways.

Ring Counter

A *ring counter*—also known as a *one-hot counter*—is the simplest digital counting circuit. Its output isn't a sequence of binary numbers; instead, the counter produces a traveling bit pattern, as outlined in Table 21-1.

Table 21-1: The Behavior of a Ring Counter with Four Output Lines, Q_0 to Q_3

Clock edge #	Q_0	Q_1	Q_2	Q_3
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1

The corresponding circuit for a 4-bit ring counter is shown in Figure 21-1.

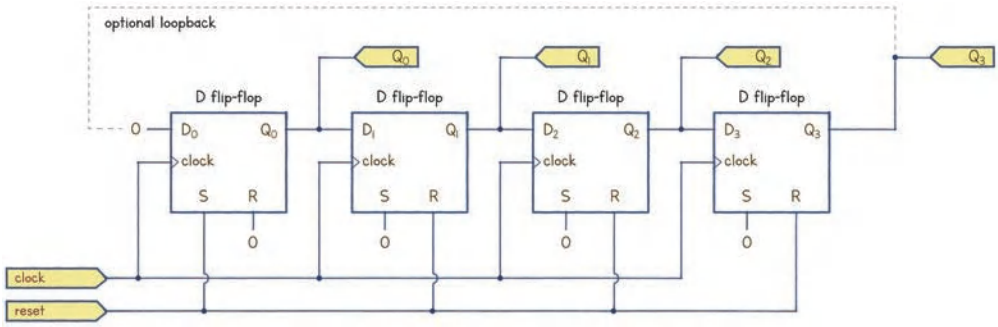


Figure 21-1: Four-output ring counter. Zeros represent nodes wired to ground (logical zero). The first input (D_0) can be held at zero or looped to Q_3.

Note that each of the D flip-flops in the schematic has an additional "set" (S) and "reset" (R) terminal. These terminals are asynchronous—that is, not dependent on the clock signal—and can be implemented with a simple modification to the pair of D latches that make up each flip-flop, as shown in Figure 21-2.

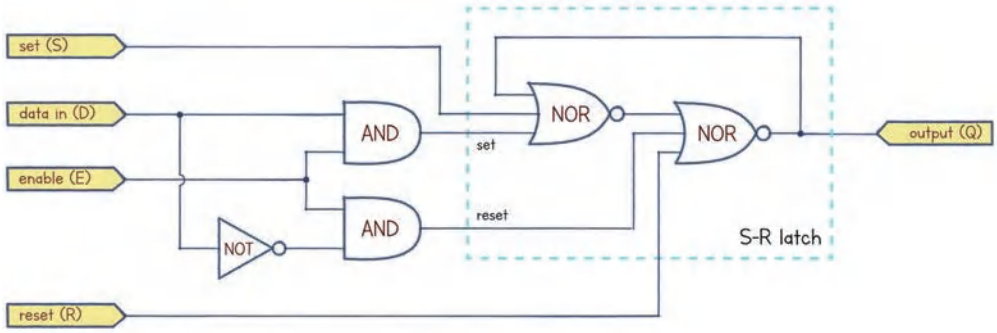


Figure 21-2: Adding asynchronous "set" and "reset" inputs to a D latch (refer back to Figure 20-4 in Chapter 20)

These signals are added so that when the circuit-wide "reset" line is pulled high, the leftmost flip-flop is initialized to a logical one while the remaining flip-flops are set to zero. In effect, outputs Q_0 through Q_3 are set to 1-0-0-0.

Remember that the propagation of data through a flip-flop happens in two stages: The input signal (D_n) is sampled while the clock is low and is propagated to the corresponding output (Q_n) only once the clock transitions to high. With this in mind, we can reason that on the next rising edge of the clock, the last value of $D_0 = 0$ will be propagated to Q_0, the earlier state of Q_0 will be moved to Q_1, and the earlier value of Q_2 will be shifted to Q_3. After these shifts, the circuit will output 0-1-0-0. The same shifting pattern will repeat on subsequent clock cycles, producing 0-0-1-0 and then 0-0-0-1.

What happens after reaching the end of the line depends on whether D_0 is permanently wired to a logical zero or to Q_3. In the first case, once the traveling bit is shifted out, the counter will cease to function, settling on a constant output of 0-0-0-0. In the second scenario, the counter will wrap around, and the sequence will repeat.

Ring counters are useful for activating circuits one after another, both in digital computation and beyond. For example, they can be used to cycle through the rows of LED matrix displays as part of the line-by-line screen refresh scheme outlined in Chapter 10. If a ring counter is operated at a low frequency and the outputs are connected to a series of LEDs, the circuit can also function as a retro "LED chaser" amusement device: When the diodes are triggered in sequence, the resulting pattern creates the illusion of a moving light.

Dedicated ring counter chips aren't available on the market, but the circuit can be constructed by wiring a D flip-flop chip (for example, the 74HC74) in a particular way. Another approach is to use a closely related IC: a serial-in, parallel-out shift register. We'll examine shift registers soon.

Ripple Counter

A *ripple counter* is a digital circuit that outputs a sequence of successive binary numbers—say, 0000_2, 0001_2, 0010_2, 0011_2, and so forth. Because this architecture uses a positional numeral system instead of a walking-bit pattern, it can count to higher values with far fewer parts. To count to 1,000 with a ring counter, we'd need a string of 1,000 flip-flops; with a ripple counter, 10 flip-flops would suffice.

Figure 21-3 shows an example of a 4-bit ripple counter implemented using rudimentary T flip-flops that change state on every edge of the supplied clock signal.

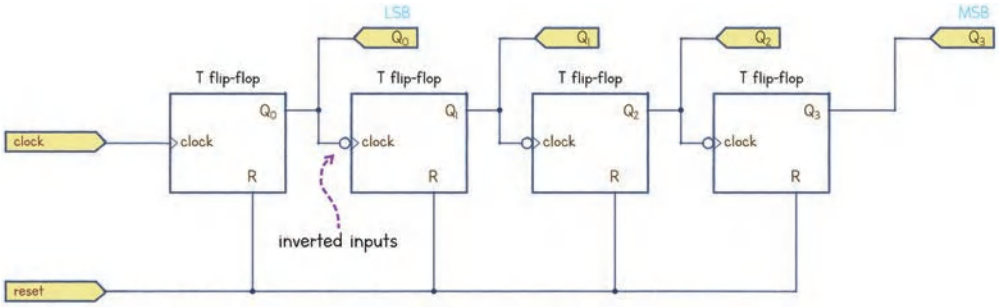


Figure 21-3: A 4-bit binary counter. The toggle (T) input is omitted but can be assumed to be permanently tied to a logical one.

The reset signal initializes the counter to 0-0-0-0. The external clock signal is connected only to the leftmost flip-flop, so on the first rising clock edge, Q_0 transitions to a logical one. This signal is then routed to the clock input of the next flip-flop. But note the circle symbol that signifies that the input is inverted (NOT); this means that the second flip-flop will toggle on the *falling* edge of Q_0. So far, the transition only

produced a rising edge, so the state of the second flip-flop doesn't change, and the output of the counter is 0-0-0-1 (if written in the usual order with the least significant bit on the right).

On the next edge of the external clock signal, the leftmost flip-flop toggles again and Q_0 transitions from one to zero. This produces a falling clock edge for the next flip-flop, causing its output (Q_1) to change to a logical one. The value is now 0-0-1-0. The next clock tick results in 0-0-1-1; on the cycle that follows, both Q_0 and Q_1 flip from one to zero, the third flip-flop is toggled, and the resulting count is 0-1-0-0. The entire pattern is outlined in Table 21-2.

Table 21-2: Output of a Ripple Counter

Clock edge #	Q_3 (MSB)	Q_2	Q_1	Q_0 (LSB)	Numerical value
0	0	0	0	0	$0000_2 = 0$
1	0	0	0	1	$0001_2 = 1$
2	0	0	1	0	$0010_2 = 2$
3	0	0	1	1	$0011_2 = 3$
4	0	1	0	0	$0100_2 = 4$
5	0	1	0	1	$0101_2 = 5$
6	0	1	1	0	$0110_2 = 6$
7	0	1	1	1	$0111_2 = 7$
8	1	0	0	0	$1000_2 = 8$
9	1	0	0	1	$1001_2 = 9$
10	1	0	1	0	$1010_2 = 10$
11	1	0	1	1	$1011_2 = 11$
12	1	1	0	0	$1100_2 = 12$
13	1	1	0	1	$1101_2 = 13$
14	1	1	1	0	$1110_2 = 14$
15	1	1	1	1	$1111_2 = 15$

After reaching 1111_2, all the digits wrap around to 0000_2 and the cycle repeats.

Binary counters have a wide variety of uses. In addition to keeping track of time, they can count any pulses supplied on the clock input and then store the tally in binary form.

Another interesting property of these circuits is that each output bit divides the frequency of the input clock by a power of two. Note that the Q_0 column in Table 21-2 completes one full cycle every two clock edges; the same cycle takes four edges for Q_1, eight edges for Q_2, and sixteen edges for Q_3. This property can be exploited in analog circuits too: A single counter IC can offer precise, selectable frequency division for electronic musical instruments, timers, and more.

Ripple counter ICs are readily available on the market and are fun to play with; good examples include the 74HC4020 (14-bit), 74HC4040 (12-bit), and 74HC393 (dual 4-bit). Note that all of those chips trigger on the falling edge of the clock signal. In their datasheets, this may be indicated by an overline for the clock signal—for example, $\overline{\text{CLK}}$ for falling-edge signals versus CLK for rising-edge ones. That said, this notation is somewhat ambiguous and isn't used consistently.

Synchronous Counter

A drawback of ripple counters is that they are relatively slow. Each flip-flop has a limited response speed, governed in part by internal gate capacitances; because the clock for every subsequent stage must be generated by the preceding flip-flop, the worst-case settling time for the circuit grows in proportion to the number of bits.

A solution to this problem is the *synchronous counter* architecture shown in Figure 21-4. The circuit uses toggle flip-flops with a common clock signal, along with strategically placed AND gates that generate a carry signal for each cell.

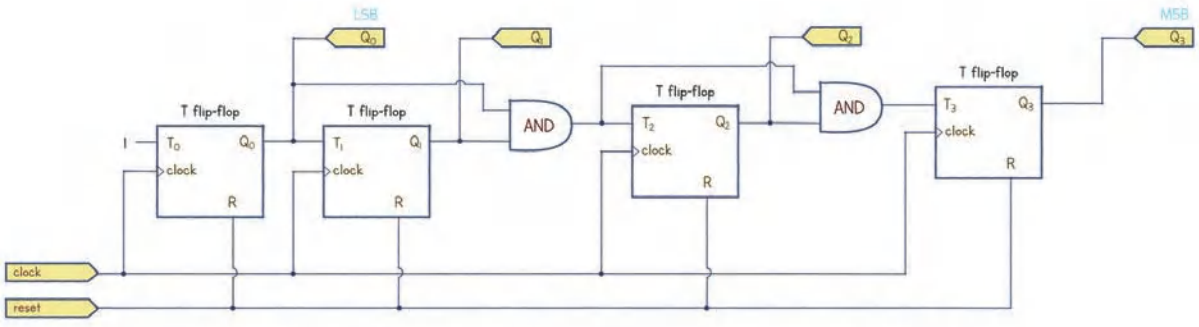


Figure 21-4: A 4-bit synchronous counter

In this design, every T flip-flop shares the same clock signal. The circuit is initialized so that Q_0, Q_1, Q_2, and Q_3 are at a logical zero—that is, the output is 0-0-0-0. To understand what happens next, we can focus on the toggle signals for every digit position. For the least significant digit (on the left), T_0 is always equal to 1, so the value toggles on every positive clock edge. For the next digit, T_1 is tied to Q_0, so the value advances only in the subsequent cycle after Q_0 reaches 1; together, the flip-flops produce a repeating pattern of 00_2, 01_2, 10_2, 11_2.

For the third digit, the toggle signal is effectively calculated as $T_2 = Q_0$ *AND* Q_1. This means that the digit flips only after the two preceding digits reach 11_2. The resulting three-digit output pattern is 000_2, 001_2, 010_2, 011_2, 100_2, 101_2, and so forth. Finally, the toggle signal for the most significant digit is $T_3 = T_2$ *AND* Q_2, which can be expanded as $T_3 = Q_0$ *AND* Q_1 *AND* Q_2; the digit flips only after the preceding digits reach 111_2, completing the pattern of a 4-bit binary counter. This design can be extended to any number of binary digits by adding additional stages wired so that $T_n = T_{n-1}$ *AND* Q_{n-1}.

Note that in each clock cycle, there is no need for the input signal to ripple through multiple flip-flops. This allows synchronous counters to operate at higher speeds and without propagation-related output artifacts.

A good example of a synchronous 4-bit counter IC that can be preset is the 74HC161.

Data Routing

Counters allow a digital circuit to keep track of sequential operations. However, if the set of steps to be followed is baked into the hardware architecture, the resulting circuit will be less versatile than it could be.

A better approach to general-purpose computing is to let the user supply an operating plan of their choice. The instructions—a computer program—could be entered by toggling switches on a front panel, inserting a punched card, or encoding a

sequence of bits on magnetic tape. Each encoded instruction could specify where to fetch data from, what operation to perform on it, and where to store the result.

The program is just a sequence of binary numbers, so to bring this concept to life, we could start by giving every subsystem of our circuit a numerical identifier. For example, if memory location *a* is assigned an identifier of 01_2, if memory location *b* is 10_2, and if the addition circuit is 0000_2, we could encode "add *a* to *b*" as 01-0000-10_2.

The part we're still missing is a way to take these numbers and use them to flexibly route signals to different components.

Line Decoder

A circuit that translates a binary number to a 1-out-of-*n* output signal is known as a *line decoder*. The design of a decoder that translates a three-digit input value into a signal on one of the eight output lines is shown in Figure 21-5.

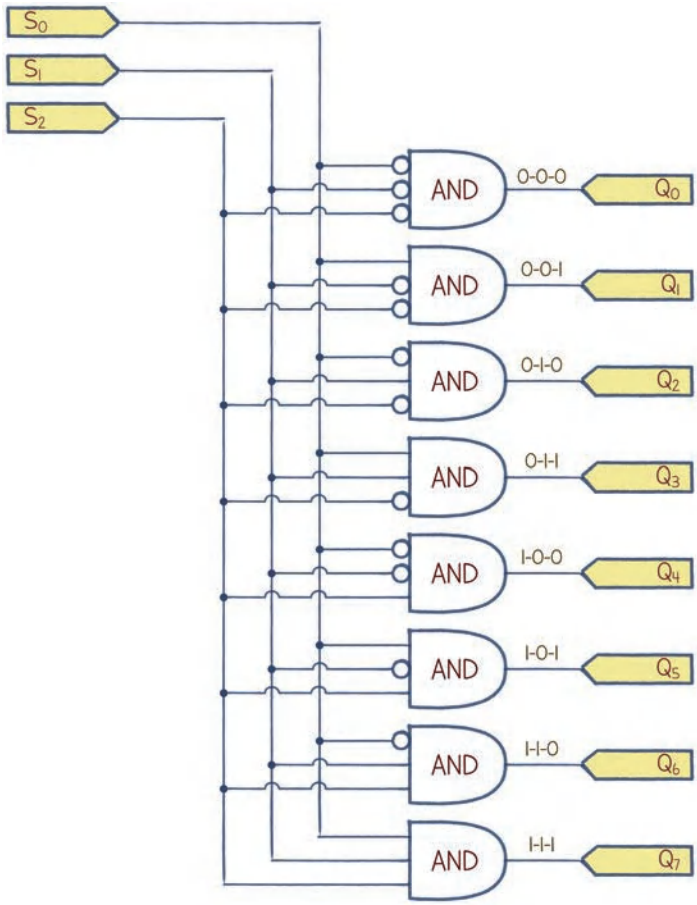


Figure 21-5: A 3-bit (3-to-8) line decoder

The operation of the circuit is simple. Note that each of the AND gates has a different arrangement of inverting and noninverting inputs, so only one gate is active for each possible input value. For example, line Q_0 becomes active only if the input is 0-0-0, while Q_7 is active only for 1-1-1. Table 21-3 is a complete truth table for the circuit.

Table 21-3: Truth Table for a Line Decoder

S_2	S_1	S_0	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Once again, a practical application of a line decoder is to take a binary number supplied by the user and then activate a specific arithmetic circuit. For example, if there's an adder connected to line #7, we would select it with an operation code (op-code) of 111_2.

Multiplexer

A *multiplexer* is a complementary circuit topology that allows a single output terminal to be connected to one of several selectable inputs. A simple multiplexer with four input lines is shown in Figure 21-6.

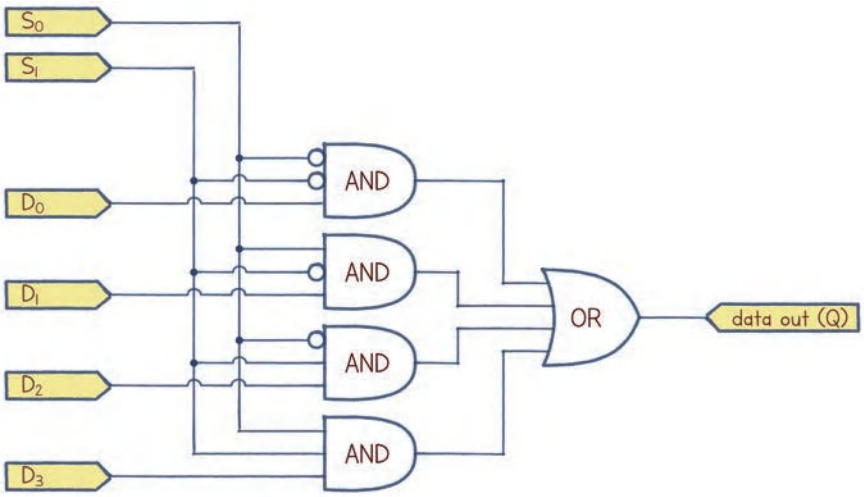


Figure 21-6: A four-input multiplexer circuit

Like the line decoder, the circuit uses a pattern of inverted and noninverted inputs for the bottom two terminals of each AND gate; this ensures that for every possible "select" value (S_0 and S_1), only one of the four gates will have both of these inputs at a logical one. If the "select" signal is 0-0, this will be the top gate; for 0-1, it will be the gate below it.

For all gates except the selected one, the value of the corresponding data input (D_n) doesn't matter: The output will always be zero. For the selected gate, however, the value of the data input is effectively passed through: *1 AND 1 AND a = a*. The OR gate on the right then passes through this signal to the output leg; the always-zero signals from the non-selected lines have no bearing on the result. Table 21-4 is a truth table for the circuit.

Table 21-4: Truth Table for a Multiplexer Circuit

S_1	S_0	D_0	D_1	D_2	D_3	Q
0	0	0	*any value*	*any value*	*any value*	0
		1	*any value*	*any value*	*any value*	1
0	1	*any value*	0	*any value*	*any value*	0
		any value	1	*any value*	*any value*	1
1	0	*any value*	*any value*	0	*any value*	0
		any value	*any value*	1	*any value*	1
1	1	*any value*	*any value*	*any value*	0	0
		any value	*any value*	*any value*	1	1

The circuit allows data from any number of sources to be selectively routed to the same location for processing.

In addition to computation, the approach has uses in communications: If connected to a counter, a multiplexer can take a multi-bit input value and transmit it over a single data line bit by bit. This is known as *serial transmission* and is common in situations where a parallel data bus would be impractical—for example, if we wish to avoid bulky cables with dozens of conductors and connector pins.

Demultiplexer

A *demultiplexer* is notionally the inverse of a multiplexer: It's a circuit that takes a single input and routes it to the selected output line, as shown in Figure 21-7.

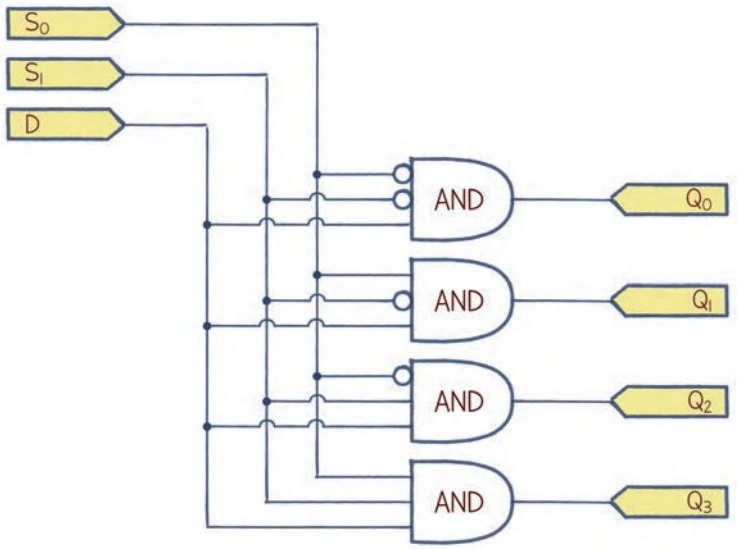


Figure 21-7: A basic four-line demultiplexer

The circuit is covered in most textbooks, but by itself it has limited utility. For example, it can't readily convert a serial transmission back to the original multi-bit binary number because it doesn't retain previous values on the currently deselected output pins. The problem is illustrated in Table 21-5.

Table 21-5: Truth Table for the Demultiplexer

S_1	S_0	D	Q_0	Q_1	Q_2	Q_3
0	0	0	0	0	0	0
		1	1	0	0	0
0	1	0	0	0	0	0
		1	0	1	0	0
1	0	0	0	0	0	0
		1	0	0	1	0
1	1	0	0	0	0	0
		1	0	0	0	1

Note that the inactive lines are always zero. In effect, unless equipped with output-side latches, the circuit can simply be thought of as a line decoder with an added "enable" pin: It selects one out of *n* lines if D is high and outputs all zeros when D is pulled low.

Shift Register

A serial-in, parallel-out *shift register* is a more practical circuit that does what a demultiplexer can't: It takes a stream of serially transmitted bits and reconstructs the underlying binary number piece by piece. The design of a 4-bit shift register is shown in Figure 21-8.

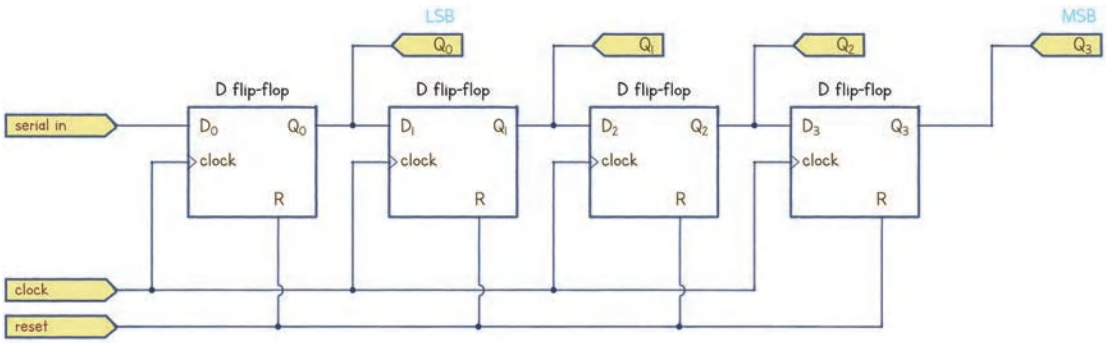


Figure 21-8: A 4-bit serial-in, parallel-out shift register

Note the similarity of this circuit to the ring counter (Figure 21-1). There are just two differences. First, the input for the flip-flop on the left—D_0—accepts an external input signal instead of being tied to 0 V. Second, the reset circuitry is configured so that the output is initialized to all zeros rather than a preset value of 1-0-0-0. However, the circuit still works the same way: On every clock cycle, it shifts its contents one position to the right and then loads a new digit from D_0.

If we use letters *a*, *b*, *c*, and *d* to signify four subsequent serial inputs on the D_0 pin, we can summarize the behavior of this shift register as shown in Table 21-6.

Table 21-6: Output of a Ripple Counter

Clock edge #	Serial in	Q_3 (MSB)	Q_2	Q_1	Q_0 (LSB)	Binary out
0	*a*	0	0	0	0	0000_2
1	*b*	0	0	0	*a*	$000a_2$
2	*c*	0	0	*a*	*b*	$00ab_2$
2	*d*	0	*a*	*b*	*c*	$0abc_2$
3	–	*a*	*b*	*c*	*d*	$abcd_2$

An example of an 8-bit register of this type is the 74HC164.

It's also possible to construct or buy parallel-in, serial-out shift registers (for example, the 74HC165). They work the opposite way: They take a multi-bit input value and then push it out bit by bit on every rising edge of the clock. Such devices can be a simpler alternative to multiplexers for serial communications, chiefly because they can be operated in a fire-and-forget mode: They don't need an external counter and don't require the transmitted value to be held in place until the transfer is done.

It's an enlightening experiment to wire the 74HC165 so that the eight parallel inputs can be set with switches and then have the outputs of a complementary 74HC164 chip connected to eight LEDs. If both ICs are supplied with a common clock signal, you should be able to toggle the LEDs at a distance over a single data line.

At this point, we have almost everything we need to piece together a computer. To wrap it up, we'll continue with digital-to-analog interfaces and then take a look at timing sources.

22

ANALOG TO DIGITAL AND BACK



In this chapter, we'll examine how digital logic can interface with the analog world. Thus far in Part IV of the book, we've focused on the building blocks for digital computing: We talked about arithmetic operations, memory cells, and circuits that decode instructions and route binary signals down the correct path. However, computation can't be forever stuck in the world of zeros and ones. Every now and then, a computer needs to read data from a sensor or drive an analog output device.

Some types of peripherals are naturally suited for binary logic: A keyboard is just an array of switches, each either open or closed. Similarly, an indicator LED is either on or off; there's usually no need for any states in between. Yet many other inputs and outputs are fundamentally analog, requiring some sort of signal translation for the digital circuit to work.

Digital-to-Analog Converters

Digital-to-analog converters (DACs) take binary numbers and translate them to analog voltage. The usual approach is to map the lowest possible binary input to zero volts and the highest possible input to a fixed upper reference voltage. For example, a 4-bit DAC operating at V_{ref} = 3 V might produce the mapping shown in Table 22-1.

Table 22-1: Inputs and Outputs for a 4-Bit DAC

Binary input	Output voltage
0000_2 (0)	0.0 V
0001_2 (1)	0.2 V
0010_2 (2)	0.4 V
0011_2 (3)	0.6 V
...	...
1110_2 (14)	2.8 V
1111_2 (15)	3.0 V

The outputs produced by the DAC are analog in the sense that they take more than two possible values. Nevertheless, the signal is quantized, changing only in discrete increments dictated by the number of possible binary input values. The accuracy of the DAC is fundamentally limited by the size of the quantization step; in the 4-bit example, the step works out to 3 V / 15 = 200 mV.

For a practical demonstration of the effects of quantization on an audio track, you might enjoy the following video: *https://lcamtuf.coredump.cx/slc/quantization*. In a nutshell, soundtracks quantized to 4 bits (16 voltage levels) are rather unpleasant to listen to; the source of the apparent high-frequency hiss and cracks can be understood by calculating the difference between a sine wave and its quantized image (Figure 22-1).

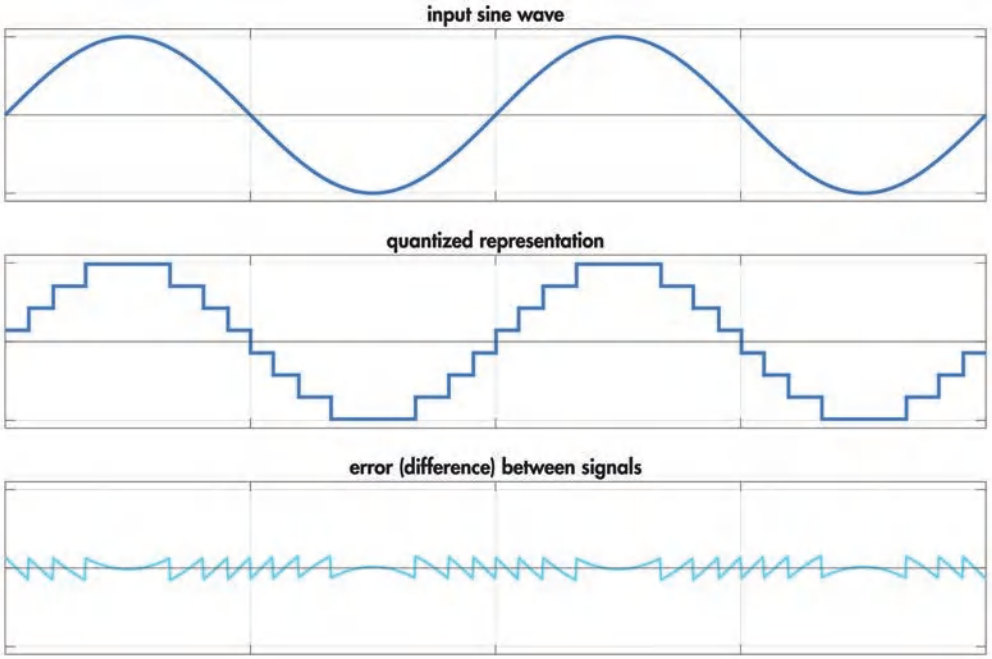


Figure 22-1: An illustration of the high-frequency noise component injected by the quantization of a lower-frequency sine wave

From the same audio recording, we can infer that 8 bits (256 levels) should be acceptable for many uses but will be accompanied by some low-grade distortion and hiss. Finally, about 14 bits (16,384 levels) are enough for a flawless listening experience, although this is often rounded up to 16 bits to align the size of an audio sample with a standard computer data type.

The resolution of a DAC is important, but the parameter doesn't tell the whole story. In every practical circuit, additional sources of nonlinearity and noise can overshadow the impact of the quantization step. Inexpensive audio equipment often claims 24-bit resolution, but that's just marketing hype. Even if true, it wouldn't matter—our ears can't resolve sound waves nearly that well.

Binary-Weighted DAC

Now that we've outlined the theory of digital-to-analog conversions, we can look at the simplest implementation of the idea: the *binary-weighted DAC*. A 4-bit version of this circuit is shown in Figure 22-2. It consists of nothing but four resistors, each twice the value of the preceding one.

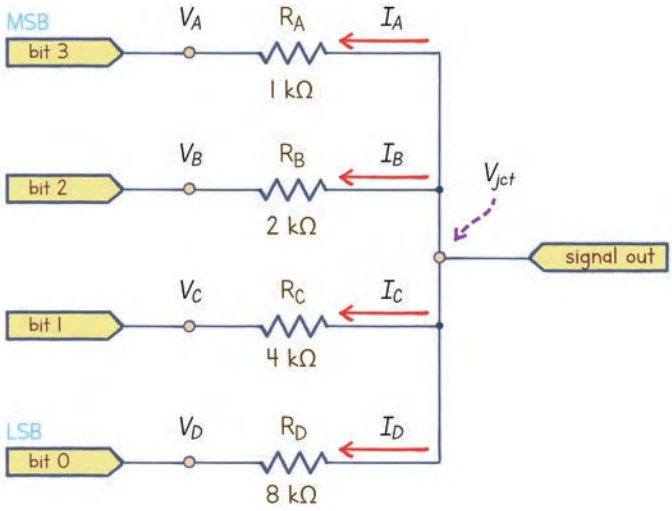


Figure 22-2: A 4-bit binary-weighted DAC, with voltages and currents annotated

We can start the analysis of this circuit by noting the obvious: If the binary input is 0000_2—that is, if all the input lines are shorted to ground—the output voltage must be 0 V. In the same vein, if the input is 1111_2, the output will equal V_{supply}.

To examine what happens between these extremes, we can apply the technique we previously used to analyze the summing circuit in Chapter 15. We denote the junction voltage as V_{jct}, choose n to represent the first resistance (1 kΩ), and then use Ohm's law to write the equations for the current flowing through each resistor:

$$I_A = \frac{V_A - V_{jct}}{1n}$$

$$I_B = \frac{V_B - V_{jct}}{2n}$$

$$I_C = \frac{V_C - V_{jct}}{4n}$$

$$I_D = \frac{V_D - V_{jct}}{8n}$$

If we assume no substantial output loading, from Kirchhoff's current law we can also assert that the currents must balance out: $I_A + I_B + I_C + I_D = 0$ A. If we combine

the formulas and move all the fractions to a common denominator, we get:

$$\frac{8 \cdot (V_A - V_{jct}) + 4 \cdot (V_B - V_{jct}) + 2 \cdot (V_C - V_{jct}) + 1 \cdot (V_D - V_{jct})}{8n} = 0\,\text{A}$$

After multiplying both sides by $8n$, gathering common terms, and solving for V_{jct}, the result is:

$$V_{jct} = \frac{8 \cdot V_A + 4 \cdot V_B + 2 \cdot V_C + 1 \cdot V_D}{15}$$

The junction voltage is the same as the circuit's output voltage, while voltages V_A, V_B, V_C, and V_D correspond to input bits 3, 2, 1, and 0. So, if the binary input is 1110_2 (14 decimal), the output voltage will be equal to $^{14}/_{15} \cdot V_{supply}$—precisely what we'd expect in a well-behaved DAC.

The design works because the values of the resistors double for every subsequent bit position, so the influence of the corresponding input signal is halved compared to its predecessor. This logic mimics the properties of positional notation for binary numbers: From left to right, every digit is half as important as the one before it.

The downside of binary-weighted DACs is that the required resistor values quickly become impractical. To avoid high idle currents, the resistance attached to the most significant bit can't be too low; in most cases, R_A = 1 kΩ would be a sensible starting point. Yet for a standard 16-bit DAC used in audio applications, this necessarily puts the LSB resistor at 2^{15} kΩ (about 32.8 MΩ). Such a range of resistances is difficult to manufacture with reasonable precision and good temperature stability on the die of an integrated chip.

R-2R DAC

The issue with untenable resistor values needed for a binary-weighted converter can be solved with a clever topology known as the *R-2R DAC*. This architecture, consisting of twice as many resistors but with minimal variation in component values, is outlined in Figure 22-3.

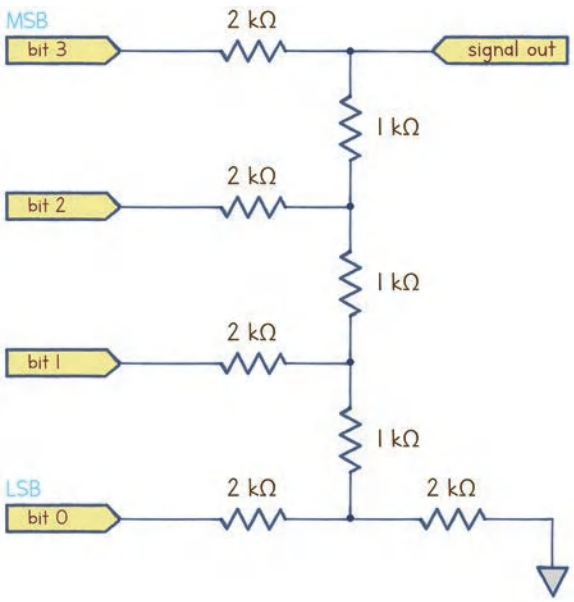


Figure 22-3: A 4-bit R-2R DAC

To make sense of the design, let's start with the section at the bottom, where we find two horizontally oriented 2 kΩ resistors next to the LSB input line. A zoomed-in view of this section, annotated with currents and voltages, is shown in Figure 22-4.

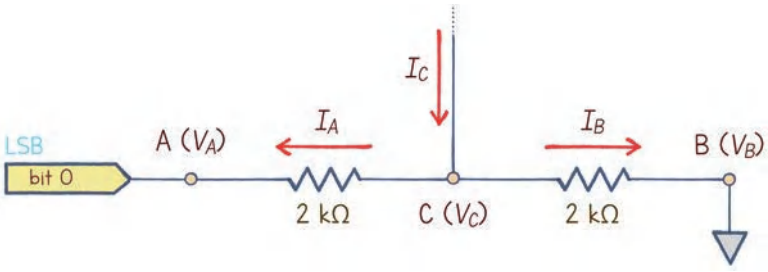


Figure 22-4: An analysis of currents and voltages in the bottom section of the R-2R DAC

From Kirchhoff's current law, we know that the current flowing into the central node C must equal the sum of the currents flowing out: $I_C = I_A + I_B$. We can't immediately calculate any of these currents, but it doesn't matter. For the resistor on the left, which straddles points A and C, we can write the following:

$$I_A = \frac{V_C - V_A}{2\,\text{k}\Omega}$$

The same reasoning can be applied to the resistor on the right:

$$I_B = \frac{V_C - V_B}{2\,\text{k}\Omega}$$

If we plug these formulas into the overall KCL equation, we obtain:

$$I_C = I_A + I_B = \frac{2 \cdot V_C - V_B - V_A}{2\,\text{k}\Omega}$$

The equation can be further simplified by dividing both the numerator and the denominator by two:

$$I_C = \frac{V_C - \frac{V_A + V_B}{2}}{1\,\text{k}\Omega}$$

This new formula is interesting. Upon closer inspection, it looks the same as the Ohm's law equation we'd write for a single 1 kΩ resistor placed between the junction point C (V_C) and some imagined, synthetic input voltage equal to the average of the voltages present at points A and B—that's $(V_A + V_B)/2$.

