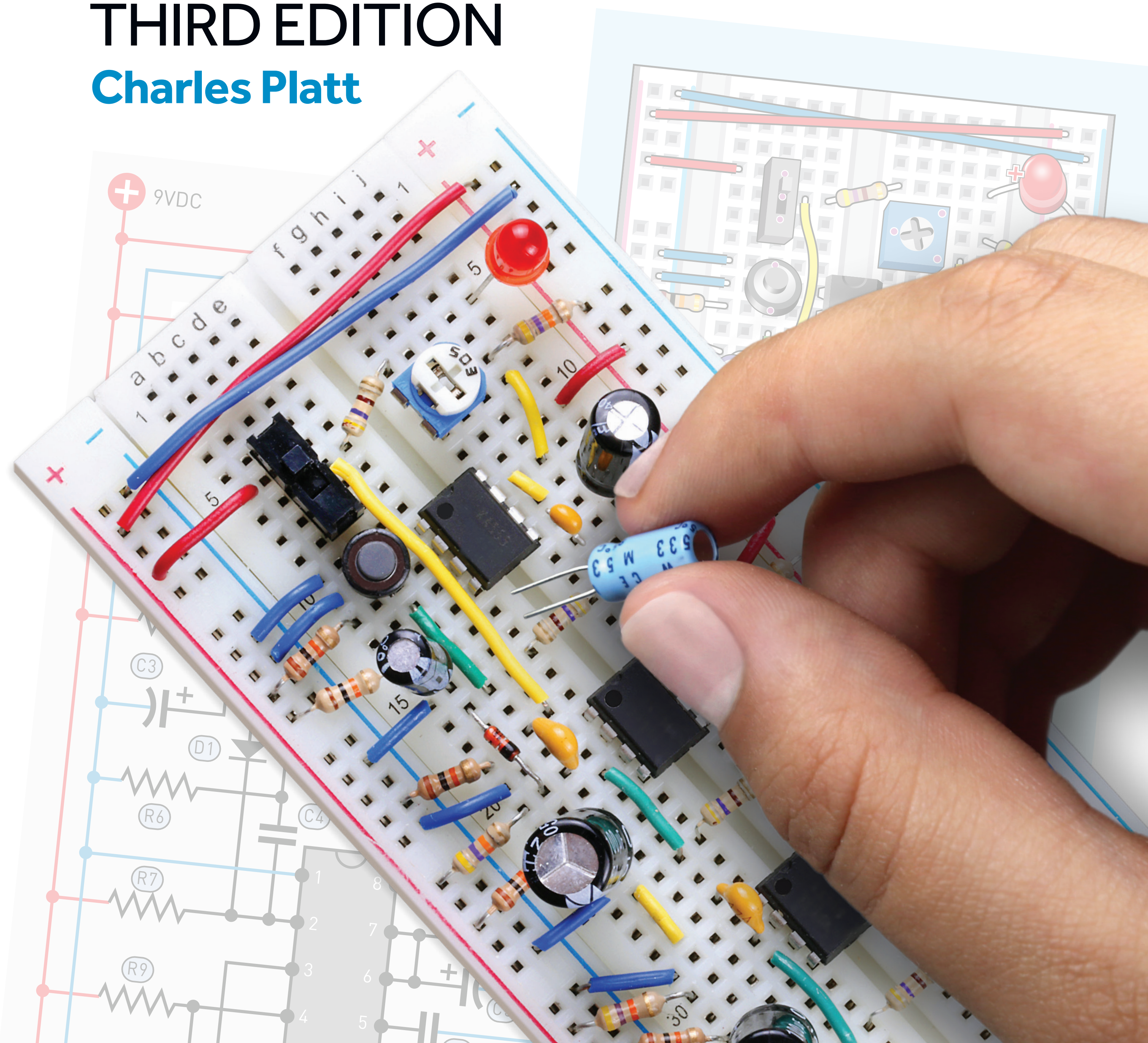
Burn Things Out, Mess Things Up — That's How You Learn.

Make: ELECTRONICS

THIRD EDITION

Charles Platt



TECHNOLOGY & ENGINEERING / Electronics / Circuits / General

Make: ELECTRONICS THIRD EDITION

"This is teaching at its best." —Hans Camenzind, inventor of the 555 timer, the most widely used integrated circuit chip in history

A "magnificent and rewarding book... expertly illustrated with photos and crisp diagrams... This really is the best way to learn." —Kevin Kelly, in Cool Tools

Make: Electronics revolutionized intro-level guides with the concept of "learning by discovery" in 2009 and has sold more than 200,000 printed copies in the United States alone. Now this Third Edition has made the best book even better.

Beginning with the most basic concepts, you can learn from your own hands-on experiments, using affordable parts and tools.

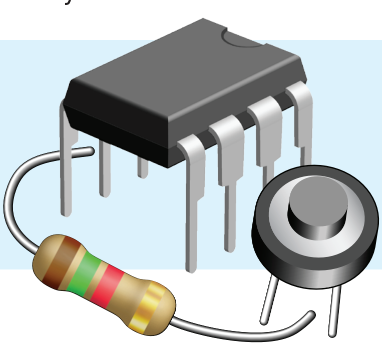
Along the way you can blow a fuse, make a relay buzz, and burn out a light-emitting diode. In *Make: Electronics* there's no such thing as a failed experiment because all experiments are a valuable learning process.

Within a few hours, you'll build a reflex tester, an intrusion alarm, a quiz game, or a combination lock — and modify them to do much more.

After learning the basics of voltage, current, resistance, capacitance, and inductance, you'll discover fundamentals of logic chips, radio, microcontrollers, and electromagnetism. Each project fits on a single breadboard, and most require no soldering.

All of the experiments use safe, low voltages, mostly supplied by a single 9-volt battery.

Today, *Make: Electronics* has attracted readers of all ages, from 10-year-olds to retirees who finally have free time in which to satisfy their curiosity about electronics.



Charles Platt is a contributing editor to *Make:* magazine, and was a senior writer at *Wired* magazine. He became hooked on electronics when he built his own telephone answering machine at the age of 15. He has said, "This is the book I wish I could have read when I was a teenager."

US $34.99 CAN $46.99
ISBN:978-1-68045-687-5
9 781680 456875 53499>

Make:

THIRD EDITION

Make: Electronics

Charles Platt

Make: Electronics

Third Edition

By Charles Platt

Printed in the United States of America.

Published by Make Community, LLC
150 Todd Road, Suite 100, Santa Rosa, CA 95407

Make: books may be purchased for educational, business, or sales promotional use.
Online editions are also available for most titles.
For more information, contact our corporate/institutional sales department: 800-998-9938.

Publisher: Dale Dougherty
Editor: Patrick Di Justo

October 2021: First Printing of the Third Edition
Revision History for the Third Edition:
9/21/2021

See www.oreilly.com/catalog/errata.csp?isbn=9781680457124 for release details.

978-1-68045-712-4

Make: Community
150 Todd Road, Suite 100, Santa Rosa, CA 95407

How to Contact Us:
Please address comments and questions concerning this book to the publisher:
Make: Community, 150 Todd Road, Suite 100, Santa Rosa, CA 95407

You can also send comments and questions to us by email at books@make.co.

Acknowledgments

Many people assisted me during the writing and production of this book. I'm especially grateful to David Cursons, Jolie de Miranda, Assad Ebrahim, Brian Good, Paul Henley, Brian Jepson, Roger Stewart, and Frederick Wilson for sharing their knowledge and noticing my errors. Thanks also to Jeff Palenik for his Civil War game, and most of all to Fredrik Jansson, the most patient and insightful collaborator a writer could ever hope to find.

Cover and back cover design by Juliann Brown, who also provided guidance regarding the preparation and production of this book. Interior design, photographs, diagrams, and schematics are by Charles Platt.

Front cover photograph by Charles Platt of hand by Neon, assisted by C. Dawes, with thumbnail by Family Dollar.

My editor, Patrick DiJusto, gave encouragement. Dale Dougherty and Gareth Branwyn allowed me exceptional freedom to write the first edition of *Make: Electronics* in the way that I wanted to write it, before anyone had heard of "Learning by Discovery."

Dedication

This third edition is dedicated to the memory of Hans Camenzind, a brilliant designer of analog integrated circuits who came from Switzerland to the Bay Area in the early days of Silicon Valley. For a while he worked at Signetics, then quit to create the 555 timer entirely on his own. It became the most widely used integrated circuit in history, as many billions of copies were manufactured over a period of fifty years. Even now, it is used at some point by almost everyone who learns electronics.

Table of Contents

Introduction

How to Have Fun with This Book

Make: Electronics reverses the traditional system for learning. Instead of beginning with a theory and then suggesting an experiment to verify it, I prefer to begin with an experiment and then encourage you to figure out the theory. I call this system Learning by Discovery, and I like it for two reasons:

- It's more interesting.
- It's closer to the way in which science is done in the real world.

In experimental science, observations can lead to a new understanding of some natural phenomenon. Why shouldn't someone learning electronics enjoy a similar experience? Discovering how components work sounds more interesting, to me, than knowing the answer before you start.

The only disadvantage of my approach is that to get full value from it, hands-on projects are necessary. Fortunately, component suppliers have developed kits for this book so that you can obtain everything you need with one-stop shopping for a relatively modest price.

What's New in the Third Edition

The first and second editions of *Make: Electronics* have sold hundreds of thousands of printed copies, and there are several foreign-language editions. I've been surprised and delighted by this success, but my book will only continue to do well if it satisfies the needs of readers. With this in mind, I have created the Third Edition.

Much of the text has been rewritten.

Most of the schematics and diagrams have been updated. Breadboard layouts now use clearer images of components.

Suggestions for tools have been updated, partly in response to feedback from readers.

Clearer photographs have been used in many instances.

Some experiments have been revised in response to feedback from readers.

A couple of the projects have been redesigned to use fewer components in circuits that I think are now easier to understand.

The last three chapters introducing the Arduino have been revised, and I added an overview of other types of microcontrollers.

I worked with a leading supplier of kits for this book in an effort to reduce and simplify the range of components needed in the experiments, so that you will be able to pursue them at lower cost.

One consequence of these improvements is that kits for the Second Edition won't provide the exact range of components that you need for this Third Edition of the book. I will mention this repeatedly, because I don't want readers to be disappointed if they buy an old kit, only to find that it doesn't quite match the new text. Please look carefully for the words "Third Edition" if you buy a kit.

The Purpose of This Book

Everyone uses electronic devices, but many people are not clearly aware of what goes on inside them.

You may feel that you don't need to know. You can drive a car without understanding the workings of an internal combustion engine, so why should you learn about electricity and electronics?

I think there are three reasons:

- By learning how technology works, you become better able to control your world instead of being controlled by it. When you run into problems, you can solve them instead of feeling frustrated by them.
- Learning about electronics can be fun, so long as you approach the process in the right way. Also, it is affordable.
- Knowledge of electronics can enhance your value as an employee, or perhaps even lead to a whole new career.

Messing Things Up

One important aspect of Learning by Discovery is that you should expect to make mistakes. A circuit may not work, or you may burn out some components.

I think of this as a positive aspect, as mistakes are a valuable way to learn. I want you to burn things out and mess things up, to see for yourself the behavior and limitations of the parts that you are dealing with. The very low voltages used throughout this book may damage sensitive components, but they will not damage you.

Never be afraid to make errors. Transistors and LEDs are inexpensive and easy to replace.

Will It be Difficult?

I assume that you're beginning with no prior knowledge. Consequently, the first few experiments will be extremely simple, and you won't even use a prototyping board or a soldering iron.

I don't believe that the concepts will be hard to understand. Of course, if you want to study electronics more formally and do your own circuit design, that can be challenging. But in this book I have kept theory to a minimum, and the only math you'll need will be addition, subtraction, multiplication, and division. You may also find it helpful (but not absolutely necessary) if you know how to multiply and divide by 10 by moving decimal points from one position to another.

How This Book Is Organized

Most of the information is presented in tutorial form, with just a few sections that are intended for future reference.

I have introduced concepts and topics in a cumulative sequence. You can dip into the book at random, but the experiments in later chapters require knowledge that you gain in the earlier chapters, so I suggest that you proceed through them in numerical order, skipping as few as possible.

If Something Doesn't Work

Usually there is only one way to build a circuit that works, while there are hundreds of ways to make mistakes that will prevent it from working. Therefore, the odds are against a happy outcome if you don't work in a methodical manner.

I know how frustrating it is when components just sit there doing nothing, but if you build a circuit that doesn't work, getting annoyed with it is counter-productive. The only way to find the problem is by examining every detail systematically.

All of the experiments have been bench-tested, so I know that the circuits are good. If something doesn't work for you, these are the most likely problems:

- You made a wiring error. Everyone makes wiring errors; I made one myself, just today. Your chances of seeing the error will improve if you walk away from your work table for half an hour, and do something else before returning to take another look.
- You may have overloaded a component such as a transistor or a chip, so that it doesn't work anymore. Try to keep some spares, just in case.
- There may be a bad connection between a component and a breadboard. Try wiggling loose components, measuring voltages, and if necessary, moving key components to a slightly different location on the board.

I will have more detailed advice on fault-tracing later in the book. I'm mentioning the topic here because I need to advise you on your ultimate recourse if you can't get a circuit to work: Unlike most writers, I maintain an email address that you can use to contact me directly. All I ask is that you should follow some guidelines.

Asking a Question

My time is obviously limited, but I try to answer all messages. Please be patient. Sometimes I can reply the same day, but at other times I may take a week to respond.

If you contact me, please:

- Attach photographs of any project that doesn't work. I must be able to see details such as the colors of stripes on resistors.
- Tell me which project you have been working on, and mention the title of the book in which it appears. Bear in mind, I have written several books about electronics, so I need to know which one you are using.
- Describe the problem clearly! Tell me about the problem in the same style as if you were describing a physical symptom to a doctor and asking for a diagnosis.

Send your message to

`make.electronics@gmail.com`

and put HELP in the subject line.

Reporting a Mistake

When I'm writing a book, I have even more ways to make mistakes than when you are building a circuit. Naturally I do everything I can to minimize errors, but if you find one, please report it. You can use my personal email address for that purpose, or you can go to the "errata" page maintained by O'Reilly and Associates, who distribute this book. The advantage of writing to me is that I can respond personally to you and discuss the problem if necessary. The advantage of the O'Reilly system is that you can read other people's reports, and see if you have run across something that has already been resolved. Also, after you make a report to the O'Reilly web site, other people can read it. The O'Reilly site is here:

`www.oreilly.com/catalog/errata.csp?isbn=9781680456875`

Receiving Updates

Even if you don't have any problems or requests, I encourage you to register your email address with me. I will be able to use it for the following purposes:

- I will notify you if any significant errors are found in this book or in its sequel, *Make: More Electronics*, and I will provide workarounds.
- I will notify you of any errors or problems relating to kits of components sold in association with this book or in *Make: More Electronics*.
- I will notify you if there is a completely new edition of this book, or of my other books. These notifications will be only at intervals of one or two years.

I won't use your email address for any other purpose, and I won't sell it or share it with anyone. (I wouldn't actually know how to sell email addresses, or who might want to buy them.)

If you register your email address, I will send you an unpublished electronics project with construction plans as a two-page PDF. It will be fun, it will be unique, and it will be relatively easy. You won't be able to get this in any other way.

The reason I am encouraging you to participate is that if there's an error in my work, and I have no way to tell you, and you discover it later on your own, you're likely to get annoyed. This will be bad for my reputation, so I want to avoid a situation where you have a complaint.

Just send a blank email (or include some comments in it, if you like) to

`make.electronics@gmail.com`

Please put REGISTER in the subject line.

I have to process emails manually, because sometimes people want a personal reply, even when they are just registering. Do not expect an immediate automated registration process! If I go on vacation, you may not receive your "special bonus project" for a couple of weeks. But you will get it eventually. Delays are the inevitable consequence of me doing things on my own.

Going Public

If you get frustrated, you may want to complain, and one way in which people complain is in reader reviews, especially on amazon.com. If you want to do this, please contact me first to see if I can address your complaint.

Be aware of the power that you have as a reader, and please use it fairly. A single negative review can create a bigger effect than you may realize. It can certainly outweigh half-a-dozen positive reviews. In a couple of cases, people have been annoyed over small issues such as being unable to find a source for a component. I would have been happy to help them if they had asked me.

Online sales are my primary source of income, and my four-and-a-half-star rating is important. Of course, if you simply don't like the way in which I have written this book, you should say so.

Going Further

After you work your way through *Make: Electronics,* you will have grasped many of the basic principles involved. I like to think that if you want to know more, my sequel *Make: More Electronics* is the ideal next step. It is slightly more difficult, but uses the same "Learning by Discovery" method. My intention is that you will end up with what I consider an "intermediate" understanding of electronics.

I am not qualified to write an "advanced" guide, and consequently I don't expect to create a third book with a title such as *Make Even More Electronics.*

You may consider buying the reference books that I wrote: *The Encyclopedia of Electronic Components* is in three volumes, two of which were written in collaboration with a very smart researcher named Fredrik Jansson. The components are listed by category, so that if you look one up and it isn't exactly what you want, the very next one in the book—which you may have never heard of—could be the answer to your problem.

And just in case you know someone who is younger, with a short attention span, I wrote a much briefer book titled *Easy Electronics* which I like to think is the simplest possible introduction to basic ideas. A kit is available for that book, and the projects are so easy, you don't even need tools to assemble them. Imagine that: A hands-on book that doesn't require tools!

If you have an interest in fabricating things, I must mention my book *Make: Tools,* which is a guide to using hand tools, following the same hands-on approach as *Make: Electronics.* It begins by describing the use of a hand saw, and ends by showing you how to build little enclosures out of plastic—which could be just the thing for your electronics projects.

—Charles Platt

Section One

The Basics

This section contains experiments 1 through 5.

In Experiment 1, I want you to get a taste for electricity—literally! You'll experience electric current and discover the nature of electrical resistance.

In experiments 2 and 3 you'll use a meter to measure current and voltage, and in Experiment 4 you'll calculate wattage. Along the way you can burn out an LED, blow a fuse, and deduce a fundamental law in electronics.

Experiment 5 will be an entertainment, using everyday items to generate electricity on a tabletop.

These experiments will clarify some important concepts. Please give them a try before venturing into the rest of the book, even if you have some prior knowledge.

Necessary Items for Section One

Each section of this book begins with pictures and descriptions of the tools, equipment, components, and supplies that you will need. If you lack experience in buying some of these things, you'll find more details in Appendix A, beginning on page 290. If you need to know about where to find components and supplies online or in stores, sources are listed in Appendix B, beginning on page 299.

If you prefer not to buy your own components, at least two ***kits*** are currently available, containing parts that you need for projects in this book. The kits are created by independent suppliers, and I have no control over them or financial interest in them, but I have verified that the components are correct. The suppliers are listed in Appendix B.

Kit vendors may ship their products overseas, but unfortunately postage from the US to other countries is expensive because the United States Postal Service is not government-subsidized. If you live outside of the US, you may do better to buy components from Asian sources, where postal rates are lower and the components themselves are cheaper.

The Multimeter

A handheld ***multimeter*** is the most essential tool when you are learning electronics. It will tell you what's going on inside a circuit, just as an MRI machine tells a doctor what's happening inside the human body.

The "multi" in "multimeter" means that it can measure multiple functions, the most important ones being voltage, current, and electrical resistance. When electronics engineers refer casually to "a meter," they probably mean a multimeter. Initially it may appear complicated or intimidating, but really it's simpler than a modern phone, and no more difficult to use than a camera.

The type of meter you need is properly known as a ***digital multimeter,*** because it has a digital display. You can sometimes find an ***analog multimeter*** which moves a needle across a scale, but it's not so easy to use, and I don't recommend it.

One of the smallest, simplest meters that I have seen is shown in Figure 1-1. Its specification was set by one of the kit manufacturers for this book, referenced in Appendix B, but you can find similar meters online. If you want to minimize the amount that you spend, a product like this will be sufficient to take you through all the experiments from 1 through 30, and you can skip the rest of my discussion regarding meters. On the other hand, if you want to know how you may benefit by spending a little extra money, read on.

Auto vs. Manual

The most obvious feature that you can get in a more expensive meter is *auto-ranging*. To explain this, imagine that you want to measure temperature. If you're using an oven thermometer, you'll be happy if it's accurate within five degrees in a range from 200 to 500 degrees Fahrenheit. But if you want to measure body temperature, you'll want an accuracy of maybe 0.1 degrees in a narrow range from 95 to 105.

The situation is similar when measuring voltage or other values in electronics. Sometimes you're interested in low numbers and high accuracy, but other times you want high numbers, and you'll accept less accuracy.

A *manual-ranging* meter requires you to choose a range of values by turning a dial before you make the measurement. For example, to test the voltage of a 1.5-volt AA battery, you would set the meter to measure up to 2 volts, after which it will tell you the actual voltage with good accuracy.

An *auto-ranging* meter would sense the voltage and choose an appropriate range by itself. That sounds nice, and auto-ranging meters are becoming more affordable—but personally, I don't really like them. The meter takes a couple of seconds each time it tries to decide which range to use, and I tend to be an impatient person. Also, because you didn't select the range, you won't know immediately what the numbers mean on the display. Suppose you see `1.48`. Would that be volts or millivolts? The display will show a little V or mV to tell you, but if you forget to look, mistakes are possible.

- I suggest you should use a manual-ranging meter. You'll have fewer chances to make errors, it should cost less than a comparable auto-ranging meter, and it will be less frustrating, if you're impatient like me.

How do you tell if a picture on a web site is of an auto-ranging meter or a manual-ranging meter? If a meter does auto-ranging, usually the product description will tell you—but when in doubt, inspect the dial on the front. An auto-ranging meter won't have a lot of numbers, and may look like the example in Figure 1-2. A manual-ranging meter may look more like the one in Figure 1-3.

The rest of my discussion about meters will mostly refer to those that do manual ranging.

Figure 1-1. A bare-bones digital multimeter. The squares behind it are at intervals of 1 inch.

Figure 1-2. An auto-ranging meter.

Figure 1-3. A manual-ranging meter.

The Price

Offering advice on how much to spend when buying a meter is like advising someone on buying a car. The ratio between the price of the cheapest car and the price of the most exotic model may be around 100:1, and the same is true of meters. Also, prices may change over time.

I'll address this issue by referring to the meter in Figure 1-1 as the ***baseline model***. What will you gain if you buy a meter that costs more?

One answer may be longevity. I haven't used that particular meter for a prolonged period, but generally speaking, the contacts of the selector switch on the front of the meter may wear out over time. This may not matter to you if you don't know, yet, about your long-term interest in electronics.

You can also acquire more features by spending more money, but this is a difficult topic, because features entail some terminology. I haven't explained anything about voltage and amperage yet, let alone transistor testing—so I'll just show you the symbols and abbreviations that you are likely to see around the dial on the front of a meter, and I'll suggest which ones are important. You'll learn their exact meaning as you continue through the book.

In Figure 1-4, the items in red are essential. The ones in black are nice to have, but not essential for the experiments in this book.

Meter manufacturers are constantly coming up with additional features that look impressive, but many of them aren't very useful. Here are some examples that you don't really need:

- *NCV* means "no contact voltage" testing. When you hold the meter near an electrical outlet or a wire in your home, the meter will tell you if voltage is present. This is not relevant to *Make: Electronics*.
- *Temperature measurement.* The meter may be able to find out if a component is overheating, but for our purposes, touching a component with your finger will be good enough.
- *Max/Min* and *Hold* buttons. Useful if you are trying to capture a value that varies rapidly, but you are unlikely to be doing that.

Dial Positions

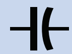
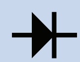
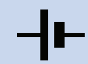
V	Voltage (electrical pressure).	A	Amperage (electrical flow).
Ω	Electrical resistance (in ohms).	mA	Milliamps (thousandths of an amp).
 OR F	Capacitance (in farads).	Hz	Electrical frequency (in hertz).
	Direct current (DC).		Alternating current (AC).
	Diode testing.		Battery testing.
 OR 	Continuity testing (the meter will beep).	hFE and/or NPN PNP	Transistor testing.

Figure 1-4. The most widely used symbols and abbreviations selectable on multimeters. Those in red are essential.

- *Backlighting* of the display. Generally you'll use a good desk lamp when working with components, in which case your meter does not need backlighting.

The six letters and symbols in the top half of Figure 1-4 are often preceded by ***multipliers***. For instance, **m** is a multiplier with a value of 1/1,000, so the term **mV** means 1/1,000 of a volt, which is a ***millivolt***. The Greek letter **μ** (pronounced "mew") is a multiplier with a value of 1/1,000,000, so the term **μA** means 1/1,000,000 of an amp, which is a ***microamp***. Multipliers are summarized on the next page in Figure 1-5.

- Note that a lowercase **m** means "divide by 1,000." Uppercase **M** means "multiply by 1,000,000." Try to avoid getting them mixed up!

At the bottom of Figure 1-5 I have shown the ranges that you may find in a meter. Some meters don't use range values beginning with 2; their values may begin with 4, as in 40, 400, 4K, and so on. Some meters have range values beginning with 6. For the experiments in this book, I don't feel there's a particular advantage either way.

Multipliers			
p	"pico" 1/1,000,000,000,000	**m**	"milli" 1/1,000
n	"nano" 1/1,000,000,000	**k**	"kilo" x 1,000
µ	"micro" 1/1,000,000	**M**	"meg" or "mega" x 1,000,000

Ranges						
V (volts) DC	200m	2	20	200		
V (volts) AC	200m	2	20	200		
A (amps) DC	200µ	2m	20m	200m	10 or 20	
A (amps) AC	200µ	2m	20m	200m	10 or 20	
Ω (ohms)	200	2K	20K	200K	2M	20M
F (farads)	2n	20n	200n	2µ	20µ	200µ

Figure 1-5. Multipliers and ranges that are often used by meters.

A wider range of values is nice to have, but may entail spending more money. I think the red values are the most important, while the ones in black are optional.

Where **F** (farads) is concerned, the range that I've shown is irrelevant if your meter doesn't measure this unit—but if it does, these are the values you would hope to find.

Now I'll show you a couple more meter dials to illustrate how the ranges appear in real life. In Figure 1-6 the panel around the circular selector is divided into sections, each identified with a letter or symbol such as **V** or **A**. **V** is the Volts section. This range applies to both AC and DC, selectable by a switch, which I have circled. If you aren't entirely sure what the difference is between AC and DC, this isn't important right now. Notice that the range from 200mV to above 200V is a little better than my suggested range in Figure 1-5.

Continue around the dial, and you will see that the important items from Figure 1-4 are all there, circled, and the ranges are complete. This meter looks like a good choice.

Now check Figure 1-7. This meter doesn't have a switch to select AC or DC. Instead, it provides dedicated positions on the dial. On the left, you see letter **V** with the symbol that means DC. On the right, another **V** has a wavy line beside it which means AC, and refers to the two white switch positions beside it. The lightning-bolt symbol simply means "caution," and indeed you should be cautious if you are planning to measure 600 volts, although I can't imagine when you might want to. Low voltages are more important for our purposes, and this meter doesn't have a low range for AC volts. I'm not happy about that, because you may want to measure fluctuating voltages sometimes—in the output from a timer chip, for instance.

Figure 1-6. The dial of a meter that is able to perform all the functions which I think are desirable in this book.

I don't see any provision for measuring AC amps. There's a letter **A** followed by the symbol for DC, but no letter **A** followed by a wavy line for AC. That's a bit disappointing.

Another drawback of this meter is that it cannot measure capacitance in farads. Wait—what about that letter **F**, which I circled? No, it has a degree symbol beside it, so it means "degrees Fahrenheit." That's a bit misleading, especially as the meter is not shipped with a temperature probe, so you have no way to use this feature.

Lastly, I have circled **2M**, which is the upper limit for measuring electrical resistance. Really 20M would be better. Overall, I'm not impressed by this meter. You could still use it for the experiments in the book, but it costs more than the baseline model in Figure 1-1 and doesn't offer significant additional value for money.

Figure 1-7. Inspecting this meter dial reveals that some desirable features are missing. See text for details.

So how much money should you be willing to spend? Look online for a meter such as the baseline model in Figure 1-1, and whatever it costs, think of that amount as $B. If you spend between twice and four times $B, you should be able to get all the features that I have recommended. The meter in Figure 1-3, which I bought for testing while I was writing this book, cost about $B x 3, and has done well. Moving up the price scale, the autoranging meter in Figure 1-2 cost $B x 6.

In figure 1-8 you see my favorite meter at the time of writing. Notice that it displays four digits. Some cheaper

Figure 1-8. This model costs about 20 times as much as the one in Figure 1-1.

meters have started to appear with four-digit displays, but an extra digit doesn't necessarily mean that the electronics inside the meter are ten times more accurate than in a 3-digit meter. You'd have to compare the manufacturer's specifications carefully to find out. For the purposes of this book, 4-digit accuracy is not necessary.

The only problem with the meter in Figure 1-8 is that it costs about $B x 20. I regard it as a long-term investment. I'm happy with its accuracy, and I'm hoping it will last for many years, but these considerations may not be important if you don't know, yet, how interested you are in electronics.

If you have read all the suggestions above, but you still feel unsure about which meter to buy, browse ahead a little to get an idea of how you will be using a meter in experiments 1, 2, 3, and 4. Then make your decision.

This concludes my dissertation on meters. Your other purchasing decisions will be simpler.

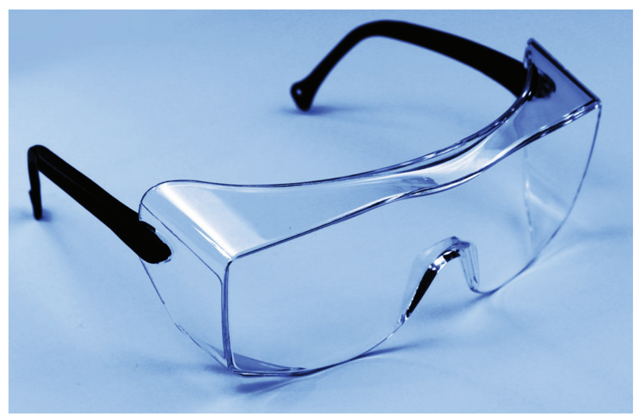


Figure 1-9. Safety glasses.

Safety Glasses

From time to time, there may be a slight risk to your eyes when you are working on electronics projects. For instance, when you are snipping a brittle piece of wire sticking out of a component such as an LED, a fragment could fly up toward your face.

Any cheap safety glasses will provide adequate protection, or regular eyeglasses are an acceptable substitute. Simple safety glasses are shown in Figure 1-9.

Test Leads

You will use *test leads* (pronounced "leeds") to connect components in the first few experiments. The type of leads I am referring to are *double-ended*.

Surely, any piece of wire has two ends? Yes, but in this case the term means that each end is fitted with an *alligator clip* as shown in Figure 1-10. Each spring-loaded clip can make an electrical connection by grabbing something and gripping it securely, freeing you to use your hands elsewhere. For the experiments in this book, very short leads are good, like the ones shown. Longer ones will work, but they tend to get tangled.

You don't want the kind of test leads that have a small single-pin plug at each end. Those are sometimes known as *jumper wires*.

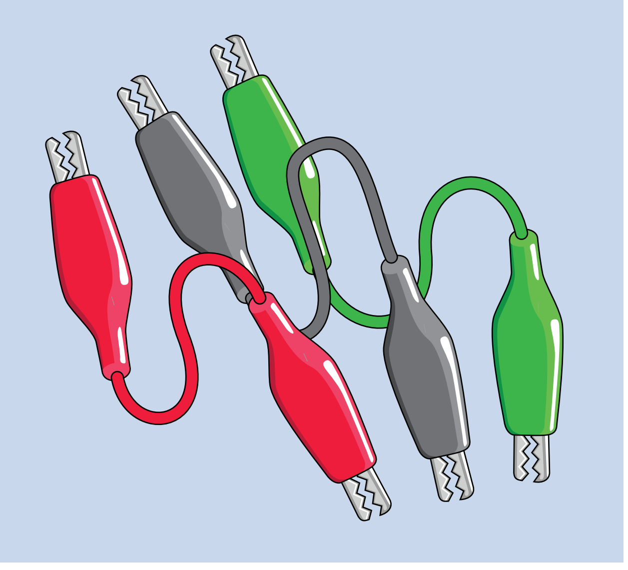


Figure 1-10. Test leads.

Power Supply

Almost all the experiments in this book will use a power source of 9 volts. You can obtain this from an everyday 9-volt alkaline battery of the type sold in supermarkets and convenience stores. It doesn't have to be a name brand. Later I'll suggest an upgrade to an *AC adapter*, but you don't need that right now.

A 9-volt battery has positive and negative terminals. Don't get them mixed up! If the positive terminal is not clearly identified, tag it with a red marker pen.

- Only use a 9-volt *alkaline battery* for experiments 1 through 4. Do not try to use a larger battery, or a battery that delivers more than 9 volts. Note that lithium batteries can be hazardous, and should not be used for any projects in this book.

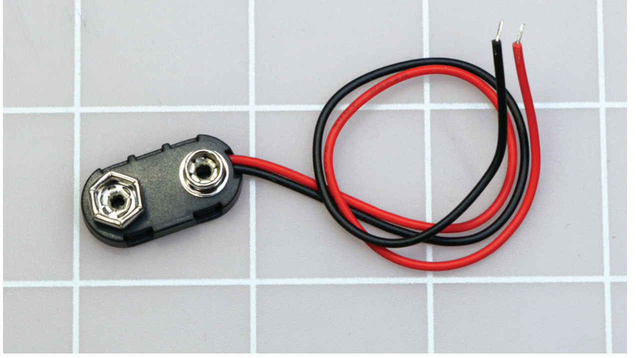


Figure 1-11. A connector for a 9V battery.

Battery Connector (optional)

My illustrations will show alligator test leads clipped onto the terminals of a 9-volt battery, but if you want to make a more secure connection, you can buy a connector which has snaps to fit your battery terminals and two wires with bare ends, as shown in Figure 1-11.

Fuse

A *fuse* interrupts a circuit if too much electric current passes through it. You will need a couple of glass cartridge fuses of the kind shown in Figure 1-12, or you can use automotive fuses available from auto parts stores. Either way, you will need one fuse rated for 1 amp and one rated for 3 amps (the cartridge type will have 1A and 3A engraved on their steel end caps, respectively). This illustration is a closeup view of a 2AG fuse that has a diameter of about 5mm.

Cartridge fuses are often rated for 250 volts, but any rating of 10 volts or higher will do. (The term "rating" means the maximum value the manufacturer thinks is appropriate for this product.)

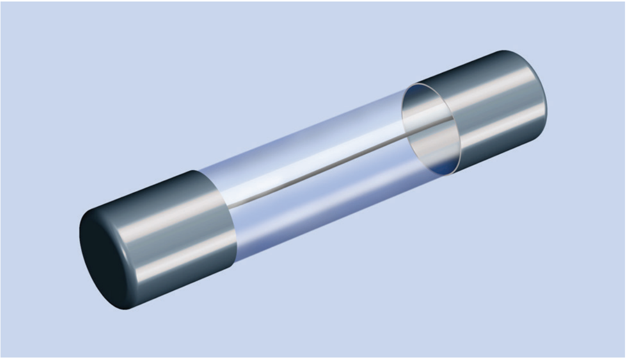


Figure 1-12. Closeup view of a 2AG fuse, 5mm in diameter.

Light-Emitting Diodes

More commonly known as ***LEDs***, they come in various shapes and forms. The ones we will be using are properly known as ***LED indicators***, and are often described as ***standard through-hole LEDs*** in catalogs. In the first two sections of this book, LEDs with a diameter of 5mm will be easier to handle, but I'm recommending 3mm LEDs for the remainder of the book, as they will be easier to fit into some of the circuits where components are crowded together. A typical red LED is shown in Figure 1-13.

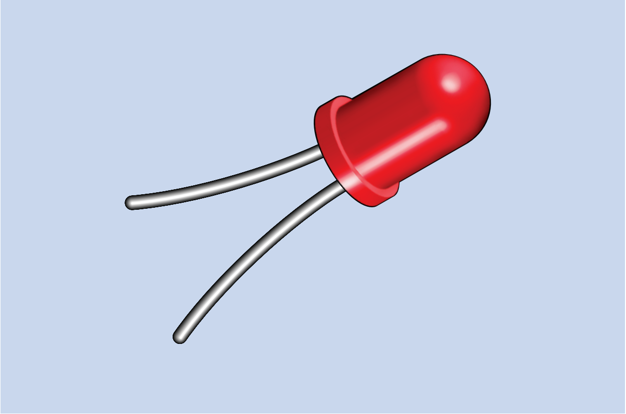


Figure 1-13. Closeup of a standard through-hole LED indicator.

Throughout this book I will often refer to ***generic red LEDs***. I want them to be red, because red LEDs will work with less current and a lower voltage than some other colors, which will be important in some experiments. By "generic" I mean the cheapest ones that are commonly available. They are used in so many applications, it's useful to keep at least a dozen.

Some generic LEDs are encapsulated in ***water clear*** plastic or resin, and may surprise you by emitting a color when power is applied. Other LEDs, such as the one in Figure 1-13, are known as ***diffuse***, as they are encapsulated in plastic or resin tinted with the same color that they will display. Water-clear LEDs are brighter, if all other aspects are the same, but I think diffuse LEDs are more pleasant to look at.

Resistors

You'll need a variety of ***resistors*** to control the voltage in various parts of a circuit. Two resistors are shown in closeup in Figure 1-14 (in real life, each of them would be less than 1/2" long). Later I will explain how the colored stripes tell you the value of each resistor. The color of the body of the resistor is not important for our purposes.

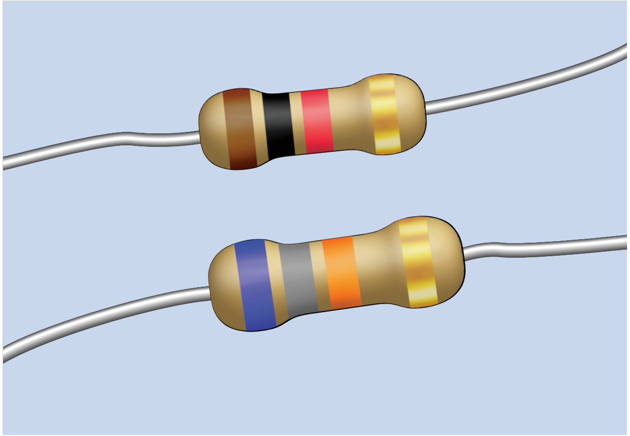


Figure 1-14. Two sample resistors.

If you are buying your own resistors, they are so small and cheap, you would be foolish to select just the two or three values listed in each experiment. Get a prepackaged selection in bulk from surplus or discount sources, or a site such as eBay. If you want to know exactly which values of resistors are required for each experiment in the book, check the tables in Appendix A.

Hardware

In Experiment 5, I'll be showing you how to make your own lemon-juice battery. You'll need some copper-plated pennies for this little project (or some other objects with a copper surface), and also some zinc-plated hardware such as mending plates about 1 inch long, like the one in Figure 1-15. Four will be sufficient, or small brackets will do instead. You can find them at any hardware store.

As for the pennies, new ones will work better than old ones, as they will be less tarnished. If you live in a part of the world where copper-plated coins don't exist anymore, I've suggested some other options in the buying guide in Appendix A.

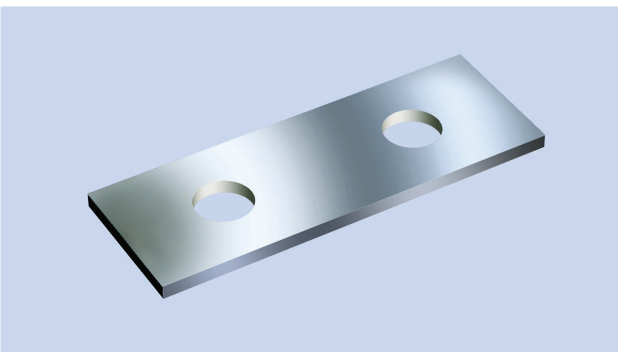


Figure 1-15. A zinc-plated mending plate. Try to find some that are about 1 inch long. Small brackets will do instead.

Experimenter's Notebook

Every time you conduct an experiment, you really need to keep a record of how you set it up and what happened. You can make your notes on a computer or using your phone, but an old-fashioned notebook with paper pages has some advantages. You don't have to open an application to update the entries, and it's safe from accidental data deletion. Keep it on the corner of your desk, and it may turn out to be more useful than you expect.

That's the last item on my list, so let's get started!

Experiment 1
Taste the Power!

Can you taste electricity? It feels as if you can. Using a battery-energized Tongue Test, this project will demonstrate electrical resistance.

You Will Need:

- 9V alkaline battery (1).
- Multimeter (1).

That's all!

Caution: No More than 9 Volts

A 9V alkaline battery won't hurt you. But ***do not*** try this experiment with a higher-voltage battery, and ***do not*** use a bigger battery that can deliver more current. Absolutely positively do not try to use a car battery or an alarm battery! Also, if you have metal braces on your teeth, be careful not to touch them with the battery.

Testing Your Tongue

Moisten your tongue and touch the tip of it to the metal terminals of a 9V battery, as shown in Figure 1-16. (Maybe your tongue isn't quite as big as the one in the picture. Mine isn't. But this experiment should work regardless of how big or small your tongue may be.)

Do you feel that tingle? Now set aside the battery, stick out your tongue, and dry the tip of it very thoroughly with a tissue. Touch the battery to your tongue again, and you should feel less of a tingle.

- What if you don't feel anything? A very few people seem to have unusually thick skin, or dry tongues, or perhaps both. A few have emailed me over the years to report that they didn't feel any tingle at all. If you have this problem, dissolve a pinch of salt in a few ounces of water, and moisten your tongue with it. That should do the trick!

What's happening, here? You can use a meter to find out.

Figure 1-16. An intrepid Maker tests the characteristics of a 9V alkaline battery.

Setting Up Your Meter

If you have a new multimeter, does it have a battery preinstalled? Select any function with the dial, and wait to see if the display shows a number. If the display window is blank, you may have to open the meter and put in a battery before you can use it. I can't tell you how to do this, or what type of battery you need, because the requirements of meters vary a lot. You'll have to check the instructions that should have been supplied.

Meters are supplied with two leads, one red and one black. I'll refer to them as ***meter leads*** to distinguish them from the test leads that you will also be using. Actually, the word "lead" can refer to almost any piece of wire connecting with a device or a component.

Each meter lead has a plug on one end and a steel probe on the other end, as shown in Figure 1-17. You insert the plugs into the meter, then touch the probes at locations where you want to know what's going on. The probes can measure electrical flow, or can detect voltage. The projects in this book entail such low voltages and currents, the probes cannot hurt you (unless you poke yourself with their sharp ends).

You can buy other types of meter leads as accessories. Some are very short, and some terminate in alligator clips or little spring-loaded hooks known as ***mini-grabbers***. I like short leads and mini-grabbers myself, but meters are supplied with probes by default.

Now, where should you plug the leads into your meter? This is not as simple as it sounds.

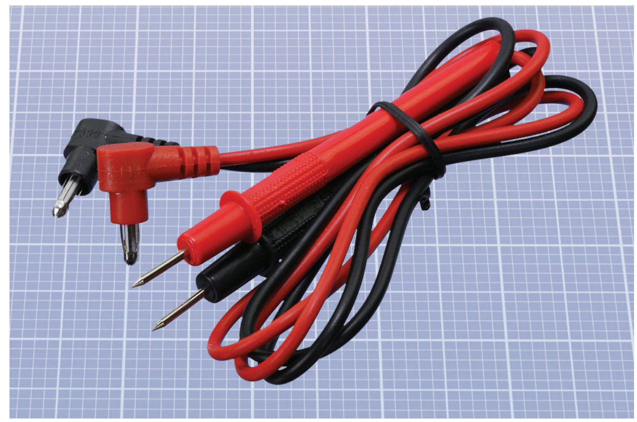


Figure 1-17. Typical meter leads. The large squares in the background are at intervals of 1 inch, divided into tenths.

First I'll deal with the black lead, which is easy. Its plug should go into the socket on your meter labeled **COM**, which is ***common*** to all your measurements. After you plug in the black lead, you will never have to unplug it again.

Another socket should have a letter **V** beside it, and also a Greek symbol named ***omega***, which looks like the sample in Figure 1-18 and represents electrical resistance.

Ω

Figure 1-18. The Greek letter omega represents units of electrical resistance.

The socket may have some other symbols beside it as well, but V and the omega symbol will always be there, indicating that this socket can measure electrical resistance or voltage. Plug the red lead into it. See Figures 1-19 and 1-20.

You may see a third socket labeled **mA**, meaning *milliamps*. I'll get to that later. You should see a fourth socket labeled 10A or 20A, meaning 10 amps or 20 amps. This, too, I will deal with later. Don't use these sockets now.

Figure 1-19. Where to plug in the meter leads.

Figure 1-20. Different meter, same places to plug the leads.

Ohms	Kilohms	Megohms
1 Ω	0.001K	0.000001M
10 Ω	0.01K	0.00001M
100 Ω	0.1K	0.0001M
1,000 Ω	1K	0.001M
10,000 Ω	10K	0.01M
100,000 Ω	100K	0.1M
1,000,000 Ω	1,000K	1M

Figure 1-21. Conversion table for units of resistance.

What Is Resistance?

Electrical resistance reduces the flow of electric current. Almost every substance in the world has at least some resistance, even including your tongue.

We measure distance in miles or kilometers, and temperature in Fahrenheit or Celsius degrees. We measure electrical resistance in *ohms*, which is an international unit named after Georg Ohm, who was an electrical pioneer.

The Greek omega symbol that I showed in Figure 1-18 represents ohms.

For resistances above 999 ohms the uppercase letter K is used, which means *kilohm*, equivalent to 1,000 ohms. Sometimes K has an omega symbol printed after it, just to make things absolutely clear; but more often, it doesn't. For example, a resistance of 1,500 ohms will usually be referred to as 1.5K. Above 999,999 ohms, uppercase letter M is used, meaning *megohm*, which is a million ohms. In everyday speech, a megohm is often referred to as a "meg." If someone is using a "two-point-two meg resistor," its value will be 2.2M.

```
1K = 1,000 ohms
1M = 1,000K = 1,000,000 ohms
```

A table showing intermediate values is in Figure 1-21.

In Europe, you may find that a resistance value has a letter R, K, or M where you would expect a decimal point to be. This reduces the risk of errors, because sometimes a point can disappear when it is badly printed. Thus, 5K6 in a European circuit diagram means 5.6K, 6M8 means 6.8M, and 3R3 means 3.3 ohms. I won't be using the European style here, but you may run into it in some circuit diagrams elsewhere.

A material that has very high resistance to electricity is known as an *insulator*. Most (but not all) plastics, including the colored sheaths around wires, are insulators.

A material with very low resistance is a *conductor*. Metals such as copper, aluminum, silver, and gold are excellent conductors.

Is your tongue an insulator or a conductor?

Let's find out.

The Tongue Assessment

Inspect the dial on the front of your meter, and you'll find one position, or a set of values, identified with the omega symbol. On an auto-ranging meter, simply turn the dial to point to the symbol as shown in Figure 1-22. Then touch the probes to your tongue about an inch apart, and wait for the meter to choose a range automatically. Watch for letter **K** in the numeric display.

On a manual meter, you must choose a range. The way you do this is to select the *maximum* value that you expect. The meter will measure resistances up to that value, but not above it. For a tongue measurement, setting your meter to 200K or 400K (200,000 ohms or 400,000 ohms) should be about right. See the closeups of manual meters in figure 1-23 and figure 1-24.

What if your tongue has a resistance higher than 200K? A manual meter will display an error message, which usually looks like **OL**. It means "open leads," as if the meter leads are not connected to anything. What do you do? Simply turn the meter dial to the next higher value, such as 2M. Your tongue is unlikely to have a resistance higher than that.

- When you see **OL**, select a different range.

Whatever value you find for the resistance of your tongue, please write it in your experimenter notebook. I'm going to refer to it later.

Now put aside the probes, stick out your tongue, and use a tissue to dry it carefully and thoroughly, as you did before. Without allowing your tongue to become moist again, repeat the test, and the reading should be higher.

Here are two conclusions from your tongue tests.

- When you were touching your tongue with a battery, more moisture seemed to allow more electricity to flow, creating a bigger tingle.
- When you were using the meter, more moisture seemed to create a lower resistance.

I use the phrase "seemed to," because we haven't proved anything yet. All we have so far is a theory. Even if a lower resistance does allow more current to flow, I would like to know how much. And what exactly is "current," anyway? During the next few experiments, you'll discover the answers to these questions. By the end of Experiment 4, all the mysteries will be resolved.

What if you can't get any resistance value for your tongue, and you just see the **OL** error message? Try cleaning the probes, first with a detergent such as dish washing liquid, and then with something very slightly abrasive, such as toothpaste. Don't use a highly abrasive cleanser, such as a bath cleanser; it will damage the plating on the probes. Rinse and dry the probes after cleaning them.

If all else fails, add salty water to your tongue, as I suggested when you were using the battery.

Figure 1-22. Selecting resistance in an auto-ranging meter.

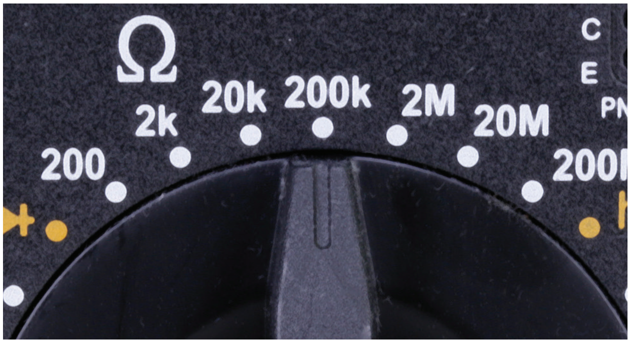


Figure 1-23. A manual meter requires you to select the range

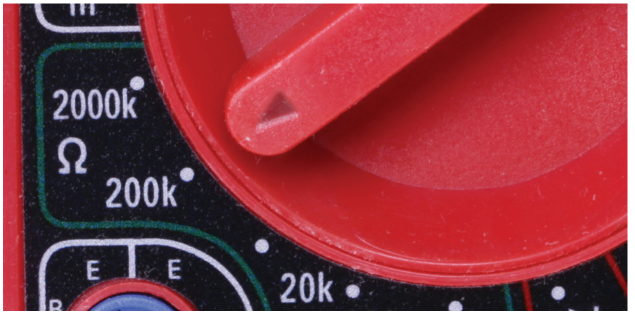


Figure 1-24. Different meter, same feature.

Other Resistances

A successful experiment should give you the same result every time, without any random factors interfering.

The tongue test is full of random factors, which are properly known as *uncontrolled variables*. Moisture on your tongue was one variable. I suspect that another variable is the distance between the probes, and we can investigate that.

Hold the meter probes so that their tips are only 1/4" apart. Touch them to your moist tongue. Now separate the probes by 1" and try again. What readings do you get?

- When electricity travels through a shorter distance, it encounters less resistance, if all other factors remain the same.

Try a similar experiment on your arm, as shown in Figure 1-25. If you get no reading, moisten your skin. You can vary the distance between the probes in fixed steps, such as 1/4", and note the resistance shown by your meter. Do you think that doubling the distance between the probes doubles the resistance?

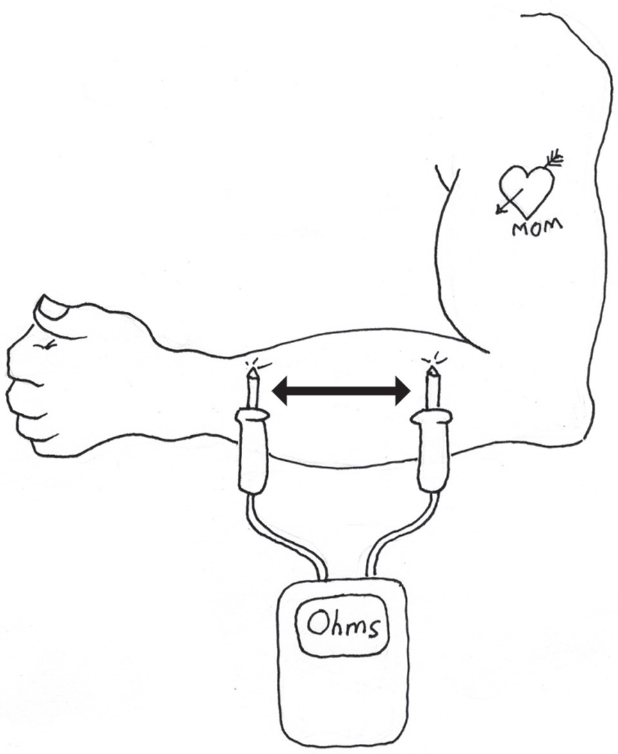


Figure 1-25. Vary the distance between the probes, and check the reading on your meter.

There is one more variable that I haven't discussed, which is the amount of pressure between each probe and the skin. If you press harder, I suspect that the resistance will diminish. Can you prove this? Why do you think it happens? How could you design an experiment to eliminate this variable?

If you get tired of measuring skin resistance, you can try dunking the probes into a glass of water. Then dissolve some salt in the water, and test it again. No doubt you've heard that water conducts electricity, but the full story is not so simple. Pure water has a relatively high resistance, as you can find out for yourself if you obtain some distilled water, which is usually sold cheaply in supermarkets. Salt, or some other impurities, will lower the resistance.

To get a better understanding of what is happening in these experiments, you need to know more about the flow of electricity, which is known as *current*. (Sometimes it is referred to as *amperage*.) I'll show you how to measure that in Experiment 2.

The Man Who Discovered Resistance

Georg Simon Ohm, pictured in Figure 1-26, was born in Bavaria in 1787 and worked in obscurity for much of his life, studying the nature of electricity using metal wire that he had to make for himself (you couldn't truck on down to Home Depot for 100 feet of doorbell wire back in the early 1800s).

Despite his limited resources and inadequate mathematical abilities, Ohm was able to demonstrate in 1827 that the electrical resistance of a conductor such as copper varied in inverse proportion with its area of cross-section, and the current flowing through it is proportional to the voltage applied to it, so long as temperature is held constant. Fourteen years later, the Royal Society in London finally recognized the significance of his contribution and awarded him the Copley Medal. Today, his discovery is known as *Ohm's Law*. When you get to Experiment 4, you will be able to discover Ohm's Law for yourself (although really, you'll be rediscovering it).

Figure 1-26. Georg Simon Ohm, after being honored for his pioneering work, most of which he pursued in relative obscurity.

Cleanup and Recycling

Your battery should be almost as good as new. You can use it again.

Remember to switch off your meter before putting it away. Most meters will switch themselves off after a while, or will beep to remind you, but you can prolong the battery life if you switch off the meter more promptly.

Experiment 2
Go with the Flow

In this experiment you'll build your first circuit, in which you'll learn about electric current by taking an LED to its limits—and beyond.

You Will Need:

- 9V battery (1).
- Resistor, 15 ohms, brown-green-black (1).
- Resistor, 150 ohms, brown-green-brown (1).
- Resistor, 470 ohms, yellow-purple-brown (1).
- Resistor, 1.5K (1,500 ohms), brown-green-red (1).
- Generic red LED (2).
- Test leads (1 red, 1 black, 1 other color).
- Multimeter (1).

Rating a Resistor

Often we need to add electrical resistance to a circuit, for reasons that you will soon see. This can be done very easily by using components known as (guess what) *resistors*, which have their values measured in ohms.

Resistors that you acquire for the projects in this book may or may not be labeled—but that's okay, as you can find out what their values are. First I'll show you how to measure the values, and then I'll explain how to decode them.

Some resistors have a number clearly stated on them in tiny print that you can read with a magnifying glass, as shown in Figure 2-1, on the next page. Unfortunately, most manufacturers don't print numbers on resistors. They use a color code.

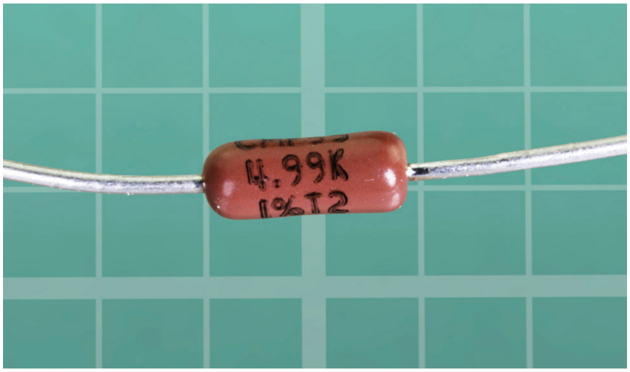


Figure 2-1. An unusual resistor that has its value printed on it.

The parts list that I provided for this experiment mentions the colors of bands printed on each resistor. You can see them in Figure 2-2. The wires coming out of them are *leads*, even though they look unlike the leads for your meter or the test leads with alligator clips on each end. The wires coming out of your LED are leads, too.

Your resistors may have silver bands instead of the gold bands that I have shown, but that's okay. I'll explain the difference in a moment.

First, I want you to check the values of the resistors. Make sure the red meter lead is still in the volts-ohms socket of your meter, as in Experiment 1. The 15-ohm resistor

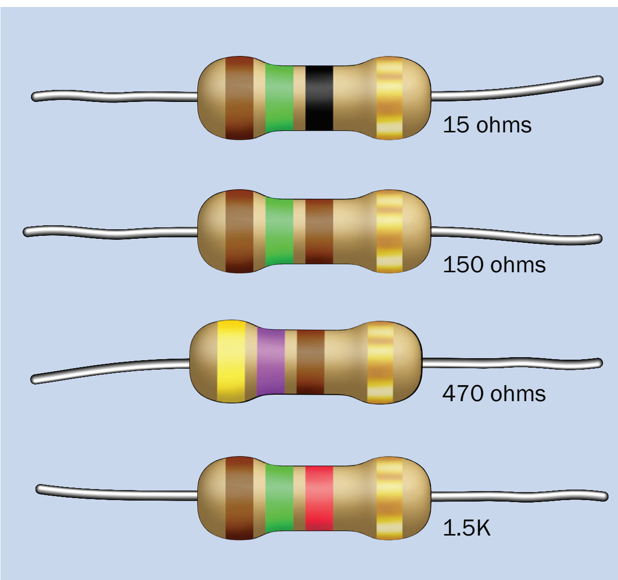


Figure 2-2. The resistors that you need in this experiment.

has a value that is much lower than that of your tongue, so you need to select a different resistance range. A value of 200 ohms is often the lowest available, so try that.

The resistor can be either way around; it makes no difference. Place it on a surface such as wood or plastic, which doesn't conduct electricity, and hold the probes by their plastic handles. If you touch the metal ends of the probes while trying to measure the resistor, you'll also be measuring the resistance of yourself, which is not what you want. See Figure 2-3.

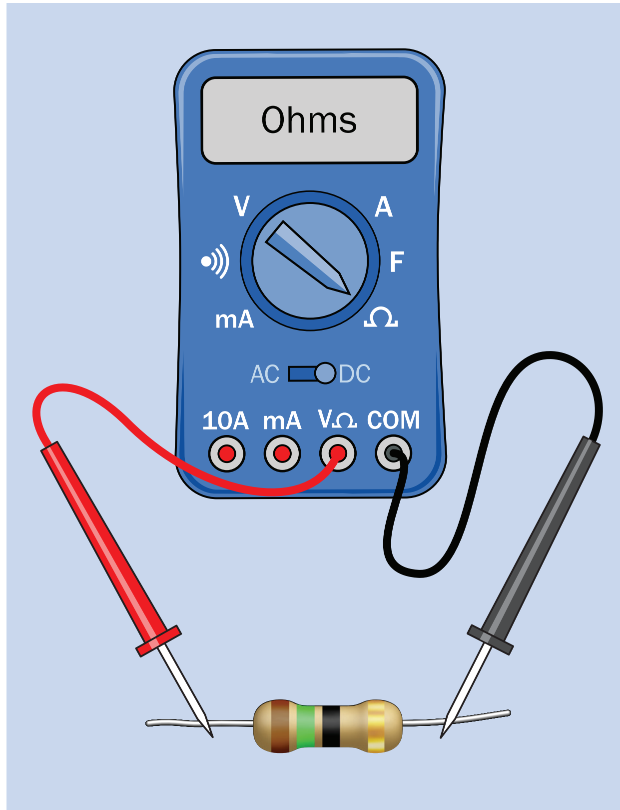


Figure 2-3. Checking the value of a 15-ohm resistor.

Press hard to make a good connection. If this seems awkward, you can try adding a couple of test leads, as in Figure 2-4. Now you can do hands-free resistor testing, and the results should be very nearly the same.

Write the value of your measured resistance in your notebook. Remove the 15-ohm resistor, and substitute the 150-ohm resistor. After you measure that, try the 470-ohm resistor—although, you'll get an OL error mes-

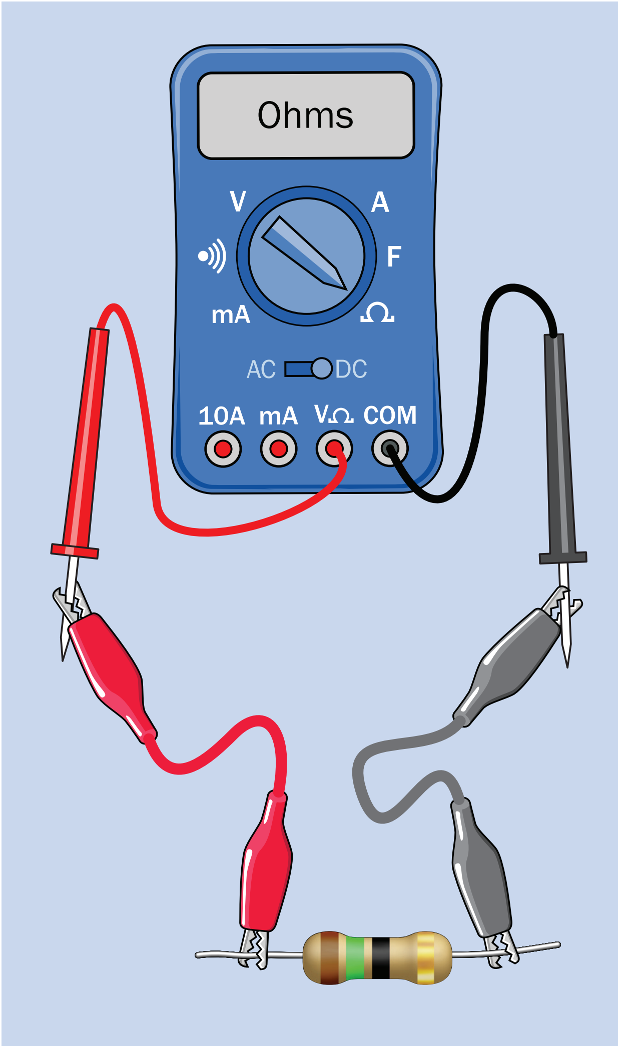


Figure 2-4. Using test leads to grab the resistor.

sage until you change the range on your meter dial from 200 ohms to 2K. Finish by checking the 1.5K resistor.

I'm betting that the values which you measure will not be exactly what you expect. The values I found when I tried this were 15.1 ohms, 148 ohms, 467 ohms, and 1,520 ohms.

You have just encountered a basic truth in electronics:

- Measurements are never precise.

Your meter isn't absolutely precise, and nor are the values of the resistors. Other factors can interfere, such as room temperature, which affects electrical resistance. There is also a tiny amount of resistance between the probes of the meter and the leads of the resistor. Your goal is to get as close as possible to accurate measurements, but total precision is impossible. That's just the way things are when you deal with electronic hardware.

Decoding a Resistor

Now that you've verified your resistor values—more or less—I'll explain their coding. The system is shown in Figure 2-5, and the procedure goes like this.

- Ignore the color of the body of the resistor.

- If there is a silver or gold stripe, turn the resistor so the stripe is on the right-hand side. Silver means that the value of the resistor is accurate within plus-or-minus 10%, while gold means that the value is accurate within plus-or-minus 5%. These percentages are known as the ***tolerance*** of the resistor. Some resistors have a 1% tolerance, or even better—in which case,

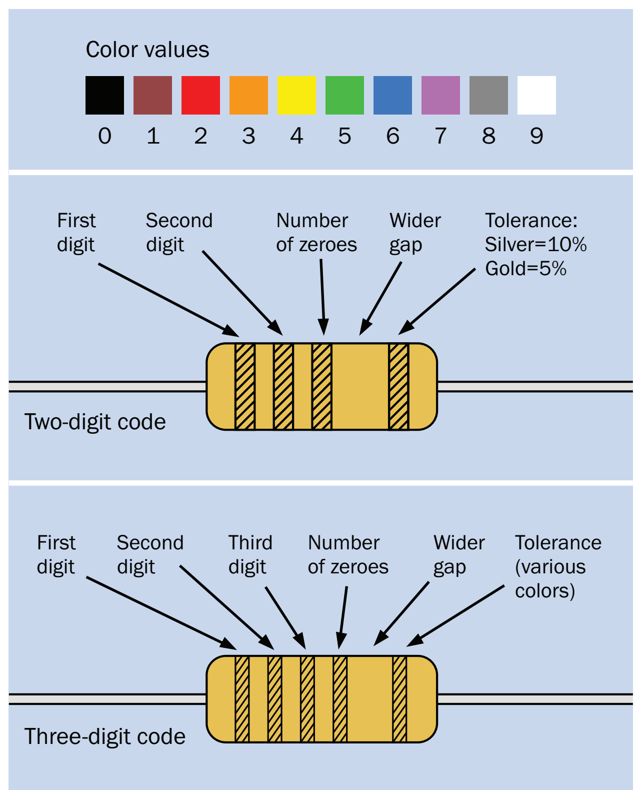


Figure 2-5. The color-coding system for resistors.

they won't have a silver or gold stripe, and some other color will be used. Whatever it is, you will find a gap between it and the other stripes, this gap being wider than the gaps among the other stripes.

- At the left end of the resistor will be three or four colored stripes. If the group consists of three stripes, the colors of the first two tell you the first two digits in the value of the resistor. If the group consists of four stripes, the colors of the first three tell you the first three digits in the value of the resistor.
- The last stripe in the group tells you how many zeroes follow the digits in the value of the resistor.

For example, look at your 1.5K resistor. Its colors are brown (1), green (5), and red (two zeroes). In other words, 1500, which is 1.5K.

A resistor with a group of four colored stripes is likely to have a better tolerance than 5%. The exact tolerance is not important for the projects in this book.

Mysterious Numbers

If you check a few resistors (or shop for them online) you'll notice that the same pairs of digits keep turning up. In hundreds of ohms, we often find 100, 150, 220, 330, 470, and 680. In thousands of ohms, the typical sequence is 1.0K, 1.5K, 2.2K, 3.3K, 4.7K, and 6.8K. In tens of ohms, we find 10, 15, 22, 33, 47, and 68. Why is this?

Long ago, manufacturing resistors accurately was a challenge, so the tolerance was 20%. In other words, the actual value could be a full 20% higher or lower than it was supposed to be. In the case of a 15K resistor, its actual resistance could be as low as 12K, because:

```
15K - 20%   = 12K
```

On the other hand, a 10K resistor could have a value 20% higher, like this:

```
10K + 20%  = 12K
```

So, a 15K resistor and a 10K resistor might both have the same value, and there was no point in manufacturing resistors with intermediate values.

Figure 2-6 illustrates this. The white numerals are the ***nominal values*** to create resistors, meaning the values that manufacturers are trying to achieve. You can see how cleverly they were chosen so that the range of possible values, 20% above and below, allowed hardly any overlap.

Resistors are manufactured much more accurately today, but everyone got into the habit of using the old range of values, and these are still the ones that you are most likely to find. Bearing this in mind, I have used them throughout this book.

20% less than nominal	0.8	1.2	1.76	2.64	3.76	5.44
Nominal Value	**1.0**	**1.5**	**2.2**	**3.3**	**4.7**	**6.8**
20% more than nominal	1.2	1.8	2.64	3.96	5.64	8.16

Figure 2-6. The original range of multipliers for resistor values, and the range of values plus-or-minus 20%.

The First Circuit

Take a look at one of your red LEDs. Old-fashioned light bulbs wasted a lot of power by converting electricity into heat, but LEDs are smarter: They convert almost all their power into light, and they last almost indefinitely—if you don't mistreat them.

What if you do mistreat them? This may be the time to find out.

Start with your 1.5K resistor. That's the one with brown-green-red colored bands. Grip it in a test lead connected with the negative side of your 9V battery. The resistor can be either way around; it doesn't care.

Now use another test lead to grip your LED, as in Figure 2-7. You ***do*** need to be careful to connect the LED the right way around, as shown.

- When you pass electric current through an LED, its longer lead must always be more positive than its shorter lead, as if the long lead had some extra length added (+) to it.

Touch the loose leads of the LED and the resistor together. Your LED lights up! Amazing. You have your first circuit.

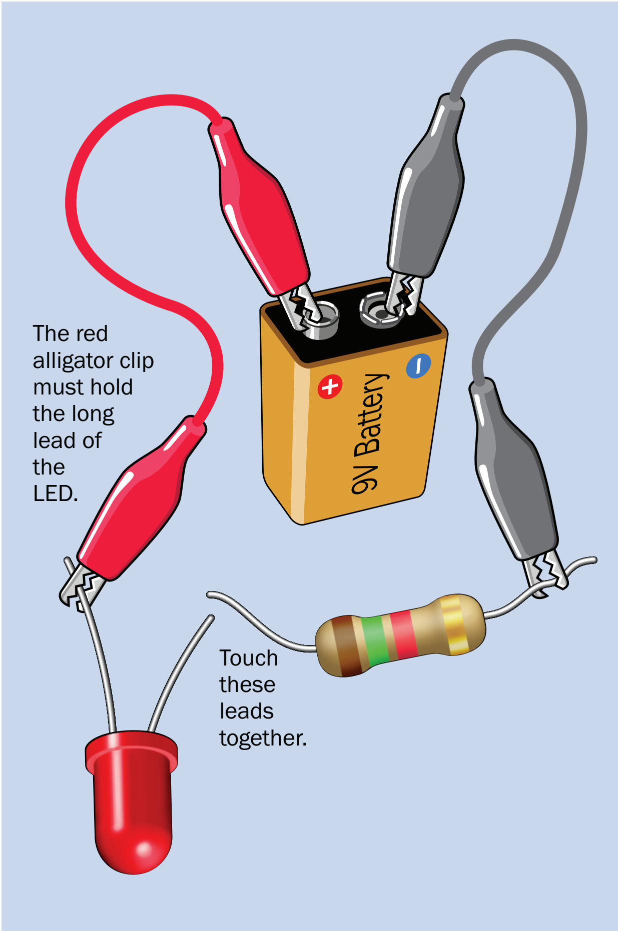


Figure 2-7. Testing an LED.

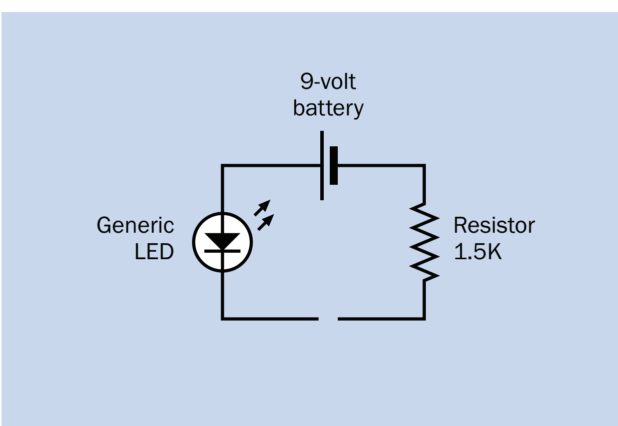


Figure 2-8. Schematic version of the LED circuit.

Polarity

When dealing with a power source such as a battery, remember:

- The "plus" symbol always means "positive."
- The "minus" symbol always means "negative."
- The distinction between positive and negative is called ***polarity***.

When information from a manufacturer or a book tells you that a component such as an LED has polarity, make sure to connect it the right way around.

If you connect positive voltage to the short lead of the LED instead of to the long lead, you are applying ***reverse polarity***, and the LED won't work. You may also shorten its life.

What if you trim the leads on an LED to make it fit into a circuit that you're building, and you can't remember which lead is the long one anymore? Not a problem! An LED of the type you are using has a flat spot in its round base, marking the ***shorter*** lead (the more-negative one).

Schematics

Take a look at Figure 2-8 and notice the similarity with Figure 2-7. The circuit is the same, but I redrew it using symbols. This is known as a ***schematic***. I will be using schematics throughout the book, because they're easy to understand and they don't take too much space. You'll find more examples of schematic symbols when you get to Experiment 6.

Notice the two instances of polarity in Figure 2-8:

- The long line in the battery symbol indicates the positive side of the battery.
- The big triangle inside the LED symbol always points from positive to negative.
- The small arrows are just there to remind you that this is a light-emitting diode. (Other types of diodes exist, which don't emit light.)

In Europe, the symbol for a resistor is just a rectangle with its value printed in it. I'm using the zigzag American symbol here, and some people in Europe still use it, too.

Overloading an LED

Now disconnect the 1.5K resistor and substitute the one rated for 150 ohms: brown-green-brown. What happens? The LED is much brighter.

Moistening your tongue reduced the resistance between the probes of your meter, so that when you applied the 9V battery, you got more of a tingle. Reducing the resistance in your circuit gives your LED more of a tingle.

Do you think you can make it even brighter? Maybe you can guess what will happen—but guessing isn't good enough. Remove the 150-ohm resistor, substitute the 15-ohm resistor, but don't touch the wires together yet.

Anyone who knows anything about electronics will be saying, at this point, "No, no, don't do that!" But when someone tells me not to do something, I always want to see what will happen when I do it.

One thing, though. The wires may get a bit hot if you hold them together for long. If you want to be super-cautious, use a glove.

When the lead from the LED touches the lead of the 15-ohm resistor, for a short moment the LED gets really bright. But—no, that was too good to last. The LED fades away until there's just a dim glow.

Some LEDs fail more dramatically than others. I'm not sure why. A couple of readers have told me that their LEDs split in half when they were overloaded. I haven't been able to make this happen myself, although I've tried. I do know that 3mm LEDs self-destruct more quickly than 5mm LEDs.

Anyway—now you see how easy it is to destroy electronic components. Throw away the LED that you overloaded, because it will never work properly again. Its life has been sacrificed in the interests of providing you with a learning experience. Fortunately, although LEDs are an amazing achievement in electronics, they don't cost much.

The question is, why exactly did the LED burn out? You might guess that too much electric current flowed through it—but again, I don't like guessing. I think you need a way to measure current, and your meter will do it. First, though, I will give you a definition.

Defining Current

Electric current is the flow of electricity, per second, usually measured in ***amperes***, which are abbreviated as ***amps***. The flow consists of ***electrons***, each of which is a tiny particle carrying an electrical charge. If your human senses were fast enough to count electrons as they zip through a wire, you could measure 1 amp as 6.25 quintillion electrons per second, in American quintillions.

Ampere Basics

The ampere is an international unit, abbreviated with letter A. One milliamp (usually written mA) is 1/1,000 of an ampere. A microamp (usually written μA) is 1/1,000 of a milliamp.

```
1mA = 1,000μA
1A = 1,000 mA = 1,000,000μA
```

In Figure 2-9, you'll see a table showing intermediate values.

Microamps	Milliamps	Amps
1μA	0.001mA	0.000001A
10μA	0.01mA	0.00001A
100μA	0.1mA	0.0001A
1,000μA	1mA	0.001A
10,000μA	10mA	0.01A
100,000μA	100mA	0.1A
1,000,000μA	1,000mA	1A

Figure 2-9. Conversion table for units of current.

Measuring Current

Every meter I have ever seen has a socket for measuring current labeled either **10A** or **20A**, telling you the maximum number of amps that it can deal with. You won't be measuring a high current like that right now.

Your meter will also have a socket labeled **mA**, meaning milliamps, for measuring smaller currents. The maximum for this socket is usually printed beside it. Often it's 200mA, but sometimes 400mA. Look carefully at your meter and make a mental note of how many mA it will stand.

The **mA** socket is often separate from all the others—but not always! Some meters combine the measurement of volts, ohms, and mA all in one socket. An example is shown in Figure 2-10. This meter is unusual in that it locates the **COM** socket in the middle, but it still functions the same way, and the right-hand socket is clearly labeled for volts, ohms, microamps, and milliamps.

Figure 2-10. This meter allows you to measure volts, ohms, µA, and mA all from one socket.

This is a nice feature, because you can leave the red lead plugged into the same socket almost all the time. But your meter may have a separate socket for mA, as many do. The meters that I showed in Figures 1-19 and 1-20, in the previous experiment, have this arrangement.

To measure current, it has to flow through your meter. This requires some caution, because too much current can blow a fuse inside the meter, in which case you have to open the meter to remove the fuse, and then you must find a replacement fuse online that is exactly the right type, and while you're waiting for it to come in the mail, you can't measure current with your meter. In a cheap meter, the fuse may not be socketed, which makes it difficult to remove, and—well, you get the idea, it's a hassle! So, don't push too much current through your meter. If you take just a couple of simple precautions, it shouldn't happen. Personally it's been about five years since I last blew a fuse in my meter, and at that time I bought a couple of spare fuses, just in case I got careless again.

For the circuit that you're using in this experiment, you can safely use the **mA** socket on any meter, because I happen to know that you won't be measuring more than 25mA. Turn the dial to the next-highest value in the range, which is usually 200mA.

What if you didn't have me telling you how much current you would be measuring? You could do a calculation,

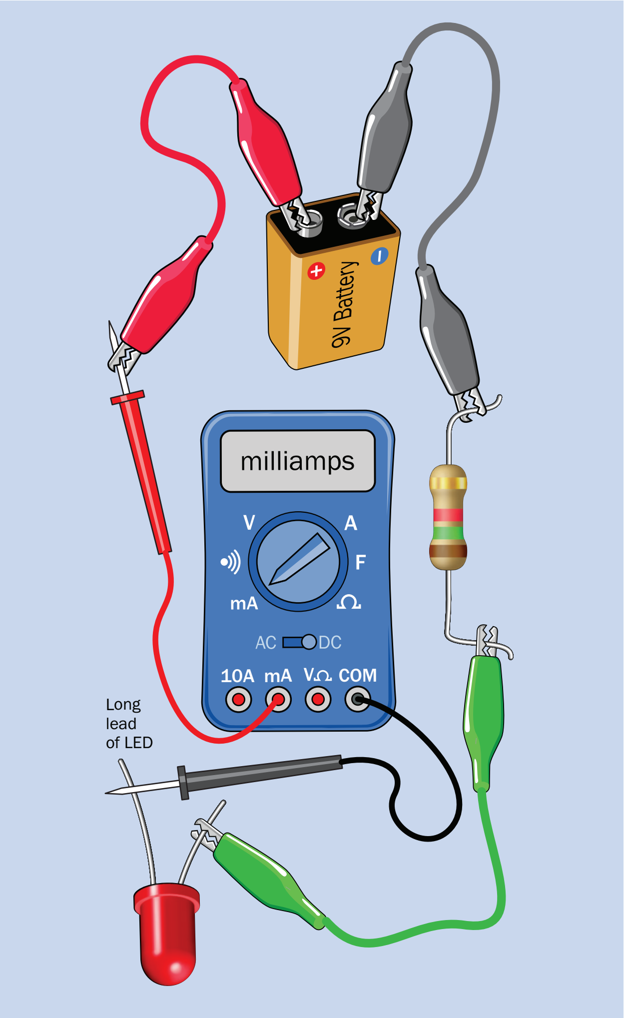


Figure 2-11. Measuring current with your meter.

which I'll explain in Chapter 4, or you could start by using the socket labeled **10A** (or **20A**) and work downward.

Current in the LED Circuit

I want you to rebuild your circuit with a new LED. Make sure you use the 1.5K resistor, this time—not the 150-ohm resistor, and definitely not the 15-ohm resistor. You don't want to mistreat more LEDs than necessary.

Insert the meter into the circuit, as shown in Figure 2-11, and make sure the red lead is in the **mA** socket, not the volts-ohms socket (assuming your meter has a separate socket for mA).

- Remember, when you are measuring current, the meter is inserted into the circuit, with current passing through it.

From the positive battery terminal, current goes through the meter, through the LED, through the resistor, and back into the negative side of the battery.

You might be concerned that the black lead of the meter connects with the long lead of the LED. But this is quite okay. The logic goes like this:

- The red lead of the meter must be more positive than the black lead, so it connects with the positive side of the battery.

- The black lead of the meter connects via the LED and the resistor to the negative side of the battery.

What value do you see on your meter? Whatever it is, make a note of it. I obtained 5.1mA in my circuit. Actually it was 5.08, but I rounded it to 5.1, because this kind of measurement isn't very accurate, for reasons that I will explain shortly. It's not good practice to include extra decimal places when the measurement does not support them.

- When you remove a decimal place, if the digit that you omit was 5, 6, 7, 8, or 9, add 1 to the preceding value. So, 5.08 becomes 5.1. This is known as ***rounding up***.

- When you remove 1, 2, 3, or 4, you don't have to change the value of the preceding digit. This is known as ***rounding down***.

Now remove your meter from the left side of your circuit, and insert it into the right side. The two meter positions are suggested in Figure 2-12. Make sure the black lead of the meter connects with the negative side of the battery. When I did this, my meter reading was exactly the same as before. Is that what you find, in your circuit? I hope so, because in a simple circuit like this, the current has nowhere else to flow. Therefore, it has to be the same all the way through.

- The current is the same at all points throughout a simple circuit.

Is 5.1mA an acceptable current for the LED? Well—it seems okay. The LED doesn't burn out. But would it still be okay if you ran 5.1mA through it for a day or two? Um—I guess so.

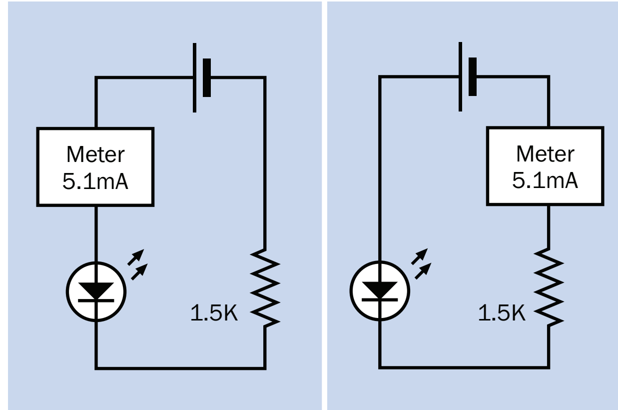


Figure 2-12. Two ways to measure current through a circuit.

Alternatively, do you think maybe you could get just a little more light out of the LED, without damaging it, if you reduced the resistance a teeny bit? There's an easy way to eliminate the guesswork. You ask the manufacturer.

Getting the Data

Almost any component has a datasheet published online by the manufacturer. Finding it is really easy, so long as you know the part number of the component. When you're buying components yourself, you will see the part number. When you are using a component from a kit, the supplier will usually show the part number.

Suppose my red LED is a Cree C503B-RAN. I simply go to my usual search engine and type in:

```
cree C503B-RAN datasheet
```

A moment later, all the information I could possibly want is on the screen. In fact there's more than I need, and some of it is quite technical, so I've snipped just the relevant parts from the datasheet in Figure 2-13.

When you look at a datasheet and see the phrase "Absolute Maximum Ratings," think of it as being like a sign telling you the headroom on a low bridge. The numbers are serious! They really mean it! If you exceed these limits, the component will be damaged. Always stay ***well below*** the maximum.

For this LED, you can see 50 mA is the absolute maximum, so my value of 5.1mA is just a fraction of that. Incidentally, ***forward current*** means that it is flowing in

ABSOLUTE MAXIMUM RATINGS (T_A = 25°C)

Items	Symbol	Absolute Maximum Rating	Unit
Forward Current	I_F	50 Note1	mA
Peak Forward Current Note2	I_{FP}	200	mA
Reverse Voltage	V_R	5	V
Power Dissipation	P_D	130	mW
Operation Temperature	T		

TYPICAL CHARACTERISTICS (T_A = 25°C)

Characteristics	Condition	Minimum	Typical	Maximum
Forward Voltage	I_F = 20 mA		2.1	2.6
Reverse Current	V_R = 5 V			100
Dominant Wavelength	I_F = 20 mA	[illegible]		

Figure 2-13. Sections clipped from an LED datasheet.

the correct direction through the component. As for ***peak*** forward current, that would be just for a fraction of a second, if your power supply is fluctuating for some reason. It shouldn't normally happen.

Now check the "Typical Characteristics" in the datasheet. The word "Typical" is sometimes abbreviated as "Typ." These are the realistic values that you should use on a routine basis. Notice in the "Condition" column, it says "IF = 20mA." I'll explain that weird term "IF" a bit later, when I deal more with datasheets, but right now, 20mA is what you need to know. That is what you should really be using on a "typical" basis.

So, putting the 1.5K resistor in the circuit was a very safe bet. You could try substituting lower-value resistors and measuring the current till it was closer to 20mA, but I'll just give you the quick answer: 470 ohms will be a conservative option with a 9V battery, for a generic red LED. You can check this yourself.

Of course, if you wanted your LED to be dimmer for some reason—perhaps to extend the life of your battery—there's nothing stopping you from using a higher-value resistor.

Now you have a circuit which should keep the LED happy on an indefinite basis. So the lesson is: Always read a manufacturer's datasheet!

Keep the 470-ohm resistor in your circuit for now, but remove the alligator clip from the battery so that you don't run it down.

Father of Electromagnetism

Born in 1775 in France, André-Marie Ampère (shown in Figure 2-14) was a mathematical prodigy who became a science teacher, despite being largely self-educated in his father's library. His best-known work was to derive a theory of electromagnetism in 1820, describing the way that an electric current generates a magnetic field. He used this principle to make the first reliable measurements of what came to be known as ***amperage***.

Figure 2-14. André-Marie Ampère.

He also built the first instrument to measure the flow of electricity (now known as a ***galvanometer***). And, since he seems to have had some spare time, he discovered the element fluorine. People weren't distracted by text messages and cat videos in those days.

Direct and Alternating Current

The flow of current that you get from a battery is known as ***direct current***, or ***DC***. You can think of it as a steady stream of electrons in one direction, like the flow of water from a faucet.

The flow that you get from power outlets in your home is ***alternating current***, abbreviated as ***AC***. The ***live*** side of a wall outlet changes from positive to negative, relative to the ***neutral*** side, at a rate of 60 times each second (in many foreign countries, including Europe, 50 times per second). The word "live" is pronounced as in "live music," not as in "the place where I live." The live side of an outlet is sometimes called the "hot" side.

AC is useful because your power utility company uses transformers to reduce the high voltage in power lines to a safe level for your home, and for reasons that I'll get to later, a transformer only works with alternating current. AC is also useful in motors and domestic appliances.

The parts of a power outlet are shown in Figure 2-15. This style of outlet is found in North America, South America, Japan, and some other nations. European outlets look different, but the principle remains the same.

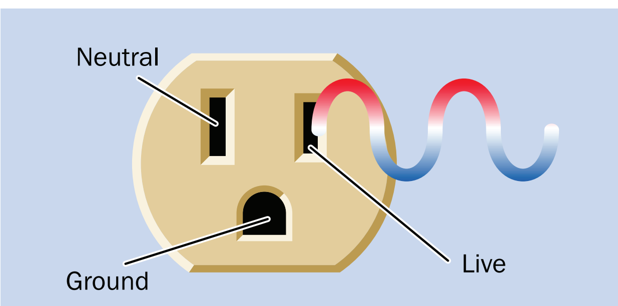


Figure 2-15. The sockets in a power outlet.

If an appliance develops a fault, such as an internal loose wire, it should protect you by sinking the voltage through the ***ground socket***. In England and some other countries, this may be called the ***earth socket***.

For most of this book I'm not going to be dealing with AC, for two reasons: Most simple electronic circuits are powered with DC, and the way that DC behaves is much easier to understand.

Cleanup and Recycling

You'll be continuing to use the same circuit in the next experiment, so there's nothing you need to put away, and the battery should still be good.

Experiment 3
Applying Pressure

In this experiment, you'll be learning all about voltage.

You Will Need:

- 9V battery (1).
- Resistor, 470 ohms, yellow-purple-brown (1).
- Resistor, 1K, brown-black-red (2).
- Resistor, 1.5K, brown-green-red (1).
- Resistor, 2.2K, red-red-red (1).
- Resistor, 3.3K, orange-orange-red (1).
- Generic red LED (1).
- Test leads (1 red, 1 black, 1 other color).
- Multimeter (1).

Potential Differences

Because your meter will be measuring volts, now, it won't be inserted in the circuit anymore. Your circuit should look like Figure 3-1, and the schematic is shown in Figure 3-2.

You're going to measure voltage, so remove the red meter lead from the **mA** socket and put it back in the **volts-ohms** socket.

- If your meter has a separate socket for mA, always check that the red lead is in the volts-ohms socket before measuring volts. Always!

Set the dial on your meter to measure up to 20VDC, or just volts DC if your meter does auto-ranging. See figures 3-3 and 3-4.

Now you can use the meter to check the voltage between any two points in the circuit. The meter has very high resistance when it's set to measure volts, so hardly any current passes through it. You can even check the voltage between the terminals of the battery, as in Figure 3-5, on the next page, and the meter won't be damaged. Make a note of the voltage that you find.

- When you measure voltage, the meter must ***not*** be inserted into the circuit. It sits outside the circuit, like an observer at a sporting event, keeping score.

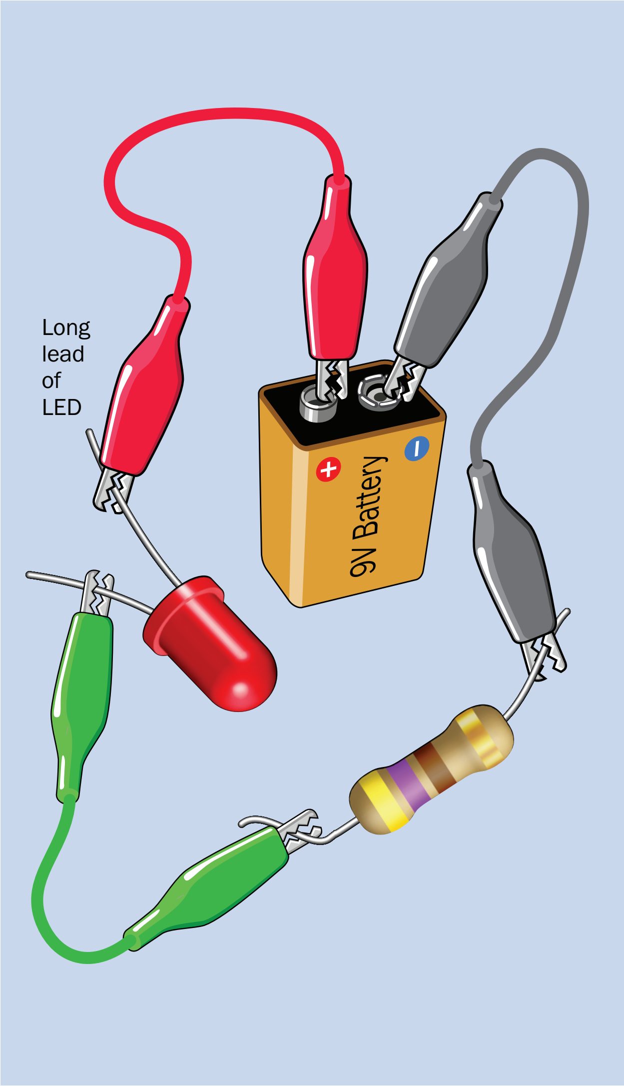


Figure 3-1. Begin with the circuit connected like this.

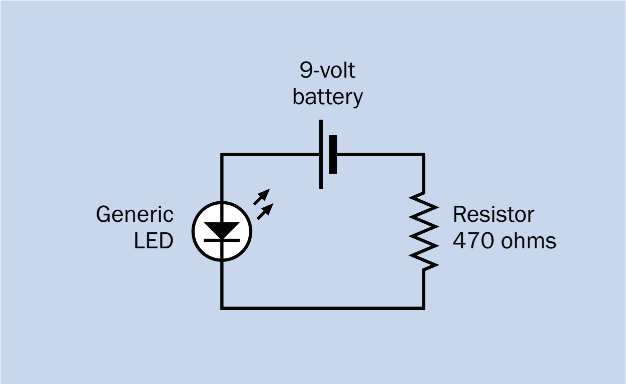


Figure 3-2. The circuit from Figure 3-1, as a schematic.

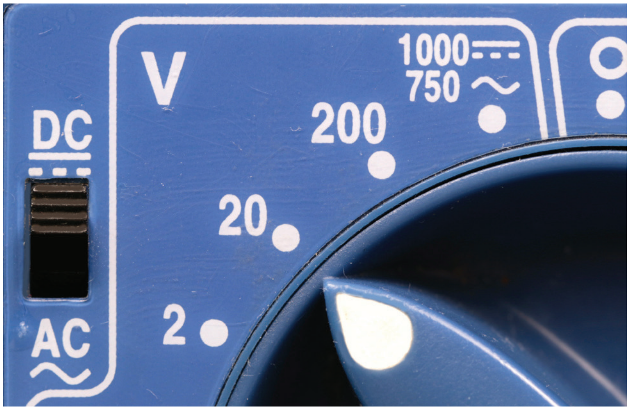


Figure 3-3. Setting an appropriate voltage range on a manual-ranging meter.

Figure 3-4. Setting an auto-ranging meter to measure volts.

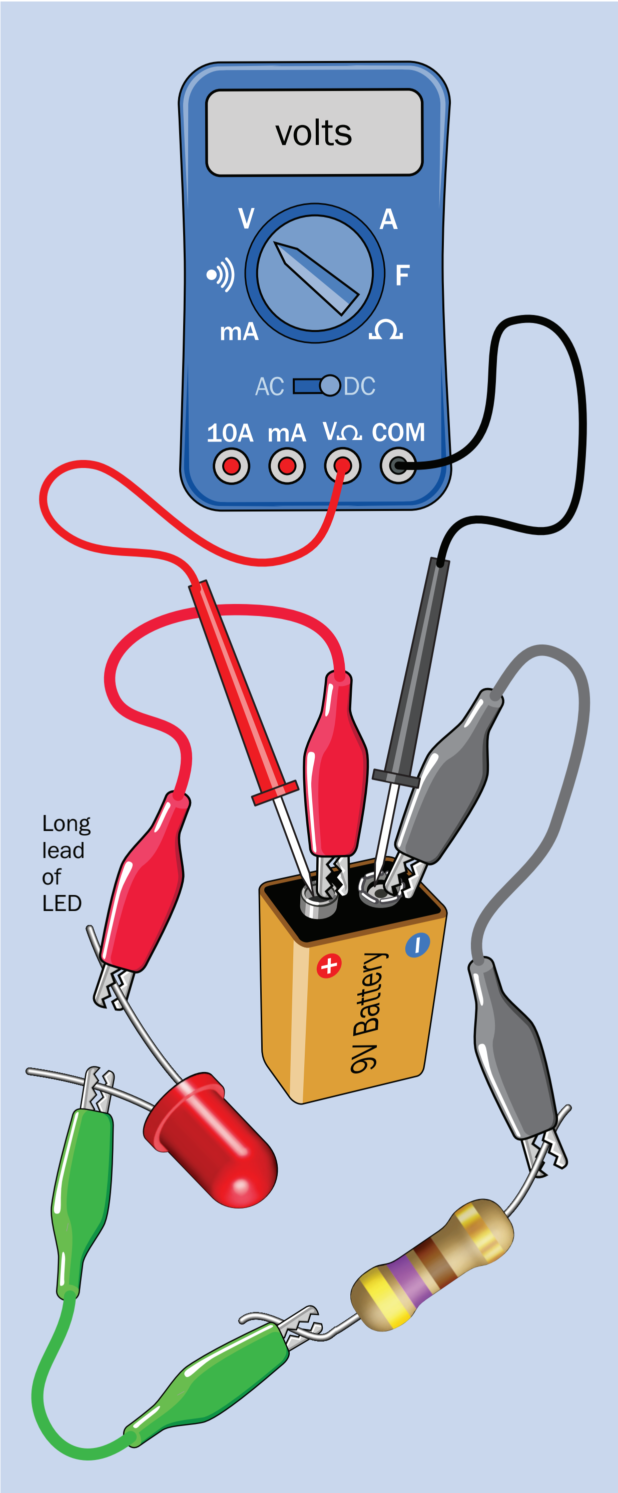


Figure 3-5. Checking your battery voltage in the circuit.