In this instance, V_A is the voltage representing the least significant input bit (#0), which we can also denote as V_0; meanwhile, V_B is fixed at 0 V, so the new synthetic input signal works out to $\frac{1}{2} \cdot V_0$. The analysis tells us that we can freely replace the two-resistor section we started with by the arrangement marked on the first panel of Figure 22-5 without changing the circuit's behavior in any way.

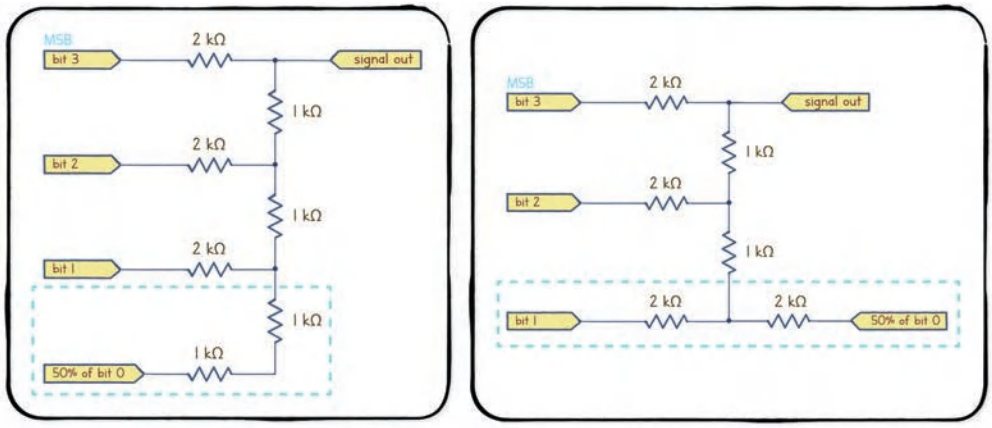


Figure 22-5: Substitutions in the R-2R DAC

The new synthetic input we've conjured is effectively connected to the next junction through two 1 kΩ resistors in series; we can further simplify this to a single 2 kΩ resistance, netting us the arrangement shown on the right panel of Figure 22-5. The section we ended up with repeats the earlier topology that we've solved using Kirchhoff's law. Applying the same logic, the section can be replaced with a single 1 kΩ resistor connected to a new synthetic input voltage that's the average of the two input signals—that is, a 50-50 split of V_1 and the previously calculated value equal to ½ of V_0:

$$V = \frac{V_1 + \frac{1}{2} \cdot V_0}{2} = \frac{1}{2} \cdot V_1 + \frac{1}{4} \cdot V_0$$

We can keep going until we run out of resistors; the collapsed equivalent of the 4-bit R-2R DAC is just a synthetic voltage source $V_{out} = \frac{1}{2} \cdot V_3 + \frac{1}{4} \cdot V_2 + \frac{1}{8} \cdot V_1 + \frac{1}{16} \cdot V_0$. We can also normalize the result so that each multiplier is an integer, netting us the following form:

$$V_{out} = \frac{8 \cdot V_3 + 4 \cdot V_2 + 2 \cdot V_1 + 1 \cdot V_0}{16}$$

That's the influence-halving behavior we need in a DAC, although it diverges from our earlier model in a small way: The numerator is 16 instead of 15, so the input of 1111_2 will produce a voltage that's equal to $\frac{15}{16} \cdot V_{supply}$. One quantization step is lost due to the fixed connection to ground at the bottom of the R-2R ladder. Still, the difference is minor and easy to compensate for.

Although the R-2R DAC is easier to build, manufacturing tolerances eventually become a problem for this topology too. In a 16-bit DAC, the influence of the least significant bit is supposed to be just about 0.003 percent of the MSB. If the topmost resistor attached to the most significant bit has an accuracy of "just" 0.01 percent, the error introduced on this side compromises the linearity of the converter far beyond what we might expect.

Oversampling Averaging DAC

The difficulty of producing ultra-precise resistors led to the development of *oversampling averaging DACs*. These devices generate a train of pulses that have low voltage resolution but change much faster than the converter's intended output frequency. The signal is then passed through an analog filter, effectively averaging the voltages over time.

As a basic example, let's consider a DAC that can output just two output voltage levels: 0 V or 5 V. The sequence of binary pulses produced by the control logic is then fed to a first-order R-C low-pass filter. Figure 22-6 shows a discrete-time simulation of the process. For the first half of the simulation, the duty cycle is 20 percent (1-0-0-0-0); for the second half, it's increased to 80 percent (1-1-1-1-0).

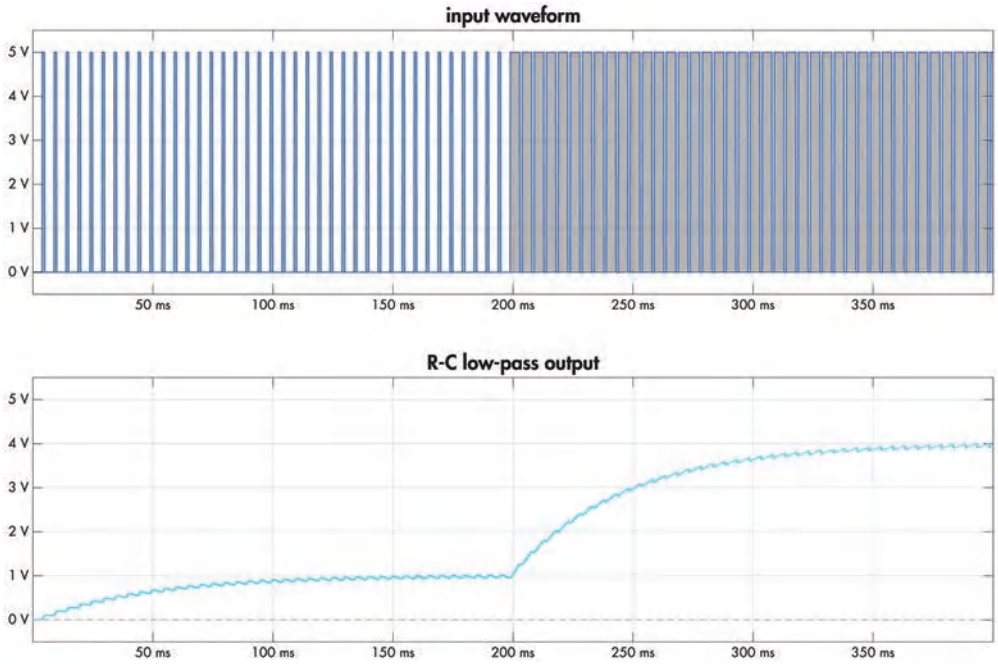


Figure 22-6: The behavior of a rudimentary 1-bit PWM DAC with R = 10 kΩ and C = 4.7 µF

As expected, during the first half of the simulation, the capacitor moves toward an equilibrium at roughly 1 V (20 percent of the supply voltage); in the second half, the output moves to about 4 V (80 percent of V_{supply}).

The speed of this DAC can be improved by modifying the modulation scheme to take the anticipated output error into account. For example, around the 200 ms mark, we changed the duty cycle to move the output voltage from 1 V to 4 V. That's a big move, so we could've briefly switched to 100 percent duty cycle to let the capacitor quickly catch up instead of blindly outputting 80 percent from the start. An error-compensating scheme that accounts for this is known as *delta-sigma modulation*. A model of the approach is shown in Figure 22-7; note that the output voltage converges more quickly than before.

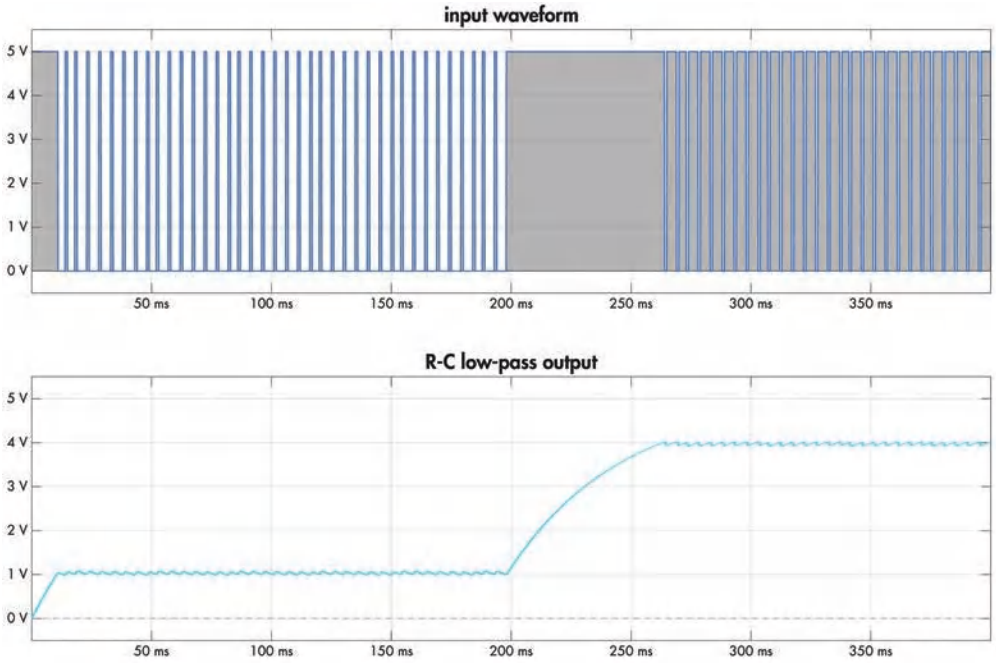


Figure 22-7: An improved, error-compensated modulation scheme for a 1-bit DAC

To further improve the converter's linearity, the R-C stage could be replaced with a noninverting integrator. We discussed integrator circuits toward the end of Chapter 15: They use a capacitor and an active feedback loop to provide a linear sum of input voltages over time. An integrator output stage would eliminate the pattern of exponential decay seen in the ramp-up sections of Figures 22-6 and 22-7.

Oversampling averaging DACs can use a high-frequency output stage that produces more than two voltage levels, but the benefit of the 1-bit approach is its excellent linearity. The performance of the design doesn't depend on the relative values of output resistors, only on the stability of the timing signal. As we'll discuss in the next chapter, ultra-precise timing is a problem that we know how to solve.

Rapid binary switching in an oversampling DAC produces more noise than resistor-based converters, but precisely because the frequency of the noise is very high, it's relatively easy to filter out. In some applications, the slow-changing stair-step pattern produced by conventional DACs can be more of a headache.

The trade-off with oversampling DACs is that the circuits are inherently slower because multiple clock cycles are needed to converge on the desired output voltage. When speed is more important than perfect linearity, a binary-weighted or R-2R DAC can get you the output you want in a single step.

Analog-to-Digital Converters

Compared to DACs, *analog-to-digital converters (ADCs)* are more challenging. It's not enough to cobble together a couple of resistors to get a binary representation of an arbitrary analog voltage. On the next few pages, we'll review a couple of ADC architectures, from the simplest to the most bizarre.

Flash and Pipelined ADCs

A *flash ADC* is a relatively simple active circuit that supplies the same input to a number of independent voltage comparators, each configured to compare the signal to a different internal reference point. An outline of the design is shown in Figure 22-8.

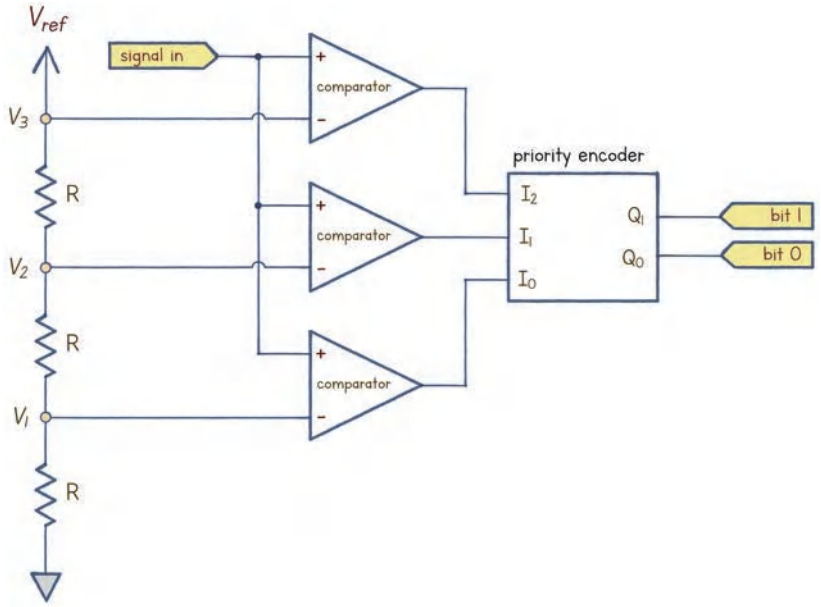


Figure 22-8: A simple 2-bit flash ADC

The resistor-based divider on the left splits V_{ref} into three voltages: $V_1 = \frac{1}{3} \cdot V_{ref}$, $V_2 = \frac{2}{3} \cdot V_{ref}$, and $V_3 = \frac{3}{3} \cdot V_{ref}$. These signals are then supplied to the negative input legs of three independent comparators. This causes each comparator to trigger on a different threshold of the common input voltage.

The outputs from the comparators are then fed to a simple logic circuit known as a *priority encoder* that generates a binary number representing the highest active input. Table 22-2 summarizes the behavior of the encoder.

Table 22-2: Truth Table for a 2-Bit Priority Encoder

I_0	I_1	I_2	Q_0	Q_1
0	0	0	0	0
1	0	0	0	1
any value	1	0	1	0
any value	*any value*	1	1	1

In effect, the circuit assigns a unique, sequential binary number to every sensed voltage threshold. The construction of the encoder circuit itself is left as an exercise for the curious reader.

Flash ADCs are used in specialized applications where speed matters more than anything else. However, the design doesn't scale because the number of comparators must double for every additional output bit. To build a 16-bit ADC, we'd need 65,535 comparators connected in parallel. It's possible to pack this many transistors on a single chip, but the IC would run into a host of other issues, from excessive power consumption to unacceptably high capacitance at the ADC's input pin. For these reasons,

true flash analog-to-digital converters are made only for fairly low resolutions, usually not more than 4 to 8 bits. When higher resolution is needed, we use an incremental solution known as a *pipelined subranging ADC* (Figure 22-9).

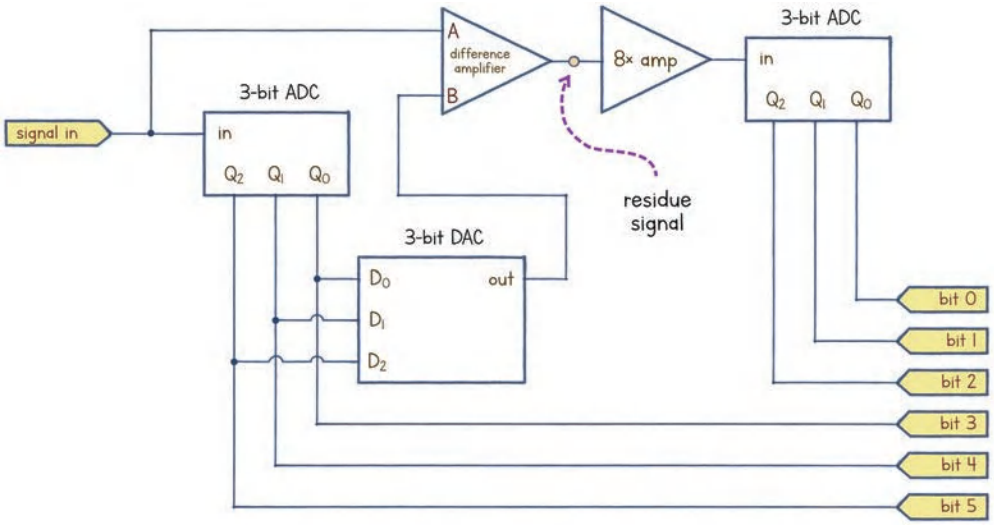


Figure 22-9: A sketch of a two-stage, 6-bit pipelined subranging ADC

The design starts with a low-resolution flash converter; in this instance, the ADC quantizes the voltage to 3 bits (eight steps). The output produced by this stage—a coarse quantization of the input signal—is then supplied to an internal DAC that synthesizes the corresponding analog voltage. The reconstructed voltage is subtracted from the original input using an analog difference amplifier circuit, similar to the topologies discussed in Chapter 15. The result of the subtraction is a "residue" representing the quantization error of the first stage. The residue is always smaller than a single quantization step (here, $V_{ref} / 8$) and represents the data that did not make the cut but would have been represented by lower-significance bits in a hypothetical higher-resolution ADC.

In our pipelined design, these additional bits can be resolved by amplifying the residue by an appropriate factor and feeding it to a second 3-bit ADC stage. In fact, the entire subtract-amplify-quantize process can be repeated several times before the cumulative analog error becomes too great. ADCs of this type usually achieve resolutions of around 12 to 16 bits.

Slope-Based ADC

Pipelined analog-to-digital converters are fast, but their accuracy is limited by the complexity of the analog signal path. For greater precision—at the expense of speed—we can use a *slope-based ADC*.

The architecture is shown in Figure 22-10. The circuit uses a single voltage comparator and a reference voltage that changes over time in a predictable way. A rudimentary example could involve a capacitor being charged through a resistor, but as before, a more linear slope can be obtained with an integrator.

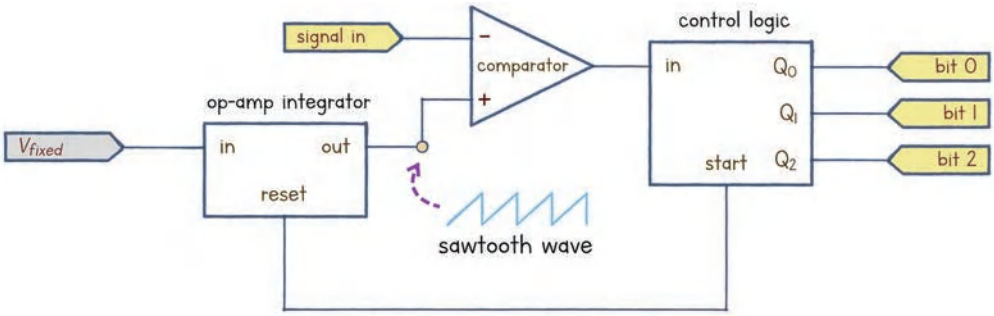


Figure 22-10: A rudimentary single-slope ADC

The integrator's output signal ramps up or down at a constant rate dictated by the values of the input resistor, the feedback capacitor, and the driving voltage (V_{fixed}). It follows that if we know these parameters and can measure the time it takes for the comparator to trigger, we can infer the unknown input voltage.

To remove the dependency on component values, the circuit can be modified as outlined in Figure 22-11, resulting in what's known as a *dual-slope ADC*. The circuit resets the integrator to a constant starting voltage (V_{start}), connects it to an unknown input signal for a constant time t_1, switches the integrator to a fixed reference, and measures the time t_2 needed for the integrator voltage to return to the starting point.

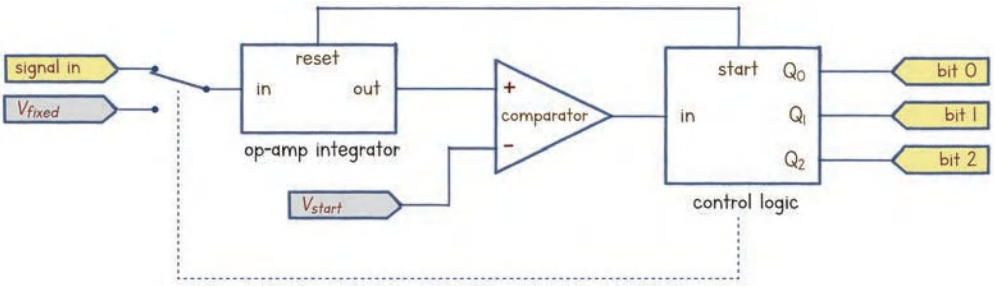


Figure 22-11: A dual-slope ADC

During the fixed-time charging stage, the integrator output will shift less if the values of R and C are larger, but the same effect also prolongs the fixed-voltage discharge period. The influence of R and C cancels out, and the unknown input voltage can be calculated solely from the ratio of t_2 to t_1.

Dual-slope ADCs are supremely accurate and exhibit low noise, but they are quite slow, usually achieving 30 samples per second or less. They are found in multimeters and other precision instruments but are not used in high-speed equipment.

Successive Approximation Register ADCs

The speed of a slope-based converter can be greatly improved with a *successive approximation register (SAR)*, a fancy name for an ADC equipped with an internal DAC that generates reference voltages and implements what's known in computer science as *binary search*. The idea is to have the DAC first generate a midpoint voltage (V_{ref} / 2); if the comparator indicates that the input signal is higher than this reference, the embedded digital logic in the ADC can instantly discard all values below the midpoint and perform the next test at the midpoint of the remaining range ($\frac{3}{4} \cdot V_{ref}$). Because each step halves the search space, a high-resolution approximation can be made in just a handful of steps.

The main limitation of the SAR ADC is the loss of precision caused by DAC linearity errors and the added digital switching noise that is absent in slope-based ADCs. Even so, the architecture is the sweet spot for many general-purpose applications and is common in electronic devices around the home.

Delta-Sigma ADC

The analog-to-digital converters we've discussed so far attempt to make a linear measurement of the input voltage and then express the reading as a binary number. A more unconventional design is the *delta-sigma ADC*, which outputs a high-frequency train of digital pulses and uses an unusual feedback loop to minimize the error between the analog input voltage and the ratio of zeros and ones over time.

The circuit, shown in Figure 22-12, is another application of the delta-sigma principle introduced earlier in the chapter, though it takes a bit more effort to understand exactly why it works.

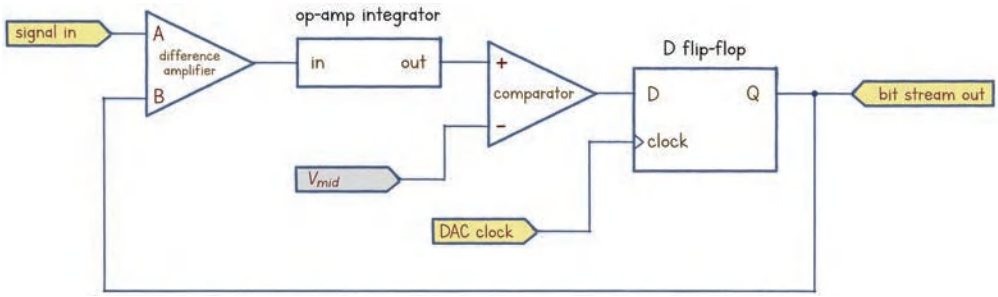


Figure 22-12: An outline of a simple delta-sigma ADC

To begin, let's assume that the analog input voltage equals $\frac{1}{3} \cdot V_{supply}$ and that the ADC is currently outputting a logical one (equal to V_{supply}) because the last output voltage of the integrator happens to be slightly higher than the midpoint of the supply (V_{mid}). Both the analog input and the digital output are fed to a difference amplifier that calculates the error between the input signal and the output of the ADC. In this instance, the output error is $-\frac{2}{3} \cdot V_{supply}$.

The calculated error is then fed to an integrator, which sums the values over time. A negative error will pull the integrator output below V_{mid}, prompting the circuit to output 0 V after the next clock cycle.

If the analog input voltage remains the same, the next calculated error will be a positive value equal to $\frac{1}{3} \cdot V_{supply}$. Critically, however, this error is half the size of the previous negative error, so it will take two clock cycles to undo the shift in the integrator output voltage. The circuit will then output a repeating pattern of 1-0-0 (Table 22-3).

Table 22-3: Delta-Sigma ADC Output for $V_{in} = \frac{1}{3} \cdot V_{supply}$

Clock cycle	ADC output	Momentary error
1	1	$-\frac{2}{3} \cdot V_{supply}$
2	0	$+\frac{1}{3} \cdot V_{supply}$
3	0	$+\frac{1}{3} \cdot V_{supply}$
4	1	$-\frac{2}{3} \cdot V_{supply}$
5	0	$+\frac{1}{3} \cdot V_{supply}$
6	0	$+\frac{1}{3} \cdot V_{supply}$
. . .	. . .	. . .

We can also analyze this for a different analog input signal. For example, for $\frac{1}{2} \cdot V_{supply}$, we get what's shown in Table 22-4.

Table 22-4: Delta-Sigma ADC Output for $V_{in} = \frac{1}{3} \cdot V_{supply}$

Clock cycle	ADC output	Momentary error
1	1	$-\frac{1}{2} \cdot V_{supply}$
2	0	$+\frac{1}{2} \cdot V_{supply}$
3	1	$-\frac{1}{2} \cdot V_{supply}$
4	0	$+\frac{1}{2} \cdot V_{supply}$
5	1	$-\frac{1}{2} \cdot V_{supply}$
6	0	$+\frac{1}{2} \cdot V_{supply}$
. . .	. . .	. . .

This time, we get a 50-50 split of zeros and ones. In other words, the duty cycle of the digital output pattern is determined by the input voltage, because that voltage sets the relative magnitude of the positive and negative errors produced by the difference amplifier.

This method also works for signals that vary over time, provided that the converter's clock runs much faster than the frequency of the signal to be sampled. Figure 22-13 shows a simulation of the response of a delta-sigma ADC to a slow-changing sine wave.

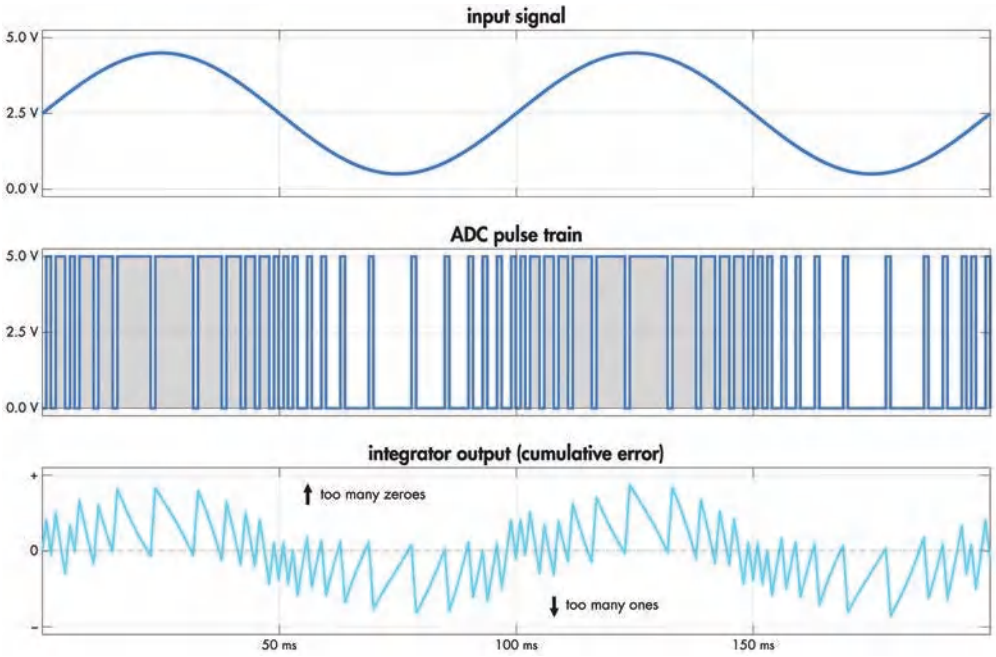


Figure 22-13: Simulated behavior of a simple, low-resolution delta-sigma ADC

In most applications, the generated sequence of bits needs to be processed by digital logic to convert it into binary numbers. The simplest algorithm is a moving average, although several other options exist.

Delta-sigma ADCs share their advantages and disadvantages with the corresponding DAC architecture: They are highly linear but not as fast as some alternatives. Because they are inexpensive and can handle audible frequencies well, they are commonly used in consumer audio equipment. The associated marketing term is *1-bit DAC*.

This concludes our analog detour on our journey through the world of digital computing. Next, let's take a look at what makes modern computers tick.

23

CLOCKS IN DIGITAL CIRCUITS



In this chapter, we'll look at the timing signals that synchronize the operation of digital circuits. All practical computing systems in use today rely on clock signals to ensure the orderly execution of tasks and to allow enough time for voltages to settle after any internal state change. In theory, clockless computing architectures are possible, but they greatly increase complexity and offer little in return.

Although digital circuits are almost always synchronous, this doesn't mean every part of a device must use the same timing signal. For example, in a desktop computer, a fan controller doesn't need to run at the same speed as the CPU core. Even within a single integrated circuit, clock signals may be divided or multiplied to suit specific needs. In addition, digital peripherals often have their own independent clock sources and communicate with the host using a carefully designed protocol that doesn't require their clocks to be in sync. To understand how these architectures work, we'll start at the source of the timing signal and work our way up.

Clock Sources

In Chapter 20 and Chapter 21, we noted that sequential digital logic is usually triggered by the rising or falling edges of binary timing signals. In some instances, these signals are generated on demand by other digital logic. However, most systems also include at least one oscillator that produces a continuously running square wave and sets the device's baseline operating frequency.

R-C Oscillators

By far the most common method for generating system clocks is to build an R-C relaxation oscillator in which a capacitor is repeatedly charged and discharged through a resistor. Such an oscillator can be integrated onto the die of many plug-and-play integrated circuits—microcontrollers, switching regulators, or certain memory chips—allowing these devices to operate with fewer external components.

We introduced R-C relaxation oscillators in Chapter 14 and revisited them in Chapter 15. Figure 23-1 shows another simple hybrid digital-analog design that uses a capacitor, a resistor, and a single Schmitt trigger NOT gate.

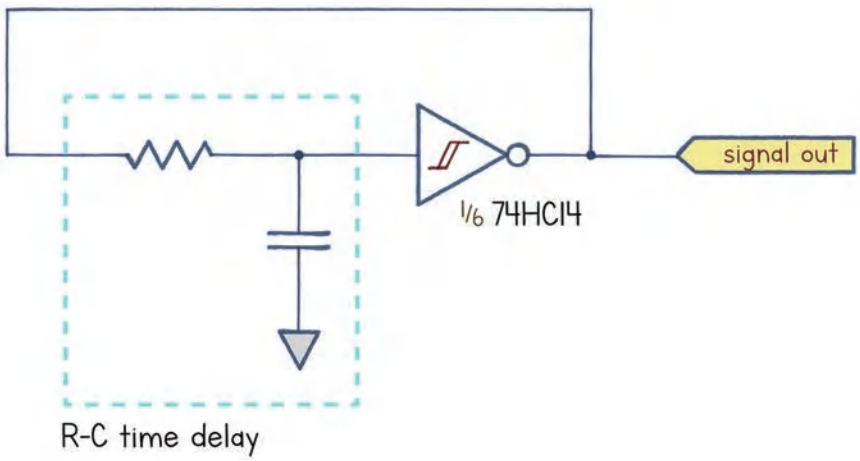


Figure 23-1: A simple digital relaxation oscillator constructed with a Schmitt trigger inverter

The operation of the circuit should be clear. The capacitor starts in a discharged state, putting the NOT gate's input at a logical zero and producing a logical one on the output leg. This signal is fed back through a resistor and begins to charge the capacitor. Once the capacitor reaches the Schmitt $0 \rightarrow 1$ transition voltage, the inverted output flips to zero and the capacitor starts discharging. Because the $1 \rightarrow 0$ transition voltage is significantly lower than the $0 \rightarrow 1$ threshold, there is no stable equilibrium, and the circuit continues producing a stream of zeros and ones.

Because of semiconductor manufacturing limitations, it's difficult to produce precisely trimmed R-C oscillators inside digital chips. Even after calibration, they're typically accurate only to about +/−5 percent across the design temperature range. Some clocks perform worse; for example, in Microchip's PIC32CZ MC70 family of microcontrollers, the internal slow clock used for power save modes is specified at between 20 and 57 kHz—a spread of about +/−50 percent.

Crystal Oscillators

The accuracy of R-C oscillators is sufficient for general-purpose computation but not for precise timekeeping: A wristwatch accurate to +/−5 percent could lose more than an hour per day. Similarly, an R-C–based digital radio receiver tuned to the San Francisco Bay Area's preeminent country station (92.1 MHz) could—to the listener's horror—end up receiving classical music broadcast at 90.3 MHz.

When better accuracy is required, we typically turn to quartz crystals (Chapter 9). These passive electromechanical components exhibit a dip in impedance whenever the driving AC voltage reaches the crystal's laser-trimmed resonant frequency. At around that frequency, you can also observe a brief change in the phase shift

between current and voltage: Normally, the crystal behaves like a capacitor, with current leading the driving signal. Near the resonant frequency, this relationship briefly reverses, making the component resemble an inductor instead (for a visual reference, see Figure 9-10 in Chapter 9).

Crystals don't generate timing signals on their own; they're just resonant elements. Therefore, an external circuit is required to drive the crystal and operate near its resonant region. One common example is the *Pierce oscillator* architecture, shown in Figure 23-2.

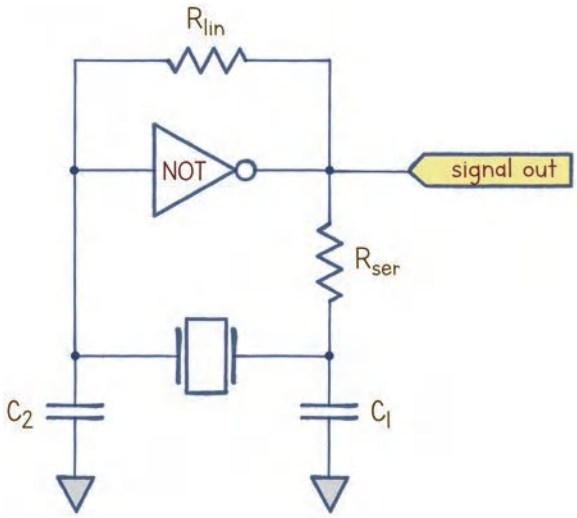


Figure 23-2: A Pierce oscillator constructed with a NOT gate

The Pierce oscillator is a minimalist phase-shift architecture, conceptually similar to the three-stage R-C circuits we discussed toward the end of Chapter 15. The NOT gate serves the same role as an inverting amplifier, while the linearizing resistor R_{lin} provides a degree of negative feedback between the output and the input leg, effectively reducing the inverter's gain.

As discussed earlier in the book, any phase-shift oscillator requires the bottom signal path—which loops the output signal onto the input—to produce a 180-degree phase shift at some specific signal frequency *f*. This shift effectively inverts the waveform, turning negative feedback into positive feedback that feeds oscillations at that exact *f*.

To simplify analysis of the bottom feedback loop, we can tentatively split the loop into two sections. The first consists of a series resistor R_{ser} connected to a shunt capacitor C_1. In broad strokes, this section forms an R-C low-pass filter; if the target oscillator frequency *f* is much greater than the filter's f_{knee}, it can produce a phase shift approaching 90 degrees.

The second section includes a series crystal feeding a shunt capacitance C_2. At most frequencies, the crystal behaves electrically like a capacitor, so this portion forms a C-C voltage divider, and the circuit will not oscillate. However, near the crystal's resonant frequency, the phase relationship between current and voltage begins to resemble that of an inductor. Within this very narrow frequency range, the network behaves like an L-C low-pass filter that adds another 90 degrees of phase shift, allowing oscillation to occur.

To be fair, this explanation is intentionally simplified for the same reason we discussed in Chapter 15: A pair of cascaded passive filters will exhibit some amount of stage loading that alters their performance. Nevertheless, because of the crystal's

distinctive frequency response, the influence of C_1, C_2, and R_{ser} on the oscillation frequency is minimal.

Many integrated circuits are designed to interface with crystals and incorporate the top portion of the Pierce oscillator on the die. We just need to connect an appropriate crystal and then provide two small capacitors to ground, usually around 20 pF.

Quartz crystals used in such circuits can be manufactured with tolerances as fine as 10 to 20 parts per million (0.001–0.002 percent). A slightly cheaper alternative is a *ceramic resonator*, which functions similarly but uses less expensive materials and provides accuracy of around 0.3 to 0.5 percent. At the high end, "oven-controlled" (OCXO) crystals—mounted within miniature heated enclosures—achieve accuracy measured in parts per billion, though they are power-hungry and require warm-up time.