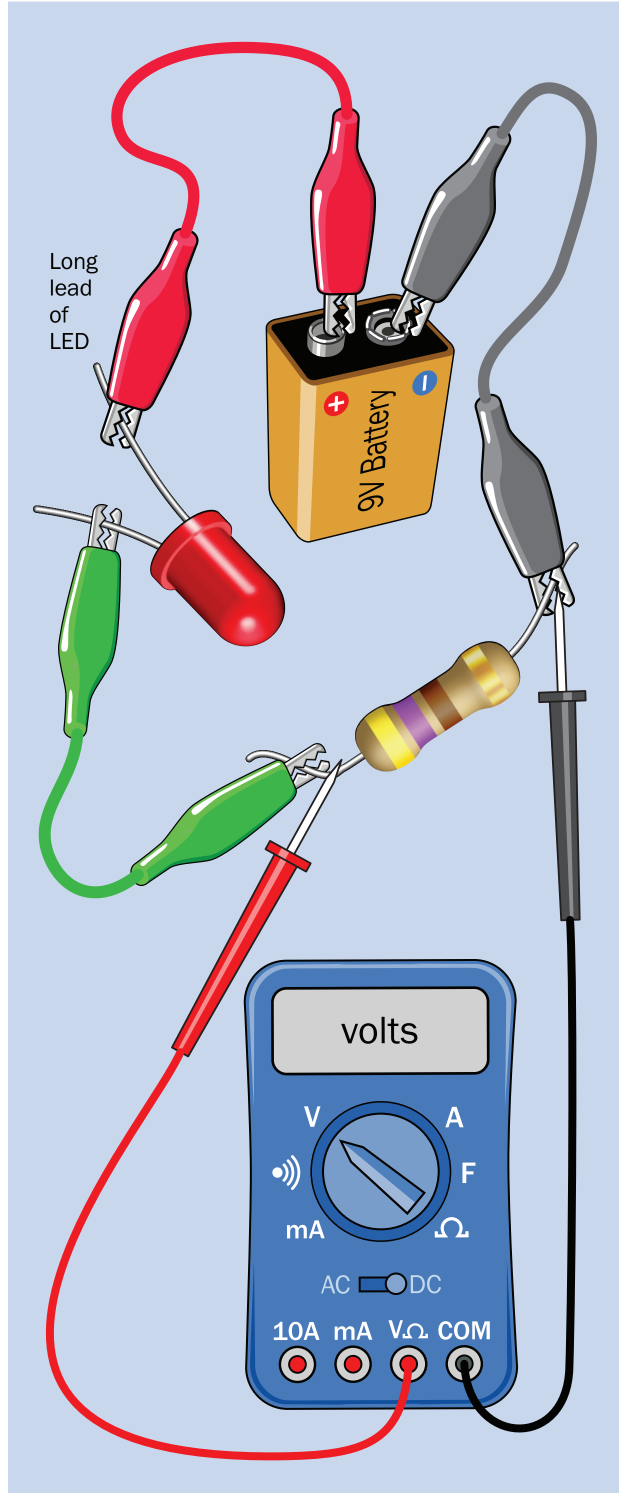


Figure 3-6. Finding the voltage drop across the resistor.

If you put the meter probes across the resistor, as in Figure 3-6, now you are finding out how much voltage drop the resistor causes in the circuit. Make a note of it.

You can do the same thing with the LED.

The schematic in Figure 3-7 shows the numbers I obtained when I tried this, rounded to one decimal place. Your numbers won't be exactly the same as mine, because my meter, my resistor, and my battery won't be exactly the same as yours. They should be similar, though.

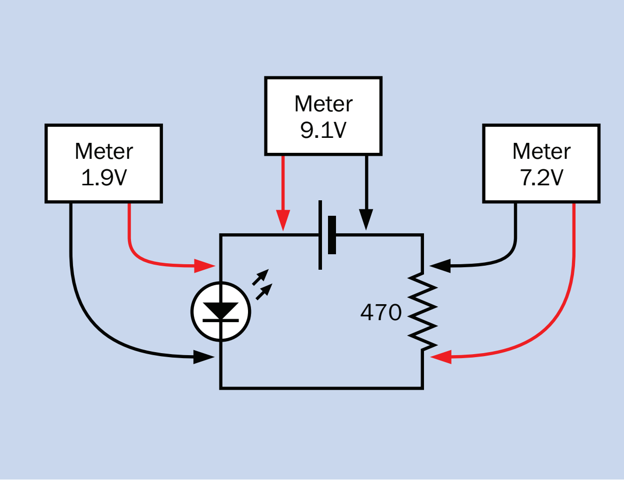


Figure 3-7. Checking voltages around your circuit.

Notice that when you add up the voltage drops across the components, the total is about the same as the voltage supplied by the battery. The components take as much as the battery provides, because there is nowhere else for the voltage to be lost—except through the meter, which only takes a tiny amount when it is measuring voltage.

Here are some things to remember when measuring voltage:

- The circuit must be powered up.
- The meter is set to measure volts.
- The meter is outside the circuit.
- Because voltage is like pressure, you always measure it between two points, one of which has more voltage relative to the other. This is often called the ***potential difference*** between them.
- If you connect the meter the wrong way around, it still makes a measurement, but you see a minus sign on the display.

The voltage across the LED is known as its ***forward voltage,*** and the datasheet in Figure 2-13 suggested a typical value between 2.1 volts and 2.6 volts. So, the circuit is just below the low end of the range.

For an LED, what the datasheet really means is that ***if*** you apply pressure between 2.1 and 2.6 volts, ***then*** the LED will satisfy the manufacturer's specification, especially regarding the amount of light it produces.

Volt Basics

The volt is an international unit, abbreviated with a capital letter **V**. A millivolt (usually written **mV**) is 1/1,000 of a volt. (You won't be measuring microvolts in any experiment in ***Make: Electronics***.)

```
1V = 1,000mV
```

Intermediate values are shown in Figure 3-8. Kilovolts are usually found only in places such as high-voltage power lines.

Millivolts	Volts	Kilovolts
1mV	0.001V	0.000001kV
10mV	0.01V	0.00001kV
100mV	0.1V	0.0001kV
1,000mV	1V	0.001kV
10,000mV	10V	0.01kV
100,000mV	100V	0.1kV
1,000,000mV	1,000V	1kV

Figure 3.8. Conversion table for voltages.

Inventor of the Battery

Alessandro Volta, shown Figure 3-9 on the next page, was born in Italy in 1745, long before science was broken up into specialties. After studying chemistry (he discovered methane in 1776) he became a professor of physics and developed an interest in the so-called galvanic response, whereby a frog's leg will twitch in response to a jolt of electricity.

Figure 3-9. Alessandro Volta discovered that chemical reactions can create electricity.

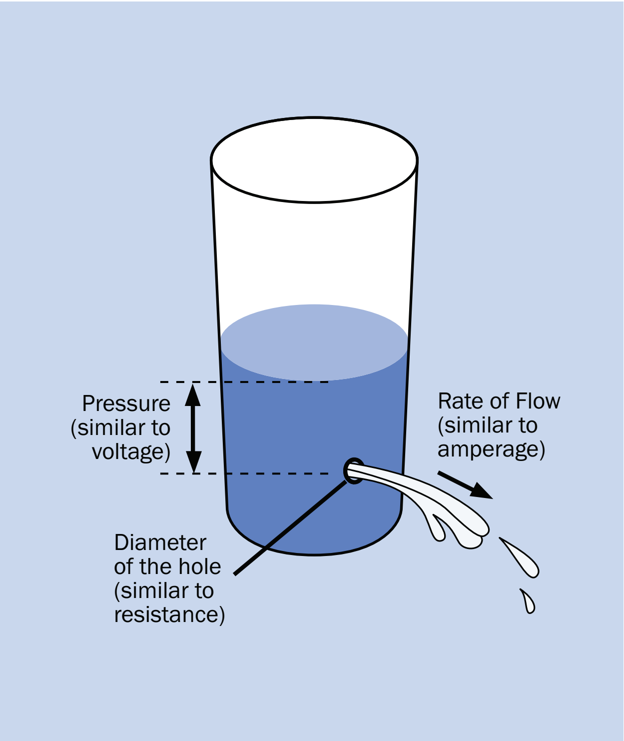


Figure 3-10. The water analogy.

Using a wine glass full of salt water, Volta demonstrated that the chemical reaction between two electrodes (one made of copper, the other of zinc) will generate a steady electric current. In 1800, he refined his apparatus by stacking more plates of copper and zinc, separated by cardboard soaked in salt and water. This "voltaic pile" was the first electric battery.

Imagine living in the days when you could make a major discovery about electricity using just a couple of metal plates, salt water, and cardboard.

Voltage vs. Amperage

You saw that if you decrease a resistance, more current flows. What if you increase the voltage while keeping the resistance the same? Do you think this is another way to push more current through?

Yes, that's what happens, and the best way to visualize it is by thinking of electricity as behaving like water. In fact, in the early history of science, some researchers did actually think that electricity was a kind of fluid, because the flow of electrons behaved that way.

In Figure 3-10, the height of the water in the cylinder, relative to the hole, will create water pressure. The rate of flow will be similar to amperage, while the diameter of the hole is similar to resistance.

When you add more water to the tank, you create more pressure, as in Figure 3-11. If the hole in the cylinder is still the same size, the increase in pressure will create an increase in flow.

- Think of voltage as pressure. It is measured between two points.
- Think of amperes, or amps, as the rate of flow, properly known as current.
- Think of resistance as limiting the flow.

In Figure 3-12, you can see another way of looking at the situation.

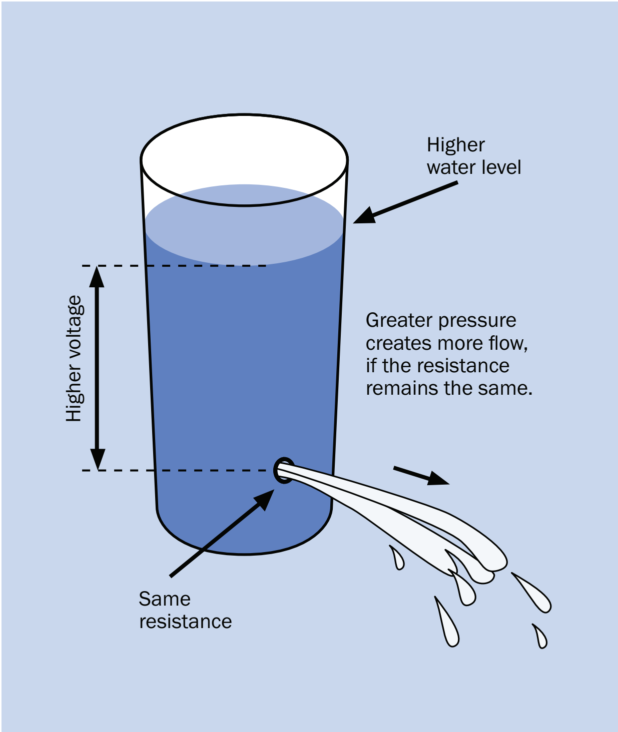


Figure 3-11. When the pressure increases, the flow increases, if the resistance remains the same.

Now I want to find out *exactly* how voltage, current, and resistance are related to each other—and fortunately, this can be discovered quite easily.

Asterisks and Parentheses

This experiment will entail some simple arithmetic, in which I will be using the following symbols:

```
- minus
+ plus
/ divided-by
* multiplied-by
```

The last one in the list may be the only one that seems surprising. The reason I am using an asterisk for multiplication, and not a **x** symbol, is that most computer languages use an asterisk in this way.

Also, I will be using *parentheses*. These are sometimes referred to as *brackets*, but really [and] are brackets,

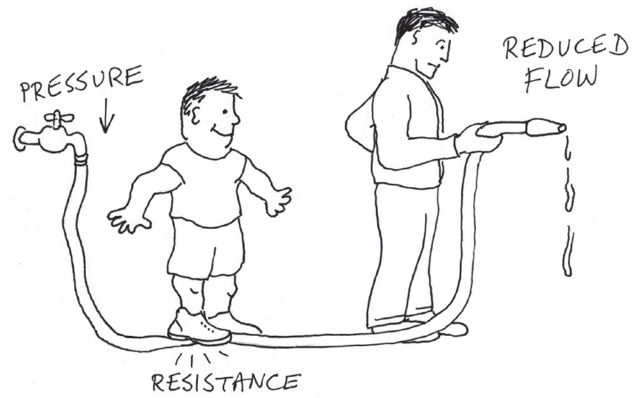


Figure 3-12. Another analogy for electrical pressure, flow, and resistance.

while (and) are parentheses. Suppose I give you a sum like this:

```
A = 13 * ( 12 / ( 7 - 3 ) )
```

If you're wondering which operation you're supposed to do first, the parentheses provide guidance. Notice there is one pair inside another pair. You always begin with the most deeply embedded pair. That would be `(7-3)` in this case, which equals `4`. So you can simplify the formula like this:

```
A = 13 * ( 12 / 4 )
```

And because `12 / 4` is inside parentheses, deal with that next, substituting `3`.

```
A = 13 * 3
```

So the answer is, `A = 39`.

The same rules apply if you are dealing with letters to represent values such as voltage or current. Always start with the most deeply embedded parentheses, and work outward.

It's the Law!

The purpose of this experiment is to make four measurements, and then reach a conclusion about the way that electricity works. This is how science is done: First you need some data, and then maybe you can use it to draw a conclusion.

Disconnect your components and set aside the LED. You won't need it now. You will need the four resistors with values 1K, 1.5K, 2.2K, and 3.3K.

Check the voltage of your battery as in Figure 3-13. It's okay to do this when the meter is set to measure voltage, because the meter has such a high resistance, this protects it from too much current. Never do this with the meter set to measure mA; you can blow the fuse in the meter. Write down the value that you find.

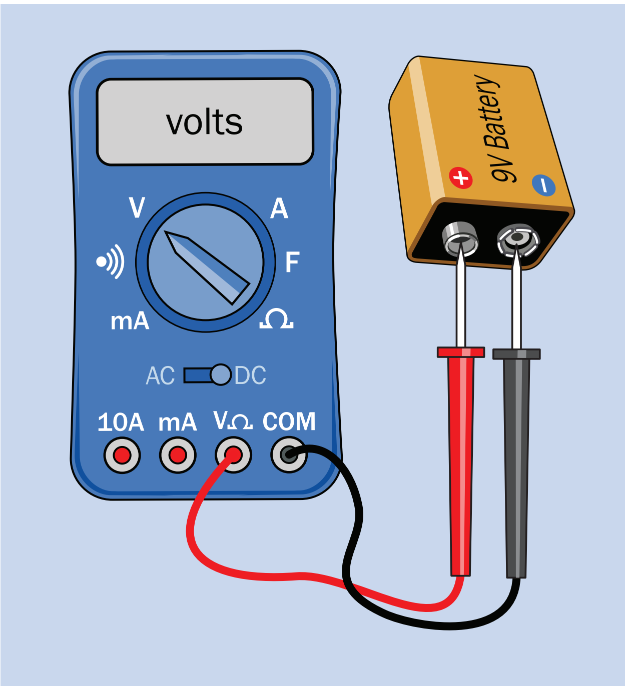


Figure 3-13. If your meter is set to measure volts, you can do this.

Now move the red meter lead to the mA socket, if your meter has one, and set the meter to measure up to 20mA (or 200mA if the range doesn't go as low as 20mA). I want you to make four measurements as suggested in Figure 3-14, which shows the values that I obtained, rounded to two digits.

Notice I have included the actual voltage from my 9V battery. It was 9.6V, because it was a brand-new battery.

Next, I copied the values into a table, as shown in Figure 3-15. For the last column in the table, I used my calculator to multiply the preceding numbers for mA and K. You can do the same with your values. Once again, I rounded mine to one decimal place.

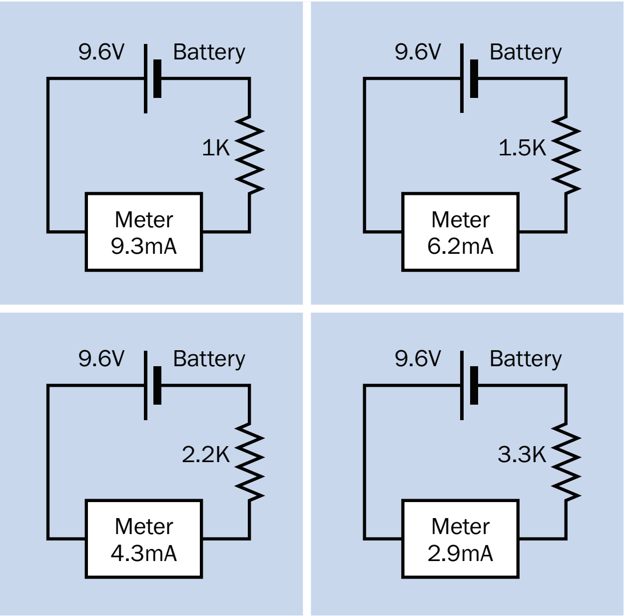


Figure 3-14. Four current measurements obtained with four different resistors.

Do you notice anything odd about this table? The values in the last column are all very similar. And, they are all close to the voltage of the battery.

Anytime you see a coincidence like this, you have to wonder if there is a pattern. If you were doing serious research, you'd try this experiment many times using many other resistors and different power supplies, to make sure. You would also use a very accurate metering system. But I can tell you, the result would be the same every time. It will look like this:

```
voltage = milliamps * kilohms
```

Current in mA	Resistor value in K	mA * K
9.3	1	9.3
6.2	1.5	9.3
4.3	2.2	9.5
2.9	3.3	9.6

Figure 3-15. Table of values transcribed from the measurements shown in Figure 3-14.

I can rewrite it to use the basic values of amps and ohms instead of milliamps and kilohms. Remember:

```
1mA = 1 amp / 1,000
1K = 1 ohm * 1,000
```

Therefore:

```
voltage = (amps / 1,000) * (ohms * 1,000)
```

But the thousands cancel each other out. So:

```
volts = amps * ohms
```

Suppose I use letter V for volts and letter R for resistance in ohms, and letter `I` for amps. Why letter `I`? Because current used to be measured by the effects it *induced*. This is a complicated topic, but in any electronic formula, when you see `I`, it means current. So:

```
V = I * R
```

Remember, V must be in volts, `I` must be in amps, and R must be in ohms. This is known as *Ohm's Law*, the most basic formula in electronics. You just discovered it—or rediscovered it.

In Figure 3-16, the resistance can consist of one resistor or a whole series of them.

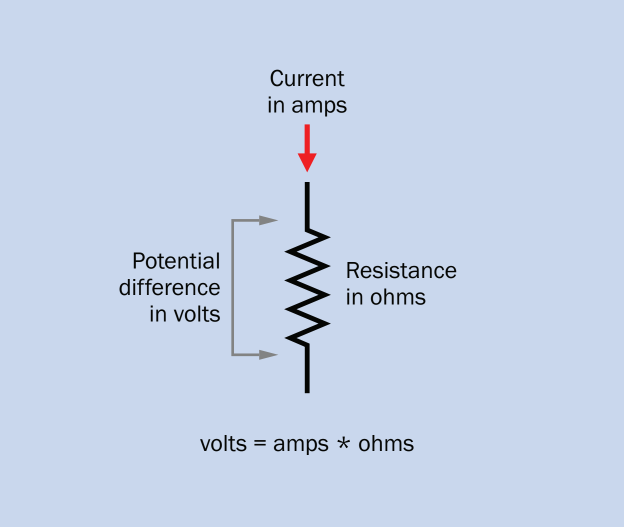


Figure 3-16. How to apply Ohm's Law.

Format Conversion

If you remember high-school algebra, you'll realize that the formula for Ohm's Law can be rewritten by moving the letters around.

If you want to know the resistance, you use this version of the formula:

```
R = V / I
```

If you want to know the current, you use this version of the formula:

```
I = V / R
```

If you want to know the voltage, you use the original formula:

```
V = I * R
```

Decimal Conversion

Legendary British politician Sir Winston Churchill is famous for complaining about "those damned dots." He was referring to decimal points. Because Churchill was Chancellor of the Exchequer at the time, supervising all government expenditures, his difficulty with decimals was a bit of a problem. Still, he muddled through in time-honored British fashion, and so can you.

Because Ohm's Law uses values in volts, amps, and ohms, often you will need to convert milliamps to amps, kilohms to ohms, and back again. Suppose you have a value of 1.2mA and you want to know what that is in amps. When you're converting from a small unit to a unit that is 1,000 times as big, move the decimal point three spaces to the left:

```
1.2 milliamps = 0.0012 amps
```

Suppose you have 230 ohms and you want to know what that is in kilohms. Once again you are converting from a small unit to one that is 1,000 times as big. Imagine that there is a decimal point following 230, because it is really 230.0. Now move the point three spaces to the left:

```
230 ohms = 0.23 kilohms
```

Alternatively, of course, you can use a calculator to multiply or divide by 1,000.

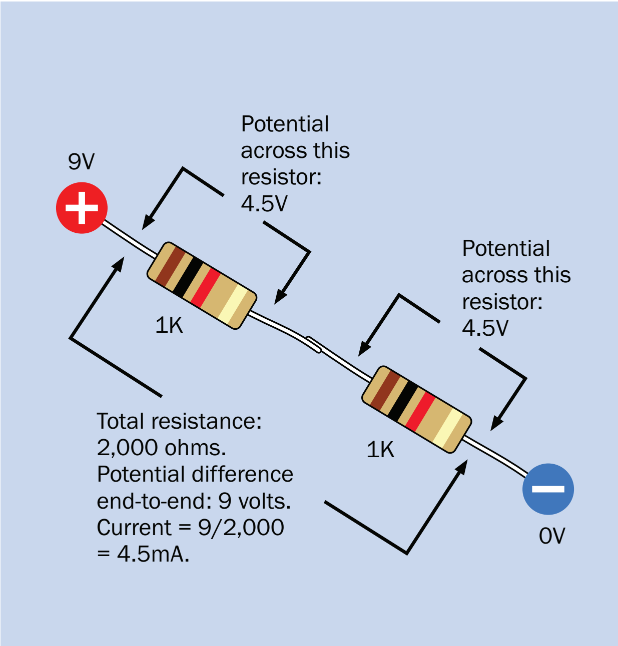


Figure 3-17. To find the total resistance of two equal resistors in series, just add their values.

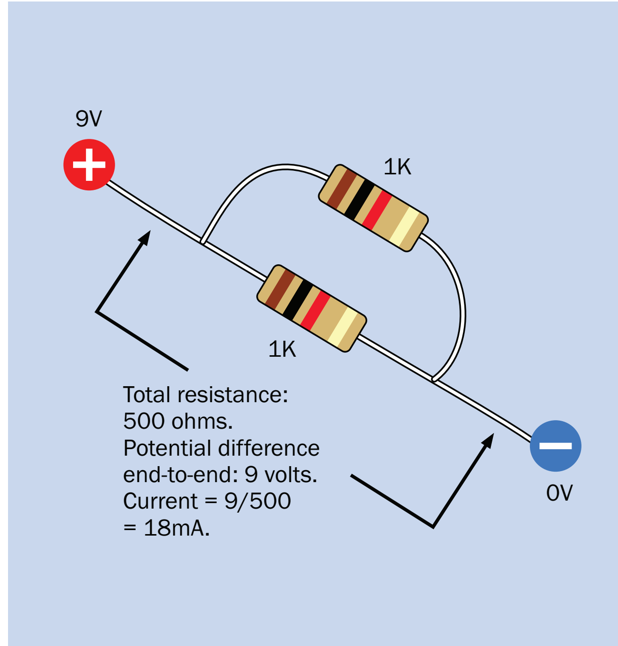


Figure 3-18. To find the total resistance of two equal resistors in parallel, just divide the value of one resistor by 2.

Meter Participation

I skipped over the exact effect that the meter may have when measuring volts. I mentioned that its resistance is very high, but surely it must have some tiny effect?

Yes, it does, which is one reason why I didn't get a consistent result when I multiplied `mA * K`. But there's nothing you can do about it. I rounded the numbers down to 2 digits because I knew the meter would affect the accuracy of its own measurements.

This doesn't sound very satisfactory, but as a general rule, in all aspects of science, the process of measuring something tends to affect the value that you are trying to measure.

Series and Parallel

In the simple circuits that you have built so far, the electricity ran through one component after another. The components were in ***series***.

- When you have two resistances in series, you add their values to get the total resistance. See Figure 3-17.

What if you put two equal resistors beside each other, as in Figure 3-18? Well, now the electricity has two paths to take, so if the resistors are of equal value, the overall resistance is divided by 2. We now say that the resistors are in ***parallel***.

What if the resistors are not equal in value? If they are in series, you add their values as before; but if they are in parallel, it's a bit trickier.

You could investigate this by combining various resistors, and use your meter to measure their resistance in parallel. But let me make it easier for you. If R1 is the value of one resistor, and R2 is the value of another resistor, and they are in parallel, the overall resistance, R, is found like this:

```
1 / R = ( 1 / R1 ) + ( 1 / R2 )
```

That's not very helpful! You want to know R, not 1 divided by R. All right, high-school algebra tells you that you can convert the formula like this:

```
1 / R = ( R1 + R2 ) / ( R1 * R2 )
```

And therefore:

```
R = ( R1 * R2 ) / ( R1 + R2 )
```

Did you really need to know this? Well, yes, it can be useful, and I would not be doing my duty if I didn't mention it. I realize that some people don't enjoy algebra very much, so I won't go into any more detail here. But I do have to include a little math in the next experiment, telling you how to figure out the *power* required by a circuit. That is important.

Cleanup and Recycling

You won't be using the LED in the next experiment, and the only resistor will be one rated for 15 ohms. You can store the resistors that you're not using by putting them in little bags or envelopes, and writing the value on each. Small containers that people use for beads are a better storage option, and I'll have more to say about that in the beginning of Section Five.

The 9V battery should still be good.

Experiment 4
Heat and Power

Electricity can generate heat. You know this must be true, because there are electric water heaters, electric stoves, and electric hair dryers. Also, when the 9V battery burned out the LED, perhaps that was because the LED became too hot.

Perhaps? Once again I have to nail this down with numbers.

You Will Need:

- Nine-volt battery (1).
- Resistor, 15 ohms, brown-green-black (1).
- Cartridge fuse, 5mm diameter, rated 1A (1).
- Cartridge fuse, 5mm diameter, rated 3A (1).
- Test leads (1 red, 1 black).
- Multimeter (1).

Overheating a Resistor

Connect the 15-ohm resistor with your battery as shown in Figure 4-1. But don't try it for long! You'll find that the resistor gets hot quite quickly.

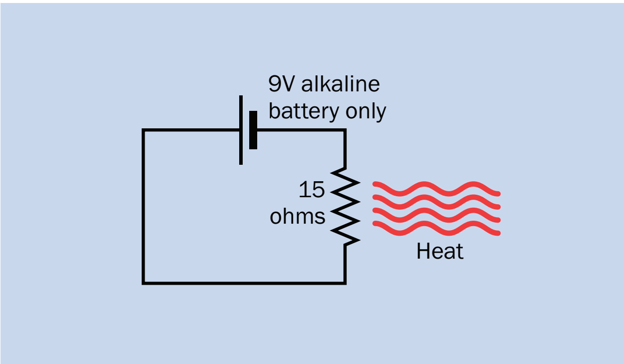


Figure 4-1. If you apply 9 volts across a 15-ohm resistor, it will quickly get hot.

Important note: Don't try this with a bigger battery. Definitely don't try it with lithium batteries that provide power for laptop computers, power tools, phones, and other portable devices. Lithium batteries can behave badly, as in Figure 4-2.

Figure 4-2. This is why it's a bad idea to fool around with lithium batteries.

It's also a bad idea to mess around with lead-acid batteries. Bear in mind that a car battery can deliver enough current to weld two pieces of metal together. Anyone who has ever dropped a wrench across the terminals of a car battery will know this—assuming they survived the experience. This is why a car battery is sold with a plastic cap over the positive terminal. See Figure 4-3.

The 9V alkaline battery in this experiment can't deliver much current, so it should be safe to play with.

Now that you've established that electrons traveling through a resistance do create heat, I want to calculate how much. This will require working in watts.

PS. Don't forget to disconnect that 15-ohm resistor!

Temperature, Heat, and Power

Temperature is an attribute of an object, like its size or its weight.

Heat is a transfer of thermal energy from one place to another. This can be by conduction, convection, or radiation, and is measured in joules, abbreviated **J**. If you get hot by sitting in front of a camp fire, joules of heat energy have been transferred to you from the fire because its temperature is higher than yours.

Power is defined as the rate at which heat is transferred, and is measured in watts. One watt is equivalent to 1 joule per second.

Watts can be defined differently when dealing with electricity:

```
watts = volts * amps
```

However, letter P is often used to represent power, so you may see this formula:

```
P = V * I
```

Figure 4-3. It's a bad idea to fool around with a car battery. It's an even worse idea to drop a wrench across the terminals.

When your 15-ohm resistor got hot, you could feel it with your finger. Really, you were feeling the transfer of heat into your finger, and the faster it happened, the more painful it would be. Also, the more quickly electricity generates heat inside a component, the less chance the component has to get rid of the heat into the world around it. Therefore, the rate of heat transfer is important, and watts are the unit we use to express it.

Figure 4-5. James Watt.

Watt Conversions

Watts can be preceded with a lower-case "m," for "milli," just like volts:

```
1W = 1,000mW
```

Because power stations, solar installations, and wind farms deal with much larger numbers, you may also see references to kilowatts (using lower-case letter k):

```
1kW = 1,000W = 1,000,000mW
```

Intermediate values for milliwatts, watts, and kilowatts are shown in Figure 4-4.

Stereo systems are calibrated in watts. Light bulbs and LEDs are also calibrated in watts.

Milliwatts	Watts	Kilowatts
1mW	0.001W	0.000001kW
10mW	0.01W	0.00001kW
100mW	0.1W	0.0001kW
1,000mW	1W	0.001kW
10,000mW	10W	0.01kW
100,000mW	100W	0.1kW
1,000,000mW	1,000W	1kW

Figure 4-4. Conversion table for units of power.

The Origins of Wattage

James Watt, shown in Figure 4-5, is known as the inventor of the steam engine. Born in 1736 in Scotland, he set up a small workshop in the University of Glasgow where he struggled to perfect an efficient design for using steam to move a piston in a cylinder. Financial problems and the primitive state of the art of metal working delayed practical applications until 1776.

James Watt's development of steam power enabled the industrial revolution, and despite difficulties in obtaining patents (which could only be granted by an act of parliament in those times), he and his business partner eventually made a lot of money from his innovations. Although he predated the pioneers in electricity, his name was not assigned to the basic unit of electric power until 1889 (70 years after his death).

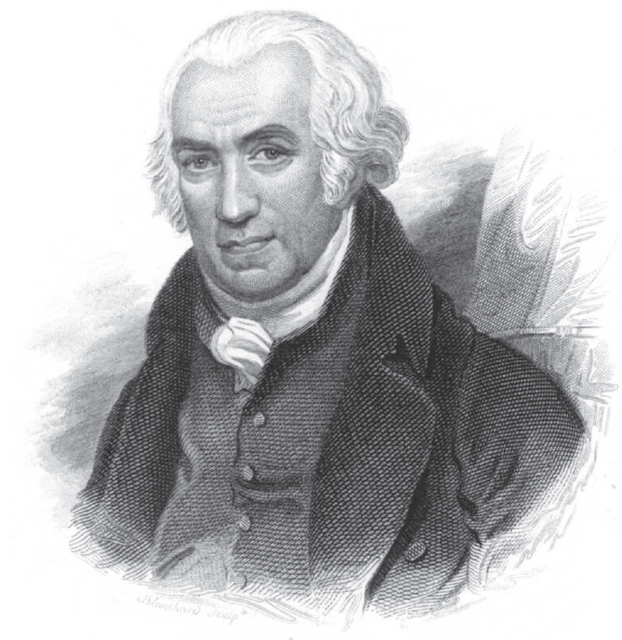


Figure 4-5. James Watt.

Why Didn't Your Tongue Get Hot?

Now you know "what's watt," you can calculate heat transfer very easily. Maybe you thought that when you touched a battery to your tongue, your tongue didn't get hot—but actually it did, just a tiny bit.

How tiny is "tiny"? Because `watts = volts * amps`, we have to begin by figuring out how many amps flowed through your tongue. I will use Ohm's Law for that. You can check back in your notebook to find what you actually measured, but let's suppose your tongue had a resistance of 50,000 ohms when you applied 9 volts across it. You don't know the current, but you want to know it, so you use this version of Ohm's Law:

```
I = V / R
```

And therefore:

```
I = 9 / 50,000 = 0.00018 amps
```

Now you can find watts by multiplying the voltage difference by the current:

```
W = V * I
```

So, `W = 9 * 0.00018 = 0.00162`

What would that be in milliwatts? Move the decimal point three spaces to the right, and the answer is 1.62mW. That's a tiny amount. The high resistance of your tongue limited the number of electrons per second, which limited the amount of heat they could generate. So you didn't feel a thing (apart from the tingle, which was just your nerves being stimulated by electricity).

Why Did the Resistor Get Hot?

The resistors that you have been using are rated to deal with a maximum of one-quarter watt. You can buy resistors that handle more power, but for the rest of the projects in this book, it isn't necessary.

I wonder—did the last experiment with the 15-ohm resistor push it a little too far?

Once again, start by using Ohm's Law to calculate the current. You put a 15-ohm resistor across a 9V battery, so:

`I = V / R`

Therefore, `I = 9 / 15 = 0.6 amps`.

Now that you know the current, you can calculate the watts:

`W = V * I = 9 * 0.6 = 5.4 watts.`

Yikes! One-quarter watt is 0.25 watts, so you were applying more than 20 times as much power to the resistor as it was designed to handle. This is why I told you not to connect it for very long. No wonder it got hot!

Use a Fuse

The formulas that I have provided are wonderfully simple, but they have their limits. Suppose you put a straight piece of wire across the terminals of a 9V battery, and the wire has a tiny resistance, maybe 0.01 ohms. In theory, Ohm's Law says that the battery will push 900 amps through it, and when you multiply that by 9 volts, the power will be 8,100 watts. This is almost enough to power a whole house. Can you disconnect your home from the grid and just wire a 9V battery into your breaker box?

Um, probably not. There's a limit to how much current a battery can deliver, because the chemical reactions inside it, which generate electricity, can't occur fast enough. Ohm's Law only works when your source of current is stable.

So how much power can you get from a 9V battery? You could simply set your meter to measure amps and hook it up to the battery, but that might not be good for your meter. I have a different plan: You can use a fuse.

I included a picture of a fuse back in Figure 1-12. Check it if you need to refresh your memory. This is a ***cartridge fuse***, consisting of a tiny glass cylinder with a metal cap at each end and a thin wire that runs down the middle. The wire is made of a special alloy that melts very easily when too much current runs through it. The purpose of a fuse is to protect other components in a circuit by burning out before they have a chance to burn out. This is known as ***blowing a fuse***.

The fuses you will be using should be 5mm in diameter, so that you can grab them with your test leads. Alternatively, you can use automotive fuses that have little tabs sticking out.

One fuse should be rated at 1A while the other is rated at 3A. They appear identical, but if you look very closely with a magnifying glass, one of the end caps of each fuse will have 1A or 3A engraved in the metal, probably with a couple other numerals or letters too.

After you check the fuses, don't get them mixed up! Maybe put a different colored dot on the end of each one, using a marker pen.

- A fuse has no polarity. You can use it either way around.

Start with the 1-amp fuse. Attach it to the battery as in Figure 4-6. Press the loose alligator clip against the fuse for only two seconds. That should be enough to melt it. Don't worry about the fuse getting hot; the element inside the fuse will melt very quickly, before you can feel heat outside the fuse.

The schematic version of Figure 4-6 is shown in Figure 4-7, which shows a variety of symbols that are used to represent a fuse. Why are there so many? I really don't know. There are even more that I didn't show here. I just chose the ones that seem most common.

Remove the fuse from the circuit, and take a close look. (Don't peer closely at a fuse which still has current

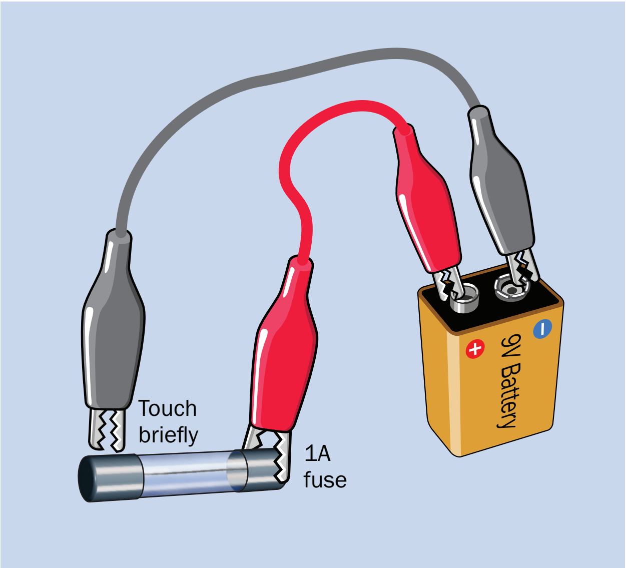


Figure 4-6. How to blow a 1-amp fuse.

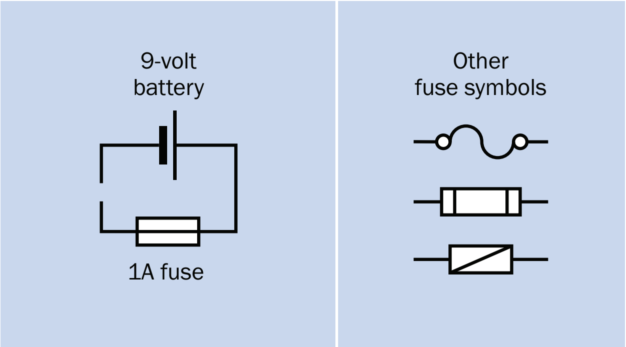


Figure 4-7. The schematic version of Figure 4-6.

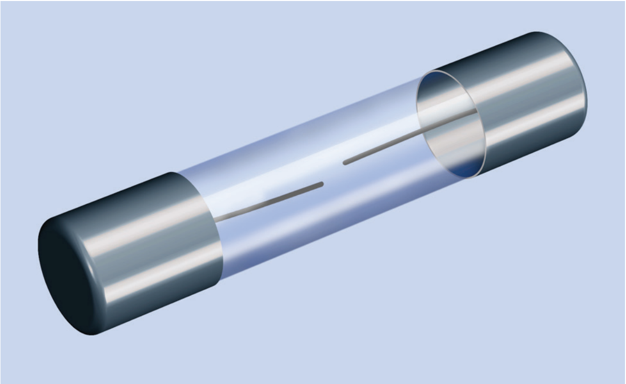


Figure 4-8. This fuse has blown.

running through it. That's a bad habit. If a fuse is heavily overloaded, it can blow up more dramatically.)

Do you see a break in the element? It should look like Figure 4-8. If you don't see a break, repeat the experiment for three seconds.

- Fuses are not accurately rated. Some 1-amp fuses may require more current to melt them than others.

Because you melted the 1-amp fuse, you can feel sure that more than 1 amp flowed through it—assuming it was accurately rated. But how much more?

Let's try the 3-amp fuse. Repeat the experiment, and this time leave the fuse connected for four seconds. Remove the fuse, and check it. I am hoping and expecting that this fuse didn't melt. It didn't, did it? So now you know that the current is more than 1 amp, and less than 3 amps—assuming the fuses are accurately rated. Even if they are not accurately rated, the 3-amp fuse surely must melt more easily than a 10-amp fuse. Therefore, it would be safe to measure current with your meter, so long as you use the socket labeled **10A** or **20A**—although I would keep the 3-amp fuse in the circuit, just in case.

Set it up as in Figure 4-9, on the next page. This time you have to make absolutely sure you have the red lead plugged into the socket marked **10A** or **20A**. Turn the dial to measure amps (there will probably be a specific setting for 10A or 20A), touch the meter probe briefly against the fuse, just long enough to read the display, and what do you see? I got 1.6A when I tried it.

Incidentally, meters are not designed to pass high currents for very long, even using the **10A** or **20A** socket. Typically they can deal with the maximum current for maybe 15 seconds, after which they need a half-hour cooling-off period.

After this experiment, move your red meter lead back to the volts-ohms socket. Now you can connect the meter directly to the battery to measure its voltage, as you did back in Figure 3-13.

My battery voltage was 8.73. In your notebook, you should have accumulated a whole series of measurements, so you can see the effects that the experiments had on battery voltage. Batteries don't last very long when they are loaded beyond the limit for which they are designed.

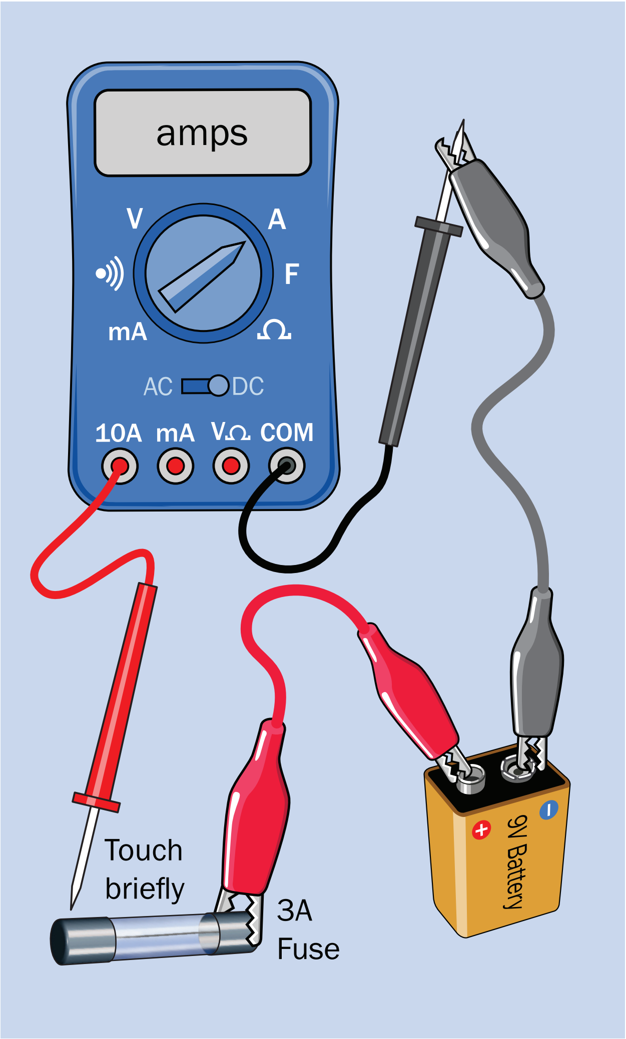


Figure 4-9. Finding out how much current you can get from a 9V battery.

Current Conclusions

What can you conclude from this experiment?

- Even a little 9V battery can deliver more than 1 amp. In theory, that's enough to light 50 LEDs, if they were wired in parallel.
- But, a few seconds delivering this current will deplete the battery.

How much current ***should*** the battery have delivered through the fuse, according to Ohm's Law? Well, that depends on the resistance in the circuit . You may think that the resistance of a meter and a fuse is next to nothing, but actually the fuse has more resistance than a piece of ordinary wire. You can take it out of your circuit and measure it with your meter. My fuse has a resistance of about 0.5 ohms.

Now, what about the meter itself? You can't tell it to measure its own resistance, but you can look it up in the specifications. The internal resistance of my meter is about 0.5 ohms when using the **10A** socket.

In addition, you used a couple of test leads. You might think their resistance is almost zero, but I think manufacturers have been economizing on the thickness of wire in test leads during the last 10 years or so. I put ten leads together and measured their overall resistance as about 2 ohms. This includes resistance where the alligator clips make contact. So two test leads probably have a resistance of about 0.4 ohms, or more if you are using long ones. And then there are the meter leads, too.

Because the meter, the fuse, and the leads were all in series in this experiment, you were loading the battery with a total of about 1.5 ohms, more or less. So, according to Ohm's Law:

```
I = V / R = 9 / 1.5 = 6 amps
```

There's still one more factor. Even the battery itself has resistance! According to sources that I checked, a 9V alkaline battery has an internal resistance of about 1.5 ohms. Therefore, the battery should have delivered 3 amps. But as I said at the beginning, Ohm's Law doesn't necessarily apply when you're stressing components beyond their limits.

Cleanup and Recycling

You can throw away the blown fuse.

The battery may still be useful, depending how long you kept it connected with the fuse. I would store it for future use if its voltage is 8.7V or higher, but check it again before using it in future experiments.

In most areas, batteries can be recycled. In some areas, a law may compel you to recycle batteries.

Experiment 5
Let's Make a Battery

Long ago, before the web existed, kids were so horribly deprived, they tried to amuse themselves with kitchen-table projects such as making a battery by pushing a nail and a penny into a lemon. Hard to believe, but true!

The old lemon-battery experiment is actually more interesting now that we have LEDs that will respond to a very small flow of current. If you've never tried it, the time is right.

You Will Need:

- Lemons (2) or, preferably, a squeeze-bottle of pure lemon juice (1).
- Copper-plated coins, such as U.S. pennies (4). If copper coins are not available where you live, see page 291 in Appendix A for some other options.
- Galvanized (zinc-plated) steel brackets or mending plates from a hardware store. 1" minimum (4).
- Test leads (1 red, 1 black, 3 of another color).
- Multimeter (1).
- Generic red LED (1).

Setup

A battery is an ***electrochemical*** device, meaning that chemical reactions create electricity. Naturally, this only works if you have the right chemicals working for you, and the ones I'm going to use are copper, zinc, and lemon juice.

The juice should be no problem. Lemons are cheap, or you can buy one of those little yellow plastic squeeze-bottles of concentrated juice.

Pennies are not made of copper anymore, but they are still plated with a thin layer of copper, which is good enough. Just make sure that your pennies are new and bright. If the copper has oxidized, it will be a dark, dull brown, and the experiment won't work as well.

Zinc is a bit more of a problem. What you need is a metal part that is galvanized, meaning it is coated in zinc to prevent rust. Small galvanized steel brackets (1" on each side) or mending plates (1" long) should be available at your local hardware store. A galvanized finish is silver-gray in appearance.

Lemon Test: Part One

Cut a lemon in half, and push a penny into it. As close as possible to the penny, ***but not touching it***, push in your galvanized mending plate or bracket. Now set your multimeter so that it can measure up to 2V DC, and hold the red probe against the penny while you hold the black probe against the mending plate. You should find that your meter detects between 0.8V and 1V.

To power your LED, you need more forward voltage. How can you get it? By putting batteries in series. In other words—more lemons! You can use alligator test leads to link the batteries, as shown in Figure 5-1. Notice that

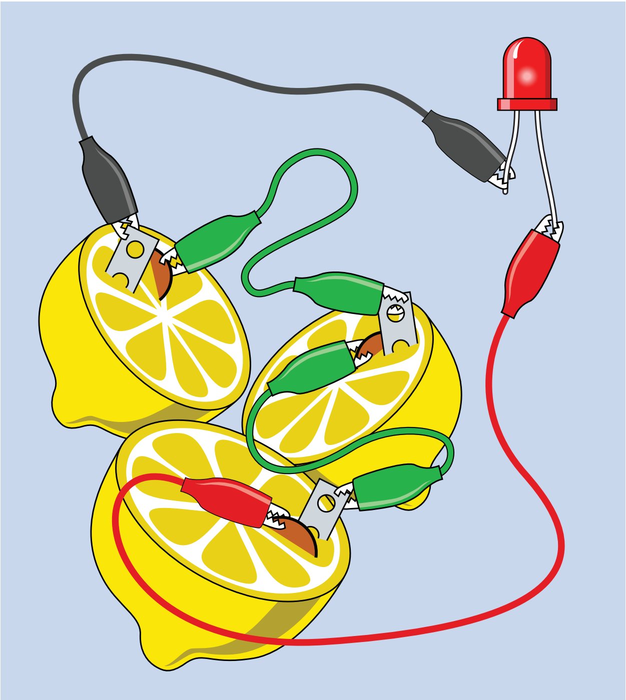


Figure 5-1. A three-lemon battery should generate just enough current to drive an LED.

each lead connects a bracket to a penny. Do not connect pennies to pennies or brackets to brackets.

If you set things up carefully, keeping the pennies and brackets close to each other but making sure they don't actually touch each other, you should be able to illuminate your LED with three lemon-juice batteries in series.

Another option is to use a little parts box, as shown in Figure 5-2. The divisions in the box create cells that can be assembled in series, and they also make it easier to grip the pennies and brackets. When everything is nicely

Figure 5-2. Lemon juice, either from lemons or from a squeeze bottle, will produce reliable results.

aligned, squeeze in some concentrated lemon juice. Vinegar or grapefruit juice may also work.

I decided to have four cells in my juice battery, because the load applied by the LED pulls the voltage down somewhat, and the battery is not capable of delivering enough current to do any damage. Your juice battery has a relatively high internal resistance!

The setup in the photograph worked immediately.

The Nature of Electricity

To understand why the lemon battery works, you have to start with some basic information about atoms. Each atom consists of a nucleus at the center, containing particles called ***protons***, which have a positive charge. The nucleus is surrounded by ***electrons***, which carry a negative charge. As I have mentioned previously, electric current consists of electrons.

Breaking up the nucleus of an atom requires a lot of energy, and can also liberate a lot of energy—as happens in a nuclear explosion. But persuading a couple of electrons to leave an atom (or join an atom) can take relatively little energy. When an atom has lost or gained one or more electrons, it is known as an ***ion***. If it's short of electrons, it's a ***positive ion***, and if it has a surplus, it becomes a ***negative ion***.

A chemical reaction between zinc and lemon juice results in positive zinc ions being pulled out of the zinc electrode, leaving the electrons behind. They are free to roam—but where?

I tend to think of electrons as being bad-tempered little people. After all, they do suffer from mutual repulsion, as suggested in Figure 5-3. But your lemon battery allows them an opportunity to liberate themselves, as in Figure 5-4. They can run along a wire, through a load of some kind, such as an LED. They lose a little energy into

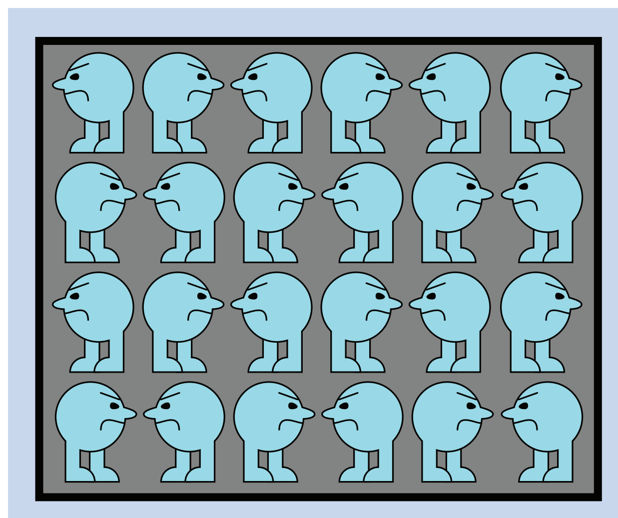


Figure 5-3. A group of electrons experiencing mutual repulsion.

the load, but they continue on into the copper electrode. Lemon juice surrounding the copper just happens to contain positive hydrogen ions that would like to have negative companions.

Pairs of electrons emerge from the copper and hook up with pairs of the hydrogen ions, forming tiny bubbles of hydrogen gas. The bubbles rise up out of the lemon juice, and the electrons leave with their new-found friends, in search of fresh adventures.

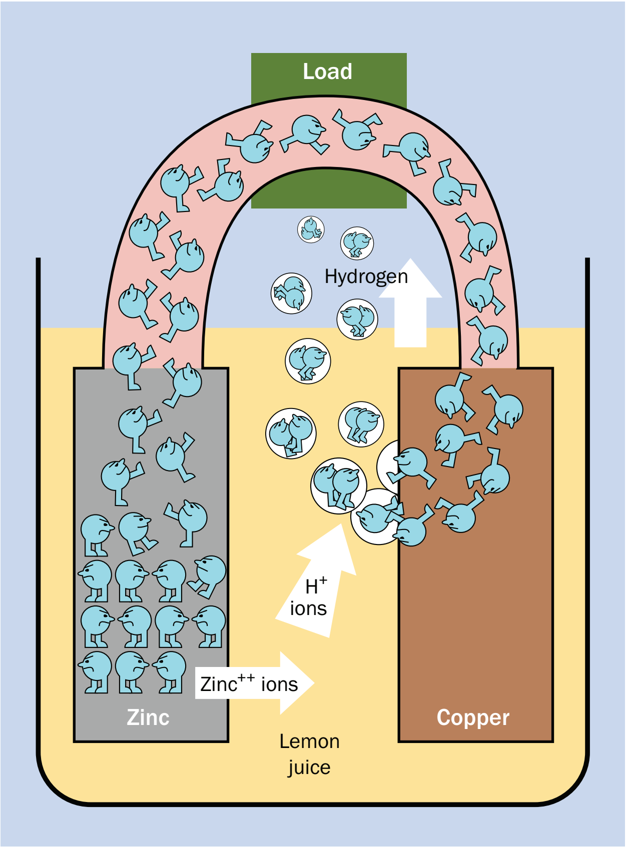


Figure 5-4. Electrons being liberated from a lemon battery.

The reality isn't quite as simple as I have made it sound, but I leave you to read more about it if you want to know the full story.

This description applies to a ***primary battery***, meaning one that is ready to generate electricity as soon as a connection between its terminals allows electrons to transfer from one electrode to the other. The amount of current that a primary battery can generate is determined by the speed at which chemical reactions inside the battery can liberate electrons. When the raw metal in the electrodes has all been used up in chemical reactions, the battery can't generate any more electricity and is dead. It cannot easily be recharged, because the chemical reactions are not easily reversible, and the electrodes may have deteriorated.

In a rechargeable battery, also known as a ***secondary battery***, a smarter choice of electrodes and electrolyte does allow chemical reactions to be reversed.

Positive and Negative Confusion

I've told you that electricity is a flow of electrons, which have a negative charge. In that case, during the experiments that you have performed so far, why have I been talking as if electricity flows from the positive terminal to the negative terminal of a battery?

The answer is that in the year 1747, Benjamin Franklin concluded from his observations of thunder storms that lightning travels down from positive storm clouds and grounds itself in the earth below.

In fact he was partially correct: Some storm clouds may have a higher positive charge than the earth, but under those conditions, someone who is "struck by lightning" may actually be hurt by an electrical discharge traveling up from the ground, through his body, and out from the top of his head, as in Figure 5-5.

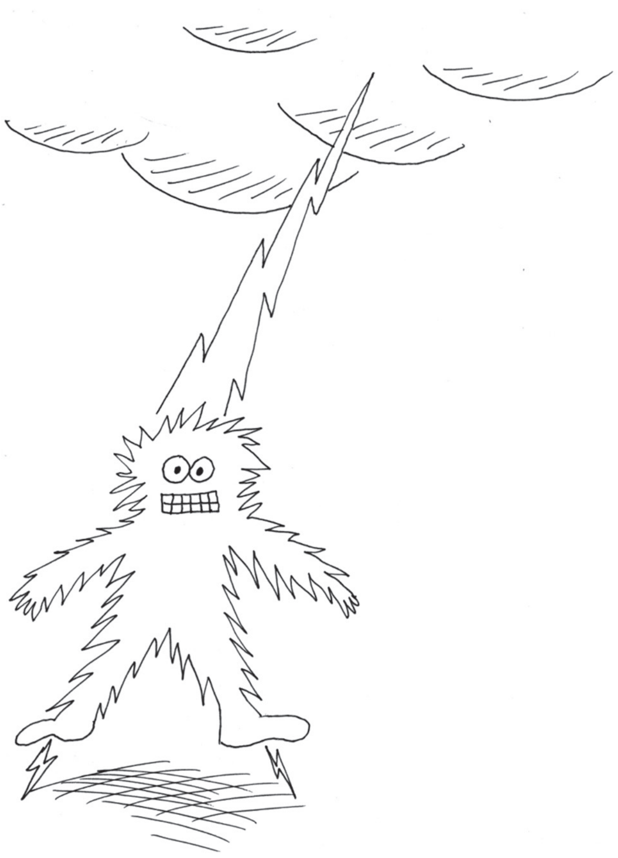


Figure 5-5. Although this unfortunate person may think he has been struck by lightning, it could be just the opposite.

Franklin's error was not resolved until 1897, when physicist J. J. Thomson announced his discovery of the electron and proved that electricity is a flow of negatively charged particles. In a battery, electrons originate from the negative terminal and flow to the positive terminal.

You might think that when this fact was established, everyone should have forgotten about Ben Franklin's error. But really a flow of negative charges to a more positive location is equivalent to a flow of positive charges to a more negative location. When the electron leaves home, it removes a small negative charge from that location; therefore, its home becomes a bit more positive, almost as if a positive charge flowed in the opposite direction. Moreover, all of the mathematics describing electrical behavior are still valid if you apply them to an imaginary flow of positive charges.

As a matter of tradition and convenience, the concept of flow from positive to negative survived, because everyone had been thinking in those terms for more than a century, and in the end, it made no difference.

- The flow from positive to negative is now properly referred to as ***conventional current***.

In the schematic symbols that represent components such as diodes and transistors, you will find arrows reminding you which way these components should be placed—and the arrows all point from positive to negative. Throughout this book, I will describe current that way.

Cleanup and Recycling

The hardware that you immersed in lemons or lemon juice may be discolored, but it is reusable. I think you should avoid eating the lemons, as some metals will have been deposited in them by the electrochemical reaction.

Section Two

Switching

Using a *switch* to control electric current seems a very basic idea, but in this section I won't just be talking about switches that you push with your finger. One flow of electricity can switch another flow, using a *relay* or a *transistor*. This type of switching is so important, all digital devices depend on it.

Also in this section I will deal with *capacitance*, because the concept is as fundamental as resistance in electronic circuits.

As in Section One, I will begin with a catalog of recommended tools, supplies, and components, so that you can recognize them easily. You may already have some of these items, but if not, please turn to the back of the book for instructions on how and where to buy them:

- See Appendix A for specifications which you will need to make purchases.
- See Appendix B for sources that I recommend when you are shopping online or in physical stores.

Miniature Screwdrivers

You will need *miniature screwdrivers* for the little screws that you find on some electronic components. I suggest a set such as the one shown in Figure 6-1.

Some low-cost sets look very similar to the one shown, but if you pay slightly more for a name brand, you may find that the quality of materials and manufacture is better.

Figure 6-1. Miniature screwdrivers, four with flat blades, two with Phillips blades.

Small Long-Nosed Pliers

The type of *long-nosed pliers* that you need are no more than five inches from end to end. You'll be using them to bend wire precisely, or to pick up small parts where your finger and thumb are too big and clumsy. The pliers shown in Figure 6-2 have spring-loaded handles, although some people prefer pliers without this feature. See the photograph on the next page.

Because you won't be using these pliers for heavy work, you can buy the least expensive ones that you can find.

Figure 6-2. Appropriate pliers for electronics work should be no more than 5" long.

Wire Cutters

Pliers usually have cutting edges near the joint, but to snip wire in an inaccessible location, you need ***wire cutters*** such as those shown in Figure 6-3. They should be no longer than five inches. They do not have to be of high quality, as you will be using them mostly to cut soft copper wire.

As in the case of pliers, wire cutters are available with or without spring-loaded handles.

Figure 6-3. Wire cutters should be no more than five inches long.

Flush Cutters (optional)

Flush cutters, shown in Figure 6-4, are thinner and smaller than wire cutters, but may be less robust. Whether you choose them or wire cutters is a matter of personal preference. Personally I like wire cutters.

Figure 6-4. Flush cutters may reach into tinier spaces than wire cutters.

Sharp-Nosed Pliers (optional)

These are similar to long-nosed pliers, but they have slender, precise jaws that taper to a point. I use them a lot for accessing tightly packed components on a breadboard. Your best source for this tool is a web site or store that specializes in craft work such as beading. Make sure that the inner surfaces of the jaws are flat, as shown in Figure 6-5. You don't want the type of beading pliers that have rounded surfaces for making loops in jewelry wire.

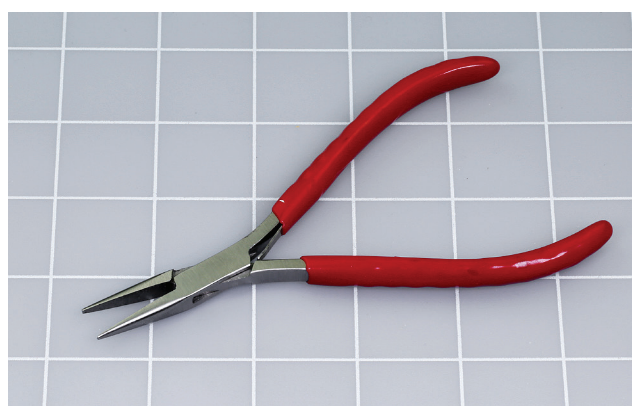


Figure 6-5. Sharp-nosed pliers enable very precise work on a small scale.

Wire Strippers

Wire of the type that you will be using is coated with plastic insulation. When you need to remove some of this insulation, ***wire strippers*** such as those in Figure 6-6 are the best tool for the job.

Some wire strippers have angled handles, some have straight handles, and some have curved handles. Personally, I don't think this matters.

Figure 6-6. Wire strippers for removing insulation from wire.

You do have to be careful that the strippers which you buy are suitable for the size of wire that you will be using. Look on the jaws of the strippers for a range of sizes containing the number 22, suitable for 22-gauge wire. Suitable ranges might be 14 to 24 or 20 to 30. Gauge numbers are explained in the description of hookup wire, on the next page.

You may run across ***automatic*** wire strippers if you shop online. Supposedly they enable one-handed stripping of insulation, as they grip the wire, bite into it, and pull the insulation off in one operation. I tried them for a while, but found they lacked precision and didn't really work well.

Testosterone-driven makers may claim that they don't need any tool at all, to strip wire, but I have two corners missing from the insides of my front teeth to remind me that this is a bad idea. See Figure 6-7.

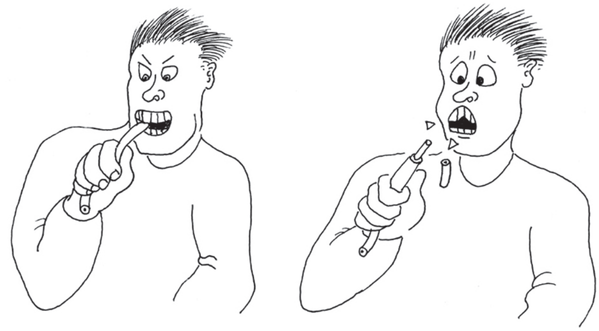


Figure 6-7. In a hurry? Can't find your wire strippers? Resist the temptation. This really can happen.

Breadboard

For electronics purposes, a ***breadboard*** is a small plastic slab drilled with holes at intervals of 0.1". Hidden inside the plastic are little spring clips which establish connections between the wires and components which you push into the holes. This is a much quicker way to build a circuit than by using the alligator test leads in Section One.

You may find that a breadboard is described as a ***solderless breadboard*** or a ***prototyping board***. These are different names for the same thing.

A ***mini-breadboard*** is shown in Figure 6-8. Often sold as being "suitable for Arduino," it doesn't have enough holes in it for our purposes, so you don't need to buy one of these.

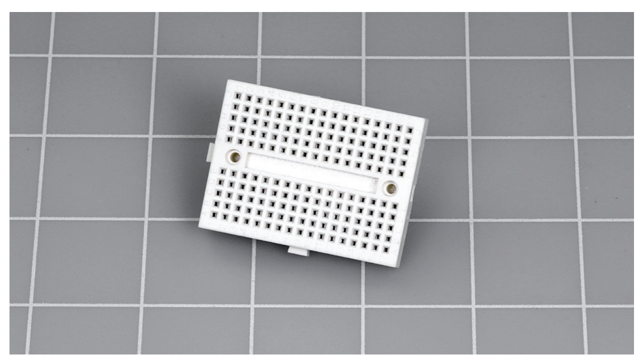


Figure 6-8. A mini-board is not big enough for most of the projects in this book.

A *single-bus breadboard* is shown in Figure 6-9. The term *bus* refers to each column of holes parallel with the long edge of the board. I outlined them in red in the figure.

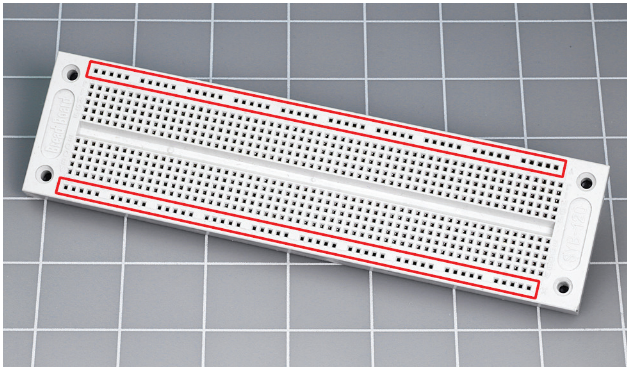


Figure 6-9. A single-bus breadboard. Each bus is outlined here in red. This type of breadboard is not recommended for this book.

A *dual-bus breadboard* is shown in Figure 6-10. This has two long rows of holes on each side, which are the dual buses. The red and blue stripes are printed on the board to provide guidance.

In the first edition of *Make: Electronics*, I used dual-bus breadboards because their layout minimizes wire runs to components. However, I found that some readers made wiring errors because the two buses on each side are easily mixed up. In the second edition of this book, I changed all the circuits to single-bus breadboards, which worked well—but single-bus boards are now uncommon and hard to find. Therefore, in this third edition of the book, I had to go back to dual-bus breadboards.

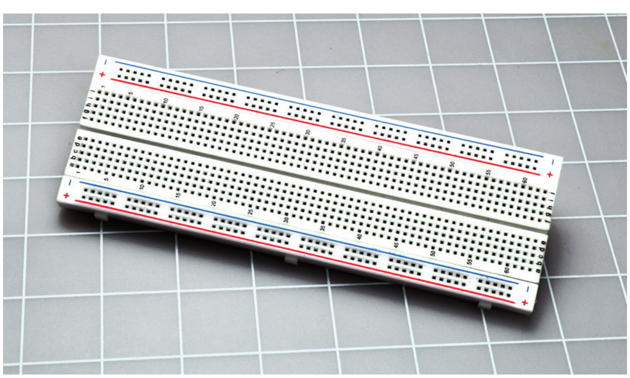


Figure 6-10. A dual-bus breadboard, used throughout this book.

The type shown in Figure 6-10 is what you need. The manufacturer is unimportant, but when you read the description of the board, it must mention at least 800 holes (often referred to as *tie points*).

I used to say that you only need one breadboard, because they are reusable. However, their price has been driven down to the point where you may consider buying two or three. This will enable you to prototype new circuits without having to disassemble old ones.

Hookup Wire

The type of wire that you use to make connections on a breadboard is often called *hookup wire* although it can be found, sometimes, under the general category of *bulk wire*. Often it is sold in 25-foot and 100-foot lengths on plastic spools, as shown in Figure 6-11.

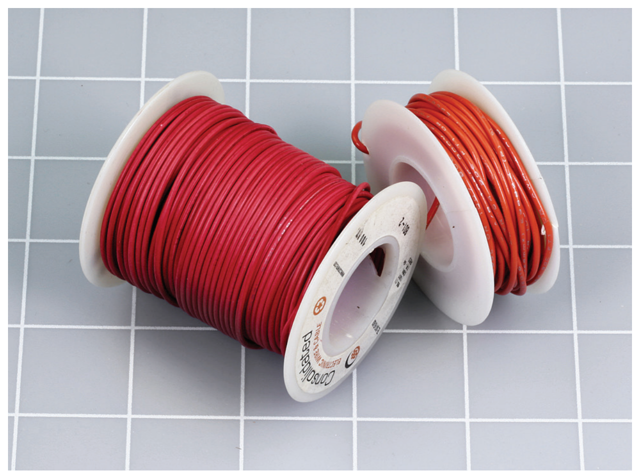


Figure 6-11. Hookup wire is available on spools holding 100 feet and 25 feet, shown here.

When the insulation is stripped away, it reveals a solid conductor, as shown in Figure 6-12. Compare this with stranded wire, shown in Figure 6-13. Stranded wire is more flexible that solid wire, but if you try to push it into a hole in a breadboard, some strands won't go in properly, and the experience can be frustrating. You really need solid wire for this purpose.

American wire gauge (often referred to with the acronym *AWG*) is a number describing the thickness of a conductor. A higher number indicates a thinner wire. If you want to convert wire gauge to inches or metric measurements,

Figure 6-12. Solid hookup wire has a single conductor inside the insulation.

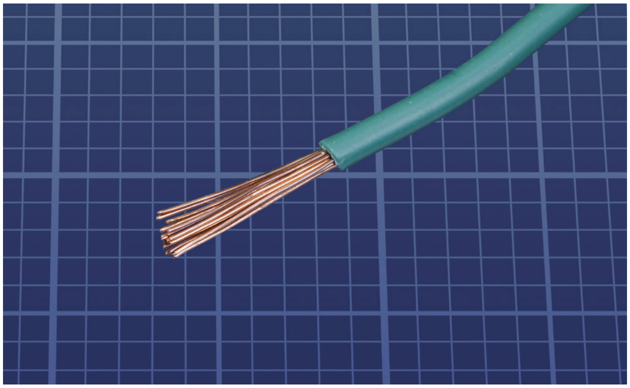


Figure 6-13. The advantages of stranded wire is that it is more flexible than solid wire.

you can easily find tables or calculators online. Use a search term like these:

```
convert awg inches
```

```
convert awg metric
```

For the projects in this book, the optimal thickness is 22 gauge. In a previous edition of this book I suggested using 20-gauge (thicker) wire because it may make a more reliable connection in a breadboard, but I discovered that some boards have tighter clips than others, and are difficult to use with thicker wire. Therefore, 22-gauge wire has to be the best compromise.

Some copper wire has a silver-colored coating which becomes visible when you strip the insulation away. This type of wire is said to be ***tinned***. For the projects in this book, you can use either tinned wire or plain copper wire.

I suggest you buy four spools of 25 feet each, in red, blue, green, and yellow. Distinctive colors are helpful to avoid making errors, or when you are trying to find a wrong connection in a circuit. I use red and blue to identify the positive and negative power-supply wires in all the circuits in this book, while green and yellow make intermediate connections. You may find that black is a more popular color than blue when you are searching for wire, and you can substitute black if you prefer.

Stranded Wire (optional)

When you are installing a finished circuit in a box, stranded wire is convenient to make connections between switches and the circuit board, because it's more flexible. If you decide to buy stranded wire for this purpose, I suggest it should be 22-gauge. If you use a color other than red, blue, green, or yellow, this will distinguish it from your hookup wire.

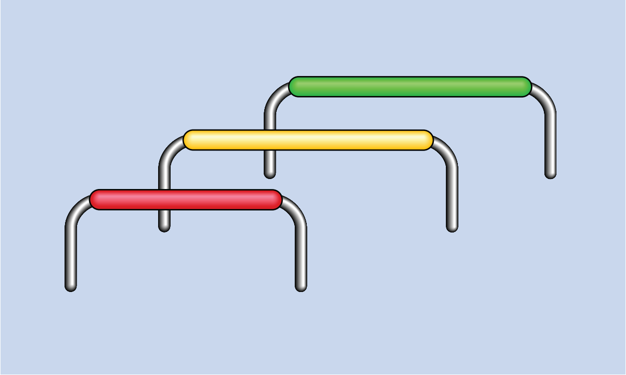


Figure 6-14. Jumpers for making breadboard connections.

Jumpers

If you strip about 3/8" insulation from each end of a short piece of wire, bend the ends down, and push them into holes in a breadboard, you have made a ***jumper*** of the type shown in Figure 6-14. It creates a connection by jumping across some intervening rows of holes, and if you cut it to the right length, you end up with a neat circuit in which errors are easy to find.

Stripping insulation and making right-angle bends should not take too long, but some people get impatient with it. Therefore you may be tempted to purchase *precut jumpers* in an assortment such as the one shown in Figure 6-15.

I used to use precut jumpers myself, but I stopped because the colors of insulation are confusing. Each color identifies the *length* of the jumper, but in my circuits I want the colors to relate to the *function* of the jumper. For instance, I want red wires to be connected to the positive side of the power supply, regardless of how long or short they are.

You can use the precut jumpers if you wish, but in addition to being confusingly colored, they cost more.

What about flexible jumper wires with little single-pin plugs on the ends, as shown in Figure 6-16? These may be the first thing you will find if you search for jumper wires online. Because they are flexible, they don't have to be cut to size, and can connect holes that are anywhere from 0.1" apart to 3" apart. One size fits all! That seems to be an attractive option.

Figure 6-15. An assortment of precut jumpers.

Unfortunately this type of jumper wire is only convenient so long as you don't make any wiring errors—and in real life, everyone makes wiring errors. The time that you saved by creating quick connections will then be lost as you struggle to find the wrong connection amid a jumper tangle.

Figure 6-17 shows a small circuit created on a couple of mini-boards using flexible jumpers. Figure 6-18 shows exactly the same circuit with hand-cut jumper connec-

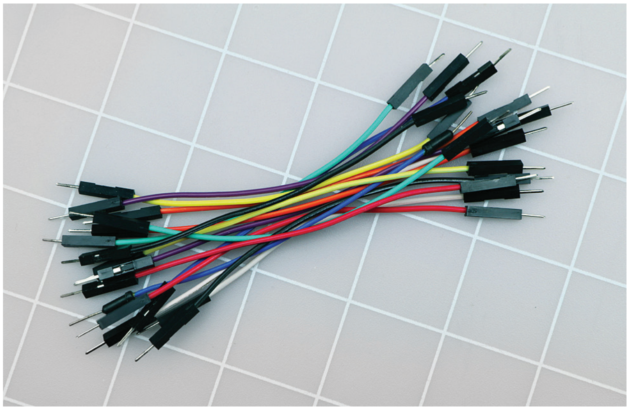


Figure 6-16. Flexible jumper wires with plugs on each end.

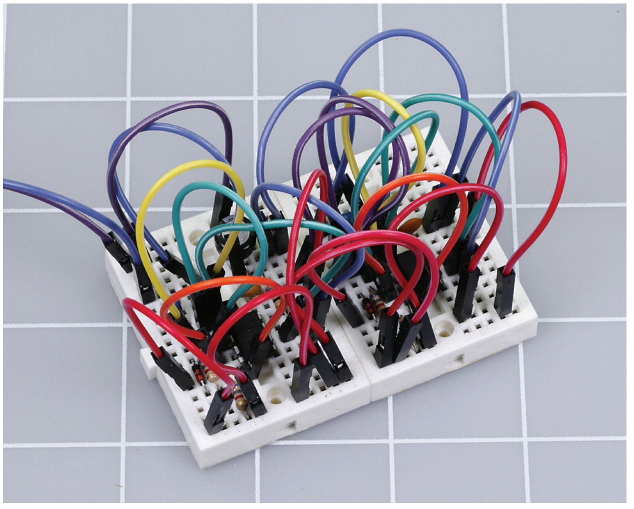


Figure 6-17. A circuit on two mini-boards using flexible jumpers.

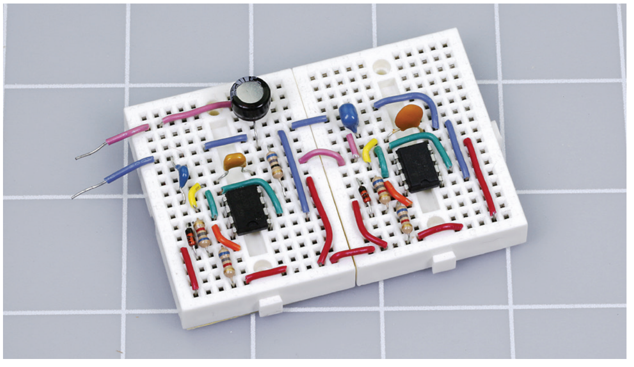


Figure 6-18. The same circuit as in Figure 6-17, using hand-cut jumper wires.

tions. Now, which circuit would you prefer to be dealing with, if you are trying to find an error?

Incidentally, if your circuit doesn't work, and you send me a photograph asking where the fault is, I may be able to help you if you use hand-cut jumper wires, but I probably won't be able to help you if you use flexible jumper wires.

If I haven't convinced you yet about the need to use hand-cut jumper wires of the right length, bear in mind that the pins on flexible jumpers are sometimes defective, and may contain loose connections even though they seem to be perfectly okay. This can make fault-tracing quite frustrating. Pins also tend to break off after you use them for a while.

Still, if you really, really want to use flexible jumpers, I suggest you buy the "premium" variety from `adafruit.com`, which seem to have more reliable plug connections.

Slide Switch

There are dozens of types of switches, but for the experiments in this book, you need *slide switches*. In Experiment 6, it will be helpful to have a couple of switches like the type shown in Figure 6-19, with pins spaced 0.2"apart, which are easy to grab with alligator test leads.

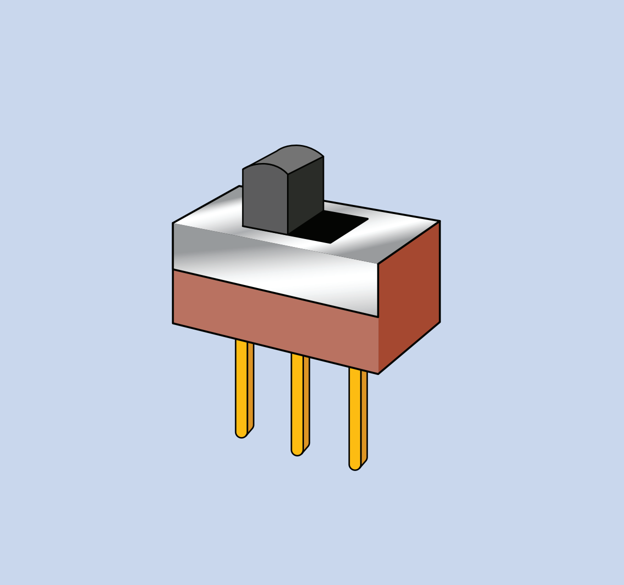


Figure 6-19. A slide switch with pins spaced at intervals of approximately 0.2".

For all other experiments in the book you will need slide switches with pins spaced 0.1" apart so that they can be inserted in a breadboard. A slide switch of this type is pictured in Figure 6-20.

If you order a kit of components for this book, it should contain two larger-size slide switches. If you don't order a kit, and you have difficulty finding larger slide switches, the smaller type will be usable with alligator test leads if you bend the pins outward a little (see Figure 6-30).

Because the experiments in this book will switch low voltages and currents, you don't need to buy heavy-duty or expensive switches.

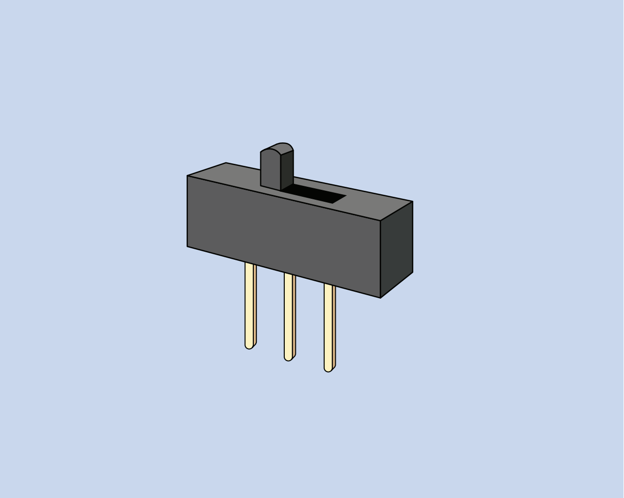


Figure 6-20. An all-plastic slide switch with pins spaced 0.1".

Tactile Switch

This component looks like a tiny pushbutton, and it works like a pushbutton, even though it is properly called a *tactile switch*. If you plug it into your breadboard, it provides a convenient way for you to trigger a circuit by using the pressure of your finger. Four common types are shown in Figure 6-21 on the next page.

Type A is the one I prefer, as it has smooth, straight, rounded pins that slide easily into a breadboard. Unfortunately this type is less common than the others.

Type B is acceptable, but the pins are usually spaced 6.5mm. This is almost, but not quite, appropriate for a breadboard (if you convert from metric to inches, 6.5mm is about 2.6"). However, you can make it fit. If you grip each pin firmly with pliers, you can flatten the kinks out. Bend the pins gently until they are 0.2" apart, and the switches that I have tested will sit securely in the holes of a board.

Type C has acceptable pins 0.2" apart, but my breadboard layouts don't allow quite enough room for the square body of this switch.

Type D is the most common tactile switch, but its four kinky pins don't slide easily into a breadboard, and they tend to pop back out. Also, the square body is a fraction too big, and my layouts won't accommodate four pins.

Part numbers and sources are listed in the Appendix A and Appendix B, along with other components.

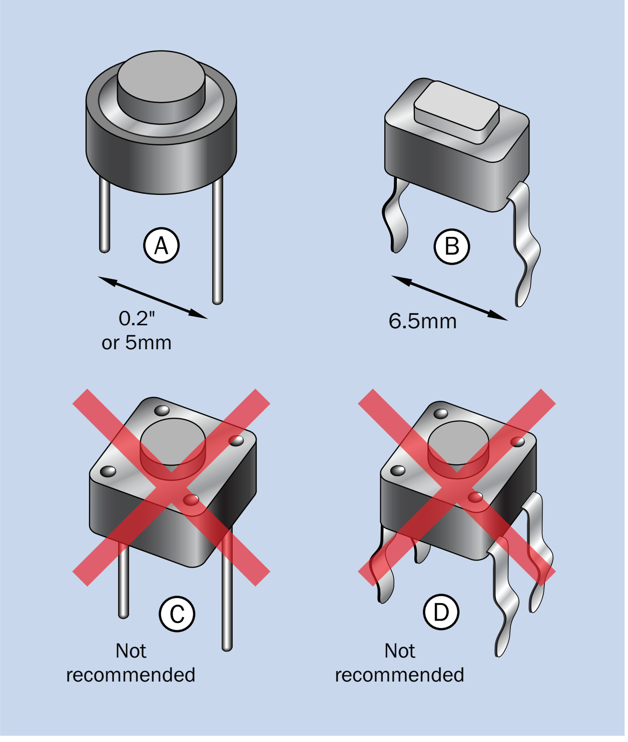


Figure 6-21. Four types of tactile switch.

Relay

A *relay* contains a switch which is activated by remote control. Because pin functions are not standardized among manufacturers, you have to be cautious about making substitutions when you buy this component. Figure 6-22 shows the `Omron G5V-2-H1-DC9`, which I recommend, but see Appendix A if you want to use a different relay.

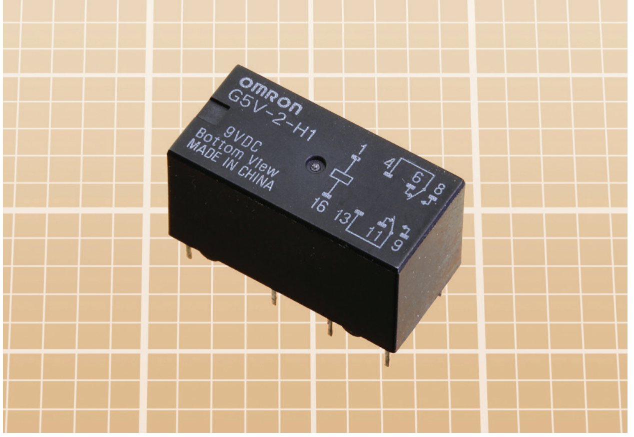


Figure 6-22. The relay recommended for use with this book.

Trimmer Potentiometer

A *potentiometer* has three connections, and the resistance between them varies when you rotate a knob or shaft. A *trimmer potentiometer*, also known as a *trimmer resistor*, is a miniature potentiometer suitable for plugging into a breadboard, and two examples are shown in Figure 6-23, designed for adjustment with a miniature screwdriver. I will be using two different values which you will find listed in the component tables in Appendix A.

The circuits in this book will require trimmers that are no bigger than 0.3" square. Larger trimmers won't fit the available space. Like so many components, trimmers do not have a standardized pin layout; see Appendix A for guidance if you are buying your own.

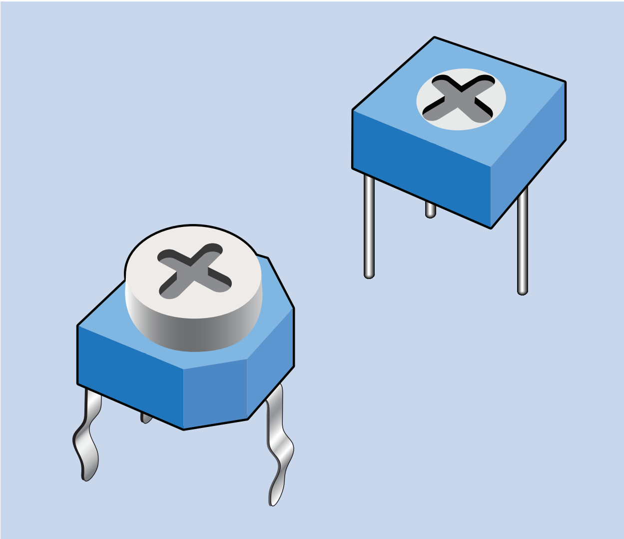


Figure 6-23. Trimmer potentiometers.

Transistor

Only one type of *transistor* is used throughout this book. Its part number is 2N3904, and any manufacturer is acceptable. An example of this transistor is shown below in Figure 6-24.

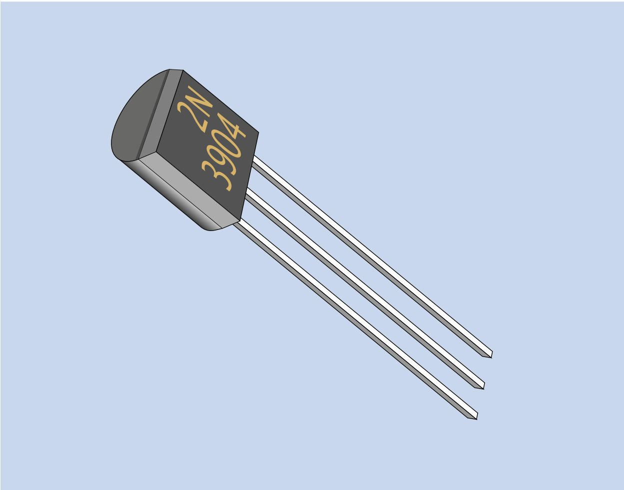


Figure 6-24. A 2N3904 transistor.

In the previous edition of Make: Electronics, I used 2N2222 transistors. They are even more common than the 2N3904 and have a higher power rating, but Motorola decided to make their own version with reversed pin functions, which caused a lot of confusion. This is why I'm not using 2N2222 transistors anymore.

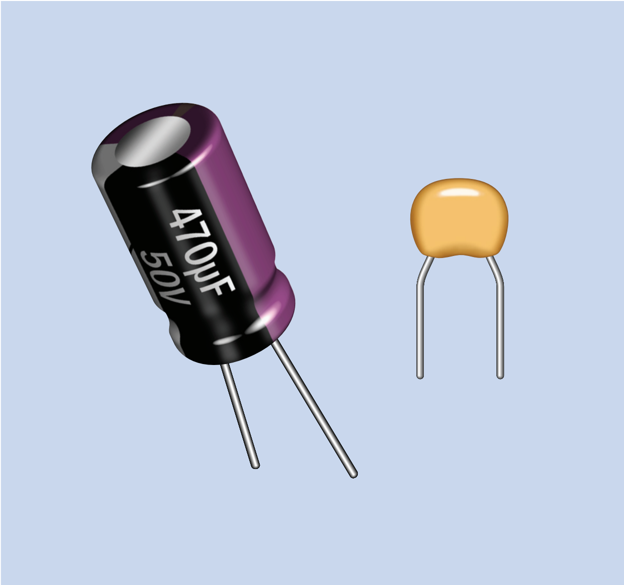


Figure 6-25. An electrolytic capacitor, left, and a ceramic capacitor, right.

Capacitor

Capacitors are not quite as cheap as resistors, but still cheap enough for you to consider buying an assortment instead of individual values. You can find buying information in Appendix A.

For small values, *ceramic* capacitors are recommended. They are encased in a blob of ceramic material. For larger values, *electrolytics* are cheaper. An example of each is shown in Figure 6-25. Their colors vary, but this is unimportant.

You will find much more information about capacitors on page 77.

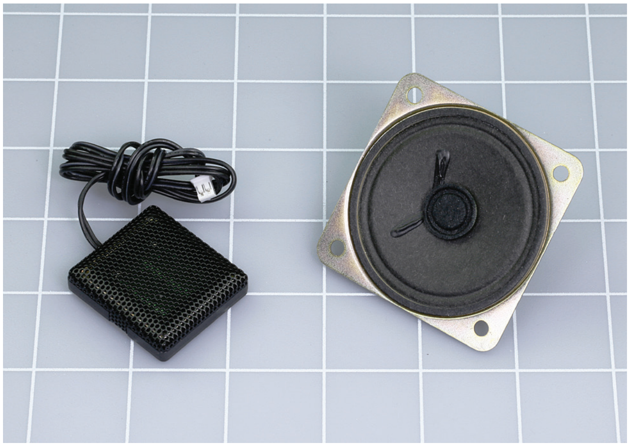


Figure 6-26. Two speakers, one measuring 1.2" diameter, the other measuring 2".

Speaker

The experiments in this book will require a ***loudspeaker*** (more commonly known as a ***speaker***) of at least 1" diameter, although 2" diameter will sound better. A speaker 3" diameter will reproduce bass frequencies more faithfully, but those low frequencies will tend to consume more power.

We will not be dealing with high-fidelity sound, so you don't need to spend much on this component. A couple of samples are shown in Figure 6-26. I suggest you use the cheapest 2" speaker that you can find, so long as it has an impedance of 8 ohms.

Resistors

You will need resistors for almost all of the projects in this book. If you did not already buy an assortment of resistors for Section 1, please check the tables in Appendix A to see what you will need.

And More?

You may be thinking that I have specified quite a lot of tools and components. Rest assured that almost all the parts that I have described will be reusable throughout the book. I'll suggest a few more items at the beginning of each section, and you will also need some integrated circuit chips, but they shouldn't cost very much.

Experiment 6
Getting Connected

The concept of a switch seems simple enough. After all, your home is full of them. You have light switches on your wall, and power switches on appliances—but switches become more interesting when they have multiple positions or when two or more of them are wired together.

You will need:

- Multimeter
- 9V battery (1)
- SPDT slide switch with three terminals spaced at intervals of 0.2" (you can use smaller switches if you can work carefully and precisely.) (2)
- Test leads (1 red, 1 black, 3 other).
- Generic red LED (1).
- Resistor, 470 ohms (1).

Very Simple Switching

Figure 6-27 shows the most primitive type of switch, sold to schools to teach classes in electricity. It is known as a ***knife switch***. I'm not suggesting that you buy one; I'm showing it here because it helps me to explain the features that exist in almost all switches.

The pivot of the lever is connected with the red knob and is called the ***pole*** of the switch. When you turn the lever, this is known as ***throwing*** the switch. You will not be surprised, then, when I tell you that this is a ***single-pole***, ***double-throw*** switch, abbreviated SPDT. (Sometimes this abbreviation is written as 1P2T.)

The metal clips at each end of the switch base are called ***contacts***, and because you can make a connection with either of them, this is an ***ON-ON*** switch. The terminology remains the same even when you are using modern switches.

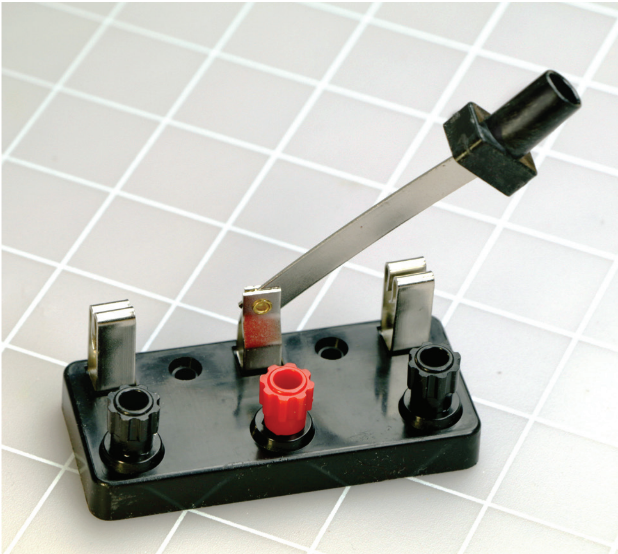


Figure 6-27. A knife switch sold for educational uses. In a modern switch, the parts still have the same names and functions.

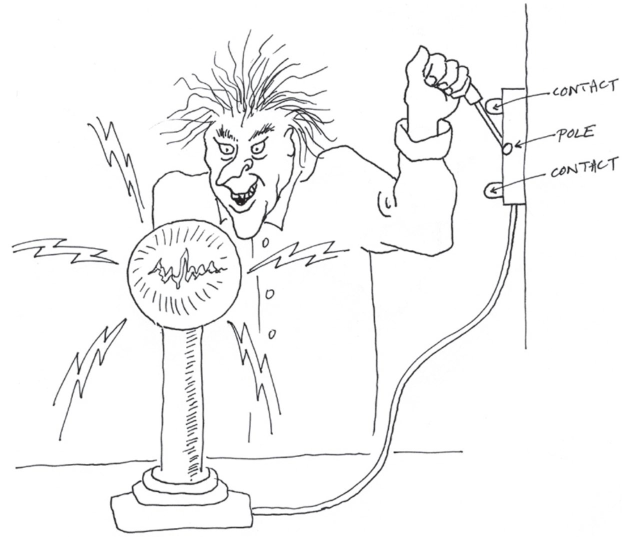


Figure 6-28. This mad scientist has equipped his laboratory with a single-pole, double-throw knife switch.

If you like old horror movies, no doubt you will have seen a mad scientist using an antique version of a knife switch in his basement lair—something like the character in Figure 6-28.

The switches in Figure 6-19 and Figure 6-20 are smaller, cheaper, and more practical. They are known as ***slide switches*** because you work them by sliding a little button on top, which is known as an ***actuator***. It makes a connection inside the switch using a system that often looks like the cutaway drawing in Figure 6-29.

Each of the slide switches that you will need for this experiment has three pins. If you're guessing that these pins function like the three knobs on the knife switch that I showed you in Figure 6-27, maybe you're right. I'll ask you to verify that in a moment.

During the past decades, popular slide switches have gradually diminished in size. This is convenient when you want to squeeze a circuit into a small space, but if you're trying to attach alligator clips to the pins on a switch, it's not so convenient. For this experiment using test leads, ideally you will have a couple of slide switches with pins that are 0.2" apart.

If you ordered a kit to go with this book, you should find two slide switches of this size. If you're shopping for components yourself, you may have trouble finding any that are big enough—but if you are patient, and you have steady hands, you can use switches like the one in Figure 6-20 with pins only 0.1" apart. Just bend the pins out a little either side to make room for your test leads as shown in Figure 6-30 on the next page.

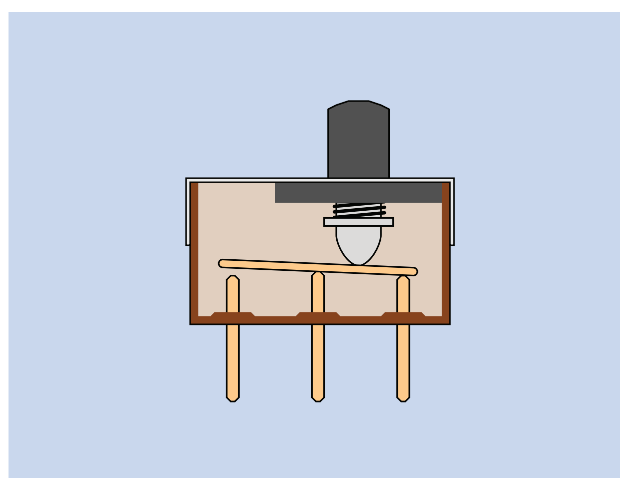


Figure 6-29. The interior workings of a slide switch.

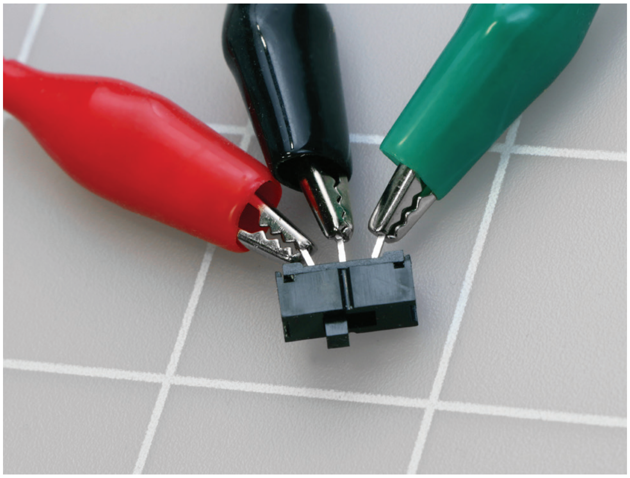


Figure 6-30. Bend the pins of a small switch outward, and you can just manage to apply the alligator clips of your test leads.

Continuity Testing

Set your meter to do continuity testing by turning the dial to the symbol shown in figures 6-31 and 6-32. Touch the two probes together, and you'll hear the meter beep. You are verifying that the circuit is continuous, which is why this is called *continuity testing*. If you are hearing-impaired, you should find that your meter also displays a message on its screen.

Now attach an alligator test lead to each of your meter probes, and clip the other ends of the leads to your slide switch as shown in Figure 6-33. Slide the actuator of the switch one way, then the other, and hear the meter beep and stop beeping. If you have a curious nature, you may be wondering what happens if you test just the two outer pins of the switch, and of course you can check that.

Figure 6-34 shows a *schematic* version of the switch test, where the symbol for the switch is quite similar to the way it actually looks inside.

Now that you've seen how easy this is, I want you to try something more interesting. Put two switches together, wired as in Figure 6-35, and refer to the schematic in Figure 6-36. Notice in the schematic, I have named the switches A and B. I could have chosen more interesting names, but A and B are more often used than names such as "Mike" or "Sheila" when you are dealing with schematics. The two positions of each switch are identified as 1 and 2.

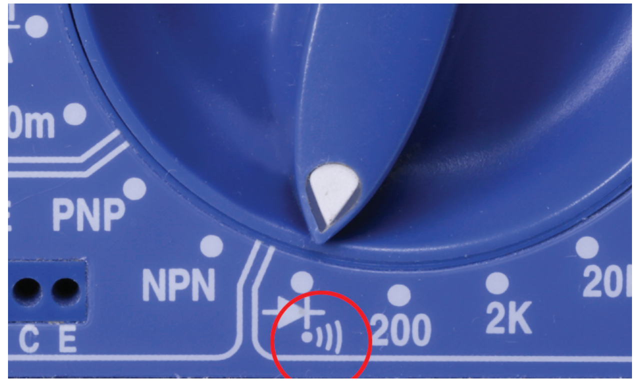


Figure 6-31. The continuity symbol on a meter.

Figure 6-32. The continuity symbol on another meter.

How many combinations of switch positions are there in this test? Switch A has two positions, and for each of them, Switch B has two positions. I'm sure you agree that 2 * 2 = 4, and therefore the total number of combinations is four. You can draw a little table in your notebook, like the one in Figure 6-37 on the page after next, showing which combinations of terminal positions make your meter beep. This table may seem too obvious to be useful, but you'll encounter a lot more tables of this type when I'm dealing with digital logic later in the book, so I want to introduce the concept now.

The switches in Figure 6-36 are in *series*, meaning that electricity has to pass through one before it gets to the other, and there's only one combination of switch positions that beeps. I colored it red in the table.

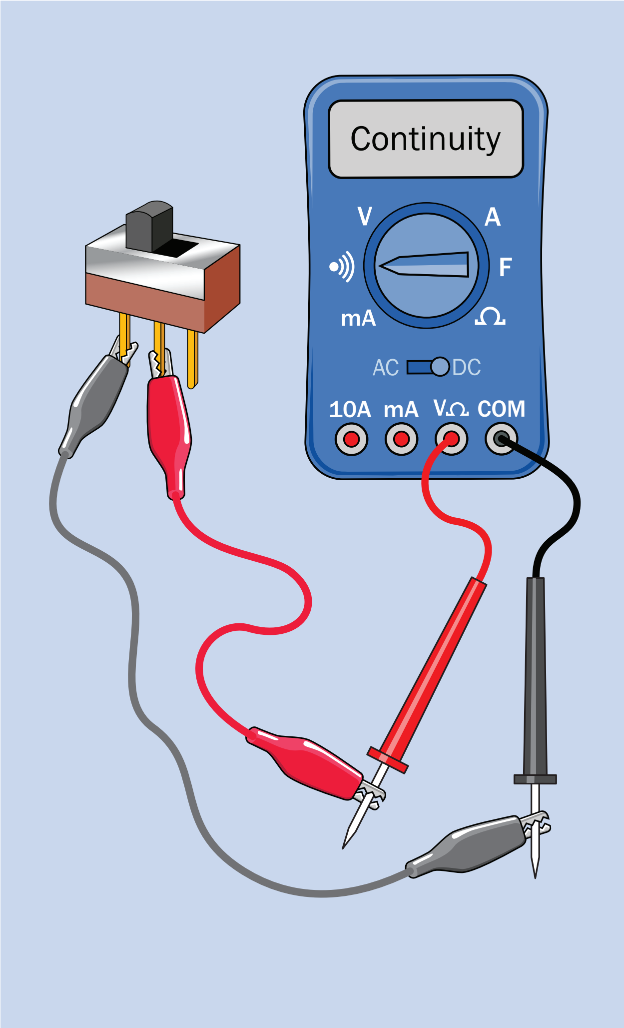


Figure 6-33. Investigating the slide switch.

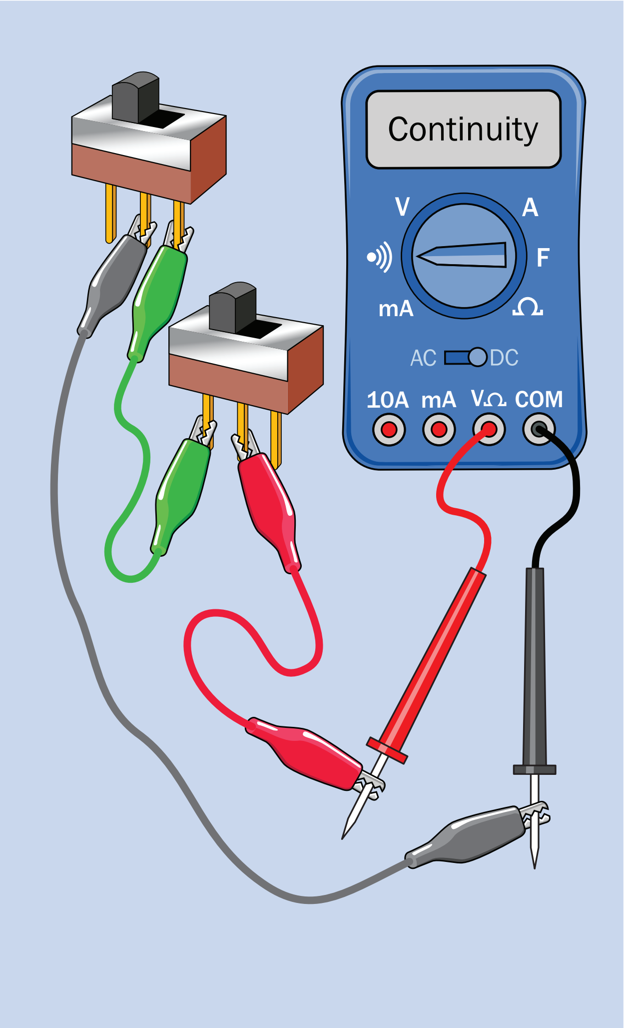


Figure 6-35. Adding a switch in series.

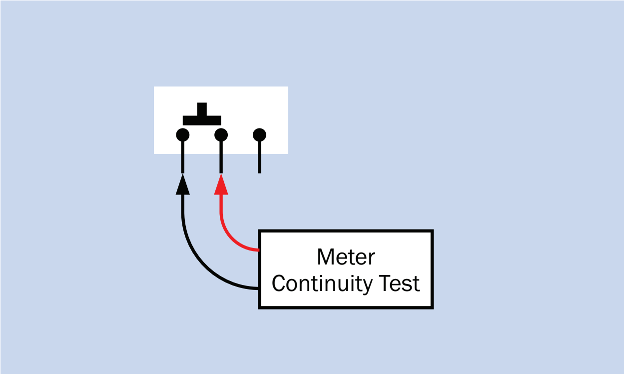


Figure 6-34. The schematic version of Figure 6-33.

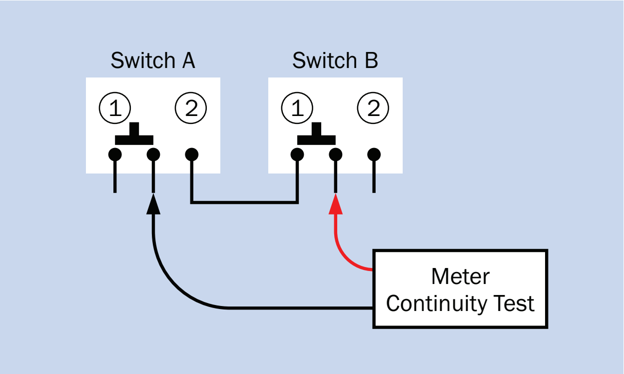


Figure 6-36. The schematic version of Figure 6-35.

Switches Wired in Series		
Switch A	**Switch B**	**Continuity**
Position 1	Position 1	●
Position 1	Position 2	●
Position 2	Position 1	● (red)
Position 2	Position 2	●

Figure 6-37. A table of switch combinations and continuity.

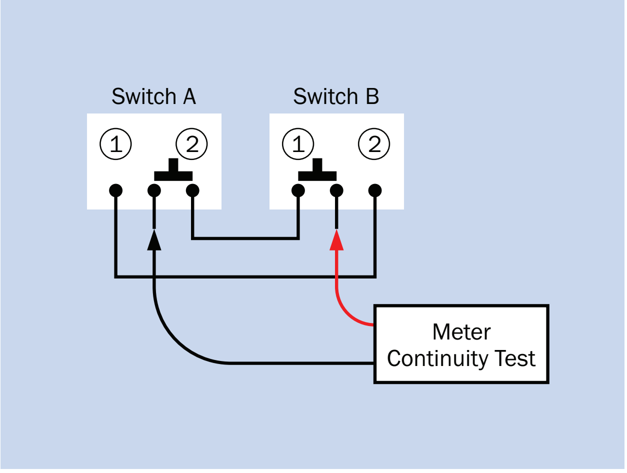


Figure 6-38. A series-parallel circuit.

Switches Wired in Series-Parallel		
Switch A	**Switch B**	**Continuity**
Position 1	Position 1	●
Position 1	Position 2	● (red)
Position 2	Position 1	● (red)
Position 2	Position 2	●

Figure 6-39. Continuity table for the circuit in Figure 6-38.

Now add another test lead, so the circuit looks like Figure 6-38. I think you can figure out how to add the lead, without me showing another picture-diagram.

I'll call this a "series parallel" circuit, since there are two series connections now, but they are connected with wires that are parallel. If you draw a new table, as in Figure 6-39, you'll find that A-1 and B-2 make the meter beep, *or* A-2 and B-1, while the other combinations do nothing.

Interestingly, if the meter is not beeping, you can move one actuator of either switch to make it beep. If the meter is beeping, you can move one actuator of either switch to make it stop. This has an application for light switches in your home, which I'll explain below. Maybe you can guess what I'm talking about.

Three-Way Switching

So far, I have used a schematic symbol that applies specifically to a slider switch. Another symbol exists that can represent almost any type of switch. I'll call it the universal symbol. (How many types of switch are there? Hundreds! Just do an online image search for

```
types of switch
```

to see what I mean.)

The universal switch symbol is more common than the slide-switch symbol. I have used it in Figure 6-40, to show the same circuit as in Figure 6-38. The way the circuit functions is still the same, because the connections are still the same. The two wires running horizontally are in parallel, but current from the meter passes through the switches in series.

If your home contains a flight of stairs, you're very likely to find at least one light wired this way. You use a switch at the bottom of the stairs to turn the light on if it's currently off, or off if it's currently on—and you can use the switch at the top of the stairs in the same way. Figure 6-41 shows what I mean. An electrician would say that you have *three-way switching*, because each switch has three terminals. In reality, you have seen that there are actually four combinations of the switch positions, but when I tried to discuss this issue with an electrician, he wasn't very interested.

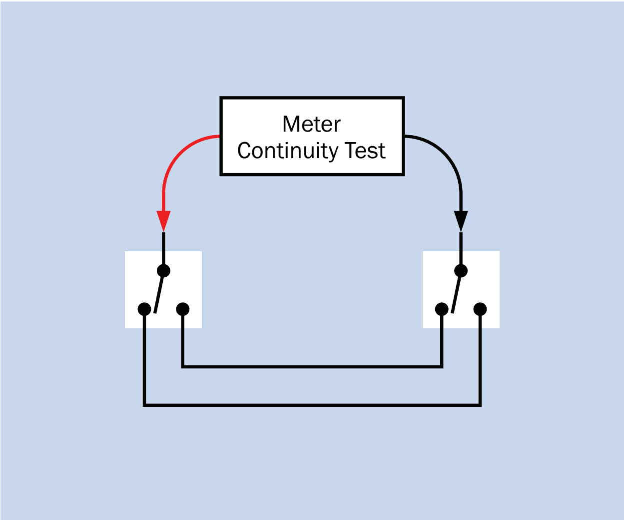


Figure 6-40. The universal symbol for a single-pole, double-throw switch is used in this version of the circuit that appeared previously in Figure 6-38.

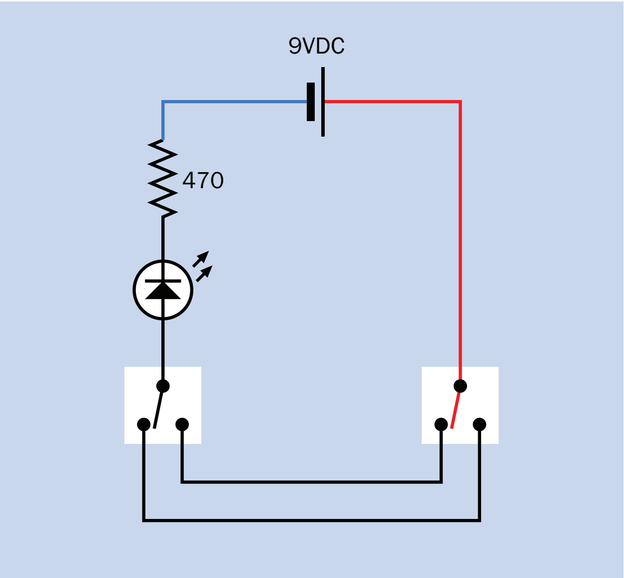


Figure 6-42. A low-voltage demo of three-way switching.

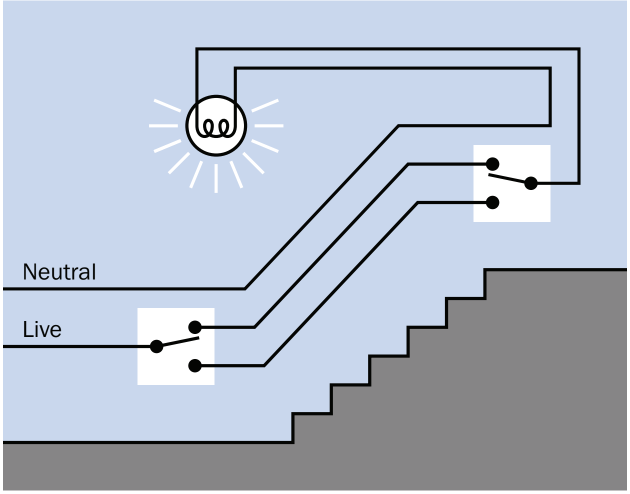


Figure 6-41. So-called three-way switching in house wiring.

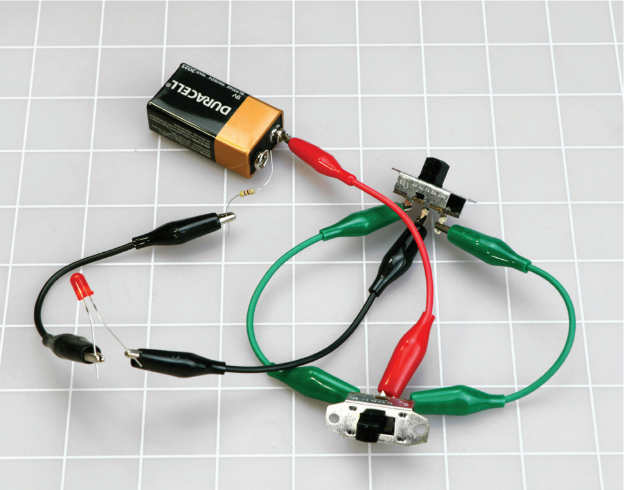


Figure 6-43. The circuit in Figure 6-42 built from actual components.

If you want to see a miniature demo of three-way house wiring, you can modify the circuit in Figure 6-40 by removing the meter and substituting an LED with a load resistor and a battery, as shown in figure 6-42.

And just to show you that I do actually build the circuits that I write about, my tabletop version of the circuit is shown in Figure 6-43.

More About Switches

Sometimes you only need a ***single-pole***, ***single-throw*** switch (which may be abbreviated as SPST or 1P1T). An on-off switch is a common example: Only one position makes a connection. The schematic symbol is shown in

Figure 6-44 in a variety of styles, all of which mean the same thing. In this book, I've shown a white rectangle around each switch for clarity.

A similar variety of styles can represent double-throw switches.

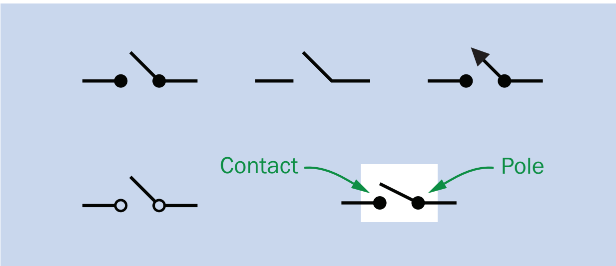


Figure 6-44. Various styles for representing a SPST switch in a schematic.

Some switches have two entirely separate poles operated by one actuator, so you can make two separate connections simultaneously. These are called ***double-pole*** switches, abbreviated DP (or, sometimes, 2P). A double-pole switch may be single-throw or double-throw, depending on what you want to use it for. Two symbols for a DPDT switch are shown in Figure 6-45. The dashed line in each symbol tells you that when you throw the switch, both contacts are made simultaneously, even though they have no electrical connection.

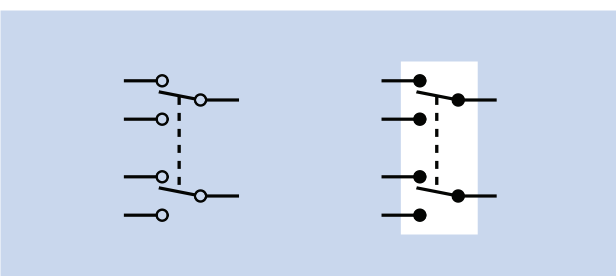


Figure 6-45. Symbols that can represent a DPDT switch.

Why would you need two separate poles? Well, suppose you have a stereo system and you want to switch its output from two speakers in the living room, to two speakers in the dining room. You want just one switch to do this, but the two audio channels have to remain separate, so you use a DPDT switch, with the output from your stereo attached to its pole contacts.

The table in Figure 6-46 summarizes the concepts that I have introduced so far. Switches are available with one, two, three, or more poles, and each switch can make just one contact (single-throw) or a choice of two contacts (double-throw).

Maybe you didn't think there would be much to learn about switches, but there's a lot more. Some switches are spring-loaded, so they snap back to a default position when you release pressure on them. These are called ***momentary switches***, and a simple pushbutton is an example. Maybe you are assuming that when you press the button, a pair of contacts will close inside the switch—and this is often true, in which case the contacts are ***normally open***, abbreviated NO. But some spring-loaded switches open the contacts when you press the button, and these are referred to as ***normally closed***, abbreviated NC.

	Single Pole	Double Pole	Three Pole
Single Throw	SPST or 1P1T	DPST or 2P1T	3PST or 3P1T
Double Throw	SPDT or 1P2T	DPDT or 2P2T	3PDT or 3P2T

Figure 6-46. Types of switches.

A single-pole, single-throw momentary switch with NO contacts can also be described as (ON)-OFF, and the parentheses tell you which state is the one where you have to hold down the button. If it has NC contacts, it's an ON-(OFF) switch.

You can find double-throw momentary switches. In other words, they switch between two "on" positions, but one of them is spring-loaded. This would be an ON-(ON) SPDT momentary switch.

Finally there are double-throw switches which have an actuator that has an additional center position, which is off. These may be spring-loaded, too.

You are not going to need any unusual switches for the experiments in this book, but you may run into them later, so I am summarizing them here for reference. In Figure 6-47, the term "alternate action" means a switch

	Alternate Action	Momentary
Single Throw	ON–OFF	Normally Closed ON–(OFF)
		Normally Open (ON)–OFF
Double Throw	No Center Position ON–ON	No Center Position ON–(ON)
	Center Off ON–OFF–ON	Center Off (ON)–OFF–(ON)

Figure 6-47. Switch configurations.

which is not spring-loaded and will remain in each alternate position. "Momentary switch" means that you have to push it and hold it to maintain that position.

You do need to remember that one type of momentary switch has its own symbol: The pushbutton. Three symbols to illustrate it are shown in Figure 6-48. I will be using the style on the right to represent any type of pushbutton, including a tactile switch.

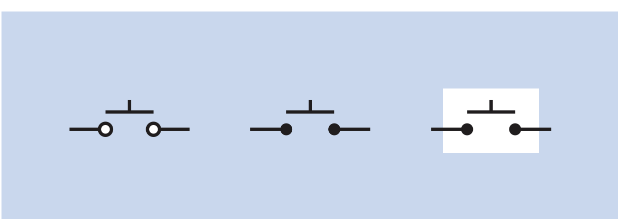


Figure 6-48. Three ways to portray a pushbutton, which is the simplest type of momentary switch.

There are even more types of switches, which I haven't mentioned—such as rotary switches, which can have five or even ten positions, and multiple poles. I won't be including them in this book, because they are not so commonly used anymore.

Sparking

When you make an electrical connection, it tends to create a spark as current jumps between the contacts an instant before they close. When you break a connection, another spark tends to occur, and this is bad for the switch. Sparking erodes contacts until they don't make a reliable connection anymore.

This is less of a problem when you are drawing a small amount of current at a low voltage, as in the circuits in this book. However, if you are switching a motor on and off, you have to be careful to use a large enough switch, because a motor sucks an initial surge that is at least double the amount that it uses when it settles down and runs constantly. Therefore, if you have a 2A motor, you should probably use a switch rated for 4A to turn it on and off.

Even in house wiring, ratings can be important. I have very powerful overhead lighting in my workshop, drawing about 8A, and the lights take an initial surge which is higher. The electrician who wired the place didn't realize that I would be using such powerful lights, so he provided me with just a typical, generic light switch. When I saw that, I thought to myself, "I wonder how long it will last." About three years later, the contacts finally burned out as a result of sparking, so I replaced it with a switch rated for 15A.

You may feel tempted to use a switch that is rated for less than the application which you have in mind, because—hey, it works well enough! Eventually, though, it is likely to fail.

Early Switching Systems

Switches seem to be such a fundamental feature of our world, and their concept is so simple, we can easily forget that they went through a gradual process of evolution. Primitive knife switches were adequate for pioneers of electricity who simply wanted to connect and disconnect some apparatus in a laboratory, but a more sophisticated approach was needed when telephone systems began to proliferate. Typically, an operator at a "switchboard" needed to connect any pair from among hundreds of lines on a board. How could it be done?

In 1878, Charles E. Scribner (shown in Figure 6-49) developed the "jack-knife switch," so called because the part of it that the operator held looked like the handle of a jack knife. Sticking out of it was a plug 1/4" in diameter, and when the plug was pushed into a socket, it made contact inside the socket. The socket, in fact, contained the switch contacts.

Audio connectors on guitars and amplifiers still use exactly the same system. When we speak of them as being "jacks," the term dates back to Scribner's invention.

Figure 6-49. Charles E. Scribner invented the "jack-knife switch" to satisfy the switching needs of telephone systems in the late 1800s.

Alternate Symbols

I showed you a variety of symbols that are used to represent a single-throw switch. Now I'll show some alternatives for symbols that you have already encountered, such as a battery. You need to see the options in case you run into one in a schematic somewhere.

Figure 6-50 shows various ways to indicate power in a circuit. Along the top row are three options for representing a battery. Traditionally, a single pair of lines represented a single 1.5-volt cell, while two pairs indicated a pair of cells adding up to 3V, and so on. But when circuits used higher voltage, the person drawing the schematic would usually show a dashed line between cells instead of drawing dozens of them in a row. Most people probably don't bother to show multiple cells anymore, but if you happen to see them, this is what the symbols mean.

How do you know the total voltage of a battery, when you see a dashed line? You don't, but the person who drew the schematic probably showed the voltage numerically, too.

Battery symbols are not used so often anymore. Instead, you are likely to see an abbreviation such as Vcc, V+, +V, or V with a number added. This indicates where positive power is applied in a circuit. Originally the term Vc referred to the voltage at the collector of a transistor, while Vcc meant the supply voltage to all the collectors—but Vcc is now used even if a circuit has no transistors in it. Many people say "vee cee cee" without knowing where it came from.

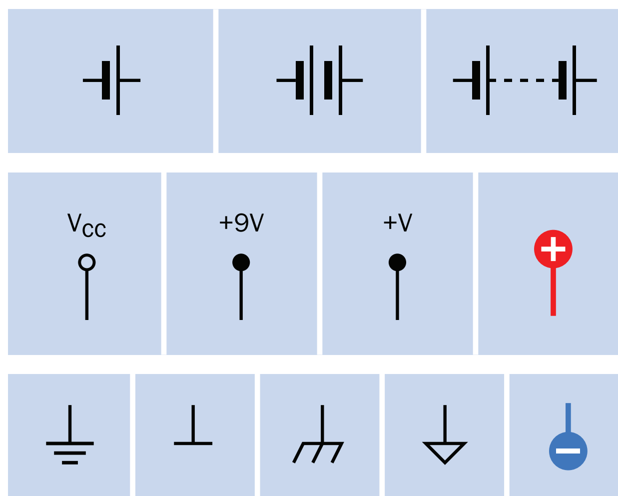


Figure 6-50. Different ways to represent power in a schematic.

In this book, I have used a plus symbol in a red circle to indicate positive power input, so there should be no misunderstandings.

The negative side of the power supply can be shown with any of the symbols in the bottom row of Figure 6-50, and may be referred to as ***negative ground*** or simply ***ground***. Because many parts of a circuit may be grounded, multiple ground symbols may be found scattered around a schematic. When you actually build the circuit, you'll need to connect all the ground points together somehow. You can use wire, or connections inside a breadboard—or maybe you can learn how to obtain your own printed circuit boards, with copper traces on them.

I have chosen to use the minus sign in a blue circle because it is so intuitively clear. You'll find the other negative-ground symbols in schematics drawn by other people. They all mean the same thing, in a circuit with DC power: Zero volts relative to the positive power supply.

In a gadget that uses ***alternating current*** from a wall outlet, the situation is more complicated, because the outlet has three sockets in it for ***live, neutral***, and ***ground*** connections.

A schematic using current from a wall outlet typically shows the AC source as an S shape turned on its side, as in Figure 6-51. Often the value of the power supply is shown, and in the US it is usually 110, 115, or 120 volts. Elsewhere in the circuit, the symbols shown on the right-hand side of Figure 6-51 refer to the metal chassis of the device in which the electronics are mounted. The chassis is then connected to the ground pin of a power outlet, if a three-conductor power cord is used.

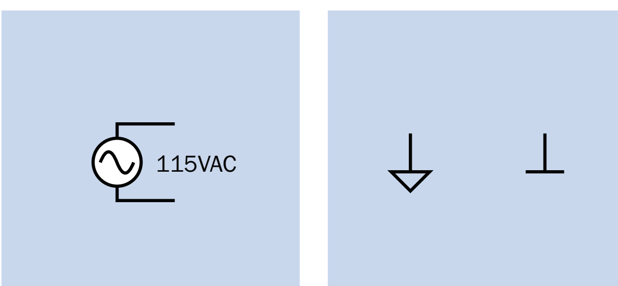


Figure 6-51. Power source and ground in an AC circuit.

Note that the ground pin in an AC power outlet in the home really does connect with the ground outside the building. When I was arranging for power to be brought to some land in a wilderness area, the utility company required that I should hammer an 8-foot copper-plated stake into the dirt, to ground the house that I was building. Planet Earth has a huge ability to store electrons.

In the UK, a grounded device is sometimes referred to as being ***earthed***.

Now let me backtrack to symbols for standard through-hole LEDs. They can be represented in many styles which all mean the same thing. I've included a selection in Figure 6-52. The big arrow-head inside the symbol indicates that conventional current has to flow through it in this direction, while the little arrows indicate that the component emits light. Why is there more than one version of this symbol? It's a mystery. Personally, I continue to use a white outline to distinguish the component from the rest of the circuit.

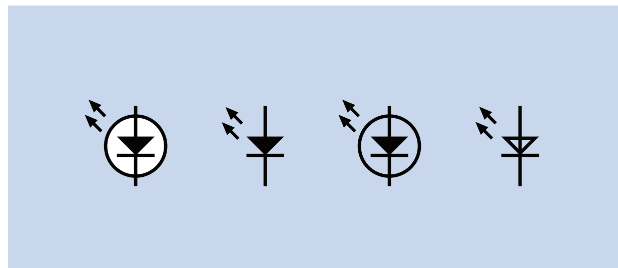


Figure 6-52. Four ways to show an LED in a schematic.

Note that an LED symbol may be found oriented in any direction, depending on what was convenient for the person drawing the schematic, and the pair of arrows may be angled to the left or to the right. It all means the same thing.

Even the humble resistor can be portrayed in a choice of styles. In Figure 6-53 I have shown the zig-zag symbol that is popular in the United States, but the style on the right is used in Europe. These two resistors happen to have a value of 4.7K, but you could see any other value. Remember I mentioned that in European notation, letter R, K, or M is placed instead of a decimal point to tell you that a resistance value is in ohms, kilohms, or megohms.

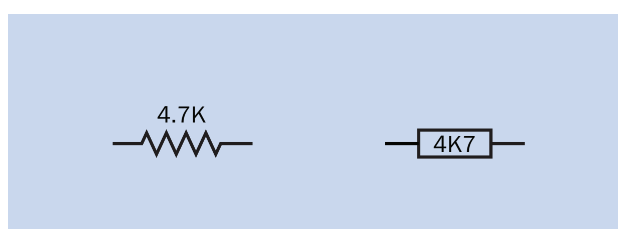


Figure 6-53. A 4.7K resistor shown in American style (left) and European style (right).

Schematic Layout

Many people like to go online and use circuit-drawing software when they want to create a schematic. You can find sites which enable this if you use a search term such as:

```
online circuit simulator
```

This seems easy, because you just drag components around and stretch lines between them to indicate connections, and the simulator then shows you what will happen when the circuit is powered up.

Figure 6-54 shows a typical schematic of this kind. It includes some symbols that you have not encountered yet, but still you can see the positive power supply at the top and the ground symbol at the bottom (labeled **GND**), and you can recognize the LEDs (without circles around them) and resistors (in European format, although for some reason the decimal values are shown in US format).

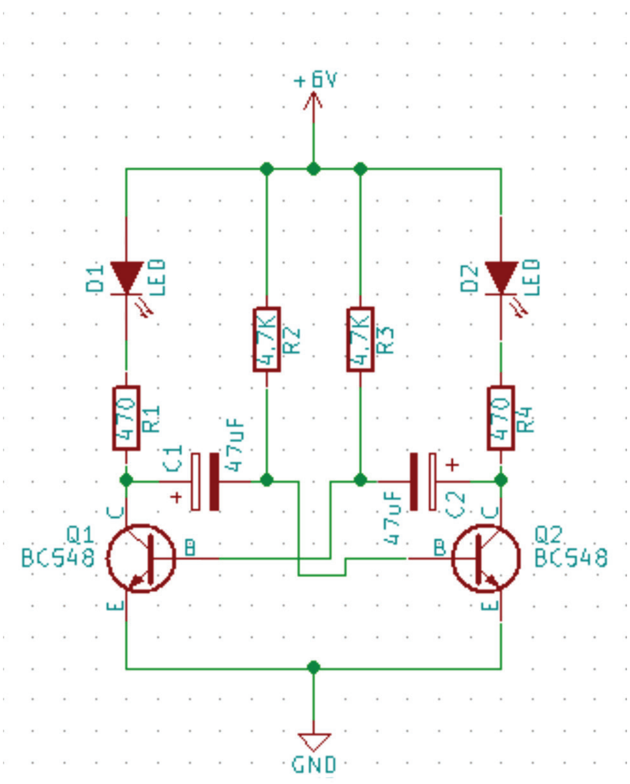


Figure 6-54. A schematic generated by circuit-drawing software.

Each component is identified with a code such as D1 or R1 so that you can refer to them easily if you write something to explain the circuit. The LEDs are shown with light-emitting arrows pointing downward instead of upward, which I find odd, but that's just the way the circuit-drawing software happened to position them. Adding it up, all the information is there, although the circuit isn't very nice to look at.

People often put the positive power supply at the top of a schematic and negative ground at the bottom, because you can imagine conventional current running down through the components like water. The disadvantage of this layout is that if you want to build the circuit on a breadboard, current will move horizontally between the vertical buses. Electronics books usually assume that you will perform some kind of mental trick to convert a schematic layout to a breadboard layout, but this isn't very easy, and can lead to errors. Therefore, most of the schematics in this book are organized to look as much as possible like a breadboard.

Crossovers

As circuits become more complicated, we often have to show wires that pass over each other without making an electrical connection. The top half of Figure 6-55 illustrates three ways to do this. I'm including a couple of obsolete styles, in case you run across them.

"Very old style" has one big advantage: No one could misunderstand it. However, circuit-drawing software doesn't support this style anymore.

The "Old style" was used during the 1960s, but it was confusing, and it, too, has become generally obsolete, although you may see it if you are browsing through old books.

The third style is now almost universal, and I've used it throughout this edition. The rule is very simple:

- In a schematic, no dot means no connection.

Conversely, if there is a dot, there is always a connection, as shown in the lower section of Figure 6-55. Sometimes you will see schematics where small dots are used, requiring you to examine them very carefully—and if someone scans or photocopies the circuit, the dots may tend to disappear. Personally, I always use big dots.

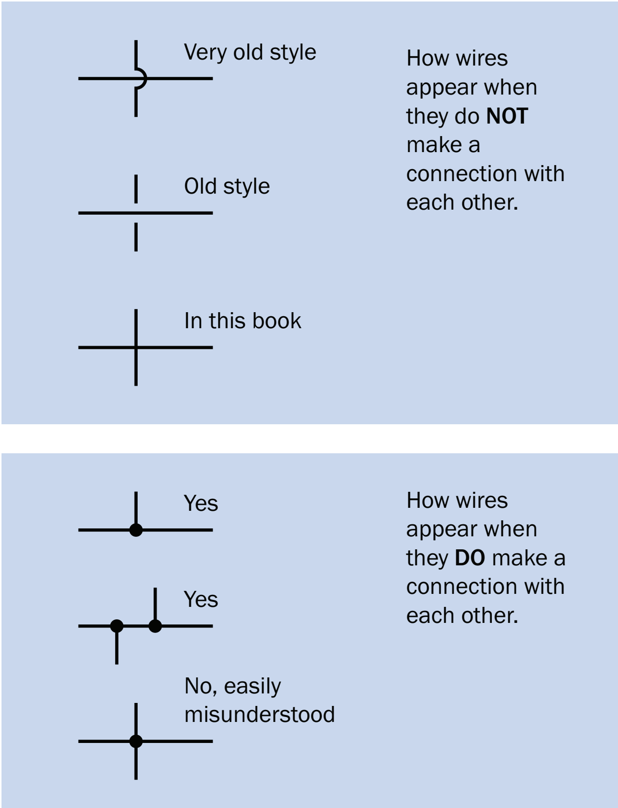


Figure 6-55. Various styles for depicting wires that do or do not connect.

One other point I must add: The "crossroads" connection at the bottom of Figure 6-55, where four wires are connected, is quite commonly used but easily misunderstood, as it looks like a crossover without a connection, especially if the dot is poorly printed. The layout immediately above it is a much better alternative.

Color-Coded Connections

Because I never want you to get confused between the positive and negative sides of a power supply, I'm going to be coloring all the positive conductors red in schematics, while the negative/ground side will be blue. You may have noticed that I already used this style in Figure 6-42.

I realize that some people are color-blind, but red-blue color blindness seems to be relatively rare.

Experiment 7
Investigating a Relay

The next step in your exploration of switching is to use a remote-controlled switch which turns itself on when you send it a signal. This kind of switch is known as a ***relay***, because it relays an instruction from one part of a circuit to another. It is an ***electromechanical*** device, because it uses and switches electricity but contains mechanical components including contacts and a lever. Transistors have replaced relays in many applications, but not all. Automobiles, for instance, still contain relays, and I think you will probably find at least one inside a dishwasher, a refrigerator, or an air conditioner.

You Will Need:

- 9V battery (1).
- DPDT 9VDC relay (1).
- Optional: Additional relay (1).
- Tactile switch (1).
- Test leads, (1 red, 1 black, 1 other color).
- Utility knife (1).
- Multimeter (1).

The Relay

The type of relay that I want you to use has two pins at one end and six at the other. The six are clustered in two lines of three, like the six spots on dice, as shown on the next page in Figure 7-1 where the relay is upside-down with its pins in the air. For more details about the particular relay that is suitable for this experiment, see Appendix A.

Some old-school, heavy-duty relays are packaged in transparent plastic cases, so that you can see the mechanism inside. Unfortunately most relays do not allow you this luxury, but you can cut one open for investigational purposes. If you do this very, very carefully, it should still be usable afterward. If not—well, you paid a small sum for Learning by Discovery.

Caution: Polarity Problems

The power that you apply to a relay, to make it work, is called the *operating current*. This power is applied to the pair of pins at one end, which are connected with a *coil* inside the relay.

In many relays, the coil has no polarity—the positive side of the power supply can be connected with either pin. In others, polarity is important. The type of relay that I suggest for this experiment isn't fussy about polarity, but if you use some other type of relay, always check its datasheet to make sure.

Making It Beep

The first step is to make sure that your relay is working. In Figure 7-1, a relay is wired so that a pushbutton (properly known as a tactile switch) connects battery power to the coil inside a relay. Your meter, set to detect continuity, should beep when the switch inside the relay connects the two pins being touched by the meter probes. You should also hear a faint "click" when the relay responds—or if you're hard of hearing, you may be able to feel it give a tiny jump when the switch moves inside it.

Now try moving the black test lead to the vacant pin that is one step closer to you. The behavior of the meter is reversed, so that it beeps when you don't press the button, and then stops beeping when you do press it. If you are thinking that maybe the relay has a double-throw switch inside it, you're exactly right. The power from the battery flips the switch.

Why is this useful? Because a relay can be triggered by a small voltage and a small current, but can switch a larger voltage or higher current. When you start your car, for instance, a relatively small, cheap ignition switch (or a sensor controlled by a handheld remote) sends a small signal down a thin, inexpensive piece of wire to a relay that is near the starter motor. The relay activates the motor through a much thicker, more expensive piece of wire, capable of carrying as much as 100 amps. Similarly, inside a washing machine, somewhere there is a timer which sends a signal to a relay that connects power to a motor which turns a drum full of wet clothes.

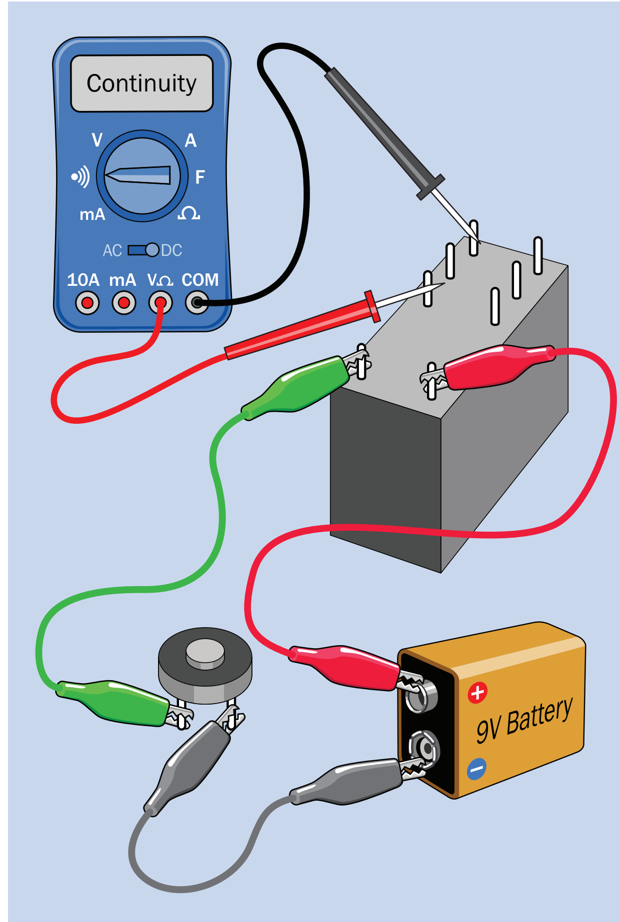


Figure 7-1. The first step in figuring out what's happening inside a relay.

What's Going on Inside

Figure 7-2 shows an x-ray view of the interior of the relay before you pressed the button.

In Figure 7-3, the button has been pressed and the magnetic field from the coil closes the switches inside. Notice that this happens to be a DPDT relay—it has two poles, although we're only using the one on the left.

You may be wondering why the coil in the relay seems to push the internal switch away from it. The reason is that there is a lever inside the relay which converts a pulling force to a pushing force. You'll be able to inspect this when I get to the point of opening up the relay, later in this experiment.

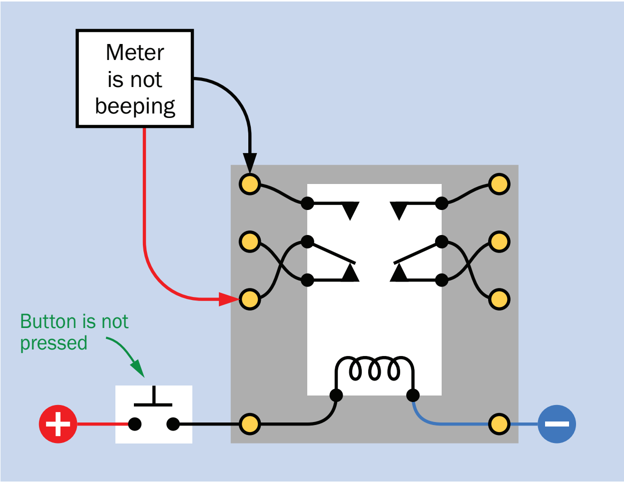


Figure 7-2. Inside the relay.

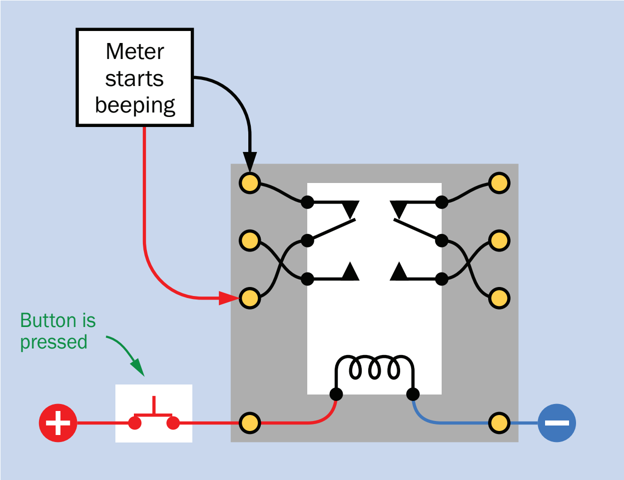


Figure 7-3. The coil moves the switches inside the relay.

Other Relays

I believe the pin functions that I have described are the most common for this size of relay. However, some relays do things differently. When you encounter a relay for the first time, you can find out how it works by checking the datasheet, or by testing different pairs of pins with your meter while you apply voltage to the coil. Using a process of elimination, you can figure out how the pins are connected. Almost always, one pair of pins will be separated from all the other pins, and this pair will activate the coil.

If you look at a datasheet, it should contain diagrams such as the one in Figure 7-4, which was provided by a manufacturer of the relay that you have been using. The style of this diagram is different from the style that I used in Figure 7-3, but you can see that the connections are the same.

Here are some useful facts about relays:

- Some relays are ***latching***, meaning that the internal switches remain in either position when the power is off. This type is less common, but it does have an advantage: You don't have to apply power to it continuously, to keep it "on." You send it a quick pulse to flip it into one state, and then a different pulse to flip it back again.
- Some relays have two poles, some have only one; some are double-throw, and some are single-throw.
- Some coils use AC operating current instead of DC.

Figure 7-5 shows a selection of schematic symbols for various types of relays. Type A is single-pole, single-

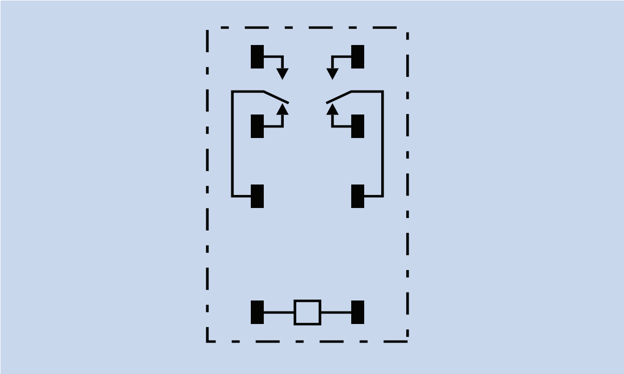


Figure 7-4. Pin functions shown on a relay datasheet.

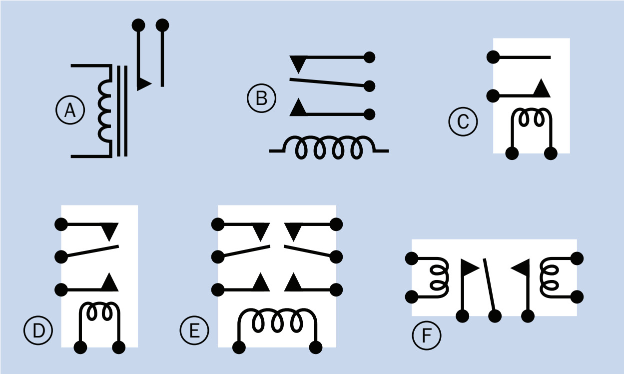


Figure 7-5. Schematic symbols for various styles of relays.

throw. Type B is single-pole, double-throw. Type C is single-pole, single-throw, drawn in the style that I like to use, with a white rectangle reminding you that the parts are enclosed in a single component. Type D is single-pole, double-throw. Type E is double-pole, double throw. Type F is single-pole, double-throw, latching.

Most relay symbols ask you to imagine that voltage in the coil will pull the pole contacts toward it. Unfortunately, as you've seen, many actual relays create the impression that they're doing the opposite. Always check a datasheet to figure out what's really happening.

Relay schematics are always drawn with the internal switch in its ***relaxed*** position, when power is not applied—with the exception of the latching relay, where the position of the switch is arbitrary.

The type of relay you have been testing is a ***small-signal relay***, meaning that it can't switch a lot of current. The datasheet will tell you its limits. Larger relays may be capable of switching many amperes, and it's important to choose a relay with contacts that are rated for the maximum current in your circuit, in just the same way that you have to use any switch rated for the correct power.

In future experiments you'll discover some practical uses for a relay—for example, in an electronic combination lock. Before you get to that, I'm going to show you how to turn a relay into an oscillator. And before ***that***, I think we should take a look inside.

OpeningIt Up

If you're an impatient sort of person, you can open your relay using methods such as those in Figure 7-6 or Figure 7-7. Generally, though, you may be better off using a most mundane piece of equipment: A box cutter or utility knife.

Figures 7-8 and 7-9 illustrate the technique that I like to use. You apply a utility knife to the edges of the plastic shell, beveling them until you see just a hair-thin opening. Don't go any farther; the parts inside are very, very close to your knife blade. Now pop the top off. Repeat this procedure with the remaining edges of the shell, and if you were really careful, the relay will be exposed but will still work when you energize its coil.

Figure 7-6. Option 1 for opening a relay (probably not recommended).

Figure 7-7. Option 2 for opening a relay (definitely not recommended).

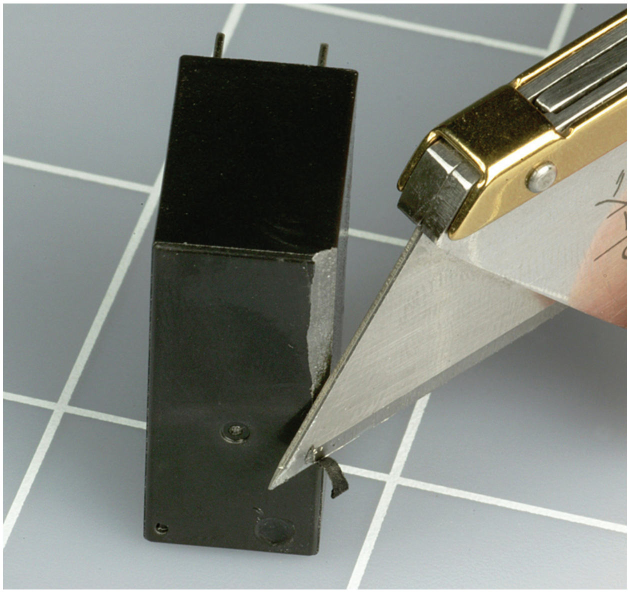


Figure 7-8. Shaving the edges of the plastic box of a relay is a first step to opening it. Always cut away from you and downward toward your work bench.

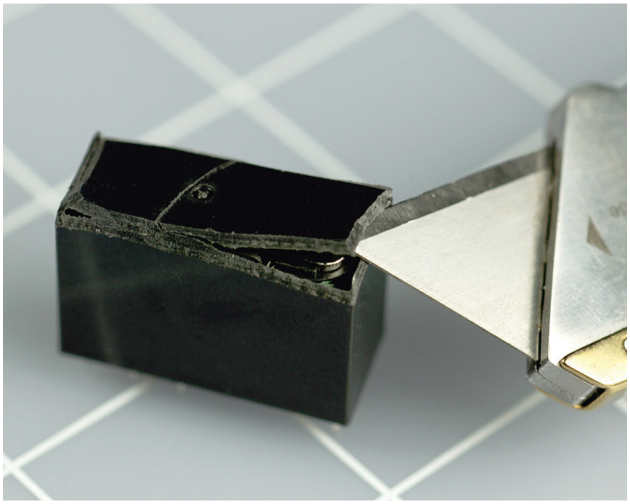


Figure 7-9. After shaving the edges, you should be able to pry open one section of the case.

This procedure is safest if you can hold the relay with a clamp or a vise while you work on it. Always keep your fingers as far away as possible from the cutting edge of the knife blade, and always cut downward.

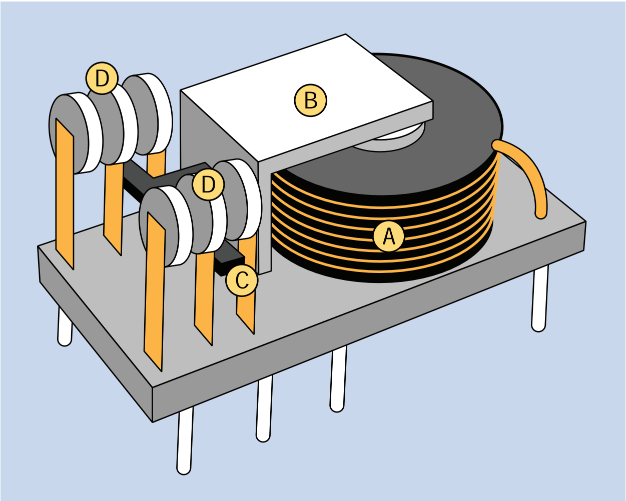


Figure 7-10. Parts of a relay. See text for details.

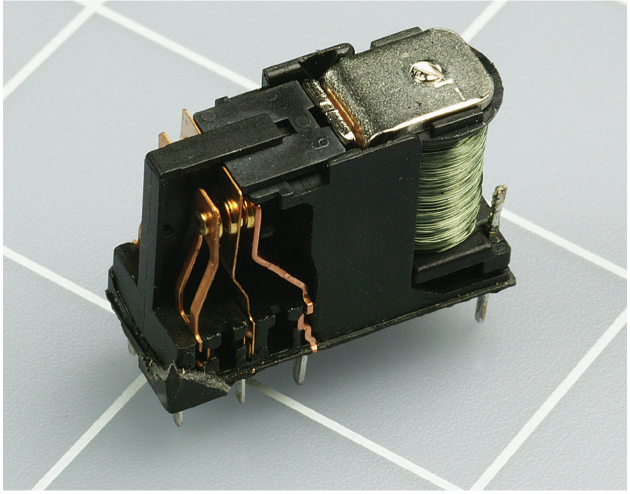


Figure 7-11. Inside a small signal relay.

What's Inside?

Figure 7-10 shows a simplified view of the parts in a typical relay. The coil, A, attracts a lever, B. A plastic extension, C, pushes outward against flexible metal strips and moves the poles of the relay, D, between the contacts.

You can compare the diagram with an actual relay that I opened up, in Figure 7-11. It is standing on a mat divided into one-inch squares.

Various sizes of relays are shown with their cases removed in Figure 7-12. All of them happen to be designed for 12 volts DC. The automotive relay at far left is the simplest and easiest to understand, because the people who designed it didn't have to worry much about its size. There's quite a lot of room inside the body of a car for relays that measure an inch along each edge.

Smaller relays are more ingeniously designed, more complex, and more difficult to figure out. Usually, but not always, a smaller relay is designed to switch less current than a larger one.

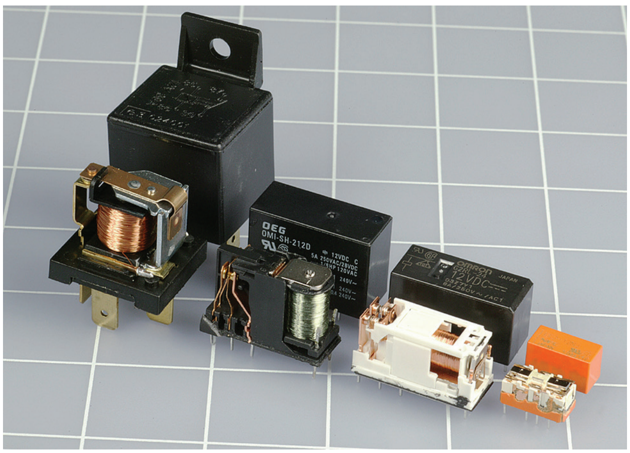


Figure 7-12. Various 12VDC relays.

Relay Terminology

Coil voltage is the ideal voltage that the relay likes to receive when you energize it.

Set voltage is the less-than-ideal minimum which the coil requires, to close the switch. In practice, a relay will probably work with even less than the set voltage, but its function won't be guaranteed.

Operating current is the power consumption of the coil, usually in milliamps, when the relay is energized. Sometimes the power is expressed in milliwatts.

Switching capacity is the maximum amount of current that the contacts inside the relay can switch without damage. Usually this is for a resistive load, meaning a passive device such as an old-school incandescent light bulb. Remember that a load such as a motor takes an initial surge of current, often twice as much as the current it consumes after it is running.

Experiment 8
A Relay Oscillator

When you used test leads with alligator clips in previous experiments, they had two big advantages: you could assemble a circuit quickly, and you could see the connections easily.

Sooner or later, though, you have to get acquainted with the most widely used prototyping device: A ***solderless breadboard***, such as the one back in Figure 6-10.

Long ago, in the 1940s, circuits were prototyped on a wooden base that really did look as if it could be used for slicing bread. Wires and components were nailed, stapled, or screwed into it, because this was easier than the alternative, which was mounting them on pieces of sheet metal. Remember, plastic barely existed back then. (A world without plastic—can you imagine it?)

Today a solderless breadboard is a little slab measuring about 2" by 7", and no more than 0.5" thick. This is a much quicker way to build a circuit than by nailing it to a piece of wood, but it does have a drawback: It creates internal connections between the components that are impossible to see and difficult to visualize. I will try to help you to deal with that.

First I think you should dive right in and assemble a circuit, taking the previous experiment with a relay one step further.

You Will Need:

- 9V battery (1).
- Battery connector (optional) (1).
- Breadboard (1).
- DPDT 9VDC relay (1)
- Generic red LED (2).
- Tactile switch (2).
- Resistors, 100 ohms (1), 470 ohms (1), 1K (2).

- Electrolytic capacitors, 100µF (1), 1,000µF (1).
- Ceramic capacitor, 1µF (1).
- Pliers, wire cutters, wire strippers (1 each).
- Hookup wire, at least two colors, about 6" each.

Making Jumpers

To build a circuit on a breadboard, you will need jumpers like the ones I illustrated in Figure 6-14. Your first step is to make a few for yourself.

Grab a piece of hookup wire between appropriate-sized notches your wire strippers, as shown in Figure 8-1. Your wire should be 22 gauge, and if you bought some appropriate strippers, one pair of notches in their blades will be identified as "22." The notches are the right size to bite into the plastic insulation without cutting the copper conductor inside.

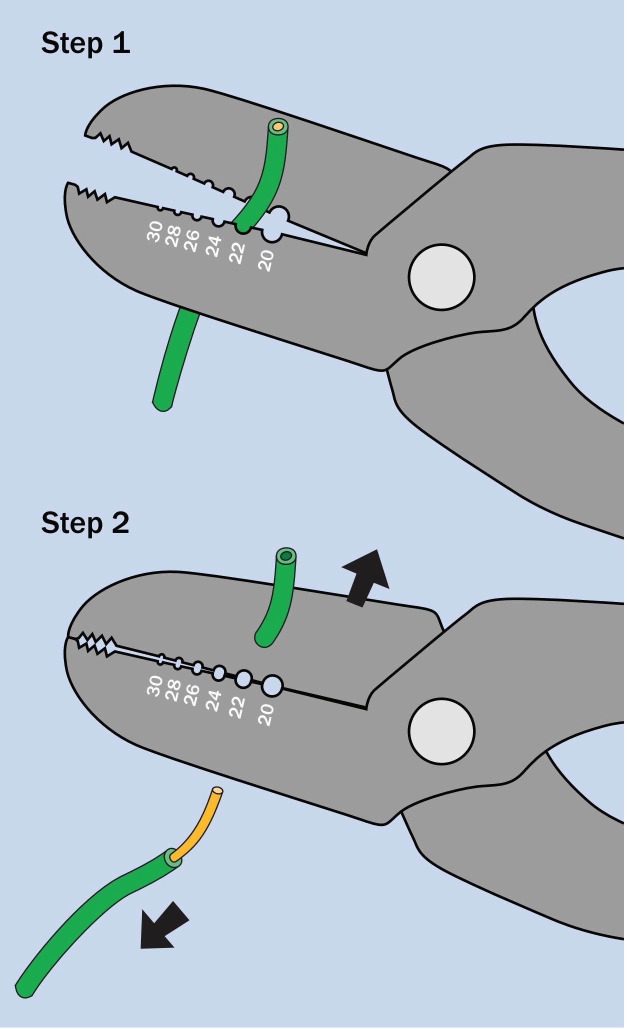


Figure 8-1. Stripping insulation from the end of a piece of wire.

In Step 2, after closing the wire strippers, you pull them upward while pulling the wire downward, as indicated by the arrows in the figure.

Now that you know how to remove insulation, I'll suggest the easiest way to make jumpers of a specific size. The steps are shown in Figure 8-2. First, remove a couple of inches of insulation and throw it away. Second, measure along the remaining insulation a distance equal to the length of the jumper that you want to make. Remember, holes on your breadboard are spaced 0.1" apart.

I happen to know that you're going to need some red and blue jumpers that are 1/2" long. One pair will supply current to the breadboard from a battery, and two more pairs will be used in the board itself. With this in mind, the distance labeled "X" in Figure 8-2 should be 1/2", and you'll need three red and three blue jumpers.

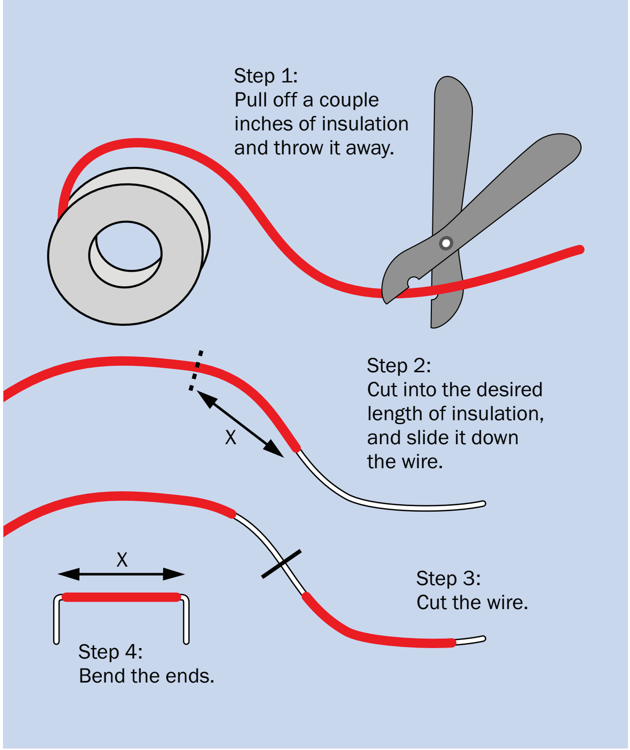


Figure 8-2. How to make jumpers.

Bringing in Power

A *bus* is an electrical conductor which carries power to some components. You supply power at one end of the bus, and the components tap into it.

Your breadboard has a pair of buses on each side, embedded in the plastic. All of the buses are physically identical, but they are marked externally with red and blue stripes to remind you which ones you will use for positive and negative power. All of my breadboard diagrams will show a red stripe at far left, so you should get into the habit of using your breadboard in that orientation.

Initially I will only be using one bus in each pair, because I won't be making complex connections. You need to supply positive power to the far-left red bus and negative to the far-right blue bus. One way to do that is shown in Figure 8-3, where jumpers have been pushed into the board and have been given a quarter-twist so that alligator test leads connect them easily with the battery.

Another option is shown in Figure 8-4, if you happen to have a battery connector with wires terminating in bare ends. You can push the ends directly into your breadboard—which seems a nice easy option, except that the ends of the wires are very thin, and may be difficult to insert. Also, they won't be as secure as 22-gauge hookup wire, and can pull out when you least want them to. Personally I prefer to use test leads as in Figure 8-3.

The First Board

I believe that diagrams are clearer than photographs when illustrating breadboard layouts, so I will be using them throughout the book, and Figure 8-5 shows some of the components that you will see. You have not encountered most of them yet, but you can refer back to this figure for reference.

Here are some important things to remember about the pictorial symbols:

- Concealed pins. Where a component has pins underneath it that I cannot show easily, I will indicate their locations as pink dots with white outlines. You can see them in the tactile switch (pushbutton), the slide switch, the trimmer, and the relay.

- Many components are shown in a perspective view, seen slightly from one side, to make them look more realistic. When you insert them into the breadboard, they should actually fit flush with its surface and will stand vertically.

- I will add a plus sign to each LED beside its long, positive lead, while each electrolytic capacitor will have a minus sign to remind you to look for minus symbols printed on that side of its aluminum shell. (I'll be introducing capacitors later in this experiment.) The negative end of a diode will be marked in some way by the manufacturer, but I'll add a negative sign

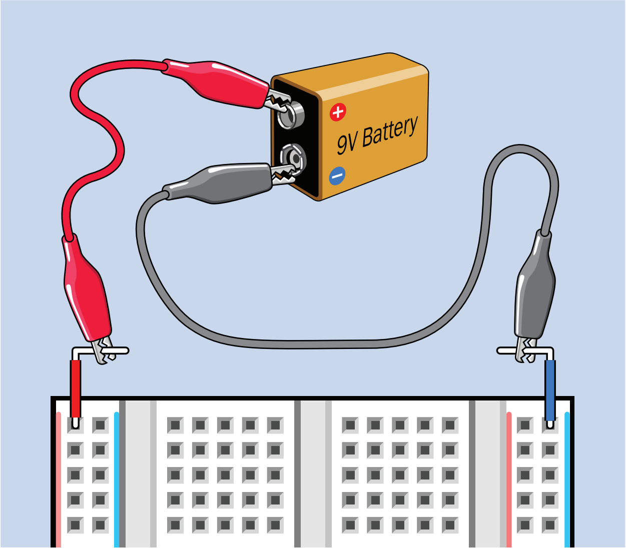


Figure 8-3. Powering your board with test leads and jumpers.

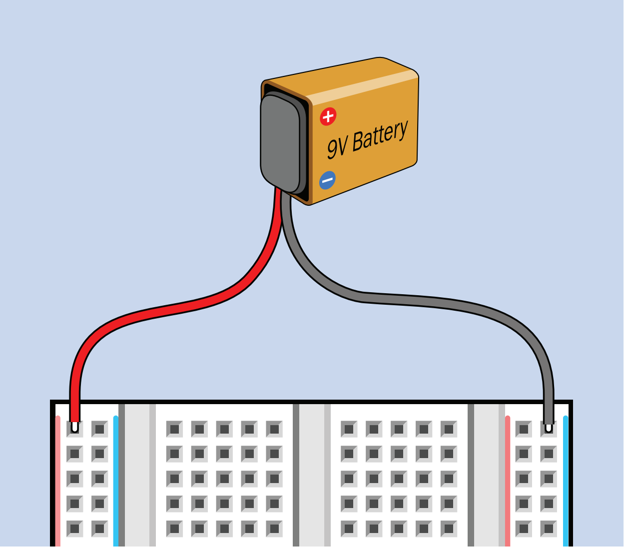


Figure 8-4. Powering your board with a battery connector.

just to make sure. If you're wondering, this type of diode preceded the type which you now know as an LED.

- I've shown an x-ray view of contacts inside the relay, to remind you how it works.

In Figure 8-6 the circuit uses the same relay that you used in Experiment 7. Conveniently, the two rows of pins are 0.3" apart, which allows you to plug them in across the channel down the center of the breadboard. You'll also recognize a resistor which is identified as being 470 ohms, and two red LEDs. The pushbutton is a tactile switch with pins 0.2" apart, so they plug conveniently into the board.

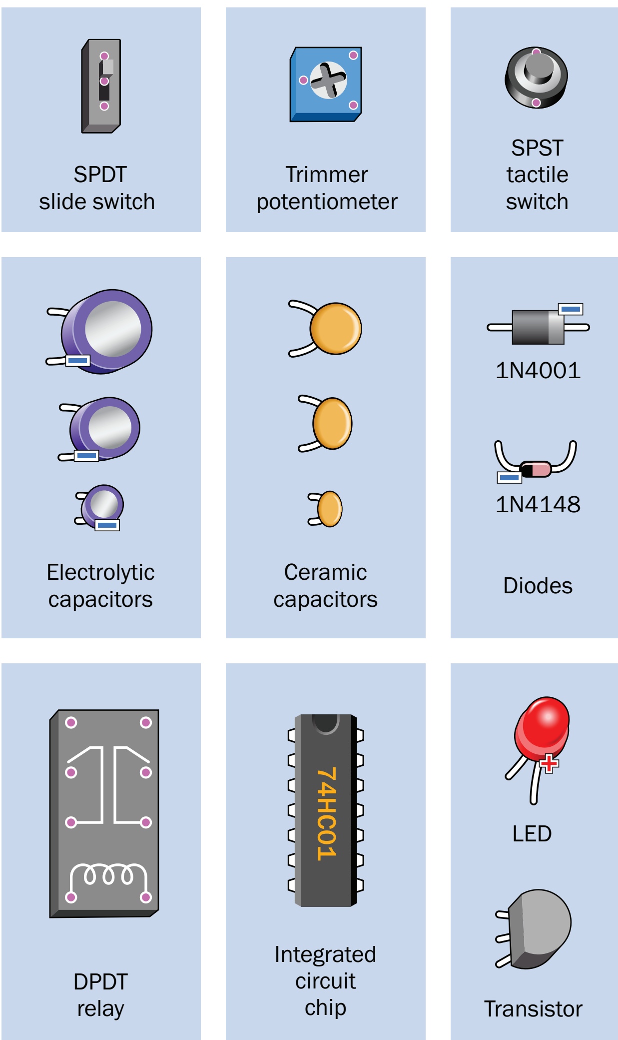


Figure 8-5. Components that will be breadboarded, now and later.

Note that the two red wires, on the left, and the two blue wires, on the right, are jumpers that have been inserted into the board.

- A note about formatting: In breadboard diagrams, anytime I show the value of a component it will be in a pale blue oval, so that you can distinguish it easily from the pattern of holes in the board. I won't add the blue oval in schematics, because it isn't necessary.

Now apply power, and the left-hand LED should light up. Press the tactile switch (the pushbutton at bottom-left), and the left-hand LED goes out, while the right-hand LED lights up. There it is: Your first breadboarded circuit. Now you need to understand the connections inside the board that make it work.

Inside the Board

In Figure 8-7, on the next page, I have shown the hidden strips that create connections, with little dots that indicate where each lead of a component can make contact.

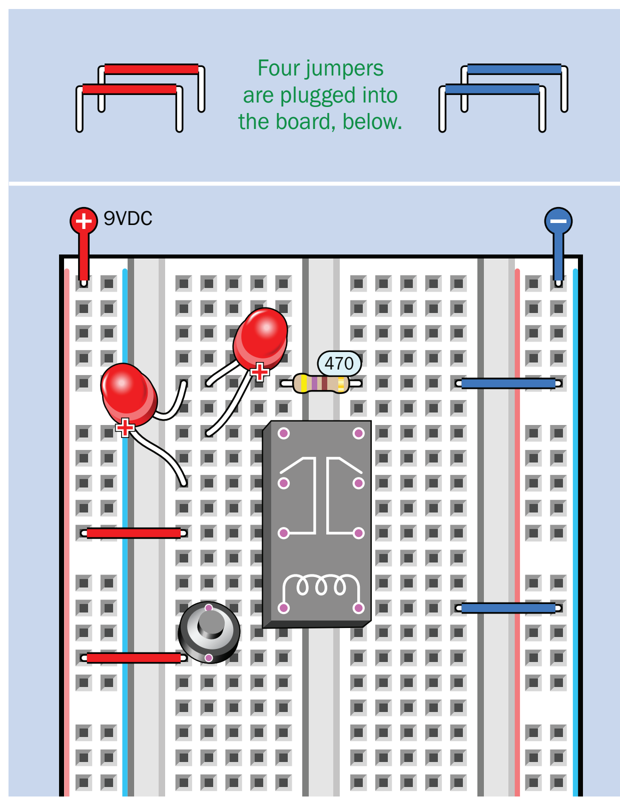


Figure 8-6. The first breadboarded circuit.

- Anytime you have difficulty fitting a component into a circuit in the way that I have shown it, you can move it one space left or right on the breadboard (if there is room to do so), because the leads will still make contact with the same horizontal strips inside the board.

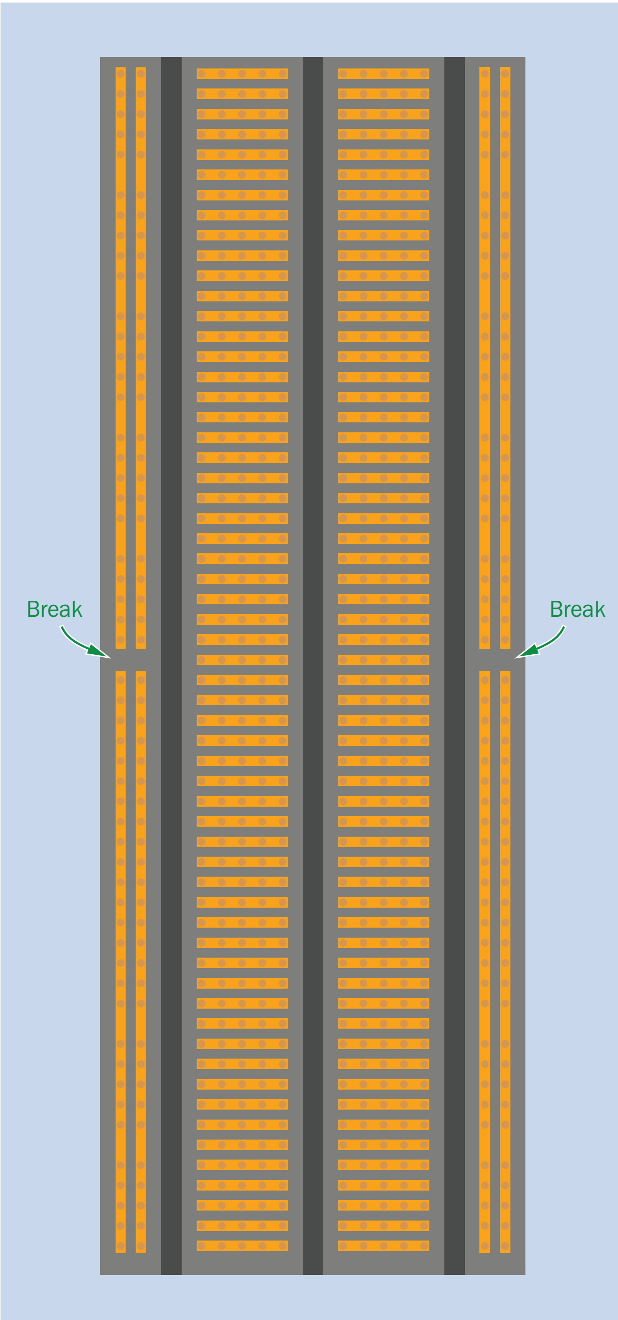


Figure 8-7. Connections inside a breadboard.

You'll notice that each bus has a break in it. Some boards have this feature, while others don't. Its purpose is to allow you to use two different power supplies, one for the top half of the board and one at the bottom. In practice you're unlikely to do this, and the breaks in the buses are annoying, because you may forget that they're there. When you build a circuit that extends down the board, and you find a mysterious lack of power around the halfway mark, you may realize that you forgot to add jumper wires bridging the gaps in the buses.

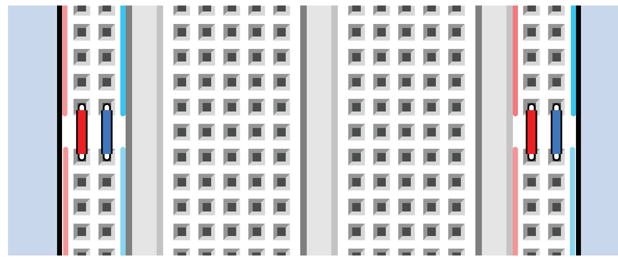


Figure 8-8. When a board has gaps at the center of its buses, add jumpers as shown.

To find out if your board has a gap in each bus, insert a jumper at each end, leaving their ends exposed, and use your meter to check for continuity between them. If necessary, add jumpers to bridge gaps in the buses before you start to build circuits. Figure 8-8 shows the center section of a board with the jumpers added.

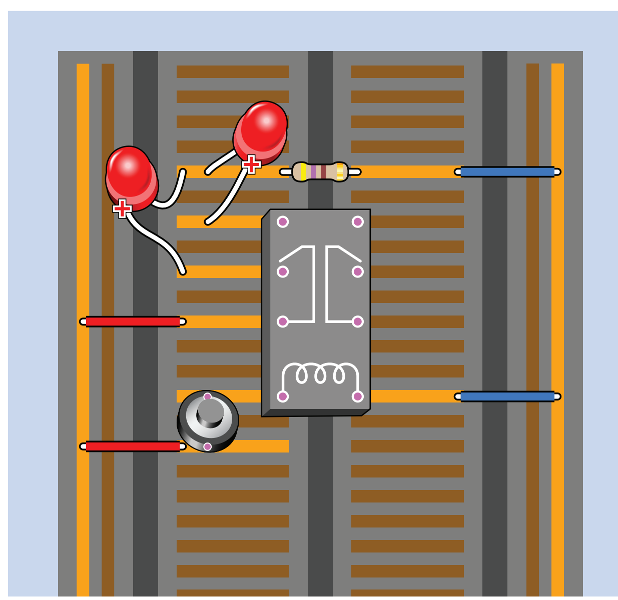


Figure 8-9. Connections between components in the relay circuit. Strips which are not connected have been darkened.

Relay Circuit, Revealed

I'm hoping that Figure 8-9 will help to explain how the relay circuit works, as it shows the way in which components are connected by strips inside the board. I have darkened the strips which don't connect any components and are not doing anything. You can imagine electric current following a zigzag path, starting from the positive bus, passing through the relay contacts, an LED, and the 470-ohm resistor, and finally reaching the negative bus. The resistance of the strips is so low, the length of the path doesn't matter.

Now take a look at the schematic for the same circuit in Figure 8-10. I arranged it to resemble the breadboard as much as possible. Eventually I'm going to rely on schematics more, and will only include breadboard diagrams for larger circuits.

You can see how current from the positive bus is connected with the left-hand LED through the relay contacts, even though the tactile switch is not powering the relay yet. The contacts are relaxed in this position.

In addition to coloring the wires to show which ones are approximately 9V and which ones are approximately 0V, I colored the LEDs to suggest which one is bright and which one is dark. The wires between the LEDs and the resistor are black, because the voltage will be somewhere between 0V and 9V, but I don't know the exact value.

Now look in Figure 8-11, and you see the relay coil receiving power and activating the switch, so that the other LED lights up.

If you're wondering why there is only one 470-ohm resistor to protect two LEDs, it's because the LEDs only light up one-at-a-time.

Making It Buzz

The next step is to modify your circuit to make it more interesting. Look at the new schematic in Figure 8-12, and compare it with the previous version in Figure 8-10. Can you spot the difference? In the new version, the current that comes from the bus has to circle around through the contacts inside the relay before it reaches the button, which can energize the coil.

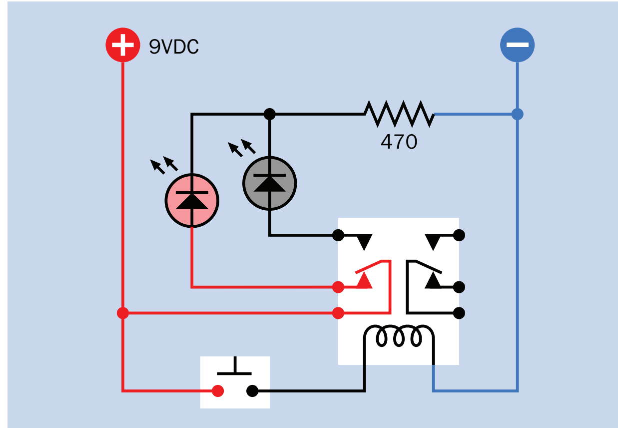


Figure 8-10. Schematic for the relay circuit.

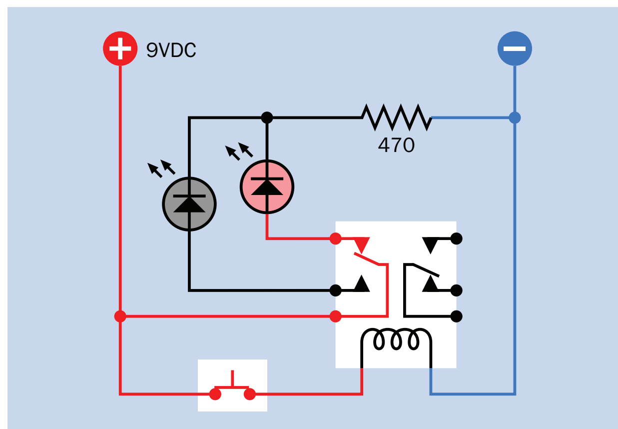


Figure 8-11. The relay with power applied.

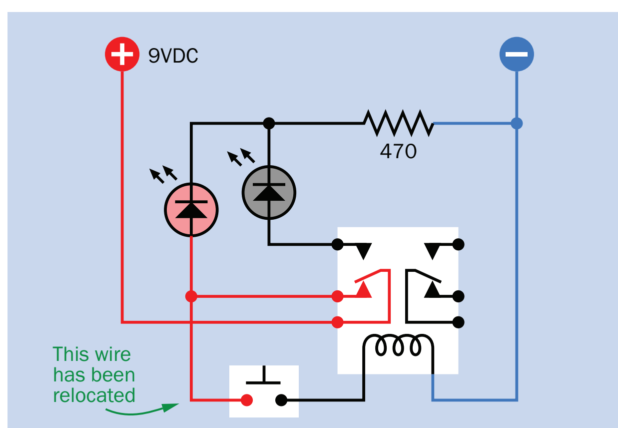


Figure 8-12. The modified schematic.

What effect will this have? Try it, and you'll find that the relay makes a buzzing sound. (If your hearing is not good, touch the relay to feel it vibrating.)

The same circuit is shown in Figure 8-13. The red jumper now supplies power to the pole of the relay. It passes through the relaxed relay contacts, out through the pin and down along the new green jumper that I added. This supplies the tactile switch. Now when you press the button, the relay moves its contacts—but wait! When the contacts open, they cut off power to the button, so the relay coil isn't energized anymore, and the contacts relax again. This reconnects power, so the contacts open again—and the cycle keeps repeating. The relay is vibrating between its two states.

Because you're using a small relay, it switches on and off quite fast. In fact, it vibrates perhaps 50 times per second (too fast for the LEDs to show what's happening).

When you force a relay to behave like this, it's liable to burn itself out or destroy its contacts—so don't hold down the pushbutton for long. To make the circuit less self-destructive, I want everything to happen more slowly, and I'm going to achieve this by using a ***capacitor***.

Adding Capacitance

Check the breadboard diagram in Figure 8-14, and you'll see I have added a new component at the bottom. This is an ***electrolytic*** capacitor with a value of 1,000μF, pronounced "one thousand microfarads." I'll explain these terms in a moment, after you see what the capacitor does.

Be sure to plug it in the right way around, with its negative lead facing the top end of the breadboard. The lower end of the capacitor connects through a yellow jumper to the other side of the coil, and in this configuration we say that the capacitor is wired ***across*** the coil.

If your capacitor has long leads, they may reach across the relay coil pins without needing the yellow jumper.

When you hold down the button now, the relay should click intermittently instead of buzzing, and the LEDs will flash alternately.

A capacitor is like a tiny rechargeable battery, but it acquires voltage in a fraction of a second, before the relay even has time to open its lower pair of contacts. Then, when the contacts are open, the capacitor releases its

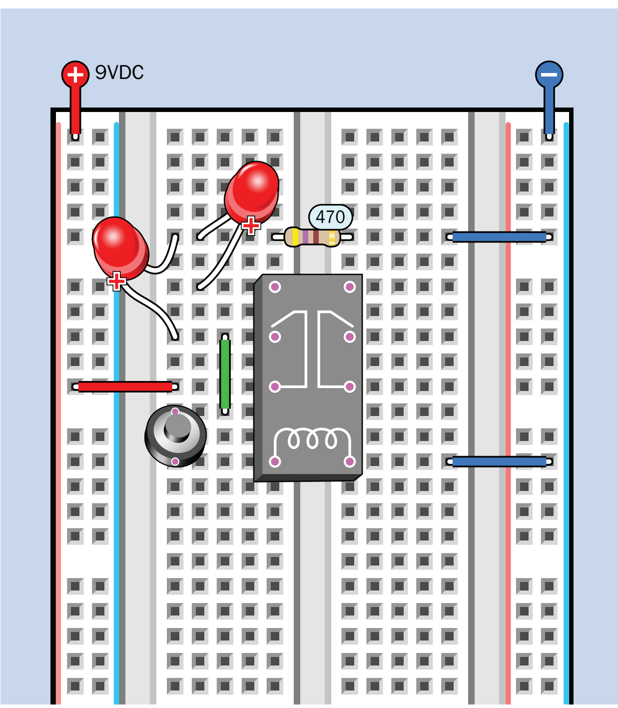


Figure 8-13. The breadboard version of the schematic previously shown in Figure 8-12.

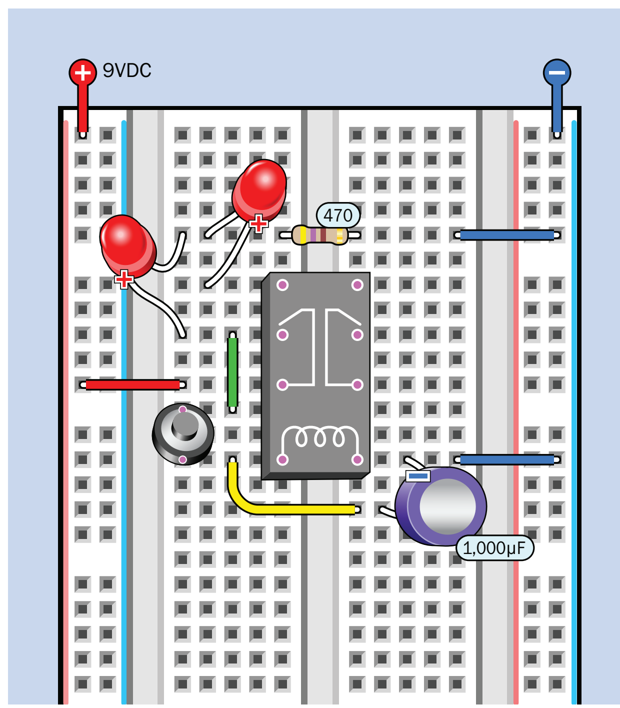


Figure 8-14. A 1,000μF capacitor has been added to the circuit shown in Figure 8-13.

power to the relay (and to the left-hand LED) to keep the coil of the relay energized for a moment.

The capacitor takes power from the circuit initially, then gives it back. During this process, the capacitor is ***charging*** and ***discharging***.

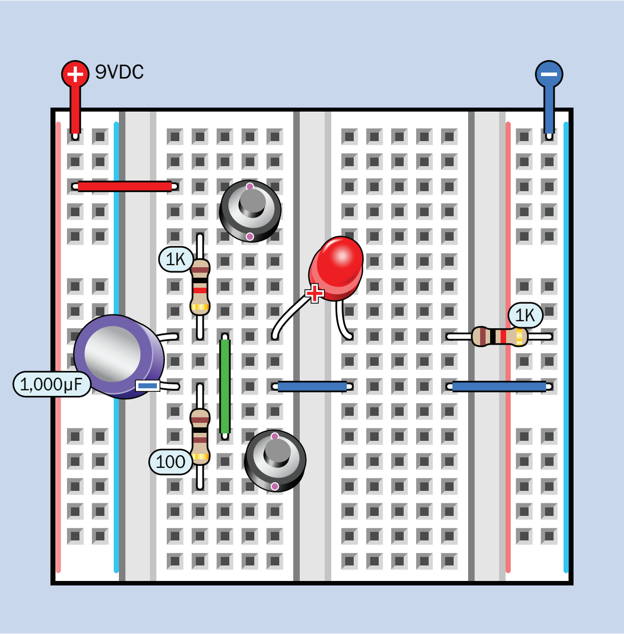


Figure 8-15. A circuit that reveals the charging and discharging of a capacitor.

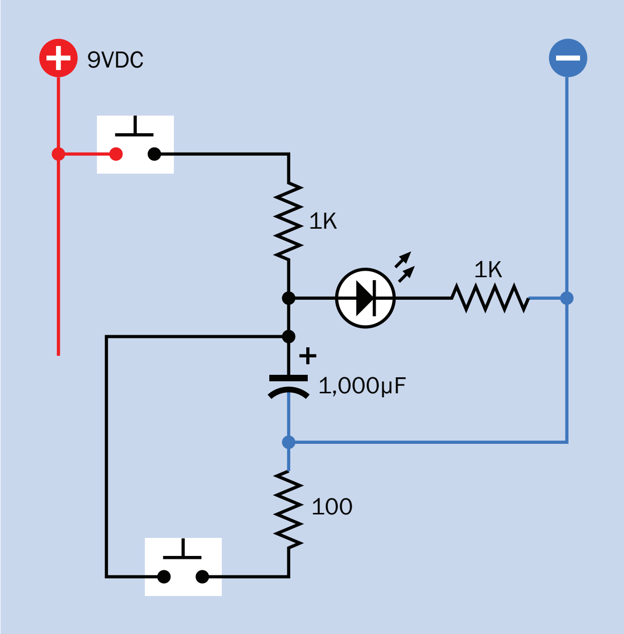


Figure 8-16. The schematic version of the circuit shown at left in Figure 8-15.

and the LED gradually fades away. Imagine the path of current, and you'll see that when you press the upper button, power goes through a 1K resistor to the upper lead of the capacitor. The capacitor has such a low internal resistance when it starts to charge, it sucks up current—but as voltage increases on the capacitor, the LED shares it, because the conductor inside the breadboard passes it along. This all happens relatively slowly, because the 1K resistor restricts the current.

Storing an Electric Charge

To get a clearer idea of what is actually going on inside a capacitor, build a new circuit as shown in Figure 8-15. Notice that the capacitor has been turned around with its negative side now facing the bottom end of the breadboard.

First press the lower button for a second. This is connected through the board and a 100-ohm resistor, across the two leads of the capacitor. When you hold down the button, any charge on one side of the capacitor neutralizes the charge on the other side, so you zero it out.

If you're wondering why the resistor is needed, it's to protect the contacts in the tactile switch from the sudden surge of current, as they are rated only for 50mA.

Let go of the lower button. Now hold down the upper button, and the LED lights up slowly; release the button

- A series resistor makes a capacitor charge more slowly.

Touch the lower button again, and the LED goes out very quickly, as the charge on the capacitor prefers to ground itself through the 100-ohm resistor instead of the LED.

You can see the same circuit redrawn as a schematic in Figure 8-16, including a new symbol that I am using to represent the capacitor.

You can also see an x-ray view of the breadboarded circuit in Figure 8-17 on the next page.

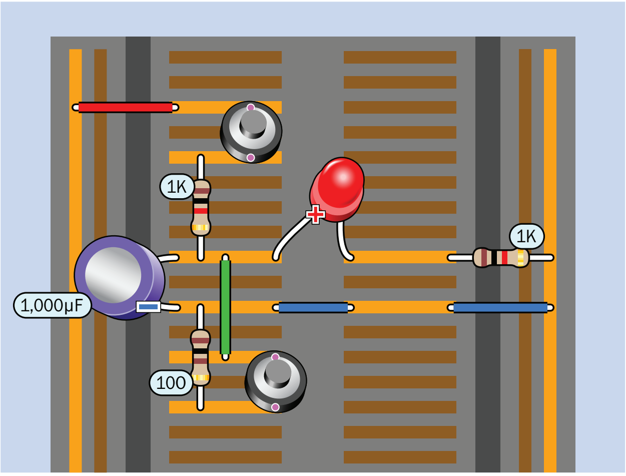


Figure 8-17. An x-ray view of Figure 8-15.

Here are some take-home messages from this very simple circuit.

- The capacitor has to have its lower lead grounded, otherwise the circuit doesn't work. You can verify this for yourself by removing the blue jumper at the center of the board.

- When a resistor and a capacitor are in series, the combination is known as an ***RC network*** (which stands for a "resistor-capacitor" network).

- A higher-value resistor in an RC network will make the capacitor charge more slowly. You can try this for yourself. Substitute a higher-value resistor for either of the 1K resistors.

- A higher-value capacitor in an RC network will charge more slowly. If you don't have a capacitor larger than 1,000µF, substitute a smaller one and see if the LED brightens and dims more quickly.

- A capacitor of the type that you are using may contain two strips of very thin aluminum foil rolled up together, each attached to one of the leads on the capacitor. The foil strips are often referred to as ***plates***, because very early capacitors did consist of two metal plates with a small gap between them.

- When one plate has a positive voltage, it attracts negative voltage to the other plate, which is why the lower lead of the capacitor in this circuit must be grounded. The blue wires provide a source of electrons.

The concept of opposite charges attracting each other inside a capacitor is shown in Figure 8-18.

- A capacitor holds its charge after you disconnect it from a circuit, but the insulation inside it is not perfect, and allows gradual leakage of charge between the two plates.

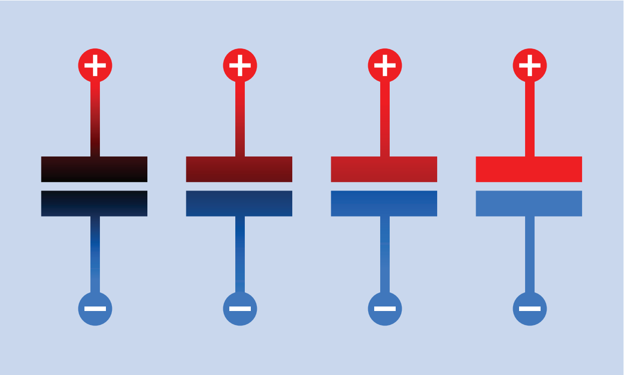


Figure 8-18. Four snapshots illustrating the process by which a capacitor acquires a charge.

Farad Basics

The storage capacity of a capacitor is measured in farads, represented by an uppercase F. The term is named after Michael Faraday, another in the pantheon of electrical pioneers.

Large capacitors with values in farads are difficult and expensive to make, but fortunately in modern electronics we have little need for them. Circuits usually contain capacitors calibrated in microfarads, nanofarads, and picofarads. Working from the smallest unit to the largest:

```
1,000 picofarads (pF) = 1 nanofarad (nF)
1,000 nanofarads (nF) = 1 microfarad (µF)
1,000,000 microfarads (µF) = 1 farad (F)
```

See Figure 8-19.

What about millifarads? Do they exist? They should, shouldn't they? After all, we have milliamps and millivolts. Would there be 1,000 microfarads in a millifarad, and 1,000 millifarads in a farad?