Ready-made oscillator packages with integrated amplification are also available. These may employ quartz crystals or MEMS structures and are useful when PCB space is limited. Otherwise, performance is roughly comparable.

Clock Adjustment

At their core, quartz crystals rely on mechanical resonance, which makes them difficult to manufacture for frequencies above about 40 MHz—a far cry from the gigahertz speeds found in cutting-edge computing gear. On the other hand, as noted earlier, many parts of a digital system don't need to run nearly as fast as the CPU core. In fact, even for the CPU, it's useful to temporarily slow the clock when the system is idle to reduce power consumption and limit electrical noise.

In this spirit, the next few pages explore how to divide and multiply fixed timing signals to suit specific needs.

Division

Clock division is straightforward. The simplest solution was foreshadowed in the discussion of binary counters in Chapter 21. The circuit shown in Figure 23-3 uses a 74HC393 ripple counter to generate square-wave signals that complete one cycle for every two, four, eight, or sixteen cycles of the input clock. If its operating principle is not clear, refer back to Figure 21-1.

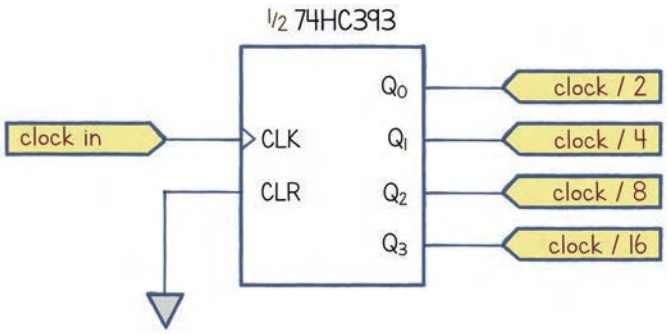


Figure 23-3: Clock division with a counter IC

Divisions by other factors are simple too. All that's needed is a free-running counter and a multibit comparator that triggers when the counter reaches a specified output value. When the comparator fires, it generates an output pulse that also loops

back as a reset signal for the counter. The counter's limit value can be hardwired or stored in a set of latches that form a binary register.

Multiplication

Compared with division, clock multiplication is a bit more involved. At first glance, the task may even seem impossible—how can we create something from nothing and conjure additional clock cycles where none exist?

The solution is elegant. The process begins with a *phase error detector*. There are several ways to construct this circuit; one robust and easy-to-understand design is shown in Figure 23-4.

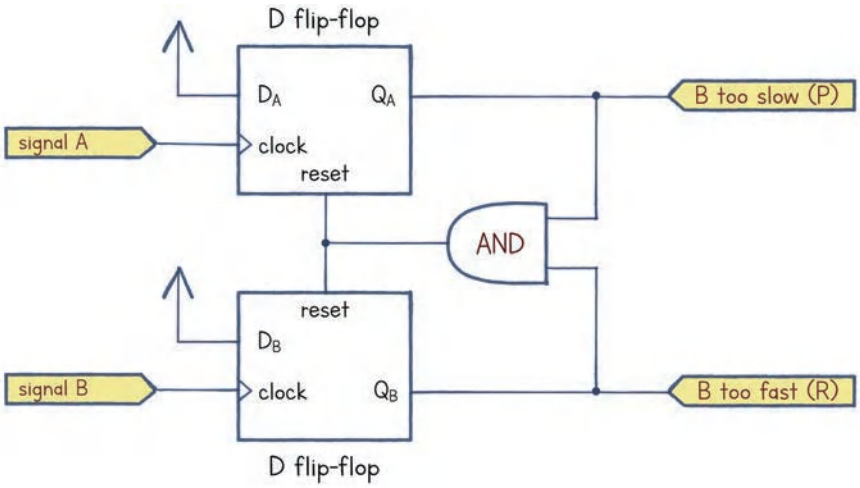


Figure 23-4: An all-digital phase error detector circuit

The purpose of the detector is to compare clock signal B to a reference clock supplied on input A. If the positive edge on input A occurs before the positive edge on B, the output of the upper latch (Q_A) goes high before the output of the lower latch (Q_B), indicating that clock B is running too slowly. Conversely, if the edge on B arrives first, the circuit generates a complementary output showing that B is running too fast. Once both clocks are high, the latches are reset through the AND gate.

An illustration of this behavior is shown in Figure 23-5.

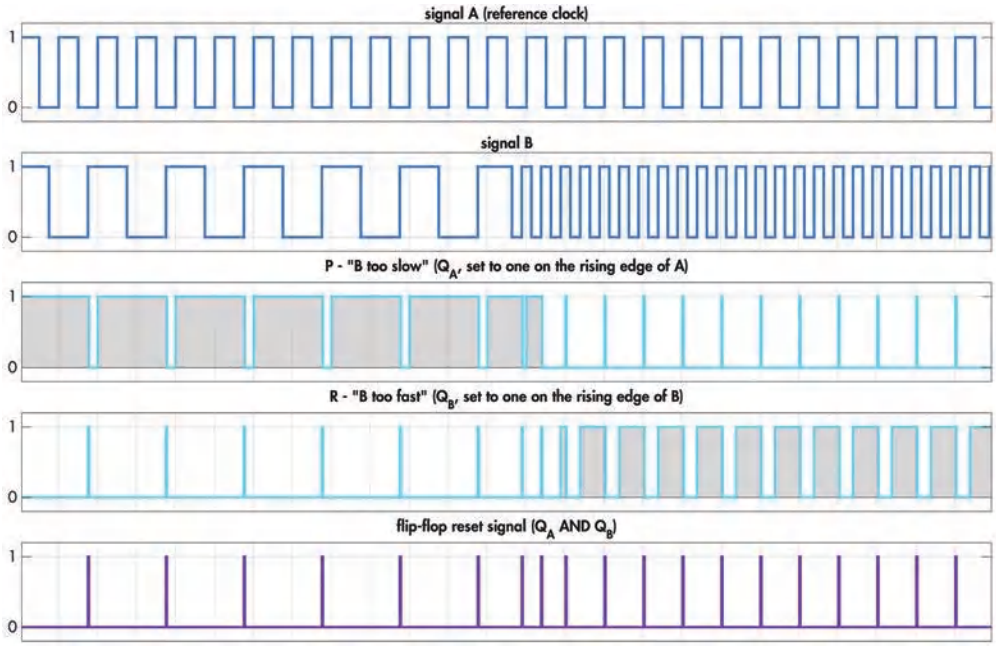


Figure 23-5: Output from the phase detection circuit

In effect, the detector produces longer pulses on the output labeled P when the analyzed clock signal lags behind the reference and longer pulses on the other output (R) when the signal runs ahead.

The detector is the fundamental building block of a circuit known as a *phase-locked loop (PLL)*. Despite its name, the primary function of a PLL is to align the device's output frequency with an input signal of some kind.

To understand the PLL architecture, let's begin with the top panel of Figure 23-6 and analyze it right-to-left. The output stage of the PLL is a voltage-controlled oscillator (VCO), which generates a square wave whose frequency is proportional to the applied input voltage. We briefly discussed VCOs in Chapter 15.

The VCO voltage is determined by a simple switched capacitor section in the middle of the circuit. This section has two digital inputs labeled U ("up") and D ("down"). Applying a digital signal to the "up" input turns on a transistor that gradually charges the output capacitor, raising its voltage and increasing the VCO frequency. Supplying a signal on the "down" input discharges the capacitor, lowering the voltage and slowing the VCO.

The final element of the circuit is the phase error detector, which compares the looped-back VCO output with a reference signal. The detector produces long pulses on the P output when the VCO frequency is lower than the local clock, or on the R output when the VCO runs too fast. These pulses adjust the capacitor voltage, nudging the VCO until its frequency and phase align with the input waveform.

This fundamental architecture has many uses in communications. PLLs are valuable for radio demodulation and for clock reconstruction in digital data streams. However, the circuit as shown doesn't yet perform the function we're after—it's not a clock multiplier.

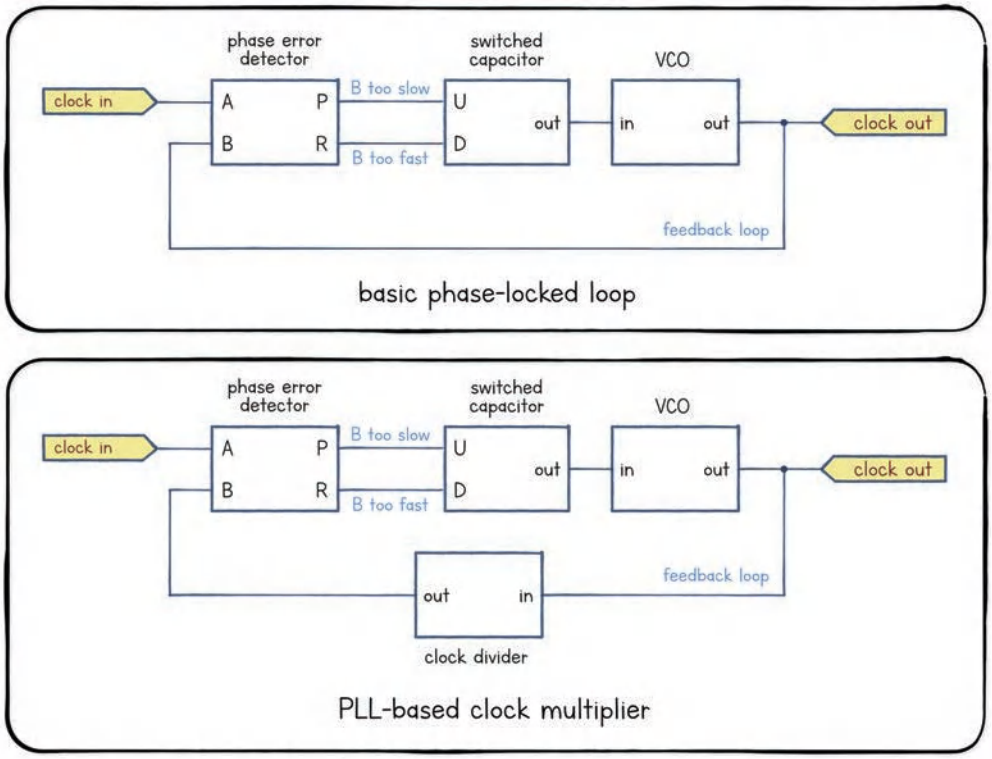


Figure 23-6: The basic architecture of a phase-locked loop: the fundamental circuit (top) and the use of a PLL as a clock multiplier (bottom)

To achieve that, we make a small but ingenious modification shown in the lower panel of Figure 23-6: We add a frequency divider in the feedback loop. With this addition, the phase error detector will continue adjusting the VCO until the *divided* output signal matches the input clock. If the feedback frequency is half the VCO frequency, the oscillator must run at twice the input clock rate for the PLL to achieve lock. This gives us a precise and versatile frequency doubler.

Experimenting with a PLL circuit is a worthwhile rite of passage for any aspiring electrical engineer, so I encourage you to spend some time with it. You can use a purpose-built chip such as the 74HC4046 or create an all-digital solution once we reach microcontrollers and embedded programming. If you have a knack for writing software, a discrete-time desktop model of a PLL can also be an engaging project. For example, could you use it to manipulate vocals in an audio track?

With clock generation, division, and multiplication now covered, we have all the building blocks for a thinking machine. Next, we'll explore how to assemble these blocks into a logical whole.

24

A COMPUTER AT A GLANCE



Throughout Part IV of this book, we've explored the mechanics of digital circuits, working our way from individual transistors to arithmetic logic, memory, operation sequencing, and data routing. Along the way, I've tried to relate these concepts to the architecture of a universal computing machine. In this chapter, we'll tie everything together by outlining a simple architecture for such a device.

Before we continue, a brief word of caution: Once the concepts click, the urge to build a do-it-yourself computer can be hard to resist. The dream is noble, but a noncommittal design that can do anything will probably excel at nothing. A CPU made from discrete transistors or logic gates will take months—if not years—to bring to life. If all it can do is solve equations and blink a few LEDs, it's unlikely to impress your friends. In electronics, focusing on a practical goal is more rewarding; for instance, you might enjoy building an old-school arcade cabinet to play a simple game.

Architectural Outline

With that advice in mind, a high-level diagram of a feasible architecture for a simple computer appears in Figure 24-1.

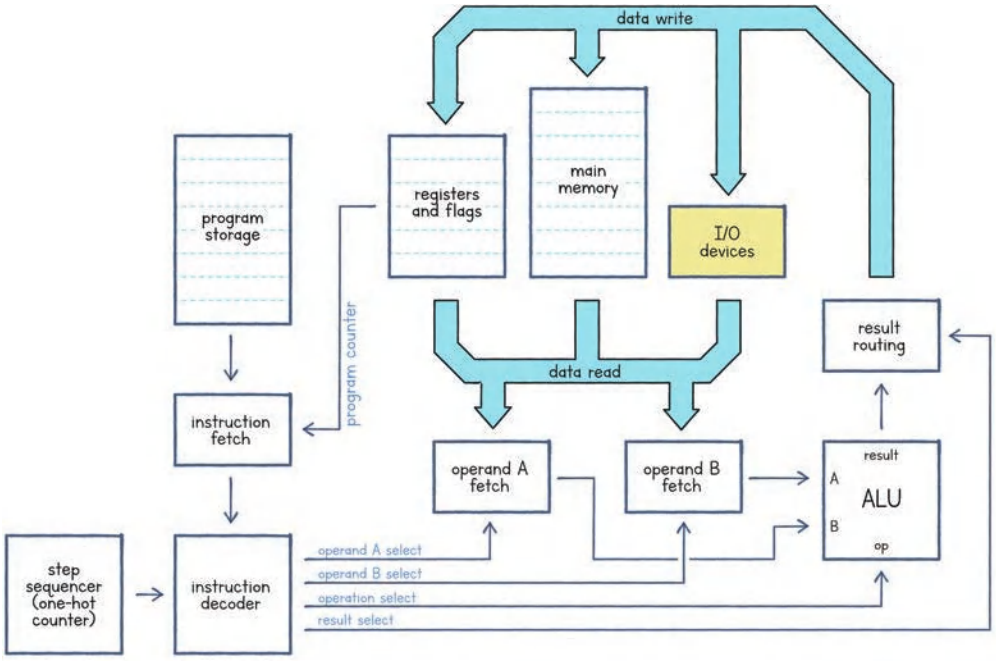


Figure 24-1: An outline of a rudimentary computer, clock signal routing omitted

Let's start with *program storage memory* (top left). This component holds the set of instructions the computer will follow, often implemented using some form of non-volatile technology that retains data even when power is removed. In the earliest computers, programs were entered bit by bit using punched cards or switches. For a proof-of-concept computer, an array of switches or other manually operated storage is still a reasonable starting point. In more practical systems, the initial program is typically stored in *flash memory*, where data is preserved as charges injected onto the gates of MOSFET-like cells through the application of a higher voltage. However the subsystem is designed, the program itself usually consists of similar components. As discussed in Chapter 21, each instruction is a sequence of bits that specifies which arithmetic operation to perform and the associated inputs and outputs. For example: "Fetch operands from memory locations X and Y, add them together, and write the result to output port Z."

To facilitate computation, a typical computer architecture also includes *registers*—small sections of tightly coupled temporary (volatile) memory, often implemented as banks of clocked flip-flops. The size of each register usually matches the input bit width of the arithmetic circuitry; for instance, in an 8-bit computer, most registers are 8 bits wide.

In most architectures, there is also a special register called the *program counter (PC)*. This register holds the address of the instruction to be executed. It normally advances automatically but can also be set to a specific value by certain instructions, allowing the program to jump around as needed—for example, to implement conditional execution or loops. At the beginning of each instruction cycle, the program counter's value is sent to the instruction fetch circuitry (middle left), which retrieves the corresponding instruction from program memory and latches it. The data is then passed to the *instruction decoder* (bottom left).

In its simplest form, the instruction decoder translates the compact binary representation of an instruction into a series of "select" signals that control several data buses. One of these buses connects to the *arithmetic logic unit (ALU)*, selecting which

arithmetic operation to perform. Another set of buses determines the data sources for the calculation and the final destination for the result. In practice, executing a single instruction might involve several discrete steps. In that case, the instruction decoder is coupled to a one-hot counter that cycles through multiple internal states for every tick of the program counter.

Operands for the calculation stage are retrieved using multiplexers (center). These employ the basic AND gate structure outlined in Figure 21-6 in Chapter 21 to route one of many possible inputs to their output terminal based on the "select" signals. The circuits switch between registers, main memory locations, and I/O lines (keyboards, displays, and so on) according to outputs from the instruction decoder.

The input values selected by the multiplexers, together with the identifier of the arithmetic operation, are fed to the ALU. The calculated output appears on the ALU's output lines and is then routed to the appropriate destination as dictated by the instruction decoder. This design allows the computer to flexibly receive data, store it in working memory, perform repeated calculations, and communicate the results to the outside world.

Instruction Set

From a theory-of-computation perspective, surprisingly little is required for a CPU to achieve *Turing completeness*—the ability to execute any known computer algorithm. All Turing-complete machines are conceptually equivalent: To a theoretician, your gaming rig is more or less the same as the vacuum-tube UNIVAC.

In fact, even the UNIVAC is overkill. An abacus, paired with a handwritten rule sheet, can reach the same goal. Combining a single arithmetic operation, such as subtraction, with a conditional jump ("change program counter to X if the register is zero") is enough. With sufficient effort, any other calculation can be restated in these terms—similar to how complex logic circuits can be built using just one type of gate, such as NOR or NAND. The semantics of the requisite instruction set could be summarized in a sentence or two.

Of course, such minimalist computing environments are not something we aim for in practice. They're known as *Turing tarpits*—languages so minimal that writing programs in them is an exercise in frustration. Though their theoretical power is unlimited, they perform even simple operations painfully slowly. Because transistors are inexpensive, we design real-world instruction sets to be both expressive and user-friendly. A typical processor supports about a hundred instructions—many redundant, yet helpful for expressing ideas clearly and performing common tasks in a straightforward way. These instruction sets support a variety of arithmetic operations (addition, subtraction, multiplication, division, and so on), allow data to be compared based on diverse criteria, and provide a wide range of convenient abstractions to manage program and CPU state.

This pursuit of convenience leads to an interesting design challenge. On modern chips, many instructions perform similar tasks of significant complexity—acting as small computer programs in their own right. This prompted some microprocessor manufacturers to develop *microsequencing*: an additional translation step that converts the user-visible instruction set into elaborate sequences of simpler operations (*microcode*) that are executed behind the scenes. The approach adds complexity to instruction decoding, but it also offers more flexibility in designing and revising the instruction set. In addition, it simplifies the internals of the ALU, because you might no longer need near-identical copies of the same circuit to perform related tasks.

Further Optimizations

In many modern CPUs, each major stage—data fetching, instruction decoding, arithmetic evaluation, and write-back—occurs as a separate step that's allocated a varying number of clock cycles. This sequencing allows slower operations to keep pace without delaying the entire processor. For example, because memory and peripherals typically run at lower speeds than the CPU core, we can give them extra clock ticks without slowing down the operations that don't require that kind of I/O.

In this architecture, it's hard to escape the conclusion that much of the time, most of the processor is doing nothing. During the result write-back cycle, for example, the ALU is essentially sitting there and twiddling its thumbs. This leads to the optimization known as *pipelining*, in which multiple instructions are processed simultaneously at different stages. For instance, decoding of the next instruction can begin while the ALU is still calculating the result of the previous one. In the same vein, the next operands can be retrieved in parallel with storing the previous result.

The challenge with pipelining is that it only works when instructions are independent. If the result of one operation is immediately needed by the next, the optimization may produce incorrect results. To fix the issue, the CPU needs to be able to detect data interdependencies in the program. A delay due to such an interdependency is known as a *pipeline stall*. A potential workaround is *instruction reordering*, a scheme in which the circuitry looks ahead at the program to spot instructions that can be grouped and executed in parallel even if they don't actually appear next to each other in program memory.

To do this safely, the CPU needs to know for certain that the reordering will not change the program's outcome. The question is often easy to answer, but it also limits the scope of the optimization in a fairly severe way. To take the trick to the next level, many chip vendors work around this constraint with *speculative fetching and execution*. In a nutshell, even if the processor can't decide with certainty what the program will do and thus can't be sure if some instructions are safe to execute out of order, it can still act on a hunch. It just needs to store the outcome of this speculative execution in a temporary holding location; if the prediction turns out to be correct, the result is immediately committed to memory without the need to recompute it. If the guess turns out to be wrong, the data is discarded—and life goes on.

All of this underscores the futility of building computers from scratch. The amount of clever work that goes into making CPUs perform well would be incredibly time-consuming for any single person to replicate. The underlying architecture is worth knowing about and can come in handy when trying to solve narrowly scoped tasks. That said, most of the time, we just grab a tiny $2 computer off the shelf, plop it onto the PCB, and tell it what to do. And that's the topic of Part V of the book.

PART V

SOFTWARE EATS THE WORLD

25

EXPLORING MICROCONTROLLERS



In Part V of this book, we'll learn how to solve hardware problems using programmable chips known as *microcontrollers (MCUs)*. An MCU allows many circuit design challenges to be reframed as software problems—essentially, as the task of writing computer code. The approach is often easier than solving transfer functions or constructing elaborate truth tables. Just as importantly, it lets us revise and extend designs without reaching for a soldering iron or ordering a new PCB.

For decades, embedding small computers into commodity electronics would have seemed absurd. The technology was too expensive and too difficult to use, and early processors required extensive external circuitry and considerable programming expertise. That began to change in the 1990s with the advent of user-friendly chips that integrated all the essentials on a single die and could be programmed in familiar languages using open source tools.

Today, microcontrollers are small, cheap, and easy to use, making them the go-to solution even for simple tasks. For example, instead of building a discrete oscillator, a designer can program an MCU to output a specific frequency and duty cycle—and adjust those values on the fly. Similarly, a complex analog audio filter can be replaced by a microcontroller performing real-time math on data streamed from an ADC.

Some old-timers scoff at the perceived decadence of placing tiny computers everywhere. In a sense, they have a point—but they're crying over spilled sand. These chips are often the easiest, most power-efficient, and most versatile way to solve the problem at hand.

Features of a Microcontroller

In a traditional computer architecture, the CPU is not a standalone product. It focuses solely on executing programs but depends on numerous external components: voltage regulators, clocks, memories, I/O controllers, and so forth. Although modern portable electronics have moved toward greater integration, the chip that powers your laptop or phone is still far from plug-and-play.

A microcontroller, by contrast, is designed from the ground up as a complete, embeddable computer. While there is wide variation among models, most include internal voltage regulation, built-in clock sources, on-board memory, and a range of peripherals—from ADCs and DACs to programmable-gain op-amps, display controllers, USB ports, and Ethernet networking. Heck, some even include built-in Wi-Fi radios.

The pinout of a small, versatile microcontroller—Microchip's AVR128DA28—is shown in Figure 25-1.

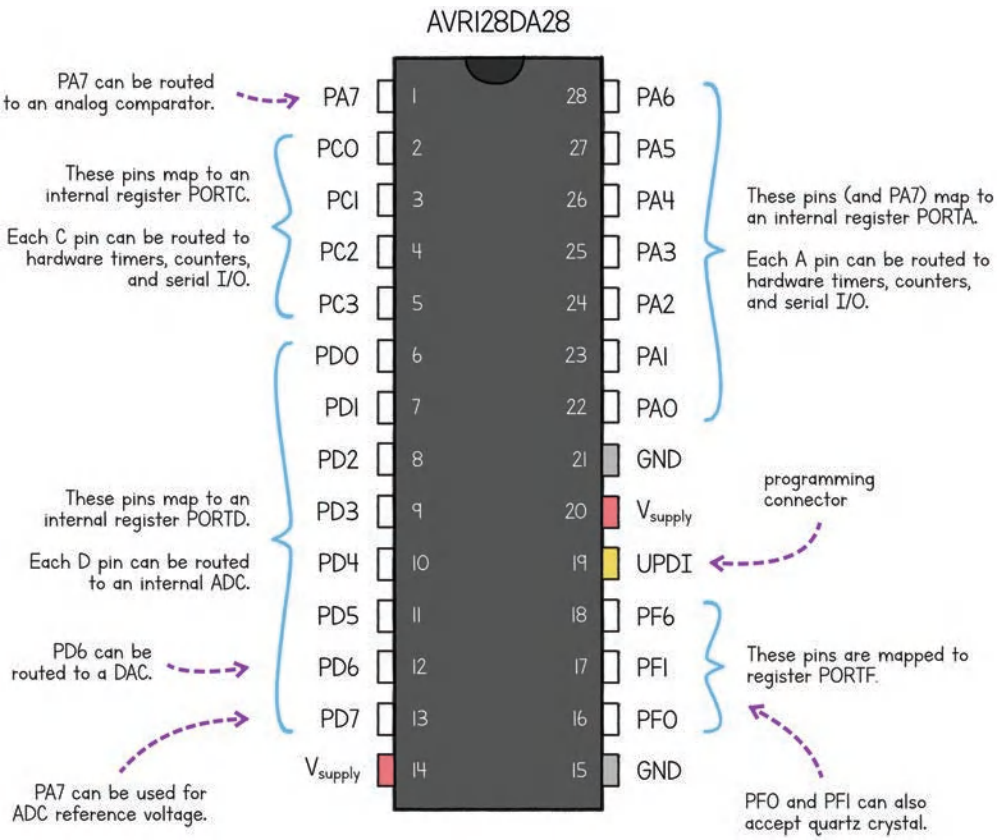


Figure 25-1: The pinout and a partial list of alternative functions for the AVR128DA28

Most of the pins default to working as digital inputs or outputs, with signal direction controlled by a binary flag in a special register. Writing the correct value to another register allows select ports to be switched to various alternative functions, including analog I/O.

To power up this particular chip, we simply need to supply a positive voltage between 1.8 and 5.5 V on pins 14 and 20 and connect ground to pins 15 and 21. It's also essential to add a ceramic decoupling capacitor of about 1 µF to shield the chip from supply ripple and buffer the digital switching transients that occur internally.

Uploading the program to the chip's nonvolatile memory is achieved by connecting a computer dongle to the *unified program and debug interface (UPDI)* pin.

Choosing the Right Device

Shopping for microcontrollers can be overwhelming: There are dozens of manufacturers and countless product lines to choose from. For one-off projects, remember that the cheapest option isn't always the best—you might save 20 cents only to spend hours working around the chip's limitations. On the flip side, going overboard on a high-end device may leave you with a product that's needlessly power-hungry and time-consuming to master.

Another complication is vendor lock-in. Each manufacturer designs its chips a bit differently, so experience gained when working with STMicroelectronics devices might not carry over to Microchip products. Most engineers pick a vendor they like and stay with it—but that also means that if you start with a niche chip maker that caters to a narrow range of applications, your options may be limited down the line.

I'll do my best not to sway you toward any particular brand. We'll start by reviewing common MCU use cases, then examine a specific architecture to keep the discussion grounded. That said, my goal will be to emphasize the commonalities.

Process Control

The most common use for microcontrollers is simple automation—taking input from sensors or control surfaces and applying straightforward logic to operate motors, small displays, lights, and so forth. The category covers everything from automatic-door sensors to electronic musical instruments to retro-style arcade games.

In such applications, we typically don't need gobs of memory and breakneck clock speeds. For years, the space was dominated by 8-bit microcontrollers, notably Atmel's ATmega series and its successor, Microchip's AVR Dx line. Nowadays, those chips face increasing competition from stripped-down 32-bit designs, including STMicroelectronics' STM32L0, Microchip's PIC32CM, and WCH's CH32V003. These devices typically rely on ARM Cortex-M0 cores licensed from a British company named Arm Holdings, or open source RISC-V equivalents.

When we describe computers, the number of bits usually refers to the unit of data that the ALU can handle in a single step. An 8-bit chip can manipulate 16-, 32-, or 64-bit integers but must break the operation into smaller steps. The task is handled transparently by the compiler that translates the programming language of your choice into the MCU's hardware instruction set, so the only practical consequence is that operations on some numbers will take more than a single clock cycle to complete.

Although "8-bit computing" sounds retro, modern chips in this category are orders of magnitude faster than the eponymous home computers of the 1980s. For example, the AVR Dx series runs at clock speeds up to 24 MHz, compared to 1 MHz for the Commodore 64. These architectures persist because they prioritize hassle-free simplicity over raw speed. The chips accept wide ranges of supply voltages, have uncomplicated memory and clock architectures, consume minimal power in sleep modes, and are packed with useful peripherals on the otherwise sparsely populated die. As a corollary, although 32 bits may sound better, these architectures offer little advantage in this niche: The limit we usually bump into is memory and I/O bandwidth, not the complexity of the ALU. In other words, it's best not to get sidetracked by this distinction. If you're working on simple automation, either of these solutions is fine.

Serious Compute

The limitations of budget MCUs become apparent when dealing with high data rates. A rudimentary 8-bit or 32-bit chip isn't up to the task of performing object detection in a camera feed, decompressing a video stream, or running 3D video games. These applications belong to a class of higher-end chips equipped with clock speeds up to 500 MHz, substantial amounts of memory, hardware floating-point units (FPUs), and direct memory access (DMA) controllers that execute background transfers between memory and I/O ports.

As of this writing, top-of-the-line products in this category are typically based on the ARM Cortex-M7 architecture, although some manufacturers bundle several lower-end cores into a single package. The three most common hobbyist-friendly options include STMicroelectronics' STM32H7 series, Microchip's PIC32CZ, and the RP2350 from Raspberry Pi, a company best known for its small, single-board computers (SBCs) that have no direct relation to this particular chip. A number of other companies—NXP, Infineon, Renesas, and others—also sell performance MCUs, but their products are less accessible due to weaker documentation, proprietary tools, and a general scarcity of reference code.

As noted earlier, high-end 32-bit chips are somewhat finicky because their internal architectures are far more complex. For example, past a certain point, embedded memories and I/O subsystems can't keep up with the arithmetic core, so the device may rely on multiple internal clocks. As a result, speeding the chip up or slowing it down often means adjusting various clock dividers and PLLs. If you get the sequence of operations wrong, the device may crash and require a special reset procedure to recover. Likewise, something as simple as driving an I/O pin might require enabling the correct subsystem and supplying it with a separate clock. The added complexity isn't something to fear, but it's not always worth your time.

Internet of Things

The final MCU use case involves devices that need connectivity over Wi-Fi or Bluetooth. The ecosystem of connected household appliances and wearables is commonly referred to as the *Internet of Things (IoT)*. The underlying radio protocols are complex and computationally demanding, so for a long time the functionality was added through separate hardware modules: purpose-built, high-performance RF microcontrollers running only the radio code.

Those modules are still available and can be connected to standard MCUs, but over time their manufacturers realized they could carve out some silicon and let others run programs directly on the Wi-Fi chip. The best-known example of this design is the ESP32 series from Espressif. If you own a "smart" coffee maker or litter box, there's a good chance an Espressif chip is driving it. These Wi-Fi MCUs are not as capable and varied as the products you can buy from other manufacturers, but when you're building a gadget that needs to be controlled from a smartphone, their convenience and price are hard to beat.

Another approach to networking is to forgo microcontrollers and use an SBC. The best-known brand is Raspberry Pi (again, not to be confused with the RP2350 microcontroller from the same company). There's also a rotating cast of competitors—Orange Pi, ODROID, Radxa ROCK, BeagleBoard, Arduino UNO Q, and others. These devices typically use conventional ARM Cortex-A processors and can run Linux while providing a PCB header that allows you to connect peripherals to individual I/O lines. The main perk of SBCs is familiarity: They're regular computers you can log into, play games on, or browse the web with. The trade-offs are higher cost, poor

power efficiency, and various failure modes that come with having a complex operating system between your program and the hardware pin you're trying to toggle.

Essential MCU Toolkit

With the chip in hand, you still need some equipment, software utilities, and know-how to program the device and troubleshoot potential issues in your code. To wrap up the chapter, let's go over the general prerequisites now—but don't feel pressured to take notes. We'll walk through a more specific example in Chapter 27.

Programming Language

The code that runs on a microcontroller can be written in a low-level, hardware-specific language known as *assembly* that's executed directly by the processor. That said, it's almost always preferable to use a human-friendly, platform-independent dialect that's translated to assembly by a compiler running on your PC. The compiled code is then uploaded to the chip.

The most common language used for MCU programming is C. It's uniquely suited to the task because it makes few assumptions about the operating environment and allows direct, low-level access to memory. If you're unfamiliar with C, my advice is to start by learning it on a desktop computer; embedded programming is comparatively unforgiving because it's harder to directly observe what your program does—or why it fails. An excellent, concise introduction to the language is *Effective C* by Robert C. Seacord (No Starch Press, 2024).

Some other languages, including Rust and Zig, have also gained limited traction in embedded work. However, as of this writing, none can be recommended without reservation: Their toolchains support far fewer chips than C, and where support does exist, it's often incomplete and poorly documented. With time, this will likely improve—but for now, C remains the path of least resistance.

The main difference between the MCU environment and the desktop environment is that you don't have access to the facilities normally provided by the operating system. For example, there is no standard way to print messages on the screen, open files, or connect to the internet. Depending on the peripherals connected to the chip, you might be able to do some of these tasks in an application-specific way, but there's no universal implementation of functions such as `printf()`, `open()`, or `connect()`. If you want the chip to interact with the outside world, you will need to implement these yourself (or borrow them from someone online).

At the same time, we don't need to reinvent the wheel. The embedded toolchain provides stripped-down versions of standard library functions that don't depend on specific peripherals. For example, you will still have access to string manipulation routines such as `strcpy()` and math helpers such as `sin()` or `pow()`. As noted earlier, the compiler automatically generates code to emulate arithmetic operations that aren't supported by the hardware itself. Many low-cost MCUs lack circuitry for floating-point arithmetic or integer division, but this simply means those operations will run more slowly than they would on a more capable chip.

Programmers

Modern microcontrollers can be programmed without removing them from the PCB; all you need is a USB dongle connected to ground, the positive supply rail, and the appropriate programming pins on the chip.

To make this easier, most chip manufacturers sell their own USB programming dongles. For Microchip, I recommend the basic model of Atmel-ICE. It costs about $80 and supports the *in-system programming (ISP)* protocol for older ATmega chips, the streamlined UPDI scheme for the AVR Dx series, and the *single-wire debug (SWD)* protocol for the PIC32CZ. Budget-conscious readers should also have luck with the clunkier but cheaper MPLAB SNAP unit ($10) or third-party designs such as Adafruit's UPDI Friend ($7). For STMicroelectronics, the equivalent dongle is the ST-LINK/V2, usually priced at around $20.

Some microcontrollers come with integrated USB interfaces and a baked-in bootloader in read-only memory, allowing them to be programmed without a dongle. That's convenient, but it doesn't necessarily save you money if you end up paying more for each chip or for the USB connectors that need to be soldered to every board.

Build Environment

To set up the build environment on your desktop computer, you need to consider how you prefer to write code. Many manufacturers provide an *integrated development environment (IDE)* that combines a visual editor, compiler, debugger, and programming utility. STMicroelectronics offers STM32CubeIDE, while Microchip provides MPLAB X IDE.

The complication is that many software engineers already have a favorite editor and might not be eager to switch. If you fall into that camp, you can assemble an equivalent setup using open source tools. For 8-bit AVR microcontrollers, the toolkit consists of `avr-gcc` (the compiler), `avr-libc` (the C function library), and `avrdude` (the dongle utility). These components are available in most Linux distributions through the built-in package manager, although they're not always up to date. Therefore, when working with the latest chips, it's sometimes necessary to rebuild these programs from the source code.

The situation for 32-bit ARM-based MCUs is a bit more complex. The open source toolchain exists, but some components may not be installed by default. The compiler, `gcc-arm-none-eabi`, is usually included; if not, it can be downloaded from Arm Holdings' website (*https://developer.arm.com/downloads/-/arm-gnu-toolchain-downloads*). This component needs to be paired with `newlib` (the C function library), `openocd` (the dongle tool), and the separately downloaded ARM CMSIS library, which describes the CPU core. The final piece is a set of vendor-specific files that describe all the peripherals. Microchip calls this a *device family pack (DFP)*; STM and some other companies refer to it as a *hardware abstraction layer (HAL)*.

Next, we'll review the peripherals available on a typical microcontroller. With this out of the way, in Chapter 27, we'll set everything up and get a simple program running on an 8-bit MCU. We won't be covering the same process for a 32-bit chip, but if you'd like a walk-through for open source ARM Cortex-M development, see my article at *https://lcamtuf.coredump.cx/blog/arm*.

26

WORKING WITH PERIPHERALS



In the previous chapter, I noted an important limitation of programs deployed to a microcontroller: They can perform calculations just as they would on a desktop computer, but the code has no access to the input and output facilities normally provided by the operating system. Without these facilities, it's not obvious how the program can communicate with the user or with other electronic devices.

Luckily, a good chunk of the MCU's silicon die is dedicated to filling this gap. In this chapter, we'll examine how microcontrollers interact with the outside world by highlighting some of the onboard peripherals of the AVR128DA28 chip. As mentioned earlier, I'm using this particular device to keep the discussion grounded, not because I think you should give money to Microchip. The key point is that although terminology and small details differ from one brand to another, the principles are broadly applicable to most modern chips.

As you work through the chapter, I suggest pulling up the MCU's extensive 700-page datasheet (*https://lcamtuf.coredump.cx/slc/avr-da*). Don't panic—no one actually reads it cover to cover. That said, you'll see me refer to various register names and their functions along the way; when I do, try to locate the corresponding information in the spec. This approach will make it easier to research other microcontrollers or subsystems and to understand how to interpret their diagrams.

Memory-Mapped Registers

On modern microcontrollers, most peripheral functionality is accessed by reading or writing to specific memory locations known as *memory-mapped registers*. For example, writing to a certain address may produce a logical one on an output pin, or it might initiate the conversion cycle of the internal ADC.

Memory-mapped registers also control many aspects of chip configuration. They can reroute a single physical pin to different digital or analog subsystems, change the chip's internal clock speed, enable or disable subsystems to conserve energy, and even force a full reset if desired.

When working in a higher-level programming language such as C, you don't need to know the numerical addresses of these registers. Each one has a human-readable name, such as `PORTA.OUT` or `ADC0.CTRLA`. These names are listed in the device-specific header files that your build system should automatically include. The files also instruct the compiler not to use these special memory locations for any other purpose.

Many I/O and configuration registers are organized as collections of bit fields. They have a standard bit width dictated by the ALU architecture, but individual bits often control independent functions. For example, in the `ADC0.CTRLA` register, bit 0 (the LSB) enables or disables the ADC, while bits 2 and 3 form a `RESSEL` field that can be used to select the converter resolution. Manufacturers try to simplify the structure of these fields by defining named constants that can be combined to calculate the correct numerical value to write to the register. In practice, however, it often leads to tongue-twisters such as:

```
TCA0.SINGLE.CTRLB = (TCA_SINGLE_CMP0EN_bm | TCA_SINGLE_WGMODE_SINGLESLOPE_gc);
```

The names are hard to remember or pronounce, and because these values are chip-specific and never change, this layer of indirection offers limited benefit. To keep the code cleaner, we'll instead write out the underlying binary values, and then I'll explain what each operation is trying to accomplish.

Digital Port I/O

As mentioned earlier, most general-purpose I/O pins on a microcontroller are configured by default as digital inputs and grouped into registers called *ports*. In Figure 25-1 in Chapter 25, you can see eight pins assigned to port A, four to port C, eight to port D, and three to port F. The gaps may seem puzzling, but they occur because the same semiconductor die is mounted in packages with anywhere from 14 to 64 pins. For the 28-pin package, the manufacturer exposes only a subset of pins, often based not on the primary function but on the secondary capabilities of each line.

For digital input, the logic-level voltage applied to a pin can be read through the `PORTn.IN` register, where *n* corresponds to the port identifier (A, C, D, or F). For example, bit 0 in `PORTA.IN` corresponds to the leg labeled PA0 (pin 22), while bit 5 maps to PA5 (pin 27). To check if there is a logical one present on PA5, we can write the following C code:

```
if (PORTA.IN & 0b00100000) { /* If PA5 is high... */
  ...
}
```

If you're new to C, the `if (...) {...}` construct represents a conditional code block: The statements in the brackets execute only if the condition enclosed in the parentheses evaluates to a nonzero value. The expression reads the state of the port

and performs a bit-level AND (&) with a binary constant equal to 0010000_2. In effect, the result will be nonzero only if bit 5 of PORTA.IN is set.

Like any other MOSFET-based chip, the state of an MCU input pin is indeterminate if left floating—for example, when connected to an open switch. As discussed in Chapter 12, this can be resolved with a pull-up resistor. For convenience, many microcontrollers include internal pull-up resistors, typically 20 to 40 kΩ, that can be enabled for any input pin. On this AVR device, we do this by setting the PULLUPEN flag (bit 3) in the PORT*n*.PIN*x*CTRL register. To enable pull-up on PA0, we can write:

```
PORTA.PIN0CTRL = 0b00001000; /* Pull-up enable for PA0 */
```

Most digital I/O pins can also be reconfigured as outputs by setting the corresponding bit in the PORT*n*.DIRSET register. Once a pin is configured for output, we can pull it high by writing to the corresponding offset in PORT*n*.OUTSET, reset it to zero with PORT*n*.OUTCLR, or toggle the output value with PORT*n*.OUTTGL:

```
PORTA.DIRSET = 0b00001000; /* Configure PA3 for output */
PORTA.OUTSET = 0b00001000; /* Set PA3 to a logical one */
```

It's also possible to set all outputs simultaneously to a specific 8-bit value by writing the desired binary number directly to PORT*n*.OUT.

Parallel 8080/6800 Protocols

Microcontroller input and output pins are often used to communicate with other digital chips—typical examples include display controllers, external memories, and accelerometers. These communications usually follow one of several common, quasi-standard protocols.

Parallel communication is particularly common with small OLED and LCD display modules. The scheme comes in two flavors: 8080 and 6800, named after the Intel 8080 and Motorola 6800 processors that popularized this approach. Both interfaces use multiple parallel data lines; for an 8-bit bus, the lines are typically labeled D0 (LSB) through D7 (MSB). In the 8080 variant, the data lines are accompanied by three control signals: RS (register select), WR↑ (write), and RD↑ (read).

To send data to an external device using the 8080 protocol, the MCU first sets the RS line to indicate whether it's sending a command (0) or data (1). The meaning of these terms depends on the device we're talking to—a command might instruct the display to clear its screen, while data might represent a sequence of pixels to render. After the selection is made, the RD↑ line is held high, and the WR↑ line is pulled low. The byte to be transmitted is then presented on the D0 through D7 lines. The rising edge of the WR↑ signal triggers receipt of the data, so the signal effectively functions as a bus clock. If the host needs to receive data from the peripheral, it keeps the WR↑ line high and strobes the RD↑ signal. When bidirectional communication is not necessary, RD↑ can be wired to the positive supply rail.

The 6800 protocol is similar but replaces RD↑ and WR↑ with a falling-edge clock on the E↓ (enable) line, along with an R/W signal that determines transfer direction—1 for read, 0 for write. A comparison of the two schemes is shown in Figure 26-1.

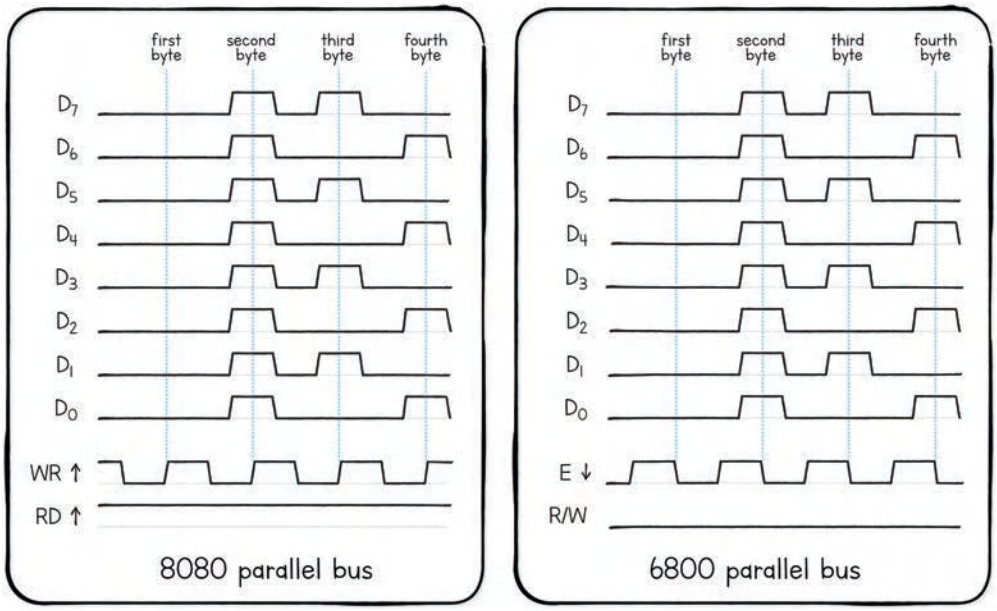


Figure 26-1: Signaling on the 8080-style and 6800-style parallel buses. In both cases, the transmitted data is a sequence of four bytes: 00000000_2, 11111111_2, 10101010_2, and 01010101_2.

The protocols don't define a maximum transfer rate; the limit depends on the internal clock of the receiving device. Older text-only LCD modules may be limited to kilohertz-range bus clocks, while more modern products can usually handle 15 to 20 MHz or more.

Owing to their simplicity, these parallel protocols are usually implemented in software. However, some high-performance microcontrollers include direct memory access controllers that can move data from memory to an output port without involving the MCU core. This can free up clock cycles—for example, allowing the core to compute the next video frame while the previous one is still being transmitted.

Serial Peripheral Interface

Parallel communication is fast but consumes MCU pins and PCB space. A slower but more economical alternative is to send bits one after another. Several serial protocols appear in digital circuits, but we'll focus on the most common variety: the *serial peripheral interface (SPI)*.

In its basic form, SPI uses one line for a bus clock (SCK), one for the bit stream going from the microcontroller to the external peripheral (MOSI or COPI), and another for data sent in the opposite direction (MISO or CIPO). An additional active-low chip select signal ($\overline{\text{CS}}$) is sometimes used to indicate the beginning and end of a transmission, although not all devices require it; when it isn't needed, the line can be tied to ground.

Figure 26-2 shows the most common operation mode of SPI, known as mode 0. The host provides the clock signal. Both endpoints drive bits onto their respective serial lines on the falling edge of SCK, and the data is loaded on the rising edge of the clock, from MSB to LSB. Other SPI modes exist that trigger on the falling edge of the clock or keep the SCK signal idle at a logical high, but they are seldom encountered in everyday designs.

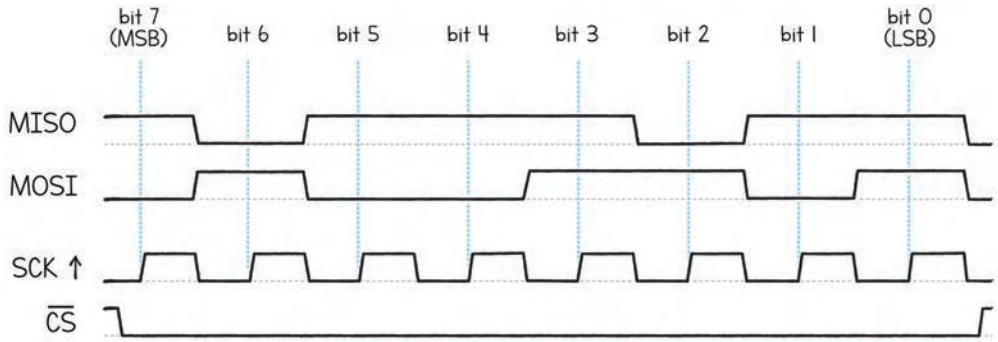


Figure 26-2: Standard (mode 0) SPI communications. The MCU is sending out 01001101_2 on the MOSI line, ordered MSB to LSB. The peripheral is using the MISO line to send back 10111011_2.

The scheme allows both endpoints to transmit at the same time; this is known as *full duplex*. In practice, full-duplex transmission isn't used often, because the peripheral usually has nothing to report until it receives a command from the host. In such cases, the host simply disregards any apparent data on the MISO line while it's sending instructions. It then follows up by transmitting bogus data (typically all zeros) solely to provide a clock signal and allow the peripheral to respond. The limited usefulness of simultaneous transmissions led to the development of *dual SPI*: a flavor of the protocol that uses the existing lines to send two bits at once in half-duplex mode. Further performance improvements include *quad SPI (QPI or QSPI)* with four data lines, and the somewhat comical *octal SPI*, which starts to resemble a parallel bus with extra steps.

The SPI protocol is commonly used to talk to small, low-pin-count devices, such as high-end analog-to-digital and digital-to-analog converters, low-speed external memories, and similar components. Most MCUs provide hardware counters and shift registers to automate the transmission of larger units of serial data in the background. On the AVR Dx series chips, we can activate the SPI transceiver by writing 00110001_2 to the `SPI0.CTRLA` register; this configures the subsystem for standard host-side operation at maximum speed. After that, a transfer can be initiated by writing a byte to `SPI0.DATA`. After a short while, the most significant bit in `SPI0.INTFLAGS` will be set to indicate completion of the transfer, and the response from the peripheral will be available in the same `DATA` register.

As with parallel communication, there is no fixed clock limit for the SPI protocol; many devices can handle speeds ranging from 10 to 40 MHz.

Pulse Width Modulation

An immensely useful feature of many microcontrollers is their ability to generate PWM signals automatically. As discussed in Chapter 16 and Chapter 22, PWM can be used for motor speed control, light dimming, voltage regulation, audio signal synthesis, and more.

Depending on the architecture of the chip, PWM may be a separate function or a configuration option for a general-purpose timer-counter circuit. On the AVR128DA28 chip, it's a feature of the 16-bit TCA0 timer. To get started, we need to access the `TCA0.SINGLE.CTRLB` register and set two mode flags: `WGMODE` (bits 0–2) must be set to 011_2 for PWM operation, and the `CMP0EN` flag (bit 4) must be set to 1 to route the signal to the default timer output pin (PA0):

```
TCA0.SINGLE.CTRLB = 0b00010011; /* PWM mode, output to PA0 */
```

By default, the TCA0 counter runs at the same speed as the main MCU clock. This clock can be divided by writing a nonzero value to the CLKSEL field (bits 1–3) of the TCP0.SINGLE.CTRLA register. The value 011_2 corresponds to a division by 8, so if the chip is running at 24 MHz, the following code will produce a timer frequency of 3 MHz:

```
TCA0.SINGLE.CTRLA = 0b00000110; /* Clock divided by 8 */
```

In this configuration, the counter counts up from zero until it reaches an upper limit programmed into the TCA0.SINGLE.PER register. After it hits this limit, the count restarts from zero. For example, if we program the value 299 into the register, the counter will cycle through 300 possible values, with an effective repeat frequency of 3 MHz / 300 = 10 kHz.

The state of the PA0 output pin is determined by comparing the counter to the value stored in the TCA0.SINGLE.CMP0 register. If CMP0 is less than the current counter value, the output will be low; if the counter is equal to or greater than CMP0, the pin transitions to a logical one. In effect, the frequency of the waveform is set by the upper limit in the counter (the PER register), while the duty cycle is controlled by CMP0 (Figure 26-3).

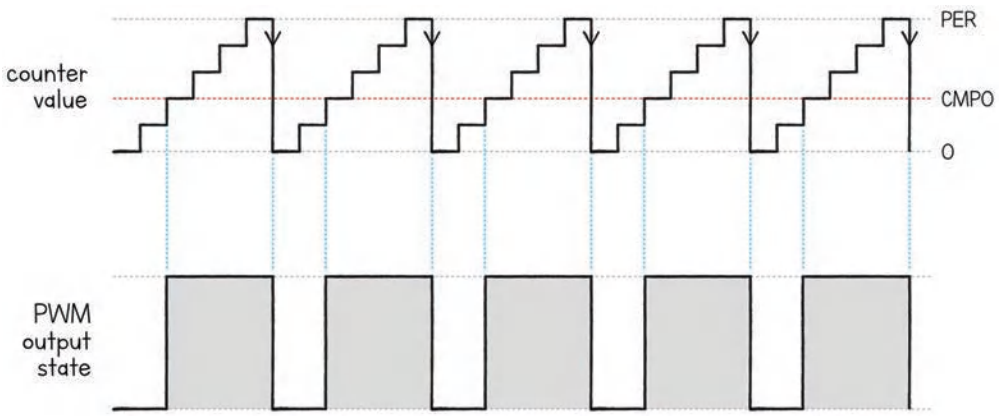


Figure 26-3: Using the AVR Dx timer-counter for PWM output. The underlying binary counter runs from zero to the value stored in the PER register; the value in the CMP0 register sets the duty cycle for the output pin.

Continuing from the earlier example, the following code will configure a 33 percent duty cycle at 10 kHz:

```
TCA0.SINGLE.PER  = 299;
TCA0.SINGLE.CMP0 = 200;
```

Zero-indexed counters can be confusing; to double-check the math, note that there are 300 counter values between 0 and 299, and 100 values between 200 and 299. The duty cycle is therefore 100 / 300, or about 33 percent.

With all the configuration parameters in place, the final step to generate a PWM signal is enabling the counter. This is done by toggling the least significant bit of the CTRLA register:

```
TCA0.SINGLE.CTRLA |= 1; /* Timer enable */
```

Further updates to the duty cycle frequency should use the buffered registers TCA0.SINGLE.PERBUF and TCA0.SINGLE.CMP0BUF. This ensures that the values are automatically copied into the underlying registers at the beginning of the next timer cycle. If

we skip the buffering step, glitches can occur—for example, if the upper limit is decreased mid-count and ends up below the current value of the counter, the `CNT = PER` condition won't trigger, and the counter will race well past the limit, wrapping around to zero only after reaching the 16-bit value limit at 65,535.

Analog Accessories

Another essential and common microcontroller feature is onboard ADCs and DACs. These converters are not the best money can buy, but they're good enough for routine tasks such as temperature and light sensing, battery monitoring, and basic audio.

The ADC in the AVR Dx series uses successive approximation to deliver 10- to 12-bit resolution at a sampling rate up to 130 kHz (sometimes written as "ksps" for *kilosamples per second*). To enable the ADC, first disconnect the corresponding input pins from any digital I/O circuitry. We do this by writing to the familiar `PORTn.PINxCTRL` registers. If the goal is to sample the analog signal on PD0 and use PD7 as a reference voltage, we must set the corresponding *input/sense configuration (ISC)* fields—`PINxCTRL` bits 0 through 2—to 100_2:

```
PORTD.PIN0CTRL = 0b00000100; /* PD0 digital disable */
PORTD.PIN7CTRL = 0b00000100; /* PD7 digital disable */
```

The microcontroller offers a flexible reference voltage subsystem for analog peripherals. To use the PD7 pin as a source of reference voltage for the ADC, write 6 to the `VREF.ADC0REF` register:

```
VREF.ADC0REF = 6; /* ADC reference on pin VREFA (PD7) */
```

If we don't want to supply a separate stabilized voltage, we can set `ADC0REF` to 5, which would reference the ADC to the positive supply rail. That said, this configuration is typically undesirable, as it exposes the ADC to digital switching noise originating within the chip.

The converter itself is configured by writing to the `ADC0.CTRLA` register. The defaults are reasonable, so we can leave them alone and just toggle the enable bit:

```
ADC0.CTRLA = 1; /* Enable ADC */
```

The actual conversion is initialized by setting the `STCONV` flag (bit 0) in the `ADC0.COMMAND` register, then waiting until the `RESRDY` flag appears in `ADC0.INTFLAGS`:

```
ADC0.COMMAND = 1;             /* Start conversion         */
while (!(ADC0.INTFLAGS & 1)); /* Wait for the RESRDY flag */
```

The `while (!(...));` syntax causes the program to loop until the nested condition is met. Once the flag appears in the memory-mapped register, the result can be retrieved by reading `ADC0.RES`.

The built-in 10-bit DAC of the AVR Dx series works in a similar manner. To use the default PD6 output pin, we need to set it for output and then disable the digital circuitry, similar to what we did for the ADC:

```
PORTD.DIRSET   = 0b01000000; /* DAC output          */
PORTD.PIN6CTRL = 0b00000100; /* PD6 digital disable */
```

Then, assuming that PD7 is already configured as an analog voltage reference, we route the signal to the converter and turn on the subsystem by writing to `DAC0.CTRLA`:

```
VREF.DAC0REF = 6;          /* DAC reference on VREFA pin (PD7) */
DAC0.CTRLA   = 0b01000001; /* Output pin enable, DAC enable    */
```

To produce the desired analog output voltage on PD6, we simply write a 16-bit value to the `DAC0.DATA` register. The DAC disregards the four least significant bits, so if we want to use the 12-bit conversion result from the ADC, we need to shift the value four bits to the left:

```
DAC0.DATA = (adc_value << 4);
```

A full tome would be needed to cover all the remaining subsystems of a typical MCU, but we've hit the essentials; the rest follow similar patterns. In Chapter 27, we'll put this knowledge to practical use.

27

HELLO, WORLD



In this chapter, we'll wrap up our discussion of embedded systems by building a few simple microcontroller circuits and running code on them. We'll start with the obligatory "hello, world" example—a blinking LED—and then quickly move to more interesting designs.

Grab a breadboard and follow along if you can. I'll stick to the AVR128DA28 as before, but if you prefer a different MCU, the required changes should be fairly superficial. The datasheet for the device should answer most questions about integrated peripherals; online examples, forums, and blog posts will usually fill in any remaining gaps. If you want to turn to AI tools for help, note that these tools are less proficient with embedded systems and tend to sound confident while giving bad advice—so you'll still need enough knowledge to spot when the chatbot is sending you down the wrong path.

Blinking LED

We'll begin with the rudimentary LED circuit in Figure 27-1. It's not meant to impress, but it's a good way to get comfortable with the toolchain that's used to compile programs and upload them to the chip.

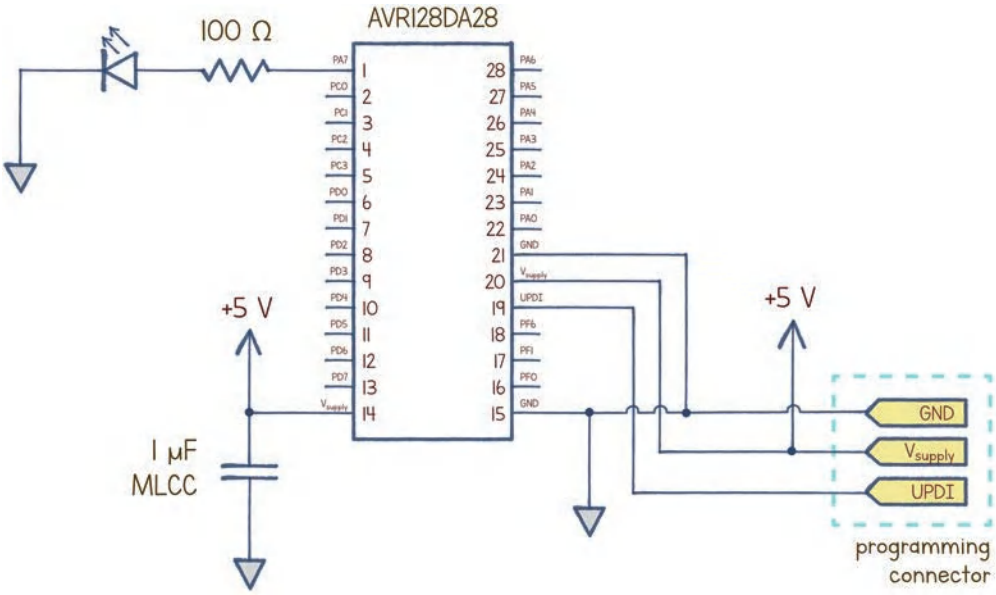


Figure 27-1: About the simplest MCU circuit

In essence, we connect pins 15 and 21 to ground, connect pins 14 and 20 to the positive supply rail (about 5 V), and place a decoupling capacitor across the rails close to the chip. We also connect a single red LED to PA7 (pin 1). The LED resistor is more or less optional, as the chip will not supply more than about 20 to 40 mA on a single pin; nevertheless, it's good practice to include it because the exact current limits of MCU output pins aren't always well-specified.

As discussed earlier, floating input pins on a microcontroller have an indeterminate state. Without a stable reference, this can cause oscillations or excess power draw in the digital input circuitry. Here, we'll address it in software by reconfiguring all unused pins as actively driven outputs.

Of course, before we can start tinkering with the software, we'll need a three-terminal connection for the UPDI programming dongle. The pinout on the programmer side will depend on the dongle you're using, with two common options shown in Figure 27-2.

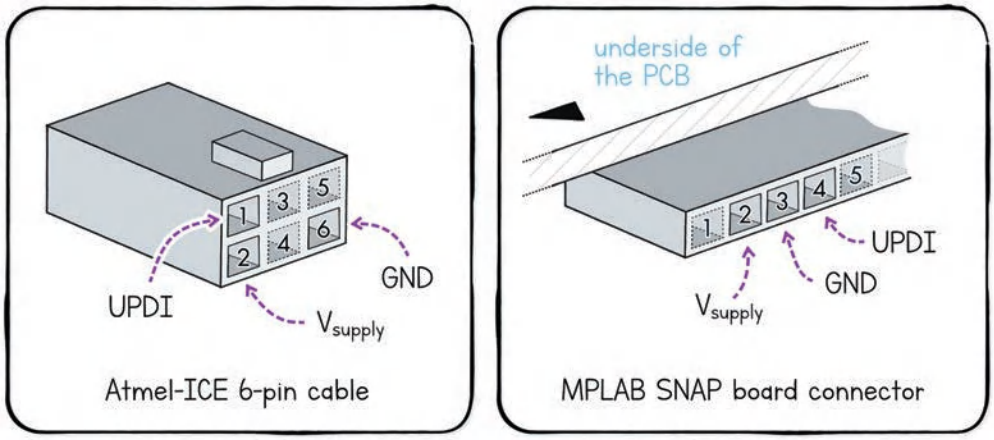


Figure 27-2: Pinouts for two Microchip programming dongles

For the Atmel-ICE, hold the bundled six-pin connector toward you with the tab on top: The UPDI data line (pin 1) is in the top-left corner, the terminal directly underneath (pin 2) is the positive supply rail, and the terminal in the bottom-right corner (pin 6) is circuit ground.

For the MPLAB SNAP, flip the programmer board over and locate the pin that's marked with a small arrow; that's pin 1. The next terminal (pin 2) is the positive rail, the one after it (pin 3) is circuit ground, and UPDI data is routed to pin 4. You'll also need to move a small jumper on the component side from the PIC position to AVR.

The Software Side

With the hardware ready, let's outline the program that will run on the chip. For now, the goal is simply to blink a single LED by changing the signal on PA7 between logical zero and logical one.

To do this at a human-friendly speed, we'll use the built-in `avr-libc` function `_delay_ms(...)`. This function executes a calibrated loop of machine instructions to waste just enough MCU clock cycles to delay the execution by the specified number of milliseconds. For future reference, this is not the most elegant method of implementing delays because it wastes energy; it's usually better to set up a hardware wake-up timer and then suspend or slow down the main MCU clock. Nevertheless, for a quick demonstration, `_delay_ms(...)` will do.

For the function to work, we need to specify the chip's clock speed; the compiler can't guess it ahead of time, as the parameter may depend on the hardware configuration and the actions taken in the code. We specify the clock speed using the compile-time macro `F_CPU`. The default boot-up speed of AVR Dx chips is 4 MHz. This can be increased or decreased by writing to `CLKCTRL.OSCHFCTRLA`, but for this project, the factory setting is fine.

The code we'll upload to the chip looks like this:

```
#define F_CPU 4000000

#include <avr/io.h>
#include <util/delay.h>

int main(void) {

  /* Set all ports as outputs */
  PORTA.DIR = PORTC.DIR = PORTD.DIR = PORTF.DIR = 0b11111111;

  while (1) {
    PORTA.OUTTGL = 0b10000000; /* Toggle PA7  */
    _delay_ms(250);            /* Wait 250 ms */
  }

}
```

For convenience, you can also download it at *https://lcamtuf.coredump.cx/slc/prog1*.

For readers who are still learning C, `main()` is the name for the function (a code block) where program execution begins. Behind the scenes, the MCU always starts executing code from memory at the same location, and the compiler provides a small, invisible startup shim that eventually jumps to the user-supplied block. Another new

concept is the `#include` blocks; these directives tell the compiler to pull in platform-specific files that spell out the meaning of symbols such as `PORTA` or `_delay_ms(...)`.

Programming the Chip

The simplest way to compile the code and program the device is to use the MPLAB X IDE from Microchip:

1. In the IDE, start a new project CTRL-SHIFT-N and stick to the defaults on the first page of the wizard.
2. On the second screen, choose **AVR128DA28** and the correct programming tool, and on the third screen, choose the `avr-gcc` compiler.
3. With the project created, add a new file CTRL-SHIFT-N. Choose **C source file** and paste in the program shown earlier.
4. Power the circuit. Navigate to the Production menu and select **Make and program device**. If the process fails, go back to project settings and make sure you have selected the correct programming tool, verify the connections between the programmer and the MCU, and make sure the correct voltage is supplied to the chip.

When using the open source toolchain, the compilation and upload process takes a bit more work. First, check that you have a reasonably recent version of `avr-gcc` and `avr-binutils` installed on the machine. Next, save the source code as *program.c* and type in the following shell command:

```
avr-gcc -mmcu=avr128da28 -ffunction-sections -fdata-sections \
 -O3 -Wall -Wno-array-bounds program.c -o program.elf
```

The first three options passed to `avr-gcc` are quite important: They specify the target architecture and allow the elimination of unused library code.

If all goes well, you should end up with a binary file named *program.elf*. This file needs to be transferred to the chip using the `avrdude` tool. As before, the tool must be reasonably up-to-date to work with newer programmers and chips. To upload via the Atmel-ICE dongle, use the following command:

```
avrdude -p avr128da28 -c atmelice_updi -U flash:w:program.elf
```

For the MPLAB SNAP tool, the process is a bit more involved. In all likelihood, before reaching for `avrdude`, you will need to upgrade the dongle's firmware by using it with the MPLAB X IDE once (this can be done without connecting the dongle to the MCU). With this out of the way, the upload can be performed in the following way:

```
avrdude -c snap_updi -x mode=avr
avrdude -p avr128da28 -c snap_updi -U flash:w:program.elf
```

Several other low-cost UPDI programmers are also supported by `avrdude`; they can be selected via the `-c` flag. In all cases, successful programming will be signaled by the steady blinking of the MCU-connected LED.

UPDATING THE LINUX BUILD ENVIRONMENT

If you're using Linux and if the compilation step fails with an error about a missing spec file, your `avr-gcc` is out of date. In lieu of a proper upgrade, you can add the missing specs by downloading the Atmel AVR-Dx Series Device Support file from *http://packs.download.atmel.com*. Change the extension of the downloaded file from *.atpack* to *.zip*.

Several files must be pulled from this archive and copied to specific locations on your disk:

1. Move the archived spec file, *gcc/dev/avr128da28/device-specs/specs-avr128da28*, to the existing *device-specs* directory on your system. You can use the `find` command to locate the folder; it likely resides somewhere in */usr*.
2. Move two binary files, *gcc/dev/avr128da48/avrxmega4/crtavr128da28.o* and *gcc/dev/avr128da48/avrxmega4/libavr128da28.a*, to the folder on your system that contains *libatxmega64d4.a*. As before, the location of the folder can be discovered by running `find`.
3. Place the include file named *include/avr/ioavr128da28.h* in the existing folder that contains *iox64d4.h*.

LED Chaser

Next, to make the earlier circuit a bit more interesting, let's construct an eight-pixel display by connecting one LED to every pin on port A (Figure 27-3). To maximize visual appeal, I recommend putting the LEDs in a single, uniformly spaced column or row. As before, the current-limiting resistors are more or less optional if you're experimenting on a breadboard and if there's no one nearby to criticize the design.

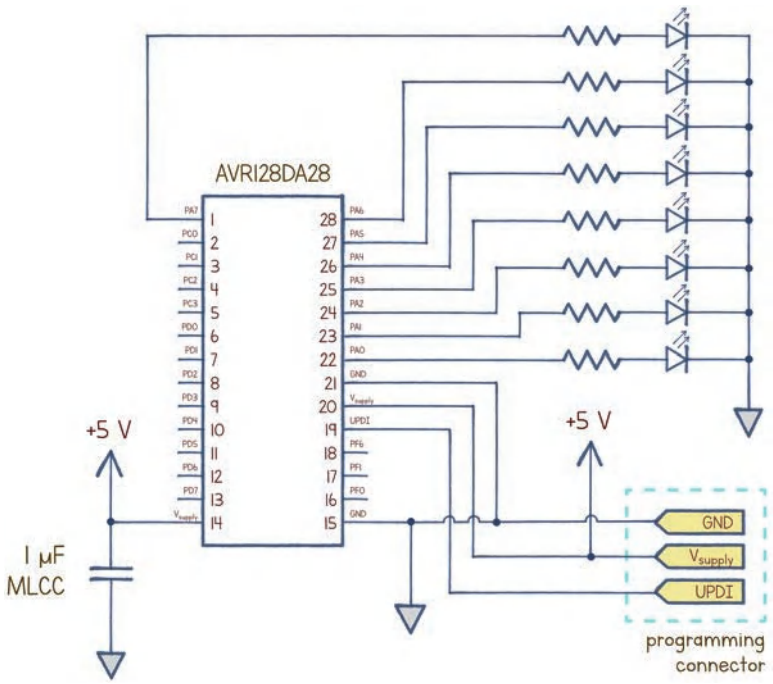


Figure 27-3: The hardware side of an MCU-based LED chaser

The revised program for the microcontroller is as follows:

```
#define F_CPU 4000000

#include <avr/io.h>
#include <util/delay.h>

int main(void) {
  uint8_t counter = 0;

  PORTA.DIR = PORTC.DIR = PORTD.DIR = PORTF.DIR = 0b11111111;

  while (1) {
    PORTA.OUT = (1 << counter);
    counter = (counter + 1) % 8;
    _delay_ms(50);
  }

}
```

The code can be downloaded at *https://lcamtuf.coredump.cx/slc/prog2*.

Let's start with a couple of hints for C newcomers. The `uint8_t` data type is an unsigned 8-bit integer; it can hold values 0 to 255. The `<<` operator shifts the first operand left by the specified number of bits ($1 << 1 = 10_2$, $1 << 2 = 100_2$, and so on). The `%` operator returns the value of the first operand modulo 8—that is, it causes the counter to wrap around to 0 after reaching 7.

If all goes well, the program should produce a rather pleasant running pattern across the impromptu LED display; this is the same effect I mentioned in passing when discussing ring counters in Chapter 21. Unlike a hardware solution, we can tweak the pattern however we like. Try modifying the code to make the light bounce back and forth or gradually speed up!

Push-to-Advance LEDs

Next, let's add an input to the circuit. The simplest approach is to connect a push button or other momentary switch between the PC0 leg (pin 2) and ground, then enable an internal pull-up resistor on the leg. This will keep the corresponding bit in `PORTC.IN` equal to one when the switch is open or to zero when the button is pressed.

With the switch connected, we can take a stab at modifying the earlier LED chaser circuit to make the pattern advance on key press:

```
#define F_CPU 4000000

#include <avr/io.h>
#include <util/delay.h>

int main(void) {
  uint8_t counter = 0;

  PORTA.DIR = PORTD.DIR = PORTF.DIR = 0b11111111;
  PORTC.DIR = 0b11111110;         /* PC0 as input           */
  PORTC.PINOCTRL = 0b00001000;    /* Enable pull-up on PC0  */
```

```
  while (1) {
    PORTA.OUT = (1 << counter);
    counter = (counter + 1) % 8;
    while (PORTC.IN & 1);         /* Wait until PC0 pressed */
  }
}
```

You can get this code at *https://lcamtuf.coredump.cx/slc/prog3*.

Alas, the program will not work properly; see if you can identify the problem yourself. The issue is that as long as the button is held, the wait loop falls through instantly, and the counter is updated at breakneck speed. Instead of an orderly pattern, all the diodes will appear dimly lit at the same time.

A simple fix is to add a second condition that waits for the button to be released before updating the counter. This is the software equivalent of triggering on the falling edge of the input signal:

```
    while (PORTC.IN & 1);    /* Wait for key press   */
    while (!(PORTC.IN & 1)); /* Wait for key release */
```

With this change, the circuit should behave *almost* correctly: If you keep pressing the button long enough, you will probably notice that the program sometimes "skips ahead." The culprit is switch bounce, a phenomenon we discussed in Chapter 9, in which mechanical vibrations inside the switch produce brief on-off pulses that are misinterpreted by the MCU. We can tune out these glitches by adding a small "debounce" delay every time we look at `PORTC.IN`. The revised main loop of the program is:

```
  while (1) {
    PORTA.OUT = (1 << counter);
    counter = (counter + 1) % 8;
    while (PORTC.IN & 1);
    _delay_ms(10);
    while (!(PORTC.IN & 1));
    _delay_ms(10);
  }
```

With this modification, the circuit should work as expected.

ADC Bar Meter

For the next challenge, let's try our hand with the ADC subsystem. We'll connect an analog signal source to PD0 (pin 6) and then repurpose the earlier LEDs to provide a bar meter that measures the input voltage. The initial source can be a 10 kΩ potentiometer connected between ground and the positive supply rail (Figure 27-4).