The answer to all of these question is "yes," and the abbreviation mF is used, but not often. One reason may

Picofarads	Nanofarads	Microfarads	Farads
1pF	0.001nF	0.000001µF	
10pF	0.01nF	0.00001µF	
100pF	0.1nF	0.0001µF	
1,000pF	1nF	0.001µF	
10,000pF	10nF	0.01µF	
100,000pF	100nF	0.1µF	
1,000,000pF	1,000nF	1µF	0.000001F
		10µF	0.00001F
		100µF	0.0001F
		1,000µF	0.001F
		10,000µF	0.01F
		100,000µF	0.1F
		1,000,000µF	1F

Figure 8-19. Conversion table for units of capacitance.

be that "mF" is too confusing, as people can misread it to mean "microfarad." You are unlikely to run into "mF" very often.

Capacitors commonly used in small electronic circuits are likely to range from 0.1nF (which is 0.0001µF) up to 1,000µF. Lower-value capacitors tend to use the same two-digit multipliers that I described in values for resistors: 1.0, 1.5, 2.2, 3.3, 4.7, and 6.8.

You need to get used to the meaning of picofarads, nanofarads, and microfarads. To change a value in pF to nF, move the decimal point three spaces to the left. To change nF to µF, move the decimal point another three spaces to the left.

To change a value in µF to nF, add three zeroes. To change nF to pF, add another three zeroes. You can see this in Figure 8-19.

Capacitor Reference

Here are some details that you may want to refer back to later.

Many varieties of capacitors exist, but the two most common are *ceramic* and *electrolytic*, both of which will be used throughout this book.

Ceramic capacitors usually look like little discs or blobs, like the one on the right in Figure 6-25. They are often, but not always, beige or blue in color, and are most commonly used in small values below 1µF. They have no polarity; you can connect them either way around.

Electrolytic capacitors are shaped like miniature tin cans, wrapped in thin plastic film of any color, like the one on the left in Figure 6-25. They are most commonly used in values above 1µF, where they become cheaper than ceramics. They do have polarity and must be connected the right way around. The negative side is marked on the can, and the negative lead is almost always shorter than the positive lead.

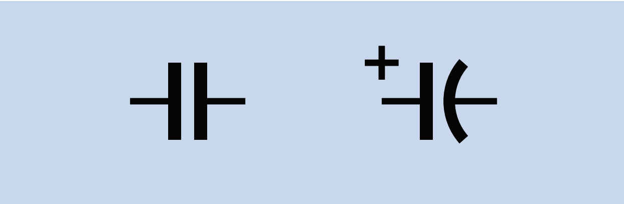


Figure 8-20. Schematic symbols for capacitors. See text for details.

The schematic symbol for a capacitor has two significant variants, shown in Figure 8-20. The symbol with two straight lines is used generically for any capacitor, and you can think of the lines as representing the plates inside it. The variant on the right, with a curved line, is only used to represent a capacitor with polarity, and the curved side is the negative side. I will always include a plus sign with this symbol, but you will find schematics where people don't bother to do this. You will also find schematics where people use the generic, straight-plates symbol when in fact you are likely to use an electrolytic capacitor, and you must figure out which side is likely to be "more positive" than the other.

Here's a general rule about capacitor substitution:

- Because a ceramic capacitor can be connected either way around, you can always use one instead of an electrolytic capacitor.
- If a circuit is designed so that current may flow in either direction through a capacitor, you should avoid using an electrolytic capacitor.

- Non-polarized electrolytic capacitors are available, but are seldom used. They actually consist of two regular electrolytics in series, with opposite polarities.

If you look back at the schematic in Figure 8-16, you'll see it's very obvious which side of the capacitor is going to be more positive than the other, because one side is connected directly to negative ground. I'll mention the polarity issue wherever it may need clarification in circuits throughout the book.

Blocking DC

You have seen that a capacitor can store and release an electrical charge. But this isn't the same as allowing current to pass through. In fact, a capacitor will block DC current—generally speaking—because the plates inside it don't touch each other.

You can run a very simple experiment to test this. Set your meter to measure mA, and put it in series with a 10K resistor and a 1uF ceramic capacitor. Apply 9V from one end of the series to the other, and you should measure no current passing though. Unplug the capacitor and plug it back in the other way around. Still you will find no current. The ceramic capacitor has no polarity.

You can try the same thing with a 100uF electrolytic capacitor. Put it in series with the 10K resistor and the meter, and when the polarity of the capacitor is correct, the current measured by the meter will settle to near zero, as the capacitor blocks current. Now turn the capacitor around, and your meter will show that the capacitor is passing some current.

- An electrolytic capacitor blocks DC current when it is used correctly, but if you use it the wrong way around, it offers relatively little resistance.

What would happen if you removed your meter and applied current directly to the capacitor, with wrong polarity, and without the 10K resistor in series to protect it?

I don't think this is a good idea. You would damage the capacitor permanently, and in fact it might get hot enough, quickly enough, to burst open.

Electrolytics are not the only type of capacitor requiring you to observe polarity. Tantalum capacitors, for example, are fussy about being the right way around. You have to be careful with them, bearing in mind that their polarity is marked with a plus sign that is so tiny, some people may not notice it. People like me, for instance.

Figure 8-21 shows what happened after I connected a tantalum capacitor backward across a power supply that could deliver substantial current. After about ten seconds, the capacitor popped open like a tiny firework and scattered small flaming pieces, some of which burned their way into the breadboard. So—the lesson is clear. Observe polarity!

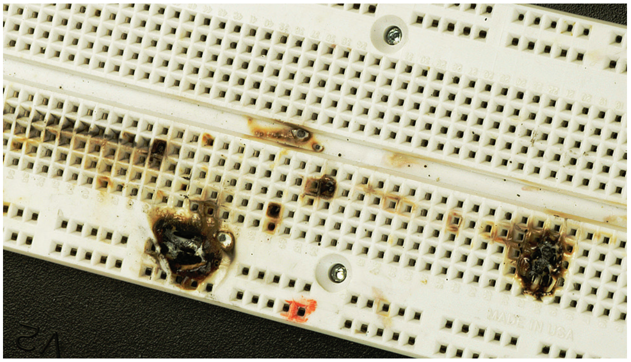


Figure 8-21. Breadboard scarred by an bursting tantalum capacitor.

Michael Faraday and Capacitors

The farad is named after Michael Faraday, who discovered that when you have two metal plates parallel to each other and very close together, a charge applied to one plate tends to attract an opposite charge to the other plate. Faraday was an English chemist and physicist who lived from 1791 to 1867, and you can see him in Figure 8-22.

Although Faraday was relatively uneducated and had little knowledge of mathematics, he had an opportunity to learn about science by reading a wide variety of books while working for seven years as a bookbinder's apprentice. Also, he lived at a time when relatively simple experiments could reveal fundamental properties of electricity. He made major discoveries including electromagnetic induction, which led to the development of electric motors. He also discovered that magnetism could affect rays of light.

His work earned him numerous honors, and his portrait was printed on English bank notes denominated in 20 pounds sterling, from 1991 through 2001.

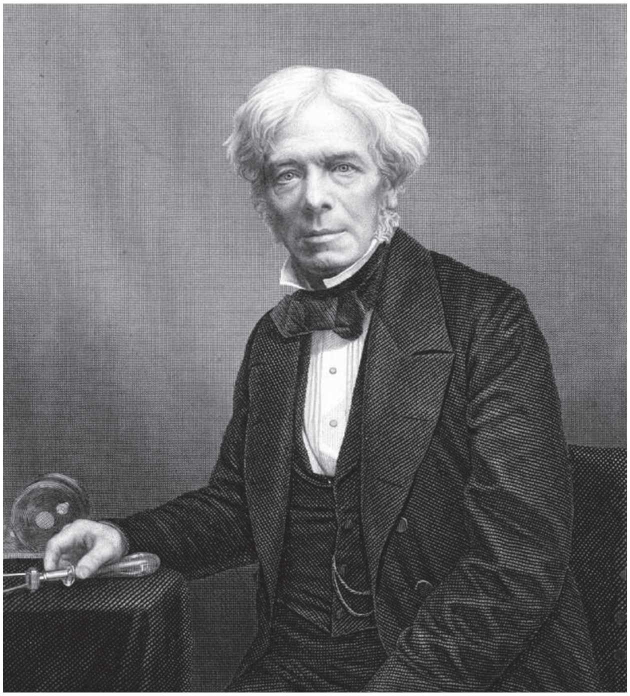


Figure 8-22. Michael Faraday, after whom the Farad is named.

Reading Electrolytic Capacitors

Electrolytic capacitors have their values printed on them, along with their working voltage, which is the maximum the capacitor will tolerate reliably. This is important, because the insulation layer inside a capacitor is very thin and can break down if it is subjected to excessive voltage.

However, a capacitor can always be used with a lower voltage than its rated working voltage.

Really? Are there any limits to this statement? Suppose you have a capacitor rated for 250V—can you use it in a circuit with a 5V power supply? Yes, you can. A higher voltage rating just means that the capacitor will probably be physically larger and more expensive. It will still work on a lower voltage.

Reading Ceramic Capacitors

Modern ceramic capacitors can contain multiple interleaved plates that are microscopic in size. Some ceramic capacitors are shown in Figure 8-23. From left to right, they are 56pF, 22pF, and 100,000pF (0.1 microfarad). The scale at the top shows intervals of 0.1".

Back in the twentieth century, disc capacitors were more common, containing two flat discs as their plates. Two examples are shown in Figure 8-24. Left: 1,500pF (1.5nF). Right: 47pF with 20% tolerance, rated for 1kV. The scale at the top shows intervals of 0.2".

As you can see, a ceramic capacitor usually has a code printed on it. Here's how to read it:

First two numerals: The beginning of the value of the capacitor.

Third numeral: The number of subsequent zeroes.

Also, there may be a letter at the end, which indicates the tolerance of the capacitor. The examples that you are likely to find are listed on the next page.

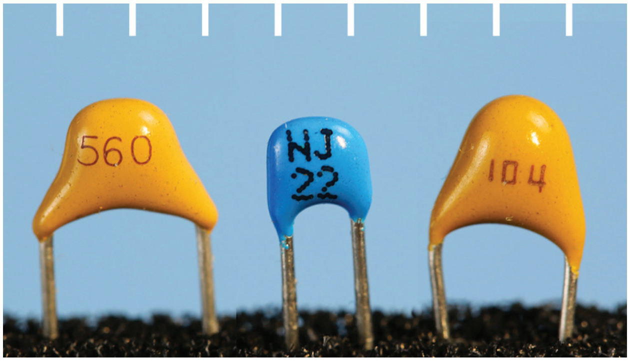


Figure 8-23. Modern ceramic capacitors are very small and usually rounded in shape. The scale divisions are spaced at 0.1".

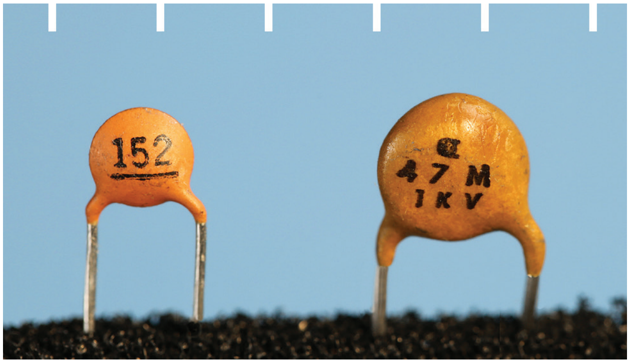


Figure 8-24. Two old-school ceramic disc capacitors.The scale divisions are spaced at 0.2".

- `J` means 5%
- `K` means 10%
- `L` means 15%
- `M` means 20%

The first pair of numbers are always in picofarads, so if you have a ceramic capacitor with 473M printed on it, the value will be 47,000pF (47 followed by three zeroes) with a tolerance of 20%.

Remember how to convert from small units to larger units by moving the decimal point left three spaces: 47,000 pF is equivalent to 47nF, which is the same as 0.047μF.

Here's another example. Suppose you have a ceramic capacitor that identifies itself as 685K. The value will be 68 followed by 5 zeroes, or 6,800,000pF. That's 6,800nF, or 6.8μF—an unusually high value for a ceramic capacitor, but they do exist. As for the letter K, it does not refer to kilo-anything. On a ceramic capacitor, it means a tolerance of 10%, unless it is used separately followed by letter V to mean "kilovolts."

What if a code consists of only two numerals, such as 22? This will be the value in picofarads, with no additional zeroes—as is the case of the blue capacitor in Figure 8-23.

But how do you know the working voltage? Often the voltage isn't shown, and you have to rely on the supplier to tell you what it is. When you buy small components, usually they will be delivered to you in little plastic bags with labels on them, and you can store the capacitors that way. Alternatively, if you want to keep components in parts boxes (such as the one shown in Figure 24-3), you'll have to make little labels of your own. My own storage system for capacitors is shown in Figure 24-7.

If you take a ceramic capacitor out of its container and forget what its working voltage is, you have a problem. Fortunately, most are rated at 25VDC or higher (unlike electrolytics, which may be rated as low as 5VDC). When I checked some suppliers, I found that fewer than 1 percent of through-hole ceramic capacitors were rated below 25VDC. Since you probably won't build circuits that use more than 12VDC, I suggest that if you buy your own components, you should always choose ceramic capacitors rated at 25VDC or higher. Then you won't have to worry about their voltage rating.

Caution: Getting Zapped

If a very large capacitor is charged with a high voltage, it can retain that voltage for minutes or even hours. Because the circuits in this book use low voltages, you don't have to be concerned about this here, but if you are reckless enough to open an old appliance that uses high voltages (such as an antique TV set with a cathode-ray tube), you may have a surprise that can be fatal. Seriously: If you have plugged in an old TV, don't go poking around inside it, even after you have unplugged it. The voltages are lethal.

Experiment 9
Time and Capacitors

This experiment will demonstrate the fascinating relationship between capacitors and time. I'll also mention how a capacitor can be used to smooth electric current, and you'll see the mysterious phenomenon of *capacitive coupling*.

You Will Need:

- 9V battery, must be relatively fresh (at least 9.1V actual voltage).
- Breadboard, hookup wire, wire cutters, wire strippers, test leads, and multimeter—as before.
- Tactile switches (2).
- Generic red LED (1).
- Resistors: 100 ohms (2), 470 ohms (1), 1K (2), 10K (2).
- Capacitors: 100µF, 1,000µF (1 of each).

Inside a Capacitor

First set your meter to measure volts, check the voltage of your 9V battery, and make a note of it, because I will ask you to refer back to it later. For this experiment, your battery voltage must be 9.1V or higher. If it isn't, you will need to substitute a fresher battery.

The circuit in Figure 8-15 gave you a quick introduction to the behavior of an RC network, but now I want you to find out exactly how a capacitor charges and discharges. This will mean using a meter instead of an LED, so the first step is to remove the LED and its series resistor. Your new circuit should look like Figure 9-1. Notice the new features:

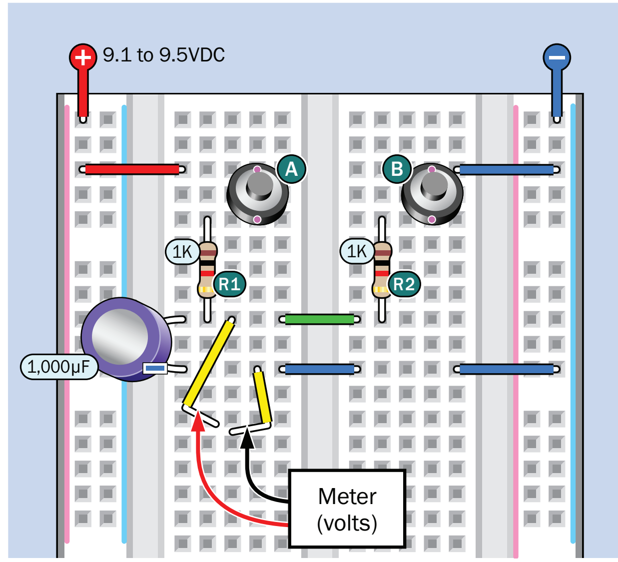


Figure 9-1. A circuit to reveal the charging and discharging of a capacitor.

- The positive voltage supply must be 9.1V or more.
- A second tactile switch has been added.
- Two yellow jumpers have their bottom ends bent outward and exposed, so that you can touch them with your meter probes to find out how much voltage is on the plates of the capacitor.

I have labeled the tactile switches A and B and the resistors R1 and R2, so that I can refer to them easily. Here are some general rules about breadboard diagrams in the remainder of the book:

- Whenever you see white type in a dark oval, this is a label which I will use to explain which component I'm referring to. The label has nothing to do with the value of a component (such as how many ohms are in a resistor).
- Values of components will always be shown using black type on a pale blue background, to distinguish them from labels. Later in the book, when I have more components on a breadboard, I will summarize all the components at the bottom of the diagrams.
- In schematics, which are visually simpler, I will use plain type for labels and component values.

Ready for the test? First touch the free ends of the two yellow jumpers together to make sure that the capacitor is totally discharged. Then separate them and attach the meter probes to them, using alligator test leads if you have them. The meter should show 0V.

You will need to time the charging process, using a phone app or a clock or watch with a seconds display. Start your timer as you hold down Button A. How long does the capacitor take to reach 9V?

When I performed this experiment, the meter showed 9V in about 3 seconds.

Substitute a 10K resistor for R1. Discharge the capacitor using the yellow jumpers, and repeat the experiment.

- Did the capacitor take ten times as long to reach 9V when you used the 10K resistor instead of the 1K resistor?
- Did the voltage across the capacitor rise at a steady rate, or did it increase faster at the beginning than toward the end?
- If you wait long enough, will the capacitor ever reach the actual battery voltage that you measured?
- When you let go of Button A but continue to measure voltage on the capacitor with your meter, does the voltage diminish very slowly?
- Substitute a 10K resistor for R2. When you hold down Button B, does the capacitor discharge at the same rate as when you charged it?

I'm going to show you how to investigate all of these questions.

Voltage, Resistance, and Capacitance

Imagine R1 as a faucet restricting a flow of water, and the capacitor as a balloon that you are trying to fill, as shown in Figure 9-2. If you screw down the faucet until only a trickle comes through, this is similar to increasing the resistance in a circuit, and the balloon will take longer to fill.

Initially, the pressure in the pipe is greater than the pressure inside the balloon. Consequently, water flows into the balloon very rapidly—but as this process continues, the rubber of the balloon stretches and the pressure inside it increases. The back-pressure slows the flow from the faucet, and the water stops flowing when the pressure inside it is equal to the pressure in the pipe—so long as the balloon doesn't burst.

The situation inside a capacitor is similar. Initially, the electrons rush in, but as the voltage increases, the newcomers take longer to find a resting place. The charging process gets slower, and slower. In fact, theoretically, the charge on the capacitor never quite reaches the voltage being applied to it.

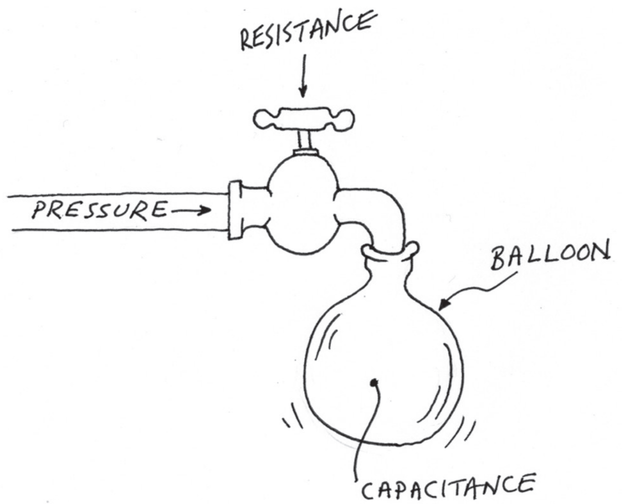


Figure 9-2. Water flowing into a balloon can be compared with electric current flowing into a capacitor.

The Time Constant

The speed with which a capacitor charges is measured with a number known as the *time constant*. The definition is very simple. Suppose you have a capacitor of F farads, being charged through a resistor of R ohms. In that case the time constant, TC, is calculated like this:

```
TC = R * F
```

The circuit in Figure 9-1 charged a 1,000μF capacitor through a 1K resistor. I can put these values into the time-constant formula—but only if I convert the units to ohms and farads. Well, 1K is 1,000 ohms, and 1,000μF is 0.001 farads, so that's easy enough:

```
TC = 1,000 * 0.001 = 1
```

Yes, it's as simple as this: For the resistor and the capacitor that you used, TC = 1.

Figure 9-3. A hungry gourmet loads his stomach as if it's a capacitor.

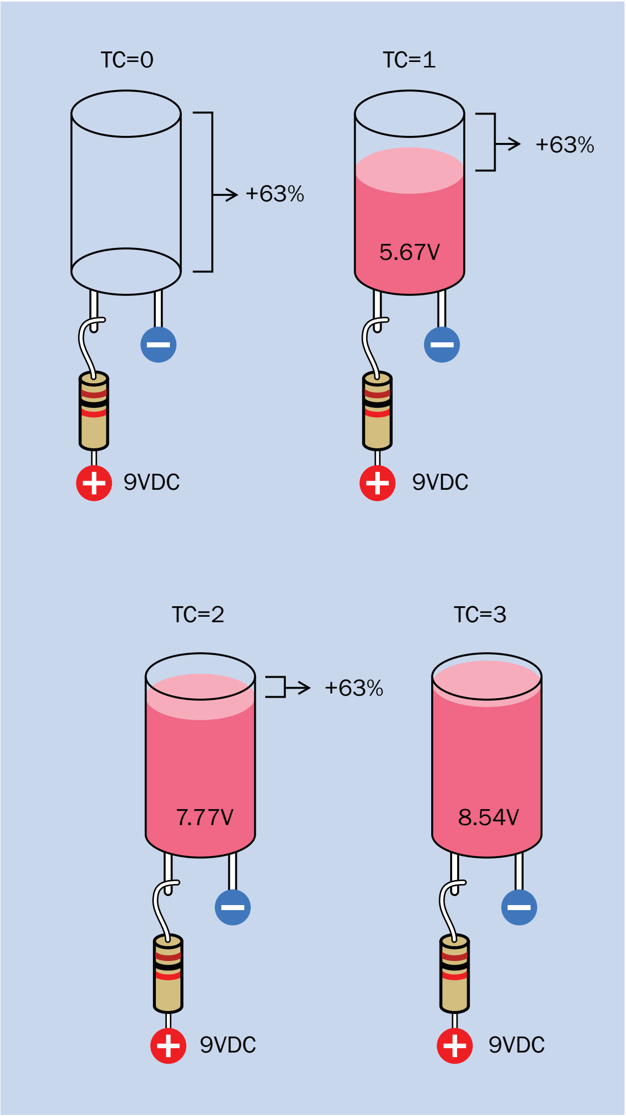


Figure 9-4. Visualizing the capacitor as a cylinder of fluid.

But what exactly does this mean? Does it mean that the capacitor will be fully charged in 1 second? No, that would be too easy, and your experiment already showed that it didn't happen as quickly as that. Here's the definition of TC, the time constant:

- TC is the number of seconds required for a capacitor to acquire 63% of the voltage being supplied to it (assuming it starts with zero volts).

Why 63%? Why not 62%, or 64%, or 50%? The detailed answer to that question is a bit technical. I will show you how it works, but if you want to know why it works, search online for

```
capacitor time constant
```

Figures 9-3 and 9-4 suggest two ways to visualize it. In Figure 9-3, the greedy individual who is planning to fill his stomach with cake is like a capacitor wanting to fill itself with electrons.

At first he's ravenously hungry, so he transfers 63% of the cake to his stomach in 1 second, which is his time constant for eating cake. In his second bite, he slows down a bit, because he's not quite so hungry now. He eats 63% of the remaining cake, which takes 1 more second. In his third bite, he takes 63% of what still remains, and still ingests it in one second. And so on. He is gradually filling himself with cake, but he never quite eats all of the cake, because he only takes 63% of whatever is left.

In Figure 9-4 I have shown the capacitor another way, as if it's a cylinder filling with pink fluid. The height of the fluid represents the voltage. After each time constant (which you will remember is 1 second if we have a 1,000μF capacitor and a 1K resistor), the capacitor acquires another 63% of the difference between the charge it had and the voltage being applied.

Although in theory the charging process for a capacitor can continue forever, in the real world there's a general rule of thumb:

- After 5 time constants, the charge is so close to 100 percent, we can think of the process as being complete.

If you prefer to see this as a graph, Figure 9-5 shows the voltage on the capacitor increasing with time. I calculated the points on the graph using the TC formula.

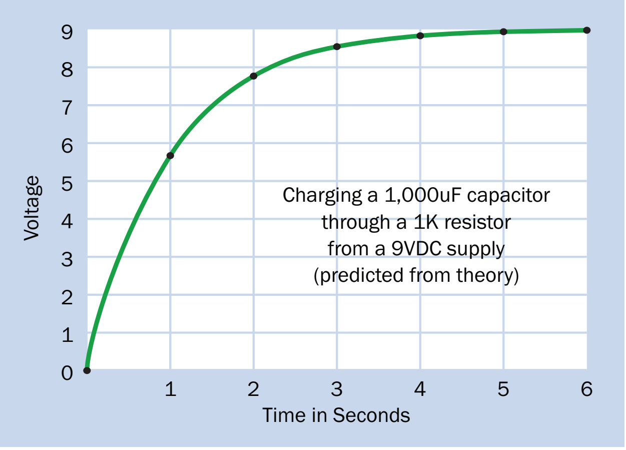


Figure 9-5. Graphing the charge on a capacitor using its time constant.

I don't expect anyone to remember exactly how much voltage a capacitor is likely to have on its plates while it is charging, but you do need to bear these points in mind:

- If you reduce the value of the resistor or the size of the capacitor, the charging process is faster.
- If you increase the value of the resistor or the size of the capacitor, the charging process is slower.

Time Constant Verification

You can check for yourself if my numbers are correct. This is always a good idea, and is very easy.

Refer back to Figure 9-1. Use two 10K resistors for R1 and R2, so that you will have time to make notes during the charging and discharging process.

Discharge the capacitor by touching the yellow jumpers together, then start timing the charging process while you hold down Button A . Note the voltage every 10 seconds until 1 minute has passed. You'll have an easier time doing this if a friend can do the timing and tell you when to write down each value.

Remember, TC is equal to the capacitor value multiplied by the resistor value, in farads and ohms. You are now using a 10K resistor instead of a 1K resistor, so TC is now 10 times bigger. However, you're only noting the voltage every 10 seconds, so your series of readings should still look like the series of voltages that I graphed at 1-second intervals.

You can also measure the time taken by the capacitor to discharge through R2 while you hold down Button B. It should be the same as the time the capacitor took to charge through Button A.

Of course, nothing in electronics is totally precise, and here are some factors that would make your readings different from mine:

- Your battery, resistor, and capacitor may be different from mine.
- Your meter is not totally accurate, and you will take a few microseconds to read it.
- Capacitors suffer from leakage, so you lose a little charge even while you are adding charge.
- Your meter has an internal resistance which is very high, but still steals a little bit of the charge from your capacitor. This is why its reading will drop very slowly if you keep it connected to the yellow jumpers while you don't touch either of the buttons.

At this point, I think you will agree with this general rule:

- The process of measuring a value always tends to alter the value that you are trying to measure.

Can You Make a Timer?

Here's an idea: Instead of using a clock to find the time it takes to charge a capacitor, you could charge a capacitor to measure time. When the capacitor reaches a particular voltage, it can switch on an LED, somehow, and the circuit will function as a stopwatch.

Can you think of a way to do it? You may need one additional component, which I will introduce in Experiment 10. At the end of that experiment, I'll show you a very simple timer.

Meanwhile, there are a few more things you need to know about capacitors. They are so versatile, there are two more tasks they can perform: Smoothing and coupling.

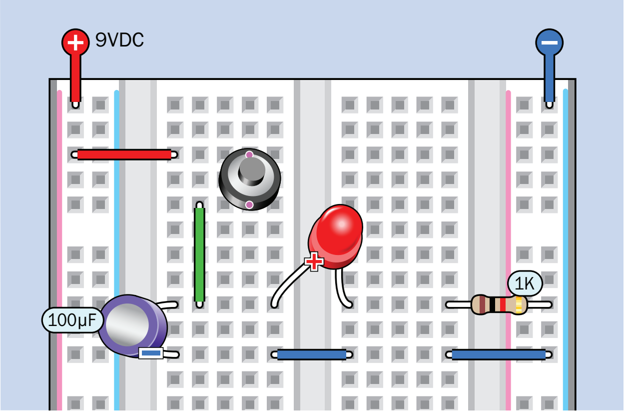


Figure 9-6. Demonstration of smoothing.

Smoothing

A capacitor can be used to smooth a signal, and the circuit in Figure 9-6 will give you a quick and simple demo. Its principle is similar to that of the circuit in Figure 8-15, except that the 1K resistor on the left has been replaced with a green jumper, the capacitor is now 100μF instead of 1,000μF, and the two yellow jumpers have been removed.

Tap the button as rapidly as you can, and watch the LED. Now remove the capacitor, and repeat the test. Notice that when the capacitor is there, it smooths the on-off transitions of the LED.

The schematic in Figure 9-7 may clarify the way in which a capacitor is used to perform the smoothing task. Suppose your power supply is not a steady 9V. It fluctuates erratically, and the "other components" in the schematic don't like that. If you add the capacitor, it absorbs each little surge, which we can call a ***spike*** in the power supply. Then the capacitor gives it back before the next spike comes through.

Voltage spikes may also be described as ***transients***, because they don't last for very long.

If you substituted a smaller capacitor, it would have less of a smoothing effect but it would accumulate a charge more rapidly. There's a basic principle, here:

- A smaller capacitor can still smooth smaller fluctuations in current that occur more briefly and rapidly.

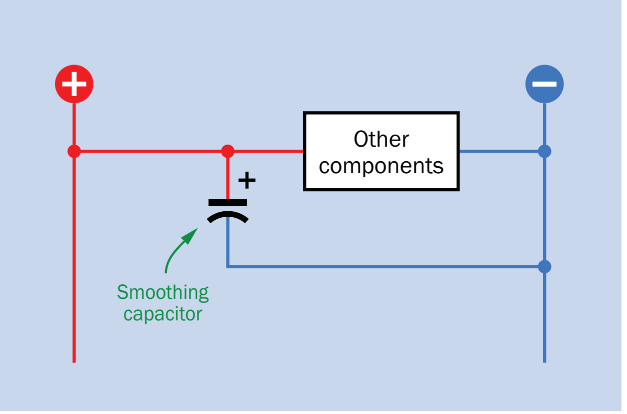


Figure 9-7. The basic configuration for a smoothing capacitor.

This capability can be useful if there is electrical noise which you want to get rid of. For instance, you may be using an AC adapter, sometimes known as a "wall wart," which you plug into an AC outlet in your home, to create DC voltage for a device such as an LED desk lamp. The adapter may create messy DC current with spikes in it. This will be good enough for a lamp, but if you wanted to use it to power a circuit using silicon chips, you could add a smoothing capacitor to smooth the current.

Here's another application: By choosing the size of a capacitor, you can make it sensitive to certain frequencies.

This can be useful in audio circuits. Figure 9-8, on the next page, shows an imaginary sound wave before and after it has been cleaned up by an imaginary capacitor.

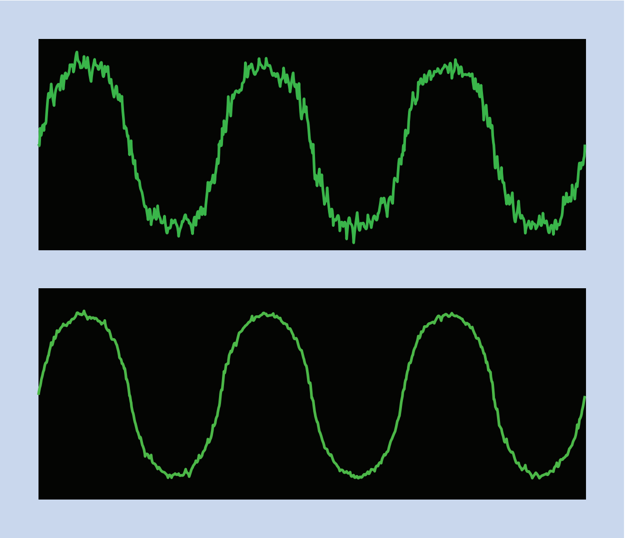


Figure 9-8. Top: An imaginary sound wave before it has been smoothed by a capacitor. Bottom: After smoothing.

Capacitive Coupling

Now I'm going to show you a particularly strange thing that a capacitor can do. It's a phenomenon that is often misunderstood.

I mentioned in Experiment 8 that a capacitor will block DC current (assuming you are using it the right way around, if it has polarity). But this blocking capability only works if the voltage is steady. Sudden changes in voltage will create the impression that current is zapping straight through.

How can this happen? It should be impossible. After all, the plates inside a capacitor don't touch each other.

I'll deal with the "how" question in a moment. The first step, as always, is to see for yourself what happens.

Take a look at the components on the breadboard in Figure 9-9. Note that I have reverted to a 1,000µF capacitor. The lower button is still being used to discharge the capacitor, and the upper button charges the capacitor. The 100-ohm resistors are only there to protect the pushbuttons from too much current.

The important feature of this circuit is that the LED and its 1K series resistor have been moved down to connect with the lower plate of the capacitor, which isn't grounded anymore. Figure 9-10 shows this new circuit as a schematic.

Begin by pressing the lower button to make sure that both plates of the capacitor are at the same potential. Now release the lower button, and hold down the upper button. You will see the LED flash and then slowly fade away, while you keep your finger on the button.

Release the upper button and press it again. This time, nothing happens. Apparently the capacitor has to begin with its plates zeroed out, for the experiment to work.

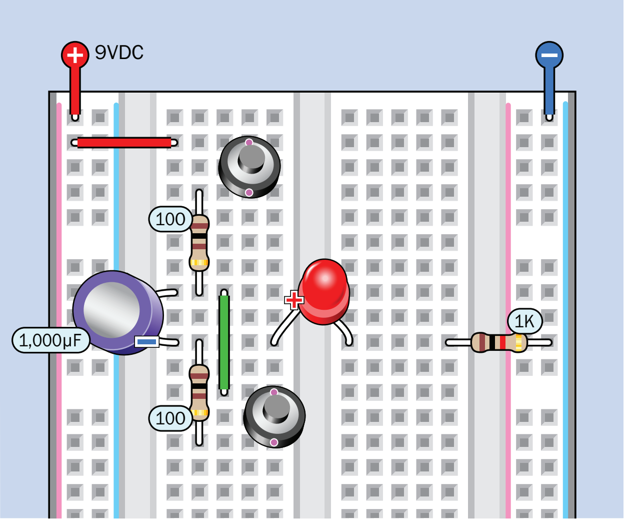


Figure 9-9. A circuit to demonstrate capacitive coupling.

Press the lower button again to discharge the capacitor, then press the upper button again, and the LED flashes again.

Where did the current come from, to make the LED flash? There's only one place. It must have come through the upper button, and down through the capacitor—somehow.

I tried this while monitoring the upper and lower plates of the capacitor using an oscilloscope, which can display very rapid changes in voltage. The curves looked look like those in Figure 9-11. I kept my finger on the upper button in the circuit for a couple of seconds. The upper plate charged very rapidly, as there is only a 100-ohm resistor to moderate this process. But a voltage also appeared on the lower plate, almost simultaneously, before gradually draining away through the LED.

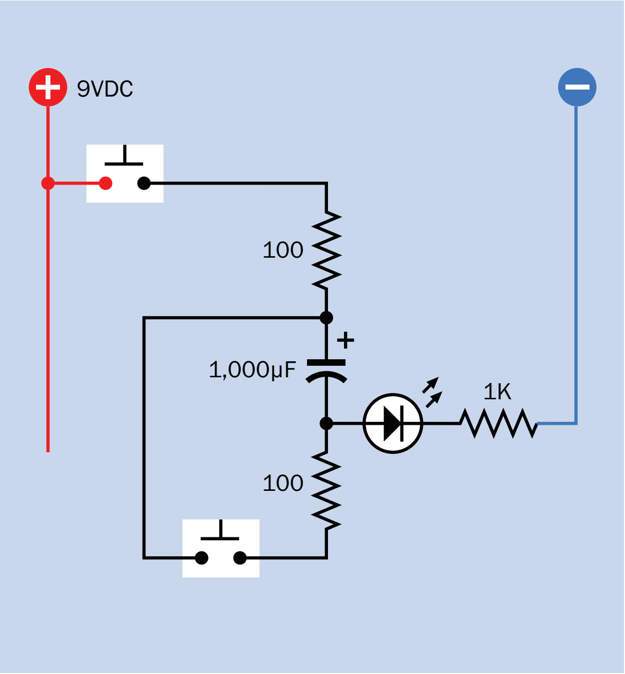


Figure 9-10. A schematic version of the circuit in Figure 9-9.

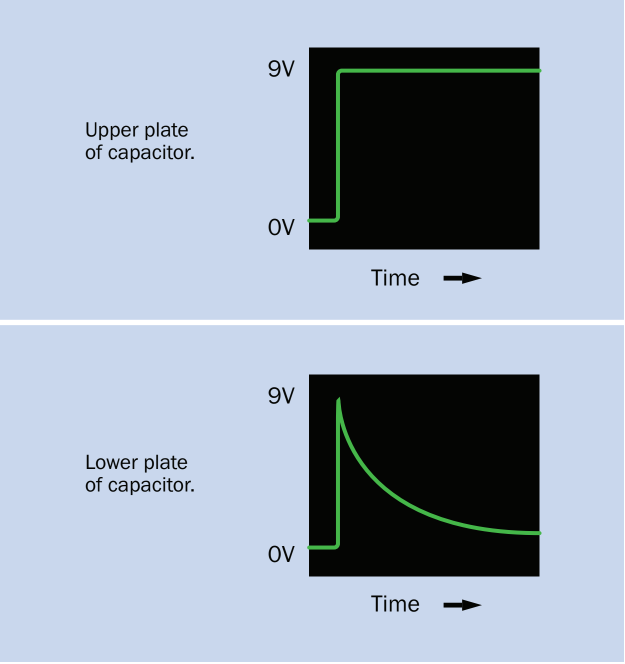


Figure 9-11. The surge in charge when a pulse passes through a capacitor and then dissipates through a load to negative ground.

In most educational materials, you will find a statement something like this:

- A capacitor blocks DC but allows AC (alternating current) to pass through.

Everyone agrees that this happens, and my little demo proves that a quick pulse does get through. If you created a rapid series of pulses, as in an AC source, they would all get through. In fact, this principle is used in billions, or perhaps trillions, of electronic devices.

The trouble is, knowing that it happens does not help me to understand how it happens.

A famous early experimenter named James Maxwell felt that the phenomenon shouldn't happen, but he saw that it did happen, so he came up with a theory to explain why it might happen. He called the pulse that passes through a capacitor ***displacement current***. In other words, the current is displaced from one side of the capacitor to the other side.

If you read up on displacement current, you may find statements like this:

"Few topics in modern physics have caused as much confusion and misunderstanding as that of displacement current. This is in part due to the fact that Maxwell used a sea of molecular vortices in his derivation, while modern textbooks operate on the basis that displacement current can exist in free space."

Molecular Vortices?

This goes a bit beyond the usual limits of an intro-level guide to electronics, but the discussion quickly becomes even more challenging, getting into some serious equations. Maybe this is why most writers tend to avoid the concept. It's not at all simple.

All you really need to know, though, is very simply summarized. Displacement current exists! A sudden change in voltage on one plate in a capacitor induces an equal change in voltage on the other plate, as if it is reacting in sympathy.

Next I will suggest a couple of ways in which this effect is useful.

Practical Coupling

Capacitive coupling can convert a long pulse into a short pulse when someone presses a button. In Figure 9-12, there's a device which needs to be triggered briefly. The 10K resistor on the right is called a ***pulldown resistor*** which holds the input to the device at around 0V. Then someone holds down the button, and a quick pulse of current gets through the capacitor. This direct connection raises the voltage on the input of the device for a moment, overwhelming the effect of the 10K resistor until the charge on the capacitor fades away and the circuit goes back to its original state. Notice that the person pressing the button can continue to hold it down, but the capacitor only allows that first pulse to go through.

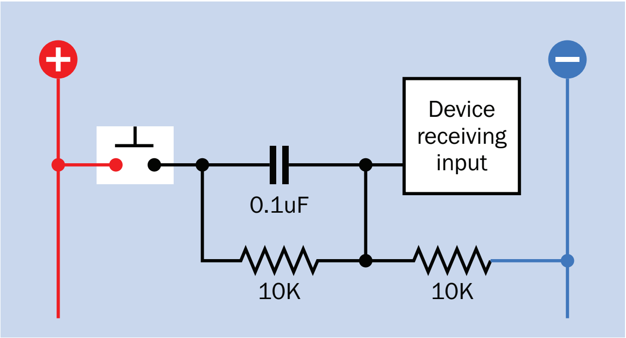


Figure 9-12. Using a coupling capacitor with a pulldown resistor to create a clean input.

The length of the pulse will depend on how big a capacitor you use. A 0.1µF capacitor would work in this application. The 10K resistor on the left, in the schematic, makes sure that the capacitor begins with an equal charge on its plates.

When you use a capacitor in this way, it's called a coupling capacitor, because it couples one section of a circuit with another section.

Maybe the device needing an input is designed to receive a negative pulse instead of a positive pulse. In that case, you could reverse the power supply. The pulldown resistor would become a pullup resistor, and the circuit would overcome the positive voltage on the device input with a negative pulse. You'll see me using this technique in several of the projects in this book.

Here's another common use for a coupling capacitor: It can connect an audio source with an amplifier. In figure 9-13, the audio source fluctuates between 8V and 9V, and the space between 8V and 0V is called an ***offset*** above 0V. You want to amplify the signal, but if you connect it directly to the amplifier, you will amplify the offset as well as the signal. If your amplifier multiplies voltages by 10, you'll end up with a range of 80 to 90 volts, which is not helpful.

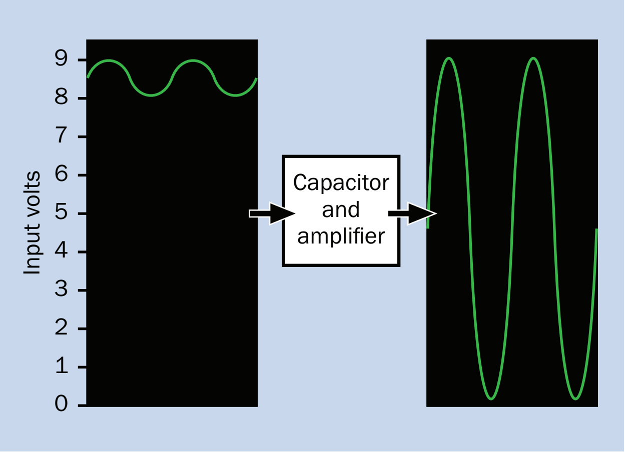


Figure 9-13. Removing the offset from an audio signal.

In principle, if you pass the signal through a capacitor of appropriate size, it blocks the offset and allows only the variations to get through. Now if you amplify them, and add a couple of additional components, you get a range of maybe 0 to 9 volts.

What's Next: Active Components

I went into a lot of detail about capacitors, because they're so important and versatile. I don't expect you to remember it all, but you can refer back to this section when I start applying the concepts later.

Resistors and capacitors are known as ***passive components***. They don't switch anything on or off, they don't amplify anything, and they function without needing their own separate power supply. They just modify the voltage and current passing through a circuit.

Now I'm going to introduce ***active components***, the first and most important being the transistor.

Experiment 10
Transistor Switching

At this point in human history, transistors have become indispensable in our daily lives. They have opened up endless possibilities.

You Will Need:

- Breadboard, hookup wire, wire cutters, wire strippers, multimeter, as before.
- 9V battery and optional connector (1).
- Transistor, 2N3904 (1).
- Resistors: 100 ohms (1), 470 ohms (2), 1K (1), 33K (1), 100K (1), 330K (1).
- Capacitor: 1μF.
- Generic red LED (1).
- Tactile switch (2).
- SPDT slide switch (1).
- 9VDC relay (as before).
- Trimmer: 10K (1).

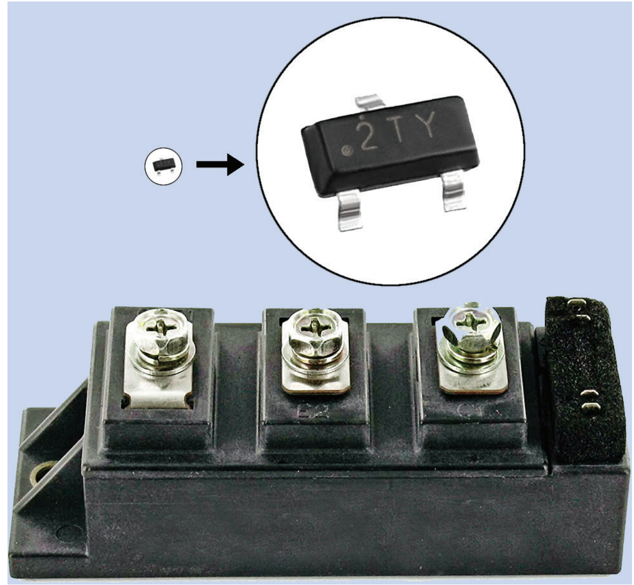


Figure 10-1. A surface-mount transistor, top left, and a power transistor, bottom, shown approximately actual size (in the white circle, the surface-mount transistor is magnified 10 times).

A Solid-State Switch

Figure 10-1 shows two transistors, both reproduced approximately actual-size. The one at top-left is a surface-mount component measuring about 1.5mm x 3mm, about as big as a grain of rice. It is magnified in the blue circle, so that you can see what it looks like. This transistor sells for a few cents and is designed to function with a 3.3V power supply. The one below is about 25 times as long and is rated for up to 100A at 600V.

When transistors were first developed in 1948, I doubt that anyone could have imagined that they would fulfill such a diverse range of purposes. The world today could not function without them.

Throughout this book I'll be using a transistor of the type shown in Figure 10-2. It is widely used in small low-voltage circuits, is rated for up to 40V, and is able to pass up to 200mA. Its part number is 2N3904, and the black plastic capsule, which is round at the back and flat at the front, measures approximately 0.25" (6mm) in each dimension.

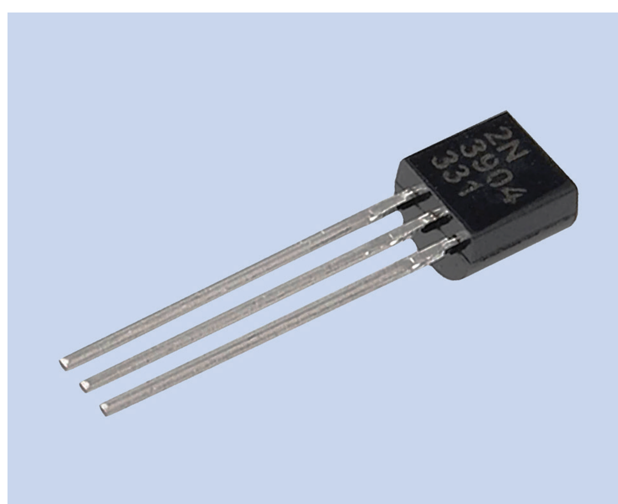


Figure 10-2. A 2N3904 transistor.

You'll need to know the names of the three leads of this transistor, as shown in Figure 10-3. The version sold in a miniature aluminum can is not very common now, but I have included it in case you run across it from a supplier of surplus components. If you turn the can so that its little tab is sticking out as shown, the leads have the same functions as the black plastic version when it has its flat side facing toward the right.

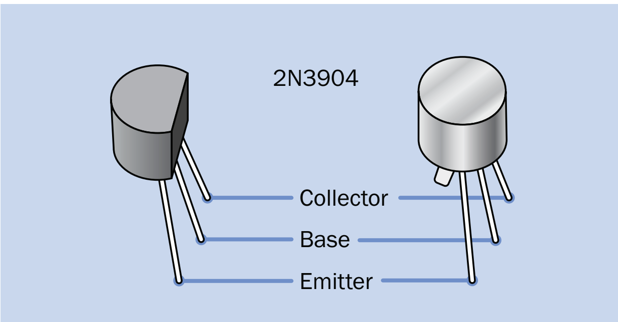


Figure 10-3. Identifying the leads of a 2N3904 transistor.

- A transistor is sensitive to polarity. You can damage it permanently if you use it the wrong way around.

A transistor can function like a relay in some respects, and the best way to understand this is by trying it. Assemble the circuit in Figure 10-4, being careful to distinguish between the two resistors, one of which is 100K (brown-black-yellow) while the other is 100 ohms (brown-black-brown). If you're concerned that the 100-ohm resistor isn't a high enough value to protect the LED, don't worry—the transistor won't allow enough current to damage it.

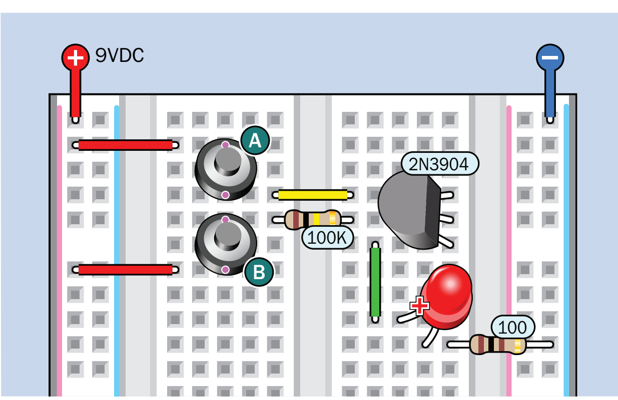


Figure 10-4. Transistor test, part 1.

Button A connects directly to the collector of the transistor, through the yellow jumper and strips inside the breadboard. Button B connects through the 100K resistor to the base of the transistor. The green jumper connects the emitter of the transistor through the LED and the 100-ohm resistor to negative ground.

Press Button B, and you should see a dim glow in the LED. Let go of that button and press button A, and you see nothing at all.

Continue to hold down Button A. At the same time, press and release Button B. You will find that Button B is controlling the current that enters the collector of the transistor and exits through the emitter. In Figure 10-5 I have illustrated this basic concept.

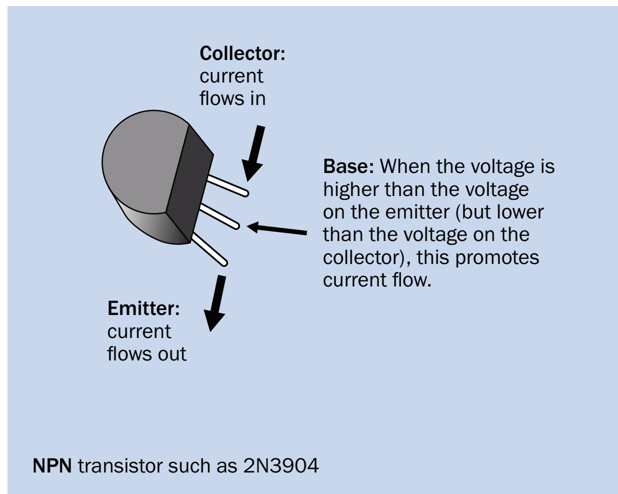


Figure 10-5. How an NPN transistor controls current flow.

This test leads to an important conclusion:

- Because the 100K resistor has such a high value, you can be sure that very little current is getting into the transistor through its center lead.
- Therefore, a tiny current is switching a much larger current which enters through the top lead.

In Figure 10-6 I have shown some variants of the schematic symbol for this transistor, which is known as an NPN type, for reasons that I will explain in a moment. When the 2N3904 is oriented with its flat side facing to

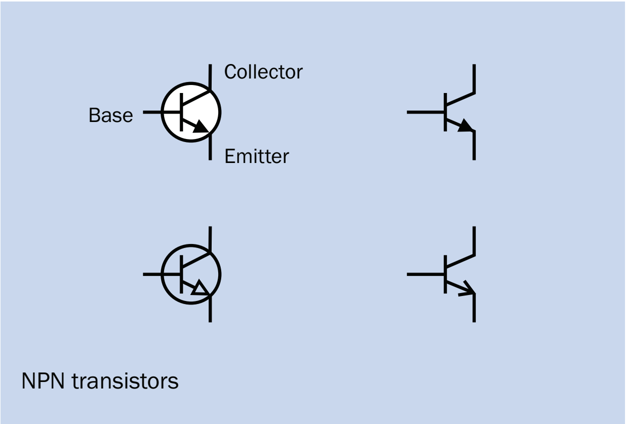


Figure 10-6. Some schematic symbols for an NPN transistor.

the right (seen from above), it matches the schematic symbol as I have shown it. In schematics you may find transistors with or without circles around them, and they may be oriented in any direction, for the convenience of the person who drew the circuit. Always be careful to notice which way the arrow is pointing in the NPN transistor symbol, so that you don't get the collector and the emitter mixed up.

- The arrow shows the direction of conventional current (positive-to-negative) through an NPN transistor.

In Figure 10-7 I have drawn a schematic version of the circuit that you just tested.

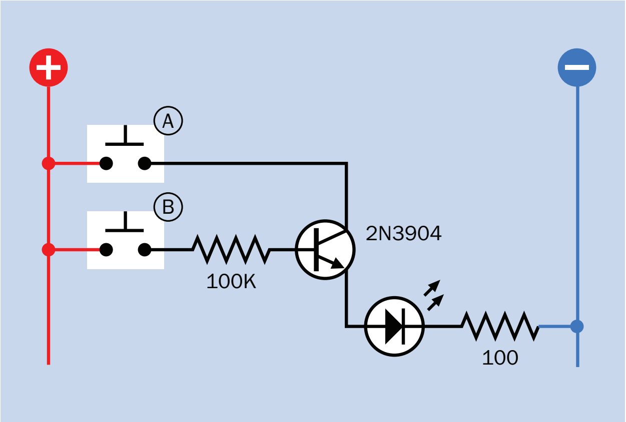


Figure 10-7. Schematic version of the circuit in Figure 10-4.

How It Works

Inside the 2N3904 is a sandwich of two types of silicon with opposite polarity. One type, known as the ***P layer***, has a surplus of ***positive charge carriers***—which actually means a lack of electrons, allowing room for new electrons to find a home. It is sandwiched between two ***N layers*** that have a surplus of electrons. This is illustrated in Figure 10-8. The voltage applied to the base (the center layer), relative to the emitter, is known as ***bias***.

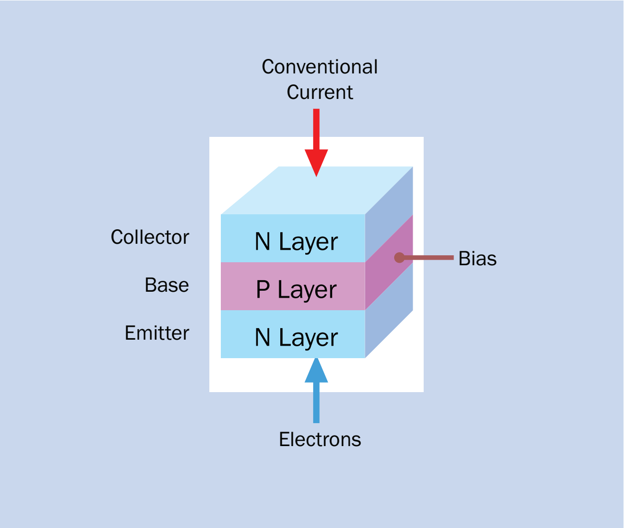


Figure 10-8. Inside an NPN transistor.

Electrons are unable to enter through the bottom N layer until they are attracted by positive bias applied to the P layer. This lures them in, and now that they are energized, they continue through the upper N layer, which has a positive voltage applied externally Remember, conventional current flows in the opposite direction to electrons, which is why the upper layer is known as the collector and the lower layer is the emitter, and is why the NPN transistor symbol has an arrow pointing outward.

- In all schematic symbols, an arrow indicates the flow of conventional current. (The two arrows added to an LED symbol indicate light, not electricity.)

PNP transistors behave oppositely to NPN transistors, as they have two P layers with an N layer between them. The symbol variants are shown in Figure 10-9 on the next page.

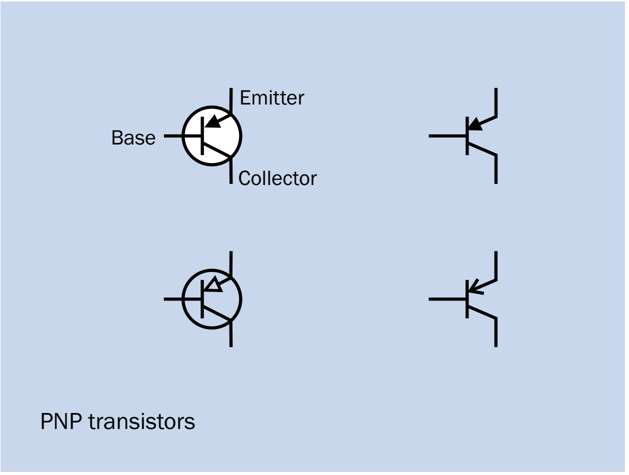


Figure 10-9. Schematic symbols for PNP transistors.

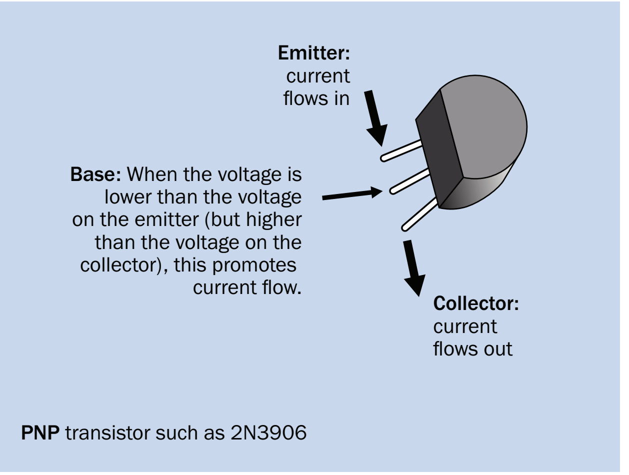


Figure 10-10. How a PNP transistor controls current flow.

A PNP is somewhat confusing (at least, I find it confusing) because it is like an opposite version of an NPN. The emitter now has an arrow pointing inward. The arrow still shows the direction of conventional current, but now the transistor switches on when the voltage on the base is low compared with the emitter, instead of high relative to the emitter.

- NPN transistor: Increasing the base voltage promotes the flow of current from collector to emitter.

- PNP transistor: Reducing the base voltage promotes the flow of current from emitter to collector.

It's easy to get the NPN and PNP symbols mixed up, but there's a simple way to remember which is which:

- The arrow in an NPN symbol is "Never Pointing iN."

Because every aspect of a PNP transistor is opposite to an NPN transistor, you should not be surprised that the current flows through it as shown in Figure 10-10.

All NPN and the PNP transistors are *bipolar junction transistors* (sometimes abbreviated as *BJT*) because they all contain two junctions between three layers. Many other types of transistors exist, which I won't have space to describe in this book.

- My diagrams have shown the names of pins for 2N3904 and 2N3906 transistors, relative to the flat side of each one. Other transistors have their pins organized differently. Check the datasheets for details.

You may wonder, why are two opposite types of transistor necessary? Because in some circuits, PNP transistors are convenient. NPN transistors are used more commonly, though, and none of the projects in this book will use PNPs.

Your experiment showed you how a positive voltage on the base of an NPN bipolar transistor can control current flow. But what happens if the base voltage varies just a little? It's time for another test, in which you will see that a transistor can *amplify* variations in current entering through its base.

Amplification

First set your meter to measure mA, and plug the red lead into the appropriate socket on the meter, if necessary. If you have a manual-ranging meter, choose the lowest range for mA.

To find out how much current is flowing in through the collector of your 2N3904 transistor, set up your circuit as in Figure 10-11. Pull one end of the red jumper out of the board, and one end of the yellow jumper, and connect the meter between them. Now try three different resistors, one at a time, for the "mystery resistor" shown as R1. You will need one resistor of 33K, one of 100K, and one of 330K. Insert them one-at-a-time, press Button B each time, and write down the current that you measure. (You don't need to press Button A, because you are bypassing it with the meter.)

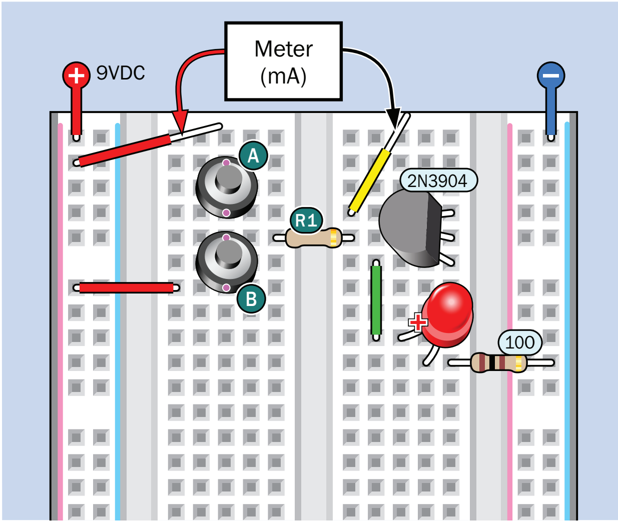


Figure 10-11. Measuring current flowing in through the collector of a transistor.

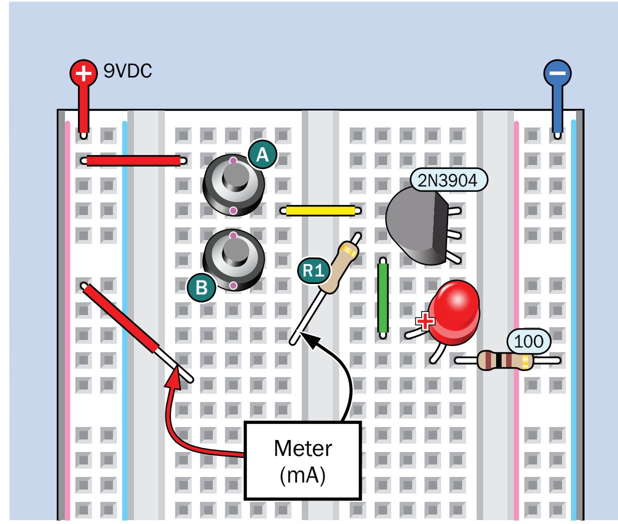


Figure 10-12. Measuring current flowing into the base of the transistor.

Now I'd like you to find out how much current flows in through the base of the transistor in each case. To do this, re-insert the red and yellow jumpers, and put your meter between the other red jumper and the resistor, as in Figure 10-12. Try each of the resistors in turn (33K, 100K, and 330K), and press Button A with each of them, so that current flows in through the collector of the transistor. (You don't need to press Button B, now, because you are bypassing it with the meter.)

What can you figure out from your data gathering? You can make a table like the one in Figure 10-13.

If you're wondering about the third column, labeled "beta," I created it using my calculator. I divided the collector current by the base current in each case, and I will ask you to do the same using your values.

- *Beta value* = collector current / base current
- The beta value is the *amplification* factor of the transistor.

For each resistor, the beta value seems to be around 190. The important lesson here is that this amplification factor stays about the same regardless of how much current is flowing into the base—at least, within the three values that you used. We can say that a transistor has a *linear response*, which is important for applications such as amplifying music.

The maximum collector current that you tried was about 20mA when using a 33K base resistor. If you had lowered the value of the base resistor so that the transistor would pass more current, you would have found that the beta value stayed almost constant up to the limit of the transistor, which is 200mA—but that would have killed your LED, and delivering 200mA would be a challenge for a 9V battery.

Base resistor	Base current (mA)	Collector current (mA)	Beta value
33K	0.12	22.0	183
100K	0.056	10.6	189
330K	0.020	3.8	190

Figure 10-13. Experimental values from your transistor test.

Now for the fun part. Simplify the circuit as in Figure 10-14, on the next page, and press your finger to the exposed red and yellow jumpers while you watch the LED. If nothing happens, moisten your finger and try again. The harder you press, the brighter the LED becomes. The transistor is amplifying the tiny amount of current flowing through your finger. (Be careful not to allow the red and yellow jumpers to make direct contact with each other. The transistor might not be happy about that.)

This fingertip switching demo is safe if the electricity ***only passes through your finger.*** You won't feel it, because it's 9 volts DC from a small battery. Please don't ever use a higher voltage in this experiment, and don't apply electricity between two hands, because this would allow the current to pass through your body. There is no chance of hurting yourself in this circuit, because the current is so small, but you should never get into the habit of allowing electricity to run through you from one hand to the other. Also, when touching the wires, don't allow them to penetrate your skin.

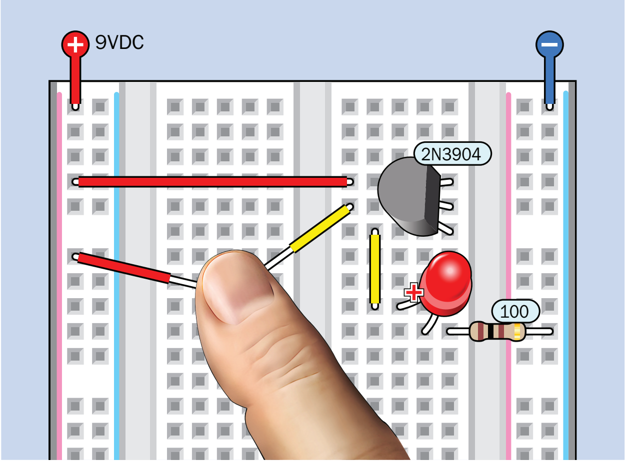


Figure 10-14. The finger test.

A couple of basic facts about amplification:

A bipolar transistor has an ***effective resistance*** to the current flowing through it. The word "effective" means that it isn't like a simple resistance which always stays the same. It changes, depending on the bias. When the bias is low, the effective resistance is so high, it prevents almost all current from flowing between the collector and the emitter. The effective resistance drops while the transistor starts to amplify the base current. Finally, the transistor is said to be ***saturated***, because its effective resistance has gone as low as it can go, and the transistor has reached the upper limit of its ability to pass current.

- When the transistor is passing as much current as it can, it is said to be saturated.
- The voltage at the emitter will always be lower than whatever voltage you apply to the collector. Think of the difference as being the payment that you must make to the transistor for its amplifying services.
- Because the voltage out is less than the voltage in, a bipolar junction transistor does not amplify voltage. It amplifies current.
- The base must be at least 0.7V "more positive" than the emitter, to enable current to pass through.

Important Facts

Before you use a transistor that you haven't encountered before, go online and check its datasheet. Remember that many transistors don't have the same sequence of emitter, collector, and base leads as the 2N3904. Also check the datasheet for the maximum current that the transistor will tolerate.

- If you connect a transistor the wrong way around, it will seem to work, although not very well. Then it is likely to stop working almost completely, and the damage may be permanent. (Guess how I know this.)
- Never apply a power supply directly between any two pins of a transistor. You can burn it out that way. Always limit the current flowing through a transistor by using another component such as a resistor, in the same way that you would protect an LED.

To find out if a transistor is still working, remove it from the breadboard and poke its leads into the little holes on your meter which are provided for this purpose and are labeled C, E, and B, meaning Collector, Emitter, and Base. Turn the dial of your meter to the testing position, which is usually labeled ***hFE***. This is an abbreviation for hybrid parameter forward current gain, common emitter, but you don't have to remember that. Just remember hFE.

Push the transistor in firmly, and you should see a number on the display of your meter which is the beta value. If you are testing the transistor the wrong way around, or if it is damaged, the meter reading will be unstable, or blank, or zero, or much lower than it should be (almost always below 50, and usually below 5).

Many more types of transistors exist—especially ***MOSFETs*** (metal-oxide semiconductor field-effect transistors), which are more efficient than bipolar transistors—but the bipolar family is the oldest. It is also least vulnerable to accidents such as too much voltage or a static discharge, although you may still be able to damage it if you are careless.

Transistor or Relay?

I'll finish this introduction by comparing a transistor with a relay. A relay cannot amplify a range of current, like a transistor, but it can switch current, so when you have a switching application, which should you use? It depends on the circumstances, summarized in Figure 10-15.

	Transistor	Relay
Long-term reliability	Excellent	Limited
Can switch in DP or DT mode	No	Yes
Can switch large currents	Limited	Yes
Can switch alternating current	Usually not	Yes
Triggerable by alternating current	Usually not	Optional
Suitable for miniaturization	Excellent	Very limited
Able to switch at very high speed	Yes	No
Price advantage for high voltage/current	No	Yes
Price advantage for low voltage/current	Yes	No
Current leakage when nonconducting	Yes	No

Figure 10-15. Comparing the characteristics of transistors and relays.

One limitation of NPN and PNP transistors is that they subtract some power to provide a service. When a relay is in its "off" position, it uses no current at all, and when it is "on" the current emerging from it is the same as the current going into it. If you use a latching relay, it doesn't take any current in either of its two positions; it just needs a pulse to flip it from one to the other.

Relays offer more switching options. Different versions can be normally-open, normally-closed, or latching. A relay can contain a double-throw switch, which gives you a choice of two "on" positions, and can be double-pole, allowing two separate circuits. Single-transistor devices cannot provide the same versatility, although you can use multiple transistors to emulate this behavior.

Transistor Origins

Although some historians trace the origins of the transistor back to the invention of diodes (which allow electricity to flow in one direction while stopping it from flowing in the opposite direction), the first practical and fully functional transistor was developed at Bell Laboratories in 1948 by John Bardeen, William Shockley, and Walter Brattain, pictured in Figure 10-16.

Shockley was the leader of the team, who had the foresight to see how potentially important a solid-state switch could be. Bardeen was the theorist, and Brattain actually made it work. This was a hugely productive collaboration—until it succeeded. At that point, Shockley started maneuvering to have the transistor patented exclusively under his own name. When he notified his collaborators, they were unhappy about this.

Figure 10-16. Front, William Shockley. Rear, John Bardeen. Right, Walter Brattain. For their collaboration in development of the world's first working transistor in 1948, they shared a Nobel prize in 1956.

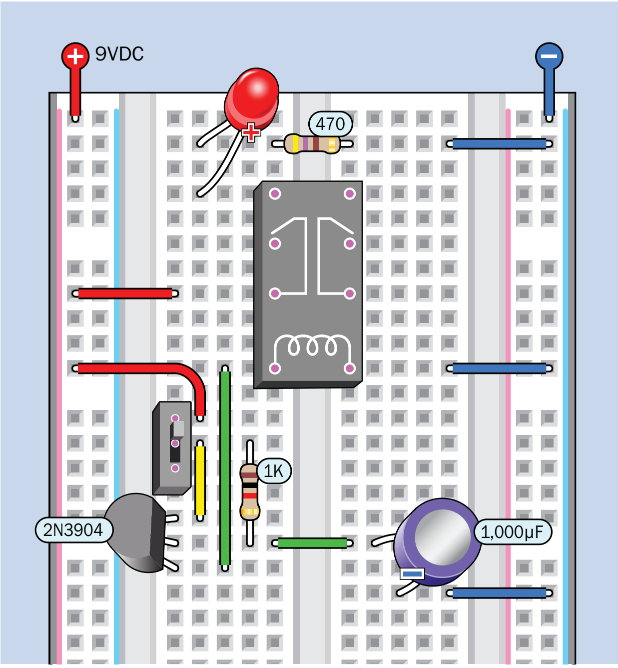


Figure 10-17. A basic timer circuit.

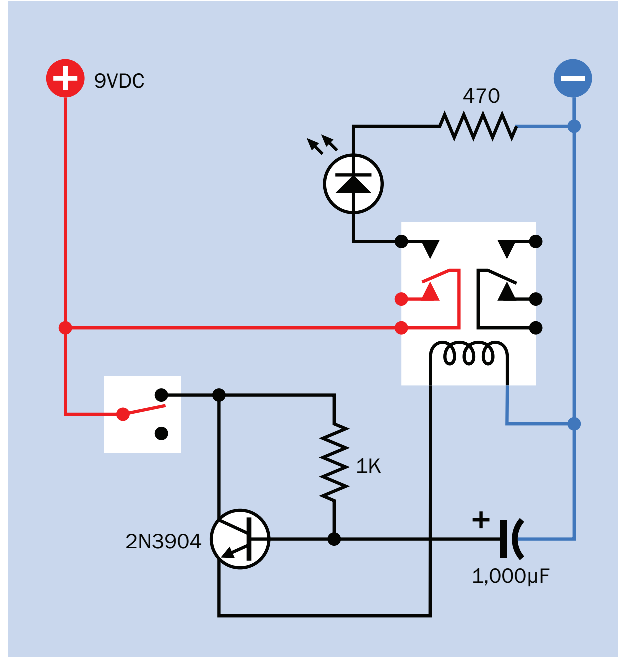


Figure 10-18. The schematic version of Figure 10-17.

The widely circulated publicity photograph didn't help, as it showed Shockley sitting at the center in front of a microscope, as if he had done the hands-on work, while the other two stood behind him. A copy of this picture appeared on the cover of *Electronics* magazine. In reality, Shockley was seldom present in the laboratory where the real work was done.

The productive collaboration quickly disintegrated. Brattain asked to be transferred to a different lab at AT&T. Bardeen moved to the University of Illinois to pursue theoretical physics. Shockley eventually left Bell Labs and founded Shockley Semiconductor in what was later to become Silicon Valley, but his ambitions outstripped the capabilities of the technology in his time. His company never manufactured a profitable product.

Eight of Shockley's coworkers in his company eventually left and established their own business, Fairchild Semiconductor, which became hugely successful as a manufacturer of transistors and (later) integrated circuit chips.

In 2016, Fairchild was acquired by On Semiconductor for $2.4 billion.

The Simplest Timer

I promised to include a timer circuit using only the components that you have learned about so far, and now, here it is. The breadboarded version is shown in Figure 10-17. Position the components carefully, because several of them are closely spaced. Start with the two-position switch with its actuator toward the bottom of the board, and then push it upward. After about one second, the LED lights up.

Only one second? That's not much of a timer!

True, but I will show you how to make it adjustable. First let's see how it works, and the schematic in Figure 10-18 is the easiest way to figure it out.

The same relay that you used in Experiment 7 is wired so that it switches on an LED when it receives power through its coil. So, to make this circuit work, I simply needed to have a capacitor that would charge upward from 0VDC, and when the capacitor reaches a high enough voltage, it triggers the relay. Simple, eh?

Well, not quite. If I wired the capacitor directly with the relay, the capacitor would be discharging itself through

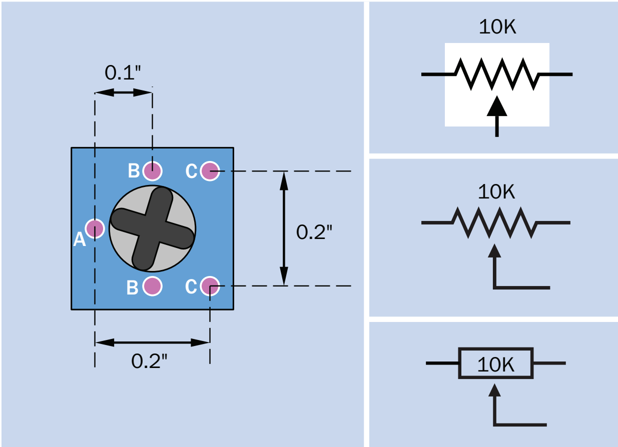


Figure 10-19. The pins underneath your trimmer must be in the locations shown by the red circles, either for the pins labeled B, or the pins labeled C. Three schematics representing any type of trimmer are shown at right, with an arbitrary resistance value.

the relay coil at the same time that it was charging through the resistor, and it would never reach a high enough voltage to trigger the relay. Remember, the coil inside the relay conducts electricity even before it moves the contacts.

I couldn't take much current out of the capacitor while it was charging. So—maybe I needed a current amplifier. That would be a transistor, wouldn't it?

When the switch on the left in Figure 10-18 is moved up, it feeds the capacitor through the 1K resistor. The capacitor is also wired to the base of the transistor, so the base acquires the same voltage as the capacitor.

Power from the switch is also applied to the collector of the transistor, and current passes out through the emitter to the relay coil. The transistor will not be overloaded, because the coil has some resistance.

The capacitor charges, the transistor amplifies, the output goes to the coil—and when the output from the transistor reaches about 6V, the relay switches on the LED. Using the values that I chose, the delay just happens to be about 1 second.

Now, how can this circuit be more useful? I need a component that has a variable resistance, to adjust the time which the capacitor will take to reach its necessary voltage. That would be a ***potentiometer***, and the type I need will plug into a breadboard. That would be a ***trimmer***.

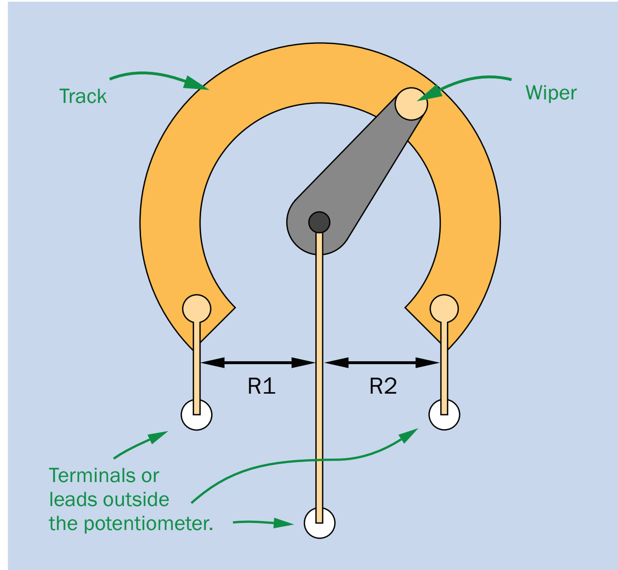


Figure 10-20. Inside any type of potentiometer. The total resistance is R1 + R2.

Trimming It

If you turn back all the way to Figure 6-23, you will see some trimmers. In Figure 10-19 you'll see a diagram of a trimmer viewed from above, with pink circles showing the positions of pins underneath it. There will be three pins, but their positions may vary. You can use a trimmer with Pin A and the two pins labeled B, or a trimmer with Pin A and the two pins labeled C.

The distances between the pins are important, so that the trimmer will plug into your breadboard. Many other types of trimmers exist, but if you are buying your own, please see the exact specifications in Appendix A.

Also in Figure 10-19, I have included three versions of the schematic symbol that is used for any type of trimmer or potentiometer.

Remember, a trimmer is a miniature potentiometer. I'm not including any full-sized potentiometers in this edition of ***Make: Electronics***, because they are not used much anymore. You can look them up online if you are interested.

Inside any potentiometer is a circular ***track*** of a resistive compound, and a ***wiper*** that touches it, as shown in Figure 10-20. The wiper can rotate around the track, and as it does so, the resistances between the wiper and

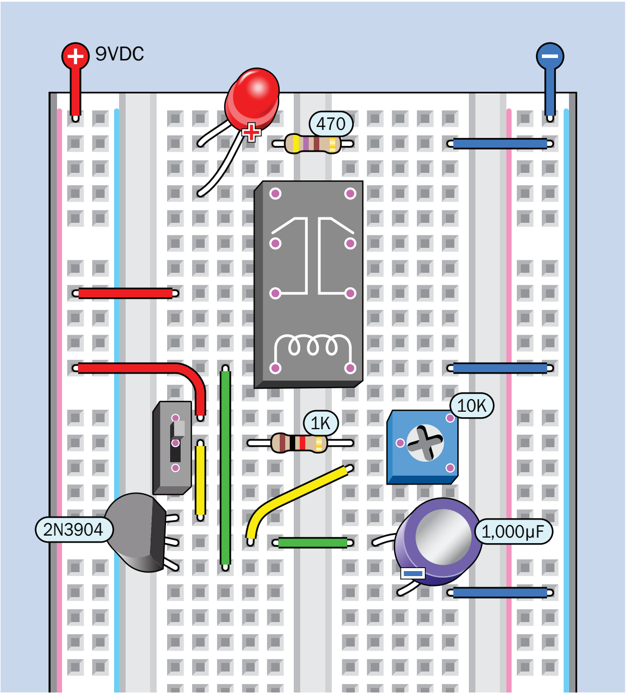


Figure 10-21. The timer circuit incorporating a trimmer, so that it is now adjustable.

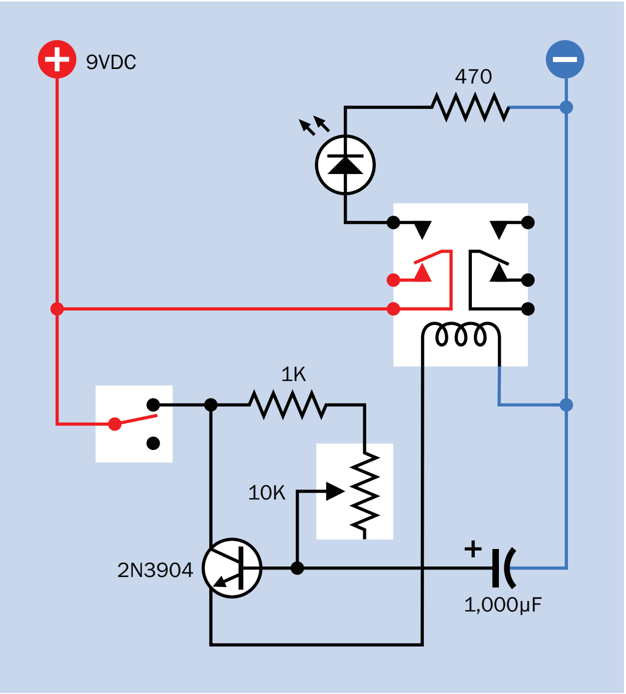


Figure 10-22. Schematic of the adjustable timer circuit from Figure 10-21..

the ends of the track will vary. If R1 is the resistance between the wiper and the left end of the track, and R2 is the resistance between the wiper and the right end of the track, the potentiometer will be sold as having a rating of R1 + R2.

For the timer experiment, I'm suggesting a 10K trimmer so that you can measure any interval of up to 11 seconds. Will a 25K trimmer work if you want to measure a longer period? Probably, but when a capacitor takes longer to charge, it also has more time to allow internal leakage of current, and it becomes less accurate.

Figure 10-21 shows the trimmer added to the breadboard. I'm suggesting a trimmer that has a pair of pins in the C positions in Figure 10-19, but a trimmer with pins in the B positions would work just as well, because they would still connect with the same strips inside the breadboard—so long as Pin A is oriented as shown.

Figure 10-22 shows the schematic version of the circuit. You'll see I added the trimmer to the 1K resistor, so that now the resistance between the positive power supply and base of the transistor can vary from 1K to 11K.

If you want to make a permanent version of your timer, I suggest you buy a full-size potentiometer and mount it in a little box. Run it from an AC adapter with a 9VDC output, and test it with various positions of the knob that you mount on the shaft of the potentiometer. Make a semicircular paper dial, and use an accurate clock or app to time each run. Eventually you should end up with a dial marked from 1 to 10 seconds, or maybe 1 to 20 seconds, or—I don't know how far you can extend the range.

Experiment 11
Light and Sound

Just about every introductory book about electronics includes a flasher circuit. Making an LED blink isn't particularly thrilling, but really there are good reasons for including it: The output is immediate, and a humble flasher can lead to greater things.

What things, exactly? Well, in this experiment, you'll progress from the flashing LED to an audio output that sounds like a car alarm. And after that, would you believe that you can tweak it to emulate bird song?

While writing this third edition of *Make: Electronics*, my only question was *which type* of flasher circuit to include.

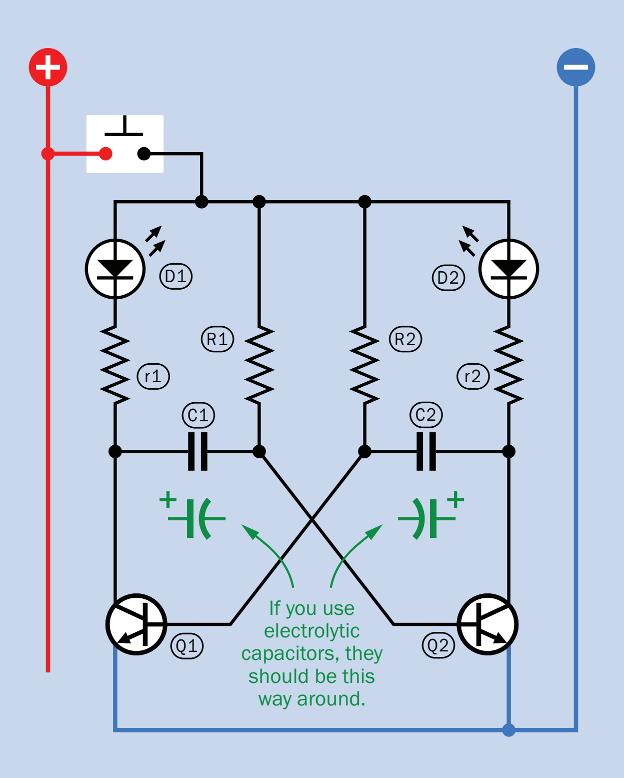


Figure 11-1. A popular astable multivibrator circuit, which looks simple but is often misunderstood.

I wanted one that would be as simple as possible to build and understand, but when I dug deeper into various circuits I found that some were temperamental and didn't always start flashing, while others used inductors, which I don't want to explain until Section Five. A few used a programmable unijunction transistor, which is almost obsolete and has an uncertain future, and many used chips which I don't want to deal with until Section Four.

In the end I came back to the same circuit that I used in the Second Edition, because it is reliable and versatile. Some readers felt that it was difficult to understand, so I have added a lot more explanation this time around.

You Will Need:

- Breadboard, hookup wire, wire cutters, wire strippers, multimeter, as before.
- Transistor: 2N3904 (6).
- 9V battery and optional connector (1).
- Resistors: 100 ohms (1), 470 ohms (1), 1K (1), 4.7K (4), 10K (1), 47K (2), 470K (2).
- Capacitors: 10nF (3), 1μF (2, ceramic preferred), 47μF (2, optional).
- Generic LED (2).
- Tactile switch (2).
- Small speaker, 8-ohm (1).

The Misunderstood Multivibrator

A flasher circuit is sometimes called an *oscillator*, although that isn't quite accurate, because the output from an oscillator should vary smoothly in a curve known as a *sine wave*.

A flasher with an output that flips on and off is properly known as an *astable multivibrator*. That's a mouthful, but you can see that it makes sense: The output is not stable, and it vibrates.

Try using this search term online:

```
transistor astable multivibrator
```

Browse through the images, and you'll see more than 100 versions of the circuit shown in Figure 11-1. Along with the schematics online, you will find descriptions of

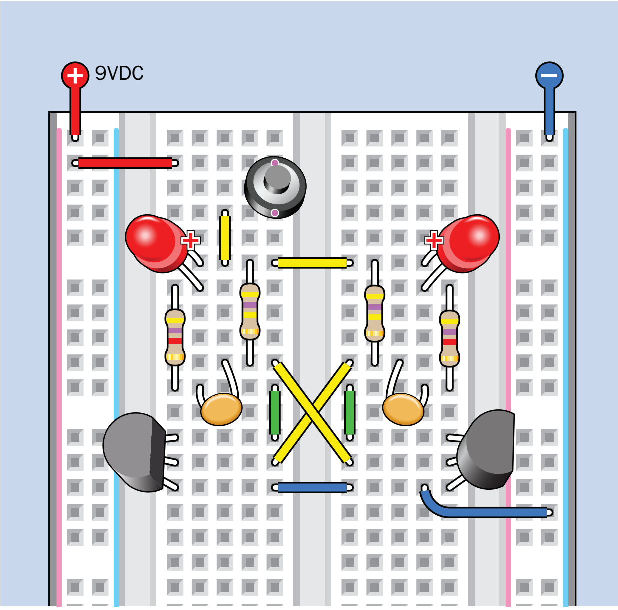


Figure 11-2. The breadboard layout for an astable multivibrator using LEDs for output.

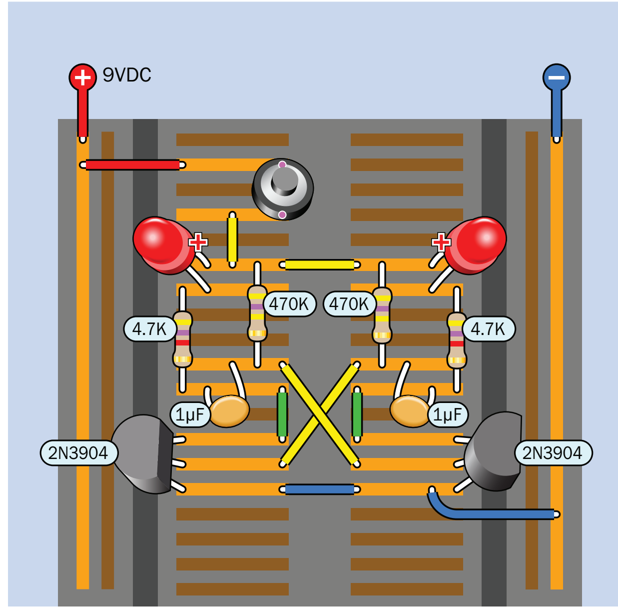


Figure 11-3. An x-ray version of the breadboard view from the version in Figure 11-2.

how the circuit works—but here's an odd thing. Few people mention displacement current, which is fundamental, as you will soon see. (Fortunately you already had a chance to observe it in Experiment 9.)

First, you need to make sure that the multivibrator circuit works. In Figure 11-1 I have labeled the components so that I can refer to them easily. In Figure 11-2 I have shown how to place them on a breadboard, and in Figure 11-3 you will find an x-ray view including the values of the components.

Take special care to position them precisely, and count holes in the breadboard to get everything exactly right.

Compare the breadboard layout with the schematic in Figure 11-1, and you should see that the connections are the same. I have used ceramic capacitors, but you can use electrolytic capacitors so long as you orient them correctly as shown in Figure 11-1. I'll explain more about using electrolytics in this circuit a bit later.

In Figure 11-3, notice that the components on the left side of the board have the same values as the components on the right, except they are arranged in a mirror-image pattern. In Figure 11-1 I used a lower-case r for r1 and r2 to remind you that these resistors have a relatively low value of 4.7K while R1 and R2 are 470K each. Don't get them mixed up!

Incidentally, when using labels, it's common to identify LEDs with letter D, as they are a type of diode. Transistors are identified with letter Q, probably because early transistors packaged in little aluminum cans had a tab sticking out which made them look like a Q.

When you hold down the button, the LEDs should take turns flashing, repeating at a rate slightly faster than once per second.

Because this circuit is a little more subtle than the others you have built so far, I will pause for a moment to give some advice on how to deal with the frustrating situation if components just sit there doing nothing. This happens to all of us, from time to time, but there are well-established coping strategies.

What to Do When It Doesn't Work

1. Try to be patient. The more annoyed and frustrated you get, the less able you will be to recognize a problem. Don't think of the circuit as an enemy that is trying to ruin your life. Think of it as your friend who just needs a bit of guidance.

2. Don't just sit there staring at it. Examine the components from all angles. Pick up the breadboard, hold it close to your eye, and look along each row of holes to check exactly where all the leads are inserted.

3. Use a magnifying lens of some kind. Personally I keep an extra pair of eyeglasses for extreme closeup work, and I also have a head-mounted magnifier of the type shown in Figure 12-11. Another option which may be helpful is to put the camera in your phone into closeup mode, and use the screen as a microscope display.

4. Always use a desk lamp. In fact, better still, use two, shining from opposite directions so that details don't get lost in shadow.

5. Scan or photograph the breadboard diagram in this book, and print a larger copy. If you print your own magnified version, you can mark it with a pen to check each component that you install.

6. Verify all values. If you mistake yellow for brown when squinting at the stripes on a resistor, you can end up using a value that is 1,000 times higher, or lower, than the one specified. Ideally you should check the value of each component with a meter immediately before you put it in the board. This is quicker than pulling components out to check them later.

Read the identifying numbers carefully on transistors, and use your meter to check them, too. You may have damaged a transistor without realizing it.

7. Check the current consumption. Set your meter to measure mA and insert it between the positive side of your battery and the positive bus on the breadboard. If a wiring error has created a short circuit, you'll see a high meter reading. This circuit should use less than 50mA.

8. Check voltages around the board. Anchor the black lead of your meter on the negative side of your power supply, using an alligator test lead. Then set your meter to measure DC voltage and touch the red probe all over the board while the power to the circuit is switched on. Any location with a near-zero voltage may indicate a bad connection. When using a battery, remember to check that it is still good.

9. Don't necessarily trust your breadboard. A cheap one may have unreliable clips inside it, especially if you have forced large-diameter wires into them previously.

10. If you still can't find an error, get up and walk away. Very often, if you take a break and then return, you'll notice an error immediately.

Perhaps all these recommendations seem a bit obvious—but almost everyone is liable to forget to do obvious things, once in a while.

Inside the Wires

To begin figuring out what's happening in this circuit, refer back to the schematic in Figure 11-1 and begin by asking yourself, under what circumstances will D1 light up? Current has to flow through it and then find its way to negative ground. It will pass through D1, then resistor r1, and then it has to pass through transistor Q1.

Under what circumstances will Q1 conduct current? When its base rises above 0.7V relative to its emitter, which is connected with negative ground.

Where does the base of Q1 get its voltage from? It is supplied via R2, but there's a snag: The bottom end of R2 is also connected with C2, and when you first switch on the power, C2 has no charge on it. So, current will rush into C2 first, until the voltage rises to 0.7V, which is what the base of Q1 requires. Then Q1 will start conducting, and D1 will light up.

Now, because this is a symmetrical circuit, the same thing is also happening on the other side of it. Since the components on both sides of the circuit have identical values, why don't the LEDs both light up together?

And—here's another question. What switches them off?

This requires a more detailed examination, so I will illustrate the action in five simplified steps, beginning with Step 0 on the next page. I'm calling it Step 0 because this is how I imagine the circuit in the moment after you apply the power but before it actually starts flashing the LEDs alternately.

I have used colors to indicate voltages in an unconventional way, because I'm hoping this will clarify the situation. Also I have colored each transistor green when it is passing current, leaving it transparent (background color) when it is in an "off" state.

I have labeled the plates inside the capacitors because I will need to refer to them. C1L is the left plate in capacitor C1, C1R is the right plate inside C1, and so on.

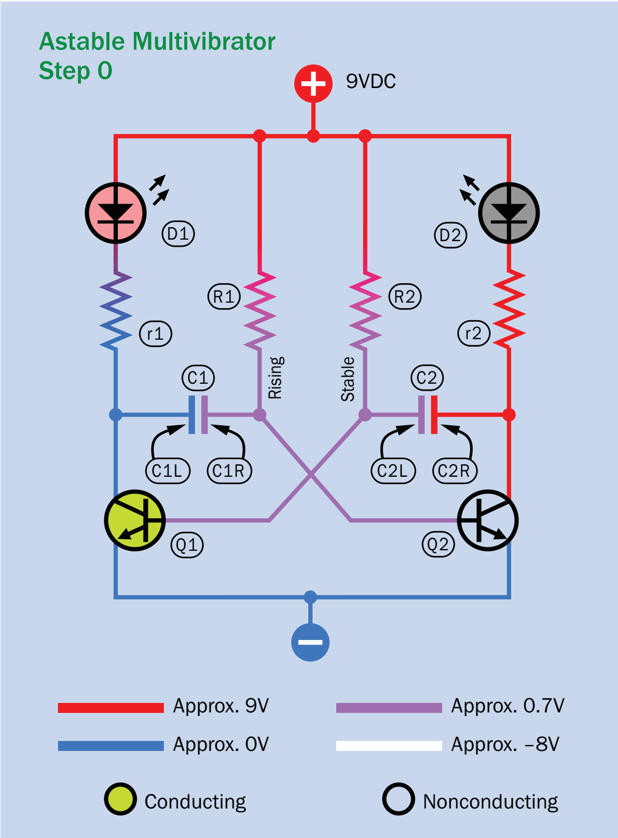


Figure 11-4. The state of the circuit when it is switched on.