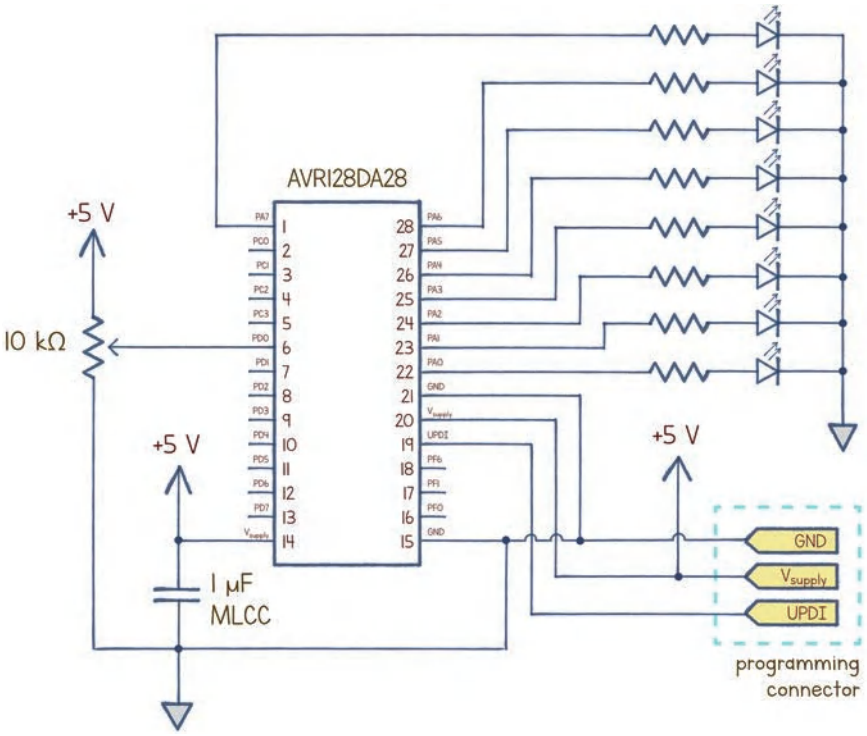


Figure 27-4: The hardware for a simple signal meter

Because the measurement is necessarily coarse—we only have eight LEDs—we'll forgo an external, stabilized reference voltage and instruct the VREF subsystem to connect the ADC reference to the positive supply rail. In all other respects, this code follows the ADC access scheme outlined in Chapter 26:

```
#define F_CPU 4000000

#include <avr/io.h>
#include <util/delay.h>

static void setup_ports(void) {
 PORTA.DIR = PORTC.DIR = PORTF.DIR = 0b11111111;
 PORTD.DIR = 0b11111110;          /* PD0 is ADC in       */
 PORTD.PINOCTRL = 0b00000100;     /* PD0 digital disable */
}

static void setup_adc(void) {
  VREF.ADC0REF = 5;               /* Supply rail reference         */
  ADC0.CTRLA   = 1;               /* ADC enable w/default settings */
}

static uint16_t adc_read(void) {
  ADC0.COMMAND = 1;               /* Start conversion     */
  while (!(ADC0.INTFLAGS & 1));   /* Wait for RESRDY flag */
  return ADC0.RES;
}
```

```
int main(void) {
  setup_ports();
  setup_adc();

  while (1) {
    uint16_t val = adc_read();           /* Read ADC                  */
    if (val) val = (val - 1) / 455;      /* Convert to a range of 0-8 */
    PORTA.OUT = 0b11111111 >> (8 - val); /* Drive LEDs on port A      */
    _delay_ms(100);
  }

}
```

For readability, I split the initialization code and the ADC routine into separate functions. The source can be downloaded at *https://lcamtuf.coredump.cx/slc/prog4*.

At the beginning of the main loop, the program reads the 12-bit value produced by the ADC; the result ranges from 0 to 4,095 ($2^{12}-1$). The next line maps the reading to values ranging from 0 to 8. It does this by subtracting one, dividing the number by 455, and then rounding down to the nearest integer. The `if (...)` statement ahead of this operation guards against decrementing the ADC reading if it's already zero, as that would cause an arithmetic underflow and the value to wrap around to the maximum possible 16-bit integer (65,535). All in all, ADC readings from 0 to 456 are mapped to zero, measurements from 457 to 910 are mapped to one, and so on; the last bin consists of readings between 3,641 and 4,095.

Finally, the program takes a binary value 11111111_2 and shifts it to the right (>>) so that a variable number of bits on the left are replaced by zeros. The extent of the shift is calculated by subtracting the previously calculated value from eight so that the highest range of ADC readings produces zero shift (all diodes lit) and the lowest readings leave all the LEDs off.

PWM Audio Toy

For the chapter's graduation project, let's try to make some noise. We won't be using the digital-to-analog converter; instead, we'll drive a speaker directly with a PWM-generated square wave, producing some gnarly beeps at a frequency proportional to the voltage present on the ADC input pin. The revised circuit is shown in Figure 27-5.

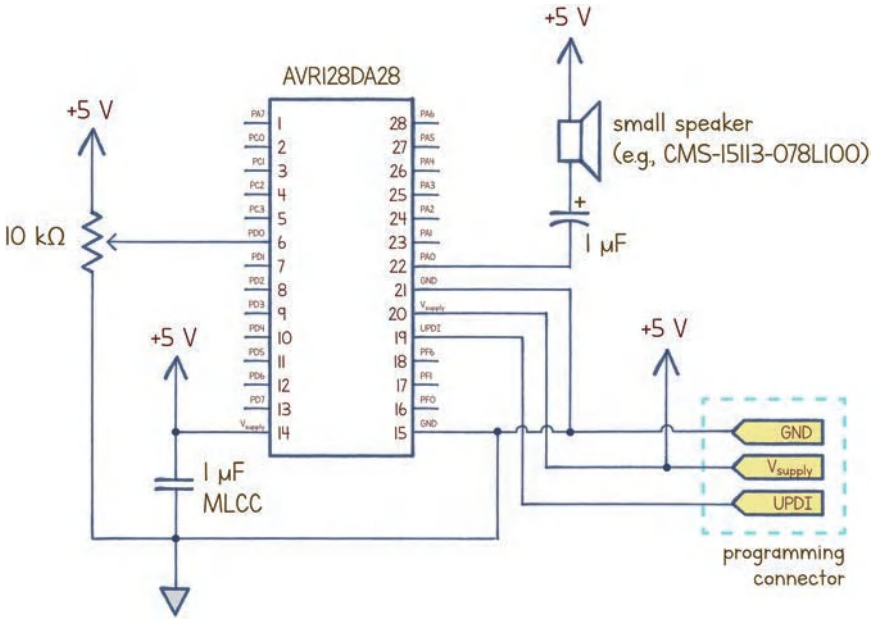


Figure 27-5: A PWM noise-maker circuit

Because the MCU can output only about 20 mA per pin, the audio won't be particularly loud. Nevertheless, the approach works as a quick workshop demonstration and will not harm the chip. You might notice that the other leg of the speaker is connected to the positive supply rail. Because of the AC coupling through a capacitor, it doesn't matter which rail we use, but if you connect it to ground, be sure to reverse the polarity of the series capacitor (unless you're using a non-polarized MLCC).

The code that needs to be sent to the chip builds on the earlier design of the bar meter, with only a couple of new sections related to the PWM subsystem:

```
#define F_CPU 4000000

#include <avr/io.h>
#include <util/delay.h>

static void setup_ports(void) {
 PORTA.DIR = PORTC.DIR = PORTF.DIR = 0b11111111;
 PORTD.DIR = 0b11111110;            /* PD0 is ADC in       */
 PORTD.PINOCTRL = 0b00000100;       /* PD0 digital disable */
}

static void setup_adc(void) {
 VREF.ADC0REF = 5;                 /* Supply rail reference         */
 ADC0.CTRLA   = 1;                 /* ADC enable w/default settings */
}

static void setup_pwm(void) {
  TCA0.SINGLE.CTRLB = 0b00010011;   /* PWM mode, output to PA0     */
  TCA0.SINGLE.CTRLA = 0b00000101;   /* Clock div 4 (1 MHz), enable */
}

static uint16_t adc_read(void) {
```

```
  ADC0.COMMAND = 1;                 /* Start conversion     */
  while (!(ADC0.INTFLAGS & 1));     /* Wait for RESRDY flag */
  return ADC0.RES;
}

static void pwm_tone(uint16_t hz) {
  uint16_t period = 1000000 / hz;
  TCA0.SINGLE.PERBUF  = period;
  TCA0.SINGLE.CMP0BUF = period / 2;
}

int main(void) {

  setup_ports();
  setup_adc();
  setup_pwm();

  while (1) {
    uint16_t val = adc_read();      /* Read ADC                  */
    pwm_tone(100 + val / 2);        /* Tone from 100 to ~2.1 kHz */
  }

}
```

This source is also available at *https://lcamtuf.coredump.cx/slc/prog5*.

In a nutshell, the program sets up the TCA0 counter to run off a 1 MHz clock and output a PWM waveform to PA0. The main loop of the program samples the ADC, maps the 0-to-4,095 reading to a sensible range of audio frequencies (100 Hz to 2.1 kHz), and then passes the calculated value to `pwm_tone(...)`. This function updates the period of the generator while keeping the waveform at 50 percent duty cycle. Because `pwm_tone(...)` is called almost immediately after turning on the circuit, I didn't bother to specify any starting values in the `PER` and `CMP0` registers when initializing the PWM circuitry.

As an exercise for the reader, I suggest modifying the circuit to add a switch and make the instrument play a potentiometer-controlled note only when the button is pressed. Some readers might also find it hilarious to amplify the output signal to produce louder and more annoying squeals.

When you're done making noise, turn the page to dive into Part VI, where we'll explore printed circuit boards and the dark art of production-grade circuit design.

PART VI

BEYOND PROTOTYPES

28

SURFACE-MOUNT COMPONENTS



Up to this point, we've focused on the theory and practice of designing electronic circuits. What we haven't discussed is how to take those ideas beyond the prototype stage. In Part VI, we'll bridge that gap: We'll cover how to design professional-looking printed circuit boards and build electronic gadgets that can withstand daily use.

Before diving into PCB design principles, it's prudent to say a few words about surface-mount technology. If you've flipped ahead to the appendix or tinkered with electronic circuits before, you know that the industry once relied predominantly on through-hole components. Parts made with this technology had prominent leads inserted into matching holes on the circuit board. Today, we do things differently: Small electronics rely almost exclusively on surface-mount devices attached to exposed pads on the PCB.

Many components are still available in through-hole packages—a handy form factor for experimentation—but surface-mount parts are a wise choice if you're planning to build the circuit on a mail-order PCB. Shopping for this type of device reveals an interesting problem, though: There's just too much variety. Through-hole components usually came in one size with leads on a 2.54 mm (0.1 inch) grid. In contrast, surface-mount technology offers hundreds of form factors to choose from.

Making the wrong choice can ruin your day, as some of the devices can be difficult to solder by hand or may have electrical properties that differ from what you expect. To make sense of this mess, let's review some of the options you might see.

Common Two-Terminal Components

Surface-mount resistors, ceramic capacitors, chip inductors, and diodes typically come in rectangular packages with a 2:1 aspect ratio and solderable caps on the short sides. A visual comparison of four common sizes of these components is shown in Figure 28-1.

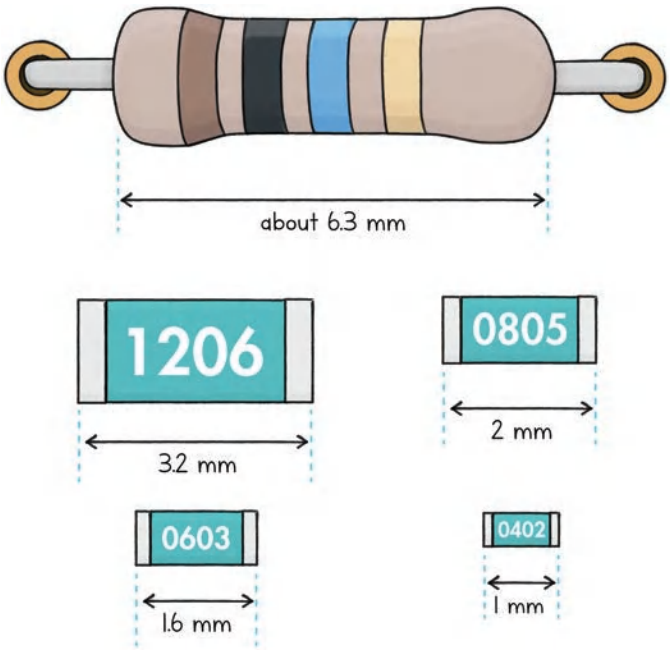


Figure 28-1: Relative sizes of a standard ¼ W through-hole resistor (top) and its 1206, 0805, 0603, and 0402 surface-mount counterparts

Let's start with the 1206 package, also known as 3216 metric. It measures 3.2×1.6 mm and is about half the size of a standard through-hole resistor. This form factor is easy to solder by hand even for beginners. The only requirement is good tweezers to hold the component in place, as the parts are lightweight enough to stick to the tip of the soldering iron. Components in this size are a good choice if you're just learning PCB design, plan to solder it yourself, and want to keep the process as simple as possible.

The next common package size is 0805 (2012 metric, 2.0×1.25 mm). These components are markedly smaller, but with a bit of practice on scrap PCBs, you should still be able to solder them with ease. The package is a good default choice for most hobby designs. It's small but reasonably easy to handle, and the components still have reasonably good electrical specs.

Another frequently encountered option is 0603 (1608 metric, 1.6×0.8 mm). This package has one-fourth the board footprint of 1206. It's small enough to fly away if you sneeze, and it exhibits a fair amount of static cling, so the components are easy to lose. This form factor is mildly challenging to solder manually; practice and a good soldering station are essential.

The smallest size most hobbyists can reliably handle is probably 0402, also known as 1005 metric (1.0×0.5 mm). These components benefit from a good desktop magnifier or an inspection microscope. You'll see this size on smaller or more densely packed boards, including computer motherboards. Even smaller passives exist—for example, 01005 (0.4×0.2 mm)—but they challenge even pick-and-place machines used

for robotic PCB assembly and are usually reserved for specialized uses such as wearables and smartphones.

As hinted at earlier, beyond soldering ease, smaller components often have worse specs. For example, 1206 resistors can typically dissipate ¼ W, but the limit for 0805 may be just ⅛ W, and 0402 might only be able to handle 1/16 W. For MLCCs, smaller packages may mean lower maximum capacitances, lower voltage ratings, and more distortion caused by dielectric saturation (Chapter 8). On the plus side, physically smaller passives have less parasitic inductance—the signal path is shorter—so they might be preferred for high-speed digital applications, especially for data buses running at hundreds of megahertz.

Transistors and Friends

Standard surface-mount transistors frequently come in the SOT-23 (TO-236) package, followed by SOT-223 (SC-70) and SOT-523 (SC-90). "SOT" stands for *small outline transistor*. These packages appear in the bottom right of Figure 28-2.

SOT-23 is a good default choice. It's roughly comparable in size to 1206 and is about as easy to solder by hand. The SOT-523 form factor, in contrast, is comparable with 0603 passives and can be more difficult to deal with. I'd reach for it only if you absolutely need to cram a lot of transistors on a small PCB.

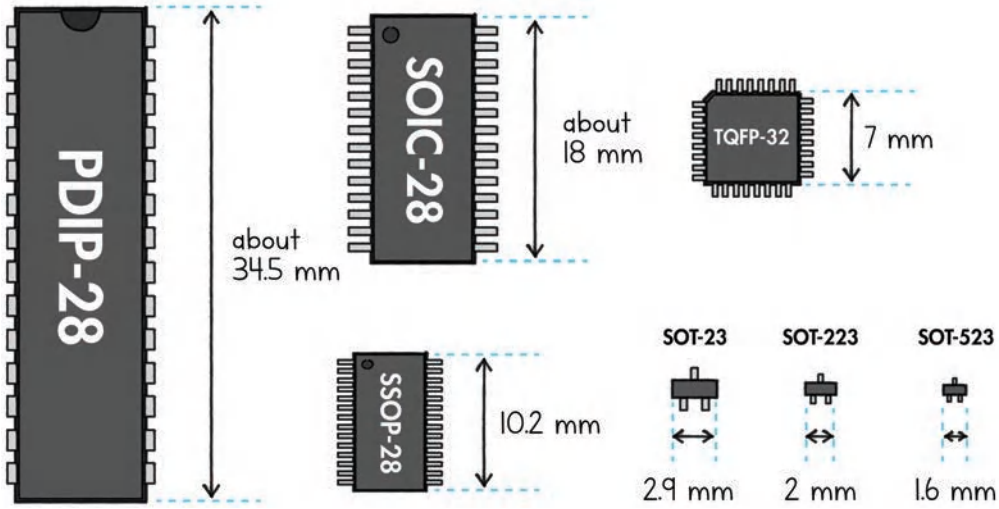


Figure 28-2: Relative sizes of a typical through-hole microcontroller; its surface-mount SOIC-28, SSOP-28, and TQFP-32 counterparts; and two common transistor packages

High-power transistors may come in larger packages with thermal pads that must be soldered to the PCB for efficient heat dissipation; an example is TO-252 (DPAK), which measures about 6.5×7 mm.

Variants of SOT-23 are also sometimes used for diodes, small integrated circuits such as voltage regulators, or individually packed op-amps (in which case two extra pins are added, producing what's known as SOT-23-5). Most chips, however, have a more recognizable appearance, and that's what we'll cover next.

Integrated Circuits

The direct counterpart to through-hole ICs is the *small outline integrated circuit (SOIC)* form factor, shown in the top center of Figure 28-2. In SOIC devices, the pins are

placed on a 1.27 mm grid—half the distance used for through-hole parts—and are fairly easy to tame in hobby soldering, but some attention needs to be paid to avoid bridged pads. It boils down to practice, a good soldering iron, and a generous use of flux. As noted in the appendix, your workshop will also benefit from a roll of solder wick in case you need to fix an accidental bridge.

Another set of common monikers you'll encounter is SSOP, TSOP, and MSOP; the main part of the acronym stands for *shrink small outline package*. An example of an SSOP chip is shown underneath the SOIC illustration in Figure 28-2. Compared to SOIC, the main difference is reduced pin spacing, often around 0.65 mm (one-fourth of through-hole). The devices are moderately challenging, in part because you need good eyesight (or some form of magnification) to ensure proper pad alignment and to spot bridged pins. That said, seasoned hobbyists can handle SSOP chips without breaking a sweat.

Integrated circuits with higher pin counts, including many high-performance 32-bit microcontrollers, may also come in square packages with leads on all sides. This package type is commonly known as QFP (*quad flat package*), TQFP (*thin quad flat package*), or LQFP (*low-profile quad flat package*); an example is shown in the top-right corner of Figure 28-2. For this class of devices, pin counts typically range from 32 to 144 and lead spacing may vary from 0.8 to 0.4 mm. Especially on the 0.4 to 0.5 mm end, soldering QFP parts requires magnification and a good amount of practice on scrap circuit boards. It's a skill you can learn, but few are successful on the very first try.

Some chips come in packages with no protruding leads; these form factors often have names ending with *-N*, for "no leads," but might also use other acronyms, such as WLP. In general, any IC package that doesn't sound like a riff on SOP, SOIC, or QFP should be treated with some suspicion. The problem is that many leadless packages can't be easily hand-soldered. Instead, they're meant to be installed by precisely applying solder paste to exposed PCB pads, placing the chip on top, and then using a temperature-controlled oven or a hot-air gun to produce an electrically conductive bond. The process can be carried out at home, but results are difficult to inspect or repair. In other words, if you have a choice, leaded packages are typically a better pick.

To be fair, some quad-style leadless packages have a small portion of their mounting pads exposed on the sides of the package, allowing these components to be hand-soldered when paired with a special, extended PCB footprint. The soldering technique involves touching the PCB-side pad with a soldering iron some distance away from the chip, which should cause solder to wick and form a robust bond. I wouldn't recommend this for your first projects, but the technique is worth keeping in the back of your head.

Odds and Ends

Even in the era of surface-mount components, not everything is standardized. Some devices, such as power inductors, connectors, or switches, may come in bespoke form factors that may need to be reverse-engineered from technical drawings in the datasheet. That said, the earlier observations about part size, lead spacing, and leadless components are broadly applicable. Before long, you should be able to navigate these choices with ease.

It's also worth noting that *some* through-hole parts may still make sense in modern designs. For one, this method of installation is mechanically stronger, so it might be worth using for audio jacks, toggle switches, potentiometers, and other components that could be accidentally pulled off the PCB. For some other components, there's

no size or cost advantage to surface mounting. A good example is larger electrolytic capacitors: Leads or no leads, the component is still a big metal can.

We have covered the basics of surface-mount technology. Next, let's look at laying out these components on your very own PCB.

29

THE BASICS OF PCB DESIGN



In this chapter, we'll go over the process of designing a simple PCB. We'll cover enough ground to construct any of the analog or digital circuits discussed in the book.

Before we begin, it's worth reflecting on how much the world of DIY electrical engineering has changed over the past 15 years or so. In the early days, most hobbyists could only dream of designing a professional-looking PCB. The most practical option was to buy blank copper-clad boards, draw a pattern of traces and pads on top, and then dunk the boards in a caustic solution to remove metal from the areas not protected by the hand-drawn mask.

Today, we can choose from a number of mail-order PCB manufacturers that accept one-off orders from retail customers, combine them into large panels, then cut the boards apart and ship them to your door. Figure 29-1 shows an example of a custom-made PCB for a handheld video game.

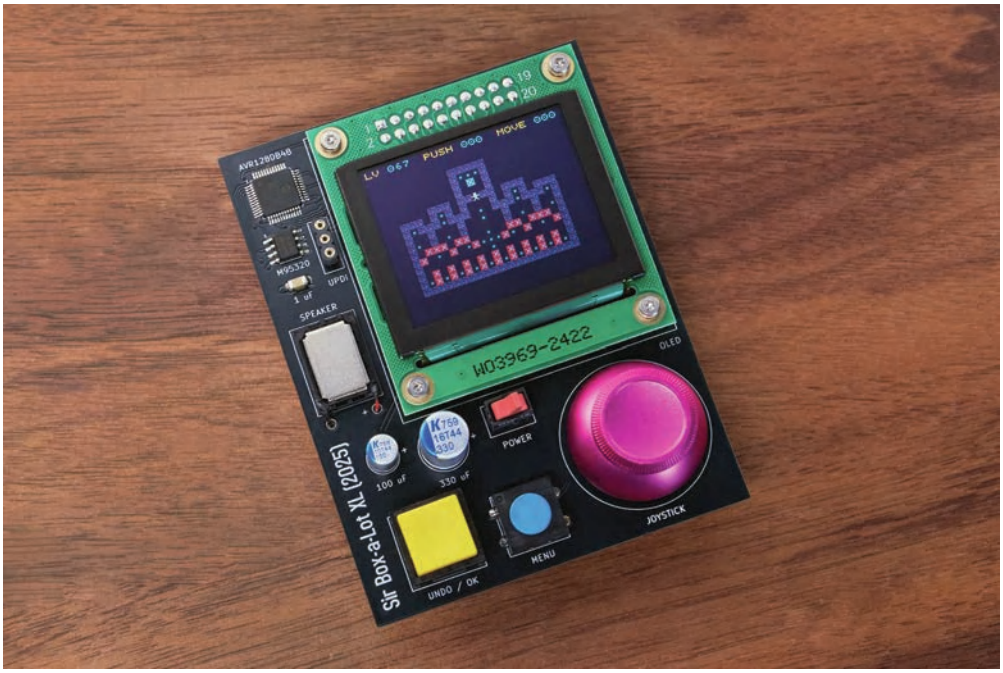


Figure 29-1: Sir Box-a-Lot*, a modern version of a 1980s game called* Sokoban*. You can read more about the project at* https://lcamtuf.coredump.cx/sir-box-a-lot.

Because the manufacturing process is largely automated, the boards are remarkably inexpensive. Best of all, we can instruct the vendor to embellish the board with silkscreen graphics or text of our design. The silkscreen layer goes on top of a tough, nonconductive coating (*solder mask*) that improves the board's appearance and reduces the likelihood of solder bridges or other accidental shorts.

Hobby PCB Vendors

Most major PCB factories in the United States and Europe prefer to work with commercial buyers or government clients. They will take a retail order if you insist, but they will likely quote $500 or more for even the simplest job.

The hobby niche is served primarily by a number of Chinese companies based in the high-tech region of Shenzhen. The largest and most dependable is JLCPCB (*https://www.jlcpcb.com*), closely followed by PCBWay (*https://www.pcbway.com*). Their capabilities and pricing differ somewhat, but for simple orders, you can't go wrong with either. Typical factory turnaround is one to two days, plus three to five days for expedited shipping via DHL. For US customers, it's not unusual for boards to arrive within a week. Even factoring in recent tariffs, the pricing is exceptionally good—it works out to about $4 each for the mid-size board shown in Figure 29-1 if you order 10 pieces or more.

If you want to support domestic manufacturers without going broke, a good option is OSH Park (*https://www.oshpark.com*). The company follows an unusual business model. Instead of making the boards, it consolidates retail orders and ships them in bulk to a big US-based manufacturer. Working through an intermediary means the process takes longer, costs more, and gives you less control over the result. Nevertheless, it's a reasonable choice for prototypes—using the earlier example from Figure 29-1, you'd pay $15 per board with a minimum order of three boards. Typical turnaround time is about two weeks.

Readers in the European Union might want to try Germany-based Aisler (*https://www.aisler.net*). Its pricing is better than OSH Park's, and the quality is excellent. The company also takes orders from the United States, although in my experience, its overseas shipping method is pretty slow.

Your First PCB

Now that we've sorted out the manufacturing landscape, let's go through the steps of designing a simple printed circuit board. The standard PCB design tool for hobbyists and budget-conscious businesses is the open source program KiCad (*https://www.kicad.org*). Several alternatives exist, such as the free LibrePCB (*https://www.librepcb.org*) or the relatively inexpensive DipTrace (*https://www.diptrace.com*), but we'll stick with KiCad because it's well known and supported by virtually every PCB manufacturing shop.

The most off-putting aspect of KiCad is its clunky user interface and a few unconventional design decisions—starting with the use of the F1 and F2 keys to zoom in and out (on a desktop computer, you can also use the mouse scroll wheel). Once you get past the interface quirks, though, the tool is quite capable and easy to use.

Schematic Capture

There are two ways to approach PCB design. One is to start with a schematic capture tool and use the schematic as the ground truth for the parts list and the connections between physical pins. Another is to forgo schematic capture entirely and lay out components directly on the PCB. The optimal approach will depend on the project at hand and your working style, so try both methods and see what feels right.

The advantage of the first approach is that you get an unattractive but functional schematic for free and reduce the chance of errors creeping in down the line. You still need to lay out parts and route traces, but the software will flag problems if you attempt an invalid connection or deviate from the schematic.

The downside is redundancy when the drawing already exists in a non-machine-readable format, or when the circuit is simple or repetitive enough that the schematic is a waste of time. Moreover, KiCad's built-in part library is rather limited, so you may spend time hunting for part definitions online or creating them manually in the Symbol Editor. For this workflow to function properly, the software requires a fully specified part—you can't just draw a generic black-box IC and call it a day.

In KiCad, you access the schematic capture tool by creating a new project and clicking the Schematic Editor tile on the main screen. The module is straightforward: Begin with a blank worksheet, place components from the parts library by pressing A, add wires with W, place junctions with J, and insert annotations with T. To pan, hold the right mouse button and drag the cursor. To edit text or component designators, press E. To modify the underlying schematic symbol in the Symbol Editor, use CTRL-E. You can also use L to place net labels—this can reduce clutter but at some cost to readability, a caveat we explored in Chapter 11.

Board Setup and Layer Navigation

The actual board design process takes place in the PCB Editor module. The default settings in KiCad are reasonable for a standard two-layer board, but I recommend opening the **File ▶ Board setup** dialog, going to the **Design rules ▶ Pre-defined sizes**

tab, and entering a few standard trace widths. This lets you quickly switch between them when routing traces. I suggest adding 0.25 mm, 0.5 mm, 1 mm, and 2 mm.

To the right of the PCB Editor window you'll see a list of board layers in the awkwardly named Appearance Manager. The list may seem overwhelming, but there are only five entries to focus on for now: F.Cu represents the top copper layer, F.Silkscreen holds the decorative text and graphics, and B.Cu and B.Silkscreen serve the same purposes for the back of the board. You can toggle between front and back by pressing the minus key.

The fifth important layer is Edge.Cuts, where you define any closed shape that serves as the board outline. The shape is usually rectangular, but you can draw nearly any polygon using line drawing tools in the palette next to the Appearance Manager. If you place a second closed shape inside the outline, it will be interpreted as a cutout. The outline and cutouts are routed by a CNC mill, so shape options can be limited—check your PCB shop's website for acceptable geometries.

In any case, let's start with something basic. Select **Edge.Cuts** and use the rectangle tool to draw a board outline. You can review the board in 3D by pressing ALT-3. This pulls up the 3D Viewer tool with a real-time render of the PCB. Use your mouse to rotate and zoom in on the model. Once you have the outline in place, we can move on to the next step.

Placing Footprints

How you begin placing parts depends on your workflow. If you started with a schematic, choose **Tools ▶ Update PCB from schematic**. This drops the component footprints into the PCB Editor and lists any potential issues, such as missing or invalid footprints.

If you're designing ad hoc, begin with a clean slate and press A to insert footprints from the library. In this mode, you don't need full part specifications; generic options, such as "SOIC- 8_3.9x4.9mm_P1.27mm" for standard SOIC-8 chips, work fine.

Most of the time, we put all the components on the same side of the PCB—usually the front—but you can flip a part to the back by pressing F.

If you can't find a suitable footprint, pick a similar one, add it to the board, and edit it by pressing CTRL-E. You can also create new footprints in the Footprint Editor by selecting **File ▶ New footprint** and saving the result to a custom library. Fundamentally, a footprint isn't much more than a couple of solder pads placed in the right spots on the F.Cu layer, possibly with some shapes or text drawn on F.Silkscreen. You can change the shape of a surface-mount pad or switch it to through-hole by selecting it in the Footprint Editor and pressing E. When creating new footprints, heed the old saying that it's best to measure twice and cut once. Double-check the dimensions or print the design at 1:1 scale to verify fit.

Nets and Rough Layout

Once all the footprints are in place, you can zoom in on their pins. If you imported a schematic, pins should be automatically assigned to a net (a set of connected nodes); this will be represented by an auto-generated name displayed below the number of each pad. If you started with an ad hoc PCB design, you will need to go through the pins of each component, selecting them one by one and pressing E to assign them to a new or existing net. You can name the nets however you want, but it's best to keep the names descriptive: "GND," "LCD D0," and so on.

Once the assignments are in place, KiCad will draw thin, straight lines that represent the missing traces on the PCB (Figure 29-2). This is known as the ratsnest view and it's what will guide you through the routing process down the line.

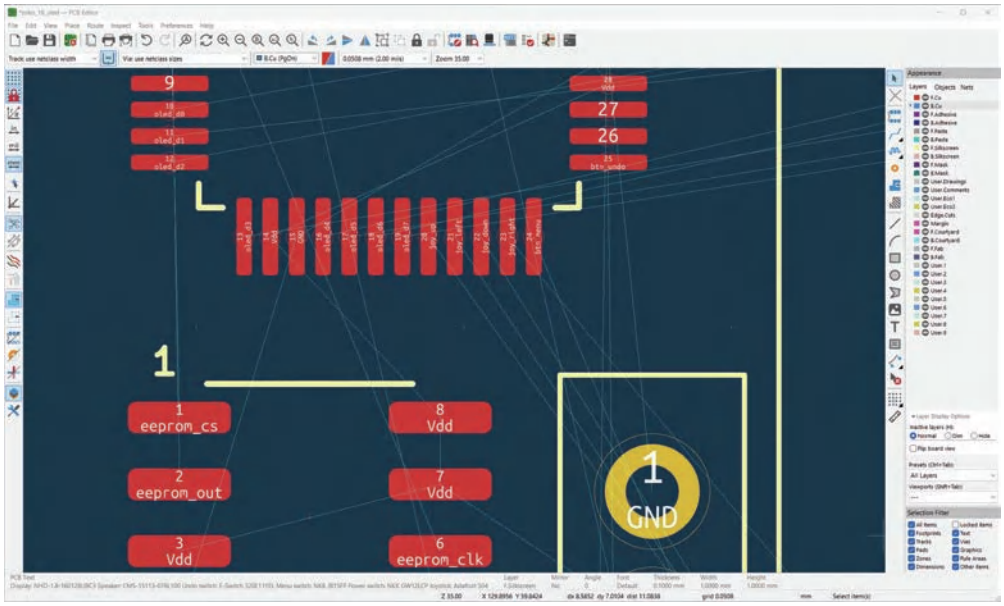


Figure 29-2: A ratsnest view of a PCB

This is a good time to take an initial pass at positioning components on the board. If the board needs mounting holes, switches, or LEDs in specific locations, start there. Otherwise, there are no strict rules for layout, but keep signal and power routing in mind. For example, if the circuit's centerpiece is a microcontroller, it often makes sense to place it near the center of the PCB and arrange the other components close to their corresponding I/O pins. Rotating parts with R can help tidy the layout. As you do this, watch the ratsnest lines—the fewer intersecting paths you have, the easier routing will be.

Routing the Traces

At this point, we can start routing copper traces between components. A major trade-off in this process is that narrow traces conserve space and allow denser layouts, but they also have higher resistance. For reference, a typical 0.25 mm (hairline-thin) connection that travels 20 cm (7.8 inches) will measure about 360 mΩ, enough to produce a voltage drop of about 100 mV for a supply current of 300 mA. The resistance of a trace is proportional to length and inversely proportional to width; a good multi-unit calculator can be found at *https://www.omnicalculator.com/other/pcb-trace-resistance*.

To avoid problems, keep traces as short as possible, especially for power lines or high-speed signals. Further, any line carrying significant current should be sized appropriately across the bulk of its length. It can narrow briefly near surface-mount chips; as long as the narrowed-down section is comparatively short, it's not a big deal.

You can start drawing a trace by selecting a pad and pressing X; repeatedly pressing W will let you cycle through the predefined widths. While drawing, press the BACKSPACE key to delete the last segment or V to place a *via*—a copper-plated hole that lets you hop between the front and the back of the PCB. By default, a trace inherits the net of the pad it originated from; KiCad will highlight acceptable routing targets on the same net and will refuse to cooperate if you attempt to make a connection between unrelated nets.

If you need to edit existing traces or move around footprints, there are two useful keyboard shortcuts to know about: U selects the remainder of the trace after clicking

a single segment, and D lets you move a component without breaking its connections—unlike the default click-and-drag action. KiCad can't always figure out a good way to do it, but the feature is worth trying.

An example of a routed board with a number of digital data lines and vias is shown in Figure 29-3.

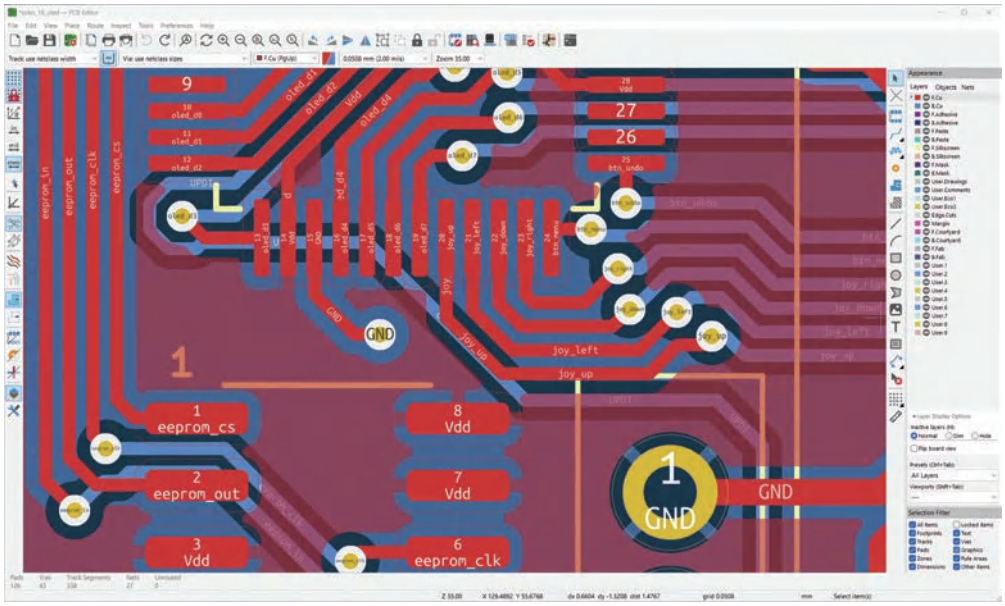


Figure 29-3: A finished PCB layout

Once all the traces are laid out, take a critical look at your work. It's rare to get everything perfect on the first try. If you're unsure whether the design can be improved, make a copy of the project, try a different approach, and revert to the original if the new strategy doesn't pan out. Once again, the 3D Viewer tool can be useful for getting a sense of whether the board looks right.