In Step 0, C1R and C2L have had a moment to begin charging through R1 and R2, so the voltage on them is almost equal—around 0.7V. But, because of slight variations in the manufacturing process, no two components are exactly alike. One resistor will have a fractionally lower value than the other, one capacitor will charge fractionally faster than the other, and on this occasion I am imagining that C2L has reached a stable 0.7V just a moment ahead of C1R. This is why the connection with C2L is now "stable" while the voltage for C1R is "rising." As a result, Q1 has started passing current while Q2 is still blocking current. Now you see why one LED switches on before the other.

Because Q1 is conducting, the voltage on its collector is almost as low as the voltage on its emitter. Consequently any voltage on C1L has drained through Q1. D1 is glowing as current passes through it and r1, down through Q1 to negative ground. This is a stable situation.

In the right-hand half of the circuit, Q2 has not started conducting yet. Because the voltage on both sides of D2 is about equal, the diode remains dark.

So much for the setup. Now look at Step 1 in Figure 11-5. The voltage on the base of Q2 finally reached 0.7V, so this transistor started conducting. Consequently D2 is now glowing, but all of the voltage on C2R was dumped through Q2. When you change the voltage very quickly on one side of a capacitor, the other side changes by an almost equal amount—because of displacement current. This applies whether voltage rises suddenly (as you saw in Experiment 9) or when it drops suddenly. Here, C2R dropped from around 9V to near 0V. This caused C2L to drop by an equal amount, from around 0.7V to ***minus*** 8V.

But—how can we have a voltage below zero?

Easily! In your battery, 9V is the ***difference*** between the two sides, limited by chemical reactions. It's convenient to label the positive side 9V and the negative side 0V, but this just means there are more electrons on the 0V side. It is a ***potential difference***. It doesn't mean the 0V side of a battery is the "most negative" it can be.

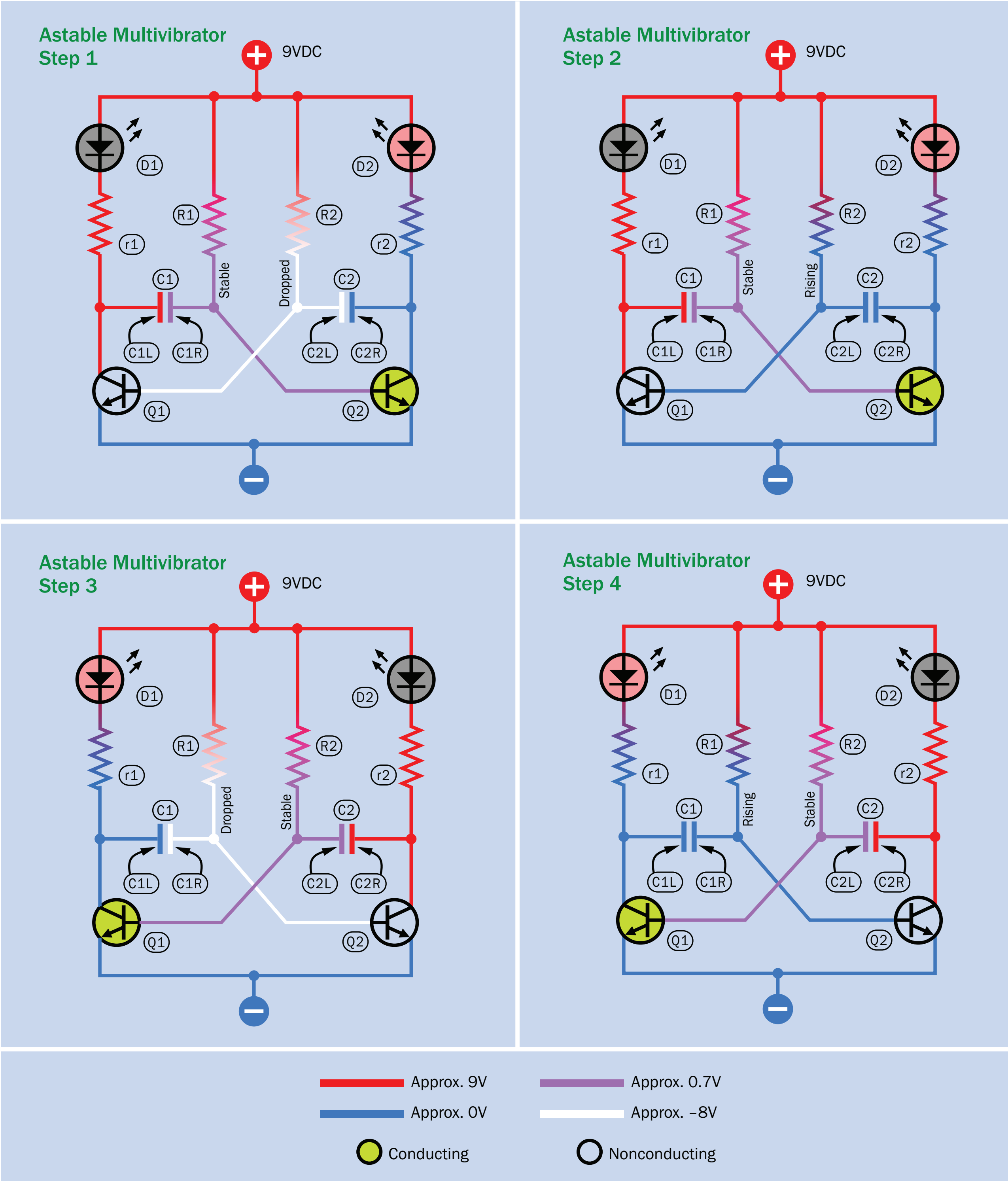
Because C2L is wired to the base of Q1, the sudden low voltage on C2L affects the transistor, and shuts it down. Positive voltage quickly accumulates on C1L, because it has nowhere else to go, now. When that process is complete, D1 has positive voltage on both its leads, so it goes dark.

This completes Step 1 of the sequence.

In Step 2, about a quarter-second later, everything is still the same on the left side of the circuit, but on the right side, C2 is recovering from its traumatic negative-voltage experience. Current has been flowing into C2L through R2, so C2L has risen from about -8V to around 0V, and this voltage is still rising, although it isn't high enough yet to switch on Q1.

In Step 3, C2L has finally climbed back to 0.7V, which switches on Q1. This dumps all the positive charge from C1L to ground, so now it's the turn of C1 to suffer a drastic loss of voltage. This forces C1R down to around -8V, which switches off Q2, and you can see that the situation in Step 3 is the mirror image of Step 1.

In Step 4, C1R recovers and rises to 0V, which is not yet high enough to reactivate Q2, but soon it will reach 0.7V—and the whole sequence will loop back to Step 1.



Figures 11-5, 11-6, 11-7, and 11-8.

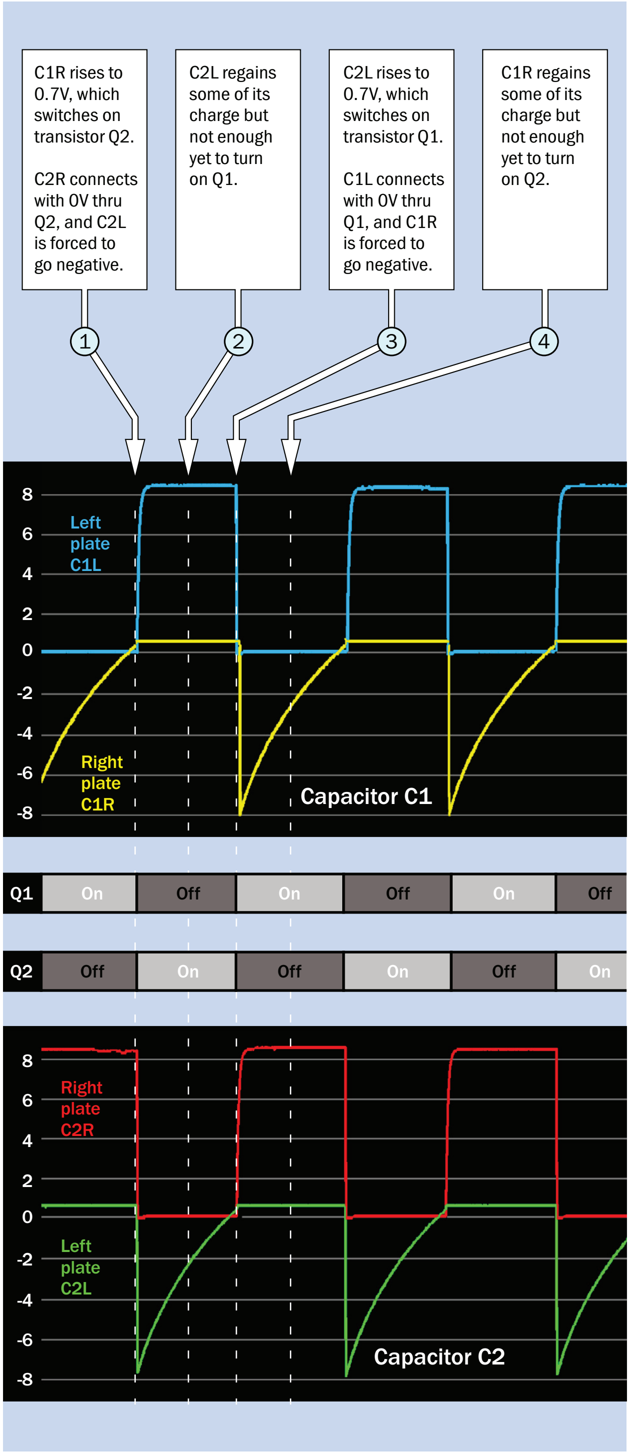


Figure 11-9. Oscilloscope screen captures during the flasher performance.

You can see that the concept of displacement current is essential, here. Since most explanations don't mention it, that's why I think of this as a Most Misunderstood Circuit.

I should just add one note. I simplified the situation when I described how the circuit begins flashing. Before the first visible flash, an oscilloscope reveals that some very small fluctuations occur which are too rapid for you to see in the LEDs. The capacitors take turns until one of them finally reaches 9V. Still, the basic principle of the circuit is the way I have described it.

The Numbers

Understanding electronics would be so much easier if wires were transparent and electrons were visible as little blue dots cruising around. Unfortunately this cannot happen, but an oscilloscope helps to reveal what's going on. Your multimeter takes a second or so to sample and display a voltage, but an oscilloscope shows what's happening by sampling a circuit thousands of times each second (or more).

The one that I use has only two probes, but I anchored one to C1L and then moved the other to C1R, then C2L, and then C2R, saving a series of screen captures. I stacked the images in Photoshop to create Figure 11-9.

Each of the curves in this figure shows voltage on a plate of a capacitor, and you can see that the steps that I numbered 1 through 4 on the previous page match the numbered arrows and dashed lines that I added to the oscilloscope captures in Figure 11-9.

At Step 3, notice how the sudden drop from near 9V to near 0V on the blue curve forces the voltage on C1R down to minus-8V, exactly as I described it. The same thing happens between C2R and C2L.

These negative voltages raise a couple of interesting implications.

First, you might question how electrolytic capacitors can be used—but remember, a polarized capacitor only requires that voltage on one plate is ***relatively*** higher than on the other. In the scope traces you can see that the blue curve is almost always at a higher voltage relative to the yellow curve, except for a tiny overlap of about 0.7V lasting for a fraction of a second. Electrolytic capacitors will tolerate this, provided the reversal doesn't last for long and the current is low. Generally speaking, I try to avoid abusing any polarized capacitor, but in this circuit, I think it's okay.

There is also a practical argument: This circuit is so popular, and has been around for so long, we would have seen people complaining if it damaged capacitors. People do tend to complain online when something like that happens!

So, you have your flasher. Now, how can you turn the output into audible sound?

Frequency

The first step toward generating sound is to build an additional copy of your flasher circuit, which will run faster.

You could just change some component values in the circuit that you built already, but I don't want to do that, because I have a plan to reuse it. Therefore, I'd like you to build the new version below the old version, as shown in Figure 11-10. The new version is printed in color, while I have grayed out the components that you placed previously.

Please be careful to leave a gap between the two circuits on your breadboard, exactly as shown in my diagram, because in the next phase of this project I want to install some extra components there. Also note that the circuit is now extending into the bottom half of your breadboard, so remember to add jumpers on the buses if each of them contains a gap halfway down.

The layout of the new audio circuit is basically the same as that of the original version, omitting the LEDs. The higher-value resistors near the center of the breadboard are now 47K instead of 470K, while the capacitors are 10nF instead of 1μF.

The round object at the bottom of the board is a *speaker*, such as either of the ones which I showed previously in Figure 6-26. Your speaker may be larger than the one in the breadboard diagram, but so long as it has a value of 8 ohms, the physical diameter doesn't matter much in this circuit.

Most speakers have no polarity, so you can connect yours either way around. If one wire from your speaker is red and the other is black, this is usually irrelevant. If your speaker only has solder tabs on it, you can use

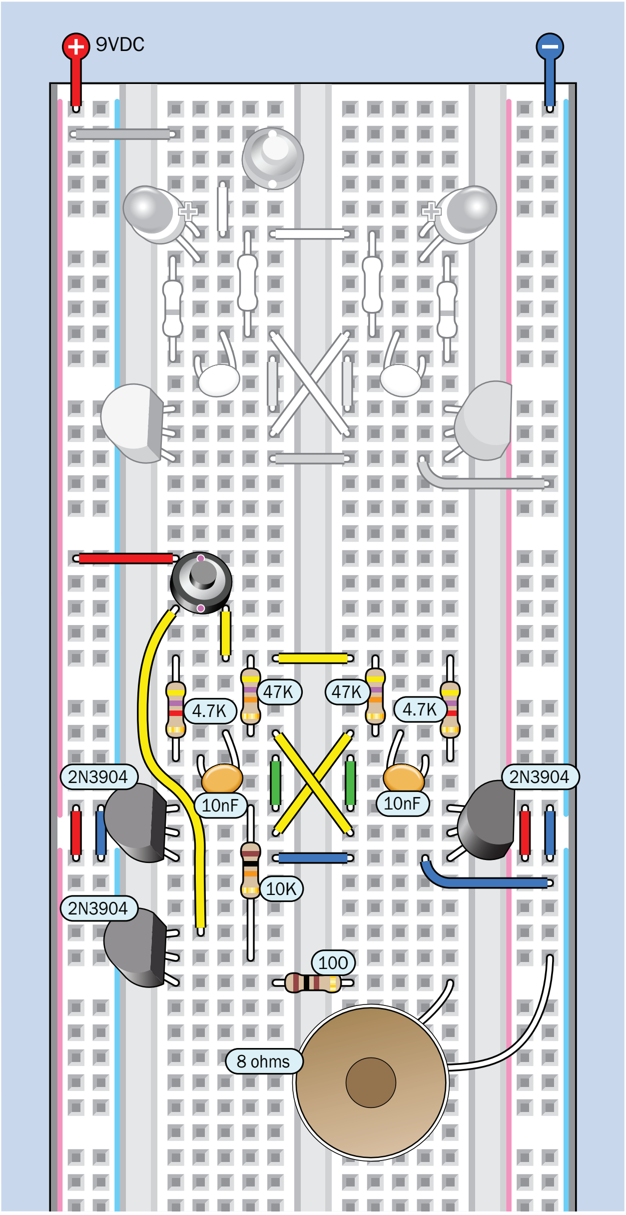


Figure 11-10. A sound-generating version of the astable multivibrator circuit has been installed below the flasher circuit.

alligator test leads to connect it with jumpers in your breadboard.

Press the button, and you should hear a thin whiny sound from the speaker. It doesn't sound very interesting, but I promise to make it more interesting as soon as possible.

Because the speaker has such a low effective resistance, I've put it in series with a 100-ohm resistor. It still requires a lot more power than an LED, so I added another transistor at the bottom of the circuit to drive it. The collector of this transistor is connected to power supplied via a long yellow jumper that snakes down from the pushbutton, while the base of the transistor is connected through a 10K resistor, through a green jumper wire, to the left lead of the 10nF capacitor. So, Q5 steals just a little current from the capacitor, amplifies the pulses, and passes them along to the speaker.

A real audio amplifier would be much more complex, but in this application the transistor only has to amplify a stream of pulses without any concern for exactly how they sound. I used my oscilloscope to check that the transistor didn't need additional resistors to limit the current or adjust the voltage levels.

The speaker contains a magnet and a coil which cause its ***diaphragm*** to vibrate. This emits waves of air pressure which your ears interpret as sound. How fast will the vibrations be? In the flasher circuit, C1 and C2 were 1µF while R1 and R2 were 470K. The flashes then repeated in slightly less than a second. In the audio version of the circuit, I used 47K resistors instead of 470K, and 10nF capacitors instead of 1µF capacitors. The resistors are therefore 1/10 the previous value, while the capacitors are 1/100. Because 10 x 100 = 1,000, the circuit now vibrates about 1,000 times as fast. My meter measured 1,700 pulses per second, which create the tinny noise through your speaker.

Because a speaker requires much more power than an LED, your 9V battery will run down if you use this circuit for long. In the next section of the book, I'll suggest an AC adapter which plugs into the wall and delivers 9VDC; meanwhile, the battery is adequate for testing purposes.

Hertz

The number of pulses per second is properly called the ***frequency,*** measured in ***hertz.*** This international electronic unit is named after Heinrich Hertz, who was yet another electrical pioneer. If you have one pulse per second, this is 1Hz, while 1,000 pulses per second are 1 kilohertz (written as 1kHz) and 1,000,000 pulses per second will be 1 megahertz (written as 1MHz). The letter **H** is capitalized because the unit is derived from a person's name, while the letter **M** is capitalized because if it was a lowercase **m**, it would mean "milli" instead of "mega."

An LED cannot respond quickly enough to display 1.7kHz, and even if it could, your eyes wouldn't see the flashes. Because of ***persistence of vision***, flashes that occur more rapidly than about 30Hz tend to blur together.

The human ear is a different matter. It hears pulses as sound when they have a frequency between about 40Hz and 15kHz (although among older people, 10kHz may be the upper limit, especially for those who went to a lot of loud rock concerts when they were young).

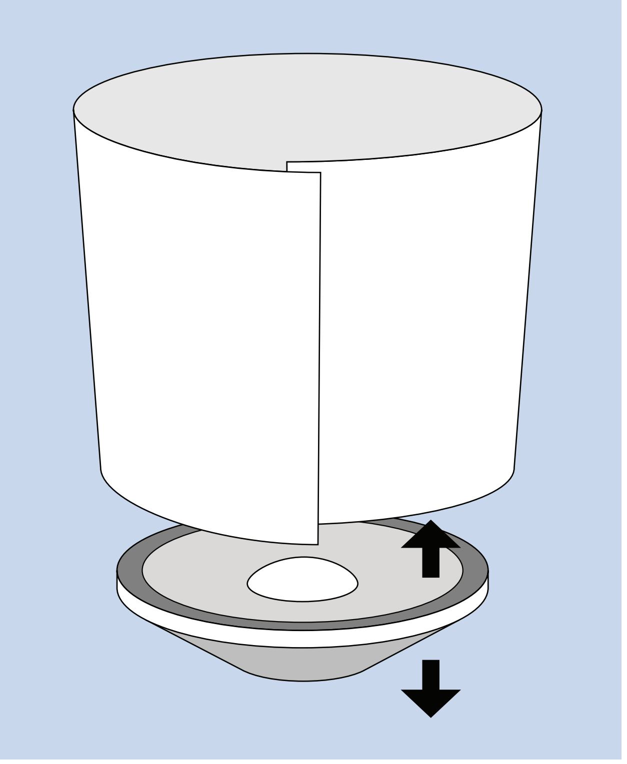


Figure 11-11. Adding a paper tube to make a speaker sound louder. Arrows indicate vibrations of the speaker cone.

Mounting a Speaker

The diaphragm of a speaker is also sometimes known as its ***cone***. The front surface is designed to radiate sound, but the back side also generates pressure waves, and since the two sets of waves are opposite in phase, they tend to cancel each other out.

The perceived output from a speaker can increase dramatically if you add a tube to separate the output from the front from the output from the back. For a miniature speaker, you can bend and tape a piece of paper around it as in Figure 11-11.

Better still, mount it in a box, so that the box absorbs the sound from the rear of the speaker. For purposes of these simple experiments, I won't bother to go into the details of *vented enclosures* and *bass-reflex enclosures,* but you can search for those terms online. You can also buy small speakers that are preinstalled in boxes, and you'll find the sound is amazingly different from the noise created by a naked speaker on your workbench.

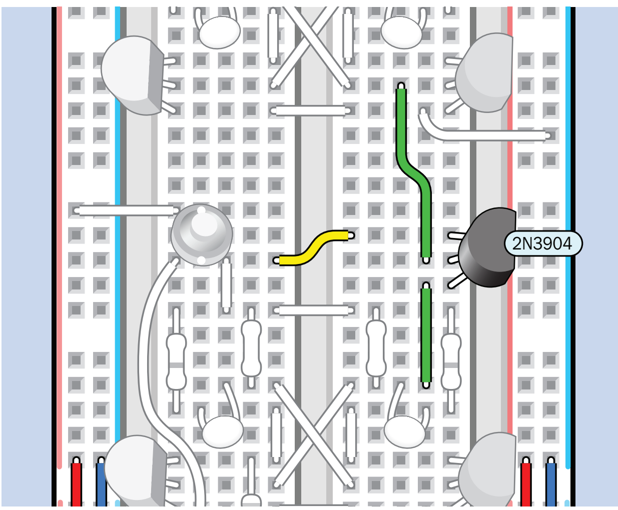


Figure 11-12. Connecting the two sections of the circuit.

Smoothing and Tweeting

I promised to give you a sound like a car alarm, so let's take care of that now. In Figure 11-12, you'll see I am asking you to add two green jumper wires, one yellow jumper, and another transistor to link the circuit at the top of the board with the circuit at the bottom.

Press the lower button—no change. Press the upper button—no change. Hold down both buttons at once—what a surprise!

The yellow jumper that you added supplies power to the new transistor, and its base is supplied through the new green jumper wire, which you can follow back to a 1μF capacitor. The signal on this wire rises gradually to about 0.7V and then drops suddenly to minus-8V because of displacement current. So, the new transistor is being driven by this weird voltage fluctuation, but its emitter connects with a 10nF capacitor which is going through a similar cycle, only 1,000 times as fast. Viewed on my scope, so far as I can see, the transistor that is connecting the top half of the circuit with the bottom half doesn't seem to be subjected to cruel and unusual punishment.

Its purpose is to *modulate* the 1.7kHz of the audio circuit. This means that it superimposes a slow frequency on the high frequency.

You might wonder why I couldn't connect the top half of the circuit with the bottom half by just using a simple piece of wire. Well—I tried that! I didn't think it would do much, and it didn't, because the lower half of the circuit tried to slow down the upper half, while the upper half tried to speed up the lower half, and I just got a frequency somewhere between the two. I needed a transistor to sample the fluctuations in the flasher circuit and add them to the fluctuations of the audio circuit.

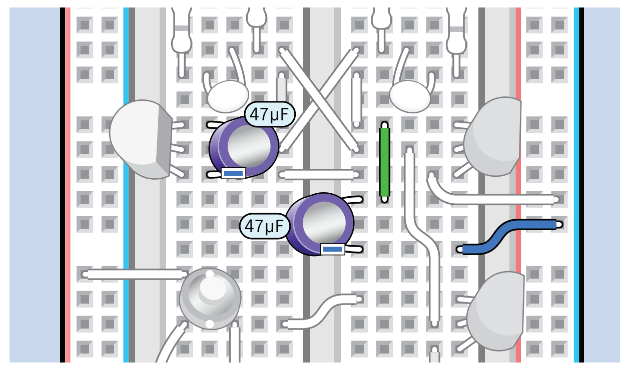


Figure 11-13. Two 47μF smoothing capacitors, the one on the right connected through one green and one blue jumper.

Here's another thing you can do, which I regard as an optional extra: Add smoothing capacitors to the flasher circuit, to make it fade in and out. Figure 11-13 shows how. On the left, a 47μF capacitor has been added across an output transistor. Another 47μF capacitor plays the same role on the right, except that I didn't have room to place it close to the transistor, so I had to run the green jumper out to it.

Now when you power up the flasher circuit (leaving the audio section off for a moment), you'll find the LEDs fading in and out instead of blinking abruptly.

Power up the audio in addition, and you'll find that the sound has changed, too.

Figure 11-14 shows one final modification, adding a 10nF capacitor to make your circuit chirp like a bird.

The complete breadboarded circuit is shown in Figure 11-15, and the schematic is in Figure 11-16. Figure 11-17 shows a components-only diagram, omitting all the wires and using a pale background, so you can see the components and their values more clearly, with a parts list added at the bottom.

You can experiment some more on your own, changing the various resistors and capacitors. Just be sure that the ratio of values between the resistors labeled with an R and the resistors labeled with an r is somewhere between 10:1 and 100:1. The circuit won't operate reliably if the ratio gets too high, because it has to remain below the beta value of the 2N3904 transistors, which is around 200:1.

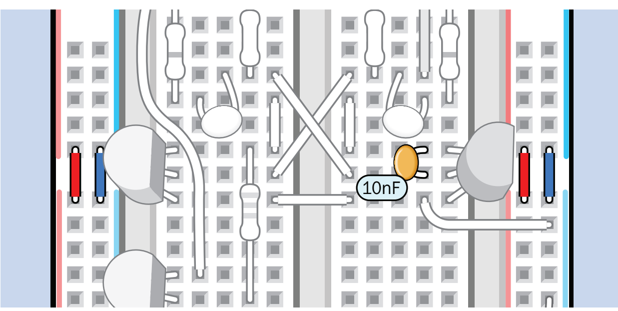


Figure 11-14. One additional 10nF capacitor.

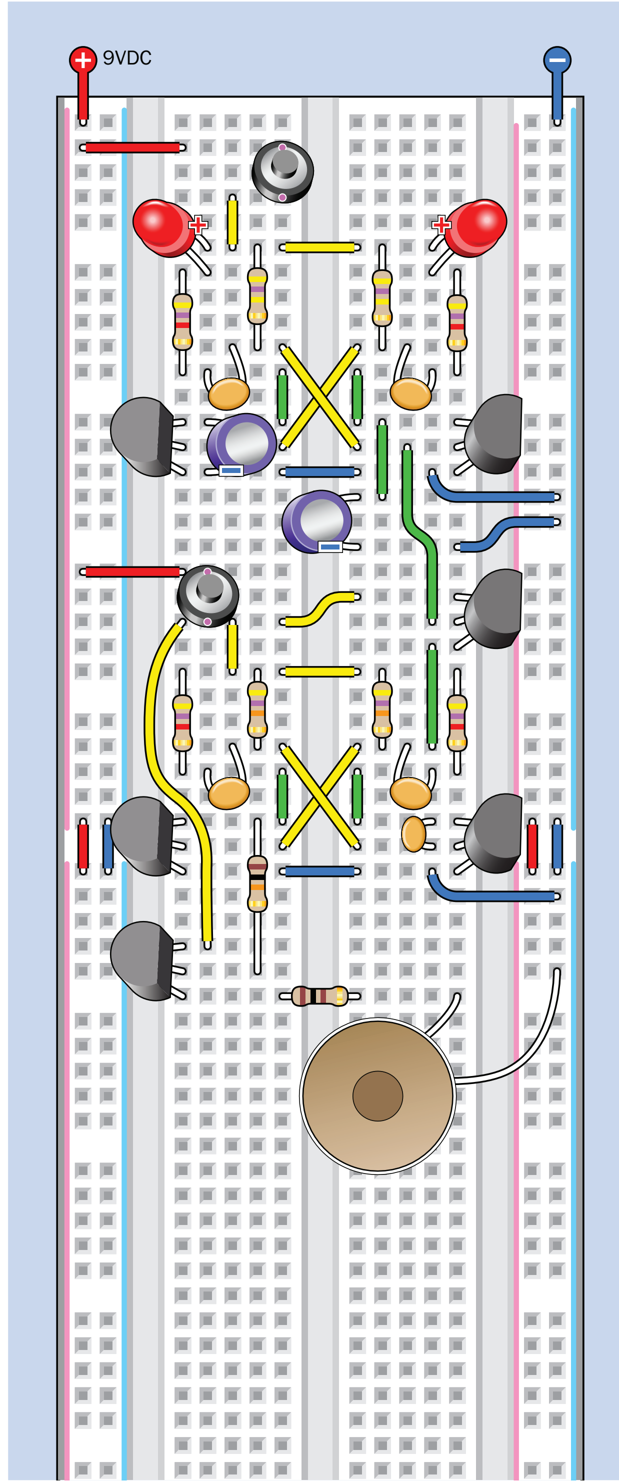


Figure 11-15. The complete breadboarded astable multivibrator with all optional components included.

Output Options

Before I leave the topic of transistors, I have to describe the two basic ways to take output from them. One is known as the *common-collector* configuration, while the other is the *common-emitter* configuration.

In the circuit so far, I used the common-collector option. Power is supplied directly to the collector, and the base and the emitter share the collector, so they have it in common. At least, that's how I think of it. Skipping ahead,

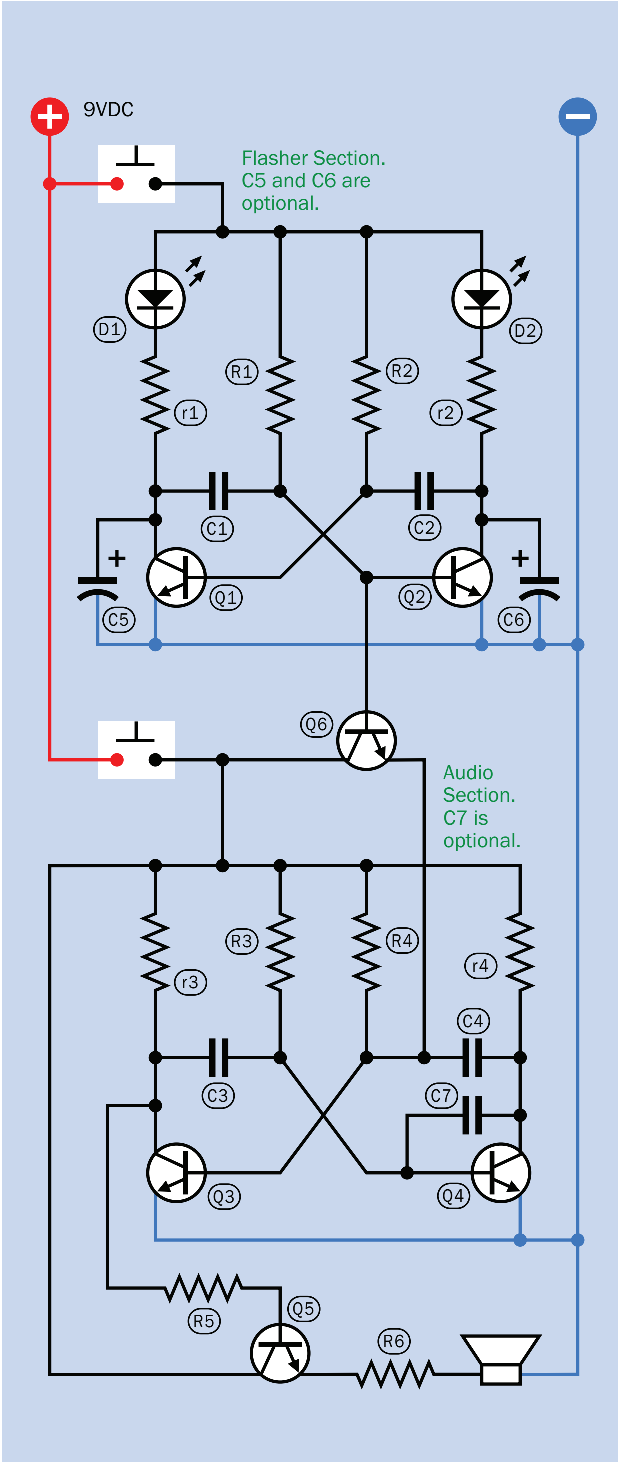


Figure 11-16. Schematic version of the complete astable multivibrator.

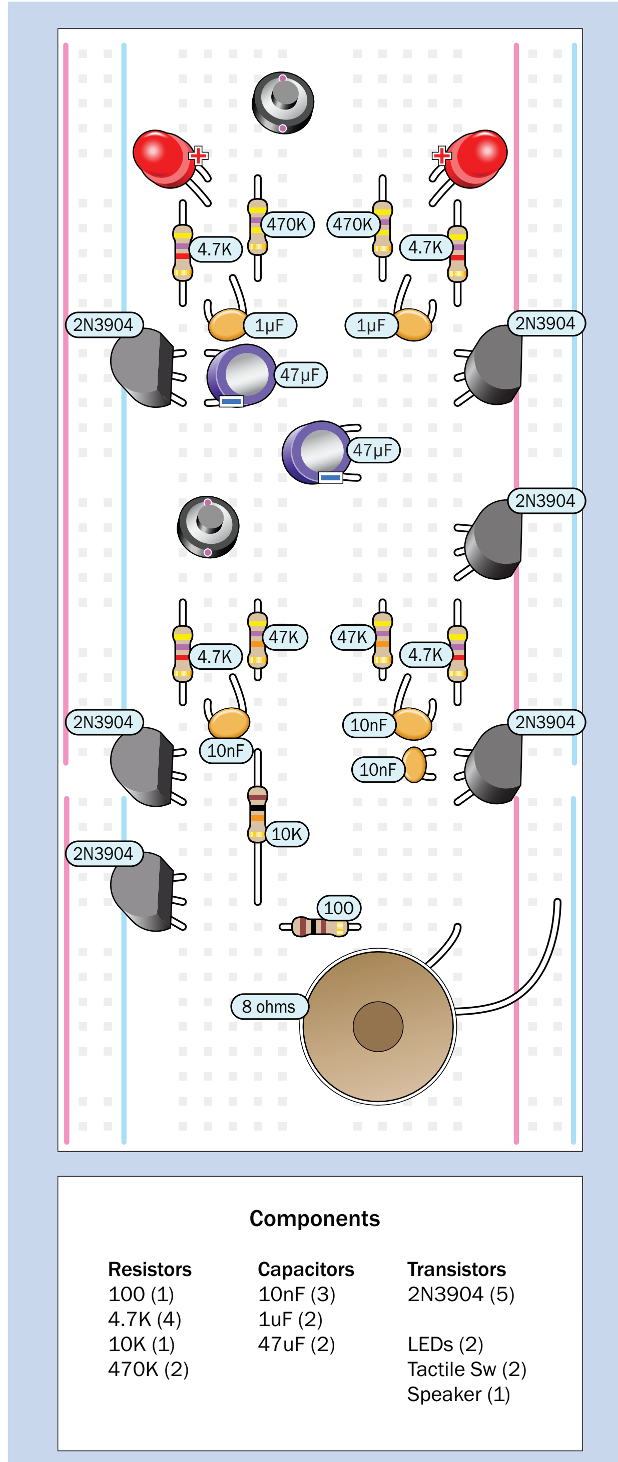


Figure 11-17. This components-only view should help you to choose and place the components before you wire them together.

you can see this in Figure 11-19, but before you study that in detail, you should hear the difference.

Try changing the circuit to look like Figure 11-18. Now current flows through the 100-ohm resistor, and then the speaker, to get to the collector of the transistor, while the emitter is connected directly to negative ground. The base and the collector are sharing the emitter, so now they have it in common, and this becomes a *common-emitter* circuit.

What you will notice, when you try this, is that the sound is twice as loud. Wow, what happened?

Figure 11-19 simplifies the circuit. The transistor and the load are in series, in both cases, but in the common-emitter configuration, the control voltage goes directly through the transistor to negative ground, while in the common-collector configuration, the control voltage is separated from ground by the load. This makes a dramatic difference.

In Figure 11-20 you can see the values that I actually measured in the two versions of the multivibrator circuit. In the common-collector configuration, the fluctuating voltage difference across the speaker (and its series resistor) ranged from 0V to 5.6V. In the common-emitter

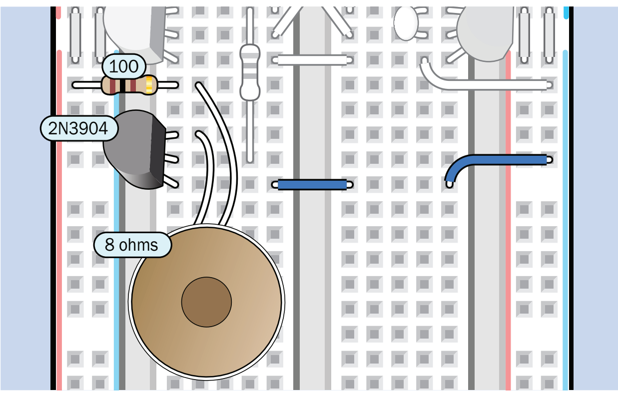


Figure 11-18. Rewiring the circuit in common-emitter mode.

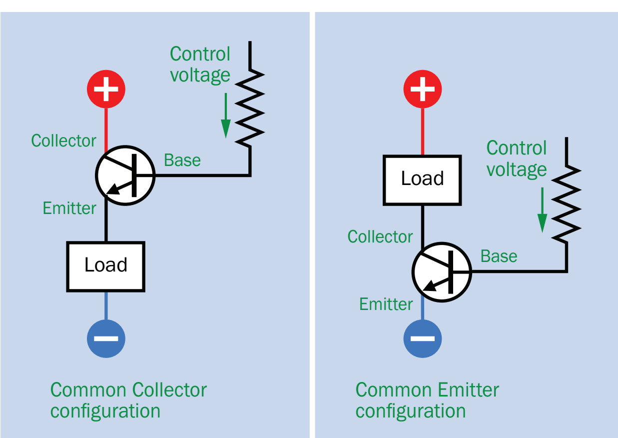


Figure 11-19. Comparing common-collector and common-emitter transistor amplifier circuits.

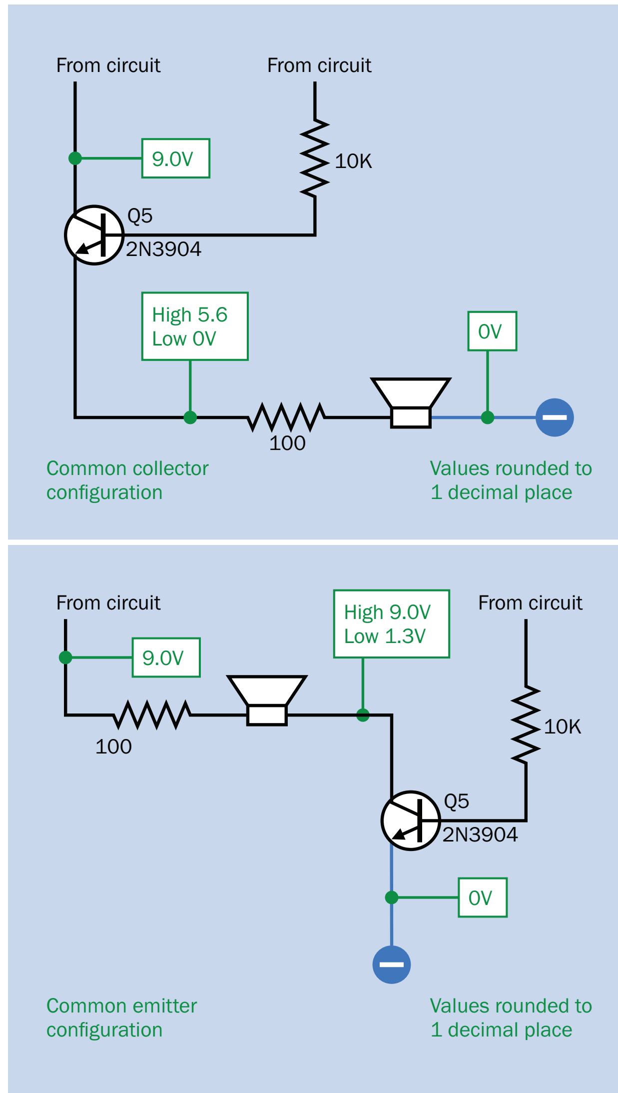


Figure 11-20. Measurements of voltage difference across the speaker and resistor in the multivibrator circuit.

version, if you subtract the speaker output from the input the range is 0V to 7.7V. No wonder it was louder!

Measuring these voltages was difficult, because they were oscillating more than 1,000 times per second. However, it's easy set up a circuit where the voltage is constant, as in Figure 11-21.

In this circuit a steady 9VDC is applied to the transistor, now using just a 100-ohm resistor as the load, because it's not a good idea to run DC through a speaker.

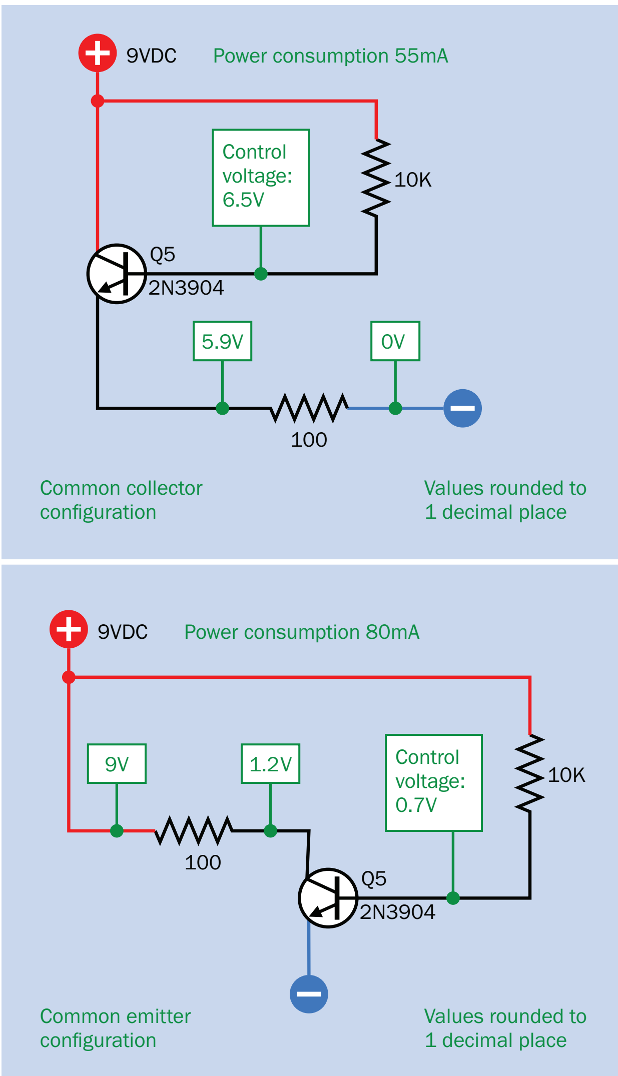


Figure 11-21. A test circuit measuring DC voltages in two configurations.

Even though the resistances are the same in each case, the potential difference across the load is 5.9V in the common-collector circuit but 7.8V in the common-emitter circuit, which also uses about 50% more power.

The current is passing through the same components in each case. They're just in a different sequence. Why should that make such a difference?

I could go into a lot of discussion about this, but the heart of the matter is, the behavior of semiconductors changes a lot depending on voltages applied to them. The effective resistance of an NPN bipolar transistor changes radically depending on the voltage difference between the base and the emitter.

I may be venturing into areas which are more technical than you want right now, but the take-home message is important.

- You can usually extract more power from a transistor if you wire the circuit in common-emitter mode.

So why did I use the common-collector mode initially in the multivibrator circuit?

- I think the common-collector configuration is more intuitively obvious. Hanging the load on the output from a transistor makes intuitive sense.
- The sound wasn't so loud, but the power consumption was much less, which I thought was important in a battery-powered circuit.
- The breadboard layout was more convenient.

Up Next

This completes my general introduction to switching, resistance, capacitors, and transistors. You now have most of the basic knowledge to build all kinds of circuits—such as an intrusion alarm, which will be the next project.

Before you get to that, I will show you how to make circuits more durable by soldering components onto perforated board. Then I'll use timer chips that create all kinds of unexpected possibilities, and I'll use logic chips that can count and make decisions.

Section Three

Soldering

After you assemble a circuit on a breadboard, would you like the option to make a permanent version? This would mean relocating the components and joining their leads together with little droplets of hot metal alloy known as ***solder***. The process is known as ***soldering,*** which is much easier than it sounds, because although solder does get quite hot, it can be applied safely in tiny amounts.

I'm going to take you step by step through the soldering process, and will show you how to make a permanent version of the flasher circuit that I described in Experiment 11. I will also suggest a power supply that you can use to eliminate the need for batteries throughout the rest of this book. This will be helpful for some circuits which use more current than those you have built so far.

- See Appendix A for buying information about components, tools, and supplies.
- See Appendix B for sources that I recommend when you are shopping online or in physical stores.

Soldering Iron

A soldering iron is pencil-shaped and 8" to 10" long, with a sharp tip that gets hot enough to melt solder. It does not have to be a high-cost item; even the cheapest will be adequate for your initial adventures. The question is, how powerful should it be?

A low-wattage soldering iron, typically 15W, is ideal for small and delicate components such as transistors and LEDs which can be damaged by excessive heat. A medium-wattage soldering iron, typically 30W, is easier to use because solder flows and sticks more readily when you have more heat—but it is more likely to damage components. Ideally you would buy a 30W iron for practicing, and a 15W iron for actually building circuits; but if you have to choose just one, you need the 15W version.

Occasionally I have seen dual-wattage soldering irons, but they were too large for detail work.

Figure 12-1. A miniature soldering iron that is good for precise work with heat-sensitive components..

The type of iron that I like is shown in Figure 12-1, and is sometimes referred to as "miniature." When you are making connections at intervals of 0.1", you may find that this is easier to use than medium-sized models.

If you decide to buy a 30W soldering iron, the one that I like is the Weller Therma-Boost, shown in Figure 12-2. When you pull the trigger, it "boosts" the heat temporarily, which is helpful if you ever have to join heavy copper wires that absorb heat away from the connection. Some people like the pistol-grip style; others find it more awkward to use; but I have no strong opinion either way.

Yet another option is a thermostatically-controlled soldering iron, which may either be a self-contained tool with two little buttons to set the temperature, or a "soldering station" which entails an adjustable power supply in a box that sits on your workbench. Personally I don't think a thermostatically controlled iron is necessary when you are learning to solder.

You can decide which type of tip you prefer on a soldering iron. A tapering conical tip is precise, but may not distribute heat so quickly. A spade tip, which looks like a flat screwdriver blade, is a popular alternative. Some soldering irons are sold with multiple tips, so you can find out for yourself which type you prefer.

Figure 12-02. The Weller Therma-Boost is useful for practicing soldering and for making connections with heavy copper wires that tend to absorb heat.

Solder

It looks like bare wire, but actually melts around the wires that you want to connect. Some samples are shown in Figure 12-3. Thin solder, 0.02" to 0.04" (0.5mm to 1mm) in diameter, is easiest to use with small components. For the projects in this book, a minimal amount of solder (half an ounce, or maybe three feet) will be sufficient.

Avoid buying solder that is intended for plumbers, or for craft purposes such as creating jewelry. It may contain an acid which is unacceptable for use with electronic components. The word "electronics" should appear in the manufacturer's description of suitable purposes for the solder, and it must have a ***rosin core***.

There is some controversy about using solder that contains lead. Some people feel that this older type of solder makes better, easier joints at a slightly lower temperature, and does not entail any health risk. I also know an electronics engineer who used to work in the U. S. Navy, and he told me that all the solder he used contained lead. Personally I lack the specialized knowledge to make a judgment call, but I do know that if you live in the European Union, you're not supposed to use solder with lead in it, for environmental reasons.

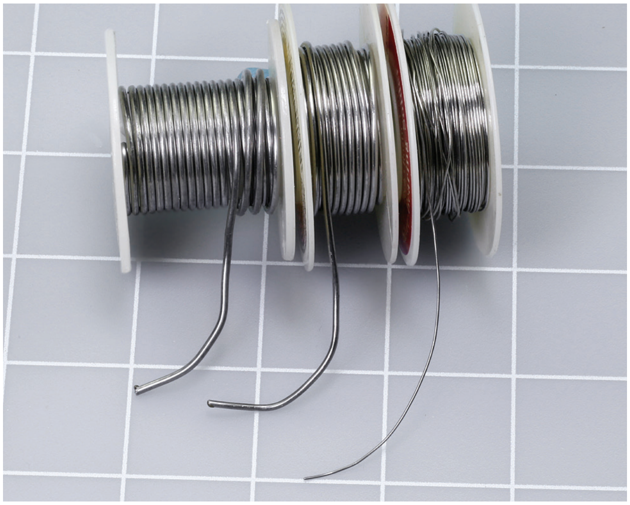


Figure 12-3. Thicknesses of solder.

Terminology and Accessories

A soldering iron may have the word ***welding*** in its product description. You can ignore this, because soldering irons do not do welding in the usual sense. I don't know why the term has come into use.

Many soldering irons are described as ***pencil type***, but this term is not very informative, because it can be applied either to a 15W or a 30W iron, if it is small to medium in physical size.

Soldering irons are often sold with a kit of ancillary items, such as various tips, a spool of solder, a stand for the soldering iron, or a "helping hand" device to hold small parts while you are working on them. I suggest you read the rest of this section to decide if you need such accessories, then search online to see if a kit is available including the items that you want. You should be able to save some money this way.

Figure 12-4. This version of the "helping hand" includes a wire spiral in which to park your soldering iron, a sponge to clean the tip, and a large magnifying glass.

Helping Hands

Sometimes known as a third hand, this device has two alligator clips that can hold components or pieces of wire precisely in position while you join them with solder. Some versions of the Helping Hands also feature a magnifying lens and a wire spiral in which you can rest your soldering iron, as shown in Figure 12-4.

Figure 12-5. Quad hands.

Various other gadgets are available to hold wires and components while you solder them, and they are more sophisticated than the old "helping hands" which use ball joints tightened with wing nuts. I like the Quad Hands shown in Figure 12-5, but this item is relatively expensive. Search online for

```
soldering third hand
```

to see a range of options. You will definitely need ***some*** way to hold your work.

Figure 12-6. A traditional-style stand for your soldering iron. The yellow sponge is for cleaning the tip.

Iron Stand

If you only want the traditional wire-spiral stand for a soldering iron, it's shown in Figure 12-6. Of course, you don't absolutely have to own one of these. You can improvise a substitute, or you can rest the soldering iron on the edge of your work bench, and promise yourself to be ***very, very careful*** not to dislodge it. When—not if—the soldering iron falls onto the floor, it will melt synthetic carpet or plastic floor tiles. Knowing this, you may attempt to catch it when you see it fall. If you grab it by the hot end, you will let go of it very quickly, so you might as well let it fall on the floor without the intermediate step of burning yourself.

Perhaps a soldering stand is really a better idea.

Figure 12-7. Copper alligator clips.

Copper Alligator Clips

When you are soldering the leads of a sensitive component, heat tends to travel up the leads into the component, and can damage it. I'll show you how to verify this by roasting an LED in Experiment 13.

If you are a person with a cautious disposition, you may want to apply an alligator clip just above the location of your soldering iron, to absorb some of the heat.

A *copper alligator clip* conducts heat more effectively than the chrome-plated alligator clips on your test leads, but is it really necessary? Personally, I like to use them, because I am one of those people with a cautious disposition. If you decide to buy a couple of copper alligator clips, make sure they are solid copper, not just copper-plated.

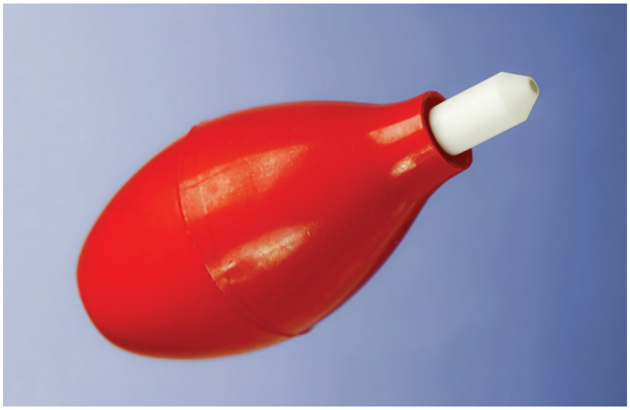


Figure 12-8. A squeeze bulb that sucks melted solder.

Desoldering Devices

When you make a soldering error, there is no "undo" option—but you can try using a ***squeeze bulb*** designed to suck solder while you melt it with a soldering iron that you hold in your other hand. I have difficulty with this technique, but I know people who use it. See Figure 12-8.

Another option is *copper braid*, which is supposed to attract at least some solder from a joint. See Figure 12-9. I am mentioning these devices because any description of soldering would be incomplete without them. Personally I don't think they work very well.

I know one person who uses braid, but he reports that to make it work, you have to apply more heat than can be delivered by a 15W soldering iron.

Figure 12-9. Copper braid to gather solder from a joint made in error.

Magnifying Lens

No matter how good your eyes are, a small, handheld, powerful magnifying lens is essential when you are checking solder joints.

The three-lens set in Figure 12-10 is designed to be held close to your eye and is more powerful than the large-diameter lens on a Helping Hand, which I do not find very useful. Plastic lenses are quite acceptable if you are gentle with them.

Don't forget you can also set the camera in your phone to macro mode, and use it as a handheld microscope.

Figure 12-10. These three lenses are identical, but if you put two or three of them together, you multiply the total magnification.

I also like using a head-mounted magnifier of the type shown in Figure 12-11. It keeps your hands free, moves as you move, and because it has a lens for each eye, you maintain the benefit of depth perception. You can spend a lot of money on head-mounted magnifiers of the type used by dental hygienists, but for our purposes, a cheaper version does the job.

Figure 12-11. A low-cost head-mounted magnifier.

Heat-Shrink Tubing

When you have joined two wires by soldering them, you often want to put some insulation around the joint, and the best way to achieve this is with some ***heat-shrink tubing***. You slip the tube over the joined wires, then apply a stream of hot air until the tube grips the joint tightly. Most heat-shrink tube can contract to half of its initial diameter.

Figure 12-12. Some sizes and colors of heat-shrink tube.

One bag or box containing an assortment of three or four small diameters will be sufficient for the projects and exercises in this book. The colors are only of cosmetic interest. See figure 12-12.

The only disadvantage is that to make this work, a heat gun as advisable.

Figure 12-13. A full-size heat gun.

Figure 12-14. A small heat gun, suitable in electronics work when activating heat-shrink tubing.

Heat Gun

Your primary use for this tool is to activate heat-shrink tubing. A full-size version is shown in Figure 12-13, but is actually more than you need for tubing. The miniature version in Figure 12-14 is sufficient, and its narrower stream of hot air is easier to confine to the joint that you are working on.

Figure 12-15. Wire nuts are color-coded. You need the smallest, which are gray.

Lower-Cost Alternatives

If you don't want to do soldering—for whatever reason—and you prefer to avoid the expense of heat-shrink tubing and a heat gun, you may still need a way to join wires together and insulate them, especially when you adapt the power supply that I will be describing shortly.

Your local hardware store will sell ***wire nuts***, which are commonly used in the United States for connections in house wiring. For low-voltage wiring you will need the smallest size, which is color-coded gray and is designed to connect pairs of wires ranging from 22 to 16 gauge. See Figure 12-15. One small packet will be sufficient.

Other countries use their own types of wiring connectors, but the type doesn't really matter, so long as it will handle 22-gauge wire.

Another option is to use a ***screw-terminal block*** as shown in Figure 12-16. The type illustrated consist of several connectors in a row, allowing you to cut off as many as you need.

Because you will only be making low-voltage connections, you can just twist wires together and use electrical tape for insulation, but it may tend to peel off over time.

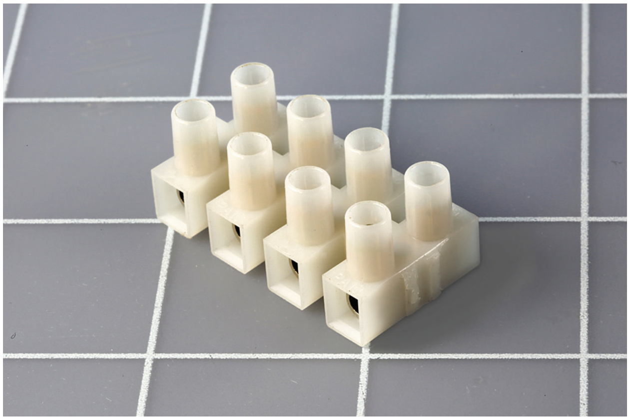


Figure 12-16. A screw-terminal block.

Perforated Board

When you're ready to build a permanent copy of a circuit, you will need something on which to mount the components before you solder them, and this is usually a piece of ***perforated board***. It is also known as ***perf board***, ***prototyping board***, or ***proto board***.

The easiest type to use is plated with copper strips in exactly the same pattern as the conductors hidden inside a breadboard. This enables you to minimize errors by keeping the same layout of your components when you move them to the perforated board. A rather fancy version sold by Adafruit is shown in Figure 12-17 on the next page. They call it ***perma-proto*** board.

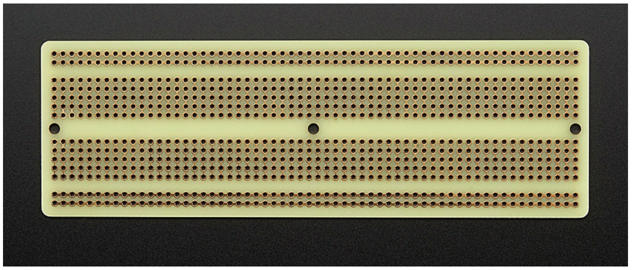


Figure 12-17. This perforated board has a pattern of copper traces identical to the pattern of conductors inside a breadboard.

The disadvantage of using this type of board is that it is not very space-efficient and consequently may not fit into a project box. To compress a circuit to minimal size, you can try ***point-to-point wiring*** on a plain perforated board of the type shown in Figure 12-18. You can buy a piece of board maybe 6" x 8" and cut off as much as you need with a hacksaw. (Glass fibers in typical perforated board will tend to blunt a wood saw.) I'll show you how to use this board in Experiment 14.

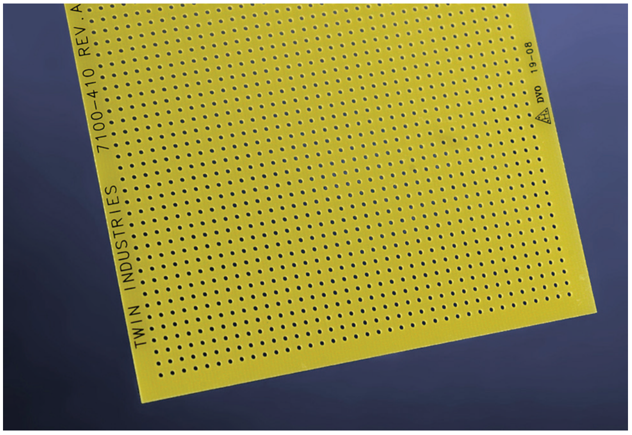


Figure 12-18. Plain perforated board (with no copper traces) for point-to-point wiring.

Perforated board also exists with various patterns of copper traces, which some people prefer. ***Cut-board***, for instance, features parallel traces which you can cut with a knife where you want to break a connection.

Everyone who does much soldering seems to have a favorite type of board configuration, but I think you'll need to get acquainted with the soldering process before you start exploring these options.

Project Boxes

This is just a small box (usually plastic, but sometimes aluminum) with a removable lid. As its name implies, its purpose is to hold one of your electronics projects. You mount your switches, trimmers, and LEDs in holes that you drill in the lid, and you attach your circuit on a perforated board that goes inside the box. You can also use a project box to contain a small speaker. A small box is shown in Figure 12-19.

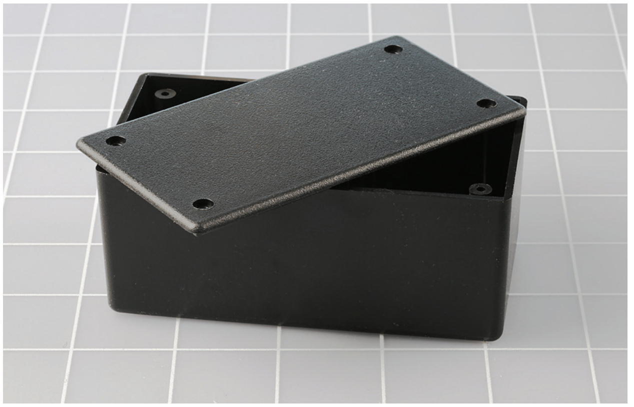


Figure 12-19. A small project box. Screws to secure the lid are included.

You can find dozens of examples of project boxes by searching online for

```
"project box" electronics
```

For the intrusion alarm project that I will describe in Experiment 15, you can use a box measuring approximately 6" long, 3" wide, and 2" high.

Mini-Grabbers

In previous experiments I have suggested that you can grab one of your meter probes with an alligator clip on a test lead, and use the alligator at the other end of the lead to grip a wire or a component.

A more elegant alternative, if you want to make hands-free measurements, is to buy a pair of ***mini-grabbers*** with little spring-loaded clips at the ends, as shown in Figure 12-20.

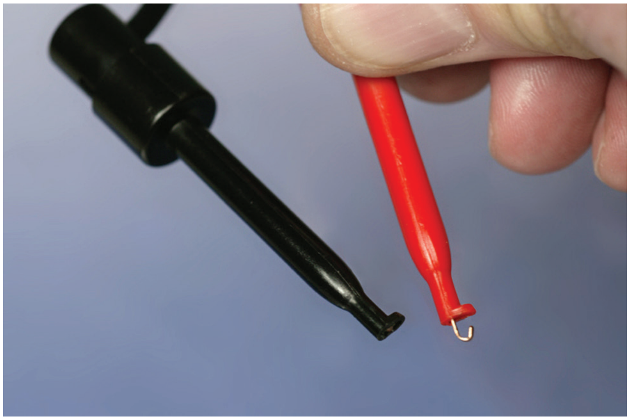


Figure 12-20. Meter leads that terminate in "minigrabbers."

Meter leads that terminate in small alligator clips are also available, such as the ones in Figure 12-21. Or, you can just continue to use test leads in the manner that I suggested previously. Bear in mind that short meter leads are convenient for breadboard work.

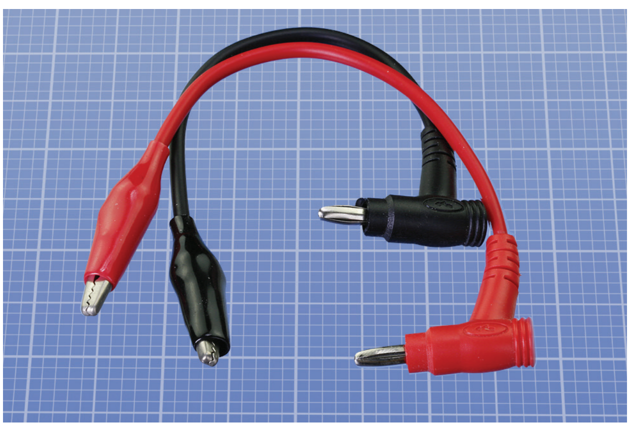


Figure 12-21. Meter leads that terminate in alligator clips.

Power Supply

I have left the most important item till last. An ***AC adapter*** plugs into an electrical outlet and delivers DC (direct current) through a wire that you can bring to your breadboard. Figure 12-22 shows a 9V adapter.

If you plan to continue building circuits, buying an adapter will eventually save money compared with the cost of using batteries. Also, almost any 9V adapter should be able to deliver more current than a 9V battery.

You can buy an adapter that delivers a single, fixed voltage, or a ***universal adapter*** which has a selector-switch allowing you to choose from a range of voltages. That sounds nice—but it costs slightly more, and it may not deliver such a well-regulated output.

Figure 12-22. A typical AC adapter.

"Well-regulated" means that the DC voltage from the adapter should be fixed regardless of how much current you are consuming, and should be smooth and stable. This is important, because in Section 4 circuits will use logic chips that are sensitive to voltage spikes. Therefore I suggest an adapter that delivers 9VDC only, and you may want to buy from a supplier that specializes in electronics parts. In the United States, that would be a source online such as All Electronics, Electronics Goldmine, Jameco Electronics, or protechtrader.com.

Figure 12-23 shows the voltage that I measured from a good adapter. You can see little bumps in the trace,

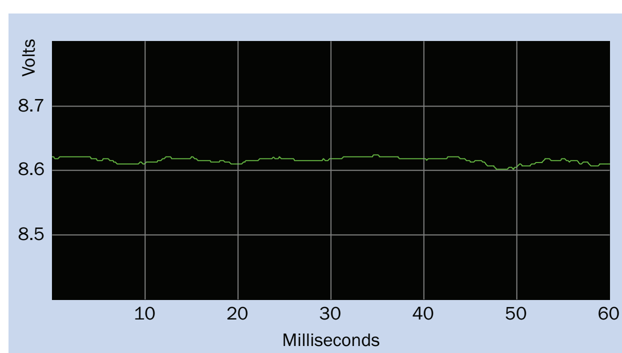


Figure 12-23. Acceptable output from an AC adapter.

but they are only within a range of around 0.02V. The average voltage is closer to 8.6V than 9.0V, but the trace was obtained while this adapter was delivering 300mA through 25 ohms, and I think 8.6V is satisfactory for the projects in this book.

Figure 12-24 shows the performance of a seriously bad adapter. The output in the lower graph shows some im-

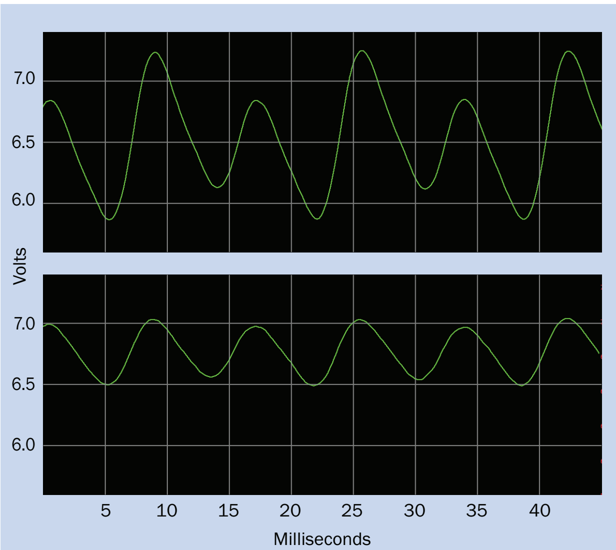


Figure 12-24. This output from an AC adapter is not acceptable. The lower trace was obtained with the addition of a 47µF smoothing capacitor.

provement from adding a 47µF smoothing capacitor, but still it's unacceptable. I suggest that when you are shopping for an AC adapter, this is one instance where you might think twice before buying the lowest-priced item you can find on eBay.

Some other things to bear in mind when buying an AC adapter:

- A cheap adapter may not deal well with an overload if you short-circuit it by accident.
- Make sure that the output is DC, not AC. Almost all adapters give you a DC output, but there are a few exceptions.
- Look for an adapter rated for at least 300mA (which may be written as 0.3A).
- It doesn't matter what kind of plug is on the end of the power-output wire, because you'll probably cut it off when you adapt the output to fit your breadboard. (I will be showing you how to do this.)
- A frivolous term for an AC adapter is a ***wall wart***, and sometimes you may see this in product descriptions.

This completes my list of items relating to the development of permanent circuits. Now it's time to explore the process of soldering.

Experiment 12
Joining Two Wires Together

Your adventure into soldering begins with the simple task of joining one wire to another, but will lead quickly to creating a full electronic circuit on perforated board.

You Will Need:

- Hookup wire, wire cutters, wire strippers.
- Soldering iron. If you only have one, I assume it is rated at 15W. If you also have a 30W soldering iron, it will make your initial experience easier.
- Thin electrical solder, 0.05" diameter (1.5mm) approximately.
- "Helping Hand" or similar device using two alligator clips to hold your work.
- Optional: Small heat-shrink tubing, between 1/8" and 1/4" diameter.
- Optional: Heat gun.
- Optional: A piece of heavy cardboard or plywood to protect your work area from drops of solder.

Caution: Soldering Irons Do Get Hot!

The steam iron that you might use to iron a shirt is actually more hazardous than a soldering iron, because it has a much greater heat capacity. Still, you should be careful when soldering.

Please take these basic precautions:

- Use a proper stand to hold the soldering iron. Don't leave it lying on a workbench.
- If you have infants or pets, remember that they may play with, grab, or snag the wire to your soldering iron. They could injure themselves (or you).
- Be careful never to touch the hot tip of the iron against the power cord that supplies electricity to the iron. It can melt the plastic in seconds and cause a dramatic short circuit.
- If you drop a soldering iron, do *not* be a hero and try to catch it.

Most soldering irons don't have warning lights to tell you that they're plugged in. As a general rule, always assume that a soldering iron is hot, even if it's unplugged. It may retain sufficient heat to burn you for several minutes after you switch it off.

Breathing the fumes from hot solder is something to avoid as much as possible. Working in a ventilated area is a good idea.

Your First Solder Joint

Plug in your soldering iron, leave it safely in its stand, and find something else to do for at least five minutes. If you try to use it without giving it time to get fully hot, you will not make good joints, because the solder may not melt sufficiently. Don't believe manufacturers' claims that a soldering iron is ready to use within a minute or less.

You need two pieces of 22-gauge solid hookup wire, each at least 2" long, with bare ends. Clamp them in your Helping Hands so that they cross each other and touch each other, as shown in Figure 12-25.

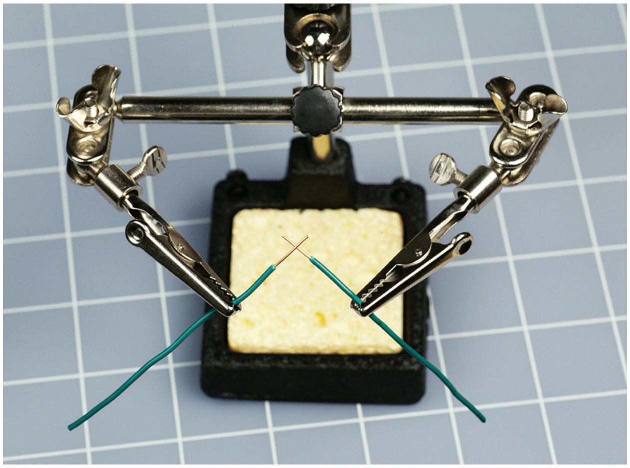


Figure 12-25. Ready for your first soldering adventure.

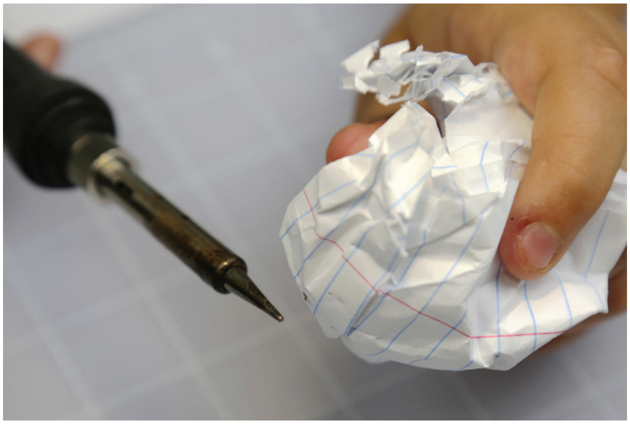


Figure 12-26. One way to clean the tip of a soldering iron. Move the crumpled paper quickly to avoid charring it.

To make sure that the iron is ready, touch a length of solder to the tip of it. The solder should melt instantly. If it melts slowly, the iron isn't hot enough yet, or the tip of it may be dirty and should be cleaned.

Many people clean the tip of an iron by using a sponge such as the one that may be built into your soldering iron stand. You wet the sponge thoroughly, then rub the tip of the soldering iron into it. Personally I prefer not to do this, as I believe that getting moisture on the tip causes thermal expansion and contraction which may open small cracks in the plating of the tip.

I use a rather primitive method: I crumple a piece of plain paper into a ball and rub it over the hot tip of the iron, quickly to avoid charring the paper, taking care that my fingers don't get burned. Then I apply a tiny amount of solder and rub the tip again, until it is uniformly shiny. Figure 12-26 illustrates the beginning of this procedure. Don't ever rub the tip of the soldering iron with anything more abrasive than paper.

Now follow the next four steps to make your solder joint, as shown in figures 12-27 through 12-30.

Step 1. Touch the tip of the iron steadily against the intersection of the wires for at least five seconds, to heat them. Don't apply the solder yet!

Step 2. Feed a little solder onto the intersection of the wires, while also maintaining contact with the soldering iron. The two wires, the solder, and the tip of the iron should all come together at one point.

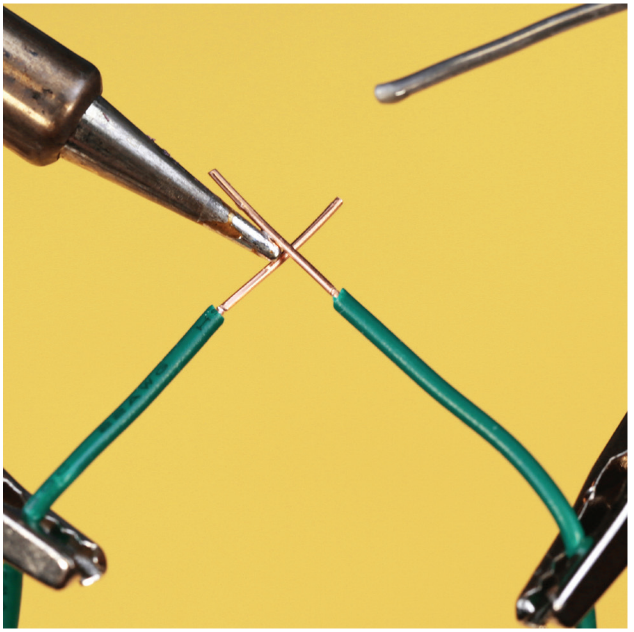


Figure 12-27. Step 1: Heat the wires.

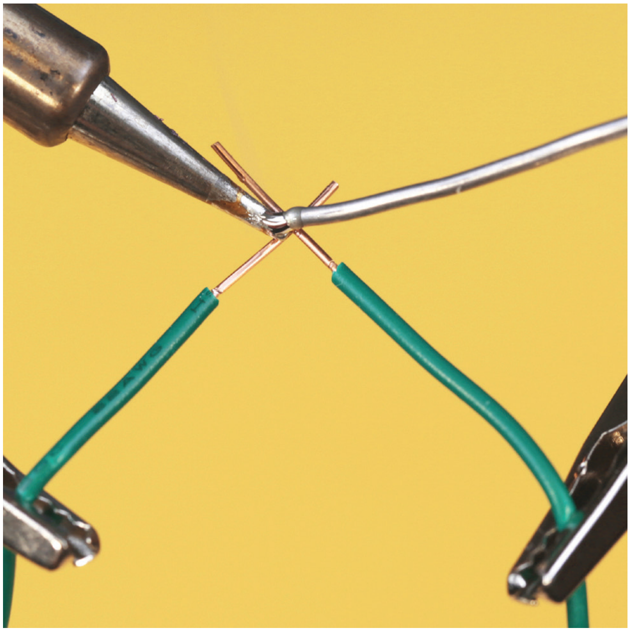


Figure 12-28. Step 2: Add solder.

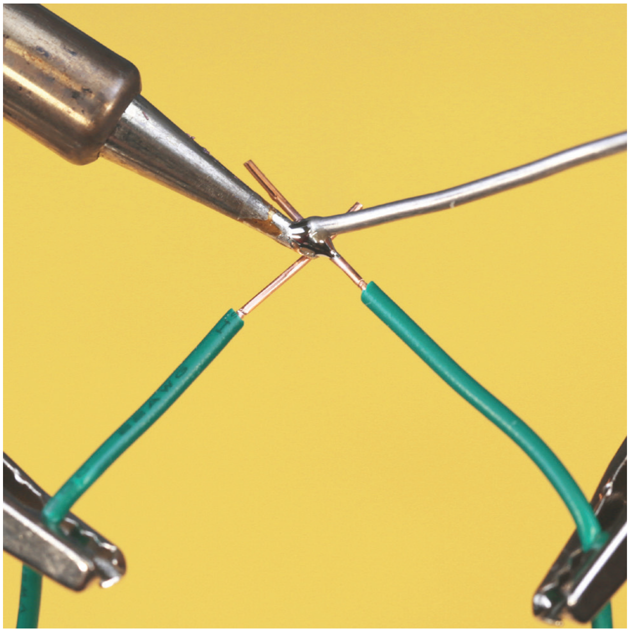


Figure 12-29. Step 3: The solder starts to run into the joint.

Figure 12-30. The solder forms a shiny blob that clings to the wires.

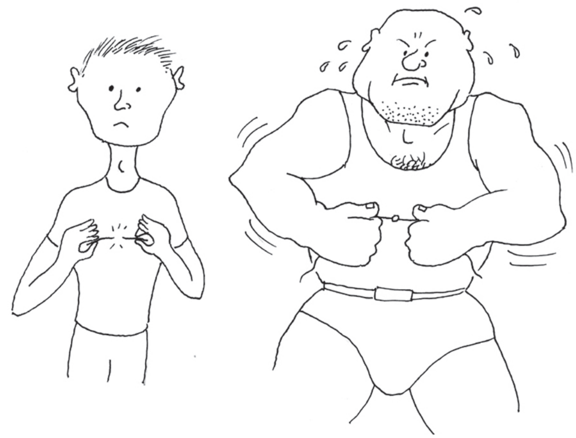


Figure 12-31. Telling the difference between a bad solder joint (left) and a good one (right) is really not very difficult.

Step 3. At first, the solder may melt slowly. Be patient.

Step 4. In Figure 12-30, you see that the solder has formed a nice round blob. If you blow on it to cool it, you should be able to touch it within 10 seconds. The joint should be shiny, uniform, and rounded in shape

When the joint has cooled, unclamp the wires and try to tug them apart. Try hard! If they defeat your best attempts to separate them, the wires are electrically joined and should stay joined. If you didn't make a good joint, you will be able to separate the wires easily, probably because you didn't apply enough heat or enough solder. Figure 12-31 gives you the general idea.

Three Soldering Myths

Myth #1: Soldering is very difficult. Millions of people have learned how to do it, and they can't all be more competent than you!

Myth #2: Soldering endangers your health with poisonous chemicals. You should avoid inhaling the fumes, but that also applies to everyday products such as bleach and paint. If you are concerned about touching solder, you can wear nitrile gloves.

Myth #3: Soldering irons are dangerous. Of course, you have to be careful. The iron is certainly hot enough to burn your skin if you touch it, and you should never grab hold of it. But power tools in a workshop are much more hazardous, in my experience.

Eight Soldering Errors

Not enough heat. The joint looks okay, but because you didn't apply quite enough heat, the solder didn't melt sufficiently to realign its internal molecular structure. It remained granular instead of becoming a solid, uniform blob, and you end up with a ***dry joint***, also known as a ***cold joint***, which will come apart when you pull the wires away from each other. Reheat the joint thoroughly and apply new solder.

Carrying solder to the joint. A leading cause of underheated solder is the temptation to melt some solder onto the tip of the iron, and then carry the solder to the location where you want to apply it. This means that the wires will be cold when you try to make the solder stick to them, and they will take heat away from the solder, so that it won't stick properly. What you should do is touch the soldering iron to heat the wires first, and then apply the solder. This way, the wires are hot, and they help to melt the solder.

- Because this is such a universal problem, I'll repeat myself. You don't want to put hot solder on cold wires. You want to put cold solder on hot wires.

Too much heat. This is not such a risk with a 15W soldering iron, but still can happen. Sustained application of heat may help you to make a good joint, but can damage everything around it. Vinyl insulation on wires will melt, and you may possibly damage semiconductors. Damaged components must be desoldered and replaced, which will take time and is a big hassle. If your attempt at soldering isn't working for some reason, pull back, pause, and allow everything to cool down a little before you try again.

Not enough solder. A thin connection between two conductors may not be strong enough. When joining two wires, always check the underside of the joint to see whether the solder penetrated completely. This is where a magnifier is helpful.

Moving a wire before the solder solidifies. This may create a fracture that you won't necessarily see. It may not stop your circuit from working, but at some point in the future, the fracture can separate just enough to break electrical contact. If you clamp components before you join them, or use perforated board to hold the components steady, you can avoid this problem.

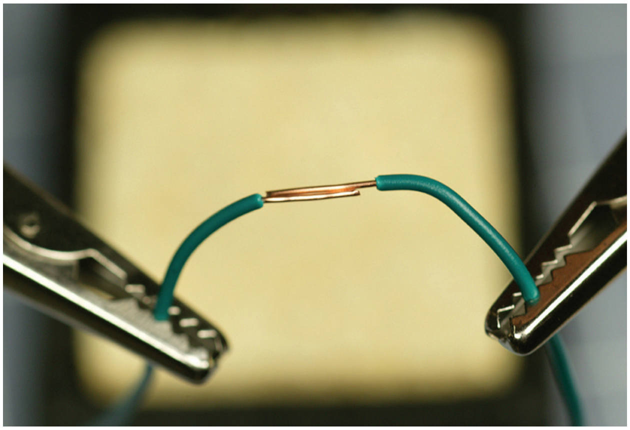


Figure 12-32. Step 1: Align the wires.

Dirt or grease. Electrical solder contains rosin that helps to clean the metal that you're working with, but a film of dirt can still prevent solder from sticking. If any component looks dirty, clean it with fine sandpaper before joining it.

Carbon on the tip of your soldering iron. The tip gradually accumulates flecks of black carbon during use, and they can act as a barrier to heat transfer. Clean the tip, as described previously.

Inappropriate materials. Electrical solder is designed for electronic components. It will not work with aluminum, stainless steel, or various other metals. You may be able to make it stick to chrome-plated items, but only with difficulty.

Failure to test the joint. Don't just assume that it's OK. Always test it by applying manual force. If you can't get a grip on the joint, slip a screwdriver blade under it and flex it just a little, or use small pliers to try to pull it apart. Don't be concerned about ruining your work. If your joint doesn't survive rough treatment, it wasn't a good joint.

Of the eight errors, dry/cold joints are by far the worst, because they are easy to make but can look okay.

Your Second Solder Joint

Time now to make a more challenging solder joint. Once again, your soldering iron should have been plugged in for at least five minutes to make sure that it's hot enough to make good joints.

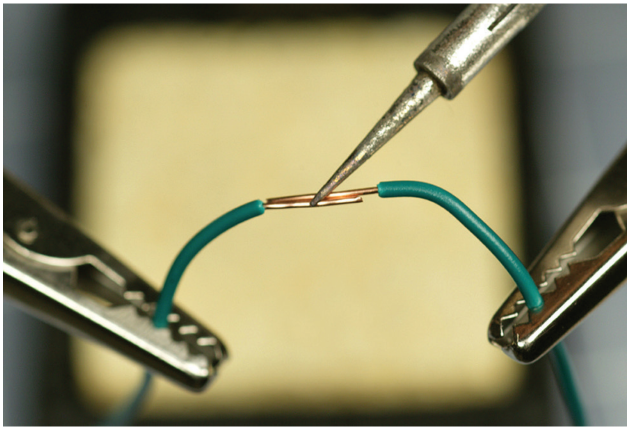


Figure 12-33. Step 2: Heat the wires.

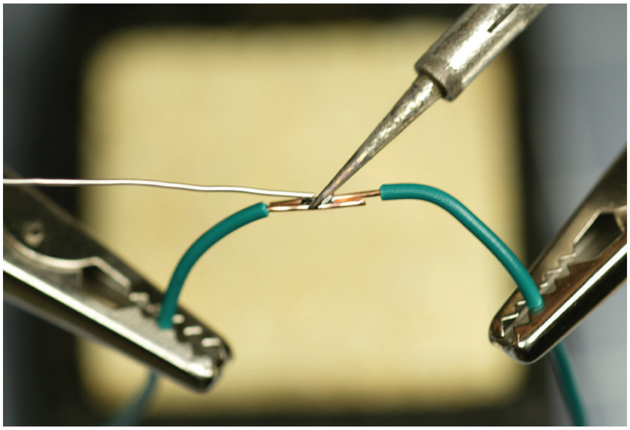


Figure 12-34. Step 3: Apply solder.

This time I'd like you to align the wires parallel with each other. Joining them this way is a little more difficult than joining them when they cross each other, but it's a necessary skill. Otherwise, you won't be able to slide heat-shrink tubing over a finished joint to insulate it.

Five steps to create this joint are shown in figures 12-32 through 12-36. At the beginning, the two wires do not have to make perfect contact with each other; the solder will fill any small gaps. But as before, the wires must be hot enough for the solder to flow, and this can take an extra few seconds when you use the low-wattage iron.

Be sure to feed the solder in as shown in the pictures. Remember: don't try to carry the solder to the joint on the tip of the iron. Heat the wires first, and then touch the solder to the wires and the tip of the iron, while keeping it in contact with the wires. Wait until the solder liquefies, and you will see it running eagerly into the joint. If this doesn't happen, apply the heat for a little longer.

The finished joint has enough solder for strength, but not so much solder that it will prevent heat-shrink tubing from sliding over it. I'll get to that in a moment.

Heat Theory

Your goal when soldering is to transfer heat from the soldering iron, into the joint that you are trying to make. For this reason, try to adjust the angle of the soldering iron so that it makes the widest possible contact. See figures 12-37 and 12-38. This is especially important if you are using thicker wire.

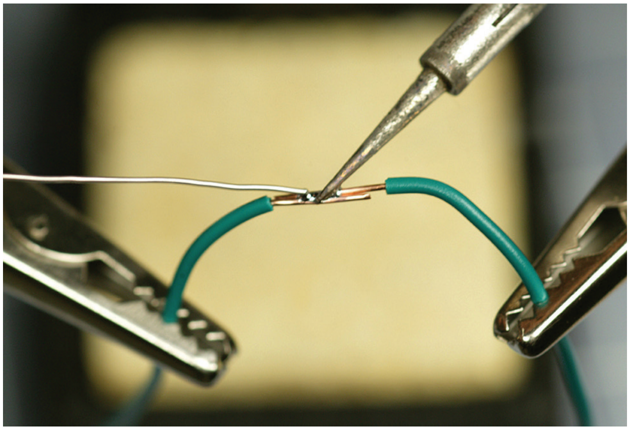


Figure 12-35. Step 4: Solder starts melting into the joint.

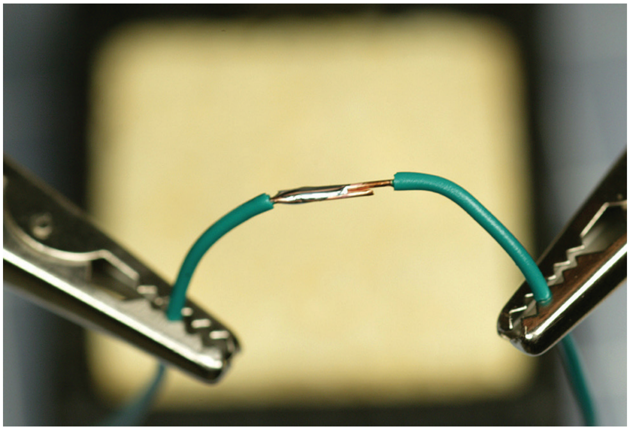


Figure 12-36. Step 5: The finished joint is shiny, and the solder has spread across the copper surfaces.

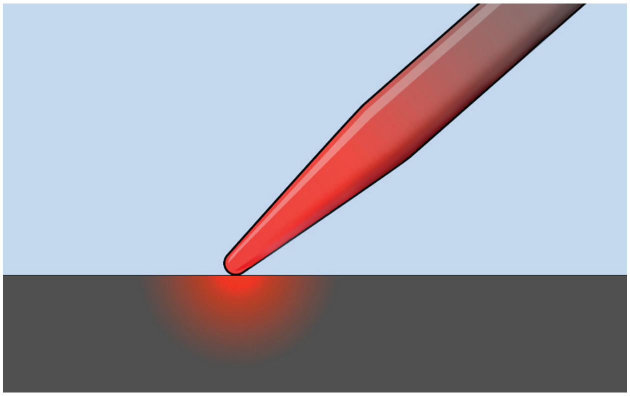


Figure 12-37. A small contact area between the iron and the working surface allows insufficient heat transfer.

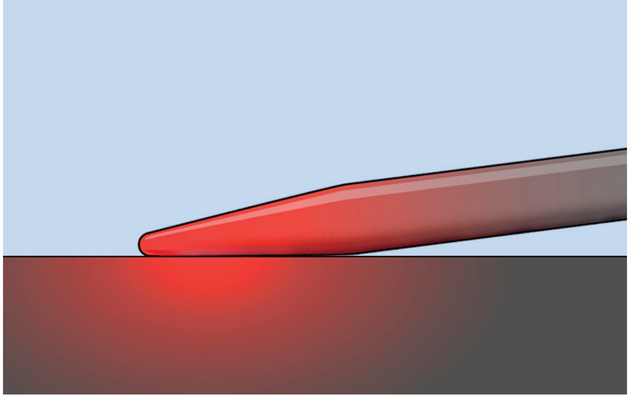


Figure 12-38. A larger area of contact increases the heat transfer.

Once the solder starts to melt, it broadens the area of contact, which helps to transfer more heat, so the process accelerates naturally. Initiating it is the tricky part.

The other aspect of heat flow that you should consider is that it can suck heat away from the places where you want it, and deliver it to places where you don't want it. If you're trying to solder a heavy piece of copper wire, the joint may never get hot enough to melt the solder, because the heavy wire conducts heat away from the joint. You may find that even a 40-watt iron isn't powerful enough to overcome this problem. At the same time, while the copper doesn't get hot enough to melt the solder, it can be hot enough to melt insulation off the wire.

As a general rule, if you can't complete a solder joint in 15 seconds, you probably aren't applying enough heat.

Heat Shrinking

If you have heat-shrink tubing and a heat gun, it will be very easy to insulate the joint that you just made between two parallel wires.

Choose some heat-shrink tubing that is just big enough to slide over the joint, cut a piece about 1" long, and slide it along until the joint is centered under it. See Figure 12-39.

Hold the tube in front of your heat gun, and switch on the gun. (Keep your fingers away from the blast of very hot

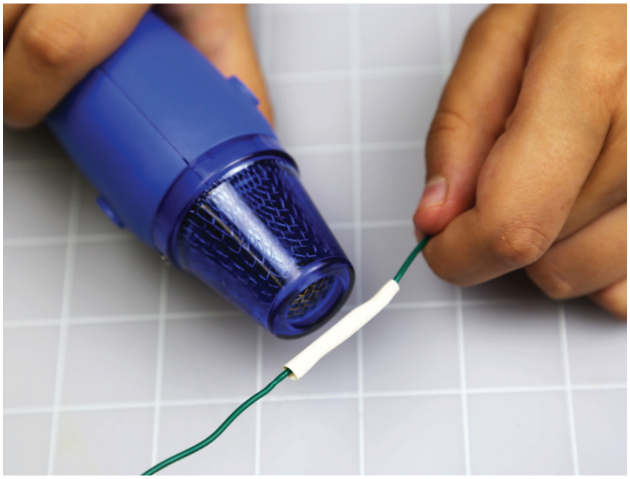


Figure 12-39. Preparing to insulate your solder joint.

Figure 12-40. The heat-shrink tubing has shrunk, and the job is done.

Figure 12-41. Members of your family should understand that a heat gun may not be substituted for a hair dryer.

air.) Turn the wire so that you heat both sides. The tubing should shrink tight around the joint within 15 seconds. See Figure 12-40.

If you overheat the tubing, it may shrink so much that it splits, at which point you must remove it by cutting it off with a utility knife before you can start over. As soon as the tubing is tight, there's no point in making it any hotter. Note that while tubing mostly shrinks at right-angles to its length, a little shrinkage also occurs along its length.

I used white tubing because it shows up well in photographs. Other colors of heat-shrink tubing generally perform the same way.

Caution: Heat Guns Do Get Hot!

If you are using a heat gun which terminates in a metal tube, it stays hot enough to burn you for several minutes after you've used it. And, as in the case of soldering irons, other people (and pets) are vulnerable, because they won't necessarily know that the heat gun is a hazard. Most of all, make sure that no one in your home ***ever*** makes the mistake of using a heat gun as a hair dryer (see Figure 12-41). Warn them about it before you start using it.

Wiring Your Power Source

Assuming you have acquired an AC adapter, how are you going to feed its current into your breadboard? You can do it the easy way, which doesn't require any soldering, or the better way, which will use the soldering skills that I described above.

I will describe both ways, but in each case your first step is to cut the plug off the end of the wire from the adapter, as shown in Figure 12-42. Needless to say, ***don't*** do this while the AC adapter is powered from a wall outlet.

Now, which of the two conductors from the adapter will be positive, and which will be negative? One of them may be marked in some way, as shown in Figure 12-43, and this will probably be positive, but "probably" is not a word that I like. Let's make sure.

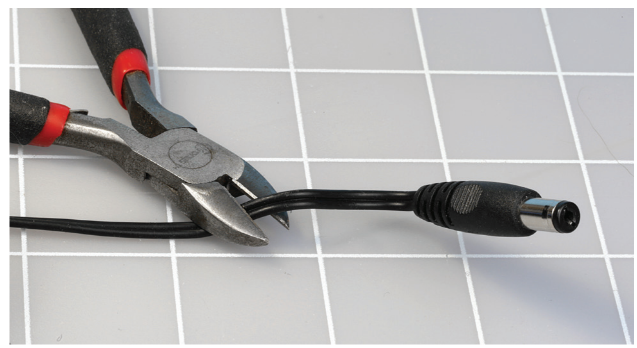


Figure 12-42. Trimming the wire from the AC adapter must be done when the adapter is not plugged into the wall.

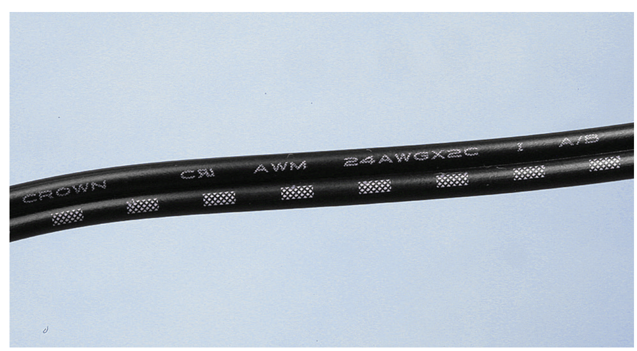


Figure 12-43. One conductor from your AC adapter may be marked, usually indicating that it is positive, but you need to make sure.

Separate the two conductors by cutting between them with a utility knife, as in Figure 12-44. Pull them apart, and strip half an inch of insulation from each, as in Figure 12-45.

Be careful to keep the wires separate. You do ***not*** want them to touch during the next step, because you have to plug in the adapter, and even though the wires will be carrying a low voltage, if they touch each other there will be a big spark that will make everyone unhappy. A good precaution is to hold the wires with alligator clips as in Figure 12-46.

Now plug in your adapter, set your meter to measure volts, and measure the voltage from the AC adapter. You simply want to know which wire is positive, and which is negative. If there is a minus sign on the meter display, swap the probes and try again.

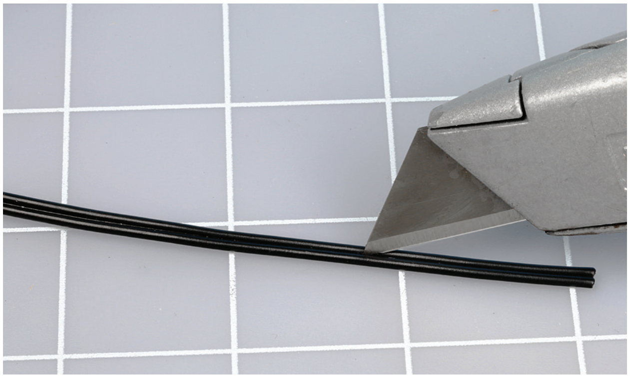


Figure 12-44. Separate the two conductors with a utility knife.

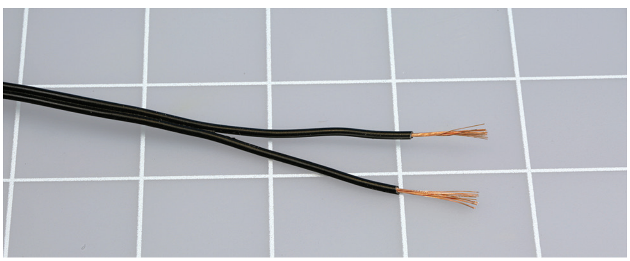


Figure 12-45. Pull the conductors apart.

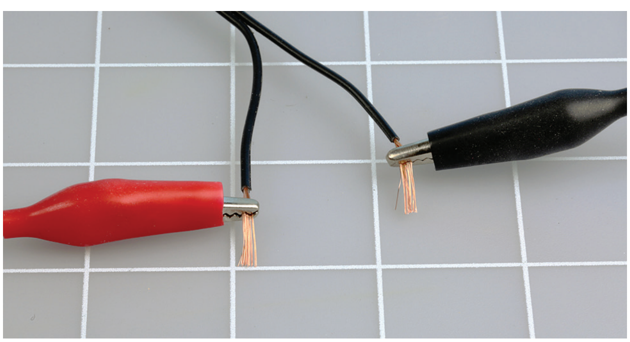


Figure 12-46. Before plugging in the AC adapter, separate the wires and keep them apart.

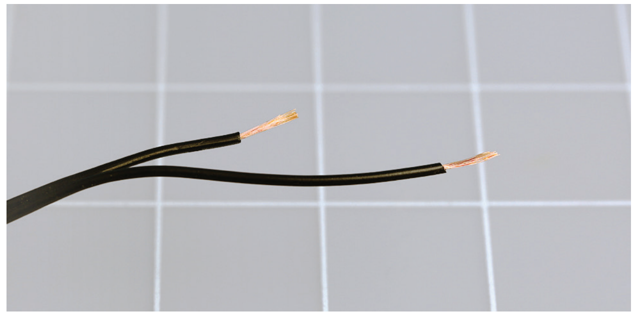


Figure 12-47. As a safety precaution, the wires from the AC adapter are trimmed to unequal lengths.

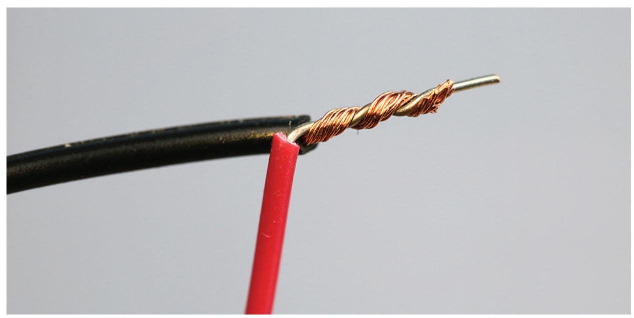


Figure 12-48. The two conductors twisted together.

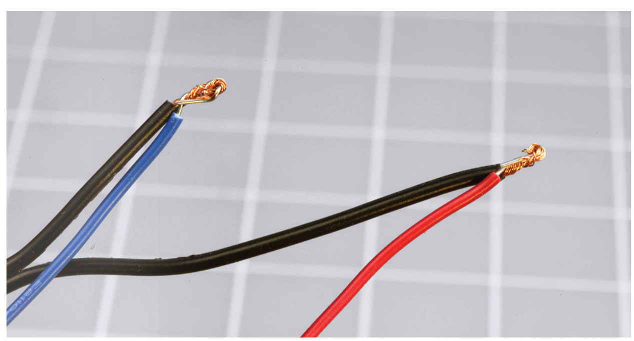


Figure 12-49. Double over the twisted wires.

When the reading on your meter is positive, you know that the red wire from your meter is on the positive side of your AC adapter. This is important, as you don't want your adapter to destroy components in your circuits by applying power to them the wrong way around.

Unplug the adapter and mark the ***positive*** conductor in some way. You can use a piece of tape or an alligator clip. Remember, the positive wire, which you identified with your red meter probe, is the one to mark.

In the previous paragraph I suggested that you should unplug the adapter. Did you do that? Please check to make sure, before continuing.

Now you want to make one wire shorter than the other, to minimize the chance of them ever touching each other if a joint that you are going to make comes apart in the future. The negative wire will be the shorter one. Chop it with your wire cutters, then strip another half-inch of insulation off it. The result should look like Figure 12-47.

Now take 3" of red hookup wire and 3" of blue or black hookup wire, which you will connect to the exposed ends of the adapter wires, either by soldering them or using wire nuts. (You did unplug the adapter, right?)

If you don't have a soldering iron:

Twist the wires together as shown in Figure 12-48. Then bend the twisted wires over to double their thickness as in Figure 12-49, so that they will fit tightly inside a wire nut in the next step. Screw on a wire nut, as in Figure 12-50, and make sure it's really tight. Add some electrical tape or any other kind of tape to keep the nut secure. I just happened to have some blue electrical tape for the negative connection.

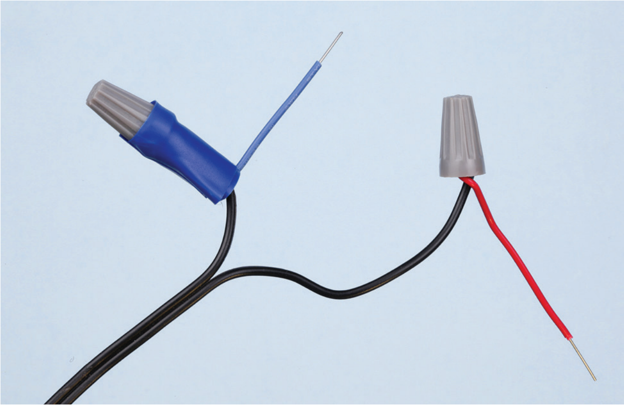


Figure 12-50. Wire nut taped for added security.

If you have a soldering iron:

Join each adapter wire to the correct color of jumper wire, using the technique described previously for soldering parallel wires. If you are doubtful about either of your solder joints, re-heat them, pull the wires apart, and try again. You don't want either of your joints to separate unexpectedly at some point in the future.

Now you need to insulate your joints. Use heat-shrink tube, if possible. If you don't have heat-shrink tube, you should be able to crimp the wires and screw wire nuts over the solder joints. Otherwise, use electrical tape, and if you don't have electrical tape, use two or three layers of scotch tape—but you must use something!

The lengths of jumper wire can now be plugged into your breadboard, and you are freed from the need to use 9V batteries.

Experiment 13
Roasting an LED

In Experiment 2, I showed you how to burn out an LED. What really happened, during that little adventure, was that excessive current passing through the LED created excessive heat, and it was the heat that killed the component.

If heat caused by electricity can destroy an LED, do you think heat from a soldering iron can do the same thing? It sounds plausible, but there's only one way to make absolutely sure.

You Will Need:

- 9V battery and connector, or 9V AC-DC adapter.
- Long-nosed or sharp-nosed pliers.
- 15-watt soldering iron.
- Optional: 30-watt soldering iron.
- LED, 3mm (1).
- 470-ohm resistor (1).
- Helping Hands to hold your work.
- Pure-copper alligator clip (1).

I recommend a 3mm LED for this experiment, as it is more susceptible to heat damage than a 5mm LED. I chose a yellow one, because it's easier to photograph than a red one. You can use any color that you happen to have.

Your purpose is to study the effects of heat, and this means you need to know where the heat is going. With this in mind, you're not going to use a breadboard, as the contacts inside the board would absorb an unknown amount of heat. I don't want you to use test leads close to the LED, either, because they too will absorb heat.

Instead, please use some sharp-nosed pliers to bend a lead from an LED into a little hook, and do the same thing with the lead on a 470-ohm resistor.

Figure 13-1 shows the setup. To minimize heat loss through conduction, the resistor dangles from one of the leads on the LED, and the power-supply wire hangs from that. Gravity should be sufficient to hold everything together.

The plastic body of the LED is gripped in an alligator clip from your Helping Hand. Plastic is not a good thermal conductor, so the lens of the LED should not lose much heat through conduction.

Apply 9V, and your LED should be shining brightly. Wait till your 15W soldering iron is fully heated, and then hold the tip of it firmly against one of the leads on your glowing LED.

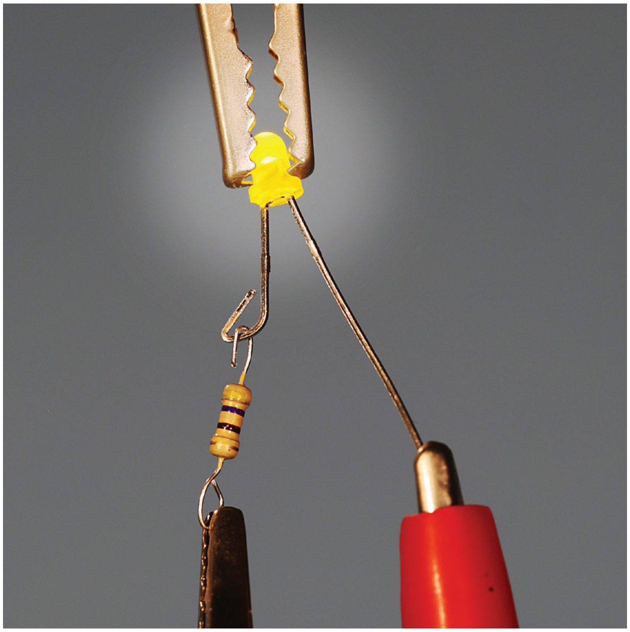


Figure 13-1. Ready to measure the heat tolerance of a 3mm yellow LED.

After about 15 seconds, you should see the LED beginning to suffer. Another 15 seconds later, and it fades to a dying ember, as in Figure 13-2.

Your LED has been sacrificed to satisfy a need for knowledge. It was an honorable death. Lay it to rest in your trash, and substitute a new LED, which we will try to treat more kindly. Connect it as before, but this time add a full-size copper alligator clip between the tip of your soldering iron and the LED. This time, you should be able to hold the iron in place for a full two minutes without burning out the LED.

Where the Heat Goes

Imagine the heat flowing out through the tip of your soldering iron, into the wire that leads to the LED—except that the heat meets the alligator clip along the way, as shown in Figure 13-3. Heat prefers to flow into the copper clip, leaving the LED unharmed, because copper conducts heat much better than the plastic capsule of the LED.

We can say that the alligator clip functioned as a ***heat sink***.

A copper alligator clip may work better than an everyday nickel-plated steel alligator clip, because copper is such a good conductor of heat.

Because burning out a component can be so annoying (especially if you don't have any spares), I play it safe and use a heat sink if I have to apply a 15-watt iron extremely close to a semiconductor for 20 seconds or more.

If you use a 30-watt soldering iron with delicate components, a heat sink is mandatory.

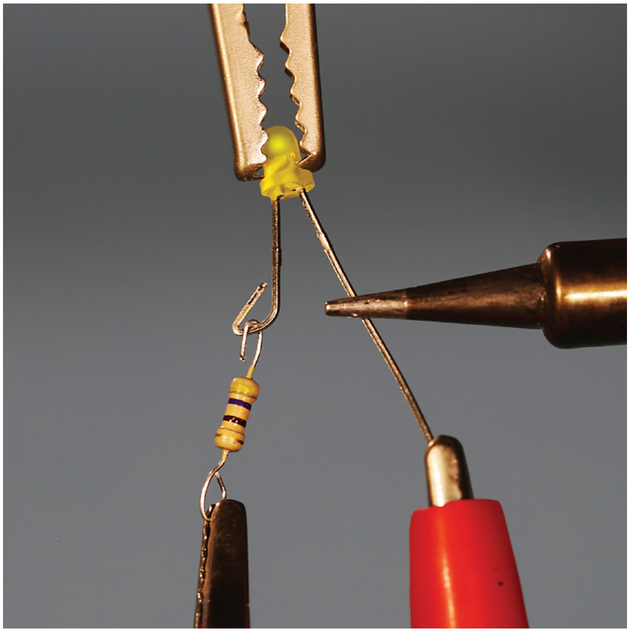


Figure 13-2. After applying heat for 15 seconds with a 15-watt soldering iron.

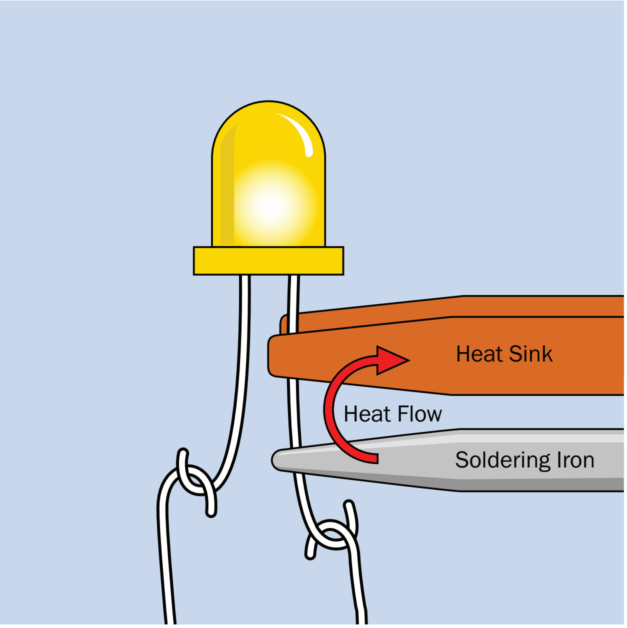


Figure 13-3. How a copper alligator clip conducts heat away from an LED.

Experiment 14
A Wearable Multivibrator

One advantage of mounting components on perforated board is that a circuit can be significantly reduced in size. It can even become wearable, if you find somewhere to put the battery. For example, if you mount the circuit on the front of a cap, you can hide the battery in the space between the top of your head and the center of the cap. In this little project, the astable multivibrator circuit from Experiment 10 is reduced to fit on a piece of perforated board measuring only 0.6" x 1".

You Will Need:

- 9V battery (or AC adapter) for testing.
- Hookup wire, wire cutters, wire strippers, multimeter.
- 15-watt soldering iron.
- Solder, .05" (1.5mm) or thinner.
- Plain perforated board (no copper plating necessary), 0.1" pitch.
- Helping Hands or similar.
- Resistors: 4.7K (2), 470K (2).
- Capacitors, ceramic: 1µF (2).
- Transistors, 2N3904 (2).
- 3mm red LEDs (2).

Adapting a Circuit to Perforated Board

The simplest way to install components on perforated board is by doing point-to-point wiring. It works like this:

1. Draw a sketch of the layout of the components, imagining how their leads can be joined together.

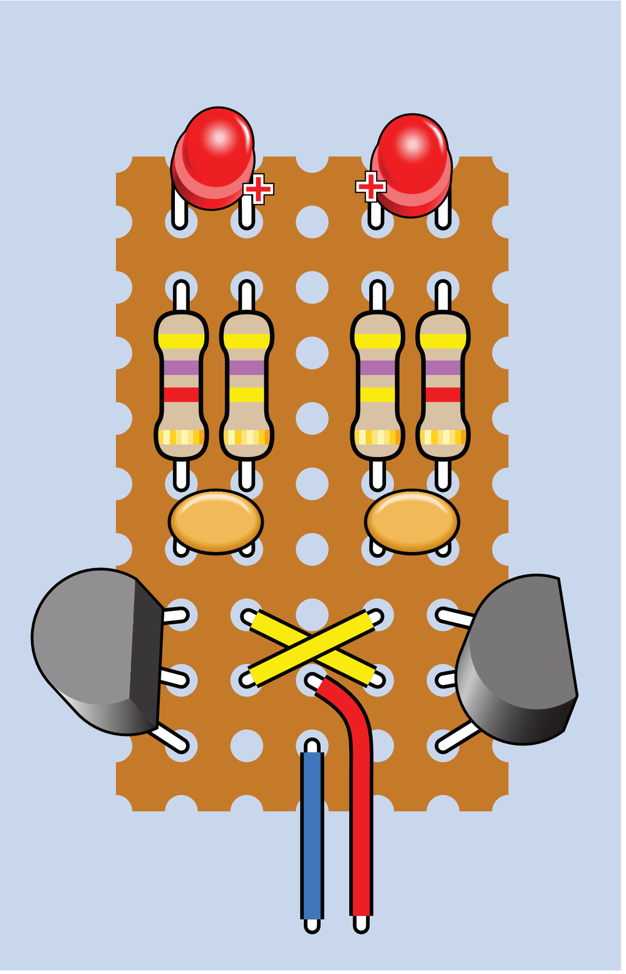


Figure 14-1. A plan for point-to-point wiring of the astable multivibrator on perforated board.

2. Push the leads through holes in the board.
3. Bend the leads so that they connect with each other on the underside of the board.
4. Make solder joints.
5. Trim the leads.

This sounds fairly easy, but there are a couple of issues to deal with.

1. The leads under the board will not be insulated, so you have to avoid crossovers. Insulated wires can be added on the upper side of the board, if crossovers are unavoidable.

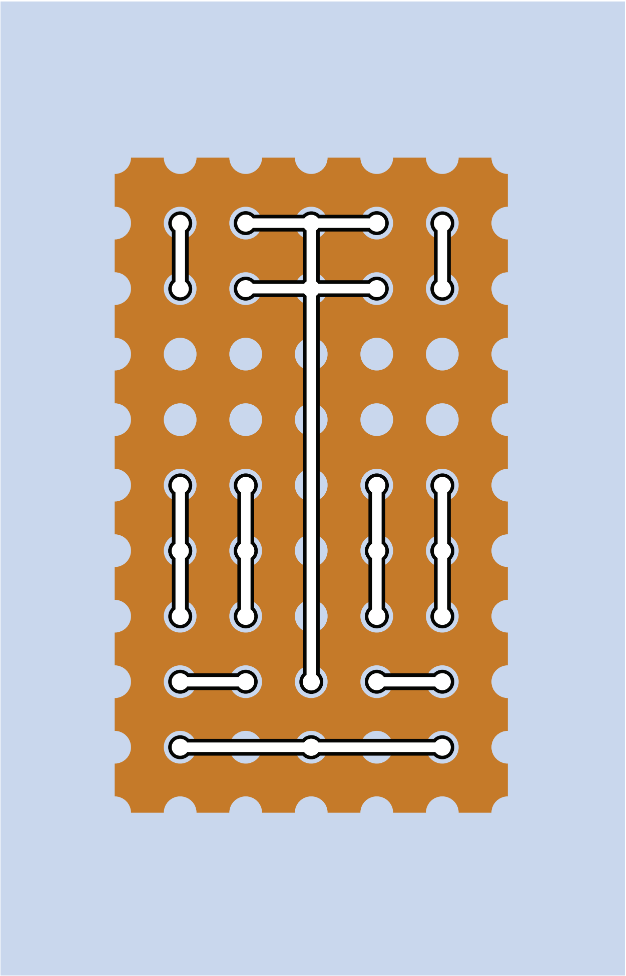


Figure 14-2. The connections underneath the components in Figure 14-1.

2. If you want to make a pleasingly compact circuit, you will be working at a very small scale. You will need steady hands, and you will have difficulty if your soldering iron has a relatively large tip.

The astable multivibrator circuit in Figure 11-2 is ideal for building on perforated board, as the components don't have to be rearranged significantly for a perf-board layout. Figure 14-1 shows the layout that I worked out. Bear in mind that the perforated board has 0.1" pitch, meaning that the holes are only 0.1" apart.

All of the leads poke through the holes and will be joined underneath as shown in Figure 14-2. This is a particularly easy circuit to visualize, because it is symmetrical.

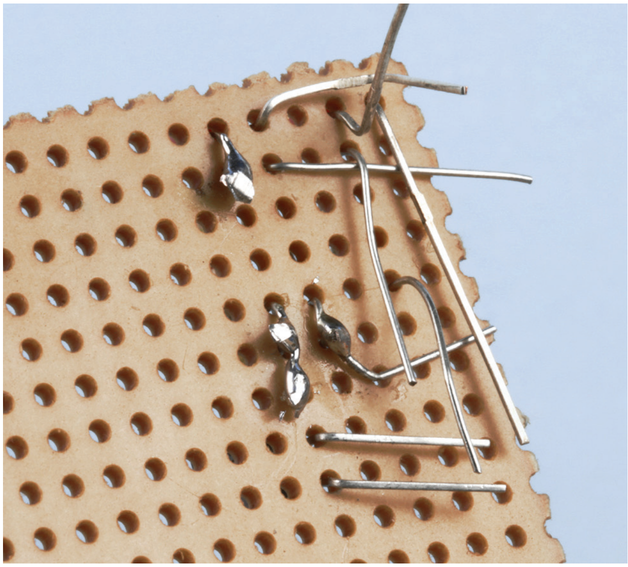


Figure 14-3. Soldering leads underneath the perforated board.

Most circuits are asymmetrical, and you may have difficulty imagining them flipped left-to-right when you turn the board over. This is another reason to make a careful drawing first, and if you can photograph the drawing and flip it with image-editing software, you'll see how it should look on the back side of the board.

Compare figures 14-1 and 14-2, and you should be able to follow the connections to verify that they are the same as in the schematic in Figure 11-2.

Now for the hard part: Building it.

The Process

Figure 14-3 shows the work in progress. Notice that I am building the circuit in the corner of a larger piece of board, so that the alligator clips on my Helping Hands can grip the board easily. I won't trim the board till the whole circuit is finished and has been tested.

My procedure is to push the leads of four or five components through holes in the board, and bend the leads so that the components won't fall out when I turn the board over. Then I refer to my sketches of the connections (as in Figure 14-2). I solder a few pairs of leads and trim them, then push more leads through. I stop often to examine the solder joints with a closeup magnifying glass, because mistakes are so easy to make.

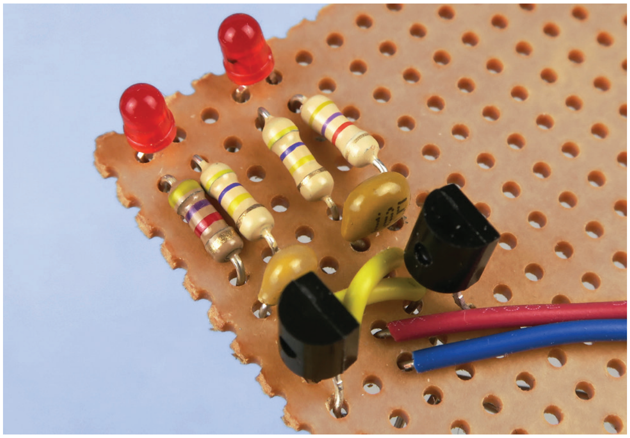


Figure 14-4. All components have been installed.

Figure 14-4 shows the upper side of the board with all the components installed. Figure 14-5 shows the underneath of the board after I trimmed it to its final size. This is not a pretty sight, but I have never claimed to be good at detail work. I hope that by revealing my result, it will encourage you to feel that you can do better.

Figure 14-6 shows the finished job. I happen to have a band saw, which is ideal for trimming perforated board, but a hacksaw is almost as easy. Naturally you should grip the unused portion of the board in a vise, not the section with components mounted on it. And, test the circuit before you trim the board.

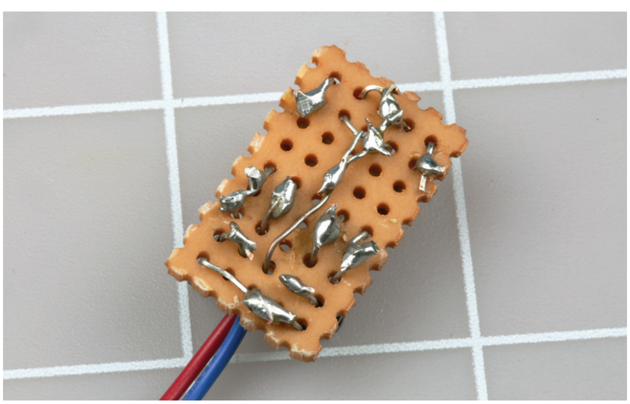


Figure 14-5. Soldering has been completed and the board has been trimmed.

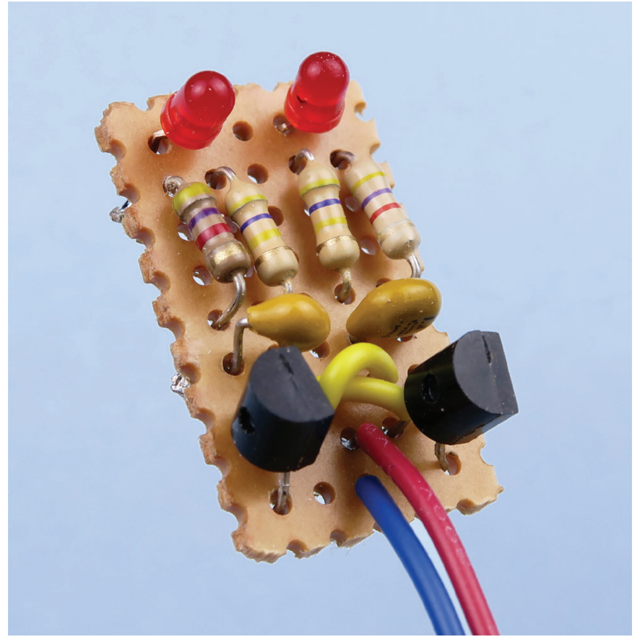


Figure 14-6. The finished astable multivibrator installed on perforated board.

Easier Options

You may feel that joining wires together is too much of a hassle. Isn't there a better way?

Yes, you can buy perforated board that has various patterns of copper traces, such as the breadboard-emulation layout in Figure 12-17. You solder each lead into its hole, and the copper trace makes the connection for you to the next lead. One lead only requires one solder connection, instead of two or more leads sharing one connection. This is easier to handle, and the result is much neater—but, as I noted previously, it won't be as compact.

Another option is to create your own custom-fabricated printed circuit board, but that goes beyond the scope of this book.

Fixing Wiring Errors

I got lucky with my point-to-point wiring for the astable multivibrator: I didn't make any wiring errors, and it worked first time. If you do make an error, how do you undo a solder joint?

Clean the tip of your soldering iron, and use it to remove as much solder as possible. You can also try using the rubber-bulb solder pump shown in Figure 12-8, or the braid in Figure 12-9, but I have expressed my skepticism about these devices.

Probably you will have to use pliers to grab one wire while you melt the remaining solder. Then you can pull the wire free.

Wiring errors are not the only factor that can prevent a circuit from working. I already mentioned problems caused by inadequate heat or solder, but too much solder can cause its own problems. If it has created a bridge between wires that should not be connected, use a utility knife to make two parallel cuts in the solder, and scrape away the little section between them.

All of these tasks will be easier if you have bright lighting.

Caution: Flying Wire Segments

The jaws of your wire cutters exert a powerful force that peaks and then is suddenly released when they cut through wire. This force can be translated into sudden motion of a snipped wire segment. Some wires are relatively soft, and don't pose a risk, but transistors and LEDs tend to have harder wires. Little segments can fly in unpredictable directions at high speed, creating a hazard for your eyes when you are doing close-up work.

Everyday eyeglasses can protect you when trimming wires. If you don't use eyeglasses, plastic safety glasses are a sensible precaution.

Maddened by Measurement

I'm going to digress here to a topic that I have avoided so far: Units of measurement.

Because I happen to live in the United States, mostly I use measurements in inches—yet I have referred to LEDs as being 5mm or 3mm in diameter. This isn't inconsistency on my part; it reflects the conflicted state of the electronics industry, where you'll find inches and millimeters both in daily use, often in the very same data sheet. For instance, the pin spacing of surface-mount chips tends to be measured in millimeters, but through-hole DIP chips still have pins 0.1" apart, and probably always will.

To complicate matters, where inches are used, two different systems coexist in the United States. Metal thicknesses, for instance, may be measured with the decimal-inch system, which causes a sheet of steel to be described as .016" thick. Drill bits, however, are measured with what I will refer to as the fractional-inch system, in increments of 1/64". A feature of this system is that when you have two 64ths, it's 1/32", and when you have two 32nds, it's 1/16"—and so on. Thus, 6/64" is written as 3/32" while 8/64" would be 1/8".

People in almost all other countries migrated to the metric system long ago, but the United States seems reluctant to change, and therefore I have created four charts on the next two pages to assist you in converting from one system to another.

The chart in Figure 14-7 shows you the fractional-inch equivalents to hundredths of an inch.

Figure 14-8 shows decimal-inch equivalents to the fractional-inch system, because you are quite likely to encounter a measurement such as 0.375", and if you know that this is the same as 3/8", the knowledge can be useful.

Many datasheets provide measurements in both millimeters and inches, but some now use millimeters only. Figure 14-9 enables conversion between millimeters, hundredths of an inch, and fractions of an inch. From this chart you can see that if you need to drill a hole for a 5mm LED, a 3/16" bit is about right (in fact, it results in a tighter fit than if you drill an actual 5mm hole).

A magnified version of the previous chart is shown in Figure 14-10, using tenths of millimeters and thousandths of an inch. This confirms that if you want a slide switch with pins that will fit into a breadboard, 2.5mm spacing is only a tiny bit less than 0.1".

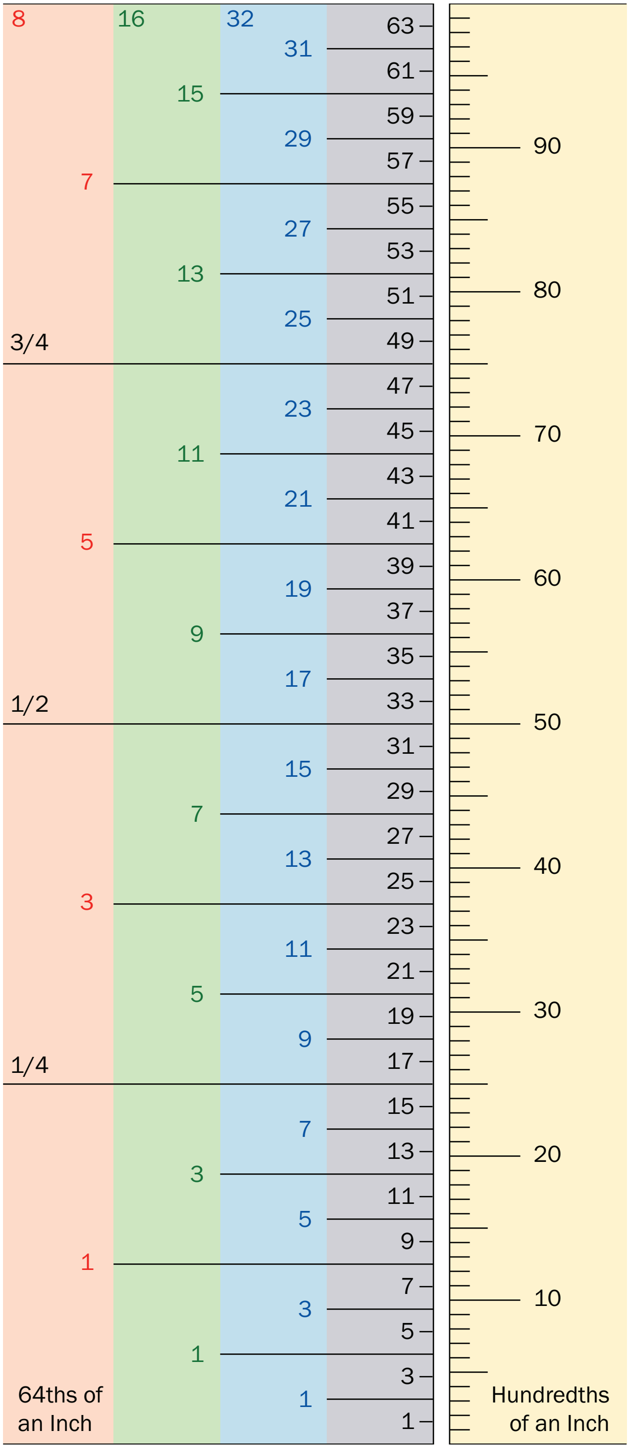


Figure 14-7. Conversion between fractions of an inch and hundredths of an inch.

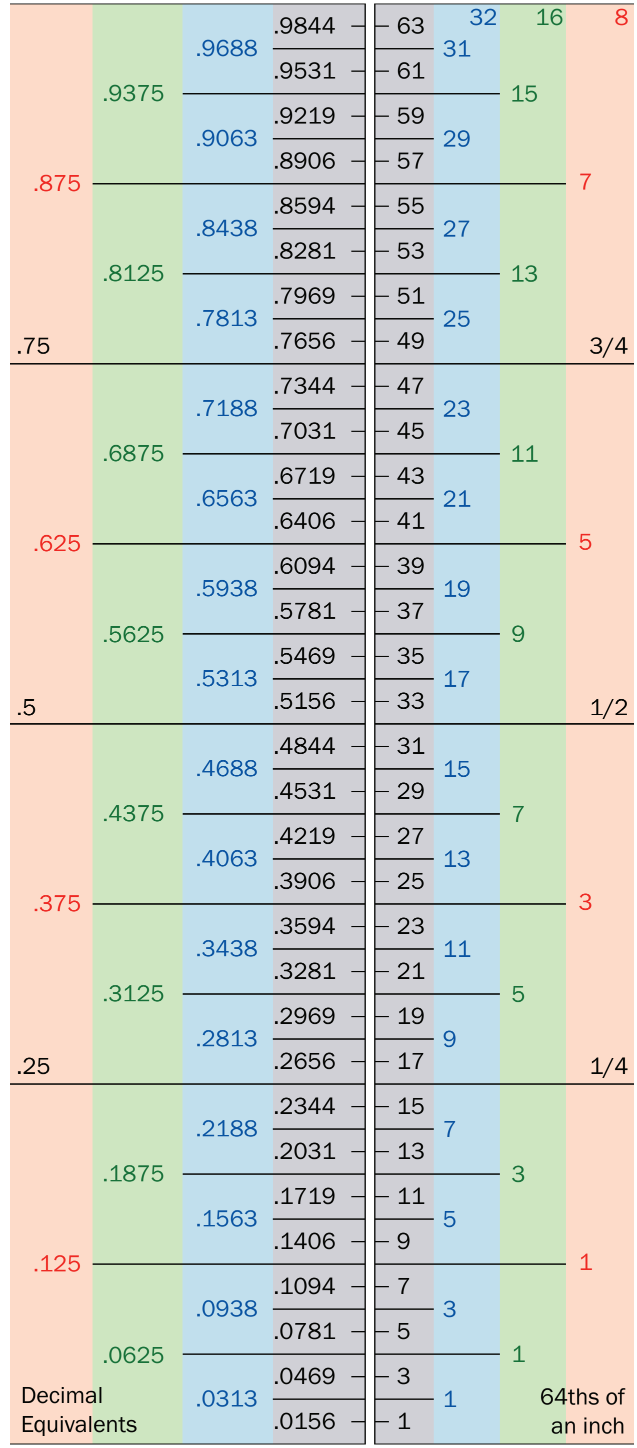


Figure 14-8. Conversion between the decimal-inch system and fractions of an inch.

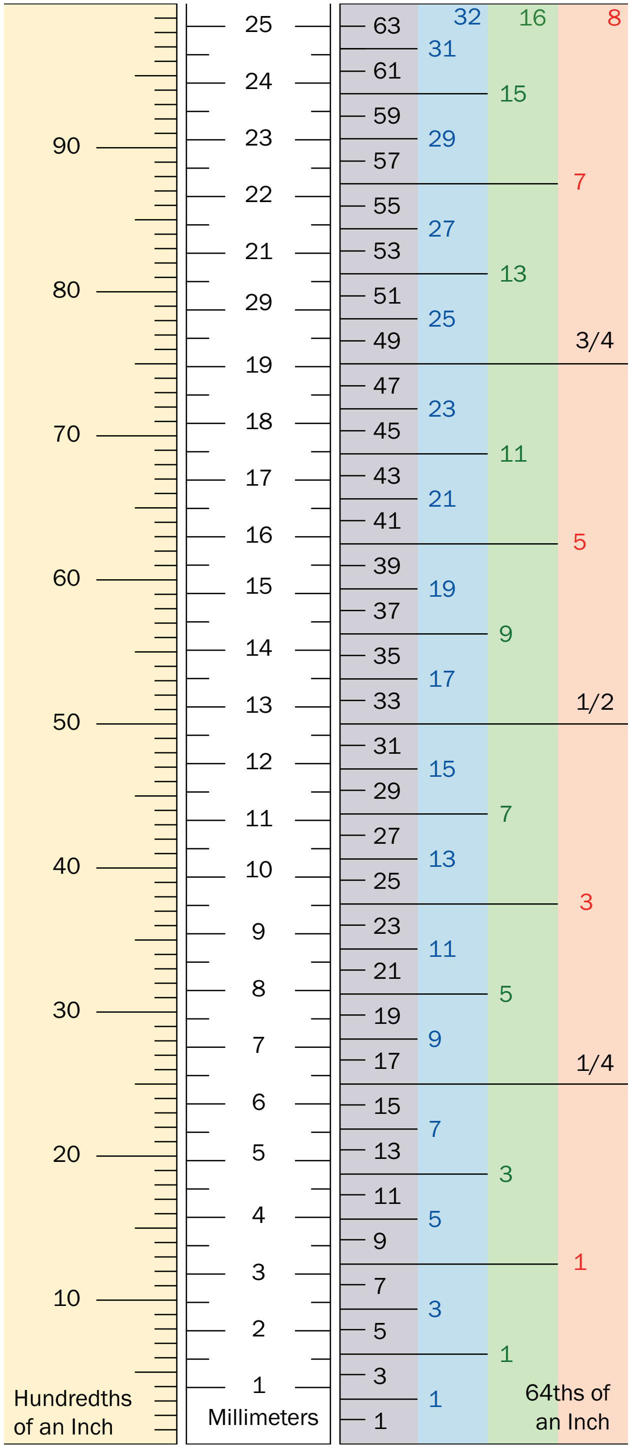


Figure 14-9. Conversion between hundredths of an inch, millimeters, and fractions of an inch.

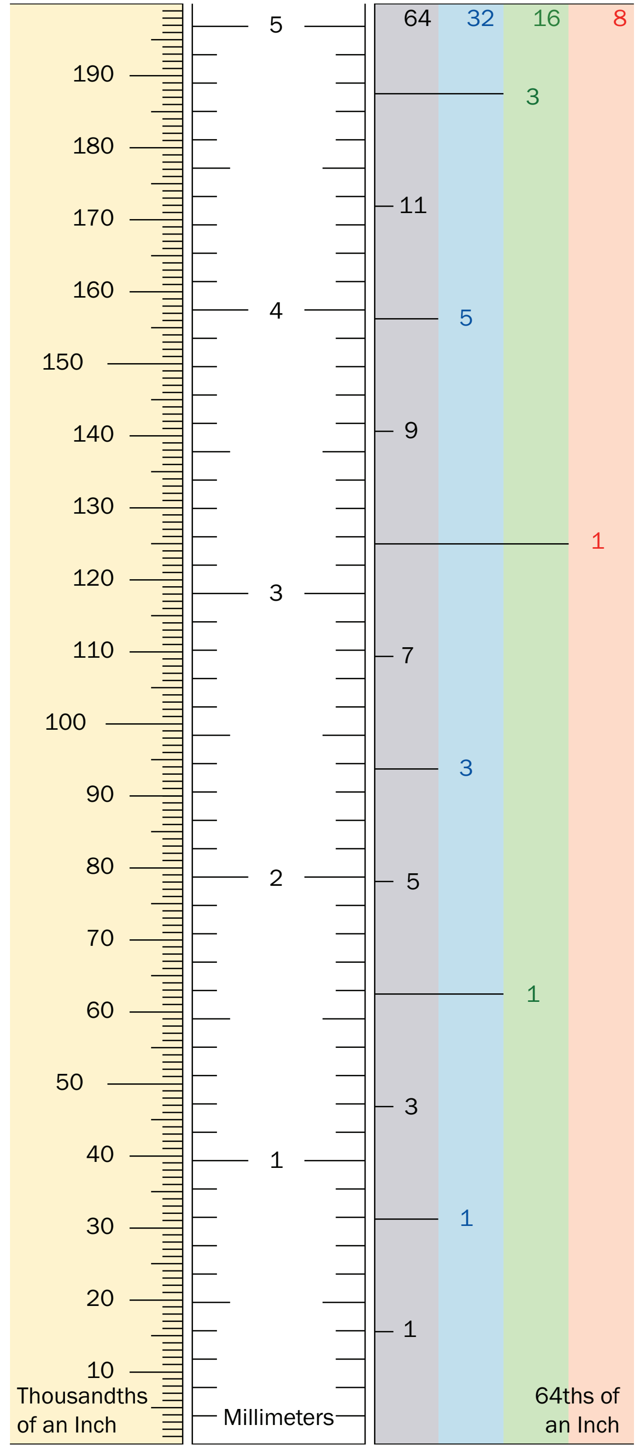


Figure 14-10. Conversion of smaller dimensions using 1,000ths of an inch, tenths of millimeters, and fractions of an inch.

Section Four

Chips, Ahoy!

Before I get into the fascinating topic of integrated circuit chips (often referred to as *ICs* or simply *chips*), I have to mention that some of the things I asked you to do in previous experiments could have been done more quickly if you had used chips.

Does this mean you have been wasting your time? No, absolutely not, because you still need to know how resistors, capacitors, and transistors interact with each other. These basic parts, properly known as *discrete components*, will continue to be used in circuits throughout the rest of the book.

Chips can simplify circuits, though, and can be addictive to play with, although you may not become quite as excited as the character in Figure 15-1.

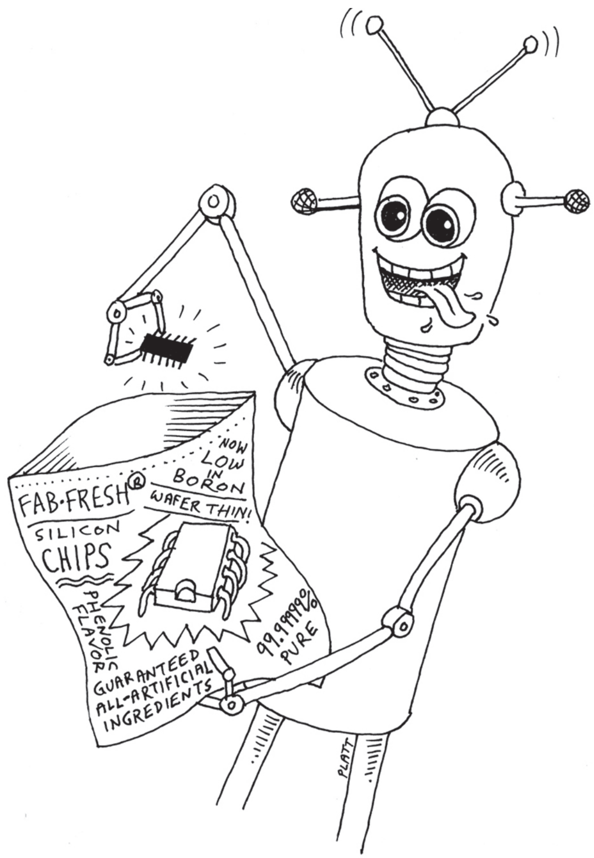


Figure 15-1. My role model.

The tools, equipment, components, and supplies described below will be useful in experiments 15 through 24, in addition to the items which I have recommended previously.

Choosing chips

Figure 15-2 shows two integrated circuit chips. The one at the top has pins spaced at intervals of 0.1" so that they will fit through the holes in your breadboard or perforated board, and for this reason it is often referred to as a *through-hole* design. I will be using this format exclusively, because it is easy to handle. The smaller chip is a *surface-mount* design, which I will not be using, because it doesn't fit breadboards or perforated boards and is difficult to handle. (Many surface-mount chips are now even smaller than the one shown here.)

The body of a chip is known as the *package*, and is made of plastic or resin. A through-hole chip is usually sold in a *dual-inline package*, meaning that it has a line of pins along each side. The acronym for it is *DIP* or (when it is made of plastic) *PDIP*.

Surface-mount packages often are identified with acronyms beginning with letter S, as in *SOIC*, meaning "small-outline integrated circuit." Numerous surface-mount variants exist, but they are all outside the scope of this book, and if you buy your own components, you should be careful not to select surface-mount chips by mistake. Your guidelines:

- Avoid all chip formats that begin with letter S, such as SMT or SMD. Also avoid part numbers beginning with SO, SOIC, or SSOP, and similar prefixes such as TSSOP or TVSOP. These are all surface-mount formats that require special handling.

Hidden inside the package is a circuit etched onto a tiny wafer of silicon. This is where the term "chip" comes from, although the whole component is now usually referred to as a chip, and I will follow that convention here. Tiny wires inside the package link the circuit with the rows of pins that protrude on either side.

The PDIP chip in Figure 15-2 has seven pins in each row, making a total of 14. Other chips may have 4, 8, 16, or more pins.

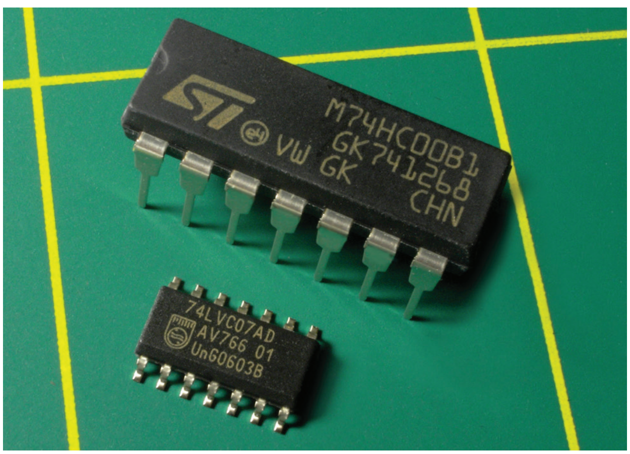


Figure 15-2. A through-hole chip (top) and surface-mount chip (bottom).

Just about every chip has its part number printed on it. Notice in the photograph that even though the chips look quite different from each other, they both have "74" in their part numbers. This is because both of them are logic chips that were assigned numbers from 7400 and upward when they were introduced several decades ago. They are referred to as members of the *74xx family*, and are still extremely useful.

You need to understand part numbers, so I have provided a quick summary in Figure 15-3. The initial letters identify the manufacturer—which you can ignore, as it makes no difference for our purposes. Then you get to 74, which is the "family," after which you find two more letters (sometimes three), which are the "generation" of the family. The 7400 family has evolved through many generations, including 74L, 74LS, 74C, 74HC, and 74AHC. There are more, but I will be using the HC generation because almost all 7400 chips are available in this form, they don't cost too much, and they don't use a lot of power. Later generations offer additional speed, but for our purposes, this is unnecessary.

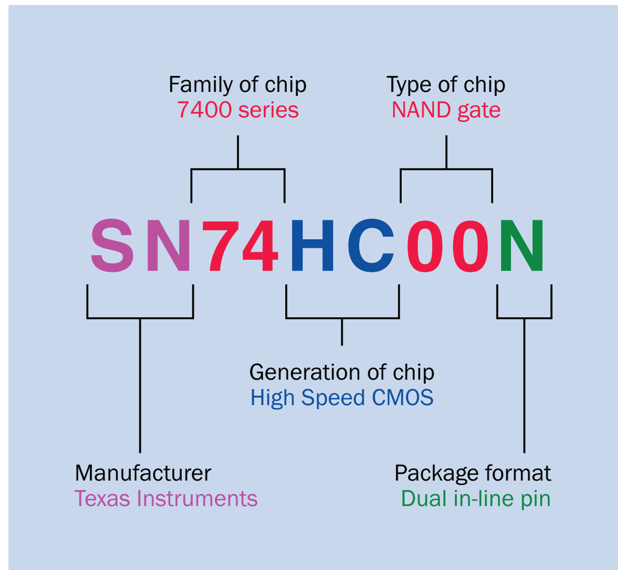


Figure 15-3. Understanding the part number of a chip in the 7400 family.

After the letters identifying the generation, you'll find a sequence of two, three, four, or (sometimes) five digits. These identify the specific purpose and function of this particular chip. If you ever want to browse through a complete list, search for

```
list 7400 integrated circuits
```

and you can start by clicking the hit for a Wikipedia entry.

Finally, the part number will terminate with another letter, or two letters, or more. Letter N means that the chip is DIP format, which is important information if you are not able to see a picture of it.

The purpose of this long explanation is to enable you to interpret catalog listings if you go chip shopping. You can search for "74HC00" and the search engines at online vendors are usually smart enough to show you appropriate chips from multiple manufacturers, even though the complete part number would have letters preceding and following your search term.

All the chips needed for the experiments in this section of the book are listed in the tables of components in Appendix A.

IC Sockets

If you plan to immortalize any of your circuits in solder, I suggest you avoid soldering the pins of chips directly, because you will have a hard time desoldering multiple pins if you make a wiring error. You can eliminate the risk if you buy some ***DIP sockets*** such as those shown in Figure 15-4.

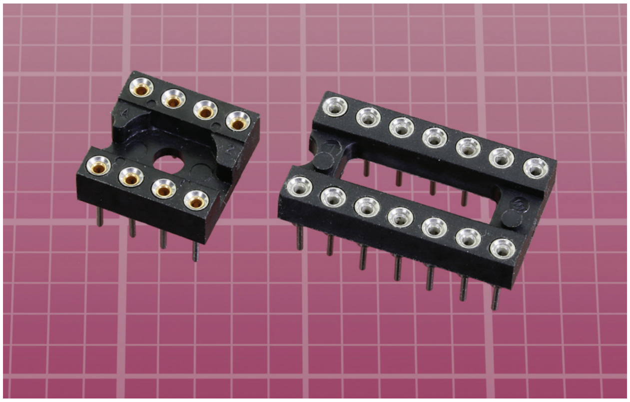


Figure 15-4. Sockets will protect you from the risk of soldering a chip incorrectly.

First you solder the sockets into the board, then you plug the chips into the sockets, and no additional soldering is required. For the projects in this book, you can use the cheapest sockets you can find (you don't need to pay extra for gold-plated contacts).

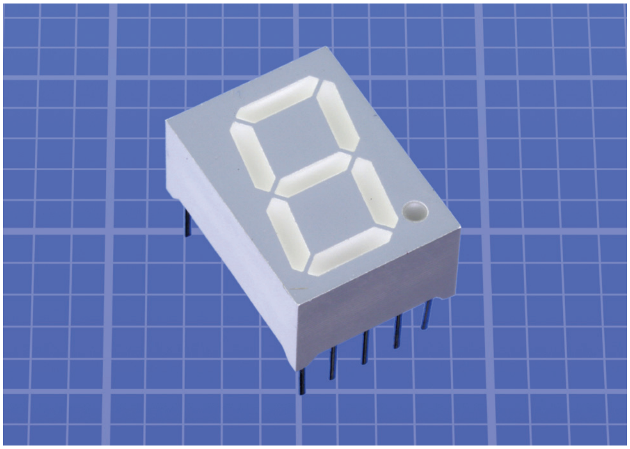


Figure 15-5. A 7-segment LED display.

Numeric Displays

One of my chip projects will display its output using 7-segment ***numeric displays***—the style of digits that you still see on some digital clocks and microwave ovens. An example of a 7-segment LED display is shown in Figure 15-5. For purchasing information, see Appendix A.

Voltage Regulator

Because many logic chips require 5VDC, you will be adding a ***voltage regulator*** to your AC adapter or battery. The regulator accepts any input voltage ranging from 7.5VDC to 12VDC and delivers exactly 5VDC as its output.

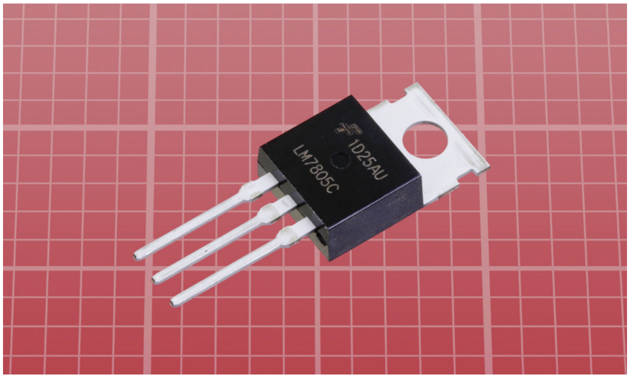


Figure 15-6. This voltage regulator delivers 5VDC when it is supplied with any voltage ranging from 7.5VDC to 12VDC.

The LM7805 is not the most modern regulator, but it is rugged and affordable. The part number will be preceded or followed with an abbreviation identifying the manufacturer and package style, which you can ignore, so long as the package looks like the one in Figure 15-6.

Diodes

One basic component that has not been required in the experiments so far is the humble ***diode***. Like an LED (which you will recall is a light-emitting diode), it passes electric current in one direction but blocks it in the opposite direction. Unlike an LED, it can be ruggedly built: A large diode can withstand significant forward current and can be subjected to substantial reverse bias so long as the voltage is within its specification.

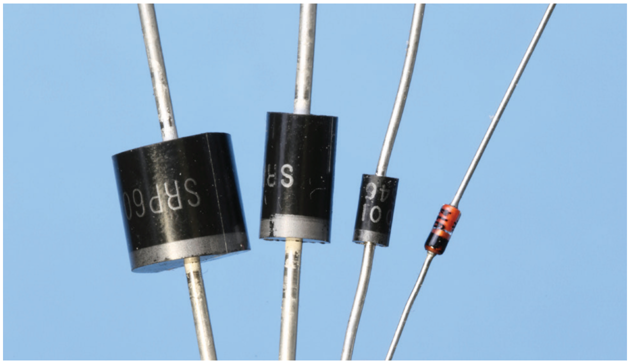


Figure 15-7. A selection of diodes.

The end of the diode that should be more-negative is known as the ***cathode*** and is marked with a line, as shown on the samples in Figure 15-7. Forward current enters through the other end, which is known as the ***anode***.

Low-Power LEDs

HC series logic chips are not designed to deliver current beyond 4mA—according to their datasheets. In reality, according to a friend in the semiconductor industry, they can source maybe twice as much current, but their specifications have never been updated to reveal this.

Based on my own experience, I am virtually certain that if you want to drive an LED from the output pin of a chip in the 74HCxx series, and your series resistor allows 10mA to flow, the chip will survive. What I don't know is how much the voltage of the output pin will be pulled down by the excessive current.

This matters because if you are using the output of one 74HCxx chip to drive the input of a second 74HCxx chip, the second chip will be trying to determine if the voltage level is "logic low" or "logic high," and if some other component (such as the LED) is sharing the output and pulling down the voltage, misunderstandings between chips may occur.

Therefore I made the following arbitrary decision for circuits in this book: Output from logic pins in 74HCxx chips will not exceed 5mA. If you nudge this up a bit because you want your LEDs to be brighter, everything will probably work out all right, but I can't guarantee it.

You may think to yourself, "Well, if the current will be low, maybe I'll buy some of those special low-current LEDs that I've read about."

Go ahead! But you may be disappointed. I tested a wide variety of LEDs, and while those that were described as "low-current" did function at a low current, their light output was miserable. I found that high-intensity red LEDs actually performed better with a current down around 2mA. Light intensity is measured in ***millicandela*** (abbreviated as ***mcd***), and LEDs rated for 400 mcd or more should provide good light output.

Either way, LEDs that are red tend to require less current, at a lower forward voltage, than comparable LEDs of blue or white. I suggest you stick to red LEDs when you want to power them with logic chips.

And, one last consideration: Some of the chip-based circuits in the remainder of the book will be quite crowded with components, leaving barely enough room for 5mm LEDs. Therefore, from this point in the book onward, I recommend using 3mm LEDs in the experiments.

The History of Chips

Before I suggest some hands-on experiments with chips, I want to tell you where they came from.

The concept of integrating solid-state components into one little package originated with British radar scientist Geoffrey W. A. Dummer, who talked about it for years before he attempted, unsuccessfully, to build one in 1956. The first true integrated circuit wasn't fabricated until 1958 by Jack Kilby, working at Texas Instruments. Kilby's version used germanium, as this element was already in use as a semiconductor. But Robert Noyce, pictured in Figure 15-8, had a better idea.

Born in 1927 in Iowa, Noyce moved to California in the 1950s, where he found a job working for William Shockley. This was shortly after Shockley had set up a business based around the transistor, which he had co-invented at Bell Labs.

Noyce was one of eight employees who became frustrated with Shockley's management and left to establish Fairchild Semiconductor. While he was the general manager of Fairchild, Noyce invented a silicon-based integrated circuit that avoided the manufacturing problems associated with germanium. He is generally credited as the man who made integrated circuits possible.

Early applications were for military use, as Minuteman missiles required small, light components in their guidance systems. These applications consumed almost all chips produced from 1960 through 1963, during which time the unit price fell from around $1,000 to $25 each, in 1963 dollars.

Figure 15-8. Robert Noyce, who patented the integrated circuit chip and cofounded Intel.

In the late 1960s, medium-scale integration chips emerged, each containing hundreds of transistors. Large-scale integration enabled tens of thousands of transistors on one chip by the mid-1970s, and some chips today contain 50 billion transistors (probably more, by the time you read this).

Robert Noyce eventually cofounded Intel with Gordon Moore, but died unexpectedly of a heart attack in 1990. You can learn more about the fascinating pioneers of integrated circuit design at the Silicon Valley Historical Society:

`www.siliconvalleyhistorical.org`

Experiment 15
Emitting a Pulse

I'm going to begin with the most successful chip ever made: the *555 timer*. You can find numerous guides to it online, but I have three reasons for including it here:

It's fundamental. You simply have to know about this chip. Until relatively recently, some sources estimated that more than 1 billion were being manufactured annually. I will be using it in one way or another in most of the remaining circuits in this book.

It's useful. The 555 is probably the most versatile chip that exists. Its relatively powerful output (delivering up to 200mA) can drive significant loads such as relays and small motors, and the chip itself is not easily damaged.

It's misunderstood. I have read literally dozens of guides, beginning with an early Signetics datasheet and making my way through various hobby texts, and have found that the inner workings of the chip are seldom explained in detail. I want you to have a clear mental picture of what happens inside a 555 timer, because this will help you to use the chip creatively.

You Will Need:

- Breadboard, hookup wire, wire cutters, wire strippers, multimeter.
- 9VDC power supply (battery or AC adapter).
- Resistors: 470 ohms (1), 10K (3).
- Capacitors: 0.01µF (1), 10µF (1).
- Trimmer: 500K (1).
- 555 timer chip (1).
- SPDT slide switch (1).
- Tactile switch (2).

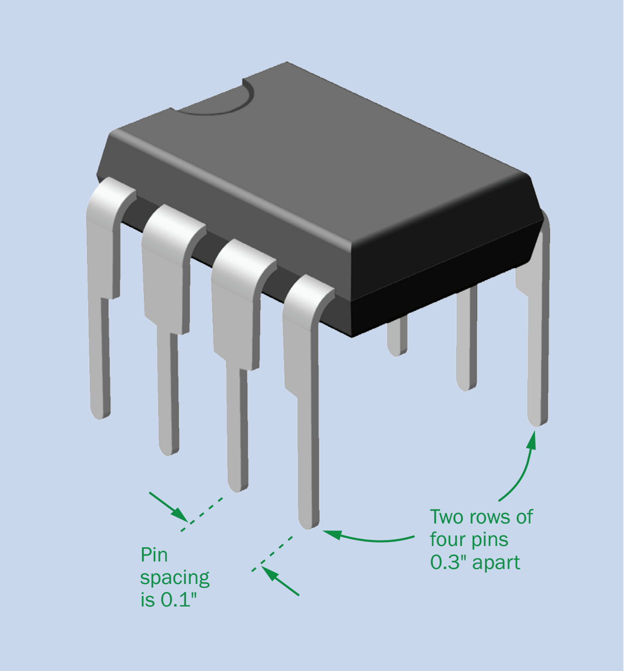


Figure 15-9. 555 timer chips are available in a through-hole 8-pin DIP package.

Recognize Your Chip

You will be using the most common through-hole package for a 555 timer: An 8-pin DIP, as shown in Figure 15-9. It is slightly less than half-an-inch square.

A part number will be printed on your 555, such as NE555P or KA555. The two-letter prefix identifies the manufacturer (Texas Instruments or On Semiconductor, in these examples), while a letter at the end tells you more about this particular version of the product, such as its temperature range.

Figure 15-10 on the next page shows information about the pin functions of a 555 chip, assuming that the timer is viewed from above. Labels like these are often referred to as *pinouts* for a chip.

In all of the chips that I will be using, the pins are numbered beginning with 1 in the top-left corner (when seen from above), and the numbering continues counter-clockwise. But how do you know which way around to hold the chip? A notch in the plastic identifies the top end, and you must orient it toward the top of your breadboard. You may also find a circular dimple adjacent to Pin 1, but some manufacturers omit this feature.

- The notch in the package of every through-hole chip must point toward the top of your breadboard.

Usually (but not always) the horizontal spacing between the two vertical rows of pins in a DIP chip is 0.3", which means that it sits neatly across the channel down the middle of a breadboard, and the conductors inside the breadboard allow access to each pin of the chip. Yes, *that's* why a breadboard is designed that way.

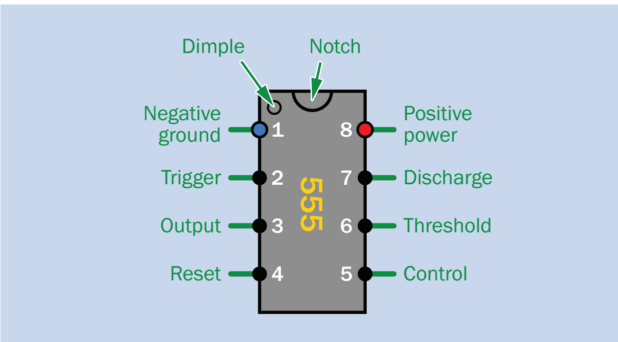


Figure 15-10. Pinouts of a 555 timer.

Monostable Test

Figure 15-11 shows the first circuit which I would like you to build, to test the features of the chip when it is running in *monostable* mode. In that mode, when the timer is triggered by a change in voltage on one of its pins, it creates a single pulse from another pin.

The timer can also be wired to function in *astable* mode, emitting a stream of pulses like the astable multivibrator that you built in Experiment 10, but I won't explore this till Experiment 16.

- In this experiment I am starting to use the dual buses of your breadboard, because they make the wiring much simpler. With this in mind, Figure 15-11 includes two long jumpers at the top of the board to distribute positive and negative power along both sides. Please be careful when running power to any chips that you use, as you can cause permanent damage if you don't observe polarity correctly.

When wiring this circuit, be careful to include the little green jumper that is only 0.1" long, near the right edge of the chip. Count holes carefully when inserting components. Attach your power supply, and if nothing happens, the timer is waiting for you to trigger it.

Turn the 500K trimmer to the middle of its range. Now press and release the top button, and the LED should light up. It stays on for a fixed period, after which it turns itself off.

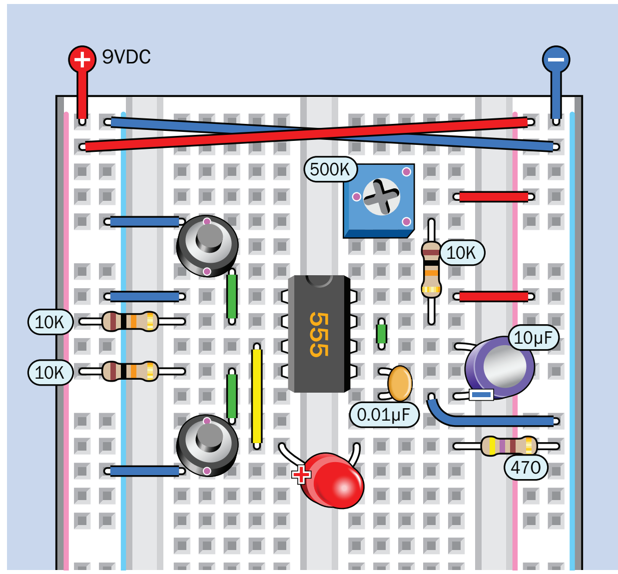


Figure 15-11. Testing the 555 timer in monostable mode.

Look back at Figure 15-10, and you'll see that Pin 2 of the timer is called the Trigger Pin. In Figure 15-11 you can see that it is connected with the positive bus on the left side of the board, through a 10K resistor. Maybe you remember, I mentioned the concept of a pullup resistor in Section Two. Well, here is an example. Its task is to maintain the Trigger Pin of the timer at the positive supply voltage—until the top button is pressed, which makes a direct connection with negative ground, overriding the 10K resistor. The drop in voltage triggers the timer, which sends a positive pulse out from Pin 3, through the yellow jumper, illuminating the LED.

- Positive voltage on the Trigger Pin prevents the timer from being triggered.

- Grounding the Trigger Pin triggers the timer.
- The LED will continue to glow for a fixed time after you press and release the button.

In Figure 15-12 you'll find the same circuit shown as a schematic, with parts labeled so that I can refer to them easily. You can set the length of the pulse in this circuit by adjusting the trimmer, P1. You can also try substituting a different capacitor for C1, because—as you may have noticed—C1 accumulates a charge through P1 + R1, which you can recognize as being an RC network.

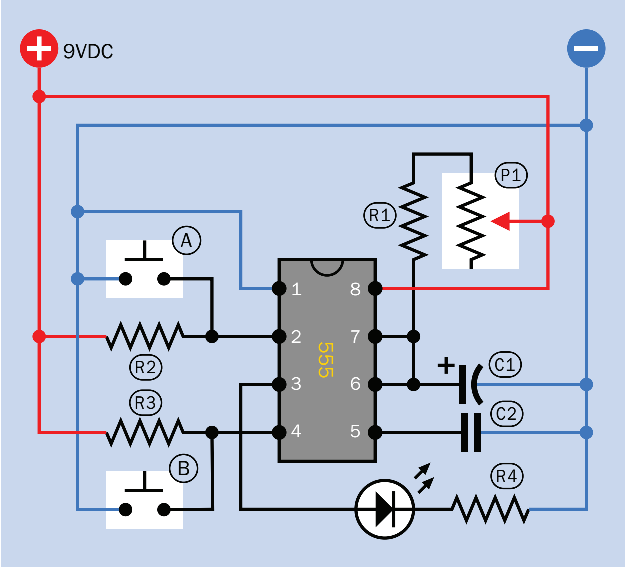


Figure 15-12. Schematic version of Figure 15-11.

Here's another thing to try. Set P1 to give a long pulse, tap Button A, and then hold down Button B. This cancels the output. If you continue to hold down Button B while you press Button A again, nothing happens.

Pin 4 is the Reset Pin. So long as Pin 4 is kept near the supply voltage by its 10K pullup resistor, the timer will respond to being triggered. When you ground Pin 4, you force the timer to interrupt whatever it is doing, and nothing more will happen until you release pin 4 from its connection with negative ground. Pin 4 should be kept in a high state to enable the timer to function.

- While the timer is running in monostable mode, Pin 4 should always be kept in a ***high*** state (near the voltage of the power supply). If it is not connected to anything, it is ***floating***, and results can be unpredictable.

Figure 15-13 shows how the timer behaves, in graphical format. The 555 converts the imperfect world around it into a precise and dependable output, and it switches on and off so quickly, the response appears to be instant.

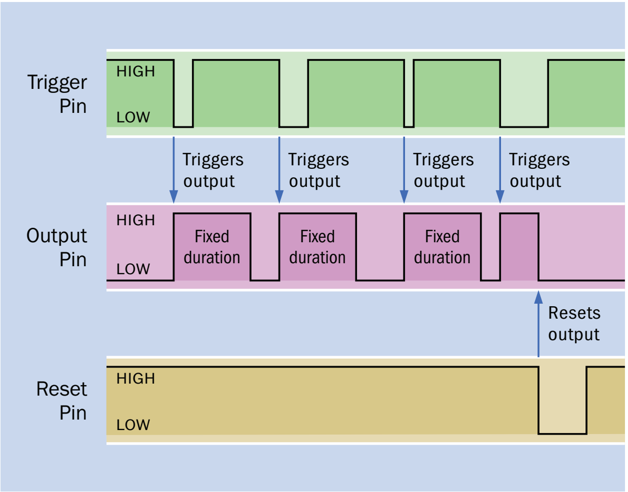


Figure 15-13. Effects of Trigger Pin and Reset Pin on the output.

In my summary of features, I used the terms "near the power supply" and "near negative ground"—but how near is "near"?

- The 555 timer thinks that on the Trigger Pin, any voltage above two-thirds of the power supply is a "high" input while any voltage below one-third of the power supply is a "low" input.
- On the Reset Pin, you may have to apply a voltage below one-third of the supply to get a result. Timers from different manufacturers are inconsistent about this.

One last test. Let go of button B, hold down button A, and continue to hold it down beyond the end of the "on" cycle of the timer. This prolongs the pulse from the timer. It stays high until you let go of button A. The datasheet refers to this as the timer ***retriggering*** itself.

- Maintaining a low voltage on the trigger pin of the timer will prolong the output pulse indefinitely.

Now, how does it work? You can use your meter to investigate the timer's behavior. In Figure 15-12, set P1 to create a long pulse, and measure the voltage on the positive side of C1, the 10µF capacitor, relative to negative ground. You should see the voltage climbing up—until it reaches about six volts, when the charge drops suddenly. Where does it go? Well, Pin 7 on the 555 is called the Discharge Pin, which discharges the capacitor into the chip. I'll explain more about the internal working of the chip very shortly.

You can also use your meter to measure the output from the timer from Pin 3, when the chip has been triggered. The voltage rises almost to 8 volts—but no higher. The timer is full of transistors, and they steal a percentage of the voltage in the same way as the NPN transistors which you dealt with previously.

Finally, you may be wondering about the purpose of C2, the 0.01µF capacitor attached to Pin 5. This is the Control Pin, which means that if you apply an intermediate voltage to it, you can adjust the length of the output pulse from the timer. Because we are not using this function yet, it's good practice to put a 0.01µF capacitor on Pin 5 to protect it from voltage fluctuations that could interfere with normal functioning. You will often see circuits in which C2 is missing, but I always include it, just in case.

Beware of Pin-Shuffling!

In all of the schematics in this book, chips are shown as you would see them on a breadboard, with their notches pointing upward and their pins in numerical sequence.

Schematics on web sites or in books do things differently. The pins may be reshuffled and resequenced to put the power source at the top of the circuit, with conventional current flowing generally downward. To give you an example, the circuit in Figure 15-14 is identical in function to the one in Figure 15-12. In some ways it may give you a quicker understanding of what the circuit is about, but if you came across this schematic and wanted to build the circuit on a breadboard, you might have to do an intermediate sketch to figure out how to organize the components. Because *Make: Electronics* is a hands-on book, my schematics are breadboard-oriented.

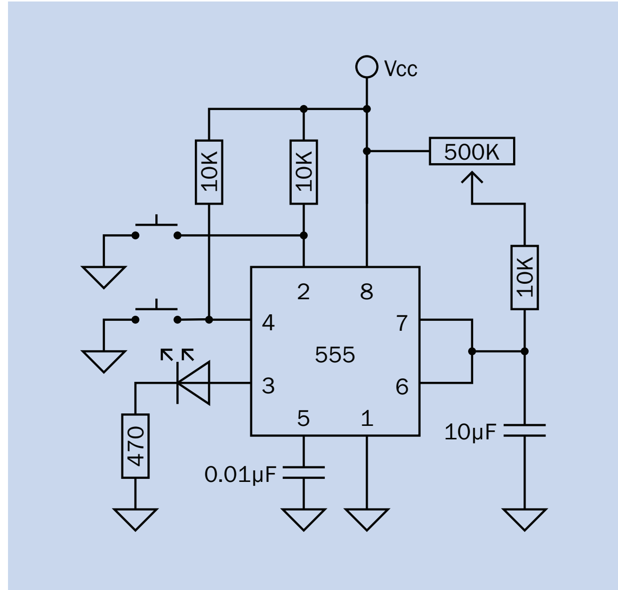


Figure 15-14. This circuit is identical in function to the test circuit shown in Figure 15-12. Note that Vcc indicates supply voltage, and the triangles pointing downward indicate negative ground.

Pulse Duration

When you made your own RC network in Experiment 9, you had to do some annoying calculations to figure out how long a capacitor would take to reach any particular voltage. With a 555 timer, everything is much easier: You can look up the duration of its output pulse in a table such as the one in Figure 15-15.

In the circuit that you built, the total value of the trimmer and the resistor in series with it controlled the charge time of the capacitor, so this total resistance is shown at the top of the table. The value of C1 is shown down the left side, and the numbers in the table tell you the approximate length of the pulse in seconds.

The 555 timer is an amazing piece of chip design. You can vary the timing resistor, the timing capacitor, and the power supply over an extremely wide range, and the chip still gives accurate, consistent results. Just follow a few guidelines:

	10K	22K	47K	100K	220K	470K	1M
1000µF	11	24	52	110	240	520	1100
470µF	5.2	11	24	52	110	240	520
220µF	2.4	5.2	11	24	52	110	240
100µF	1.1	2.4	5.2	11	24	52	110
47µF	0.52	1.1	2.4	5.2	11	24	52
22µF	0.24	0.52	1.1	2.4	5.2	11	24
10µF	0.11	0.24	0.52	1.1	2.4	5.2	11
4.7µF	0.052	0.11	0.24	0.52	1.1	2.4	5.2
2.2µF	0.024	0.052	0.11	0.24	0.52	1.1	2.4
1.0µF	0.011	0.024	0.052	0.11	0.24	0.52	1.1
0.47µF		0.011	0.024	0.052	0.11	0.24	0.52
0.22µF			0.011	0.024	0.052	0.11	0.24
0.1µF				0.011	0.024	0.052	0.11
47nF					0.011	0.024	0.052
22nF						0.011	0.024
10nF							0.011

Figure 15-15. Pulse duration, in seconds, of a 555 timer in monostable mode, for values of the timing resistor and timing capacitor. Times are rounded to two digits.

- Don't use a low resistor value between the positive power supply and the Discharge Pin. They will cause excessive power consumption. The absolute minimum is 1K, but I suggest 10K.
- If you use a timing capacitor larger than 1,000µF, its internal leakage becomes comparable with its charging rate, causing inaccurate results.
- The 555 timer requires a power supply between 5VDC and 16VDC. It is unreliable outside of these limits.

What if you want a duration that isn't listed in the table? There is a simple formula, in which T is the pulse time in ***seconds***, R is the timing resistance in ***kilohms*** and C is the timing capacitance is ***microfarads***:

```
T = R * C * 0.0011
```

Suppose you know the time that you want. You have a capacitor, and you want to know how big a resistor you should use. Rearrange the formula like this:

```
R = T / ( C * 0.0011 )
```

If you have a 47µF capacitor, C = 47. If you want a 3-second pulse, T = 3. So, you put those numbers into the formula:

```
R = 3 / ( 47 * 0.0011 )
```

First multiply 47 * 0.0011 and you get 0.0517. Divide that into 3, and you get approximately 58—so, you need a 58K resistor. That is not a standard value, but 56K resistors are common. Or, you could do it the easy way: Use a 100K trimmer, set it to about the half-way mark, and adjust it till you get precisely a 3-second pulse by trial and error.

Just be careful that you never reduce the total timing resistance to zero. That's why I put a 10K resistor in series with trimmer P1 in figures 15-11 and 15-12. Even when the trimmer is turned down to the bottom of its scale, the 10K resistor is still there. If it was not, the capacitor would be connected directly to the positive side of the power supply, so it would charge instantly and then never discharge. Worse still, the Discharge Pin would be connected directly to the power supply, which would damage the chip.

- In a 555 timer circuit, always include some resistance between Pin 7 and the positive power supply. I suggest a minimum of 10K.

Inside the 555 in Monostable Mode

I think that if you can imagine the inner workings of the 555 timer, using it may become easier. The plastic body of the chip contains too many transistors for me to show them all, but I can summarize the way they work, as shown in Figure 15-16 on the next page.

At the heart of the timer is a ***flip-flop***, which is a transistor circuit that functions like a double-throw switch. In fact, I have illustrated it that way in the diagram, to make it easier to imagine. The flip-flop is controlled by two ***comparators***, shown as A and B. Each comparator

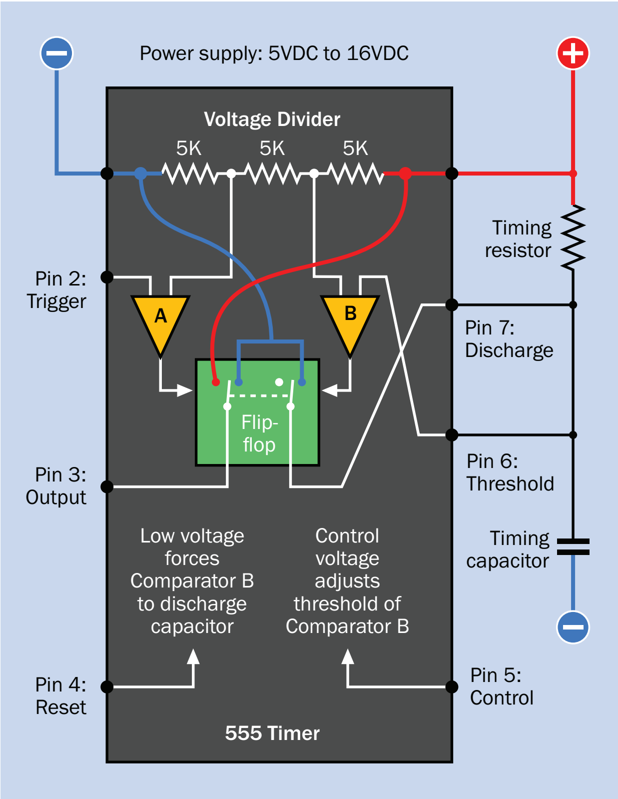


Figure 15-16. Inside a 555 timer.

compares two voltages, and changes its output depending on which voltage is higher.

Comparator A compares the voltage on Pin 2 (the Trigger Pin) with a fixed voltage of one-third of the power supply. When Pin 2 goes ***below*** that level to trigger the chip, Comparator A pulls the flip-flop in its direction, which switches positive current to Pin 3 (the Output Pin).

Comparator B compares the voltage on Pin 6 (the Threshold Pin) with a fixed voltage of two-thirds of the power supply. When Pin 6 rises ***above*** that level, Comparator B pulls the flip-flop in its direction, which grounds Pin 3 and also grounds Pin 7 (the Discharge Pin). Because Pin 7 is wired to the timing capacitor, the capacitor discharges itself into the chip. This is what happens at the end of a pulse from the timer.

Initially, let's suppose the timer has just been triggered by a low voltage on Pin 2. Comparator A senses this, and pulls the flip-flop to the left, which starts a positive pulse and un-grounds the external timing capacitor. The capacitor starts to charge, till it exceeds two-thirds of the supply voltage. Comparator B senses this and pulls the flip-flop right. This ends the pulse and grounds the capacitor.

However, at any time during a pulse, a low voltage on Pin 4 can override the status of the timer and force Comparator B to pull the switch right and discharge the capacitor, while shutting down the output from the chip.

And, a varying voltage on Pin 5 changes the reference voltage for Comparator B, to increase or decrease the cutoff voltage for the capacitor. This is of very little interest while the timer is in monostable mode, but it will have a lot of possibilities when the timer is rewired to become astable.

Startup Pulse Suppression

When a timer is configured in monostable mode, powering it up may cause it to emit a single spontaneous pulse, even though it has not been triggered yet. Sometimes this can be quite annoying, but there is a simple answer: Add a 1μF capacitor between the Reset Pin and negative ground. The capacitor sinks current from the pin when the power is first turned on, and holds the pin low for a fraction of a second—just long enough to stop the timer from emitting its waking-up pulse. After the 1μF capacitor is charged, it doesn't do anything more, and a 10K pullup resistor can hold the reset pin normally-positive, so the capacitor won't interfere with the running of the timer. Figure 15-17 shows where you might add the 1μF capacitor at the bottom-left corner of the monostable test circuit.

I'll be using the pulse-suppression concept in subsequent experiments.

Why the 555 is Useful

What can you do with the pulse of a fixed but programmable length from a 555 timer? Think in terms of the timer controlling other components, such as an outdoor light, where you only want it to remain on for a short time after it is triggered by a motion sensor. A timer could also control a toaster, or flashing Christmas-tree lights. The old original Apple][computer used a 555 to flash its cursor. In the next few projects, you'll see me applying a timer in many unexpected ways.

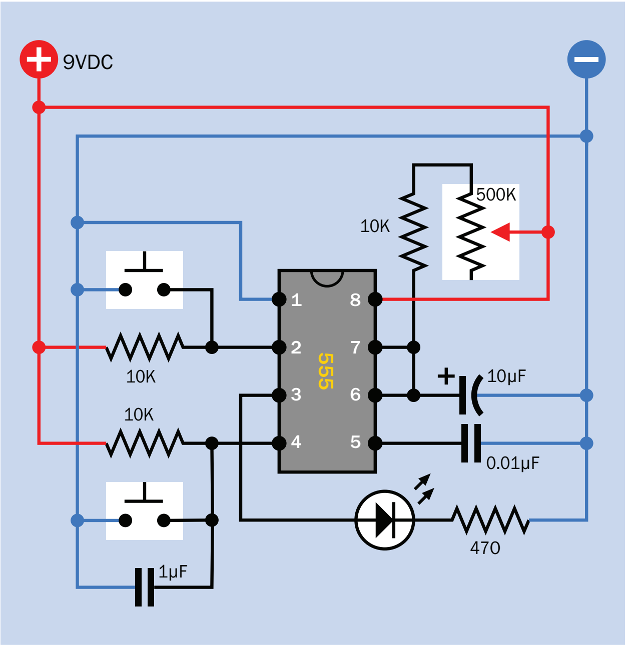


Figure 15-17. A 1µF capacitor to inhibit the startup pulse from the timer has been added at bottom-left.

Microcontrollers are now used in many applications where a 555 timer was used in the past—such as the adjustable delay for intermittent windshield wipers in a car, or in a microwave oven. But a microcontroller has to be programmed, and when you're developing a circuit, a 555 timer can be quicker and simpler. It also delivers ten times the power of a microcontroller.

When we get to the intrusion alarm in Experiment 17, you'll want a 30-second delay before the alarm arms itself, so that you have time to leave and close the door. Then you'll want another 30-second delay when you come back in, to give you time to switch the alarm off. I will use 555 chips for each of these events.

The experiment that you just performed seemed trivial, but it creates a lot of possibilities.

Bistable Mode

There's another way to use the timer, known as *bistable mode*, which is shown in Figure 15-18. This is like Figure 15-11, but with all the timing components removed, and Pin 6 connected with negative ground. Why would you want to do this?

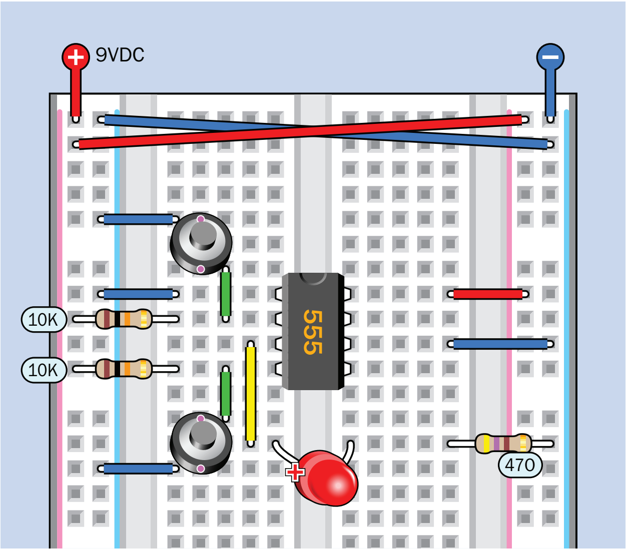


Figure 15-18. The timer is now wired in bistable mode.

Tap Button A briefly, and the LED lights up. For how long? For as long as power is supplied to the circuit. The output from the timer continues indefinitely.

Now tap Button B, and the LED goes off. For how long? Until you press Button A again.

Normally, when you trigger the timer, its output pulse ends when a timing capacitor wired to pin 6 accumulates two-thirds of supply voltage. But the capacitor has been eliminated, and pin 6 is grounded, so it can never reach two-thirds of supply voltage. Consequently, when you trigger the timer, the output pulse will never end.

You can stop the output by pressing Button B to apply a low voltage to the reset pin, but once the output stops, there is no reason for it to start again. Pin 2 has a pullup resistor on it, so it won't be triggered until you press Button A again.

This configuration is called bistable because it sticks in either state, when the output is high and when it is low. This is similar to the behavior of a flip-flop, and can also be called a *latch*.

Why would you need a 555 timer to act this way? Because you may want an application that is started by tapping one button and stopped by tapping another button. My desk lamp works this way, and so does the band saw in my workshop. When you get to the reflex tester circuit in Experiment 18, you'll find it is started and reset by a timer wired in bistable mode.

It's okay to leave pins 5 and 7 unconnected in this circuit, because I'm pushing the timer into extreme states where any random signals from those pins will be ignored. This is one situation where floating pins are okay.

Reverse Mode

I'll mention just one more way of wiring a 555 timer. In *reverse mode*, the output from the timer is reversed, meaning that it goes low whenever the input is high, or high whenever the input goes low. This is similar to the bistable circuit, except that it doesn't latch itself in one state or the other.

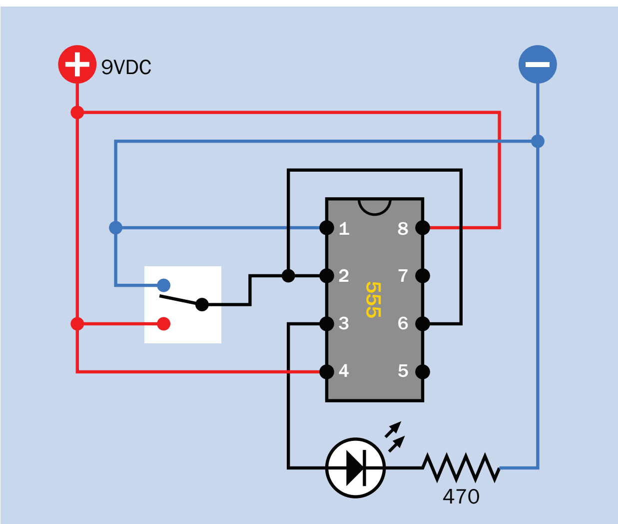


Figure 15-19. A 555 timer wired in reverse mode.

You can buy a chip in the 74xx family known as an *inverter* which does this, but it is a logic chip with a limited output. The 200mA output of a 555 timer is convenient when you need more current, perhaps to run a relay or a very small motor.

(There is also a power-supply product known as an inverter, which converts DC to AC, but it is very different from the logic chip known as an inverter.)

Why would you want to reverse an input? Sometimes it's useful. You find yourself designing a circuit where a negative input has to create a positive output, and a positive input has to create a negative output. Figure 15-19 shows how to make the timer do this.

Notice that pins 2 and 6 are wired together, so that the input to Pin 2 is also applied to Pin 6. Remember that Pin 2 is triggered by a low voltage, and it ignores a high state, while Pin 6 ignores a low state but is triggered by a high state (because it is designed to respond to a charged capacitor). Consequently, when the input is high, Pin 6 reacts by shutting down the output. When the input is low, Pin 2 reacts by starting an output pulse that will continue indefinitely. I showed a double-throw switch providing the input to remind you that Pin 2 must have either a negative or a positive voltage applied; it must not be allowed to float.

I can see a very convenient way to implement this, using a chip known as a 556. This contains two 555 timers in one package. If you are using the first timer in its normal way, but you want to reverse its output, you just feed it through the second timer, which is wired as I have described. I leave it to you to look up the 556 chip if you are interested.

In addition, a timer in reverse mode can provide the useful service of converting a smoothly varying signal into a definite high or a definite low output, with some *hysteresis* in between. I'll explain this here, as I don't expect to refer to it elsewhere in the book.

Hysteresis

Suppose you have a thermostat which you set to switch on a heating system when the temperature drops to 65 degrees Fahrenheit. Do you want the thermostat to switch the heat off again when the temperature climbs to 65.01 degrees? Probably not; the system may not run very efficiently if it switches itself off almost as soon as it has switched itself on. A thermostat will usually wait till the temperature is 67 or 68 degrees before switching off the heat.

Now the temperature is 68, and the heating system is off. Do you want it to switch on again when the temperature falls to 67.99? No, no, you want it to wait till it falls back to 65.

You can think of the thermostat as a "sticky switch." It sticks on for a while as the temperature rises, then sticks off when the temperature falls. This behavior is known as hysteresis, and it's very useful whenever you're receiving a smoothly varying (analog) input that has to be converted to an "on or off" output.

A 555 timer wired in reverse mode functions like this. If you replace the switch with a smoothly varying voltage that falls to one-third of the supply, the timer output goes high. Then if the voltage starts to rise again, the timer output stays high until the voltage reaches two-thirds of the supply. That switches the output off, and it stays off until the voltage on Pin 2 drops back down to one-third.

Some chips are specifically designed with hysteresis built in. Often they use an input known as a *Schmitt trigger*. I'm getting off-topic here, so I'll leave you to look that up.

The History of the Timer

The development of the 555 timer is an amazing story. Back in 1970, when barely a half-dozen corporate seedlings had taken root in the fertile ground of Silicon Valley, a company named Signetics bought an idea from an engineer named Hans Camenzind (pictured in Figure 15-20). It wasn't a huge breakthrough concept—just 23 transistors and a bunch of resistors that would emit a single pulse or a series of pulses. The circuit would be versatile, stable, and simple, but these virtues were not the primary selling point. Using the emerging technology of integrated circuits, Signetics could reproduce the whole thing on a silicon chip. What a concept!

Hans worked alone as an independent consultant, building the whole thing initially with off-the-shelf transistors, resistors, and diodes on a very large breadboard. It worked, so then he started substituting slightly different values for the various components to see whether the circuit would tolerate variations during the manufacturing process. He made at least 10 different versions of the circuit.

Next came the crafts work. Hans sat at a drafting table and used a specially mounted X-Acto knife to scribe his circuit into a large sheet of plastic. He couldn't use a desktop computer to do the design, because there were no desktop computers in 1970.

Signetics then reduced the image photographically by a ratio of about 300:1. They etched it into tiny wafers, and embedded each of them in a rectangle of black plastic with the part number printed on top. The head of the sales department called it a 555 because he had a feeling that this chip could be a big success, and he wanted the part number to be memorable.

It was released in 1971 and turned out to be the most successful chip in history, both in the number of units sold (tens of billions and counting) and the longevity of its design (not significantly changed in 50 years).

Today, chips are designed by large teams and tested by simulating their behavior using computer software. Thus, the chips inside a computer enable the design of new chips. The heyday of solo designers such as Hans Camenzind is long gone, but his genius lives inside every 555 timer that emerges from a fabrication facility.

In 2010, when I was first writing *Make: Electronics*, I looked up Hans online and found that he maintained his own web site, which included a phone number. On impulse, I called him, and he answered right away. This was a strange moment, to be talking to the man whose chip design I had used for decades. He was friendly (although he didn't waste words), and kindly agreed to review the text of my book. Even more kindly, after he read it, he gave it a strong endorsement.

Figure 15-20. Hans Camenzind, who developed the 555 timer entirely on his own, and sold the circuit to Signetics.

Subsequently I read a short history of electronics that he wrote himself, titled *Much Ado About Almost Nothing*. It is still available online, and I recommend it. I felt honored to have had the opportunity to talk to one of the pioneers in integrated circuit design, and was sad when I heard of his death in 2012. This edition of *Make: Electronics* is dedicated to his memory.

Not All Timers Are Equal

Everything I have said so far refers to the old original ***TTL*** version of the 555 timer. TTL is an acronym for ***transistor-transistor logic***, which preceded modern CMOS chips that use much less power ("CMOS" being an acronym for ***complementary metal oxide semiconductor***). The old TTL version of the 555 is also referred to as the ***bipolar version***, as it contains bipolar transistors.

The TTL 555 has three advantages: It is cheap, robust, and can deliver up to 200mA. However, it is not efficient, and also has a well-known bad habit of generating a voltage spike when it switches on its output. The spike is extremely brief but rises more than 50% higher than the supply voltage, as shown in Figure 15-21.

The newer version of the 555 timer, using CMOS transistors, uses less power and doesn't create a voltage spike. Unfortunately it costs a bit more, is more easily damaged by static electricity, and delivers less current. How much less? That depends on the particular manufacturer. There is a lack of standardization among CMOS versions of the 555, some manufacturers claiming to deliver 100mA while others specifying 10mA.

The CMOS versions don't have standardized part numbers. One version is known as the 7555, which distinguishes it from the old original 555, but other versions keep the 555 number and just precede it with a different group of letters, whose meaning you may not realize. For example, the LMC555CN is actually a CMOS timer.

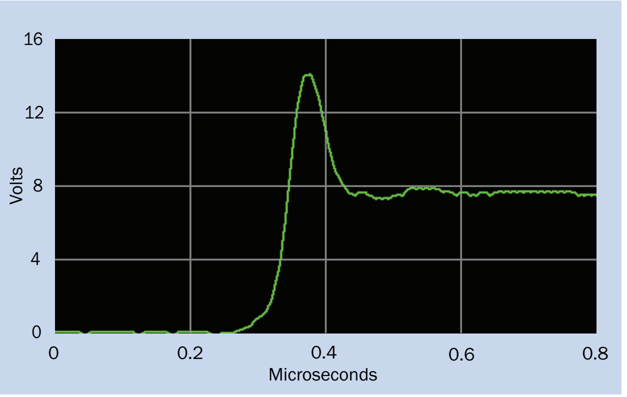


Figure 15-21. When the 555 timer switches on its output, it creates the voltage spike shown here.

Generally speaking, any CMOS version on a supplier's web page usually costs twice as much as a TTL version, and can accept a minimum supply voltage below 5VDC. Still, you can't really rely on these indicators. Really, when you buy a product calling itself a "555 timer," you should check its datasheet to see if it is a CMOS chip and what its specifications are. If you make a mistake and use a CMOS version instead of a TTL version, it may allow you only one-tenth as much output current.

In this book, to avoid confusion and keep things simple, I am only using the old TTL version of the 555 timer, also known as the bipolar version. In circuits containing sensitive digital logic chips, I will add a smoothing capacitor to suppress the voltage spike from the timer.

Experiment 16
Set Your Tone

Now that you're familiar with the 555 timer in monostable mode and bistable mode, I want you to get acquainted with it in ***astable mode***—in which it will do what your astable multivibrator did in Experiment 11, except that it will be much simpler, more versatile, and more compact.

You Will Need:

- Breadboard, hookup wire, wire cutters, wire strippers, multimeter.
- 9VDC power supply (battery or AC adapter).
- 555 timer chips (2).
- Small speaker (1).
- Resistors: 100 ohms (1), 1K (3), 10K (3), 470K (1)
- Capacitors: 0.01μF (2), 0.47μF (1), 10μF (1), 100μF (1)
- 1N4148 diode (1).
- Trimmers, 10K (2), 500K (1).

Astable Test

In Figure 16-1, the timer is wired in astable mode, to generate a stream of pulses. Compare this with the schematic in Figure 15-12, where the timer was wired to generate a single pulse when it was triggered. What differences do you notice?

First, the new schematic has no buttons to create inputs. This is because the timer doesn't need inputs, now; it runs itself.