Design Validation

PCBs are not expensive, but they take a long time to arrive. It's never fun to wait one to two weeks only to find out you've made a trivial mistake.

The most important troubleshooting tool at your disposal is the Design Rules Checker (DRC). In KiCad, you'll find it under the Inspect menu. The DRC will alert you to violations of manufacturing constraints, such as unrelated traces being too close together. More importantly, it also reports missing connections between pads that should share the same net. Figure 29-4 shows an example of the DRC catching an error before fabrication.

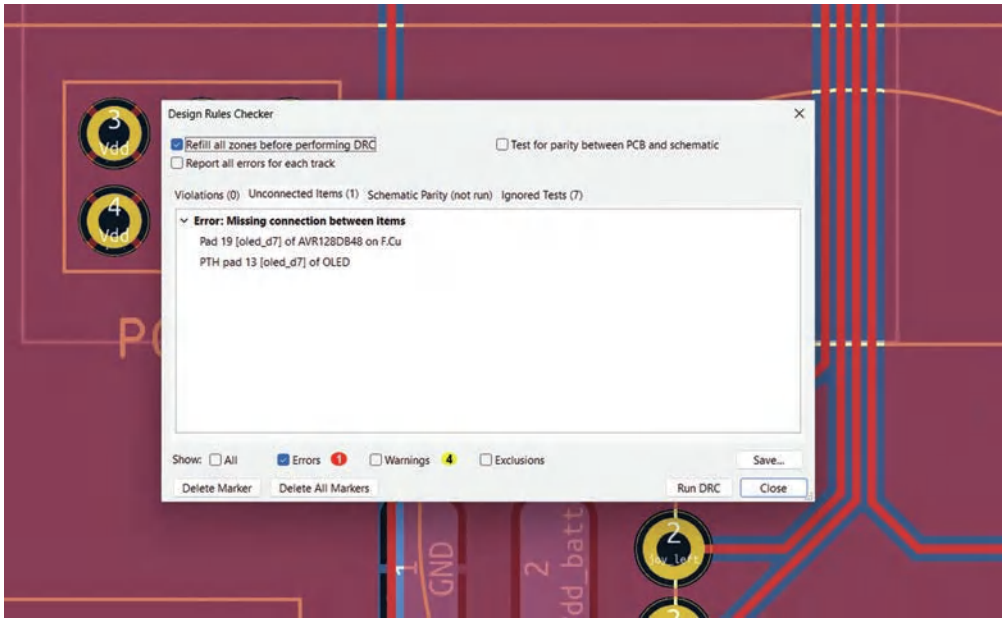


Figure 29-4: A missing connection spotted by the Design Rules Checker

It's also a good idea to verify all pin assignments against the datasheet. Last but not least, I recommend pulling up the 3D Viewer ALT-3 one last time to confirm that nothing sticks out or overlaps and that each part's form factor is correct.

Production Files and Process Choices

Once you're satisfied with the results—that is, when the board looks right and passes all checks—select **File ► Fabrication Outputs ► Gerbers**, specify a new output directory, then click **Plot**. Next, without closing the dialog, choose **Generate drill files**. You should now have a directory containing about a dozen text files. Zip the contents and send the archive to your chosen PCB fabricator.

In the web-based ordering workflow, you can leave the default settings in place; they represent the most common and most affordable manufacturing options. The only parameters worth adjusting are the board quantity and solder mask color. When ordering from JLCPCB or PCBWay, I recommend selecting the lead-free HASL process. By default, exposed pads are plated with leaded solder to prevent oxidation; the lead-free option costs only slightly more.

Most manufacturers will review each first-time order. They will not check whether your circuit is designed correctly, but they will flag manufacturing issues, such as traces placed too close to board edges or holes too small to drill. The process is friendly and fast; if there are problems, they'll notify you and give you a chance to upload a revised file.

If you can spare $50 or so, I encourage you to put the book aside for a couple of hours, put together a simple design in KiCad, and then send it to a fabrication service. The experience of getting your own board in the mail and trying it out is hard to beat.

Next, we'll look at a few more advanced PCB design rules that don't matter for simple circuits but come into play in radio communications and high-speed digital I/O.

30

HIGH-SPEED SIGNALS



Digital chips can simplify circuit design, but they're also a source of headaches. They switch output voltages so abruptly that power supplies can't keep up with the inrush currents, causing voltage spikes and dips. Fast-changing digital signals also create electromagnetic fields that can easily induce the motion of charges in nearby conductors, leading to signal integrity issues and radio interference.

To manage these effects, we can tap into a range of strategies—from software hacks to PCB layout techniques. You don't have to follow these practices religiously, but they're worth knowing. You'll notice that industrial designers take them as gospel. This is because many products sold in developed countries must pass electromagnetic interference (EMI) testing before hitting the shelves; no one likes flunking an expensive certification and having to redo a board.

What's "High Speed"?

In the real world, perfect square waves don't exist; voltage takes time to ramp up or down. In budget microcontrollers such as the AVR Dx or PIC32CM, typical rise and fall times for digital output lines are around 20 nanoseconds (ns), producing a slope comparable to a sine wave running at a frequency of 15 MHz. In higher-performance chips like the PIC32CZ or STM32H7, the value is closer to 2 ns, which translates to a 150 MHz sine wave. From this perspective, a high-speed signal is any signal with a fast rise time. Even an MCU that outputs a 1 Hz square wave will generate brief high-frequency artifacts twice per waveform cycle.

There's also another realm beyond what this book covers: At extreme signal speeds, familiar circuit abstractions begin to break down. In the microwave realm,

rise times are measured in picoseconds and frequencies in gigahertz. At that point, standard capacitors lose their effectiveness and the PCB substrate itself becomes integral to circuit behavior. It's a can of worms most readers will never need to open. But if you envision designing 5G radios or PC motherboards, Rick Hartley's YouTube videos are well worth your time.

On Picking Your Fights

Many of you probably haven't watched the 1983 techno-thriller *WarGames*, but you might be familiar with its famous line from the sentient computer WOPR: "A strange game. The only winning move is not to play."

In digital circuits, the surest way to dodge problems caused by rapid signal switching is to avoid such switching in the first place. If you don't plan to transmit data or toggle peripherals at breakneck speeds, many MCUs offer a software-based method for limiting the slew rate of output pins. For example, on the AVR Dx series, this is done by setting the `SRL` field (bit 0) in the `PORTn.PORTCTRL` register. The setting roughly halves the maximum attainable signal rate, but it also dramatically reduces switching noise. Figure 30-1 illustrates this effect.

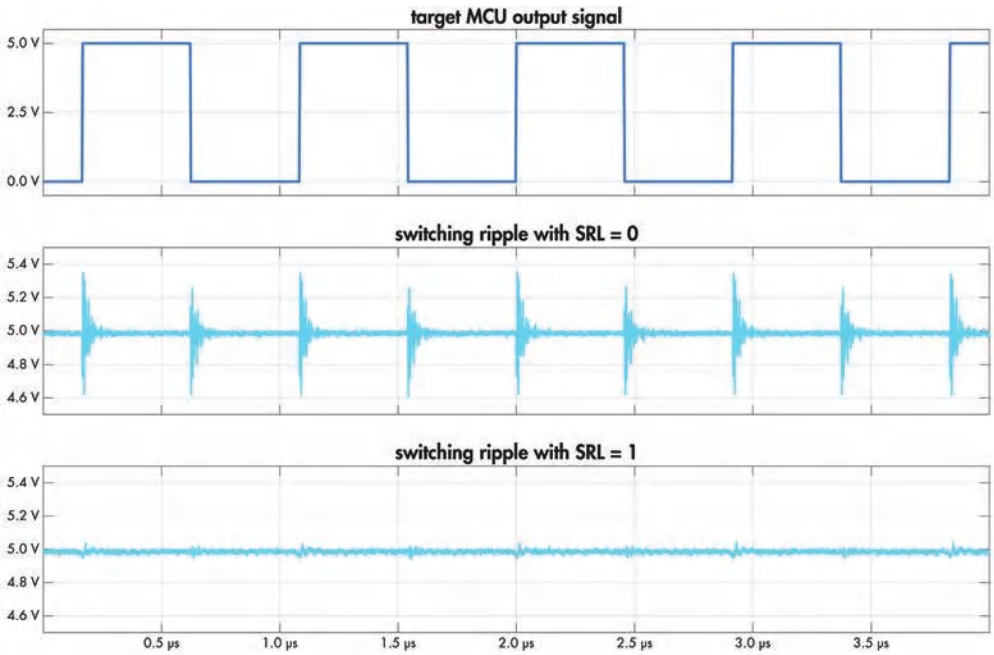


Figure 30-1: Switching-caused ripple near the supply pins of an AVR128DA28 chip

If a software setting is unavailable or insufficient, another trick is to place small surface-mount ferrite beads (Chapter 8) in a series with the signal and close to the originating chip. Similar to a conventional inductor, a bead will resist sudden changes in current, forcing the signal to ramp up more gradually. In a pinch, a series resistor (around 100–200 Ω) can serve the same purpose, but it will have a more detrimental effect on attainable data rates and signal drive capabilities.

If fast switching can't be avoided, we can reduce its impact by timing it carefully. For example, it's best to put data transfers on hold while operating sensitive circuitry such as an ADC. Another last-resort technique is to vary the digital switching frequency

slightly—for example, by adding some jitter into bus clock signals. This helps spread the noise around instead of concentrating it into one constant-frequency blip.

Supply Decoupling

As mentioned in earlier chapters, every digital chip should have at least one ceramic decoupling capacitor per power domain, typically around 1 µF. A chip might function without one, but it's not a risk worth taking given that the component we're adding costs less than a cent.

Without a capacitor, digital switching will inevitably destabilize the supply voltage around the chip. This will happen even if the power supply comes with a beefy output capacitor. The inductance of PCB traces and cables between the supply and the IC prevents the IC from being able to source inrush currents quickly enough. The resulting voltage fluctuations may cause problems for the digital chip itself or for other nearby components.

For optimal performance, the capacitor needs to be mounted as close as possible to the chip's supply pins—ideally within 1 cm (0.4 inches) or so. Figure 30-2 shows the voltage fluctuations measured near an MCU's supply pin as it repeatedly toggles an output port. The first oscilloscope trace was captured with no decoupling capacitor in place. The second trace shows the fluctuations with a capacitor placed about 15 cm (6 inches) away. The third illustrates the improvement when an MLCC is placed close to the supply pins.

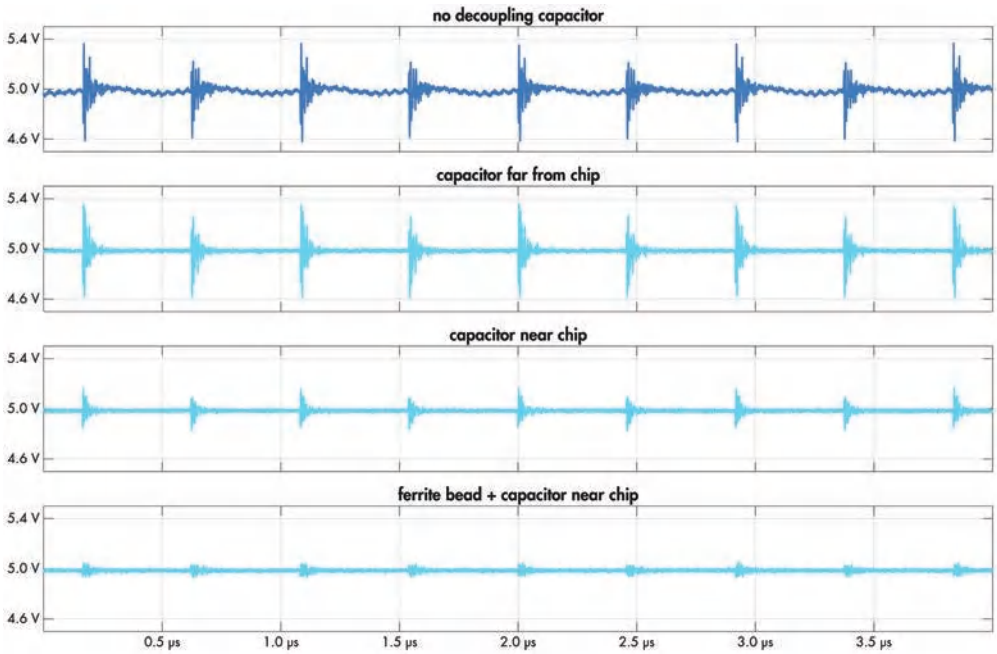


Figure 30-2: Various decoupling scenarios for the AVR128DA28

The fourth and final oscilloscope trace shows an enhanced decoupling scheme from the end of Chapter 12: a series ferrite bead followed by a decoupling capacitor placed close to the chip. The filtering outperforms a standalone MLCC because ferrite beads respond better than capacitors at high signal frequencies. The bead doesn't eliminate noise—it merely contains it on the MCU side. A capacitor on the IC side is still necessary to stabilize the supply voltage for the chip.

Current Return Paths

In Chapter 8, I discussed common-mode chokes at length. These devices consist of two conductors wrapped around a ferromagnetic core. If the currents flowing through the conductors are unbalanced, the windings generate a net magnetic field that the core absorbs. The choke then acts like a conventional inductor, storing energy in the internal field and attenuating AC signals proportionally to frequency. If the currents are equal in magnitude but opposite in direction, however, the net field acting on the core is zero. With no possibility of magnetization, the component's reactance remains low, allowing AC signals to pass unimpeded.

The emphasis on the component might have seemed odd, but the same phenomenon can be observed on a more modest scale if two conductors are placed next to each other on a PCB; the board substrate exhibits some response to magnetic fields too. If we construct a signal loop in which the forward current path and the return path are far apart, the fields around the traces will not cancel out, so more energy will be soaked up by the surrounding material or radiated into space. The inductance of a loop that circles a midsize PCB can be as high as 500–800 nH—equivalent to an inductive reactance of 60–100 Ω at 20 MHz.

Yet, if we modify the layout so that the forward and return paths are on top of each other on the opposite sides of the PCB, we'll end up with the equivalent of a common-mode choke. Balanced alternating currents in the loop will be able to flow more easily, with the reactance dropping as much as tenfold compared to an O-shaped loop.

This property can be used to our benefit, but it can also cause headaches. Figure 30-3 shows a problem scenario in which the return current might not take the intended path.

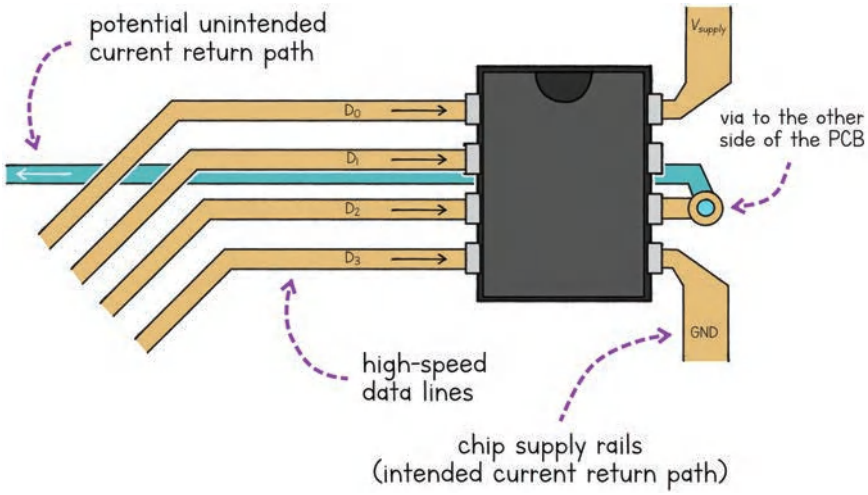


Figure 30-3: The lowest-impedance return path may differ from what was intended.

The digital signal provided on the four data lines will repeatedly charge and discharge the gates of input-side MOSFETs inside the chip. As with any capacitive structure, this process involves charge movement both in the signal line and in the chip substrate. These charges follow the easiest path. At low frequencies, that's typically the path of least resistance through a supply rail. But at high frequencies, inductive reactance may make a meandering supply rail less attractive than an unrelated signal path that happens to be routed underneath our high-speed data lines. In this scenario, we might end up with a thorny signal integrity problem due to a phantom current injected where it doesn't belong.

Copper Pours and Ground Planes

To avoid problems like the one shown in Figure 30-3, you can place a continuous conductive fill—a *copper pour*—on the underside of the PCB. The pour is typically connected to ground, and except for its shape, it's not fundamentally different from a trace.

In KiCad, you can insert a pour by pressing CTRL-SHIFT-Z. The app will let you select the associated net and then automatically generate setbacks around existing traces to avoid electrical shorts to any other nets. Pressing B recomputes all existing pours.

Board-wide copper pours offer a convenient alternative to adding designated return current paths underneath all high-speed data lines. Use them carefully, though. While pours reduce inductance, they also add parasitic capacitance to every signal line. In digital circuits, this is usually acceptable—chips can supply healthy currents, and inductance is the bigger concern. In analog circuits, the trade-offs are less clear; extra capacitance is almost always bad news in op-amp feedback loops, for example.

It's also easy to negate the benefit of a copper pour by routing a trace that cuts through it near the data bus we wanted to protect (Figure 30-4). This layout effectively severs the low-inductance path, forcing the return current to go around it, possibly through a path you truly did not expect.

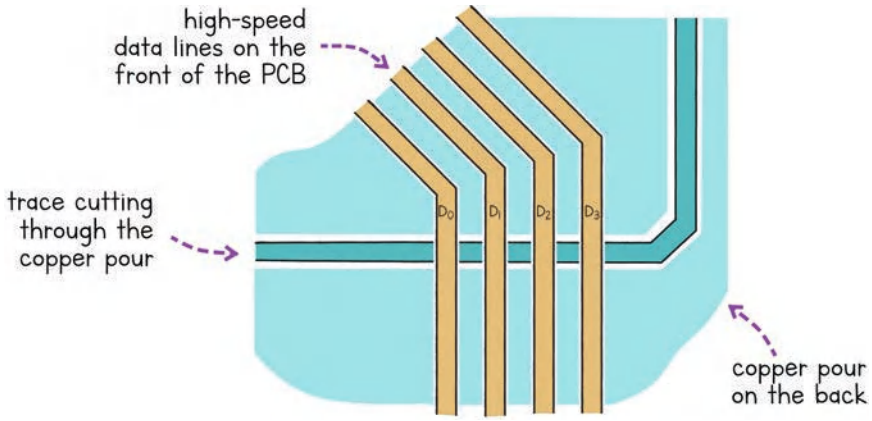


Figure 30-4: Current return path via a copper pour severed by an errant trace on the back of the PCB

Copper pours can also mask power routing problems. They seem like the perfect way to distribute supply voltages to every corner of the PCB, but if a portion of a board is reachable only through a narrow clearance around existing signal traces and component footprints, you might end up with currents having to squeeze through remarkably circuitous paths. The problem can be mitigated by placing ground pours on both sides of the PCB and stitching them with vias, but the bottom line is that you must still pay attention to the power architecture of your design.

If you have a signal integrity problem to solve and copper pours are not cutting it, a better alternative may be a four-layer PCB. The most common approach is to put signal lines on the two outer layers and keep the inner layers for ground. In this architecture, data bus inductance is lowered even further because the distances between the signal line and the return path are reduced. Further, the power planes are continuous except for the occasional via, so we don't need to worry about the routing glitches outlined earlier. The downside of four-layer boards is their cost: You should expect to pay at least twice as much as for their two-layer counterparts.

Both copper pours and power planes will interfere with radio reception and transmission, so if you're using any wireless modules—for example, the ESP32—you will need to maintain a copper-free setback region near the on-device antenna.

EMI Shielding

An EMI footprint of an otherwise well-designed digital circuit can be further reduced with shielding. The simplest approach is to sandwich the data lines between ground planes on a four-layer board—essentially the inverse of the signal-ground-ground-signal stackup we discussed before. This layout is not the default because it results in more crosstalk between data lines that are crammed more closely together; such boards are also harder to modify or repair. That said, the architecture is a budget equivalent of a Faraday cage. The integrity of the cage can be improved by placing an array of closely spaced vias along the perimeter of the board, preventing electromagnetic waves from exiting to the side.

For electromagnetic interference originating from chips and other components, it's also possible to use small rectangular metal enclosures (PCB cans; see Figure 30-5). These cans go over the problematic circuitry and are soldered to the board. The enclosures are a common sight on Wi-Fi radio chips and cellular modems, but they're not something you add unless you know you need it.



Figure 30-5: A metal can shielding the RF SoC on a now-vintage Intel Edison board

As the final line of defense, the entire device can be placed in a grounded metal container or a metallized plastic enclosure. Such conductive coatings can sometimes be found inside laptops and other compact, high-performance computing devices.

Signal Reflections and Trace Length Matching

Before we wrap up, we ought to circle back to Chapter 6. In that chapter, I talked about the speed of light and the consequences it has for electronic signaling. In particular, I noted that it's sometimes necessary to match the length of signal lines that make up a digital bus to make sure the signals arrive at the same time. This is almost never important on hobby PCBs, but you should keep this consideration in mind when working with external memory chips, video interfaces such as the MIPI-DSI, or some of the faster data modes of the USB protocol.

Another relativistic behavior worth bringing up again is signal reflection. Reflections can happen if two components are connected by a conductor that's long in proportion to signal wavelength, and if the current capacity of the transmitting side doesn't match the current demand of the receiving side. In digital circuits, a classic example may be a CMOS output stage driving a high-impedance digital input line.

On a typical PCB, reflections are usually negligible at sine wave frequencies below 200 MHz. That said, as a practical example, they may need to be managed if you're building a radio transmitter or a receiver. The solution is impedance matching. If a low-impedance source is driving a high-impedance load, the matching can be achieved by incorporating a shunt resistor to ground on the receiving side. This solution may be coupled with a small series resistor on the originating end too.

We're now up to "speed" on the fundamentals of high-speed circuit design. Next, we'll talk about reliability.

31

ELECTRONICS THAT LAST



In this final chapter, I'll share a couple of tips on how to fine-tune designs so that they can survive rough handling and difficult operating conditions. By that, I don't mean a mission to Mars; the goal is to make a circuit robust enough to install in a vehicle or hand to a clumsy friend.

We'll cover the bare essentials, but for an in-depth treatment of the subject, I recommend *Designing Electronics That Work* by Hunter Scott (No Starch Press, 2025). It covers everything from tin whiskers to regulatory compliance testing in far more detail than can fit into a single chapter of this book.

Supply Testing

By now, you're well aware of the perils of running components at excessive voltages—the story usually ends with something catastrophically failing and releasing its magic smoke. Undervoltage conditions, by contrast, are rarely catastrophic, so they tend to get less attention. In practice, though, they're quite common, especially in battery-powered devices. The nominal voltage of an AA cell is 1.5 V, but the battery is still considered usable down to 1.1 V or so, and commercial products that malfunction on partially discharged batteries are more common than you might think.

Undervoltage is typically less of an issue for devices powered by a wall wart, but it's still useful to examine how your circuit behaves when the supply is reduced by 10 to 15 percent below the baseline. Cheap power supplies aren't precise, and their voltage tends to sag in proportion to the connected load. Further, transient dips may be caused by outside factors, such as a furnace blower or other large household appliance switching on.

During testing, be sure to exercise all the functions of your device. To give you an example of why this matters, in one of my handheld games, the MCU and the display module worked fine on nearly dead batteries, but the memory chip that stored high scores did not. I got some disapproving looks from my kids for that oversight.

In a microcontroller-based circuit, it's usually possible to enable a *brown-out detector (BOD)* on the chip. This is an internal comparator that halts the MCU if the supply voltage falls below a configured threshold. In critical applications, where a graceful shutdown is better than mysterious malfunctions, a BOD can save the day.

Polarity and ESD Protection

In Chapter 14, we introduced a transistor-based polarity protection circuit. This simple design is worth including in most "consumer" devices, with the possible exception of devices powered over USB or equipped with an internal PSU. In all other cases, a single transistor can prevent unnecessary drama if a battery is inserted backward or if two identical-looking barrel-plug power supplies are mixed up.

Semiconductors are more vulnerable to electrostatic discharge (ESD) in the workshop than they are once they are mounted to a PCB. In Chapter 10, we covered some mitigations that can be used in a lab. Nevertheless, even board-mounted devices can be fried by a sufficiently strong and well-aimed zap. For that reason, transient voltage suppressor diodes are worth adding, especially near exposed connectors, small switches, and other conductive bits that people may touch with their grubby, electrostatically charged hands.

When it comes to fuses, they might be unnecessary in inexpensive battery-operated devices, but they're worth having when using lithium-polymer cells, which can supply destructive currents or overheat. More broadly, add a fuse anywhere a short circuit could conceivably start a fire.

Cable Strain Relief

Cables attached directly to a PCB will eventually break off if they're handled or allowed to move. The simplest workaround is to add a small blob of hot-melt glue, silicone caulk, or flexible polyurethane sealant to secure the wires near their attachment points. This distributes the mechanical stress and greatly improves the odds that the circuit will stand the test of time.

A more elegant but costlier solution is to use a connector. A cable soldered to a basic connector can still break off, but it can be reinforced with a heat-shrink sleeve that partially overlaps the connector body. Another trick is to immobilize the cable with a zip tie or cable clip; the tie or clip can anchor to a purpose-made cutout in the PCB.

For devices in an enclosure, strain relief and abrasion protection must also be provided for any cables entering or exiting the housing. As you drill holes, make sure to deburr the openings carefully. Slide-on strain relief parts are available if you want a professional look, but a heat-shrink sleeve can be used in a pinch.

Environmental Protection

Circuits that are meant to be used without an enclosure can be protected from moisture and accidental contact with metal objects by applying a conformal coating. Various brush-on and spray-on products are available from companies such as MG Chemicals. Most formulations use fairly toxic solvents and other additives, so I recommend

against the spray-on types; brush-on silicone is probably the safest bet. Regardless of chemistry, most of these products form a smooth, transparent layer that can later be dissolved with rubbing alcohol, scraped away with a knife, or burned off with a soldering iron if you ever need to make repairs to the board.

Most electronic assemblies are insensitive to vibration and shock, but heavy or oversized components—transformers, big capacitors, and so forth—are a special concern. For additional protection, they should be affixed to the board with adhesive. If you're building a product you intend to sell, drop-testing is advised either way; if you don't do it yourself, your users will.

Circuits that will be operating in harsh environments or that include delicate components such as displays should be housed in an enclosure. Generic enclosures made from ABS plastic or aluminum are available from companies such as Hammond Manufacturing and come in just about any size you might need. You can customize them by drilling holes or cutting openings to accommodate switches, displays, and so forth. When planning to use an enclosure, take note of the position of its internal mounting holes so that you can add matching holes on the PCB to attach it in place. Screws and brass or nylon standoffs can also be used to connect multiple PCBs together into a single, sturdy assembly.

Most electronic components are reasonably tolerant of weather, but as discussed in Chapter 8, extreme cold or heat will adversely affect capacitors, and at low temperatures you'll also see a marked drop in the performance of liquid crystal displays and most batteries. For outdoor or automotive hardware, temperature testing is a good idea. Keep in mind that on hot days it's harder for a circuit to dissipate heat, and you might need a heatsink or a fan.

You can use your freezer for the low-temperature part of the test, but you should seal the board in a zipper bag to minimize condensation. A toaster oven can serve for the high-temperature test, but domestic ovens have poor regulation at lower temperatures. The safest method is to preheat the oven, check the temperature with a thermometer, switch it off, and only then place the circuit inside.

Last but not least, if you're building any equipment that needs to operate in extreme conditions, you might want to consider *potting*. This involves placing the circuit in an enclosure and filling it with a low-viscosity silicone, epoxy, or polyurethane resin. Once cured, the circuit is encased in a solid block of plastic. The main advantage is extreme resistance to environmental stress and physical shock. The downside is that if the circuit fails, there's little chance of fixing it.

Serviceability

Even if you design your projects to be robust, it's impossible to anticipate every eventuality and work around every possible fault. To that end, my final piece of advice is to keep your projects easy to diagnose, repair, and extend. That starts with good documentation: If you're building something for other enthusiasts, consider publishing schematics, well-commented source code, and detailed project notes. Placing the code under an open license (such as the MIT License or GNU GPL) makes it easier for others to extend your design.

On the PCB itself, it pays to make sure all components are clearly labeled and to provide test pads wherever measurements or calibration might be needed down the line. Providing silkscreen legends for all switches and connectors is a good habit too. It's not uncommon for hobbyists to service vintage circuits for which the original blueprints are no longer available; the easier it is to decipher the printed circuit board, the better the odds of success.

In the same vein, some circuit designers leave a number of unpopulated but connected component positions on the board. This way, if they later decide to add an extra decoupling capacitor, add a ferrite bead, or incorporate a new feature, they can do so without replacing the entire PCB.

Once again, there's much more to say about circuits designed to withstand extreme temperatures, pressures, or high-radiation environments. That said, with the advice in this chapter, you should be well equipped to tackle the more practical challenges of hobby work.

EPILOGUE



The challenge of writing about electronics is that the field evolves rapidly; it's almost never productive to teach it the way you were taught. Many introductory books spend too much time on long-obsolete components—from unijunction transistors to tunnel diodes—while barely covering charge pumps, phase-locked loops, or digital signal processing.

At the beginning of the book, I quipped that *The Secret Life of Circuits* may be what you need to make sense of academic textbooks and Wikipedia articles. Of course, I trust it offers something more. Electronics is a powerful tool for understanding the world around us and solving everyday problems creatively. It takes practice to build the right instincts, but I hope the book has given you the confidence to explore further.

For continued analog circuit education, I recommend *The Art of Electronics* by Paul Horowitz and Winfield Hill (Cambridge University Press, 2015). For the digital realm, it's difficult to point to a single resource—the landscape of chips, peripherals, and programming languages shifts every couple of years. That said, much of the field is software engineering in disguise. Knowing the fundamentals of your programming language and reviewing the MCU datasheet will get you far. For the clever algorithms used in digital signal processing, I recommend *Understanding Digital Signal Processing* by Richard G. Lyons (Prentice Hall, 2011).

As for *The Secret Life of Circuits*, I welcome your feedback, questions, and ideas. Reach me at *lcamtuf@coredump.cx*.

APPENDIX: TOOLS AND SUPPLIES



When it comes to electronics, experimentation is half the fun. While you could read this book just for the theory, you'll get much more out of it if you're willing to get hands-on. In this appendix, we'll go over some of the equipment and consumables you'll need to set up a basic home workshop. This way, you'll be able to test new ideas, see what happens when you modify existing circuits, and dig deeper if your observations don't add up.

You don't need to buy all the gear discussed in this appendix right away. Over time, though, these items tend to find their way into every lab. Fortunately, the equipment is fairly inexpensive and doesn't take up much space. Just as important, most electronic components cost pennies. It's easy to build, modify, and tear apart circuits—and it's no big loss if you let the sparks fly.

When discussing product specifications, I'll introduce a bit of technical jargon. If you've skipped ahead to the appendix, some of the terminology might seem confusing, but don't fret. Think of it simply as a shopping list, then flip back to the earlier chapters to catch up on the theory.

Digital Oscilloscope

By far the most important circuit diagnostic tool is an *oscilloscope*. This device is essentially a specialized computer that lets you observe how electrical signals change over time. It operates much like a hospital heart-rate monitor: The horizontal axis on the

screen represents time, and the vertical motion corresponds to signal magnitude (see Figure A-1). Slow signals can be viewed in real time, while rapid changes are recorded and reviewed afterward.

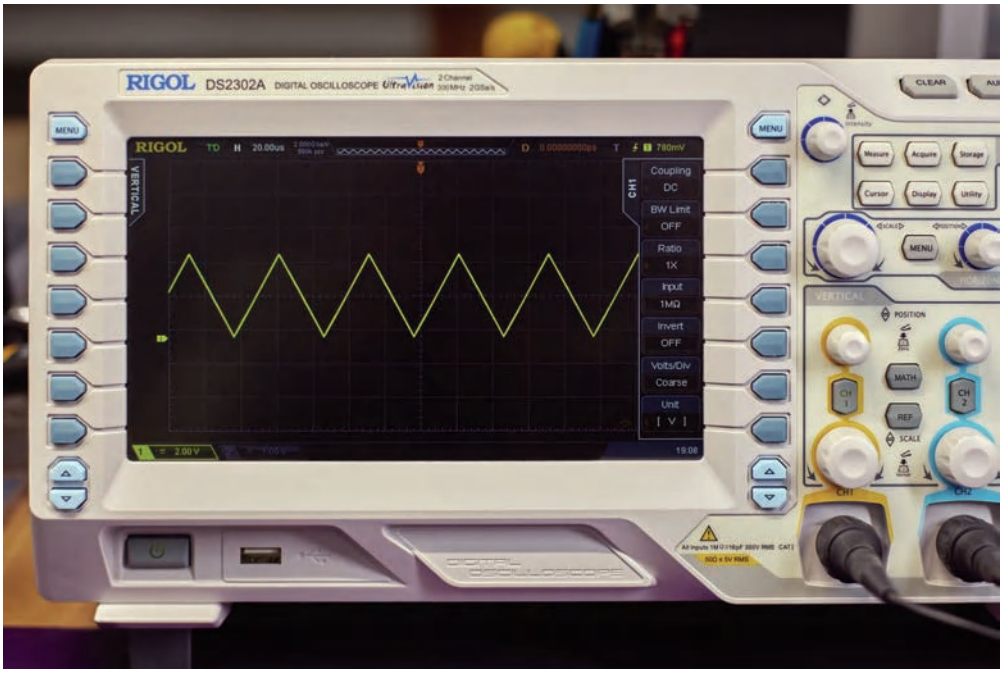


Figure A-1: An affordable oscilloscope displaying a triangle wave

The most fundamental parameter of an oscilloscope is its bandwidth, which describes the fastest-changing signal the device can faithfully record. For hobby use, a bandwidth of about 200 MHz should suffice. Another key attribute is the number of independent input channels the device can monitor simultaneously. For budget scopes, the choice is typically between two and four channels, and two will do the job.

In the past, oscilloscopes were very expensive. Today, you can buy a quality unit for about $350. There are cheaper models, but their performance and usability are fairly poor. Among the reputable brands, Rigol and Siglent, two Chinese manufacturers, are known for dependable, low-cost designs. Their product lines evolve over time, but as of this writing, the Siglent SDS1202X-E and Rigol DS1202Z-E both fit the bill. Another seemingly good alternative from the same region is the UNI-T UPO1202CS.

Some readers may be uneasy about supporting China's communist regime, but when it comes to test equipment, there are few alternatives. Even domestic brands manufacture much of their hardware overseas. In fact, you might end up paying extra just for a premium label—for example, some Teledyne LeCroy products appear to be rebadged Siglent gear.

Digital Multimeter

A *digital multimeter (DMM)* is an essential companion for your oscilloscope. It can measure a wider range of circuit parameters and with greater accuracy, though at much lower speed. These devices are inexpensive and easy to use, and they can save you time when you're performing basic circuit diagnostics, sorting through unlabeled components, or doing general DIY repair work.

When shopping for a DMM, look for a model that measures voltage, current, resistance, capacitance, and circuit continuity. Autoranging functionality is also useful. Extra features don't hurt but are rarely essential. The UNI-T brand has a good reputation for multimeters, and its mid-range UT139C model sells for about $60. A more upscale option from a US company is the Fluke 15B+, currently retailing for $110. You can find off-brand models for as little as $10 to $15, but these units usually lack vital safety features that protect users when working on mains-powered devices.

In some workshops, you'll find bench-top DMMs instead of battery-powered handhelds. These meters offer nicer displays, better user interfaces, and significantly higher accuracy. That said, there's a price to pay: Mid-range models such as the Rigol DM3058E or Siglent SDM3055 start at around $500. For most users, such an investment isn't necessary.

Lab Power Supply

Most electronics operate at low voltages supplied by batteries or simple "wall wart" *power supply units (PSUs)*. While these can be reused for experimentation, a specialized, continuously adjustable PSU is far more convenient (Figure A-2). Further, by providing precise control over current, a dedicated supply also reduces the likelihood of component damage if you wire a circuit incorrectly.