Second, it uses two timing resistors labeled R1 and R2. You should know that all reference guides for the 555 timer, including manufacturers' datasheets, refer to

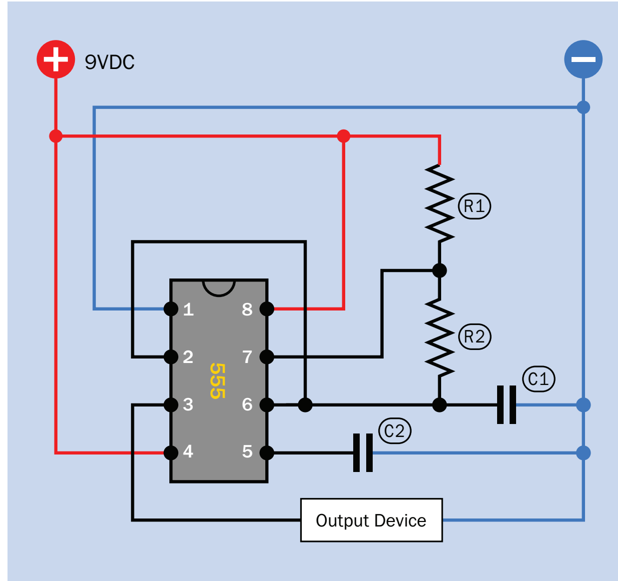


Figure 16-1. General-purpose schematic showing a 555 timer wired in astable mode.

these resistors as R1 and R2, and those same labels are used in a well-known formula for calculating the frequency of the timer.

Notice that the wire which connected Pin 6 and Pin 7 in the monostable circuit isn't there now. However, a wire has been added linking Pin 6 to Pin 2. When you build the circuit, you can run a jumper around the timer, or over the top of it, as you wish. But what do you think this new connection is for, linking the Threshold Pin with the Trigger Pin? If you're guessing it's something to do with the timer being able to trigger itself, you're right.

Notice that C2 is still there, to isolate Pin 5 from unwanted stray interference.

The "Output Device" that I have shown attached to Pin 3 can be an LED, if the timer is running slowly, or a speaker, if the timer is running at an audible frequency. The timer doesn't need any help from an external transistor to drive a speaker.

Now I suggest that instead of building just the bare, basic circuit, you should add a trimmer to R2, so that you can have the fun of making an audio output squeal or groan.

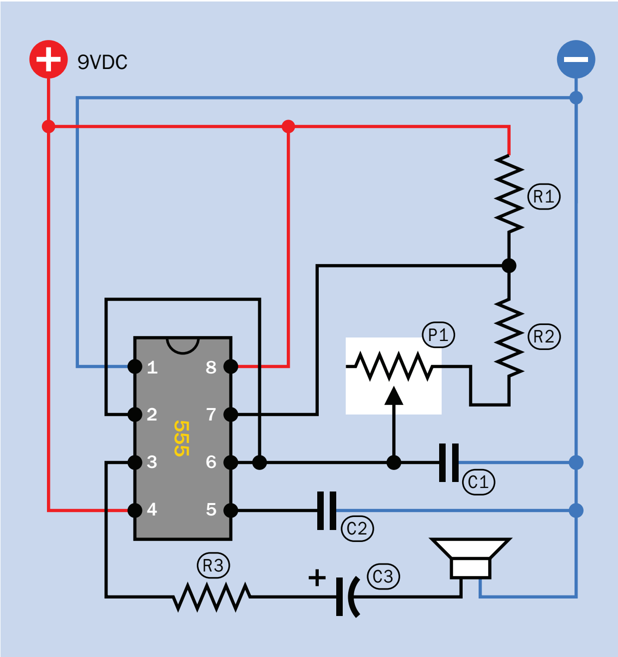


Figure 16-2. Test circuit for a 555 in astable multivibrator mode.

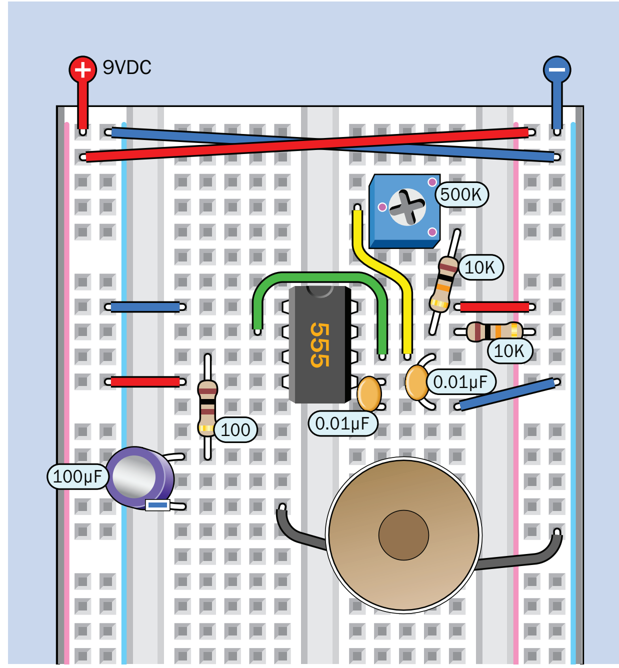


Figure 16-3. Breadboarded version of the astable test circuit.

The schematic is shown in Figure 16-2, and the values of the components appear in the breadboard diagram in Figure 16-3. Notice that I had to move the trimmer up on the breadboard, to find space for it. But if you trace the connections and compare them with the schematic, they're the same.

- In this astable circuit, as in the monostable circuit which you built previously, the resistor between Pin 7 and the positive supply voltage should be 10K or higher. It must never be zero, as this could burn out the chip.

I have included C3, a 100µF electrolytic capacitor, in series with the speaker. This is a *coupling capacitor*, which you may remember that I mentioned in Experiment 9. Pulses from the timer will seem to zoom straight through as displacement current, but the capacitor isolates the speaker from the DC component of the 555 output. This is not essential, but it does cut the power consumption of the circuit by 50 percent.

A 100-ohm resistor is included to prevent the 8-ohm speaker from overloading the timer.

Put it together, and you'll find that the trimmer can adjust the audio output frequency from low to very high—almost at the high end of human hearing. Now, how does it work?

The Charge-Discharge Cycle

Figure 16-4 shows a closeup view of the relevant pins on the timer. Please refer back to Figure 15-16 if you need to refresh your memory about the internal workings of the timer.

When you first apply power, Comparator A inside the timer moves the flip-flop inside the timer to its left position, which starts a positive pulse from Pin 3, the Output Pin, in exactly the same way as when the timer was running in monostable mode. Likewise, Comparator A isolates Pin 7, the discharge pin. So, this pin cannot discharge the capacitor yet, and the capacitor starts to accumulate voltage as a result of current flowing into it through R1 + R2. Their combined value determines the length of the pulse from the timer, which is known as the *charge cycle*.

Comparator B, inside the chip, monitors the voltage on the capacitor through the Threshold Pin, just as it

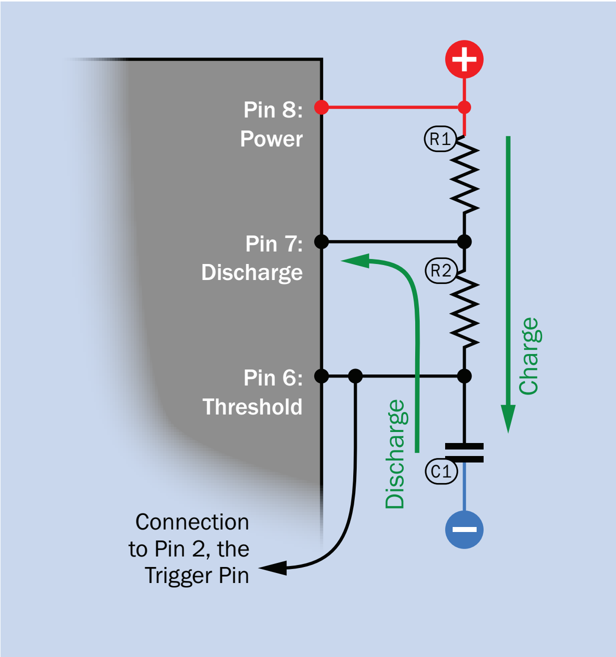


Figure 16-4. Closeup of three pins on the 555 timer.

	10K	22K	47K	100K	220K	470K	1M
47µF	1	0.57	0.3	0.15	0.068	0.032	0.015
22µF	2.2	1.2	0.63	0.31	0.15	0.069	0.033
10µF	4.8	2.7	1.4	0.69	0.32	0.15	0.072
4.7µF	10	5.7	3.0	1.5	0.68	0.32	0.15
2.2µF	22	12	6.3	3.1	1.5	0.69	0.33
1.0µF	48	27	14	6.9	3.2	1.5	0.72
0.47µF	100	57	30	15	6.8	3.2	1.5
0.22µF	220	120	63	31	15	6.9	3.3
0.1µF	480	270	140	69	32	15	7.2
47nF	1K	570	300	150	68	32	15
22nF	2.2K	1.2K	630	310	150	69	33
10nF	4.8K	2.7K	1.4K	690	320	150	72
4.7nF	10K	5.7K	3K	1.5K	680	320	150
2.2nF	22K	12K	6.3K	3.1K	1.5K	690	330
1nF	48K	27K	14K	6.9K	3.2K	1.5K	720
470pF	100K	57K	30K	15K	6.8K	3.2K	1.5K
220pF	220K	120K	63K	31K	15K	6.9K	3.3K
100pF	480K	270K	140K	69K	32K	15K	7.2K

Figure 16-5. Table of timer frequencies for values of C1 and R2, assuming R1 remains constant at 10K.

did when the timer was in monostable mode. When C1 reaches two-thirds of supply voltage, Comparator B pulls the flip-flop, and this turns off the output pulse from Pin 3. The flip-flop also grounds Pin 7, so the capacitor begins losing its charge through R2, into that pin. The value of R2 determines the length of the ***discharge cycle***.

Now, how does the chip tell itself to start a new pulse? Because Pin 2, the Trigger Pin, is connected through an external wire with Pin 6, Pin 2 knows when the voltage on C1 drops to one-third of the supply. At that point, Pin 2 does what it always does: It moves the flip-flop inside the timer to its left position, which starts a positive pulse from Pin 3, the Output Pin. The flip-flop also isolates Pin 7, the discharge pin. So, now C1 can begin charging again, and the whole process repeats.

The timer has become self-triggering.

People are often puzzled by the need for two resistors. Why does the capacitor have to charge through R1 + R2 and then discharge through R2 only?

Well, R2 has to be there, because otherwise, when the capacitor discharges, it would dump all of its charge immediately into Pin 7 when that pin is grounded, and the discharge cycle would end as quickly as it began. Consequently, the gaps between pulses would diminish almost to zero.

All right then, why do we need R1? Because when Pin 7 is grounded, R1 prevents it from being connected directly with the positive supply voltage, which would be a short circuit.

Other timers have been developed using different configurations of components, but this is the way the 555 timer is organized, and changing it isn't an option at this point. The concepts should become clearer when you start actually using it.

Figuring the Frequency

The table in Figure 16-5 will guide you if you want to choose components to make the 555 timer generate a particular frequency. Refer back to Figure 16-1 for the locations of R1, R2, and C1, and now bear in mind that the table shows values for C1 on the left side, values for R2 along the top, and it assumes that R1 is fixed at 10K.

- Remember, the table assumes that R1 is fixed always at 10K.

The numbers inside the table are frequencies in Hz and kilohertz (represented only as K, to save space). The highest is well above the audible limit.

What if you want to know the length of each cycle, instead of the frequency? First, bear in mind that a complete cycle is the length of a pulse *plus* the gap between it and the next pulse. To discover this value, simply divide any number in the table into 1. (This is known as the *reciprocal* of the number). For instance, using the number 0.015 in the top-right corner of the table:

```
1/0.015=66.7 seconds, approximately.
```

If you use higher values for a capacitor than 47µF, you can time much longer pulses, but as I noted before, leakage of voltage in a large capacitor will start to interfere with timing accuracy.

What if you want a frequency that isn't listed in the table? You can use a formula to calculate values for R1 and R2. Various versions of this formula exist, but in my version below, the frequency is in hertz, while R1 and R2 are in *kilohms* and C1 is in *microfarads*:

```
frequency=1,440/((R1+(2*R2))*C1)
```

The parentheses tell you the sequence in which you do the calculations. First double R2, then add R1, then multiply by C1, then divide the result into 1,440.

What if you don't like the idea of doing this calculation? Not a problem! Search online for

```
555 frequency calculator
```

You'll find several web sites that will do the calculation for you. Just enter the frequency that you want and 10,000 as the value in ohms for R1, and the site will show you possible values for C1 and R2.

Bear in mind that during each cycle, the length of each "on" pulse will always exceed the length of the gap which follows it. This is because the length of the "on" pulse is determined when the capacitor charges through R1 + R2, while the length of the gap is determined when the capacitor discharges through R2 alone. Figure 16-6 illustrates this.

What if you're really hard to please, and you want short pulses and long gaps? I can imagine a need for this—for example, if you want to trigger a relay briefly every minute or so. Well, there is a way to do it. You just add a diode.

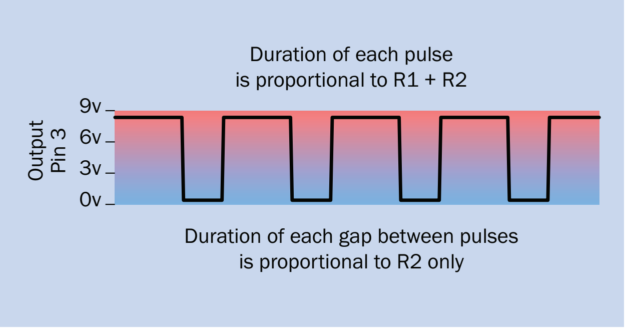


Figure 16-6. Pulses and gaps between them, when a conventionally wired 555 timer is running in astable mode.

Adding a Diode

A diode is a very simple device. It allows current to flow in one direction, but blocks the flow in the other direction. *Signal diodes* are used for forward currents as high as 200mA, while *rectifier diodes* can handle much more current. For many of the applications in this book, a very basic little signal diode known as the 1N4148 is appropriate.

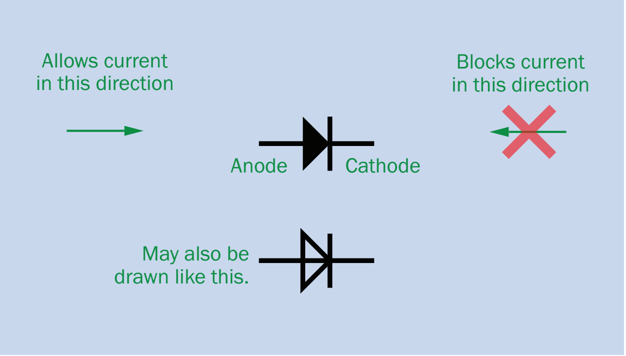


Figure 16-7. Either of these symbols can be used in a schematic to represent a diode.

- The two ends of a diode are known as the *anode* and the *cathode*. Forward (conventional) current flows into the anode and out of the cathode, which is marked with a silver or black band. See Figure 15-7.

Two schematic symbols for a diode are shown in Figure 16-7. The big arrow points in the direction of conventional current, but because a signal diode does not emit light, the little pair of arrows in an LED symbol are omitted. Also, while the LED symbol often has a circle around it, a diode symbol does not. I'm not sure why, but since everyone omits the circle, I will do the same.

Now, how can it be used in an astable 555 circuit to make pulses shorter and gaps between them longer?

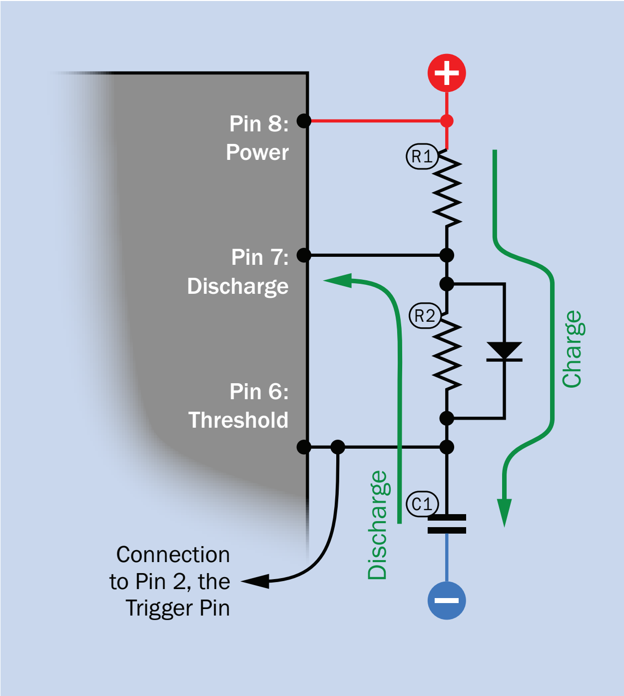


Figure 16-8. Adding a diode to bypass R2.

Figure 16-8 shows the principle of the thing. When the capacitor charges, most of the current bypasses R2 through the diode, so that R1 controls the charging rate, instead of R1 + R2. When the capacitor discharges, the diode blocks the current, so it has to go through R2, which controls the discharging rate.

In Figure 16-9 I redrew the schematic (which you originally saw in Figure 16-1) with the diode added. You can see it near in the middle, bypassing R2. Compare it with the breadboarded version, below, which is similar to Figure 16-3, except that I had to reroute the green jumper and remove the trimmer to make room.

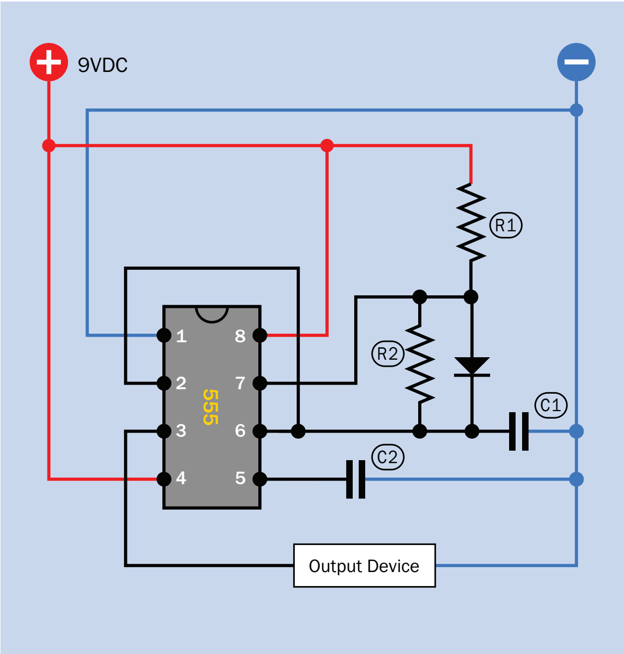


Figure 16-9. In this complete version of Figure 16-8, the diode bypasses the trimmer and the 10K resistor.

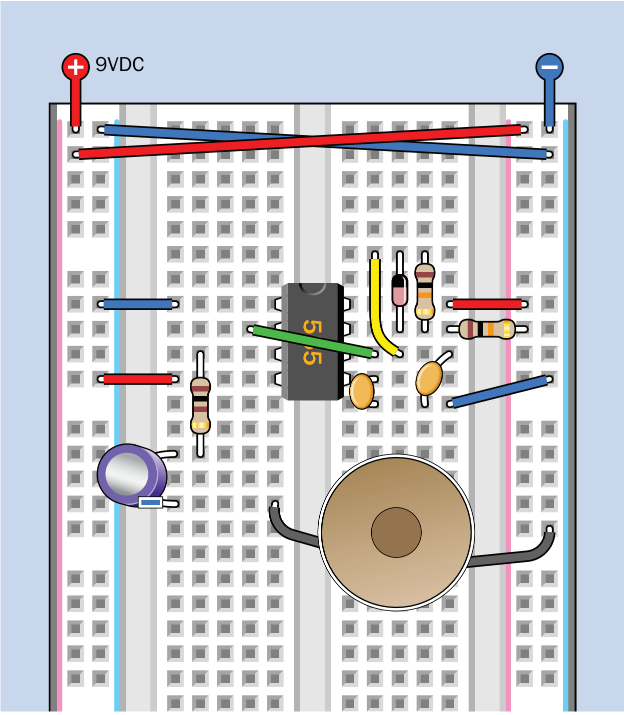


Figure 16-10. Squeezing a diode into the breadboard.

R1 now controls the charging of C1, while R2 controls the discharging of C1. You can set the pulse length with R1 and the gap length with R2—more or less.

Wait—what is this "more or less"? Well, notice that I said "most of the current" skips around R2 through the diode. You already know from your very first investigation of an LED that diodes have an effective resistance. We don't know exactly what it will be, because it varies with voltage, but we know that some resistance will be there. Consequently, although most current will flow through the diode in Figure 16-8, some current will still flow through R2—although not very much. The exact amount will depend on the value of R2 and the power-supply voltage.

One of the wonderful features of a 555 timer is that if you use it as Hans Camenzind intended, its performance is almost the same when you run it from a 16VDC supply as when you use it with 5VDC. But if you add the bypass diode, the frequency formula won't work anymore, and you can't look up a number in the table in Figure 16-5. You will have to proceed with trial and error, to find the result that you want.

The Control Pin

I have been ignoring the Control Pin, because it is not used very often. Under some circumstances, though, it can be very interesting. The principle of the thing is that when you apply a voltage to Pin 5 of a timer that is running in astable mode, the voltage will tune the frequency up and down. This opens up the possibility that if you have two timers, and you connect the output of Timer #2 with the Control Pin of Timer #1, Timer #2 will modulate the frequency of Timer #1.

This may remind you of when I linked two astable multivibrators in Experiment 11. The difference is that the 555 allows you a much wider ranger of adjustments, with results that are more precisely controllable.

First you should learn how the Control Pin works with just one timer. In Figure 16-11, at the bottom of the schematic, resistors R4 and R5 are 1K each, while P2 is 10K. The wiper of the trimmer is then connected with the Control Pin of the timer, while C2, which used to be connected with Pin 5, has been removed. Check back with Figure 16-2 to see the difference.

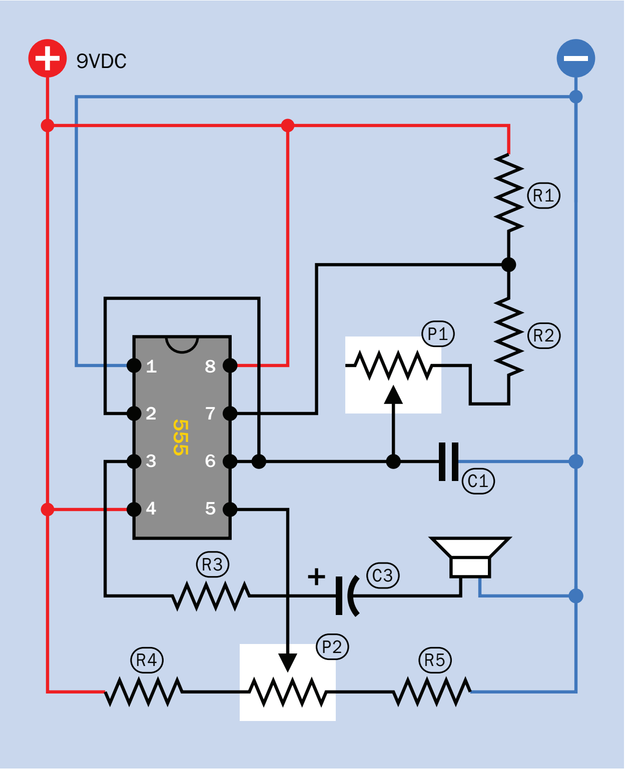


Figure 16-11. Two resistors and a trimmer, added at the bottom of the schematic, will allow you to apply a variable voltage to the Control Pin of the 555 timer.

Figure 16-12 shows the breadboarded version. I'd like you to build this, because I'm going to extend it to complete the experiment with some basic synth capabilities which I think you'll find interesting. If you already built the version in Figure 16-3, you can extend it with the new components at the bottom, while eliminating C2, the 0.01µF capacitor that used to be attached to Pin 5 of the timer.

Here's a brief note about the Control Pin:

- The control voltage on Pin 2 must range between about 20% and 90% of the power supply. Outside of that range, the timer will stop triggering itself and will go silent in protest (although it will not be damaged).

But how did I know that the components which I added would deliver the voltage range that I wanted? I did a simple calculation. This was based on my general knowledge of ***voltage dividers***, which are fundamental in electronics. I will digress into that topic here.

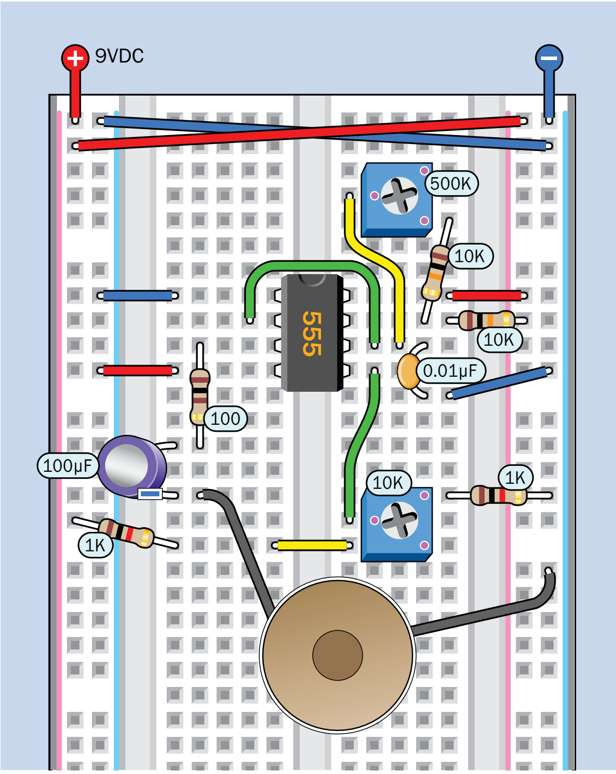


Figure 16-12. Breadboarded version of Figure 16-11.

Voltage Dividers

If you turn back to Figure 15-16, which revealed the inner workings of the timer, you can see the words "Voltage Divider" right there at the top of the diagram, above three 5K resistances in series. I didn't stop to explain this then, because I didn't want to interrupt my other explanations, but if you have three equal resistances in series between a positive supply and negative ground, Ohm's Law tells you that each of them will create an equal voltage drop, as shown in Figure 16-13. You could use any resistances—not just 5K—and you would get the same result, so long as the resistances are all the same.

Now suppose I put two of the 5K resistors together to make a 10K resistor, as shown in the second section of Figure 16-13. A 10K resistor has to be the same as two 5K resistors in series—right? So you should still get a 3V voltage as shown.

And then if you flip the resistors, as in the third section of Figure 16-13, you should get 6V.

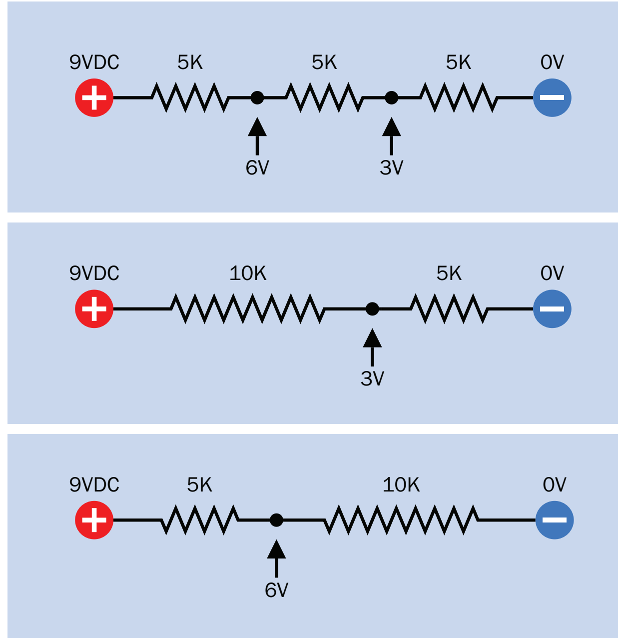


Figure 16-13. Basic concepts of voltage division.

It seems to me that I should be able to calculate the voltage at the point between two resistors by using some kind of a formula—and indeed, this can be done. Figure 16-14 shows how, using any two resistors wired between the positive supply and negative ground. I've named the resistors RA and RB instead of R1 and R2, because I already used R1 and R2 for the resistors in the timing circuit for a 555 timer. Vcc is the supply voltage, and V is the voltage between the resistors.

If I write the formula here in plain text, it looks like this:

```
V = Vcc * ( RB / ( RA + RB ) )
```

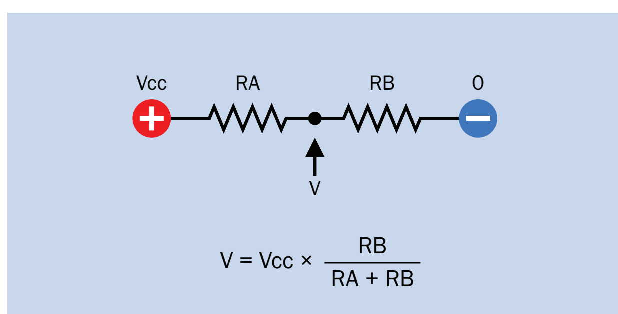


Figure 16-14. The formula for the voltage in a voltage divider.

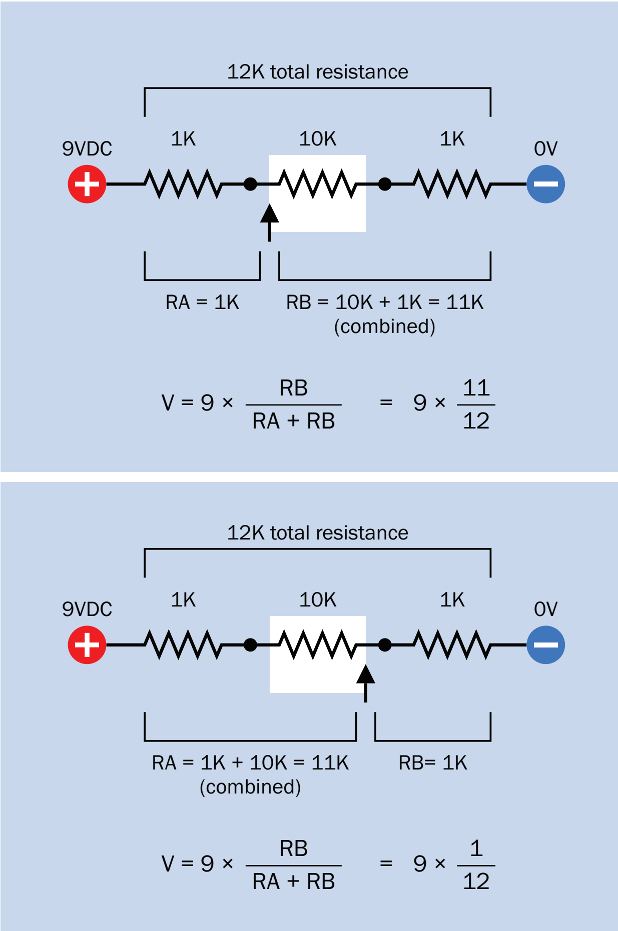


Figure 16-15. Applying the formula for a voltage divider that is connected with the Control Pin of a 555 timer.

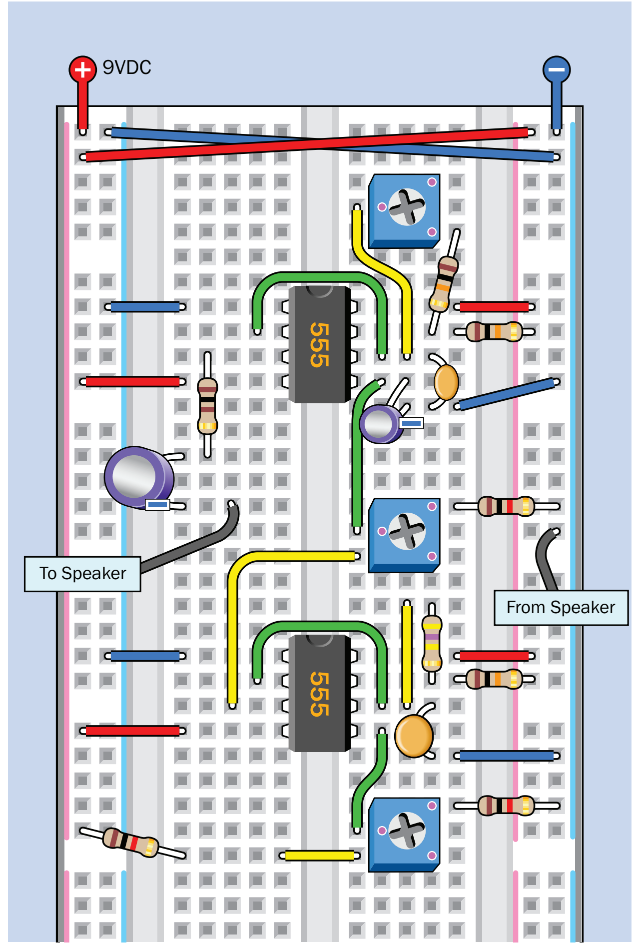


Figure 16-16: Final breadboard of the dual-555 circuit.

Now I'll show how this can be applied to the 555 circuit. Figure 16-15 shows the two 1K resistors and the 10K trimmer that I put together to provide the control voltage, but in the upper half of the figure, I have adjusted the trimmer so that its wiper is all the way at the left end of its scale. The calculation in the figure shows that the voltage at the wiper is now

```
9*(11/12)
```

which my pocket calculator tells me is 8.25V. The lower half of the figure shows that when the wiper of the trimmer is at the other end of its scale, the voltage on it is

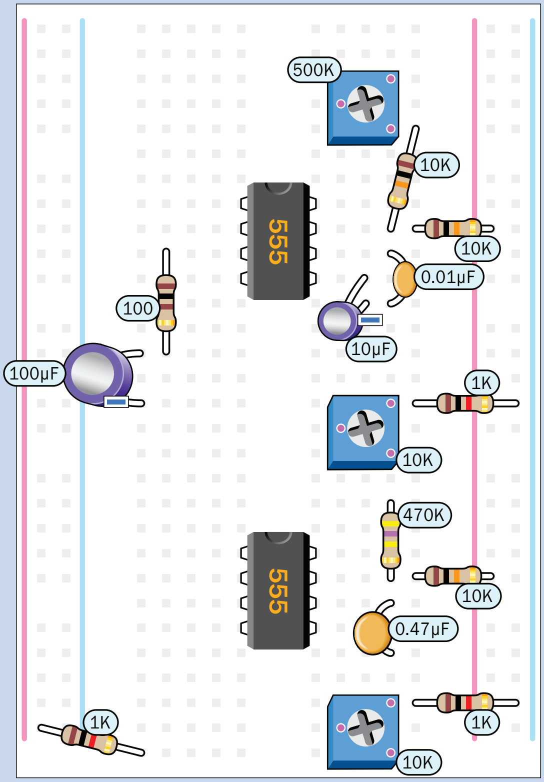
```
9*(1/12)
```

which is about 0.75V.

The low end is a bit too low—but wait, when you actually connect the voltage divider to the Control Pin, the voltage in the Control Pin affects the voltage divider, and brings the voltage up a bit. As usual, in electronics, everything tends to affect everything else.

Here are some general take-home messages about voltage dividers:

- The formula gives a result which is correct, until you attach other components to the midpoint of the divider which will influence it.
- When you do attach a component to the point between the resistors in the voltage divider, the resistance between that component and ground should be



Components

Resistors	Capacitors	Trimmers
100 (1)	0.01µF (1)	10K (2)
1K (3)	0.47µF (1)	500K (1)
10K (3)	10µF (1)	
470K (1)	100µF (1)	555 timers (2)
		8-ohm spkr (1)

Figure 16-17: Components-only layout of the dual-555 circuit.

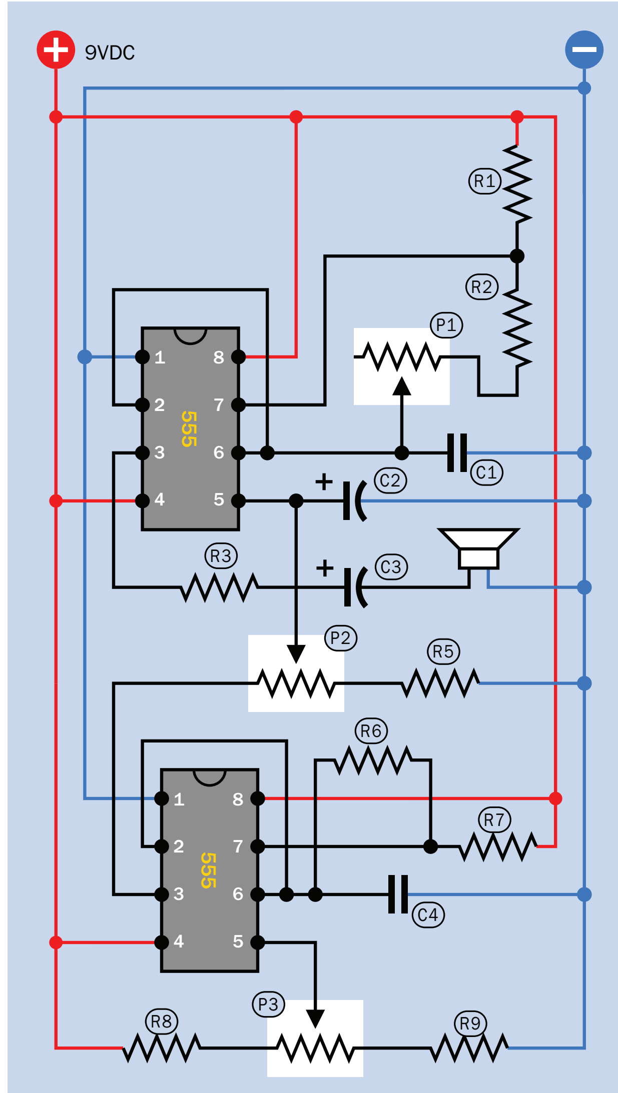


Figure 16-18: Schematic of the dual-555 circuit.

as high as possible compared with the values of the resistors in the divider.

That's another way of saying that the values of resistors in the divider should be relatively small—but of course if you make them too small, they will waste current, because they are sitting between the positive supply and negative ground. Therefore, choosing the ideal values is a compromise. For my timer circuit, I decided that two 1K resistors and a 10K trimmer were as low as I was willing to go.

Controlling One Chip with Another

Now I'll get back to the scenario which I mentioned: The output of a second timer hooked up with the Control Pin of the first timer.

Your astable circuit from Figure 16-12 should be working, right? Now you'll add a new circuit, lower down on the board, to control it. Figure 16-16 shows the complete breadboard. Figure 16-17 is the components-only version, so that you can see what you will need. Figure 16-18 is the schematic version, and if you compare it with Figure 16-11, you will see there are just a couple of differences: C2 has made a comeback, although it is now a 10µF electrolytic capacitor, for reasons which I will explain. Meanwhile R4 has been deleted, because the left side of P2 is now connected directly with Pin 3, the Output Pin, of the additional 555 timer; and the voltage on that pin will never be higher than 8V.

In the breadboard view, the speaker is no longer shown, because there wasn't room to include it; but you can see the wires that lead to it and from it.

The entire circuit fits into the top half of the breadboard, so there is no need for jumpers connecting the top half of each bus with the bottom half, even if you have a board with split buses.

If you power up the circuit, first try removing and then re-inserting capacitor C2. You'll find it creates a kind of wobbling sound, because it is functioning as a smoothing capacitor on the Control Pin of Timer #1. You can also experiment substituting other capacitors of different values.

P2 adjusts the amount of modulation on the Control Pin of Timer #1. R6 and R7 are the timing resistors for Timer #2, and C4 is the timing capacitor. I chose their values to create a frequency of about three beats per second, but this can be adjusted with a new voltage divider that I added at the bottom of the circuit, consisting of R8 and R9 in series with P3. The two resistors are 1K and P3 is 10K, like the voltage divider that you built previously.

I think you'll find that this is fun to play with. P1 still controls the pitch of the tone generated by Timer #1. P3 controls the speed of modulation created by IC2, while P2 adjusts the intensity of the modulation. If you turn down P2, you'll get something like the vibrato of an opera singer. If you turn it up, it will sound more like a car alarm.

An opera singer and a car alarm in one little circuit consisting of two timer chips? Nice!

Here's an idea. Maybe you can record your favorite sound from this circuit, and use it as a ring tone on your phone.

The only problem I foresee is that anyone who shares your home may find the sounds less interesting than you do. Be prepared for verbal feedback along the lines, "Turn that darned thing off!"

Now, are any additional modifications possible?

Of course! Additional modifications are always possible.

What do you think would happen if you added a third timer, with its output hooked up to the Control Pin of Timer #2? Just remove R8 and attach the output from the third timer to the yellow jumper at the bottom of the board, feeding in to P3. That would be worth a try. If the third timer ran very slowly, it could create a rising and falling siren effect, like a car alarm being used as a police siren. Two really irritating sounds in one!

How about if you wanted a tremolo effect, instead of a vibrato effect? Tremolo modulates the intensity of the sound, while vibrato modulates the frequency of the sound. The sound output from this circuit depends on the supply voltage, so you could adjust the supply voltage to Timer #1. Simply disconnect its power (unhook the red jumper at the top-right corner) and instead, power it with the pulsing output from Timer #2. Run a rather long wire from Pin 3 of Timer #2 to Pin 8 of Timer #1, and that's all you need. I tried this, and it works, but the on-off transition is a bit abrupt. Maybe a large smoothing capacitor would help. Or maybe a pullup resistor on Pin 8 of Timer #1. Try it, and see what happens.

So long as you don't connect power to the timers the wrong way around, I don't think you're likely to do anything that will damage them. And so long as you keep R3 and C3 in series with your speaker, they should protect it from any peculiar pulses that you send to it.

I think audio circuits are always a lot of fun to play with, but the next experiment has a more serious goal: To create the intrusion alarm that I have mentioned previously. You now have all the necessary knowledge to do this.

Experiment 17
An Alarming Idea

An intrusion alarm which sounds an audible warning inside your home won't be loud enough to alert your neighbors, but it can be useful to waken you when you're asleep. Also, if there are other people in your home, they may like the idea of an alarm which will notify them if a door or window is opened. Either way, this is an ideal project to make use of the knowledge and skills that you have acquired so far.

You will Need:

- Breadboard, hookup wire, wire cutters, wire strippers, multimeter.
- 9V AC adapter. If you run this circuit for long, it may use too much current for a 9V battery.
- Resistors: 100 ohms (1), 470 ohms (4), 10K (5), 47K (4), 470K (2).
- Capacitors: 0.01µF (2), 0.47µF (2), 10µF (3), 100µF (2).
- 555 timer chips (3).
- Tactile switch (1).
- 1N4148 signal diode (1).
- SPST or SPDT slide switch to fit breadboard (1).
- Generic red LEDs (4).

Planning

Often I have received emails from readers asking how to design circuits. I was never trained formally in this area, so I don't have an authoritative answer—but I can tell you how I approached this particular project myself, and I'll also describe how I made it work after I realized that I made a mistake halfway through.

I think the first requirement for circuit design is imagination. You have to imagine using the thing that you want to build, because if you start putting components together without thinking ahead, you're liable to find that you failed to consider something important.

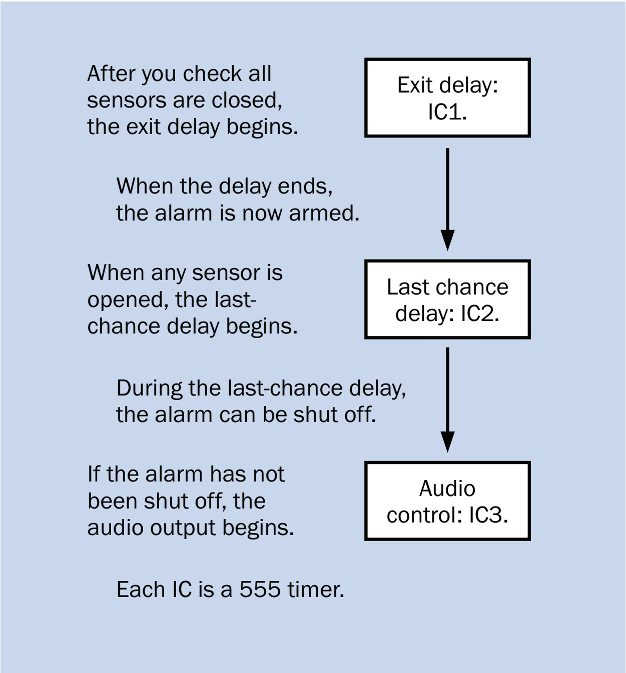


Figure 17-1. First attempt to visualize alarm functions.

With this in mind, I sketched out a preliminary plan, summarized in Figure 17-1. It refers to the timers as IC1, IC2, and IC3 because this is how integrated circuits are often described in diagrams. Here are the steps in detail:

1. You switch on the alarm. An LED lights up to confirm when all the doors and windows are closed.

2. If you're planning to leave the area, press a button to start a delay of maybe 30 seconds, during which you can open a door without triggering the alarm. I call this the "exit delay."

3. At the end of the exit delay, the alarm arms itself.

4. Now if a door or window is opened, this starts another delay, which I call the "last-chance delay." During this period, if you return to the area by opening a door, you have a last chance to cancel the alarm without it going off.

5. If you don't cancel the alarm by the end of the last-chance delay, it starts to make a noise.

I thought this sequence sounded workable, so my next step was to choose components. In the Second Edition of *Make: Electronics* I suggested an alarm circuit that used transistors and a relay, but having had a few years to reconsider my options, I decided that I could simplify the project with only three 555 timers.

Magnetic Sensors

Before I go any further, I should pause to explain how a simple alarm system works. Typically, you fit each door or window with a sensor consisting of two modules such as those shown in Figure 17-2. One module contains a magnet while the other contains a *reed switch*.

The switch consists of two contacts inside a sealed glass capsule. The contacts are magnetized, so that a magnetic field will push them together (if they are normally open) or pull them apart (if they are normally closed).

Figure 17-3 shows a reed switch, just in case you're wondering what it looks like, and Figure 17-4 shows how the alarm sensor works. (Letters N and S stand for north and south, the two poles of a magnet.) Note that if you ever buy sensors to make your own alarm system, you should choose the type with contacts that are normally open, not the type with contacts that are normally closed.

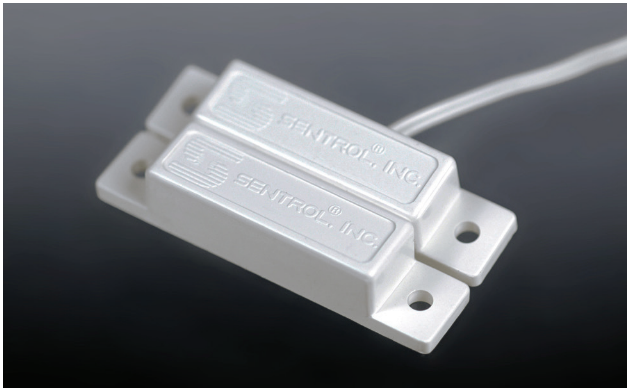


Figure 17-2. The two modules of an alarm sensor.

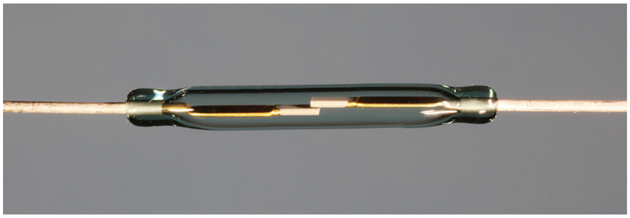


Figure 17-3. A typical reed switch. The glass capsule is about 2mm in diameter.

You mount each magnetic module on a door or a window, and you mount each switch module on the adjacent frame so that it almost touches the magnetic module when the door or the window is closed. In Figure 17-5,

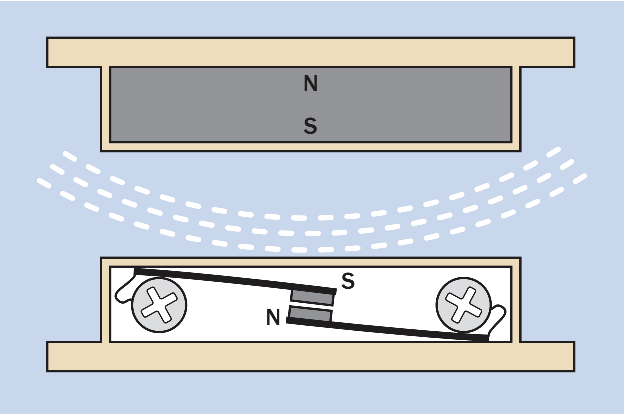


Figure 17-4. When a magnetic module approaches the reed switch, the magnetic field will move the contacts.

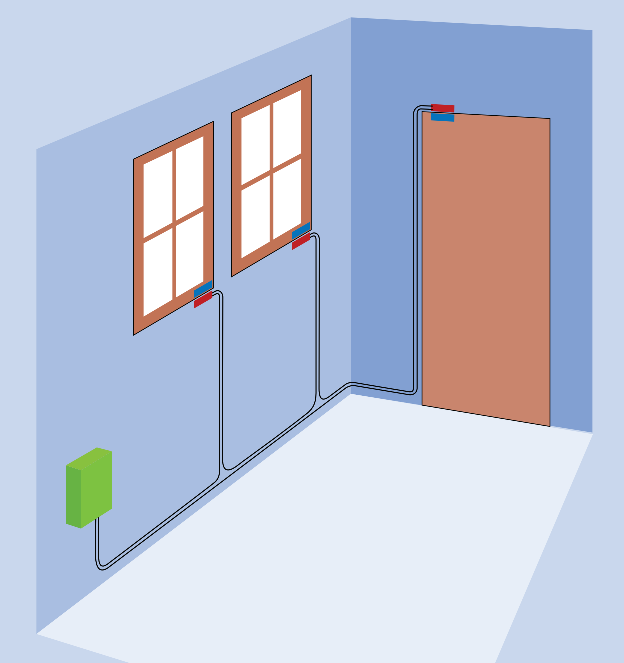


Figure 17-5. Alarm sensors wired in series. Blue indicates the magnetic module, while red contains the reed switch.

the magnetic modules are blue rectangles while the switch modules are red and the alarm is in a green box. Because the switches are wired in series, if one switch opens, the continuity of the circuit is interrupted.

This is known as a "break-to-make" type of circuit, because any break in it makes the alarm go off. The circuit will also be broken if someone tries to interfere with it—for instance, by cutting the wire.

My first step was to decide how this system would interface with IC2, which controls the mid-section of the sequence shown in Figure 17-1. If I couldn't figure out how to trigger a timer with the alarm sensors, my experiment wasn't going to get very far.

Keeping the Timers Happy

I tend to think of components as being like living creatures. My task is to keep them happy by giving them what they need, so that they live a long time and behave themselves.

In the case of a 555 timer, it needs a clearly defined voltage on its Trigger Pin. As you already know from Experiment 15, if the voltage on Pin 2 drops below one-third of the power supply, the output from the timer goes high. If the voltage stays above one-third of the power supply, the output is low and the timer just sits there and waits. But if no particular voltage is applied to the Trigger Pin, the timer will behave unpredictably. It will not be happy, and if it is not happy, I will not be happy.

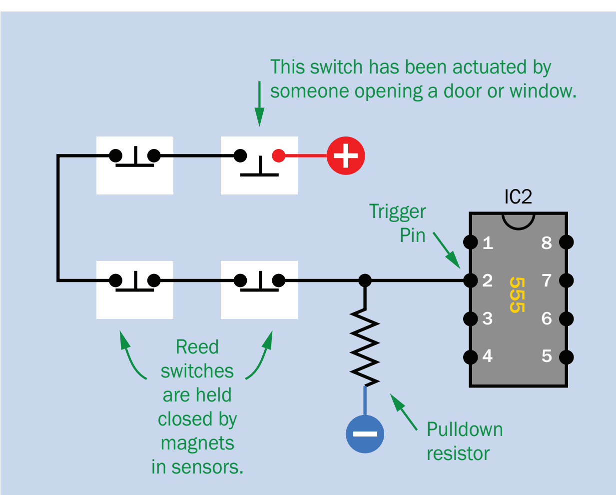


Figure 17-6. Basic concept for an alarm sensor circuit.

With this in mind, I decided to apply a positive voltage to one end of the series of alarm sensors, which lead to the Trigger Pin of the timer, as shown in Figure 17-6. So long as all the sensors are closed, the Trigger Pin has a relatively high voltage, and the timer remains happy, doing nothing.

If any one switch is opened, the positive connection is broken. In this situation a pulldown resistor takes over and triggers the timer. When the timer is triggered, its output goes high—which I thought I should be able to use in some way, although I wasn't quite sure how, yet.

Incidentally you saw a similar arrangement in Figure 15-12, except that in that case I had pullup resistors instead of pulldown resistors, and the pushbutton had a negative connection.

Next I thought about IC1. Its job is to create a 30-second pause which will provide the exit delay. A high output from IC1 somehow has to ***prevent*** IC2 from being triggered when any of the alarm sensors are opened during the exit delay.

Figure 17-7 shows a very simple way to do this. First, you press the "Go Button" when you're ready to go, and this triggers a single high pulse from IC1, because it will be wired in monostable mode. Let's suppose the output

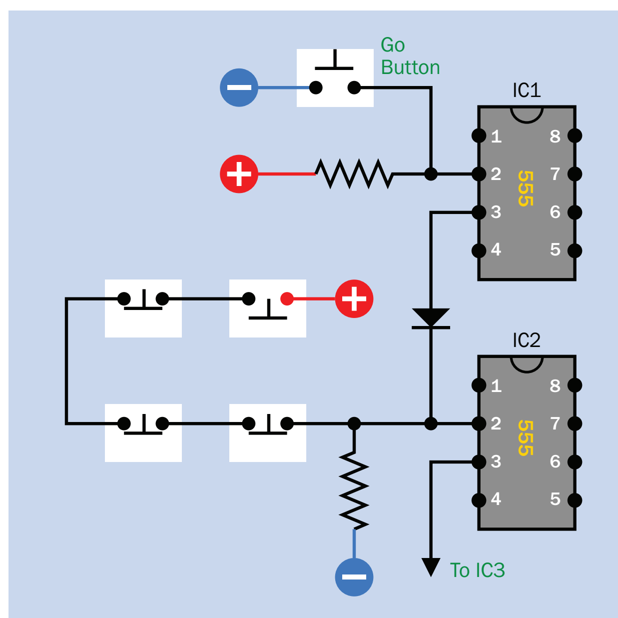


Figure 17-7. Adding IC1 to provide a 30-second grace period .

from IC1 stays high for 30 seconds. It passes through a diode, overcomes the pulldown resistor, and makes sure that the Trigger Pin of IC2 stays in its high state, even if someone opens a sensor switch. So long as the Trigger Pin is high, IC2 will ignore the sensor switches.

You can think of the diode and the pulldown resistor as being like a voltage divider. The diode has some effective resistance, but very little compared with the pulldown resistor, which will be 10K. Therefore, the voltage between them will be close to the output voltage of IC1, which is about 8V. That's much more than necessary to stop the pulldown resistor from triggering Pin 2 of IC2.

But why did I use a diode? Because the output from IC1 is low, when it is not creating the exit pulse. I didn't want the low output to get through to IC2, so I blocked it with the diode.

Summing up my progress so far:

- You press a button to start IC1, and it generates a high output lasting for 30 seconds.
- You open a door to leave the area, but IC2 doesn't notice, because the voltage from IC1 prevents it from being triggered.
- At the end of the exit delay, IC1's output goes low. It doesn't have any effect on IC2 anymore, and the alarm is sensitive to any switch being opened.
- You come home and open the door, which triggers IC2. This will be wired in monostable mode, so it will create its own high output for 30 seconds. I'm not sure exactly how to use that, yet, but during the last-chance delay you can switch off the alarm. If you don't, the high pulse from IC2 will end after 30 seconds, and will trigger IC3—somehow, to be determined.

This all seemed pretty easy. But maybe it was too easy. Do you see the big flaw in this scenario? I didn't see it, till I thought some more.

Here's the problem. After the exit delay, the alarm is armed and ready to detect an intrusion. Suppose someone enters through a door or window, and then ***leaves it open?*** The sensor switch triggers IC2, which starts the last-chance delay. But maybe you remember that when the Input Pin of a 555 timer is held low, the high output continues indefinitely. Consequently, if the intruder leaves a door or window open, the Input Pin of IC2 stays low, and this will keep retriggering IC2, so its output will stay high indefinitely. The last-chance delay will never end, and IC2 will never tell IC3 to sound the alarm.

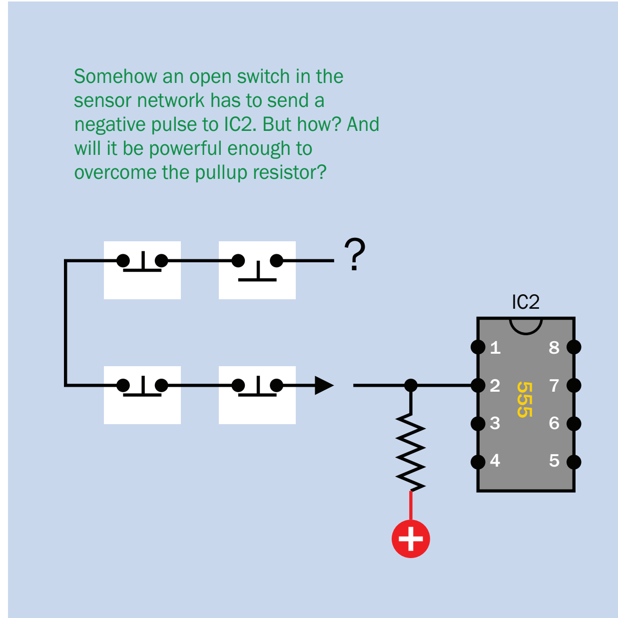


Figure 17-8. How to send a negative pulse to IC2?

Oops.

Now I had to rethink everything, and I saw the mistake that I had made: I failed to keep IC2 happy. The 555 timer should never be put in a position where it is triggered indefinitely by an input that is stuck in a low state. The timer was designed for a ***brief pulse*** to trigger it.

Really, IC2 should have a pullup resistor, not a pulldown resistor. That's what a 555 timer likes: To keep its Trigger Pin normally high. Then it can be triggered by a brief low pulse (one way or another) before the pin goes back to a high state.

How was I going to arrange this? Figure 17-8 illustrates the problem.

Well, whenever I need a brief pulse, I think of a coupling capacitor. If you flip back to Figure 9-12, you'll recall that a sudden change in potential on one plate of a capacitor sends a pulse of displacement current through to the other plate. It's a high pulse when the first plate has a sudden increase in voltage, but it will be a low pulse if there's a sudden decrease.

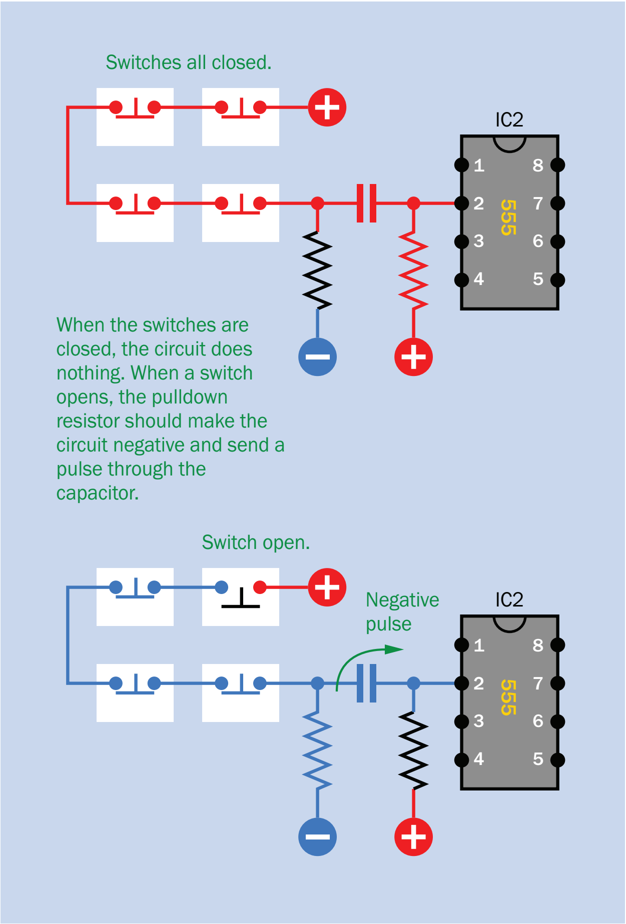


Figure 17-9. A way of wiring the sensor switches to create a negative pulse.

(Really it's not quite accurate to think in terms of a "high pulse" or a "low pulse," because both of these events are just flows of electrons in one direction or the other. But high and low pulses are a convenient way to imagine what happens.)

I wondered if I could connect the sensor switches as in Figure 17-9. With all switches closed, the only current drain is through a pulldown resistor, which is easily satisfied by the positive connection at the far end of the chain of switches. Meanwhile, the pullup resistor on Pin 2 of IC2 keeps the pin positive, and the timer sits happily doing nothing.

When a switch is opened, as shown in the lower half of Figure 17-9, the positive connection is broken suddenly. The left side of the capacitor goes negative, because of the pulldown resistor. This sudden change in voltage sends a low pulse through the capacitor, which overcomes the influence of the pullup resistor.

The colors in the schematic are a slight exaggeration, because the input to the chip won't be completely up at the level of the supply voltage, and when a switch is opened (in the lower half of the schematic) the circuit won't be completely down at 0VDC. It should be close, though.

Would it work? The only way to find out was by breadboarding it, and—no, it didn't work reliably.

Big disappointment! If an alarm system isn't reliable, it isn't much use, and at this point I felt discouraged and annoyed by my lack of success at pursuing a project that I thought should be simple.

In a situation like this, it's a good idea to take a break. By the time I returned, my head was clearer and I felt convinced that it ***should*** work, so I did what I should have done previously: I used a methodical approach. I tried some different values for the pullup resistor (which had to be there, to keep the input of IC2 high when nothing else was happening). I also changed the pulldown resistor, and the size of the capacitor (which had to be big enough to deliver a substantial pulse).

Initially I had used a 0.1µF capacitor and two 10K resistors. I ended up with a 0.47µF capacitor, a 10K pulldown resistor, and a 47K pullup resistor. Now I got a negative pulse that dipped all the way down below 1V, when I measured it on the Trigger Pin of the timer. This was far below the 3V necessary trigger the chip.

So, maybe this was going to work, after all.

Now I had to go back and include IC1 with its diode. I still needed to make sure that when the output from IC1 went high during the exit delay, and positive current flowed down through the diode, and this would be enough to keep the Input Pin of IC2 high, so that it would ignore sensor switches. In other words, the positive voltage would suppress any negative pulses coming through the capacitor. Figure 17-10, on the next page, shows what I had in mind.

This turned out to work, when I measured the voltage on the Input Pin of IC2, so I could proceed with building the final circuit. Moral of the story: When something doesn't work, take a break and be methodical.

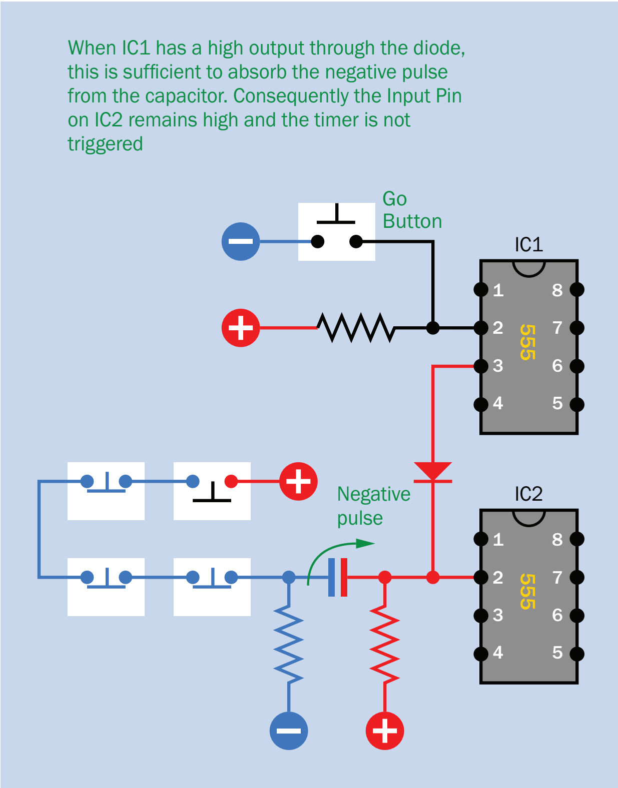


Figure 17-10. Positive current from IC1 through the diode may be sufficient to suppress the negative pulse through the capacitor.

I also figured out how IC2 could trigger IC3. I needed to connect them through another coupling capacitor, to create another low pulse. I'll describe that in a moment.

In Figure 17-11 you see the complete breadboarded circuit. If the number of components looks intimidating, bear in mind that the patterns of resistors and capacitors around each timer are basically the same as in the test circuits that you saw during Experiment 15. The only new concept is the way in which the timers are linked together.

I used just one switch at the top of the board to simulate a chain of switches in a real alarm system. When the switch is closed, this is the same as a chain of sensor switches in which they are all closed. When the switch is open, this is equivalent to one sensor switch opening.

In Figure 17-12 you see the component values. Notice that for testing and development, I used 47K timing resistors with IC1 and IC2. They create only 3-second delays, because I didn't want to hang around waiting during a 30-second delay when I was testing the circuit. If you want to build a finished version, use 470K timing resistors instead of 47K resistors. They are included in the component list.

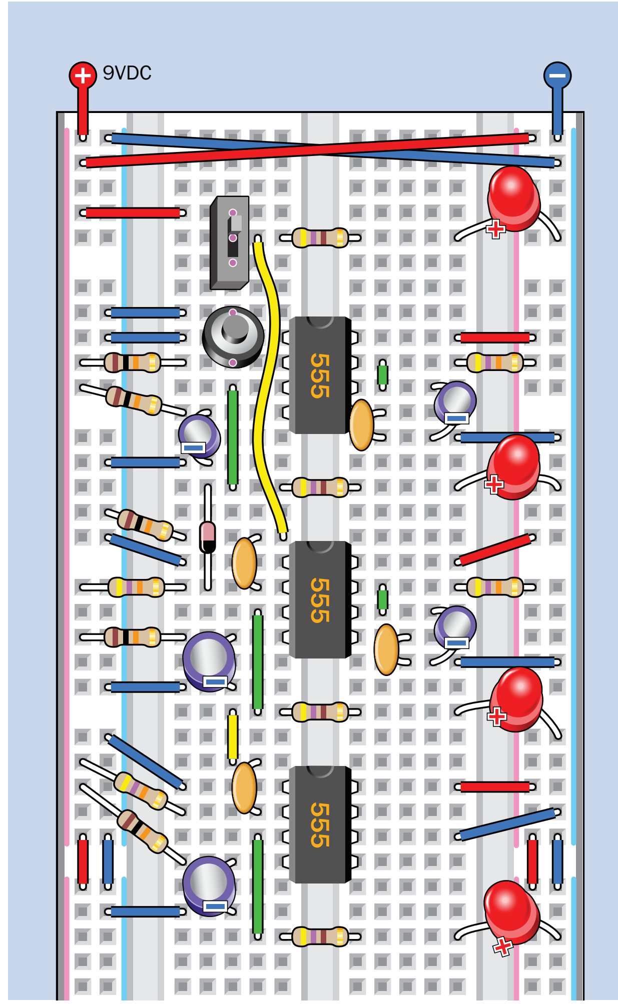
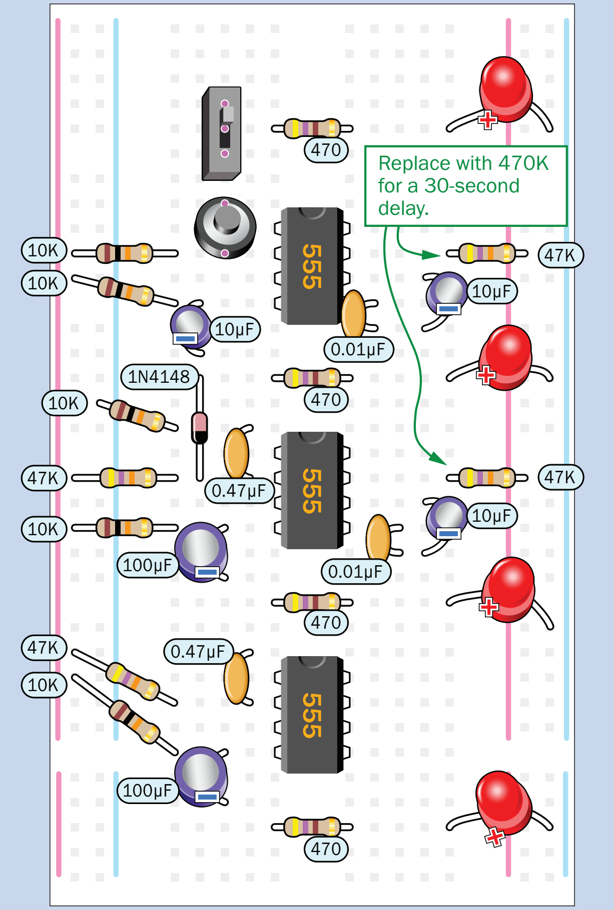


Figure 17-11. The complete circuit for the intrusion alarm.

In Figure 17-13 is the schematic. Here you will see how I added IC3, with C8, another coupling capacitor. When the last-chance delay created by IC2 comes to an end, the sudden drop in voltage from its Output Pin creates a negative pulse through C8. This triggers IC3, which is wired in bistable mode, so that its output sticks low or high. Its output in the schematic just passes through an LED, to show you what's happening, but for a real alarm, you would use the output from IC3 as a power supply for



Components

Resistors	Capacitors	LEDs (4)
470 (4)	0.01μF (2)	555 timers (3)
10K (5)	0.47μF (2)	1N4148 diode (1)
47K (4)	10μF (3)	Tactile sw (1)
470K (2)	100μF (2)	1P2T sw (1)

Figure 17-12. The components-only version of the intrusion alarm circuit.

some kind of audio alert. This could be an off-the-shelf beeper, or one of the astable multivibrator circuits that I described previously. The 8VDC output from IC3 is quite sufficient to power one of those circuits.

When you're testing the circuit, first close the sensor switch to simulate closing all the doors and windows

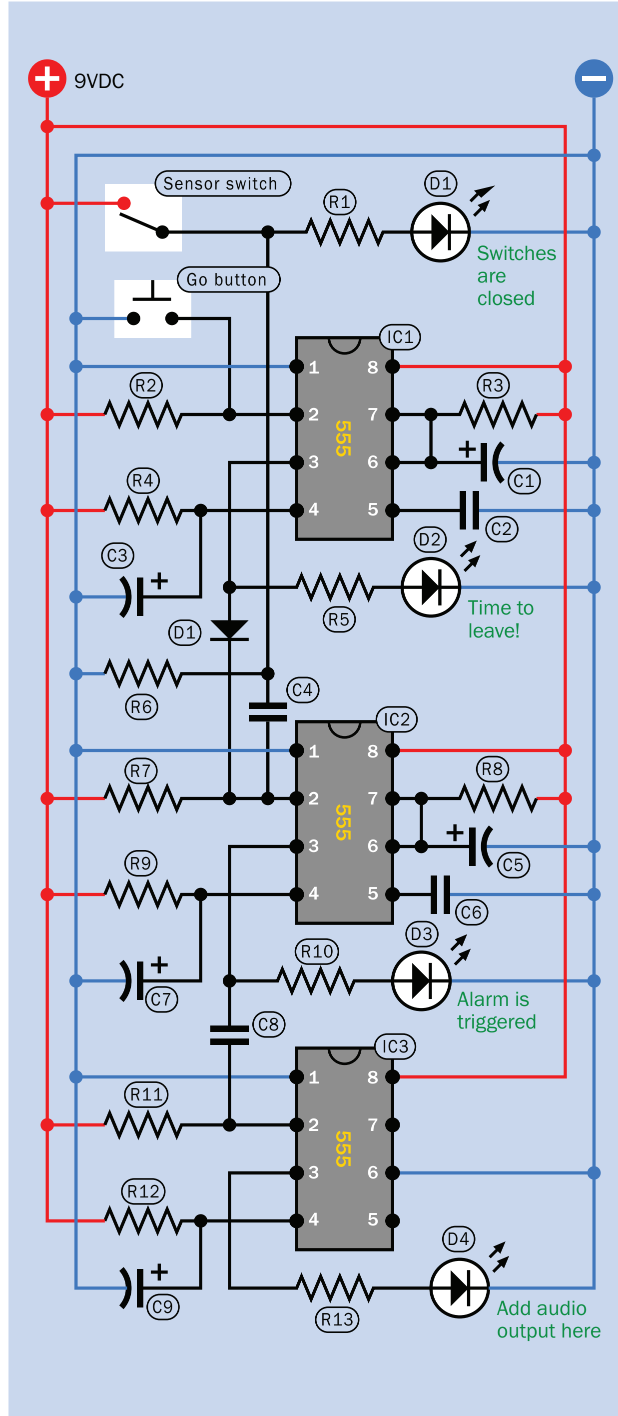


Figure 17-13. Schematic for the intrusion alarm.

where the alarm system has been installed. D1 lights up to confirm the continuity.

Now you can press the Go button, which lights D2, the "Time to leave!" indicator, which warns you that you only have 30 seconds. While D2 is on, the exit delay is in effect. You can open the sensor switch, and nothing will happen. Using the test values for R3 and C1, the delay only lasts for three seconds, but still that's enough time for you to open and close the sensor switch.

At the end of the exit delay, D2 goes out. Now the alarm is armed and ready, and if you open the sensor switch, this sends a negative pulse through C4 and triggers IC2, which begins the last-chance delay. During that period, D3 lights up to warn you that you have one last chance to switch the alarm off.

At the end of the last-chance delay, the output from Pin 3 of IC2 drops to a low state, which sends a negative pulse through C8. This triggers IC3, which switches on and keeps itself switched on, because it is wired in bistable mode. D4 lights up to tell you that if an audio source was attached, it would be receiving power from IC3 and making a noise.

Just in case this sequence still isn't entirely clear, I added a diagram in Figure 17-14 showing how the timers trigger each other.

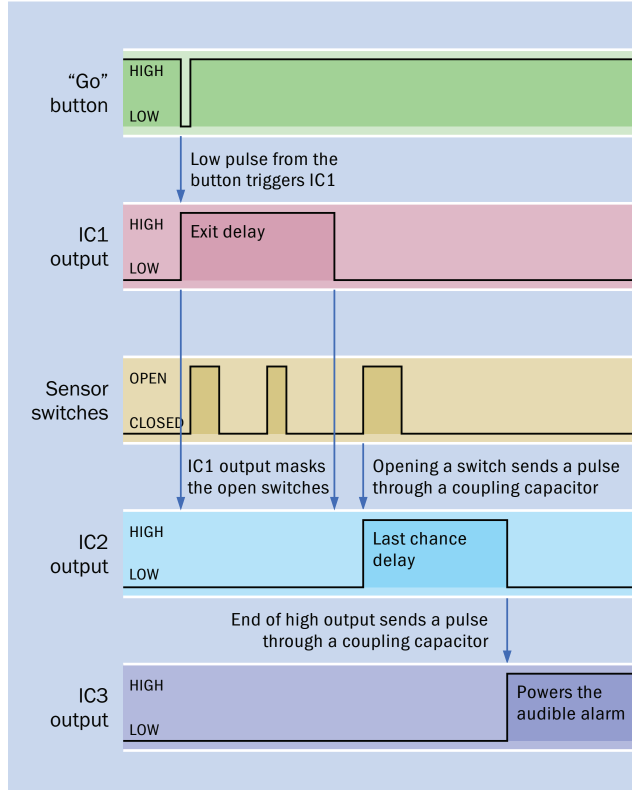


Figure 17-14. Interactions between the timers.

The only fault that I noticed while testing the circuit was that when I first turned on the power, sometimes IC2 triggered itself. Maybe this was associated with a relatively long wire run to its Trigger Pin, but I fixed it by increasing the capacitor values for C7 and C9 to 100μF instead of the usual 1μF which I use to stop timers from emitting an initial pulse.

These 100μF capacitors can cause a little snag of their own: When you switch the power off and then back on again very quickly, they still have some charge on them, so they trigger IC2 and IC3 instead of suppressing them. To avoid this, leave the circuit off for ten seconds before you switch it back on.

After I started testing the circuit I started thinking of some additional features. In particular, I would like another 555 timer that would create a stream of little beeps during the exit delay, to add a feeling of urgency. This same feature could be triggered during the last-chance delay, to remind you to switch off the alarm. I'll leave it to you to think about adding that feature.

Another nice feature would be a keypad requiring you to enter a code to switch off the power to the alarm. In the next experiment I'll provide a keypad circuit—but I've gone far enough, now, with the alarm project.

Finalizing

What if you want make a permanent, installed version of the alarm circuit? I'm going to go through the steps that would be involved. (Because these enhancements are optional, items such as switches, terminals, and a project box probably will not be included in basic kits for this book.)

This circuit is ideally suited for the type of perforated board that is plated with conductors in the same pattern as your breadboard. In fact, the components can fit on a half-size board such as the one in Figure 17-15, which I bought online from Adafruit, one of the suppliers listed in Appendix B. (Remember to consider using sockets for the timer chips. See Figure 15-4.)

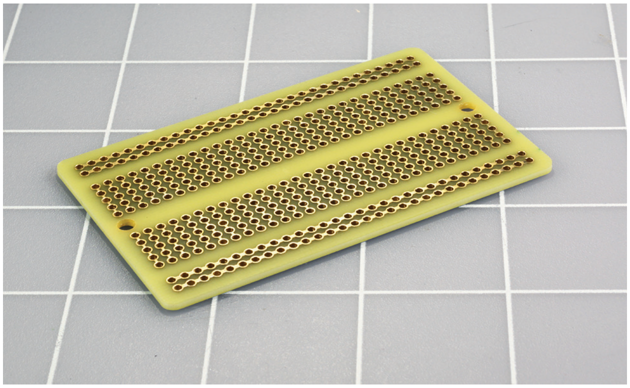


Figure 17-15. A half-size perforated board plated with conductors in breadboard configuration.

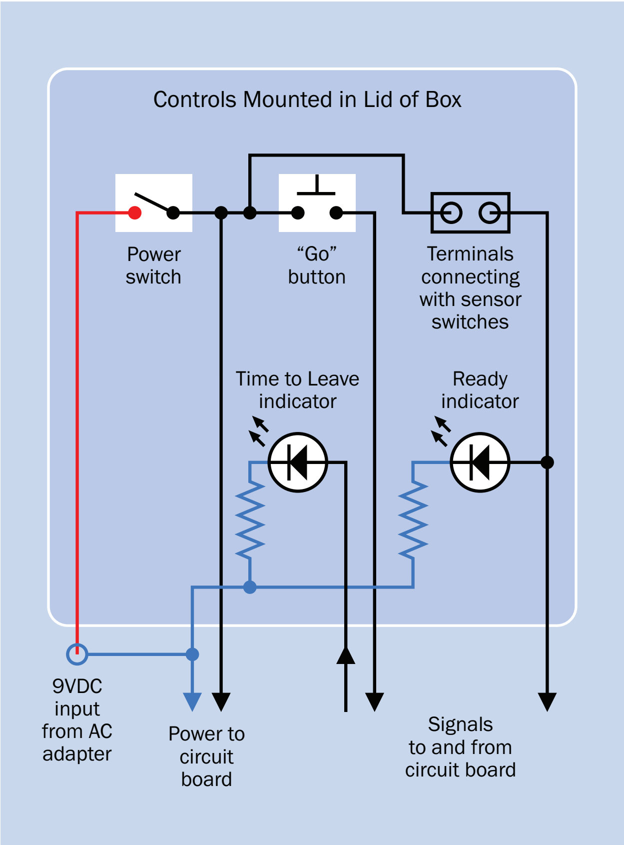


Figure 17-16. These components could be mounted in the lid of a project box.

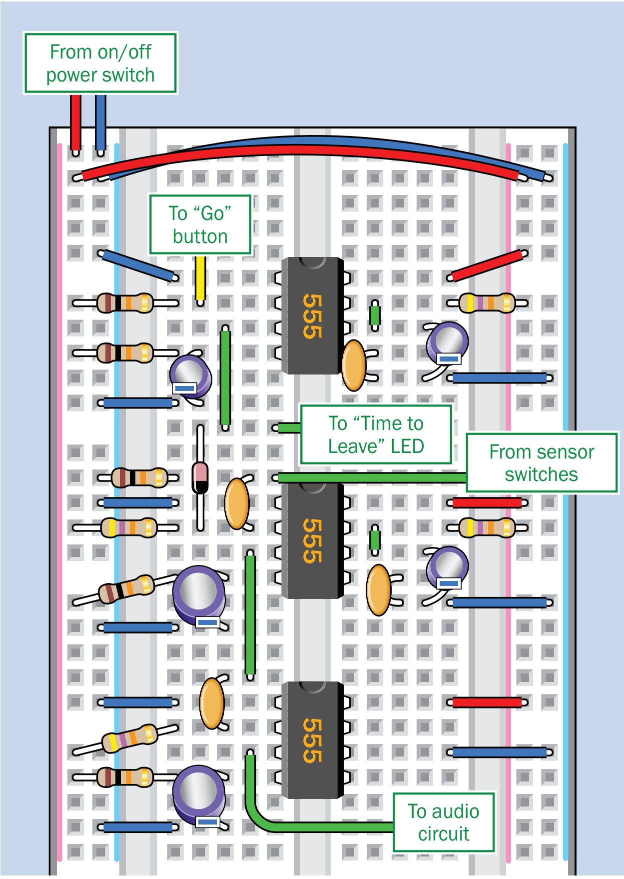


Figure 17-17. If the LEDs, switch, and pushbutton are relocated into the lid, the remaining parts will fit easily on a half-size board.

In Figure 17-16 I've suggested how some of the components could be relocated off the circuit board, under the lid of a project box.

In Figure 17-17, you can see how the circuit is minimized after the inputs and outputs have been relocated off the board.

Choosing an actual on-off switch, a pushbutton, LEDs, and terminals for the alarm wires is a matter of personal preference. Do you prefer 5mm or 3mm LEDs? Do you like a big, clunky switch, or a tiny one? Personally, when I'm shopping for switches, I like to see them up-close instead of buying them online, so I buy them at automobile accessory stores and big-box hardware stores.

If you use a plastic project box, you can print a layout such as the one shown in Figure 17-18, and mark the features in the plastic by pricking through the paper with an awl. Then measure the components and drill holes of an appropriate size, as in Figure 17-19.

I'm assuming that if you take the trouble to make an enclosure for your circuit, you'll want to solder the components, and you may find copper-plated perforated board easier to deal with than the unplated board I used in Experiment 14. Figures 17-20 through 17-23 illustrate the process. I have used 3D renderings, as actual solder joints are difficult to photograph clearly.

A copper trace surrounds each hole and links it with others. Your task is to melt solder so that it sticks to the copper and also to the wire, but not to any other copper traces or any other wires.

Figure 17-18. A layout for switches and LEDs under the box lid.

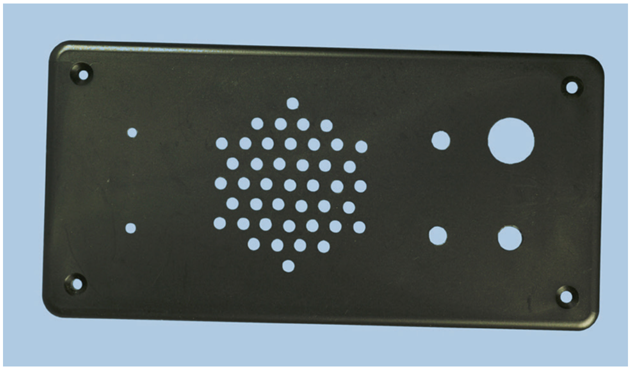


Figure 17-19. The box lid after drilling.

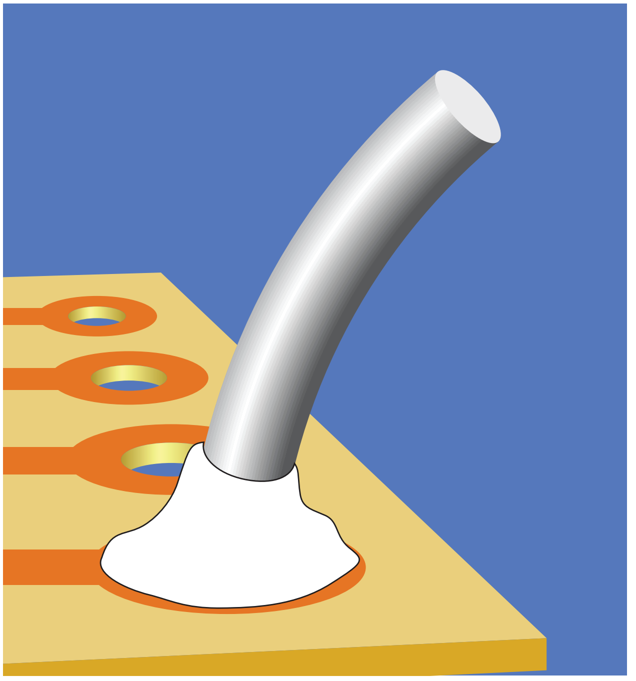


Figure 17-20. The underside of perforated board, with an ideal amount of solder.

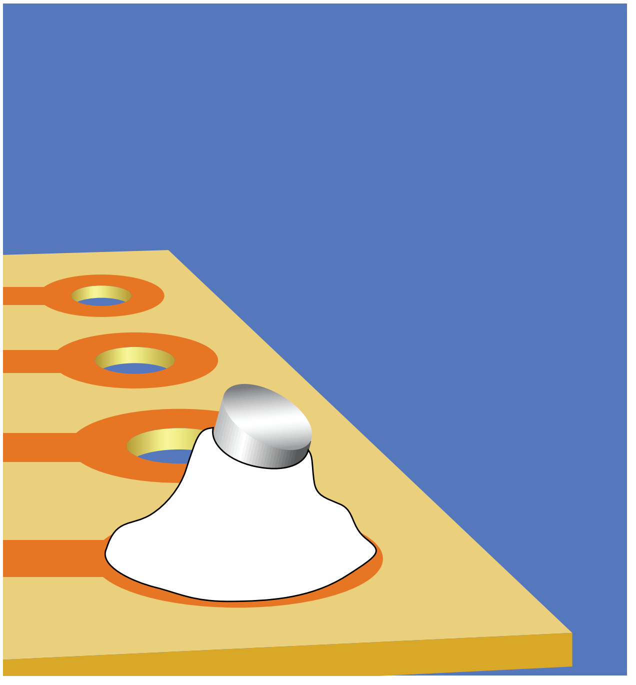


Figure 17-21. After the solder has cooled and hardened, you snip off the projecting wire.

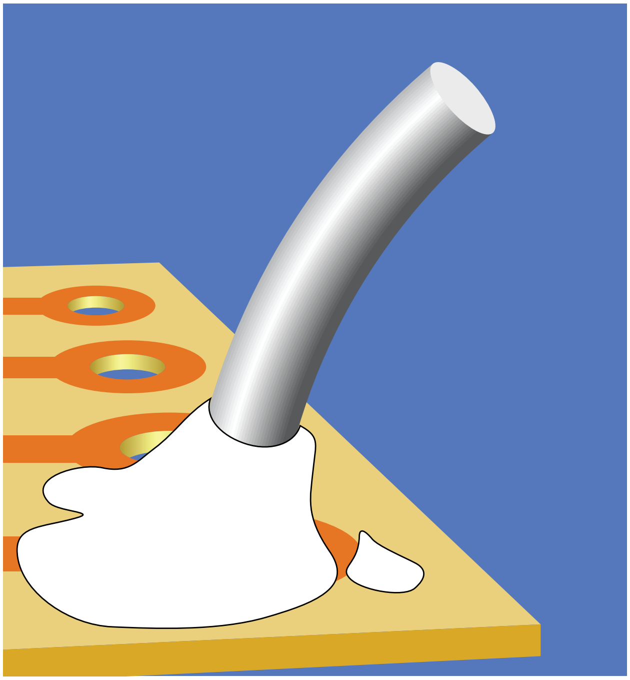


Figure 17-22. If you use too much solder, it will end up in places where you don't want it.

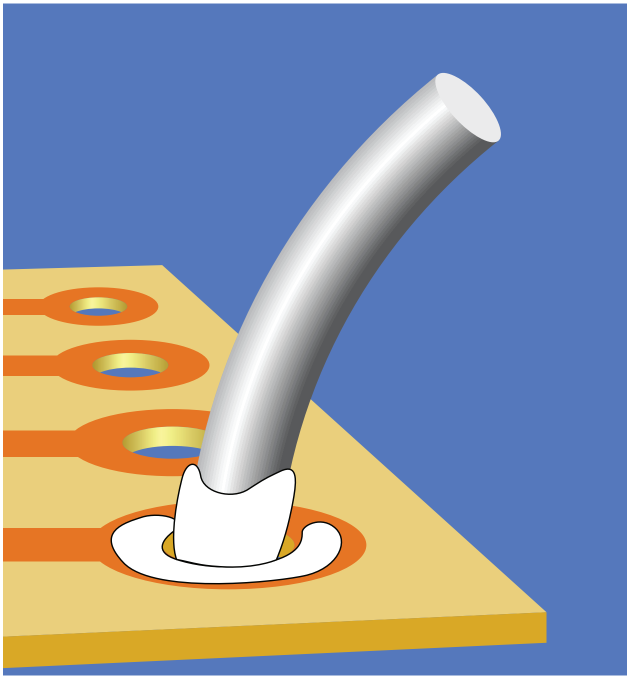


Figure 17-23. Insufficient solder may result in a tiny gap between the wire and the copper trace, causing an intermitent fault.

Hold the perforated board with Helping Hands or some similar gadget, bend the pins of the component outward slightly to prevent it from falling out of the board, and remember to heat the wire *and* the copper trace on the board before you try to apply solder. After five to ten seconds, the solder should start to flow.

Allow enough solder to form a rounded bump sealing the wire and the copper, as shown in 17-20. Wait for the solder to harden thoroughly, and then grab the wire with pointed-nosed pliers and wiggle it to make sure you have a strong connection. If all is well, snip the protruding wire with your cutters as shown in Figure 17-21.

Common Perfboarding Errors

Too much solder. Before you know it, solder creeps across the board, touches the next copper trace, and sticks to it, as depicted in Figure 17-22. When this happens, you can either try to suck it up with a desoldering kit, or scrape it away with a knife. Personally I prefer to use a knife, because if you suck it up with a rubber bulb or solder wick, some of it will tend to remain.

Even a microscopic trace of solder is enough to create a short circuit. Check the wiring with a magnifying glass while turning the perforated board so that the light strikes it from different angles.

Not enough solder. If the joint is thin, the wire can break free from the solder as it cools. Even a microscopic crack is sufficient to stop the circuit from working. Sometimes the solder sticks to the wire, and sticks to the copper trace around the wire, yet doesn't make a solid bridge connecting the two, leaving the wire encircled by solder yet untouched by it, as shown in Figure 17-23. You may not be able to see this on your work without magnification.

If you need to add more solder to any joint that didn't get enough, be sure to reheat the joint thoroughly.

Components incorrectly placed. It's very easy to insert a component one hole away from the position where it should be. It's also easy to forget to make a connection. I suggest that you print an enlarged copy of the schematic, and each time you make a connection on the perforated board, eliminate that wire on your hardcopy, using a highlighter.

Debris. When you're trimming wires, the little fragments that you cut don't disappear. They start to clutter your work area, and one of them can easily get trapped under the wires in your perforated board, creating an electrical connection where you don't want it.

Clean the underside of your board with an old (dry) toothbrush before you apply power to it, and keep your work area as neat as possible. The more meticulous you are, the fewer problems you'll have later.

And, I know I'm repeating myself, but always check every joint with a magnifying glass.

Perforated Board Fault Tracing

If the circuit that worked on your breadboard doesn't work after you solder it to perforated board, your fault tracing procedure will be a little different from that which I outlined previously.

First look at component placement, because this is the easiest thing to verify. If all the components are placed correctly, anchor the black lead of your meter to the negative side of the power supply, then switch on the power and go through the circuit point by point, from top to bottom, checking the voltage at each point with the red lead of the meter while you flex the board. In most circuits, almost every part should show at least some voltage. If there is a dead zone, or if your meter responds intermittently, you can zero in on a joint that has something wrong with it, even though looks good superficially. Remember that a gap of 0.001" or less is quite sufficient to stop your circuit from working.

Figure 17-24 shows plated perforated board part-way through the soldering process.

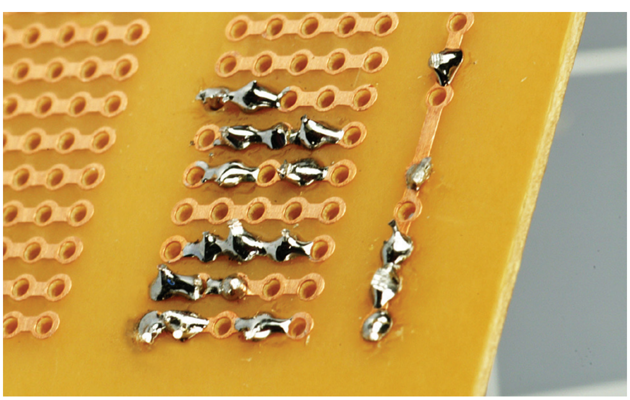


Figure 17-24. Not the neatest job, but adequate.

Installation

You can use epoxy glue to mount your LEDs and speaker into the underside of the lid of your project box, as in Figure 17-25. A switch or pushbutton is usually (but not always) manufactured with a threaded neck, and should be sold with a nut to fit. If you try to buy a nut for a pre-existing switch, you'll find a variety of sizes, some of them metric and some not. Buying a matched set of nut and switch is much easier.

If you have calipers, they'll be very useful for taking measurements of each part and drilling holes of the correct side. Otherwise, make your best guess for the measurement, using a ruler. Now choose the next-smaller drill bit, and enlarge the hole afterward if necessary. A hand tool known as a ***reamer*** is specifically designed to enlarge holes fractionally. A ***deburring tool*** can be used for the same purpose. Note that 3/16" holes are a fraction too small for 5mm LEDs, but with just a little reaming, they become a good fit.

Figure 17-25. Mounting components under the lid of the box.

Drilling large holes in the thin, soft plastic of a project box can be a challenge. The drill bit tends to dig in and create a mess. You can approach this problem in one of three ways:

- Use a Forstner drill bit if you have one. It creates a very clean hole.
- Drill a series of holes of increasing size.
- Drill a small hole, then make it bigger with a countersink bit, which won't dig into the plastic in the same way as a drill bit.

Regardless of which approach you use, you'll need to clamp or hold the lid of the project box with its outside surface face-down on a piece of scrap wood. Always drill from the inside, so that your bit will pass through the plastic and into the wood.

Before installing a switch, Use your meter to find out which terminals are connected when you flip the switch, so that you have it facing the right way up. Remember, the center terminal underneath any double-throw switch is almost always the pole of the switch. You only need a single-throw switch to supply power to this circuit, but I ended up with a double-throw switch when single-throw was unavailable in the style that I preferred.