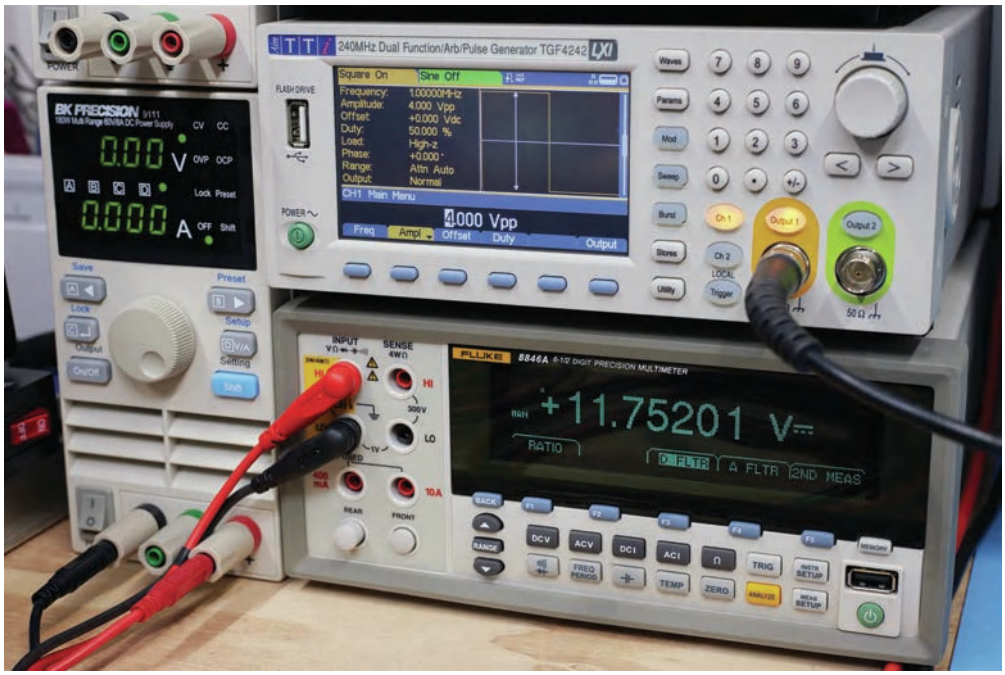


Figure A-2: Common prototyping equipment: a lab power supply (left), bench multimeter (bottom right), and function generator (top right)

Lab power supplies come in a range of voltage and current ratings, but the most versatile models are typically rated up to about 30 volts and 5 amps. For operator safety, I recommend sticking to these limits. Because PSUs are safety-critical, it's best to buy from a well-established, reputable brand. Although rare, a faulty unit could deliver you a nasty shock or cause other harm. Sensible choices include the Siglent SPD1305X, UNI-T UDP1306C, and Rigol DP711, each currently selling for about $300.

Function Generator

Function generators are single-purpose devices that produce a variety of waveforms at configurable frequencies. Despite how narrow that may sound, these gadgets are surprisingly versatile. They allow you to evaluate the behavior of analog filters or amplifiers, provide improvised clock signals for integrated circuits (ICs), and support a range of interesting experiments involving alternating currents, light, and sound.

A function generator isn't essential for a beginner lab—you can always put together one-off helper circuits to perform similar tasks. But that approach soon grows tedious. If your budget allows, a function generator is a smart investment.

Every current-production unit should include all the features you're likely to need. The main factor distinguishing one unit from another is maximum operating frequency: At least 20 MHz is desirable, and more never hurts. The UNI-T UTG962E looks great on paper—it's a 60 MHz unit selling for about $220. Another strong contender is the Siglent SDG1022X Plus, which offers lower bandwidth (25 MHz) and a higher price but comes from a manufacturer with a strong track record for reliability.

Soldering Equipment

In principle, electronics can be practiced in front of a computer screen. You can simulate circuits in software such as QSPICE, design printed circuit boards (PCBs) in an open source tool like KiCad, and send the finished files to a vendor that will etch the boards, install components using a pick-and-place machine, and ship the hardware to your doorstep, all for less than $100. The downside is that the process takes weeks, and discovering a design error at the end can be agonizing.

For this reason, it pays to prototype in the privacy of your own workshop. To do that, you'll need to learn how to solder components by hand.

Irons

If you're new to soldering, take this to heart: The learning curve depends almost entirely on the soldering iron you choose. I wrestled for years with traditional hardware-store models. These irons are about the size of an electric toothbrush and about as much of a precision instrument. Although my skills progressed from terrible to tolerable, it always felt like I was fighting the tool.

The secret to success is to get a miniature soldering iron like the one shown in Figure A-3. Such devices are shaped like pencils—if you can write, you won't need to learn any new motor skills.



Figure A-3: A pencil-style soldering iron

I think the best bang for your buck is a Weller WT1010M kit that includes a 1.3 mm chisel tip, for about $470 total. Another solid option in the same range is a Hakko FX-971 base with an FX-9703 handpiece and a 1.0 mm T50-D1 tip. Some hobbyists also speak highly of inexpensive Pinecil soldering irons from China. I can't attest to their long-term reliability, but at $40 a pop, they might be worth considering.

Wire and Accessories

When choosing solder wire, there are three parameters to pay attention to. The first is the diameter of the wire: The most common options are 1 mm (0.040 inches) and 0.8 mm (0.032 inches), but I find 0.5 mm (0.020 inches) to be a better choice for precision work. The choice isn't critical, but when working with modern surface-mount components, thinner is better.

Next, select the alloy. Traditional solder contained around 37 percent lead, and some recalcitrant hobbyists still prefer it. Modern lead-free options such as SAC305 (96.5 percent tin, 3 percent silver, 0.5 percent copper), however, perform just as well if used with a quality soldering iron. There's really no reason to track toxic heavy metals around your home—lead poses special risks to pregnant women and young children.

Finally, most solder wire has a hollow core filled with *flux*. At high temperatures, flux removes oxidation and allows the diffusion of metal to form a strong bond. Traditional fluxes are made out of pine rosin mixed with activators such as weak organic acids. Other formulations are taking over for automated assembly, but for hand-soldering, *rosin mildly activated (RMA)* flux remains the best choice. RMA-cored solder works well, has an unoffensive odor, and leaves residues that rarely need cleaning.

Taking all this into account, I have used Indium Corporation's WIREFC-52837-0454 for years, and I think it's excellent. Comparable options include the AIM Solder 14054, Chip-Quik RASWLF.020, and Weller T0051386499. Whichever you choose, I recommend buying a 1-pound spool. It costs about $80, but it will last you for years.

It should be noted that although solder wire contains a small amount of flux, the substance evaporates in a matter of seconds when you're actively working on a joint. In many situations, more flux needs to be added to avoid poor wetting, shoddy joints, and solder bridges across component pins. Paste-consistency fluxes sold in jars are common, but I find liquid formulations to be easier to apply. I use MG Chemicals' 835 and dispense it from a small, needle-tipped dropper bottle, as shown in Figure A-4.



Figure A-4: Soldering accessories: desoldering braid, flux-cored solder wire, liquid rosin-based flux, ceramic tweezers, and a desoldering gun for through-hole components

To remove excess solder from a circuit board, use braided copper wick for small spots. The product costs pennies—for example, MG Chemicals' 426 is a solid choice. For more substantial rework, a desoldering gun helps pull out old-fashioned through-hole chips, while a hot-air station takes care of their modern surface-mount equivalents. Both tools can wait, however, until you need them for a specific task.

Ceramic-tipped tweezers also come in handy for holding small components in place while soldering. Unlike metal, ceramic doesn't conduct heat well, so the tweezers don't interfere with soldering or desoldering. Figure A-4 shows a Techni-Pro 758TW265, but an off-brand variety selling for less than $10 is an equally valid choice.

Prototyping Boards

For decades, electronic components were manufactured primarily using *through-hole technology (THT)*. These parts had prominent, pointy leads arranged on a 2.54 mm (0.1 inch) grid (Figure A-5, left). These leads were meant to be inserted into holes on a PCB, then soldered from the opposite side. In industrial settings, this soldering was often done by moving the PCB over a pool of molten solder.

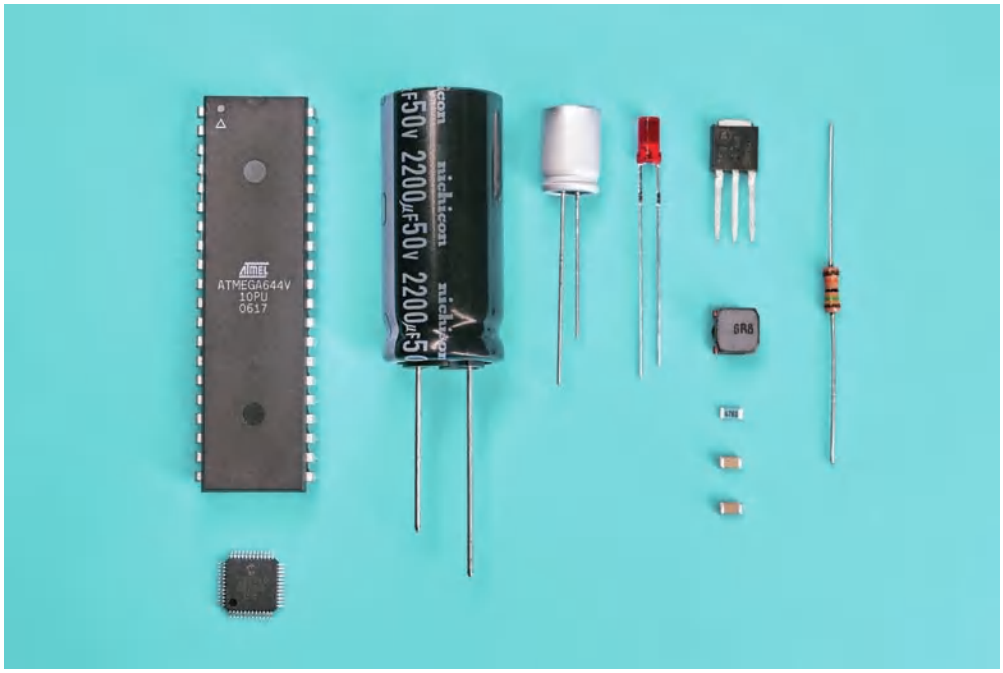


Figure A-5: An assortment of through-hole and surface-mount components

Nowadays, most consumer electronics use *surface-mount technology (SMT)*, with components called *surface-mount devices (SMDs)*. In this approach, the parts have comparatively tiny leads or pads that are as little as 0.4 mm (0.016 inches) apart and are bonded to the same side of the PCB as the component itself (Figure A-5, right). Industrially, assembly is done by precisely applying a paste-like mix of fine particles of solder and flux, placing the components in the right spots, and then heating the entire assembly in an oven until the solder melts and reflows.

Surface-mount assembly is accessible to hobbyists, but the components are small and delicate, so it's usually impractical to improvise. Even if you're doing the soldering yourself, a custom PCB needs to be designed for every task. Once again, the process is uncomplicated and inexpensive, but it takes a good while for the mail-order boards to arrive.

For this reason, I find it more rewarding to prototype with through-hole components whenever possible. It helps that most electronic devices are still being manufactured in through-hole packaging, although this variety costs a couple of cents more. Important exceptions exist—especially for many of the latest integrated circuits—but inexpensive breakout boards can "convert" many surface-mount form factors to through-hole.

In the THT world, the simplest prototyping method is the *solderless breadboard*, shown in the middle of Figure A-6. These boards allow through-hole components to be temporarily seated and then connected using jumper wires. Breadboards aren't suited for high-speed or high-power applications, and they obviously shouldn't be used for anything that's meant to last, but for quick tests, the convenience is hard to beat.

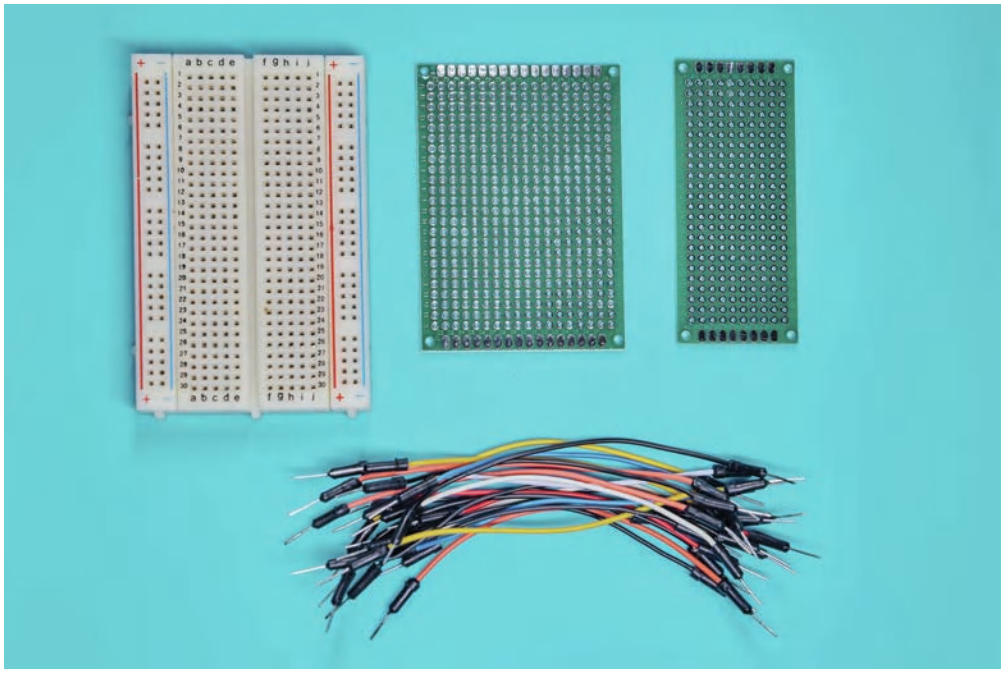


Figure A-6: Two simple prototyping options: a breadboard with jumper wires and a pair of perfboards

A more permanent solution is to use *perfboards* (Figure A-6, right). These are premade PCBs with an array of holes but typically no electrical connections between any two points. After inserting the components, you can form custom connections by soldering short pieces of wire between the populated pads.

Workholding and Wire Handling

One of the biggest frustrations in manual soldering is that you need one hand to keep the PCB in place, another hand to feed the solder wire, a third hand to hold the soldering iron, and a fourth hand to position the component you're trying to attach. As of this writing, humans lack that many appendages, so good workholding accessories are essential.

The most common gadgets for solving this problem are alligator-clip "helping hands," available for pennies on Amazon. They're finicky, though, and they offer no natural hand rest. Years ago, I ditched such gadgets for the sturdy PCB clamp shown in Figure A-7, and I've never looked back. For more advanced soldering under a microscope, low-profile clamps such as the Adafruit 3718 or Omnifixo are good choices too.

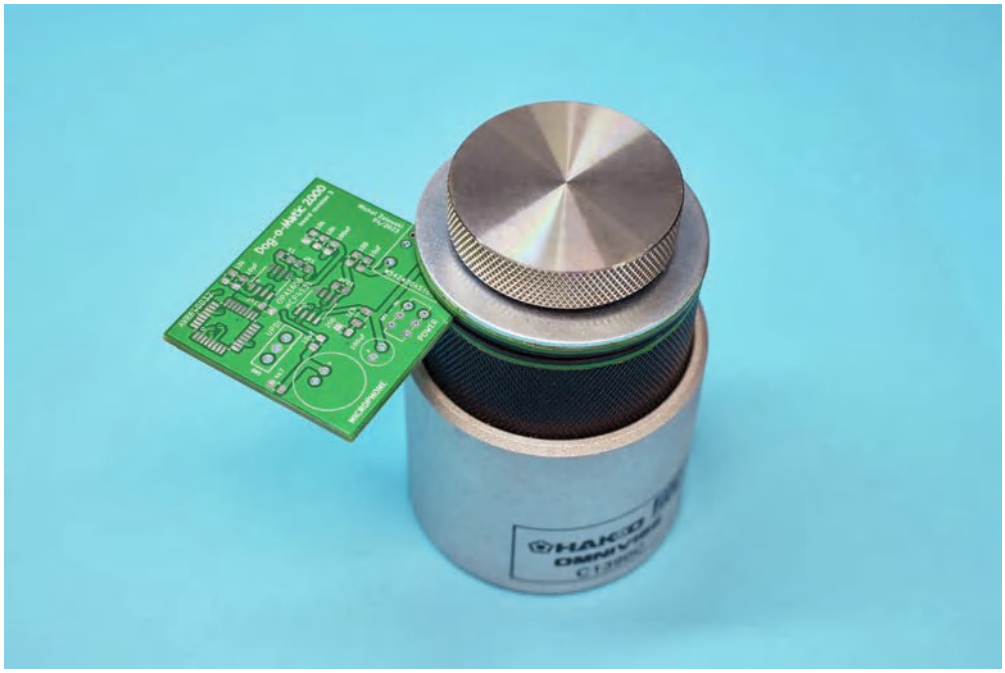


Figure A-7: A simple PCB vise: the Hakko C1390C

Several other worthwhile material-handling tools are shown in Figure A-8. These include a small wire stripper (Knipex 12 80 040 SB), which is indispensable for removing plastic insulation from cables for soldering. I also recommend flush cutters, such as the Hakko CHP-170, useful for both cutting wire and trimming component leads. Finally, bent-nose flat-jaw pliers come in handy for pushing wires into tight spots.



Figure A-8: Flush cutters, bent-nose pliers, hook-up and "wire wrap" wire, and a wire stripper

As for wire choice, standard solid 24 AWG (0.5 mm) copper wire in PVC insulation is adequate for supplying power in most electronic circuits. Much thinner 30 AWG (0.25 mm) "wire-wrapping wire" works well for routing signals on perfboards and for a variety of precision repair tasks.

Sourcing Tools and Components

Most of the equipment and consumables discussed here should be easy to procure from a wide range of reputable online retailers. In the United States, the best prices on test equipment and soldering gear can usually be found at *https://www.tequipment.net*. This New Jersey company has been around for decades and is a safer, less chaotic alternative to Amazon or eBay, where costly low-volume gear sometimes attracts fraud. On the flip side, you don't need to worry about such shenanigans when shopping for low-value consumables and tools. Wire, perfboards, and tweezers are safe to buy on Amazon (or anywhere else).

When buying electronic components, exercise some caution. Items such as integrated circuits and large capacitors are widely counterfeited in China. If you're in the United States, it's best to stick to DigiKey (*https://www.digikey.com*) and Mouser (*https://www.mouser.com*). These distributors have vast and well-organized inventories with parametric search tools that let you sort parts based on key specs. Just as important, they monitor the provenance of what they sell. To be sure, purchasing an assortment of standard resistors on Temu or AliExpress is low risk, but other parts are a gamble.

Large-scale retailers aside, there's also a selection of outlets catering to more specialized needs. For example, Kysan Electronics (*https://kysanelectronics-com.3dcartstores.com*), despite its sketchy URL, is a dependable source of miniature motors. Similarly, *https://theelectronicgoldmine.com* and other outlets like it stock obsolete and one-off components.

Lab Safety

Many hobbies carry risks, and electronics is no exception. Compared to woodworking, snowboarding, or cooking, though, this pursuit rarely threatens life and limb.

The clearest danger comes from high-voltage circuits, especially household wiring and appliances. This book sticks to low voltages that are unlikely to cause harm on brief contact with skin. If you're planning to work on mains-powered equipment, you must follow additional safety precautions. Above all, you must double-check that the device is no longer energized, confirming both visually *and* with a noncontact voltage tester such as the Fluke 1AC II. To keep others safe, you should also read up on electrical building codes or other safety standards that are relevant to the project at hand.

As with many other hobbies, it's important to pay attention to fire risks. Soldering irons can get hot enough to start fires if left unattended next to a pile of printouts or workshop rags. Some electronic components can also send sparks flying if they malfunction and overheat. To avoid mishaps, it's best to keep your work area tidy. More fires start in kitchens than in workshops, but having a fire extinguisher within reach is wise.

When it comes to chronic health risks, some electronic components may contain tiny amounts of toxic elements such as arsenic, selenium, or cadmium. In the same vein, exposure to lead is possible when servicing vintage electronics or using leaded solder, while flux fumes can bother those with respiratory conditions. To the best of my knowledge, the odds of a hobbyist developing serious health problems are minimal,

but simple habits—regular handwashing, wearing light-duty eye protection, and using wet wipes for cleanup—cost little. Folks with lung conditions might also benefit from a small and inexpensive fan or a fume extractor.

INDEX

Page numbers in italics refer to figures.

Symbols

<< bit-shift-left operator, 378
#include directive, 376
% modulo operator, 378
¬ (negation operator), 296

Numbers

1-bit DAC, 344
1N4148 diode, 271–272
-3 dB frequency, 195
3D Viewer, KiCad, 396
74HC series, 295
  74HC00, 301
  74HC02, 301
  74HC04, 298
  74HC08, 301
  74HC14, 298
  74HC32, 301
  74HC74, 317, 323
  74HC75, 316
  74HC86, 302
  74HC125, 298
  74HC161, 325
  74HC164, 330
  74HC165, 330
  74HC393, 324, 348
  74HC4020, 324
  74HC4046, 351
  74HCU04 inverter, 294
8-bit microcontroller, 361
0402 package, 388–389
0603 package, 388–389
0805 package, 388–389
01005 package, 388–389
1206 package, 388–389
1608 metric package, 388
2012 metric package, 388
3216 metric package, 388

A

A (ampere), 18
AA cell, 409
abacus, 355
AC (alternating current)
  AC-to-DC conversion, 209
  adapters, 106–108
  characteristics, 239–241
  coupling, 195
  voltage, 28
acceleration, 38–39, 42–43
accelerometers, 288
active components, 111
active filters
  cascaded stage interaction, 261–263
  cascading R-C stages, 259–261, *260*
  multistage R-C filter, 263–264, *264*
  Sallen–Key topology, 264–268, *267*
Adafruit 3718 clamp, 422
Adafruit UPDI Friend, 364
ADC0.CTRLA register, 366
ADCs (analog-to-digital converters)
  bar meter project, 379–381, *380*
  delta-sigma ADC, 342–344, *342*
  flash and pipelined ADCs, 339–340, *339*
  MCU peripherals, 371–372
  overview, 338
  resolution and sampling, 338–344
  slope-based ADC, 340–341, *340*
  successive approximation register ADCs, 341–342
adders
  full adder, 308–311, *308*
  half adder, 307–308, *308*
ADXL325BCPZ, 288
Ah (ampere-hour), 19, 105
Aisler, 395
alkaline battery, 104–105, *104*
alligator-clip helping hands, 422
alternating current. *See* AC
ALU (arithmetic logic unit), 354–355
aluminum electrolytics, 119–120

amateur radio bands, 86
Amazon, as electronics source, 424
AM demodulation, 210, *210*
ammeter, 24–25
ampere (A), 18
ampere-hour (Ah), 19, 105
amplification, 225
amplifiers
  diagrams, 170–171
  differential, 227–229, *228*
  inverting, 226–227
  overview, 225
amplitude, 30
amplitude modulation (AM), 86–88
analog accessories, 371–372
analog computation, 268
Analog Devices, 286
analog switch, 289
analog-to-digital converters. *See* ADCs
AND gate, 215–216, 299–300, *299*
angular frequency, 35
anode, 165
antenna design, 84–86, *85*
anti-resonant frequency, 140
AP3012, 282
AP62200, 282
Appearance Manager, KiCad, 396
arctangent, 68, 180
Arduino UNO Q, 362
arithmetic logic unit (ALU), 354–355
ARM CMSIS library, 364
ARM Cortex-A, 362
ARM Cortex-M0, 361
ARM Cortex-M7, 362
Arm Holdings, 361
Armstrong, Edwin, 93
assembly language, 363
Atmel ATmega series, 361
Atmel AVR-Dx device support pack, 377
Atmel-ICE, 364, 375
attenuation from capacitor, 58
audio jack, 130
audio signals, 30–32
automation, 361
average power, 31
AVR128DA28, 360, *360*, 365, 402–403, *402*
`avr-binutils`, 376
`avrdude` utility, 364, 376
AVR Dx series, 361, 369–372, *370*, 401–402
`avr-gcc` compiler, 364, 376–377
`avr-libc`, 364, 375

B

back electromotive force (back EMF), 137
banana plug, 130
bandgap, 150
bandpass filter, 259
bandwidth
  op-amp, 248–249
  signal, 87–88
barrel plug, 130
base (BJT), 160–161, *161*
batteries
  capacity of, 105
  general discussion, 102–104
  management system, 101
  rechargeable, 106
  voltage and capacity, 104–105, *104*
BeagleBoard, 362
beat frequencies, 89–92, *90*
bel, 142
bent-nose flat-jaw pliers, 423, *423*
Berlin, Howard, 285
B fields, 94
bidirectional TVS, 206
bimetallic strip, 132
binary arithmetic
  adders, 307–311, *309*
  multiplication and division, 311–312, *311*
  overview, 305–307
binary code, 293–294, *294*
binary counter, 323–324, *323*
binary input buffer, 294, *294*
binary search, 341
binary-weighted DAC, 333–334, *333*
bipolar junction transistors (BJTs), 160–164, *160*, 219, 222
bit, 307
bit numbering, 310–311
BJT (base), 160–161, *161*
blinking LED
  overview, 373–375
  programming the chip, 376–377
  software side, 375–376
BNC connector, 130
BOD (brown-out detector), 410
body diode, 158–159
Bohr, Niels, 3
Bohr model, 3–4, *4*
Boltzmann constant (k_B), 118
boost converters, 282–284, *283*
boron, 12
bound electrons, 5

box drawing, 169
brown-out detector (BOD), 410
brushed/brushless DC motors, 137–138, *138*, 286
brush-on silicone coating, 410–411
BS170 transistor, 227, *227*
buck converters, 280–282, *281*
buffer and inverter, 296–298, *297*
buffered registers, 370–371
bus, schematic, 168
button cell, 104, *104*
byte, 307

C

C_{in} (input carry), 308
C0G, 121
calculus, 27
capacitance, 76
  capacitor voltage vs. current, 55
  discrete-time capacitor, 55–57, *56*
  overview, 53–54
  reactance, 57–58
  and velocity, 76–77
capacitive reactance (X_c), 115, 156
capacitors, 53–54
  aging, 123
  charging, 189–192, *190*
  dielectric saturation, 121–122
  parasitics, 120–121
  in R-C filters, 192–201, *193*
  in R-C oscillators, 346
  temperature effects, 121
  types and ratings, 119–120
carbon resistors, 112, *112*
carrier frequency, 86–87
Cartesian plane, 39, 67–68
cascaded voltage reference, 207–208, *207*
cathode, 164–165
causality, 71
CC. *See* constant-current source
cells, electrochemical, 102
center frequency, 194–195
ceramic resonator, 348
ceramic-tipped tweezers, 420
charge, electrical, 4–6, 17–18
charge and discharge cycle, 56, *56*
charge carriers, 11–14, 17, 97, *97*, 160–161
charge/discharge cycle, 106
charge pumps, 278–280, *278*
charge separation, 108
chassis ground, 110
chip select (CS), 368
CIPO line (SPI), 368
circuit breaker, 144, *144*
circuit building blocks, 99–172, *144*
circuit protection
  diode clamping, 205–206
  fuses and PTC devices, 143–144, *144*
citrate anion, 103
citric acid, 103
clamper circuit, 214–215, *214*
class I, II, and III ceramic, 121–122
classical magnetism, 94
clockless computing, 345
clock multiplier, 351
clocks in digital circuits
  clock adjustment, 348–352, *348*
  clock sources, 345–348, *348*
closed loop, 177
CMOS (complementary metal-oxide-semiconductor), 294
CMRR (common-mode rejection ratio), 241
coaxial cable, 72–73
coil, 97–98, *98*
collector (BJT), 160–161, *161*
Commodore 64, 361
common-collector topology, 221
common-drain topology, 221
common-emitter topology, 218
common-mode chokes, 126–127, *127*, 404
common-mode current, 126, *126*
common-mode rejection ratio (CMRR), 241
common-source switch, 218–219
common-source topology, 218
commutator, 137
comparators, 289
compass needle, 98
compiler, 361
complementary metal-oxide-semiconductor (CMOS), 294
complex numbers, 181–182, 261–263
component placement, schematic, 395
component value notation, 169
computer architecture
  ALU, registers, and buses, 353–355
  further optimizations, 356
  instruction set, 355
condenser microphone, 141
conduction
  atom, structure of, 3–5
  physics of, 5–10
conduction band, 6, 11–12, 95, 150

conductors, 6
conformal coating, 410–411
connectors, 129–130, *130*
conservation of energy, 37, 40–41
constant-current (CC) source
  current mirror, 224
  power supply CC mode, 102
constant-voltage (CV) mode, 102
constant-voltage source, 104
constructive interference, 82–83
contactors, 135
control terminal, 15–16, *15*
conventional current, 24
COPI line (SPI), 368
copper, 130
copper(II) chloride, 10, *10*
core saturation, 125–126, *125*, 127
corner frequency, 195
cosine, 32–35, *33*, 46, 47
cosine angle sum identity, 90–92, *91*
coulomb (C), 17–18
counters
  overview, 321
  ring counter, 322–323, *322*
  ripple counter, 323–324, *323*
  synchronous counter, 325, *325*
covalent bond, 5
C programming language, 363, 366–367
CPU optimization, 356
crossover distortion, 223–224, *224*
crossover frequency, 69, 250–251
crosstalk, inner-data-layer, 406
crystal oscillators, 346–348
CS (chip select), 368
current, 18–19, 22
  measuring, 25, *25*
  sign conventions, 24–26
  vs. voltage in capacitors, 59–61
current mirror, 224–225, *225*
current-sensing resistor, 184–185
current-to-voltage amplifier. *See* transimpedance amplifier
cutoff frequency, 195
CV (constant-voltage) mode, 102

D

D (data line), 316
DACs (digital-to-analog converters), 371
  binary-weighted, 333–334, *333*
  linearity of, 338
  oversampling averaging, 336–338
  overview, 331–333
  R-2R DAC, 334–336, *334*
  resolution of, 333
Darlington pair, 164
data buses, 354–355
data line (D), 316
data routing
  demultiplexer, 328–329, *328*
  line decoder, 326–327, *326*
  multiplexer, 327–328, *327*
  overview, 325–326
  shift register, 329–330, *329*
dB (decibels), 142
dBA, 143
dB SPL, 143
dBV, 143
DC (direct current), 28
DC-DC converter, 278
DC lab power supply, 102
DC restoration, 214–215
dead zone, 235–236
decibels (dB), 142
decoupling
  `_delay_ms()` function, 375–376
  inductor-based, 201
  supply capacitors, 403
delta-sigma modulation
  1-bit DAC principle, 337, *337*
  in ADCs, 342–344
De Morgan's laws, 303
demultiplexer, 328–329, *328*
depletion mode MOSFETs, 159
depletion region, 13–15, *15*, 108
Designing Electronics That Work (Scott), 409
Design Rules Checker (DRC), 398
desoldering, 420, *420*
destructive interference, 82–83, 89
device family pack (DFP), 364
D flip-flop, 317–318, *317*
diaphragm, 134
dielectric, 75–76, *75*
dielectric saturation, 121–122
difference amplifier, 270–271, *270*
differential amplifiers, 227–229, *228*
differential stage, 235–236
DigiKey, 424
digital logic, 295–296
digital multimeter, 416–417, *417*
digital oscilloscope, 415–416, *416*

digital port I/O
overview, 366–367
parallel 8080/6800 protocols, 367–368, *368*
pulse width modulation, 369–371
serial peripheral interface, 368–369
digital potentiometer, 289
digital-to-analog converters. *See* DACs
diodes
circuit protection and signal clipping, 205–207, *206*
DC restoration, 214–215
general discussion, 145–147
logic gates, 215–216, *215*
rectification, 209–212
reverse-bias operation, 147–148, *148*
specialized variants, 149–150
voltage doubling, 213, *213*
voltage references, 207–209, *207*
DIP-8 package, 237
DipTrace, 395
direct current (DC), 28
directed sum, 177
direct memory access (DMA), 362
discharge curve, 104–105, *105*
discrete semiconductors. *See also* transistors
diodes, 145–150, *150*
LEDs, 150–153, *152*
vacuum tubes, 164–166, *164*
discrete-time capacitor, 55–57, *56*
discrete-time inductor, 59–60, *60*
discrete-time models, 48–49, *49*
division
binary, 311–312, *311*
clock division, 348–349, *348*
D latch, 316, *316*
DMA (direct memory access), 362
documentation, 411
dopants, 12–13, 145–146
doped semiconductors, 12–13
double pole (DP), 131
double throw (DT), 131
double word, 32-bit, 307
DPAK package, 389
DPST switch, 131
DRAM (dynamic RAM), 318–320
DRC (Design Rules Checker), 398
drift speed, 95
drill file generation, 399
drop-testing, 411
DRV5053, 288
DRV8231, 286
DT (double throw), 131
dual-amplifier chip, 237–238
dual-slope ADC, 341, *341*
dual SPI, 369, *369*
duty cycle, 78, 277, *277*
dynamic RAM (DRAM), 318–320
dynamic resistance, 158

E

E (enable signal), 315–316, *315*
E6 scale, 112
E12 scale, 112
earth ground, 110
Edge.Cuts layer, KiCad, 396
edge-triggered architecture, 317–318, *318*
Effective C (Seacord), 363
EIN (equivalent input noise), 143
electret microphone, 141, 251–252, *252*
electrical breakdown, 6, 9
electric dipoles, 74–75, 122
electric field lines, 85–86, *85*
electricity, 3–10, *7*
electric permittivity, 76
electric shock, 10
electrochemical reactions, 10
electrodes, 8, 102–103
electrolytic capacitors, 119–120, *119*
electromagnet, 133
electromagnetic interference (EMI), 401–406
electromagnetic radiation, 8
electromechanical devices, defined, 129
electromotive force
applied to capacitor, 53–54, *54*
defined as voltage, 20
driving thermionic emission, 7–8, *8*
inductor pushback voltage, 59
motor back EMF, 137
electron drift, 7
electron gas, 6, 21, *21*
electronically controlled switch, 218
Electronic Goldmine, 424
electrons, 3, 11–16, *13*
electron shells, 3–5
electron spin, 97
electron transfer, 103–104
electrostatic discharge (ESD), 158, 410
electrostatic field
binding electrons in atoms, 5–6
in capacitors, 53–54
and conduction, 7–8

electrostatic field *(continued)*
relativistic origin of magnetism, 95–97
and semiconductors, 11–16
as voltage, 20
electrostatic force, 4
elemental semiconductors, 11–12
embedded programming, 363
embedded toolchain, 363
EMI (electromagnetic interference), 401–406
emitter (BJT), 160–161
enable signal (E), 315–316, *315*
enclosures, 411
energy and power, 19–20
energy levels, 5–9
energy storage, 189
enhancement mode MOSFETs, 159
envelope follower, 210, *210*
epilogue, 413–414
equivalent input noise (EIN), 143
equivalent series resistance (ESR), 104
ESD (electrostatic discharge), 158, 410
ESP32, 362
Espressif, 362
ESR (equivalent series resistance), 104
Ewald-Oseen extinction theorem, 74–76
exclusive OR, 301–302
exponential converter, 272
exponential decay, 190

F

falling-edge trigger, 324
farad (F), 54
Faraday cage, PCB-equivalent, 406
feedback capacitor, 248
ferrite beads, 126, 402–403
ferrite toroid, 126
ferromagnetic materials
classical magnetism rules, 97
core saturation, 125, *125*
in electromagnets, 133
magnetic domains, 59
FET input op-amp, 239
field-effect transistors (FETs), 14–16, 141
filter knee, 69
fire risks, 424
first-order filter, 203
The 555 Timer Applications Sourcebook (Berlin), 285
flash ADCs, 339–340, *339*
flash memory, 354
flexible printed circuit (FPC), 130
flicker fusion threshold, 152
flicker noise, 241
flip-flop circuits
D flip-flop, 317–318, *317*
overview, 316–317
floating input pin, 374
floating-point units (FPUs), 362
Fluke 15B+ multimeter, 417, *417*
flush cutters, 423, *423*
flux, 419
FM (frequency modulation), 86–87
Footprint Editor, KiCad, 396
force, in springs, 37, *37*
forward bias, 146, 206
Fourier transform, 50
four-layer PCB, 405–406, *405*
FPC (flexible printed circuit), 130
frames of reference, 95–97, *96*
frequency, 29
frequency doubler (PLL), 351, *351*
frequency modulation (FM), 86–87
frequency response, 115, 142
Fresnel equations, 74
full adder, 308–311, *308*
full-bridge motor drivers, 286
full-duplex transmission, 369
full-wave rectifier, 211–212, *211*
function generator, 418
fuse, 143–144, 410

G

gain-bandwidth product (GBWP), 240, 251
gain formula, inverting, 248
galvanic isolation, 106, 128
ganged switches, 279, *279*
gated S-R latch, 315–316, *315*
`gcc-arm-none-eabi`, 364
Gerber file export, 399
grid, vacuum tube, 165, *165*
ground, 24
grounded metal container, 406, *406*
ground planes, 405–406
sandwich shielding, 406, *406*
ground rail, 28
ground symbols, 110

H

H (henry) unit, 59
Hakko C1390C PCB vise, 423, *423*

Hakko CHP-170 flush cutter, 423, *423*
Hakko FX-971 soldering station, 419, *419*
HAL (hardware abstraction layer), 364
half adder, 307–308, *308*
half-wave dipole, 86
half-wavelength lines, 82–84, *84*
half-wave rectifier, 209–211, *209*
Hall effect sensor, 138
Hammond Manufacturing, 411
hardware abstraction layer (HAL), 364
hardware floating-point unit, 362
harmonic motion example, 36–43, *36*
Hartley, Rick, 402
HASL process, 399
header, connector, 130
header files, 366
heat dissipation, 23
heat-shrink sleeve, connector, 410
henry (H) unit, 59
hertz (Hz), 29
H fields, 94
hiccup mode, 282
high-frequency transformer, 283
high-impedance source, 220
high-pass filter, 192–193, *193*
high-speed PCB design
  current return paths, 404–405, *404*
  defining high speed, 401–402
  design priorities, 402–403
  EMI shielding, 406, *406*
  ground planes, 405–406
  supply decoupling, 403, *403*
  trace length matching, 407–408
high-speed signal definition, 401
holes, semiconductor, 12–14, *13*, 108, 146
Hooke's law, 37
hydronium, 103
hypotenuse, 33–34
  in angle-sum identity proof, 90–92, *91*
  as impedance magnitude, 66–67
  in Pythagorean identity, 47
  in sine-derivative proof, 45–46
hysteresis, 126
Hz (hertz), 29

I

IC packages and symbols, 237–238
IDE (integrated development environment), 364
IF (intermediate frequency), 93, *93*
imaginary part, 182–183
impedance, 81–82
  complex, 181–183
  L-C in series, 62–65, *65*
  matching, 407
  mixed, 180–181
  overview, 61–62
  in parallel, 179
  R-L and R-C in series, 65–70, *65*
  in series, 178–179
incremental notation, 55
incremental rotary encoder, 133, *133*
Indium Corporation WIREFC-52837-0454, 419
induced voltage, 97
inductance
  defined, 59
  discrete-time inductor, 59–60, *60*
  inductive reactance, 60–61
  and velocity, 76–77
inductive reactance (X_L), 114
inductor-like load, 219
inductors, 201–204
  common-mode chokes, 126–127, *127*
  core saturation and hysteresis, 125–126, *125*
  current limits of, 125
  ferrite beads, 126
  parasitics, 123–124
Infineon, 362
infinitesimals, 42–43, 45, 60–61
input capacitance, 156
input carry (C_{in}), 308
input impedance for op-amps, 239
input/sense configuration (ISC) fields, 371
instantaneous change, 42–43
instruction decoder, 354–355
instruction reordering, 356
instructions, 354
instruction set, 361
insulators, 6
in-system programming (ISP), 364
integrated circuit packages, 389–390
integrated development environment (IDE), 364
integration, analog, 272
integrator, op-amp, 272–274, *272*
integrator circuits, 338
Intel 8080, 367
interface chips, 286
intermediate frequency (IF), 93, *93*
internal resistance, 104
International System of Units (SI), 18

internet of things (IoTs), 362–363
intrinsic semiconductors, 12–14
inversion layer, 16, *16*
inverter and buffer, 296–298, *297*
inverting amplifier, 247–248, *247*
inverting input, 237
inverting voltage follower, 247, *247*
I/O lines, 355
ionic bonding, 5, 10
ions, 8
IoTs (internet of things), 362–363
iron-core transformer, 128, *128*
irons, 418–419
ISC (input/sense configuration) fields, 371
ISP (in-system programming), 364

J

J (joule), 19
jacks, 130
JFETs (junction field-effect transistors)
  operating principle, 14–15
  ratings and parasitics, 156
  transfer characteristics, 153–155
J-K flip-flop, 318, *318*
JLCPCB, 394
Johnson–Nyquist noise, 116–118
joule (J), 19
Joule's law, 23–24
  for current limiter heat, 184
  in decibel definition, 142
  in fuse design, 144
  for MOSFET heat, 219, *219*
  resistor power ratings, 113, 114
junction, schematic, 168
junction field-effect transistors. *See* JFETs
junction voltage, 333–334

K

k (spring constant), 37–39, *37*
k_B (Boltzmann constant), 118
KiCad, 395–398
kilosamples per second (ksps), 371
kinetic energy, 38–39
Kirchhoff's laws, 176–177, *176*
  Kirchhoff's current law (KCL), 176, 261–262
  Kirchhoff's voltage law (KVL), 177
Knipex 12 80 040 SB wire stripper, 423, *423*
ksps (kilosamples per second), 371
Kysan Electronics, 424

L

lab frame of reference, 96, *96*
lab power supply, 417, *417*
lab safety, 424–426
latches
  D latch, 316, *316*
  S-R latch, 313–315, *314*
latching action, 131
L-C in series, 62–65, *65*
LDO (low-dropout regulator), 276, *276*
LDR (light-dependent resistor), 113
lead-acid battery, 105, *105*
leadless packages, 390
least significant bit (LSB), 311
LEDs (light-emitting diodes), 150–153, *150*
  blinking LED, 373–377, *377*
  LED chaser, 377–378, *377*
  push-to-advance LEDs, 378–379
left-to-right layout, 170, *170*
lemon battery, 102–103, *103*
length contraction, 95–96
LibrePCB, 395
light-dependent resistor (LDR), 113
light dimming, 277–278
light-emitting diodes. *See* LEDs
lightning, 9
linear regulators, 275–277, *276*
line decoder, 326–327, *326*
Linux build environment update, 377
lithium battery, 105, *105*
lithium-polymer battery, 101
LM324, 241
LM358, 241
LM741, 241
LM2766, 280
LM2776, 280
LM4040, 208
logarithmic converter, 271–272, *271*
logarithmic taper, 113
logic gates
  design of, 296–302
  diode-based, 215–216, *215*
  logic levels, 293–295
  truth tables, 295–296
  universal gates, 302–304, *302*
log-log plot, 63–65
loop gain, 246, *246*
low-dropout regulator (LDO), 276, *276*
low-impedance source, 220
low-pass filter, 192–193, *193*
low-profile quad flat package (LQFP), 390

LSB (least significant bit), 311
LSB resistors, 334
LTC1799, 286
luminiferous aether, 95

M

magnetic domains, 125
magnetic field sensor, 288
magnetic hysteresis, 126
magnetic permeability, 76
magnetism
  classical field rules, 97–98, *98*
  in inductors, 59–60
  relativistic origin of, 94–97
mains voltage, 30, 47–48
majority carriers, 14–16, 160–161
mathematical operations
  difference amplifier, 270–271, *270*
  integrator, 272–273, *272*
  noninverting integrator, 273–274, *273*
  op-amp multiplication and division, 271–272, *272*
  summing amplifier, 268–270, *270*
MAX7410, 286
maximum output current for op-amps, 239
Maxwell's equations, 74
MCP1525, 208
MCP1700, 277
MCP6022, 241–242
MCUs. *See* microcontrollers
mechanical resonance, 348
mechanical thermostat, 132
memory
  flip-flop circuits, 316–318, *317*
  latches, 313–316
  SRAM and DRAM, 318–320
memory-mapped registers, 366
MEMS (micro-electromechanical systems), 141
metal film resistor, 112
metallized plastic enclosure, 406
metal-oxide-semiconductor field-effect transistor. *See* MOSFET
metal-oxide varistor (MOV), 144
metal wire, conduction in, 6–7, *7*
mF (microfarad), 54
MG Chemicals, 410
  426 desoldering wick, 420, *420*
  835 flux, 420, *420*
mH (millihenry) unit, 59
MIC2288, 282
Microchip PIC32CZ, 362
Microchip PIC32CZ MC70, 346
Microchip Technology, 241
microcontrollers (MCUs)
  ADC bar meter project, 379–381, *380*
  blinking LED project, 373–377, *377*
  build environment, 364
  defined, 359
  features of, 360–361
  for IoT, 362–363
  LED chaser project, 377–378, *377*
  overview, 359
  for process control, 361
  programmers, 363–364
  programming language, 363
  push-to-advance LEDs project, 378–379
  PWM audio toy project, 381–384, *382*
  for serious compute, 362
micro-electromechanical systems (MEMS), 141
microfarad (mF), 54
microphones
  amplifiers for, 251–253
  decibels, 142
  overview, 141–142
  sensitivity and noise, 142–143
microsequencing, 355
microwave signals, 401–402
midpoint frequency, 194
millihenry (mH) unit, 59
minority carriers, 14
MISO line (SPI), 368
MLCC (multilayer ceramic capacitor), 119–120, 403
modulation schemes, 86–89
mole, 6
momentary action, 131
monolithic chips, 275
Monolithic Power Systems, 286
MOSFET (metal-oxide-semiconductor field-effect transistor)
  enhancement vs. depletion mode, 159
  gate insulation and ESD vulnerability, 158
  operating principle, 15–16
  pull-up and pull-down resistors with, 188, *188*
  threshold voltage, 156–157, *157*
  transfer characteristics, 157–158
MOSI line (SPI), 368
most significant bit (MSB), 311
Motorola 6800, 367

motors
brushed DC, 137
brushless DC, 137–138
overview, 136–137
motor speed control, 277
Mouser, 424
MOV (metal-oxide varistor), 144
MP6637, 286
MPLAB SNAP, 364, 375, 376
MPLAB X IDE, 364, 376
MSB (most significant bit), 311
MSOP package, 390
multilayer ceramic capacitor (MLCC), 119–120, 403
multimeter, 25
multiplexer, 327–328, *327*, 355
multiplication
binary, 311–312, *311*
clock multiplication, 349–352
Murata NME1, 282

N

NAND gate, 301–304, *301*
nanohenry (nH) unit, 59
nano- prefix, 19
NC (normally closed), 131
n-channel JFET, 153
n-channel MOSFET, 218–220, *218*
NE555, 285–286
negation, double, 297
negation operator (¬), 296
negative feedback, 217, 221–222, 240
negative resistance, 68
negative voltage generator, 279, *279*
neon signs, 9, *9*
net, schematic, 168
neutrons, 3
`newlib`, 364
Newtonian physics, 39
nH (nanohenry) unit, 59
NO (normally open), 131
node, circuit, 176, *176*
noise amplitude spectral density, 118
noise floor, 118
noise suppression, 126
noninverting amplifier, op-amp, 244–246, *244*
noninverting input, 237
noninverting integrator, 273–274, *273*
non-sine signals
discrete-time models, 48–49, *49*
sum-of-sines approximation, 50–52
NOR gate, 301–304, *301*
normally closed (NC), 131
normally open (NO), 131
north pole, magnetic, 98, *98*
NOT gate, 296–297
NPN transistor, 160–161, *160*, 219–222
n-type dopant, 12–14
n-type material, 108
n-type semiconductor, 145–146
nuclear fusion, 4
nucleus, 3
NXP, 362

O

octal SPI, 369, *369*
OCXO (oven-controlled crystal oscillator), 348
odd harmonics, 50–51
ODROID, 362
off capacitance, 206
ohm, 22
ohmic region, 154
Ohm's law, 22–23
battery internal resistance, 104
capacitor charging analysis, 190–191, *190*
current limiter analysis, 183
defining resistor behavior, 112
ideal voltage-source limits, 102
LED resistor sizing, 151, *151*
parallel resistor equivalence, 179
transimpedance amplifier formula, 249, *249*
voltage follower load, 221, *221*
Zener current limiting, 149
OLED, 151
Omnifixo clamp, 422
one-hot counter, 322, *322*
OPA1656, 242
OPA2323, 242
op-amp adder, 268
open-circuit voltage of solar cells, 108, *108*
open license, 411
`openocd`, 364
operational amplifiers (op-amps). *See also* active filters; mathematical operations; oscillators; voltage amplifiers
AC characteristics, 239–241
current-to-voltage amplifier, 249–253, *249*
DC characteristics, 238–239
defined, 235
IC packages and symbols, 237–238
old vs. new designs, 241–242
overview, 235–237

optocoupler, 286
Orange Pi, 362
orbitals, 4
OR gate, 215–216, 298–299, *299*
oscillation frequency, 232–233
oscillators, 230, 285–286
  digital-analog hybrid, 346
  phase-shift, 256–259, *257*
  relaxation, 253–256, *254*
oscilloscope, 64, 72
OSH Park, 394
output capacitance, 163
output characteristics, 218
output loading problem, 208
oven-controlled crystal oscillator (OCXO), 348
overline notation, 324
overload recovery time, 241
oversampling averaging DACs, 336–338
overvoltage protection, 205–206, *206*

P

p (pi), 34–35
parallel 8080/6800 protocols, 367–368, *368*
parallel L-C circuit, 180
parallel sum operator, 178
parametrons, 111
parasitics
  capacitors, 120–121
  inductors, 123–124, *123*
  JFETs, 156
  resistors, 114–116
part number, 169
passive components, 111
Pauli exclusion principle, 5
PC (program counter), 354–355
PCB cans, EMI shield, 406
PCB clamp, 422
PCB design
  board setup and layer navigation, 395–396
  design validation, 398–399, *399*
  hobby vendors, 394–395
  nets and layout, 396–397
  overview, 393–394
  placing footprints, 396
  production files and process choices, 399–400
  schematic capture, 395
  trace routing, 397–398
PCBWay, 394
p-channel JFET, 155
p-channel MOSFET, 218–220, *218*
peak amplitude, 30
peak-to-peak amplitude, 30
perfboards, 422
periodic waveforms, 29
peripherals. *See also* digital port I/O
  analog accessories, 371–372
  memory-mapped registers, 366
  overview, 365
permanent magnets, 97, 134, 136
pF (picofarad), 54
phase error detector, 349–350, *349*
phase-locked loop (PLL), 350
phase shift, 180, 240
phase-shift oscillators, 256–259, *257*, 347
phosphorus, 12
photocurrent, 152
photodiodes, 152, 251
photons, 71
  defined, 8
  in photodiodes, 152
  in semiconductors, 12
  in solar cells, 108, *108*
photoresistors, 113
phototransistors, 163
pi (p), 33–35
PIC32CM, 361, 401
PIC32CZ, 401
pickup pattern, 142
picofarad (pF), 54
Pierce oscillator, 347–348, *347*
piezoelectric crystals, 138–141, *141*
piezoelectric transducer, 139
Pinecil, 419
pin numbering, 238
pinout, AVR128DA28, 360, *360*
pipelined ADCs, 339–340, *340*
pipeline stall, 356
pipelining, 356
PLL (phase-locked loop), 350
plug and socket, 130
p–n junction
  body diode (MOSFET), 218
  in circuit protection, 206, *206*
  depletion region, 13–14
  forward and reverse bias, 146–147
  in solar cells, 108, *108*
PNP transistor, 163, *163*, 219
point-of-load (PoL) modules, 275
polarity-detecting switch, 219–220
polarity-protection switch, 219–220, *219*
polyethylene, 73

polyfuse, 144
PORTA.IN register, 366–367
PORTn.IN register, 366–367
ports, 366–367
positional notation, 334
positive feedback, 240
positive temperature coefficient (PTC) thermistor, 144
potential energy, 39
potentiometer, 112–113, *113*
potting, 411
power
  defined, 21–22
  RMS dissipation calculations, 31
power conditioning, 201
power ratings, 113–114
power supplies
  AC adapters, 106–108
  batteries, 102–106
  ideal, 101–102
  schematic symbols, 109–110, *109*
  solar cells, 108–109, *108*
  wireless and thermoelectric, 109
power supply rejection ratio (PSRR), 241
power supply unit (PSU), 417, *417*
primary winding, 127
principal quantum number, 5
priority encoder, 339
program counter (PC), 354–355
programming dongle pinout, 374–375, *374*
program storage memory, 354
propositional logic, 305
protons, 3
prototyping boards, 420–422, *422*
PSRR (power supply rejection ratio), 241
PSU (power supply unit), 417, *417*
PTC (positive temperature coefficient) thermistor, 144
p-type dopant, 12–14
p-type layer, 108
p-type semiconductor, 146
PUI Audio AOM-5024L-HD-R, 252
PULLUPEN flag, 367
pull-up resistor, 367
pulse-width modulation
  duty cycle control, 277, *277*
  MCU timer-counter implementation, 369–371, *369*
  switching losses, 278
pushback voltage, 54
push button input, 378–379
push-pull stage, 223–224, *223*
push-to-advance LEDs, 378–379
PWM audio toy, 381–384, *382*
PWM DAC, 337, *337*
Pythagorean theorem, 33–34
  combined impedance magnitude, 180
  complex-number magnitude, 199–200
  identity derivation, 47
  series impedance magnitude, 66–67
  sine-wave RMS, 47–48
  sinusoidal motion analysis, 42, *42*
  spring-block energy plot, 40

Q

Qalculate, 77
Q factor, 124, 203, 265, 267
quad flat package (QFP), 390
quadruple word, 64-bit, 307
quad SPI (QPI), 369, *369*
quantization, 332, *332*
quantum mechanics, 4–5
quarks, 4
quarter-wavelength lines, 82–84
quartz crystal, 139–140, *140*, 346–348

R

R-2R DAC, 334–336, *334*
radians, 34–35, *35*
radians to degrees, 68
radio-frequency (RF) engineering, 84
radio noise, 127
radio receiver, 89
radio waves
  antenna design, 84–86, *85*
  beat frequencies, 89–92, *90*
  modulation schemes, 86–89
  superheterodyne receivers, 92–93
Radxa ROCK, 362
rail-to-rail input, 236
rail-to-rail output, 239
Raspberry Pi, 362
ratsnest view, 396–397, *397*
R-C and R-L series impedance, 65–70, *65*
R-C filters, 192–201, *193*, 240, 257–258, *258*
R-C oscillators, 346
RC time constant, 192
reactance, 61–62, 178–179
real part, 182–183
rechargeable batteries, 106
Recom RFM series, 282
rectification, 209–212

rectifier, 147
reed switch, 132, *132*
reference signal, 185
register, 354–355
relaxation oscillators, 253–256, *254*, 279
relays, 135–136, *135*
reliability
  cable strain relief, 410
  environmental protection, 410–411
  polarity and ESD protection, 410
  serviceability, 411–412
  supply testing, 409–410
Renesas, 362
resettable fuse, 144
residual strong force, 4
resistance
  Joule's law, 23–24
  Ohm's law, 22–23
  of traces, 397
resistance temperature detector (RTD), 113
resistor noise, 116, *116*
resistors, 22
  as current limiters, 183–184
  as current sensors, 184–185
  fixed, 112
  Johnson–Nyquist noise, 116–118
  overview, 111–112
  parasitics, 114–116
  power ratings, 113–114
  pull-ups and pull-downs, 188–189
  testing Joule's law, 23–24
  variable, 112–113
  as voltage dividers, 185–186, *185*
  as voltage shifters, 186–188, *186*
resonant frequency, 64, 86, 139, 202–203, 346–347
RESSEL field, 366
reverse bias, 146–147, 206
reverse breakdown voltage, 147
reverse-polarity protection, 206, *206*
reverse recovery time, 148
RF (radio-frequency) engineering, 84
RG-58, 72
RG-8X, 72
rheostat, 113
right-angled triangle, 32–35
  in angle-sum identity, 90–92, *91*
  for impedance geometry, 66–67
  for RMS calculation, 47
  in spring-block analysis, 39–42, *42*
right-hand rule, 98
Rigol DM3058E multimeter, 417, *417*
Rigol DP711 power supply, 417, *417*
Rigol DS1202Z-E oscilloscope, 416, *416*
ring counter, 322–323, *322*
ripple, 106–107, *107*
ripple counter, 323–324, *323*
RISC-V, 361
R-L-C filter
  Q factor, 203
  in Sallen–Key design, 265
RMA (rosin mildly activated) flux, 419
rocker switch, 131
root-mean-square (RMS) voltage
  derivation, 30–32
  noise summation, 117
rosin mildly activated (RMA) flux, 419
rotary encoders, 132–133, *133*
rotor, 136–137
RP2350, 362
RPM (rotations per minute), 137
RTD (resistance temperature detector), 113
Rust programming language, 363

S

SAC305 lead-free solder, 419
safety in lab, 424–426
Sallen–Key topology, 264–268, *267*
salts, 10
saturation region
  BJT, 163
  JFET, 154, *154*
SBCs (single-board computers), 362–363
SC-70 package, 389
SC-90 package, 389
schematic capture, 167
Schematic Editor, KiCad, 395
schematics
  component drawing styles, 169–170
  connection symbols, 168–169
  layout and logical flow, 170–172, *170*
  op-amp symbols, 238
  overview, 167
  power supply symbols, 109–110, *109*
Schmitt trigger, 231–232, *231*, 297, 346, *346*
Schottky diode, 149
SCK line (SPI), 368
Scott, Hunter, 409
SCRs (silicon-controlled rectifiers), 164
Seacord, Robert C., 363
secondary winding, 127
second-order filter, 204

select signals, 354–355
self-discharge, 120
semiconductors, 102. *See also* discrete semiconductors
  doped, 12–13
  elemental, 11–12
  field-effect transistors, 14–16
  p–n junction, 13–14, *13*
sensors, 288
sequential operation
  counters, 321–325
  data routing, 325–330
serial peripheral interface, 368–369
serial protocol, 368
serial transmission, 328
series L-C circuit, 180–181
series voltage reference, 208, *208*
SHDN (shutdown pin), 238
Shenzhen PCB industry, 394
shift register, 323, 329–330, *329*
shrink small outline package (SSOP), 390
shunt voltage reference, 208, *208*
shutdown pin (SHDN), 238
SI (International System of Units), 18
Siglent SDG1022X Plus function generator, 418
Siglent SDM3055 multimeter, 417, *417*
Siglent SDS1202X-E oscilloscope, 416, *416*
Siglent SPD1305X power supply, 417, *417*
signal followers
  current mirror, 224–225, *225*
  push-pull stage, 223–224, *223*
  voltage follower, 220–223, *221*
signal generator, 64, 72, *72*
signal reflections
  general discussion, 77–80
  impedance mismatch causing, 81–82
  quarter/half-wavelength lines, 82–84, *84*
signals. *See also* sinusoids
  amplitude, 30
  frequency, 29
  non-sine, 48–52
  overview, 27–29
  root-mean-square voltage, 30–32
signal-to-noise ratio (SNR), 143
sign conventions, 24–26
Signetics, 285
silicon, 11–13, 145
silicon-controlled rectifiers (SCRs), 164
silicon dioxide, 15
silkscreen layer, PCB, 394
silver oxide battery, 105, *105*
sine waves
  and beat frequencies, 89–92, *90*
  and radio modulation, 86–89
single-board computers (SBCs), 362–363
single pole (SP), 131
single-slope ADC, 340–341, *340*
single throw (ST), 131
single-wire debug (SWD), 364
sinusoids. *See also* signals
  Pythagorean identity and sine-wave RMS, 46–48
  simple harmonic motion example, 36–43, *36*
  sine rate of change, 44–46, *44*
  sine wave basics, 32–36, *33*
SI prefixes, 19
Sir Box-a-Lot project, 394, *394*
slew rate, 241
slide-on strain relief, 410
slide switch, 131
slope-based ADC, 340–341, *340*
small outline integrated circuit (SOIC), 389–390
small outline transistor (SOT), 389
SNR (signal-to-noise ratio), 143
socket, 130
SOIC (small outline integrated circuit), 389–390
SOIC-8 package, 237
Sokoban, 394, *394*
solar cells, 108–109, *108*
soldering equipment
  irons, 418–419
  wire and accessories, 419–420, *420*
solderless breadboard, 421–422, *422*
solder mask, PCB, 394
solder wire, 419
solenoids, 133–134
SOT (small outline transistor), 389
sound pressure level, 143
south pole, magnetic, 98, *98*
speakers, 134–135, *135*
special relativity, 95
spectrum, frequency, 87–88
speculative fetching and execution, 356
speed of light, 7
SPI (SCK line), 368
split-terminal symbol, 110
spray-on coating, solvent risk, 410–411
spring constant (*k*), 37–39, *37*

springs
energy conservation model, 37–39
sinusoidal motion derivation, 42–43
state circle, 40–41
square wave
Fourier approximation, 50–51
introduced, 28–29
from op-amp oscillator, 258, *258*
through R-C filter, 198, *198*
SRAM (static random-access memory), 318–320
S-R latch, 313–315, *314*
SRL field, 402
SSOP (shrink small outline package), 390
stable equilibrium, 242–243
standard deviation, 117–118
standing wave, 83
state circle, 39–43, *42*
static electricity, 5
static random-access memory (SRAM), 318
stator, 136–137
steering diode, 206–207, *207*
stepper motor, 138
ST-LINK/V2, 364
STM32CubeIDE, 364
STM32H7, 362, 401
STM32L0, 361
stop-and-go conduction, 12
successive approximation register ADCs, 341–342
summing amplifier, 268–270, *270*
sum-of-sines approximation, 50–52
supercapacitor, 119
superheterodyne receivers, 92–93
supervisory circuit, 289
supply decoupling, 403, *403*
supply pin labels, 238, *238*
supply regulation
boost converters, 282–284, *283*
buck converters, 280–282, *281*
charge pumps, 278–280, *278*
linear regulators, 275–277, *276*
pulse-width modulation, 277–278, *277*
supply ripple, 208–209
supply voltage for op-amps, 238–239, *238*
surface-mount components
common two-terminal components, 388–389
integrated circuit packages, 389–390
leadless packages, 390
overview, 387
transistor packages, 389
surface-mount device (SMD), 421, *421*
surface-mount resistor, 114
surface-mount technology (SMT), 387, 421, *421*
sustained action, 131
switched-capacitor filters, 286–288, *287*
switches
bounce, 132, *132*
common-source switch, 218–219
overview, 130
polarity-detecting switch, 219–220
ratings and architecture, 131
as sensors, 132–133
switching regulators, 106, 281
synchronous counter, 325, *325*
synthetic input voltage, 335–336

T

tangent function, 68
TCA0 timer-counter, 369–370, *370*
Techni-Pro 758TW265 tweezers, 420
Teledyne LeCroy, 416
temperature gradient, 109
temperature testing, 411
Tequipment, 424
Texas Instruments, 272, 286
thermal noise, 116, 241
thermal shutdown, 276
thermal vibrations, 12
thermionic emission, 165
thermistors, 113, 144
thermocouple, 109
thin quad flat package (TQFP), 390
threshold voltage (Vth), 218
through-hole technology (THT)
prototyping with, 420–421
vs. surface mount, 387, 390–391
throw, switch, 131
thyristor, 164
timers, 285–286
timing signals, 345
TL431, 208
TLC04CD, 286
TO-236 package, 389
TO-252 package, 389
tolerance, manufacturing, 112
tools
digital multimeter, 416–417, *417*
digital oscilloscope, 415–416, *416*

tools *(continued)*
function generator, 418
lab power supply, 417, *417*
lab safety, 424–426
prototyping boards, 420–422, *422*
soldering equipment, 418–420, *419*
sourcing tools and components, 424
workholding and wire handling, 422–424, *423*
touch-sensing circuit, 159–160, *159*
toxic elements, 424–425
TPS561243, 282
TQFP (thin quad flat package), 390
trace length matching, 407–408
trace resistance specification, 397
trace routing, 397–398
transfer function H(*f*), 199–200
transformers, 94, 109, 127–128
transient response, 107, *107*
transient voltage suppressors (TVSes), 149–150, *149*, 410
transimpedance amplifier
designing microphone amplifier, 251–253, *252*
frequency response, 249–251
transistor packages, 389
transistors. *See also* bipolar junction transistors
as amplifiers, 225–229
bipolar junction transistors (BJTs), 160–164, *160*
field-effect transistors, 14–16
junction field-effect transistors (JFETs), 153–156, *156*
metal-oxide-semiconductor fets (MOSFETs), 156–160, *159*
overview, 217
relaxation oscillator, 230–234, *230*
as signal followers, 220–225
as switches, 218–220
transmission lines, 80
triangular waveform, 49, *49*
triboelectric effect, 5
trigonometric functions, 32–33
trim pots, 113
triode, 165
truth tables, 295–296
TSOP package, 390
Turing completeness, 355
Turing tarpit, 355
turn ratio, 127
TVS diode, 219
TVSes (transient voltage suppressors), 149–150, *149*, 410
two-digit adder, 307–308, *308*
two-terminal components, 110, 388–389

U

`uint8_t` type, 378
ULN2003 transistor array, 286
undervoltage condition, 409
unified program and debug interface (UPDI) pin, 360–361
unit circle, 34, 44–45
UNI-T UDP1306C power supply, 417, *417*
UNI-T UPO1202CS oscilloscope, 416, *416*
UNI-T UT139C multimeter, 417, *417*
UNI-T UTG962E function generator, 418
unity-gain bandwidth, 240
unity-stable amplifier, 240
UNIVAC, 355
universal gates, 302–304, *302*
universal solvent, 10
USB-C, 130
USB programming dongles, 363–364

V

V_{GS}, 218, 278
V_{th} (threshold voltage), 218
vacuum tubes, 8, *8*, 164–166, *164*
valence band, 11–12
valence electrons, 5
vanilla semiconductors, 12
variable resistor, 102
VCO (voltage-controlled oscillator), 255–256, *255*, 350
velocity, 38–39
dielectric effects on, 73–76, *75*
and inductance/capacitance, 76–77
measurement of, 71–73
vendor lock-in, 361
vibration and shock tolerance, 411
V-I curve, 146
voltage, 20–22, 23
across capacitor, 62
across inductor, 62
capacitor voltage vs. current, 55
DC, 28, 110
measuring, 25, *25*
root-mean-square voltage, 30–32
signs representing, 24–26

voltage amplifiers
  bandwidth control, 248–249
  inverting amplifier, 247–248, *247*
  noninverting amplifier, 244–246, *245*
  voltage follower, 242–244, *242*
voltage comparator, 228, 339
voltage-controlled oscillator (VCO), 255–256, *255*, 350
voltage divider, 216, 245–246, *245*
voltage doubler circuit, 213, *213*
voltage follower
  op-amp, 242–244, *242*
  transistor (common drain), 220–223
voltage-halving charge pump, 279–280, *279*
voltage reference, 149, *149*
voltage-to-voltage amplifiers, 251
voltmeter, 24

W

wall wart, 102, 409
WarGames, 402
water, 10, *10*
watt (W), 20
wave function, 4–5
wavelength, 76, 80
WCH CH32V003, 361
Weller T0051386499 solder wire, 419
Weller WT1010M iron, 419, *419*
`while` loop, C, 371
Wi-Fi, 362
wireless power, 109, *109*
wire stripper, 423, *423*
workholding, 422–424

X

X_C (capacitive reactance), 115, 156
X_L (inductive reactance), 114
XNOR gate, 302, *302*
XP Power RBT05W, 282

Z

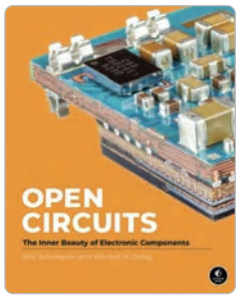
Zener diode, 149, 206–208, *207*
zepto- prefix, 19
zero crossings, 44–45, *44*, 58
zero-drift op-amp, 241
zero-indexed counter, 370, *370*
Zig programming language, 363
zinc, 102–103

The fonts used in *The Secret Life of Circuits* are New Baskerville, Futura, The Sans Mono Condensed, and Dogma. The book was typeset with LaTeX 2$_\varepsilon$ package `nostarch` by Boris Veytsman with many additions by Alex Freed, Miles Bond, and other members of the No Starch Press team *(2023/07/19 v2.4 Typesetting books for No Starch Press)*.

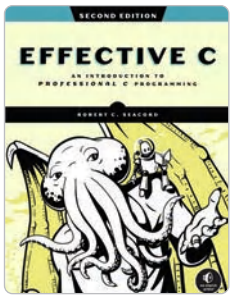
RESOURCES

Visit *https://nostarch.com/secret-life-of-circuits* for errata and more information.

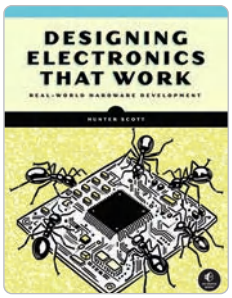
More no-nonsense books from **NO STARCH PRESS**



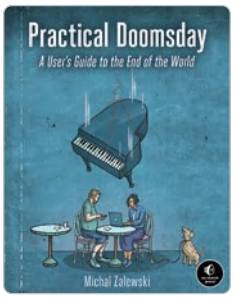
OPEN CIRCUITS
The Inner Beauty of Electronic Components
BY WINDELL OSKAY *AND* ERIC SCHLAEPFER
304 PP., $39.99
ISBN 978-1-7185-0234-5
full color



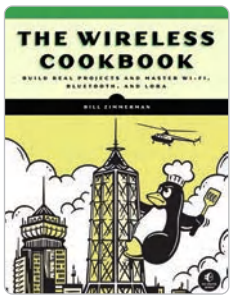
EFFECTIVE C, 2ND EDITION
An Introduction to Professional C Programming
BY ROBERT C. SEACORD
312 PP., $59.99
ISBN 978-1-7185-0412-7



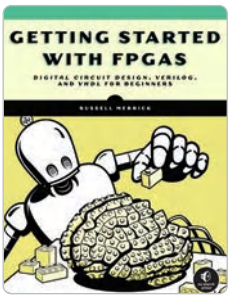
DESIGNING ELECTRONICS THAT WORK
Real-World Hardware Development
BY HUNTER SCOTT
360 PP., $49.99
ISBN 978-1-7185-0336-6



PRACTICAL DOOMSDAY
A User's Guide to the End of the World
BY MICHAL ZALEWSKI
250 PP., $24.99
ISBN 978-1-7185-0212-3



THE WIRELESS COOKBOOK
Build Real Projects and Master Wi-Fi, Bluetooth, and LoRa
BY BILL ZIMMERMAN
456 PP., $59.99
ISBN 978-1-7185-0436-3



GETTING STARTED WITH FPGAs
Digital Circuit Design, Verilog, and VHDL for Beginners
BY RUSSELL MERRICK
320 PP., $49.99
ISBN 978-1-7185-0294-9

PHONE:
800.420.7240 or
415.863.9900

EMAIL:
SALES@NOSTARCH.COM

WEB:
WWW.NOSTARCH.COM



Never before has the world relied so heavily on the Internet to stay connected and informed. That makes the Electronic Frontier Foundation's mission—to ensure that technology supports freedom, justice, and innovation for all people—more urgent than ever.

For over 30 years, EFF has fought for tech users through activism, in the courts, and by developing software to overcome obstacles to your privacy, security, and free expression. This dedication empowers all of us through darkness. With your help we can navigate toward a brighter digital future.

ELECTRONIC FRONTIER FOUNDATION

LEARN MORE AND JOIN EFF AT EFF.ORG/NO-STARCH-PRESS

ELECTRONIC CIRCUIT DESIGN, DRAWN BY HAND AND EXPLAINED FROM FIRST PRINCIPLES

The Secret Life of Circuits teaches electronic circuit design the way a working engineer actually thinks about it: starting with the physics, using math where it earns its place, the workbench never far away. Across nearly 300 hand-drawn color illustrations, Michal Zalewski builds intuition from the ground up, showing how electrons move through conductors and semiconductors, what's really going on inside op-amps and current mirrors, how digital logic works at the gate level, and how to carry a design from schematic through PCB fabrication to working firmware.

Photographed experiments. Real component behavior. The practical problems that show up in your own designs. Just the right balance of practice and theory: no Laplace transforms and no plumbing analogies.

Inside, you'll learn how to:

- Reason about real components, including the parasitics, noise, and reflections that ideal models hide
- Understand analog building blocks like amplifiers, oscillators, and filters at the device level
- Work with digital logic, memory, and clocking, and see how a rudimentary computer comes together
- Drive peripherals from a microcontroller and bring an embedded project to life
- Lay out high-speed PCBs with surface-mount components and get them to work on the first try

The Secret Life of Circuits is for inquisitive hobbyists who hit a wall in the usual books, and engineers who want to close the gap between the classroom and where the work actually happens.

MICHAL ZALEWSKI is a computer security specialist and an avid tinkerer with more than three decades of experience in electronic circuit design. He runs a popular, in-depth electronics blog and has published a number of hardware projects featured on Hackaday and in other venues. He spent more than a decade at Google building its product security program and later served as VP of security & privacy engineering at Snap Inc. He is also the author of two classic information security books—*The Tangled Web* and *Silence on the Wire*—along with *Practical Doomsday*, a guide to level-headed emergency preparedness (all from No Starch Press).



no starch press®

THE FINEST IN GEEK ENTERTAINMENT™
nostarch.com | san francisco