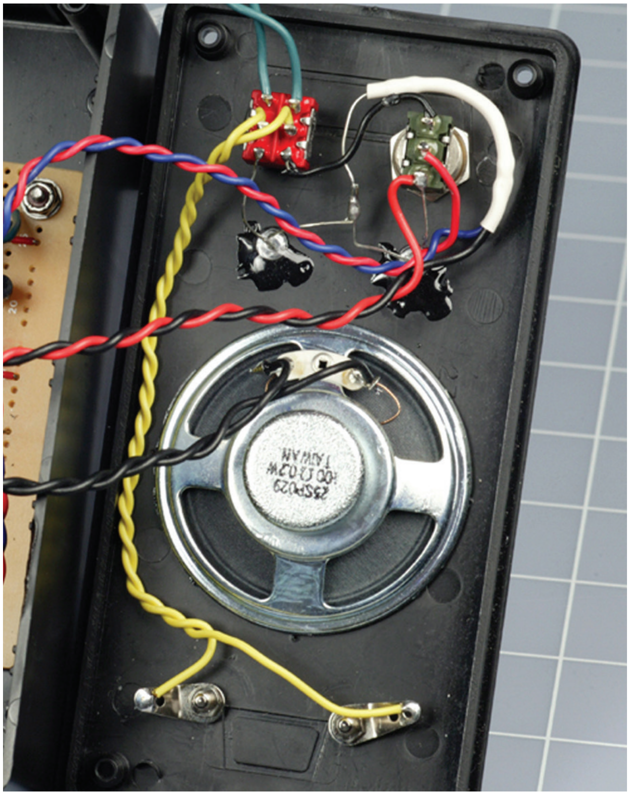


Figure 17-26. Twisted wire-pairs connecting with the board.

When you solder wires or components to the terminals on a switch, your 15-watt soldering iron will have to struggle to deliver enough heat as the terminals suck it away. A 30-watt iron makes the job much easier, but in that case you will need to use a heat sink when you solder the LEDs.

Figure 17-26 shows twisted pairs of wires soldered to components that have been mounted under the lid of a project box. In projects that are more complex than this one, you may want to minimize the wiring tangle by linking the top panel with the circuit board with multicolored ***ribbon cable***. Then you can use miniature plug-and-socket connectors called ***headers*** that will allow you to remove your circuit board later if a joint goes bad and you want to test it easily.

Placing the Board

The circuit board can sit in the bottom of the box, held in place with #4-size machine screws (bolts) with washers and nylon-insert locknuts. I prefer using nuts and bolts, rather than glue, so that the board is removable. Locknuts will eliminate the risk of a nut working loose and falling among components where it can cause a short circuit.

You may need to drill mounting holes in your perforated board, and you may have to cut it to size with a hacksaw. Remember that perforated board often contains glass fibers that can blunt a wood saw. Check the underside the board for loose fragments of copper traces after you finish cutting.

Use nylon washers or other spacers on bolts under the perforated board, instead of screwing it down against the inside of the box. Lumpy solder joints under the board may prevent it from sitting flush with the box, and if you tighten the bolts, you can impose bending stresses which may break a joint or a copper trace on the board.

If you are using an aluminum box, you must place a sheet of insulating material under your circuit board to prevent it from touching the box and creating short circuits. You can be inventive when choosing the material. An old mouse pad, for instance, can be cut to fit, or the type of plastic sheet that is sold to go in kitchen drawers. Whatever you use, it should be soft and flexible.

If you choose to create the alarm sound by using one of the astable multivibrator circuits that I discussed previously, it would be mounted on a separate board. To stack it with the main board, you can be creative. Taping the boards together with bubble wrap between them is not out of the question! My feeling is that if people aren't going to see inside the box, I don't have to be too concerned about appearances.

Wiring

To connect your box with a network of alarm sensors, you can install a pair of ***binding posts*** on your box. My completed version is shown in Figure 17-27, with the binding posts at the bottom.

If you are going to complete this project with sensor switches, test each one first by moving the magnetic module near the switch module and then away from it, while you use your meter to check continuity.

- Remember, the switch should close when it's next to the magnet, and open when the magnet is removed.

Before you start installing the switches, draw a sketch of how you'll wire them together. Always remember that they have to be in series, not in parallel! You can refer back to Figure 17-5 for the concept of laying alarm wire. Figure 17-28 shows how to join two-conductor wire so that your switches are in series. If you are going to cover the joints with heat-shrink tubing, remember to slide the pieces of tubing over the wires ***before*** you make the solder joints, and be careful not to heat the tubing by accident with your soldering iron. Leave an extra couple of inches of wire in each branch of the circuit, and cut it to the exact length after your have made the joints.

Figure 17-27. The finished box.

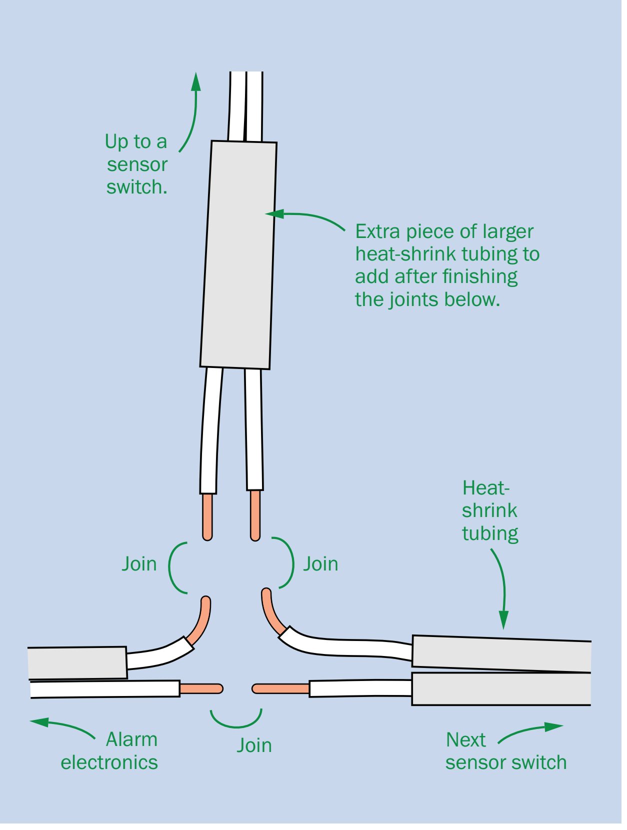


Figure 17-28. Adding a branch to your alarm wire.

The type of low-voltage, dual-conductor, white, stranded wire that is sold for doorbells or furnace thermostats is appropriate for connecting alarm sensors. It should be at least 20-gauge. You should find that the total resistance of all the wire in the circuit is less than 50 ohms, when you finish wiring it.

After you install all the switches, set your meter to test continuity, attach it to the circuit, and open each window or door, one at a time, to check that the circuit works properly. If everything is okay, attach the alarm wires to the binding posts on your project box.

The only remaining task is to label the switch, button, LEDs, and binding posts on the box. You know their functions, but no one else knows, and you might want to allow a guest to use your alarm while you're away. In any case, months or years from now, you may forget some details yourself.

Conclusion

Here's a summary of the steps in this project:

- Imagine how you'll use the device.
- Decide which types of components are appropriate.
- Sketch a block diagram of functions.
- Draw a simplified schematic.
- Put a few components together as a quick test.
- Revise your concept if it doesn't work.
- Install the components on the breadboard, and test.
- Transfer to perforated board, test, and trace faults.
- Add switches, buttons, power jack, and plugs.
- Mount everything in a box (and add labeling).

Experiment 18
Reflex Tester

Because the 555 timer can run at literally millions of cycles per second, you can use it to measure human reflexes. You can compete with friends to see who has the fastest response—and note how your response changes depending on your mood, the time of day, or how much sleep you got last night.

This circuit will only just fit on a breadboard that has 60 rows of holes, but if you proceed carefully, I think the whole project can be assembled in a couple of hours.

You Will Need:

- Breadboard, hookup wire, wire cutters, wire strippers, multimeter.
- 9V power supply (AC adapter preferred).
- 4026B counter chips (3).
- 555 timers (3).
- Resistors: 470 ohms (2), 1K (3), 4.7K (1), 10K (5), 47K (1), 470K (2).
- Capacitors: 0.01µF (2), 0.047µF (1), 0.47µF (2), 10µF (1), 100µF (1).
- Tactile switches (3).
- LEDs (3), two red and one yellow preferred, 3mm preferred.
- Trimmer, 10K (1).
- Single-digit seven-segment numeric LED displays, module size approximately 0.5" x 0.75", low-current red preferred, pin spacing 0.1" (2.54mm). Examples:

  Lite-On `LTS-547AHR`
  Kingbright `SC56-21EWA`
  Broadcom/Avago `HDSP-513E`
  Inolux `INND-TS56RCB`

Caution: Protecting Chips from Static

The 555 timer is not easily damaged, but in this experiment you will also be using a CMOS chip (the 4026B counter) which is more vulnerable to static electricity.

Whether you are likely to zap a chip by handling it depends on factors such as the humidity in your location, the type of shoes you wear, and the type of floor covering in your work area. Some people seem to accumulate a charge of static more easily than others, and I don't have an explanation for this. Personally, I have never damaged a chip with static, but I know people who have.

If static is a risk for you, you'll probably know about it, because you'll suffer from sudden little jolts when you reach for a metal door handle or a steel faucet. If you feel you need to protect chips from this kind of discharge, the most thorough precaution is to ground yourself. The ***wrong way*** to do this is by wrapping the bare end of a piece of wire around your wrist and attaching the free end to any large metal object. This is a bad idea, because if you get an electric shock through your other hand, the shock will ground itself through your body.

The ***right way*** is to spend a small sum on an ***anti-static wrist strap***. It includes a resistor which is sufficient to protect you, but will still ground a static charge. Usually the wire from the strap terminates in an alligator clip. Ideally you would attach that to a steel or copper water pipe, but a steel file cabinet will do.

- Never try to ground yourself through an electrical outlet. This is a dangerous idea.

When you receive chips via mail order, they are usually shipped in channels of conductive plastic, or with their legs embedded in conductive foam. The plastic or the foam protects the chips by insuring that all the pins have an approximately equal electrical potential. If you want to repackage your chips, but you don't want to spend money on conductive foam, you can poke their legs through aluminum foil.

A Quick Demo

The digital displays that I have specified for this project have pinouts that are standardized for that size of numeral. If you use a different display size, the pinouts will be different. If you use a larger size, it won't fit with the other components on the breadboard. The type that you want is shown in Figure 18-1.

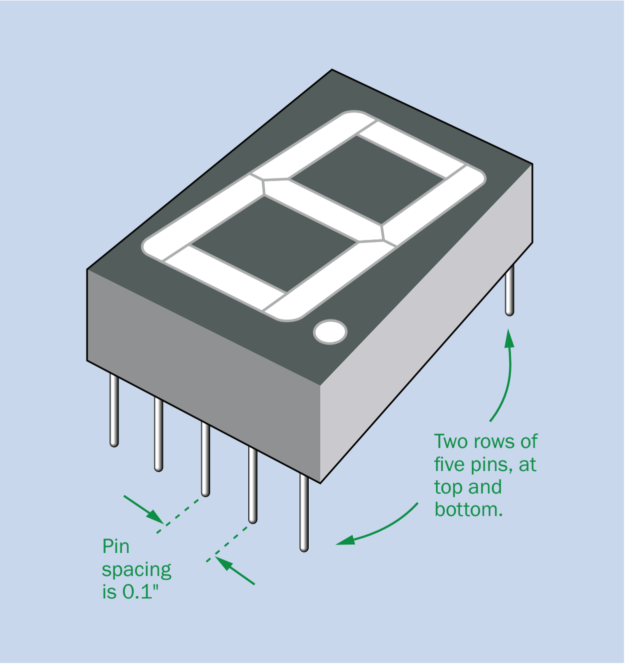
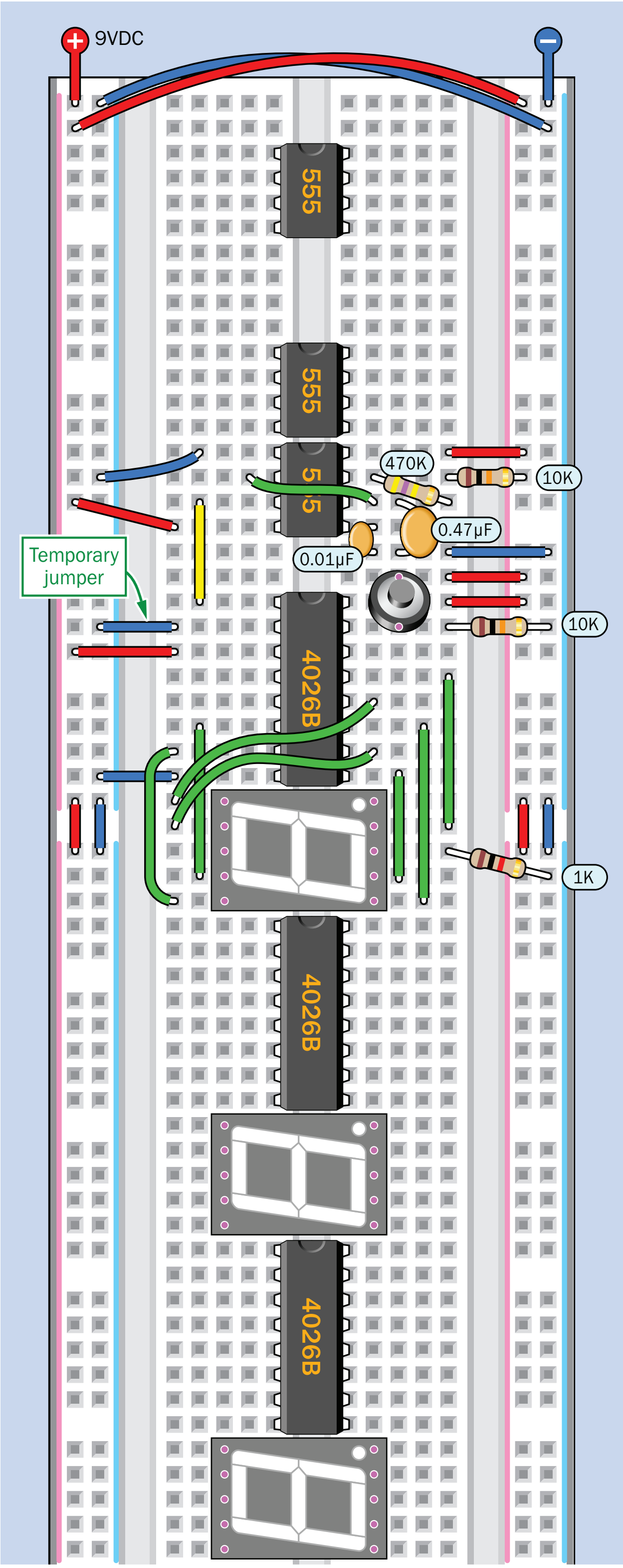


Figure 18-1. The type of 7-segment display used in this experiment.

In some displays the surface around the numeral is black, while in others it is white. This is unimportant from our point of view. When the numeral is illuminated, the segments may be red, amber, green, or blue; in my experience, red provides more light output at the low current which will be used in this experiment.

I suggest you begin with a quick test of a numeral and the 4026B chip that will be driving it. In Figure 18-2 you can see a simple breadboarded circuit that will count repeatedly from 0 to 9, while Figure 18-3 shows the schematic. I have included the extra chips and numerals that you will be using later in this experiment, because I want you to position everything precisely on the board.

Work from the top down, and count the rows of holes carefully! Also make sure that you install the digital displays the right way around. The decimal point in each numeral should be at the top-right corner when the breadboard is viewed in the usual orientation. One blue jumper is labeled as temporary, because it will be



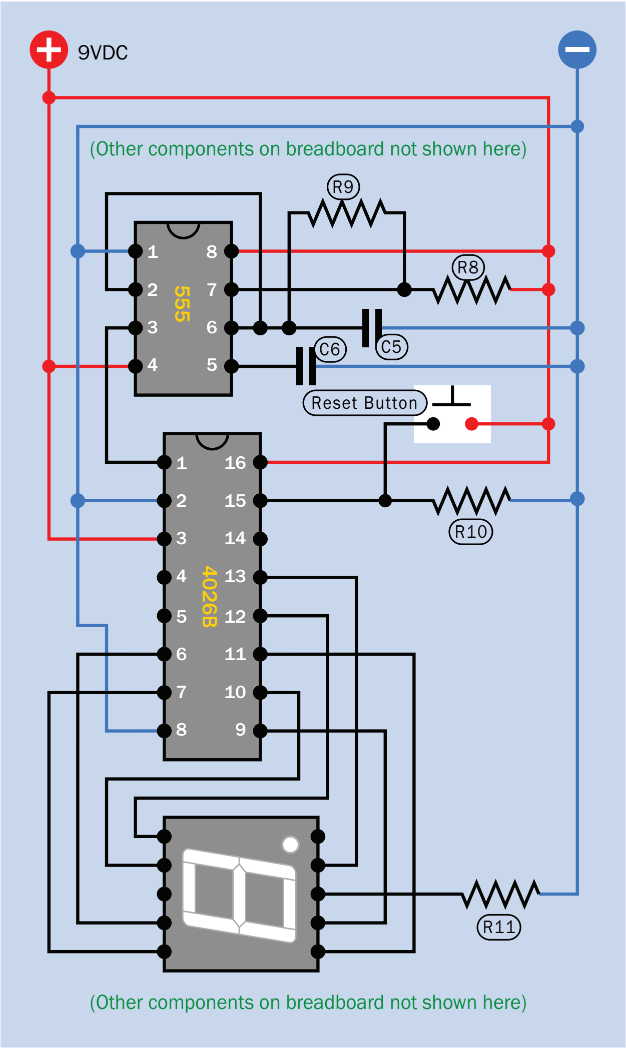


Figure 18-2 (left): A quick demo circuit.

Figure 18-3 (above). Schematic version of Figure 18-2.

removed when the additional timer chips are chained together in the second part of this experiment.

Apply power from a nine-volt battery or AC adapter, and you should see the numeric display counting repeatedly from 0 through 9.

Now let me explain what's happening, using the labels for components in Figure 18-3. These labels may seem to be numbered oddly, but that's because I'll be adding more parts above them in the second half of the experiment.

The 555 timer is wired in astable mode. The timing resistors and capacitor are not arranged exactly as you've seen them before, because I had to save space for the rest of this circuit. Still, the connections are the same as in previous experiments. The combination of R9 (470K) and C5 (0.47μF) will create a pulse rate of about 3Hz, so that you have time to see the numerals change. Naturally you can increase the speed radically by reducing the component values.

You can see that Pin 3, the Output Pin of the timer, is connected directly with Pin 1 of the 4026B counter chip. Does this mean that the timer is telling the counter how fast to count? Yes, precisely!

Hold down the reset button, and the counter resets to zero. Let go of the button, and it starts counting again. Since R10 appears to be a pulldown resistor, and the reset button bypasses it, you can figure that a high input to Pin 15 of the counter forces it to reset, while a low input leaves it undisturbed.

What if your circuit doesn't work? If you don't see any numerals at all, check for voltages around the breadboard. If some segments of the numerals are visible, but they are scrambled, you made some errors in the green wires coming from the counter. It's very easy to misplace some of these wires by one hole in either direction.

If the display shows a 0 that does not change, you wired the 555 timer wrongly, or failed to connect the timer with the 4026B chip correctly.

This circuit is the basis for a reflex tester. All you need to do is connect the additional two digits and a couple more buttons, and increase the counting speed. First, though, I need to tell you more about the components that you are using.

LED Displays

The term "LED" can be confusing. The type of component that you have used in previous experiments is a little rounded blob of plastic with two long leads sticking out of the base. This is properly known as a ***standard LED***, a ***through-hole LED***, or an ***LED indicator***, but they became so common, people started calling them simply "LEDs."

Diodes which emit light are now used in thousands of other components, including the glowing numeral currently plugged into your breadboard. This is properly known as an ***LED display***. More precisely, it is a ***7-segment single-digit LED display***.

Take a look at Figure 18-4. Two pins are labeled "Negative ground," but because they connect with each other internally, only one of them needs to be grounded. The pins either side of them connect with the segments of the numeral lettered from a through g, the decimal point being identified as dp. (Some manufacturers identify it with letter h.)

In this book I chose to use numerals in which the physical package measures about 0.5" x 0.75" (12.7mm x 19mm). These usually have pins at each end, as shown. Smaller numerals usually have pins down each side.

The seven segments light up when they receive positive power. This is called a ***common cathode*** type of LED display, because the negative ends of the internal diodes are all tied together. Remember, the positive end of a diode is the anode, and the negative end is the cathode.

In a ***common anode*** display, the situation is reversed, and you would activate the segments by applying negative power to each one, while they all share a positive internal connection. You use whichever type of display is convenient in a circuit, but common-cathode displays are probably more common, and the 4026B counter chip has positive outputs, so it works with a common-cathode display.

So far so good, but I have left out a crucial piece of information: like all LEDs, the segments in a numeral must be protected with series resistors. This is a hassle, and you may wonder why the manufacturer didn't build resistors into the package. The answer is that the display must be usable with a wide variety of voltages, and the voltage will determine the values of the resistors that you need.

Well—why can't we use just one resistor that all the segments share? Yes, we can do that, and in fact I did do that in your demo circuit. It's identified as R11 in Figure 18-3, inserted between the negative-ground pin of the numeral and the negative bus on the breadboard. However, that one resistor drops the voltage and restricts the current for sometimes 2 segments, and sometimes 5 segments, and sometimes all 7 segments, depending which numeral is being displayed. Consequently, some numerals will be brighter than others.

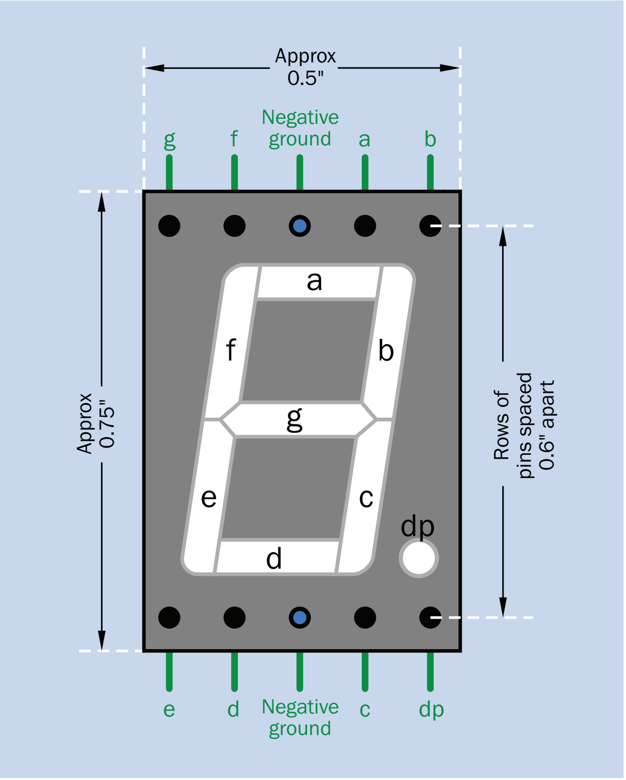


Figure 18-4. Dimensions and pinouts on the 7-segment numeric LED display chosen for this project.

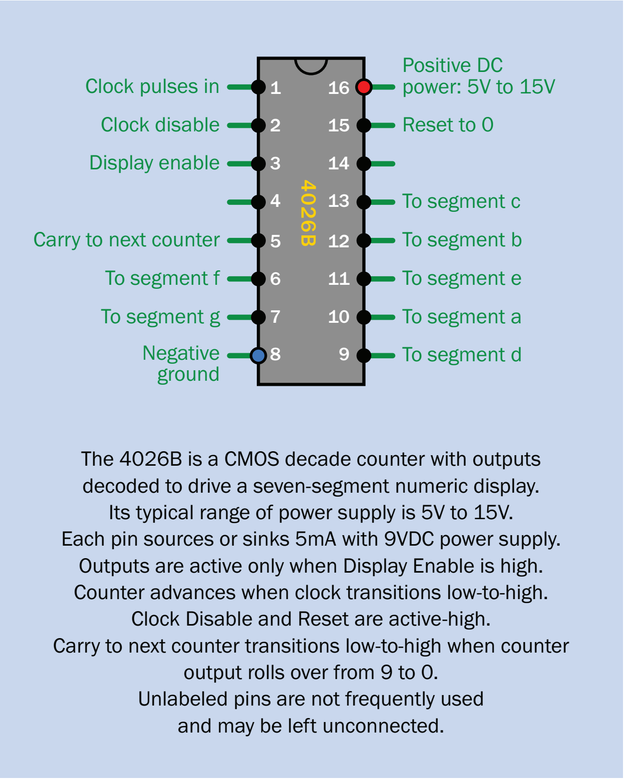


Figure 18-5. Pinouts of the 4026B counter chip.

Does this really matter? For this demo, I decided that simplicity was more important than perfection, so I used a single resistor. This is not the correct procedure, but you will be installing three of the seven-segment displays in this project, and I think you'll be happy to use only three series resistors instead of 21.

You'll notice in Figure 18-2 that the series resistor beside the numeral has a value of 1K. This will cause the numeral to look a bit dim, but it will still be usable, and I don't want to overload the counter output. If you have a high-efficiency red LED display, it may appear brighter.

The Counter

The 4026B chip is known as a ***decade counter,*** because it counts in tens. Most counters have a ***coded output***, meaning that they output the numbers in binary-coded format (which I will discuss in a later project). This counter doesn't do things that way. It has seven output pins, and it powers them in patterns that just happen to be correct for a seven-segment display. While other counters require a ***driver*** to convert a binary output to the seven-segment patterns, the 4026B gives you everything in one chip.

Some manufacturers claim that the chip can operate on power supplies ranging from 3V to 18V, but the old original Texas Instruments data sheet shows "recommended operating conditions" of 5V to 15V, which I think is more realistic.

Unfortunately the 4026B cannot deliver much output current. The datasheet advises you to draw only around 1mA from each output pin when using a 9VDC supply, but in my experience 5mA is acceptable if the counter is not running at a high frequency. The internal counting process actually consumes a lot of energy at high frequencies, and generates heat. I'll be using it at about 1kHz, which is not "high" in the world of digital chips. The counter is quite capable of running 1,000 times as fast.

Ideally you should amplify the outputs from the counter, and in fact you can buy a chip containing seven transistor pairs for exactly this purpose. It's called a ***Darlington array***. (What if you want to display the decimal point, as well as the seven segments? No problem. You can buy a different Darlington array containing eight transistor pairs.)

If I had used three Darlington-array chips to drive the three LED displays in this project, they could have been really bright—but that would have added to the complexity and expense, and I would have needed two breadboards. (Also, because of the way Darlington arrays are wired internally, I would have had to use common-anode digital displays instead of common-cathode.) Bearing all this in mind, I decided not to use Darlington arrays.

Now I'll give you some details about the internal workings of the 4026B. Take a look back at Figure 18-5, which shows the pinouts of the chip. The pins with labels such as "To segment a" are easy to understand. You simply run a wire from that pin to the appropriate pin on your LED display.

Pins 8 and 16 of the chip are for negative-ground and positive power, respectively. The chip won't work without these connections. Almost all digital chips require power to be applied to opposite corners (with the exception of the 555 timer—although, really, it is classified as an analog chip).

Pulldown resistors and pullup resistors are even more important for CMOS digital chips than they were for the 555 timers that you used. A digital chip really needs to know if an input pin is high or low, and will give you very perplexing results if you break this rule.

- In the case of output pins that are not being used, they may be left unconnected.

Sometimes a chip has an input that you won't need at all. The 4026B, for instance, tells me that Pin 3 is a "display enable" input. I want the display to be enabled all the time, so in my circuit I connected Pin 3 directly to the positive bus on a set-it-and-forget-it basis.

- If you won't be using an input pin, the pin must still have a defined state. You can wire the pin directly to the positive or negative side of the power supply.

Now I'll run through the other features of the 4026B.

The Clock Input (Pin 1) accepts a stream of high and low pulses. The chip doesn't care how long the pulses are. It just responds by adding 1 to its count, each time it senses the input voltage rising from low to high. It counts on the ***rising edge*** of each pulse.

The Clock Disable (pin 2) tells the counter to ignore the clock input. Like all the other pins on the chip, this one is ***active-high***, meaning it performs its function when it rises to a high state. On your breadboard, I ran in a temporary blue wire to hold pin 2 low. In other words, I disabled the Clock Disable pin. This is confusing, so I will summarize the situation:

- When the Clock Disable pin is in a high state, it tells the counter to ignore the stream of pulses on Pin 1.
- When the Clock Display pin is pulled down to negative ground, it allows the counter to count the incoming pulses.

Pin 4 is described by the manufacturer as **Display Enable Out**, which is of no interest in this book, so I didn't bother to label it. You can leave it floating, because it is an unused output pin.

The **Carry Out** (pin 5) is essential if you want to count higher than 9. This pin state changes from low to high when the counter tries to count beyond 9 and cycles back to 0. If you take this output and connect it with the Clock Input pin of a second 4026B timer, the second timer will count in tens. You can then use its carry output pin to signal a third timer, which will count in hundreds. I will be using this feature.

Pin 14 can be used to restart the counter after it counts through 0, 1, and 2. This is useful for the first digit in a 24-hour clock, but not relevant to us here. It is an output pin that we will not use, so I left it unconnected.

Perhaps all the features seem confusing, but if you ever find yourself confronted with a counter chip that you've never seen before, you can figure it out (if you are patient and methodical) by looking up the manufacturer's datasheet.

Then you can test it with LEDs and tactile switches, to make sure there are no misunderstandings. In fact, long ago, this was how I got acquainted with the 4026B myself.

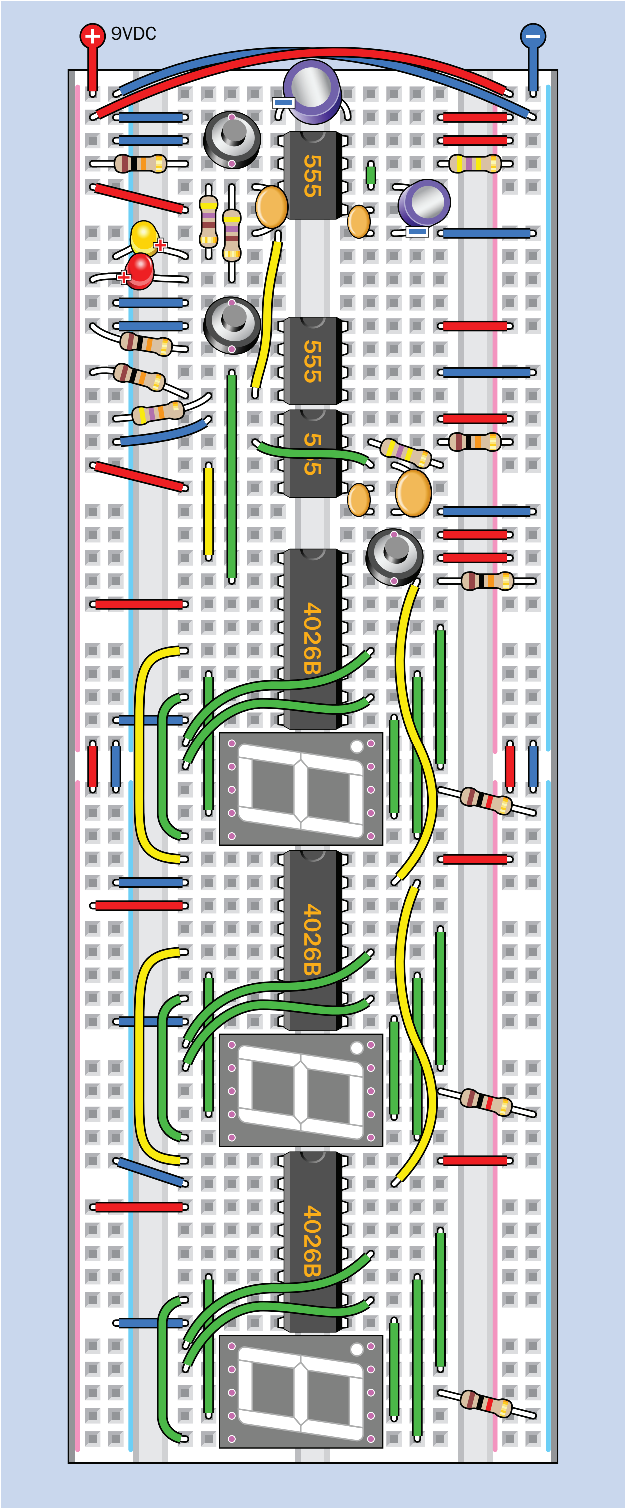


Figure 18-6. Complete circuit for the Reflex Tester.

Time for a Plan

How should the reflex timer work? Here's my wish list:

1. I need a start button.

2. After the start button is pressed, there is a delay in which nothing happens. Suddenly there is a visual prompt, challenging the player to respond.

3. Meanwhile, the count starts upward from 000, in milliseconds. (There are 1,000 milliseconds in a second.)

4. The player has to press a button to stop the counting process.

5. The count freezes, showing how much time elapsed between the prompt and the stop time. This measures the user's reflexes. Simple!

6. A reset button sets the count back to 000.

The Complete Circuit

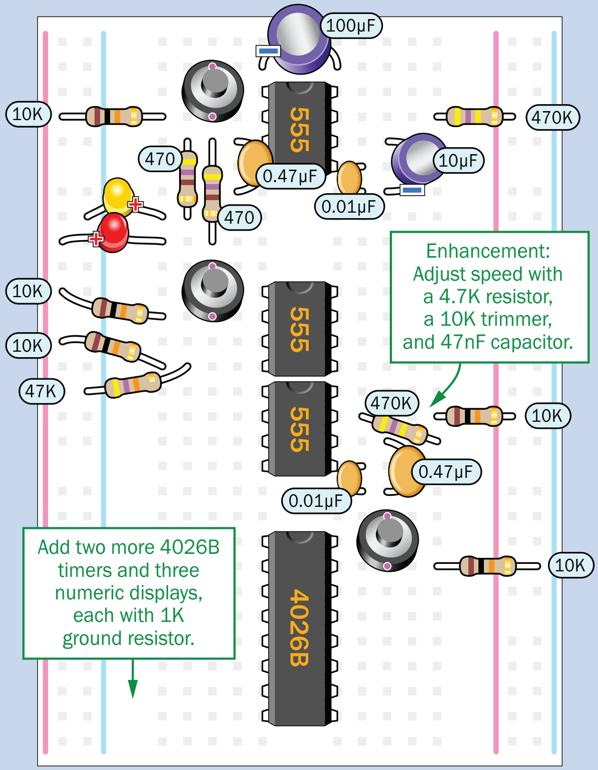
Figure 18-6 shows the complete circuit that satisfies my requirements for a reflex timer. It uses 555 timers in much the same way as in the intrusion alarm project, although here they have to work compatibly with the counter chips, which adds to the complexity. Before you build this circuit, remember to remove the temporary blue jumper that was identified in Figure 18-2.

The component values are shown in Figure 18-7. You have to add two more counters which are not shown in this figure, plus the three numerals and a 1K series resistor for each. I didn't bother to show them, because you can see where they are in Figure 18-6.

After you test the circuit, I'm going to show you how to crank up the speed and then calibrate it to give a fairly accurate measurement of time. This will require three extra components that are listed in Figure 18-7 under the heading "Enhancements." All the components that you will need are summarized at the bottom of Figure 18-7.

I'll describe how the circuit works using the labels for components in Figure 18-8.

When you apply power—assuming you haven't made any wiring errors—the counter immediately starts counting without being asked to do so. This is annoying, but easily dealt with. Press the Stop Button to stop the count. Press



Components

Resistors	Capacitors	
470 (2)	0.01µF (2)	LEDs 3mm (2)
1K (3)	0.047µF (2)	Tactile switch (3)
4.7K (1)	0.47µF (2)	555 timers (3)
10K (5)	10µF (1)	4026B chips (3)
47K (1)	100µF (1)	LED numerals (3)
470K (2)		Trimmer 10K (1)

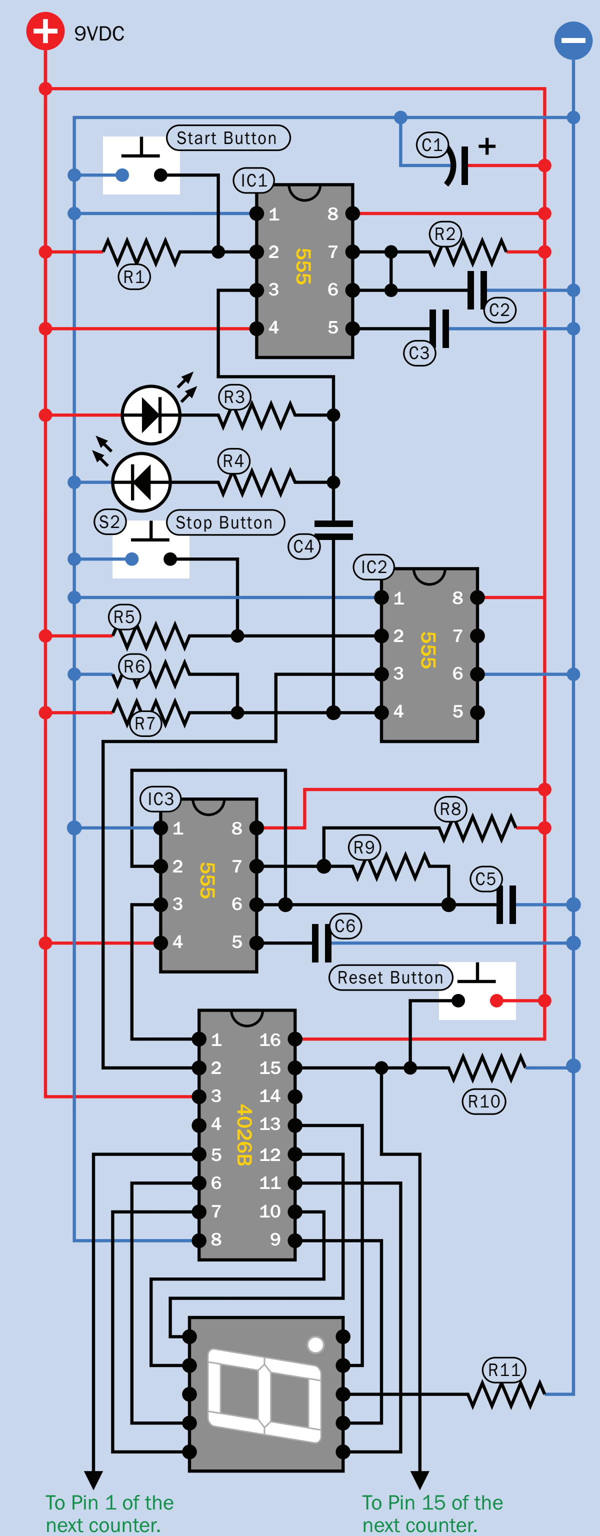


Figure 18-7 (above). Components-only view of the reflex tester.

Figure 18-8 (right). Complete schematic for the reflex tester.

the Reset Button to reset the count to zero. Now you're ready for action.

Press the Start button, which creates an initial delay. During this delay, the first LED indicator lights up. The delay lasts for about seven seconds, at which point the LED goes out and the second LED comes on, which is your signal to press the Stop Button as quickly as you can. The count freezes to show you how much time you took to respond. Now you can press the Reset Button and try again.

Because IC3 is running in slow motion, you won't have much trouble stopping the counter before it counts very high. I suggest you should let it run for a bit, to check that all three of the numbers display correctly. Also, you need to verify that the second display will advance from 0 to 1 when the first display reaches 9 and begins again at 0, and you need to verify that the third display will do the same thing.

Now, how does it work?

In Figure 18-9 you can see how the counters, buttons, and timers interact with each other. The best way to understand this diagram is to start at the bottom and work up.

IC3, at the bottom, runs constantly in astable mode. It never stops, and its output is permanently connected with Pin 1 of the first counter.

The Reset button is very simple, and you already used it. This button simply tells the counter to reset its display to 0.

In Figure 18-6 you can see that Pin 15 of the first counter is connected by a long yellow jumper to Pin 15 of the second counter, and another yellow jumper links it with the third counter, so that when you reset the first counter, you reset all three.

The counting process is controlled by IC2, which is a 555 timer wired in bistable mode. Its output goes to Pin 2 of the first counter. Remember, Pin 2 on the counter is the Clock Disable Pin. A high voltage tells it to ignore the pulses coming in on Pin 1. A low voltage allows it to count them.

When you press the Stop button, it delivers a low pulse to the Trigger Pin of IC2, flipping its output into a high state and stopping the counter from responding to the incoming pulses. Then when IC2 receives a low pulse to Pin 4 (its Reset Pin), this flips its output to a low state—which tells the counter to resume counting pulses.

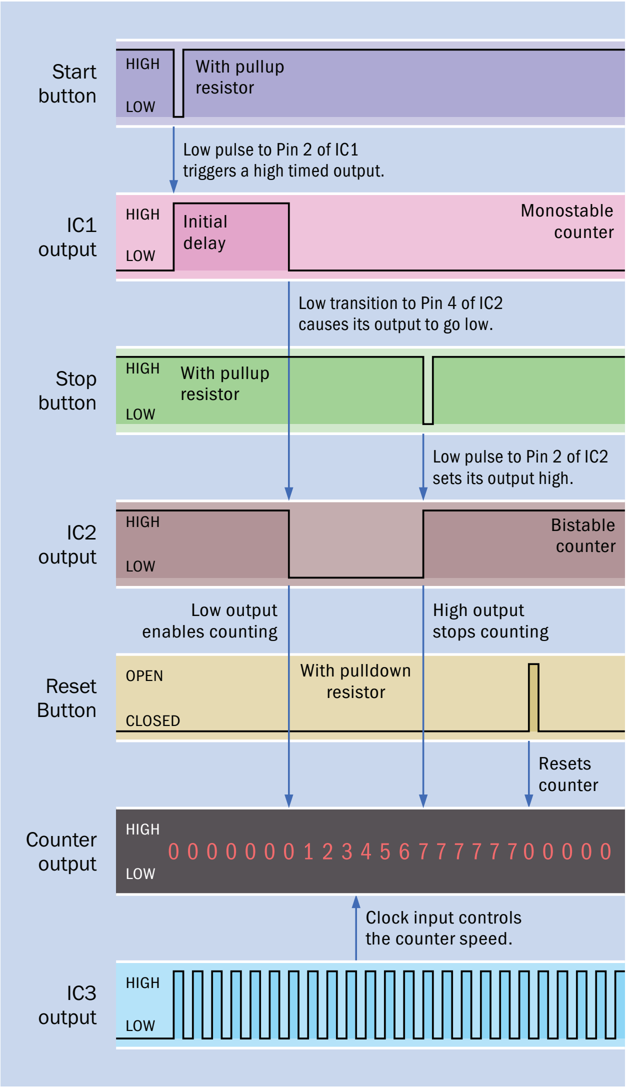


Figure 18-9. The interactions between counters, buttons, and timers in the Reflex Tester circuit.

How does IC2 receive its pulse to Pin 4? From IC1, the timer which creates the initial waiting period. At the end of that period, the output drops, and this transition passes through a coupling capacitor to Pin 4 of IC2.

The whole sequence of events begins when you press the Start button, which triggers the high timed output from IC1. The end of that output sends a negative pulse to IC2, making its output go low. The low output allows the counter to start counting.

The difficulty in designing this circuit was in making the 555 timers get along with the 4026B counters. All the chips had to be happy, but they don't speak the same language. In the world of 555 timers, a low input on Pin 2 triggers a high output on Pin 3, which is often used to tell some other component to start doing something. But a 4026B counter thinks that a high input (on its Pin 2) means it should ***stop*** counting. Consequently I had to ask IC2 if it would mind having a high output for most of the time, and dropping it to tell the counter to start counting.

This required me to trigger IC2 by sending a pulse to its Reset Pin, which turned out to be problematic. The Trigger Pin of a 555 timer is very predictable: If its state goes below one-third of the supply voltage, it triggers the timer. The Reset Pin actually needs a lower voltage than that, to stop the timer. How much lower? This seems to depend on the manufacturer. The exact voltage isn't mentioned in any of the datasheets that I have seen.

In the schematic in Figure 18-8, you can see two resistors, R6 and R7, both connected to the Reset Pin of IC2. R6 is 10K, while R7 is 47K. These resistors function as a voltage divider, pulling the voltage on Pin 4 down a bit, so that when a pulse comes through to it via C4, the voltage on Pin 4 will go right down to zero. Anything greater than zero may not be low enough to satisfy IC2.

If you have any problem getting IC2 to start the counter, substitute a slightly lower resistor for R7.

I specified 3mm LED indicators for this circuit because there isn't a lot of room for them. You can use 5mm if you can make them fit. I also suggest different colors, since one LED tells you to "Get ready" and the other tells you to "Go."

In Figure 18-8 you can see that the two LEDs are pointing in opposite directions, so that the first one sinks current into IC1 when its output is low, and the second one lights up when the output from IC1 is high. I haven't mentioned before that you can sink current into Pin 3 of a 555 timer, but in fact it has no problem sinking current when it is in a low state.

The only other feature that I need to explain is C1, the 100µF capacitor right at the top of the circuit. In Figure 18-6 it looks like an afterthought, but in fact it's very important. It suppresses the voltage spikes which the 555 timers create when they switch their outputs on. In that case—shouldn't it be closer to the timers? No, those nasty spikes travel through the whole circuit, and C1 seems to deal with them in the location as shown.

That concludes my explanation. Now for the important part: Making it run faster, and calibrating it.

Calibration

Recall that the counting speed is controlled by IC3, the timer that is lowest of the three on the breadboard. R8, R9, and C5 control its frequency.

R8 is 10K, and as you know, I prefer not to use a lower value than that. R9 is 470K, and its value can go much lower. C5 is 0.47µF, and that value can go lower, too.

Begin by substituting a 47nF capacitor for C5. That will multiply the running speed by a factor of 10. Then if you use a value of about 10K for R9, the table in Figure 16-5 tells me that you should get a frequency of about 1kHz, which is what you want, bearing in mind that 1kHz means that the timer will be delivering 1,000 pulses per second.

It won't be precise, though. You'll need to fine-tune it, and that will require a trimmer.

Fortunately I left a little space on the breadboard that's just big enough for a 10K trimmer. In Figure 18-10, you can see how it fits there.

Before installing the trimmer, remove R9, the 470K resistor which you have been using. Replace it with a 4.7K resistor, as shown in Figure 18-10. I think you'll find that the leads on a resistor are long enough to bridge that gap.

Now if you turn the 10K trimmer to its mid-point, it will give you a resistance of about 5K, and if you add that to the 4.7K resistor, you have a total that is near to 10K.

Don't forget that you also have to substitute a 47nF capacitor for C5.

Now when you activate the circuit, the top numeric display is counting thousandths of a second, the middle dis-

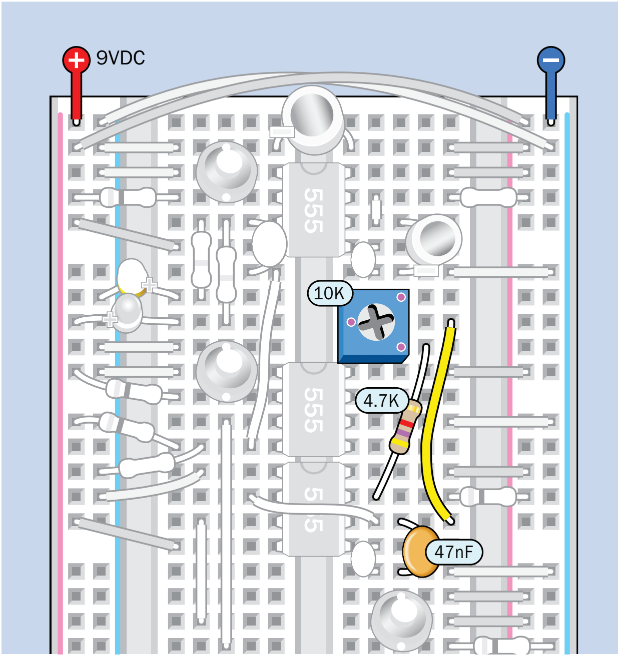


Figure 18-10. Adding a trimmer to adjust the counting speed.

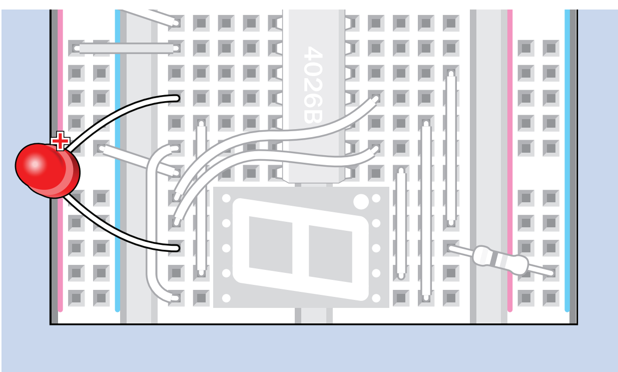


Figure 18-11. Inserting an LED to make calibration easier.

play is counting hundredths, and the bottom display is counting tenths—approximately.

"Calibration" means adjusting the circuit so that its speed matches a source that you can trust. Fortunately there are many sources of accurate time, such as the stopwatch feature on your phone. But how are you going to match that with the flickering numerals on the bottom display?

I will suggest a way.

On the bottom counter, Pin 5 is the "Carry Out" pin. It goes high whenever the counter reaches 9 and resets to 0. This means that while the counter is counting in tenths of a second, Pin 5 only changes once each second. Ah-hah!

Figure 18-11 shows how you can insert an LED between Pin 5 on the bottom counter, and the center pin of the bottom numeral. Remember that the center pin on the left connects through to the center pin on the other side of the numeral, which is grounded through a 1K resistor. So if you insert an LED as shown in Figure 18-11, it won't overload the counter, and it will flash every second. (Actually, it flickers a bit as the load on the counter changes during the counting cycle, but it will be clear enough for you to see the flashes.)

Now you just need to put your trusted time source right alongside the flashing LED. Start the Reflex Tester, and let it run. Adjust the trimmer, and you should be able to synchronize the LED with the time source if you are patient and you watch very carefully.

Human Reflexes

When you measure your reflexes, perhaps they seem a bit—slow?

In fact, human reflexes are slow, especially when compared with electronic components. A typical reaction time to a visual stimulus is 250 milliseconds, which of course is a quarter of a second. How can human beings do things such as driving race cars or flying aircraft, when they take so much time to react to an event? I don't know, but that's the way it is.

After you calibrate your reaction-timer circuit, if you get a result of 200 or less for your reaction time, you're doing well.

Reflexes are affected by numerous factors, including consumption of prescription drugs and alcohol. One thing I must ask you is please, don't use the Reflex Tester to find out if you are sober enough to drive. Ability to drive safely depends on judgment as well as reflexes, and is always impaired by alcohol. The Reflex Tester is for entertainment purposes only.

Experiment 19
Learning Logic

A counter such as the 4026B is technically a ***logic chip***. It contains ***logic gates*** that enable it to count. Every digital computer uses logic gates for its fundamental processes.

Because logic is so fundamental, I'm going to delve into it in detail. The magic words AND, OR, NAND, NOR, XOR, and XNOR will open a whole new world of digital intrigue.

When you deal with logic gates individually, they're extremely easy to understand. When you chain them together, they can be perplexing. With this in mind, I'll begin by using them one at a time. If they seem too simple, be patient: I guarantee that they will become more complicated.

This chapter contains a lot of explanations and factual summaries. I don't expect you to absorb all the details; the idea is for you to return here whenever you need to refresh your memory. You will be able to build the projects in experiments 20 through 23 without fully understanding logic gates, but when you want to know how they work, use this chapter for reference.

You Will Need:

- Breadboard, hookup wire, wire cutters, wire strippers, test leads, multimeter.
- 9VDC power supply (battery or AC adapter).
- Tactile switches (2).
- 74HC32 quad 2-input OR chip (1).
- 74HC08 quad 2-input AND chip (1).
- Generic red LED (1).
- LM7805 voltage regulator (1).
- Resistors: 1K (1), 10K (2).
- Capacitors: 0.1µF (1), 0.47µF (1).

The Regulator

Digital chips whose part numbers begin with 74 are more demanding than the 555 timer or the 4026B counter that you used so far. Most require a precise 5VDC, with no fluctuations or "spikes" in the flow of current.

Fortunately, this is easy and inexpensive to achieve: Just set up your breadboard with an LM7805 voltage regulator. When you supply it with a voltage input ranging between 7.5VDC and 12VDC, it gives you a 5VDC output on a totally reliable basis.

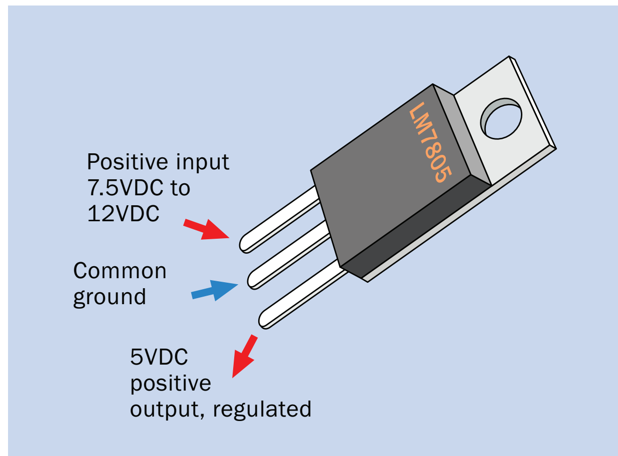


Figure 19-1. The LM7805 voltage regulator.

The regulator is shown in Figure 19-1. It looks quite different from other components that you have dealt with so far, because it is designed to cope with a current of up to 1.5A. The aluminum back of the regulator absorbs heat, and the hole at the top allows you to bolt it to a larger heat sink. For our purposes this is unimportant, because the logic circuits for which I will be using the regulator will take very little current. You can install it simply by pushing it into your breadboard. The flat pins are a tight fit, but with a bit of persuasion, they will go in.

The pictorial representation that I will be using for the regulator in breadboard diagrams is shown in Figure 19-2, above a schematic symbol. In schematics that you are likely to see elsewhere, the component is usually represented just as a rectangle with the part number in it.

I will be continuing to use a 9VDC power source, so that you don't have to buy a separate 5VDC power supply. Therefore you should place the voltage regulator at the top of your breadboard, along with two smoothing capacitors which are mandatory. The output from the regulator will then go to the positive buses on the breadboard, as shown in Figure 19-3. A schematic illustrating this arrangement is in Figure 19-4.

- Note that the negative side of the 9V power supply and the negative pin of the voltage regulator both share the same negative bus.

Be careful not to supply 9V to any of the 74xx series of logic chips that you'll be using. They might not survive this experience. In my breadboard diagrams, I will show the voltage regulator wherever it's required. In schematics, I won't include the regulator but I will label the bus with the voltage it is supposed to carry. Also I will show red wires with black dots on them to remind you that they are delivering 5V, not 9V.

- In schematics, a red wire with black dots is delivering 5VDC.

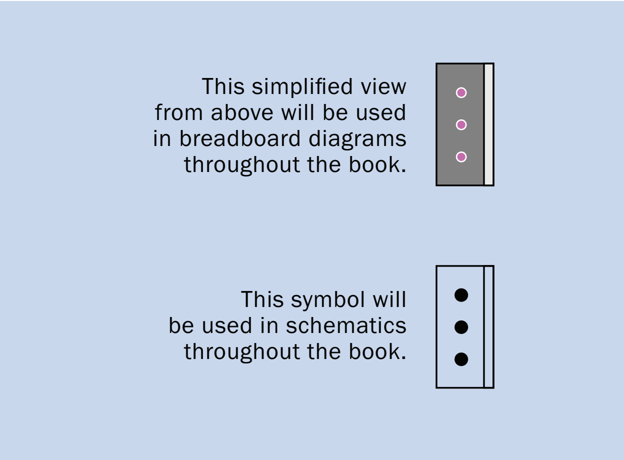


Figure 19-2. How the LM7805 will be represented in this book.

Caution: Inappropriate Inputs

DC, not AC. Remember that the LM7805 is a DC-to-DC converter. Do not confuse it with an AC adapter, which uses alternating current from an outlet in your home. Do not apply AC to the input of your voltage regulator.

Maximum Current. The LM7805 does a wonderful job of maintaining its output at an almost constant voltage, regardless of how much current you draw through it—so long as you stay within its rated range. I suggest 1A as a practical maximum without a heat sink.

Maximum Voltage. Although the voltage regulator is a solid-state device, it behaves a little like a resistor in that it radiates heat in the process of reducing voltage. The higher the voltage you apply to a regulator, and the more current passes through it, the more heat it must get rid of. This is why I suggest a maximum input of 12VDC.

Minimum Voltage. Like all semiconductor devices, the regulator delivers an output voltage that is lower than its input voltage. This is why I suggest a minimum input voltage of 7.5VDC.

After you install your voltage regulator, set your meter to measure DC volts and measure the voltage between the positive and negative buses on the breadboard, just to make sure. Note that the two smoothing capacitors are important to suppress oscillations from the LM805.

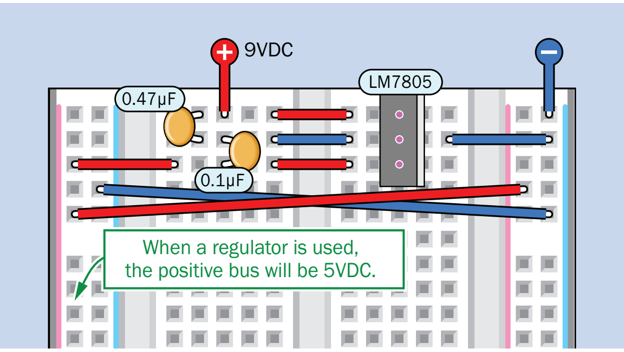


Figure 19-3. How to place a voltage regulator at the top of a breadboard. Don't omit the smoothing capacitors.

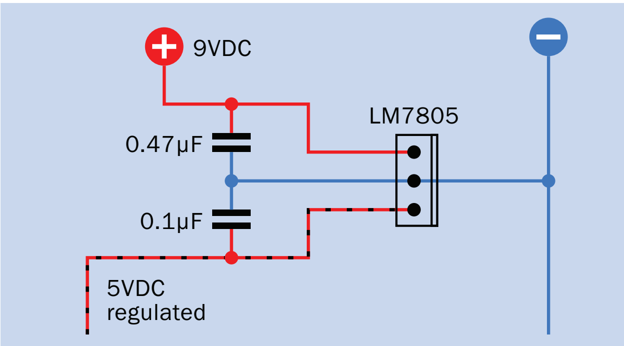


Figure 19-4. The schematic version of the breadboard in Figure 19-3. A red wire with black dots means, "5V power supply."

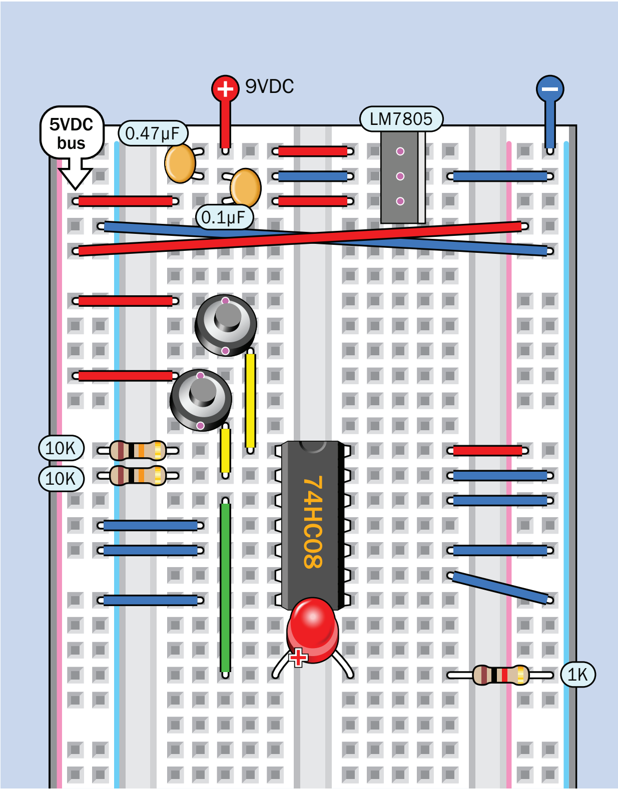


Figure 19-5. Your first logic-chip demo.

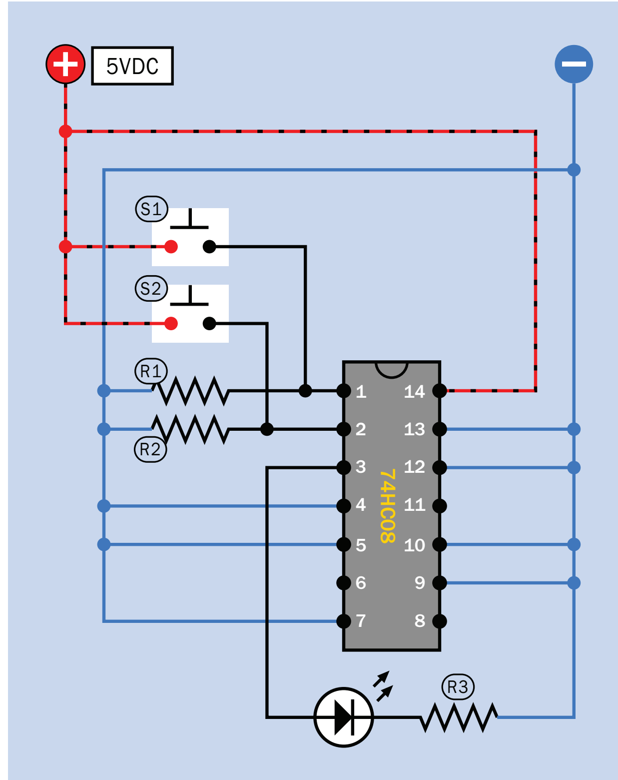


Figure 19-6. Schematic version of Figure 19-5.

Your First Logic Gate

Now that you have your 5VDC breadboard ready, take a couple of tactile switches, two 10K resistors, an LED, and a 1K resistor, and set them around a 74HC08 logic chip as shown in Figure 19-5.

Because logic chips have a limited ability to deliver current, I'm asking you to use 1K as the value for a series resistor with an LED, even though the supply voltage is now only 5VDC. The LED will be dim, but you should have no trouble seeing it. In Figure 19-6 the same simple circuit is shown in schematic form.

You can see that many of the pins of the chip are connected directly to the negative side of the power supply. This is because they are unused input pins, for reasons which I will explain in a moment.

When you connect power, nothing will happen. Press one of the tactile switches, and—nothing happens. Press the other switch, and still nothing happens. Now press both switches simultaneously, and the LED should glow.

Pins 1 and 2 of the 74HC08 are ***logic inputs***. The 10K pull-down resistors, labeled R1 and R2 in the schematic, keep the logic pins at a low state until the pushbuttons bring them up to a high state, in just the same way that you are familiar with from 555 timer. In this case, though, the terminology is a little different:

- When an input or output associated with a 5V logic chip is near 0VDC, we say it is ***logic-low***. This means "below 1V," in the HC generation of 74xx chips.
- When an input or output associated with a 5V logic chip is near 5VDC, we say it is ***logic-high***. This means "above 3.5V," in the HC generation of 74xx chips.

The two logic inputs go to a ***logic gate*** inside the 74HC08. A gate is just a bunch of transistors that process high or low inputs and deliver an output. This particular gate is

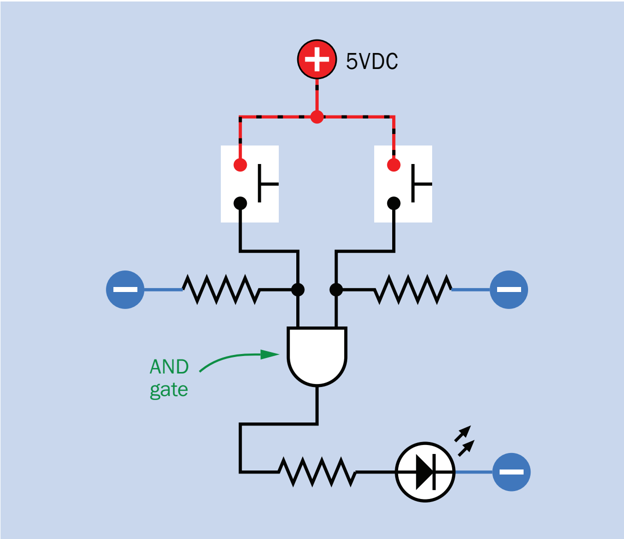


Figure 19-7. A logic diagram featuring an AND gate.

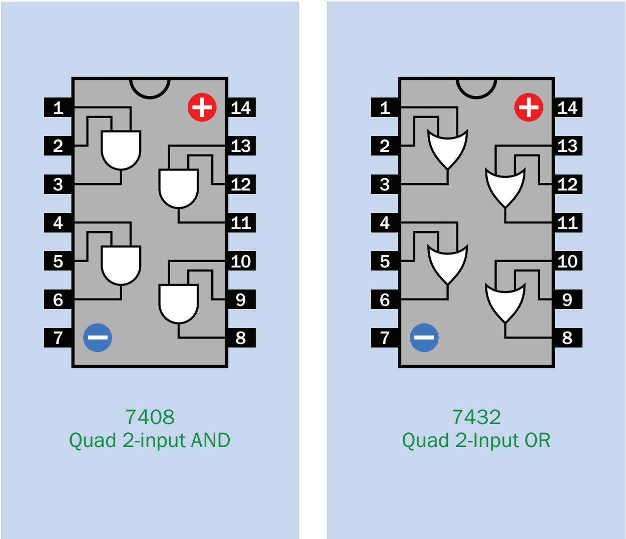


Figure 19-8. Pin functions for AND chip (left) and OR chip (right).

an AND gate, which only delivers a high output through Pin 3 when Pin 1 AND Pin 2 are high.

Remove the 74HC08 from the board, taking care not to bend any of its pins. Sliding a thin screwdriver under it in the channel at the center of the breadboard is a good way to lever it out. Substitute a 74HC32 OR chip, in exactly the same position where the 74HC08 was located, and with all the other components and jumpers unchanged. Now you'll find that if Pin 1 is high, OR Pin 2 is high, OR both of them are high, the output is high. The 74HC32 contains an OR gate, as you might have guessed.

This may seem elementary, but all digital computing operations are performed with logic gates, and only seven types of gates exist. You just observed two of them in action. While you can of course use plain, old-fashioned switches to achieve the same results as in this experiment, you'll see in the next experiments that logic gates can do much more.

Logic Symbols

Logic gates can be represented with special symbols that are used in a kind of schematic known as a *logic diagram.* The logic diagram for the circuit that you just built, using the AND gate, is shown in Figure 19-7.

No power supply is shown for the gate in a logic diagram, but actually the chip containing the AND gates has to receive power at pins 7 (negative ground) and 14 (positive). This is how each logic output manages to deliver more current than its inputs take in.

- Anytime you see a symbol for a logic gate, remember that it needs power to function.

The 74HC08 chip contains four separate AND gates, each of which has two logical inputs and one output. It is referred to as a "quad 2-input AND chip."

The AND gates are connected with the pins of the chip as shown in Figure 19-8, in the diagram on the left. Because only one gate was needed for the simple test that you just performed, the input pins of the unused gates were shorted to the negative side of the power supply, to stop them from floating. The inputs of this type of chip are so sensitive, they can respond to stray electromagnetic fields, and must always be grounded when they are not being used.

Most (not all) two-input logic chips in the 74xx series have interchangeable pinouts, as you just saw.

You're probably wondering how something so simple can be useful. Well, soon you'll see that logic gates can create an electronic combination lock, or a pair of electronic dice, or a computerized version of a TV quiz show where users compete to be the first to press a button. If you were insanely ambitious, you could build an entire computer out of 74xx chips. A hobbyist named Bill Buzbee actually did that, as shown in Figure 19-9.

Alternatively, if you were sufficiently motivated, maybe you'd like to build an entire two-player game simulating the battle of Antietam during the American Civil War. One of my readers named Jeff Palenik did that, and sent me photographs to prove it. He used nineteen breadboards with various logic chips and timers, and quite a lot of hookup wire, as you can see in figures 19-10 and 19-11. I'm guessing the circuit may require more than a 9V battery.

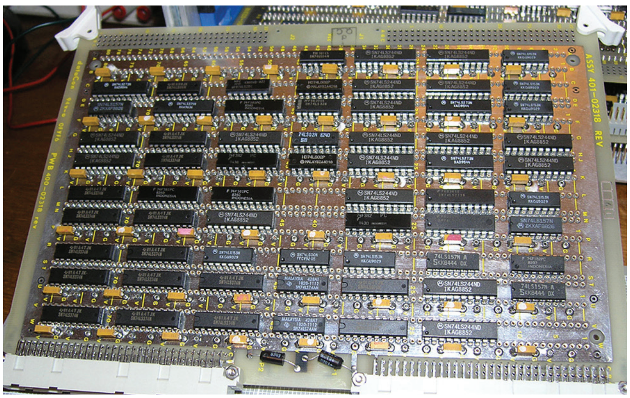


Figure 19-9. A computer motherboard hand-built from 74xx chips. It functions as the heart of a web server.

Figure 19-10. Jeff Palenik's Civil War simulation. Another control panel is at the opposite side, for the opposing player.

Logical Origins

The concept of logic gates grew out of some theoretical work by George Boole, a British mathematician born in 1815. He did something that few people are ever lucky enough or smart enough to do: He invented an entirely new branch of mathematics.

Interestingly, it was not based on numbers. Boole had a relentlessly logical mind, and he wanted to reduce the world to a series of true-or-false statements which could overlap in interesting ways.

Venn diagrams, conceived around 1880 by a man named John Venn, can be used to illustrate some logical relationships of the type that Boole described. Figure 19-12 shows a simple Venn diagram defining the properties of creatures in the world. In Figure 19-13 I have shown this concept in a different way, using what is sometimes called a ***truth table*** where red means "true" and blue means "not true." The table asks two questions: Does a creature live on the land, and does it live in the water? If the answer to both questions is "true," then it's an amphibian (which was the overlap of the two circles in the second Venn diagram). I'm referring to that as A AND B.

In the experiment that you just performed with an AND gate, suppose you labeled the switches A and B. The output from the AND gate would be exactly the same as in this truth table.

These simple ideas have far-reaching implications, but first I have to finish the story about George Boole. His

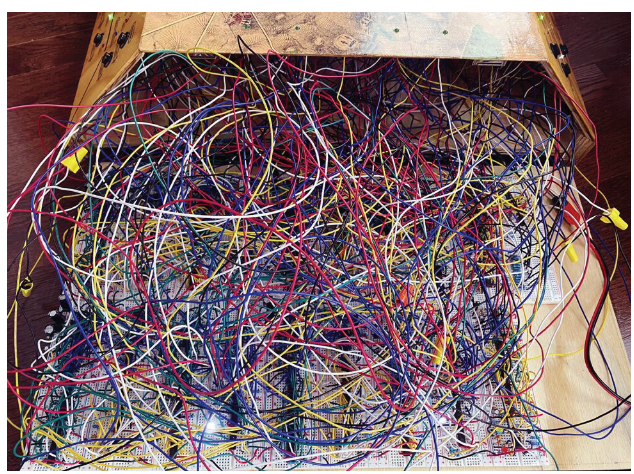


Figure 19-11. Inside the simulation game. A few breadboards are visible at the bottom of the picture.

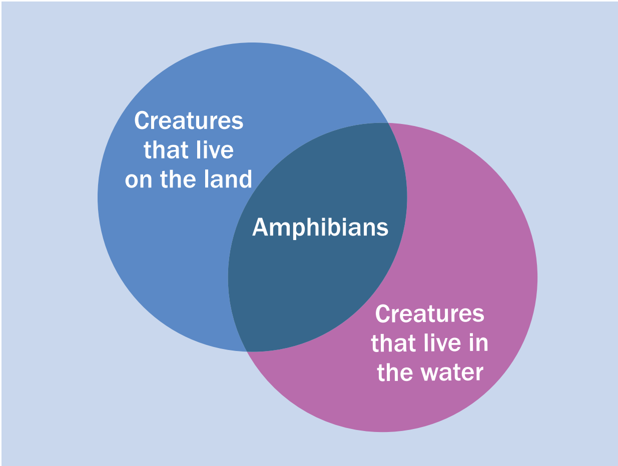


Figure 19-12. Two classes of creatures that overlap.

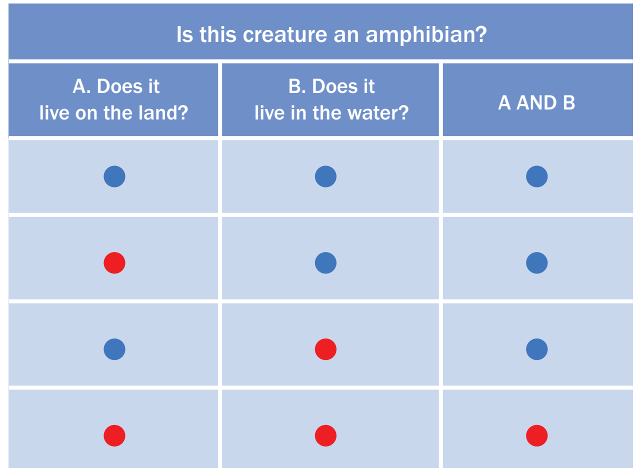


Figure 19-13. A truth table derives from Figure 19-12.

treatise on logic was published in 1854, long before transistors or even vacuum tubes were invented. In fact, during his lifetime, his work seemed to have no practical applications at all. But a man named Claude Shannon discovered the work while studying at MIT in the 1930s, and in 1938 he published a paper describing how "Boolean analysis" could be applied to circuits using relays. This had immediate practical applications, as relays were used in telephone networks, which were growing rapidly.

In those days, two customers living in separate homes in a rural area might be asked to share one telephone line. If person A wanted to use the line, or if person B wanted to use it, or neither of them wanted to use it, this was okay. But if A AND B both wanted to use it, that was a problem. Once again you can see the same logical pattern, and when engineers had to design networks that would deal with thousands of connection issues, the patterns became important.

After Shannon's application of Boolean logic to the telephone system, the next step was to see that if you used an "on" condition to represent numeral 1 and an "off" condition to represent numeral 0, you could build a system that could count. And if it could count, it could do arithmetic. The truth table in Figure 19-14 shows how an AND operation can be used when dealing with a very simple addition sum.

When vacuum tubes were substituted for relays, the first practical digital computers were built. Then transistors took the place of vacuum tubes, and then integrated circuit chips replaced transistors, leading to the desktop computers that we now take for granted. But deep down, at the lowest levels of these incredibly complex devices, they still use Boolean logic.

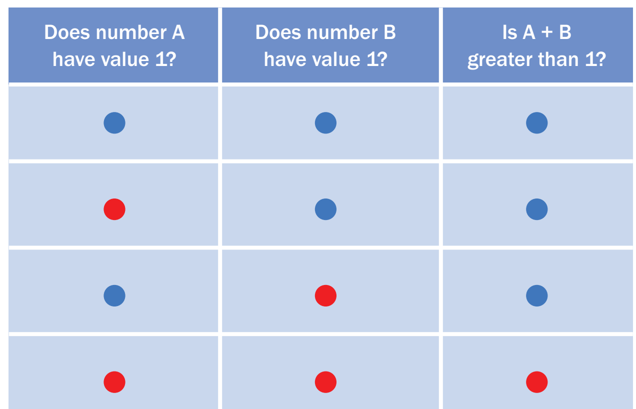


Figure 19-14. A truth table that can help to add numbers.

Logic Gate Basics

The terms AND and OR which you have encountered so far are known as ***Boolean operators***, and as I mentioned previously, seven are important to us here. They are AND, OR, NAND, NOR, XOR, XNOR, and NOT. The logic symbols that represent them are shown in Figure 19-15. Their names are usually written in capital letters.

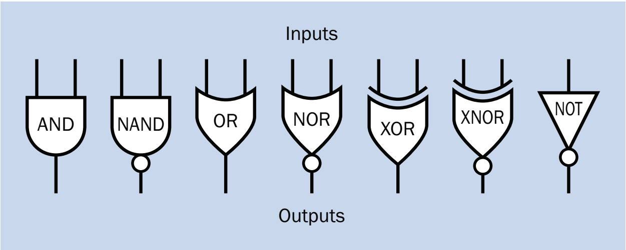


Figure 19-15. Seven logical operators used in computing devices.

The basic gates have two inputs and one output, except for the NOT gate, which has only one input and one output, and is more often referred to as an *inverter*. If it has a high input, it gives a low output, and if it has a low input, it gives a high output. I won't be using it here.

Notice that the little circles at the bottom of some of the gates invert the output. (These circles are called *bubbles*.) Thus, the output of a NAND gate is the inverse of an AND gate.

What do I mean by "inverse"? This should become clear if you look at the truth tables for logic gates that I have drawn in figures 19-16, 19-17, and 19-18. On each line of each table, two inputs are shown on the left followed by an output on the right, with red meaning a high-logic state and blue meaning a low-logic state. Compare the outputs of each pair of gates, and you'll see how the outputs are inverted.

TTL and CMOS

Back in the 1960s, the first logic gates were built with *Transistor-Transistor Logic*, abbreviated *TTL*, meaning that they contained tiny bipolar transistors. You may recall that the 555 chip is a TTL device.

Because TTL entailed significant power consumption, there was a strong incentive to use Complementary Metal Oxide Semiconductors instead, abbreviated *CMOS*. They used much less power—but they were slower.

For a while, everyone had to choose between the high performance and high power consumption of TTL, or the lower performance and low power consumption of CMOS. The two families of chips competed on that basis, and were identified with part numbers that began with 74xx for TTL and 4xxx for CMOS.

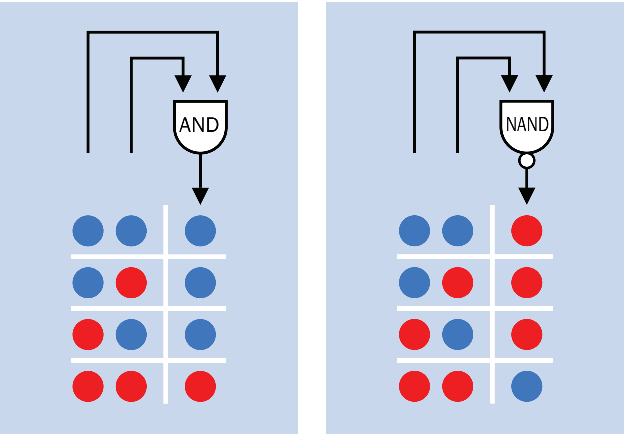


Figure 19-16. Truth tables for AND and NAND logic gates.

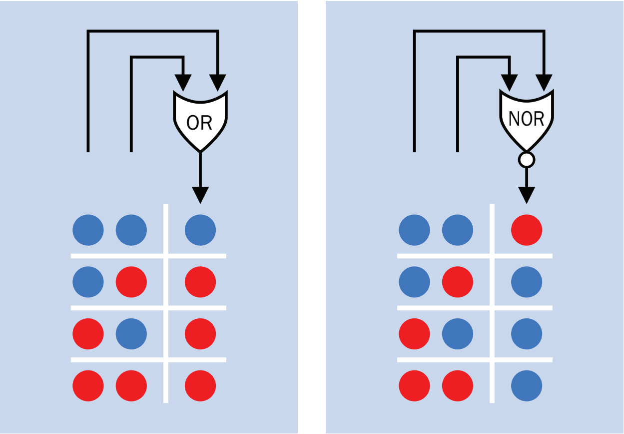


Figure 19-17. Truth tables for OR and NOR logic gates.

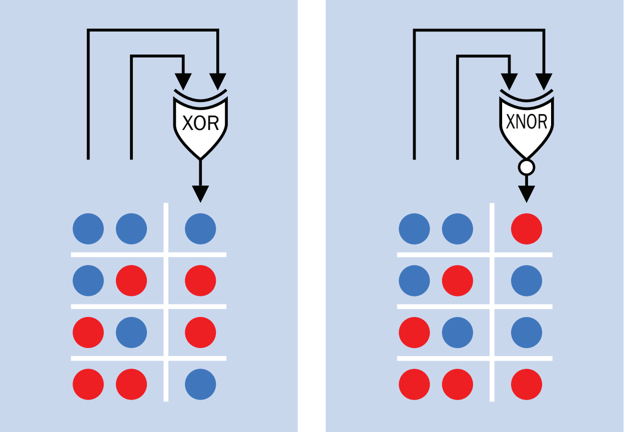


Figure 19-18. Truth tables for XOR and XNOR logic gates.

Eventually, CMOS won the contest. As technical upgrades improved its speed, manufacturers started making CMOS chips that emulated the 74xx chips, and copied their part numbers. Today, every chip in the 74xx family has CMOS circuits inside.

The original TTL versions are now obsolete, although sometimes you can still find some 74LSxx chips that are sold as spare-part replacements.

The old 4xxx chips do still exist, such as the 4026B which you used in Experiment 19. In some applications their relatively slow speed doesn't matter, and they offer a unique advantage: They can use a much wider range of voltages than the 74xx chips. In this book, I used a 4xxx chip because it could share a 9VDC power supply with some 555 timers.

Here are a few details to remember:

- The 74xx family has evolved through many generations, but the HC versions are the ones that I have chosen here (such as the 74HC08 containing AND gates) because they are so popular.
- You may still find schematics that specify old 74LSxx chips. You can use 74HCTxx chips as a substitute, because they are specifically designed to have the same specifications.
- 4xxx chips are still around, but at this point they all have part numbers ending in B. Other 4xxx chips are now obsolete and should not be used.
- The meaning of "logic high" and "logic low" for 4xxx chips is different from 74xx chips, even when they are both being powered with 5V. Usually this is not an important consideration.

If you go shopping for the HC generation of 74xx chips, include the letters HC when you search for a part number. That is, for example, you would search for 74HC08, not 7408.

However, you may notice that in my reference materials (such as Figure 19-8) I have listed the 74xx numbers without any letters, because all the generations have the same pinouts.

For more advice about the meaning and interpretation of 74xx part numbers, flip back to Figure 15-3.

Hidden Gates, Revealed

For your future reference, I have prepared 14 diagrams showing logic gates inside all the generally available through-hole logic chips. They are shown in figures 19-19 through 19-25 on the next pages. You don't know how most of these gates behave, yet, but I decided to put all the information here in one place.

You may notice that many of the logic gates have more than two inputs, even though the truth tables that I showed you previously featured two-input gates. You can understand this if you go back to figures 19-16 through 19-18 and think of them like this:

AND gate: Output is normally low.

It only goes high if *all* inputs are high.

NAND gate: Output is normally high.

It only goes low if *all* inputs are high.

OR gate: Output is normally high.

It only goes low if *all* inputs are low.

NOR gate: Output is normally low.

It only goes high if *all* inputs are low.

In these descriptions, "all" means, "all the inputs in this particular example of the gate." The maximum is eight.

XOR and XNOR gates are not usually available with more than two inputs, unless programmed for a specific application.

Rules for Connecting Logic Chips

The 74xx logic chips are amazingly simple to use, because you can chain them together without additional components. The output from an AND gate can be wired directly to the input of some other gate in some other 74xx chip, and everything will just—work!

Still, there are some rules to remember.

Permitted:

- You can tie any input of any digital gate directly to your regulated power supply, either the positive side or the negative side.

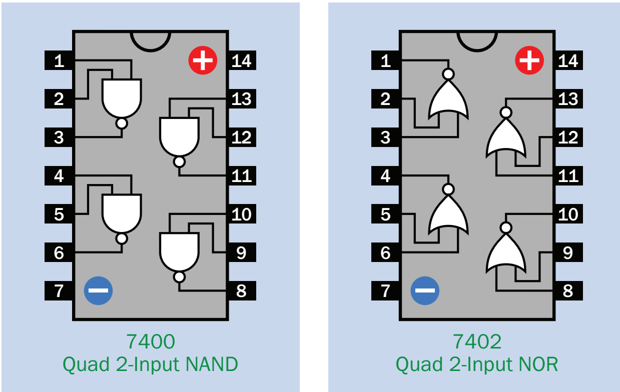


Figure 19-19. Standardized configuration for quad 2-input NAND and quad 2-input NOR chips in the 74xx family.

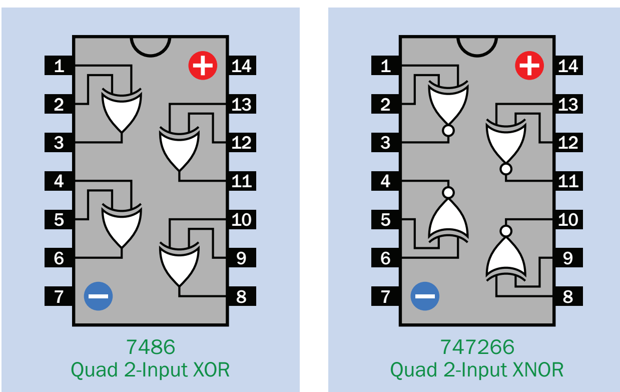


Figure 19-20. Standardized configuration for quad 2-input XOR and quad 2-input XNOR chips in the 74xx family.

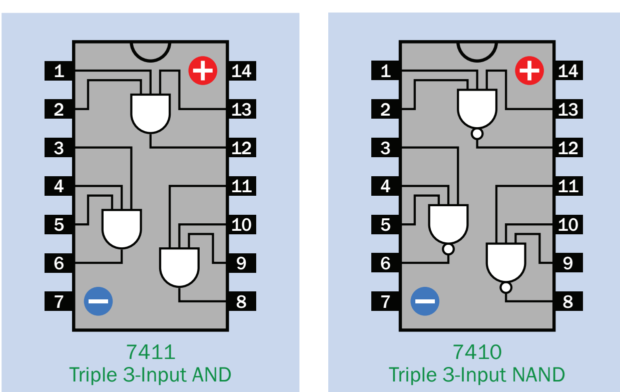


Figure 19-21. Standardized configuration for triple 3-input AND and triple 3-input NAND chips in the 74xx family.

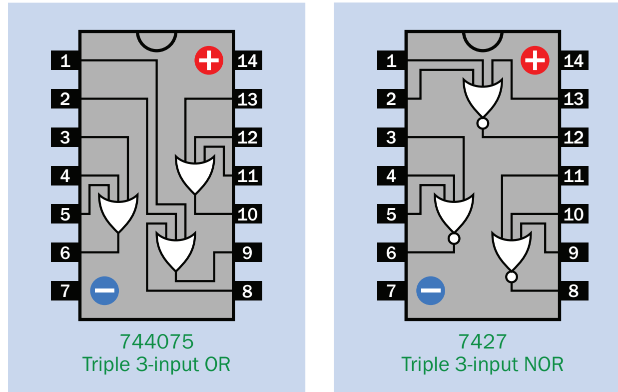


Figure 19-22. Standardized configuration for triple 3-input OR and triple 3-input NOR chips in the 74xx family.

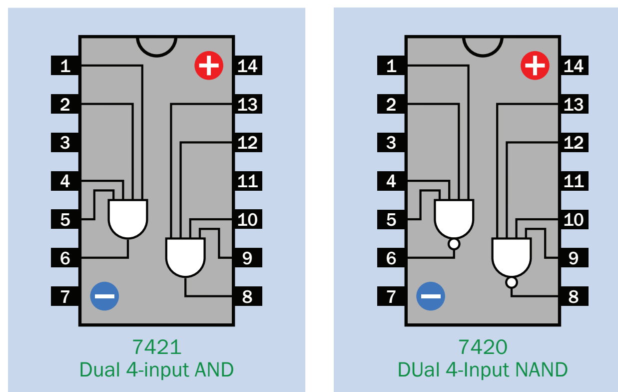


Figure 19-23. Standardized configuration for dual 4-input AND and dual 4-input NAND chips in the 74xx family.

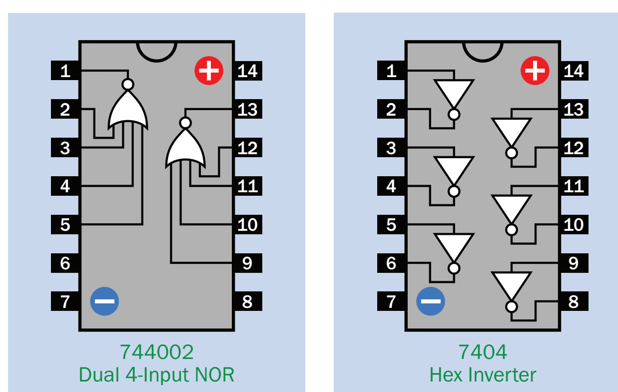


Figure 19-24. Standardized configuration for dual 4-input NOR and hex inverter chips in the 74xx family.

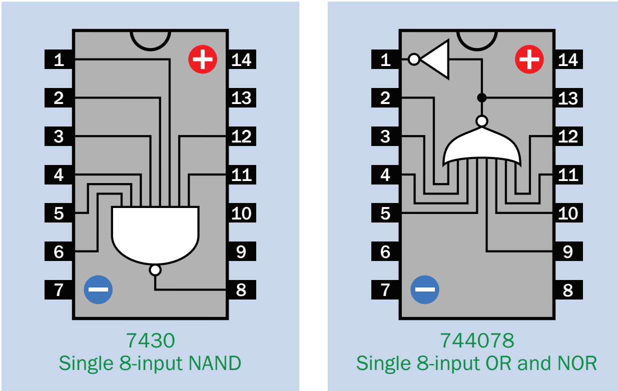


Figure 19-25. Standardized configuration for single 8-input NAND and single 8-input NOR/OR chips in the 74xx family.

- The output from one gate can be shared among the inputs of many other gates (this is known as ***fanout***). The exact ratio depends on the chip, but in the 74HCxx series, you can always power at least ten inputs from one logic output.

- The output from a logic chip can drive the Trigger Pin (Pin 2) of a 555 timer, so long as the power supply to the timer is 5VDC and shares the same negative ground as the logic chip.

- The acceptable ranges for inputs, and the minimum guarantees for outputs, are shown in Figure 19-26.

Not permitted:

- No floating inputs! On CMOS logic chips, you must always connect all input pins with a known voltage. Input pins of gates that are unused on a chip should be tied to negative ground.

- Any single-throw switch or pushbutton should be used with a pullup or pull-down resistor, so that when the contacts are open, the input to the chip is not floating.

- Don't use an unregulated power supply, or more than five volts, or less than five volts, to power 74HCxx chips.

- Be careful when using the output from a 74HCxx logic chip to power an LED. If you draw more than 4mA from the chip, this is likely to pull down the output voltage. Consequently, if you also connect the output with the input of a second chip, the voltage may drop below the minimum that the second chip needs to recognize it as logic-high. Generally, try not to use a logic output to power an LED ***at the same time as*** the input to another logic chip. (I broke this rule in one of the circuits in this book. Maybe you can find it.)

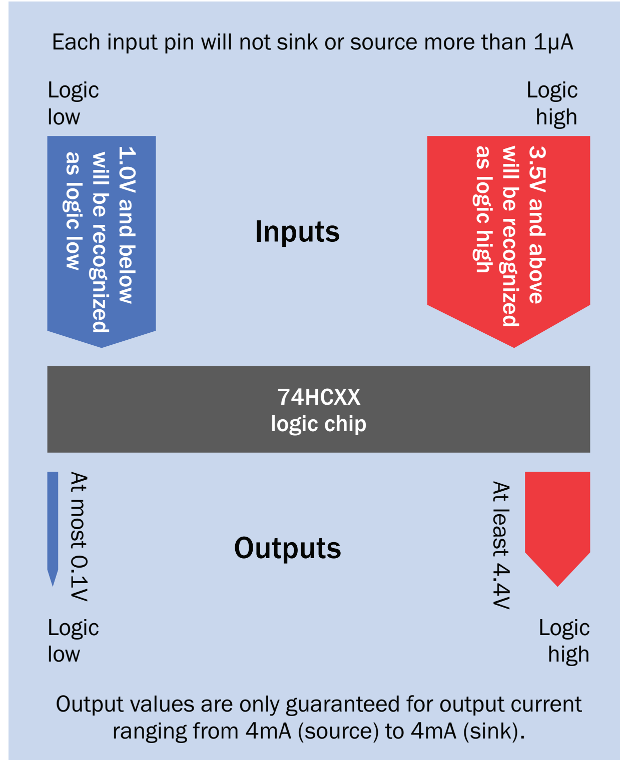


Figure 19-26. To avoid errors, stay within the recommended input ranges for logic chips.

- A logic-gate output can source current if it's logic-high, or sink current if it's logic-low, but only if you control the current. Never connect a logic-gate output directly to the power supply or negative ground.

- Avoid tying the outputs from two or more logic gates together.

So much for the do's and don'ts. Now it's time for your first serious logic-chip project.

Experiment 20
The Unlocker

Suppose you want to prevent other people from using your computer. I can think of two ways to do this: using software, or using hardware.

The software would be some kind of startup program that intercepts the normal boot sequence and requests a password, which might be a little more secure than the password protection that is a standard feature of Windows and Mac operating systems.

Personally, though, I think it's more interesting (and more relevant to this book) to do it with hardware. What I'm imagining is a numeric keypad requiring the user to enter a secret combination before the computer can be switched on.

I'll call this device an "Unlocker," even though it won't actually contain a lock. The idea of it is to bypass the "start" button that you normally use when you switch on your computer.

Caution: Think Twice About This!

The big question is whether this project is a good idea. To implement the Unlocker for your computer, you would have to open it up, cut a wire, and attach your own wires.

You would not be going near any of the circuit boards, and you would only deal with the "start" button, which uses a low voltage. Still, the phrase "voids your warranty" comes to mind.

Maybe you should just build this circuit on a breadboard, to simulate unlocking a computer. Then decide if you want to take it any further.

You could always apply it to some other device instead—such as the intrusion alarm from Experiment 17. That might be a better idea.

You Will Need:

- Breadboard, hookup wire, wire cutters, wire strippers, multimeter.
- 9V power supply, probably a battery in this case.
- Generic red LED (1).
- LM7805 voltage regulator (1).
- 74HC08 quad 2-input AND logic chips (2).
- 555 timer chip (1).
- DPDT 9VDC relay (1).
- Resistors: 470 (1), 10K (9), 100K (1).
- Capacitors: 0.01µF (1), 0.1µF (1), 0.47µF (2), 10µF (1).
- Tactile switches (9).
- 2N3904 Transistor (1).

Passcode Systems

First I'd like you to consider how a security code usually works. Suppose you are visiting someone who lives in a gated community, or you want to drop off some items at a mini-storage facility. You find yourself looking at a keypad with 10 or 12 buttons, and you have to enter a number consisting of three or four digits. If you get it right, a gate or a door opens.

That sounds simple—but really, it isn't. The first requirement, in a device of this type, is to store each key-press until the complete sequence has been entered. This requires memory of some kind, and although basic memory chips are easily available, they require other chips to make them work. You will also need to translate each key-press into a code or pattern that is suitable for storage in memory.

And then, a verification process will have to compare the digits that were entered with the correct code, which will also be stored in memory. Even that isn't the whole story, because the system has to process user errors. How should it respond if someone enters five digits instead of four? Should it ignore the first one—or the last one? Should it reject the code completely and tell the person to start over? In that case, how many tries does it allow,

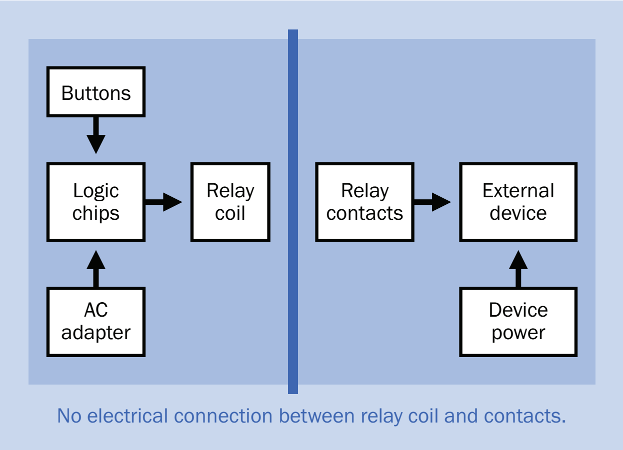


Figure 20-1. A relay makes a mechanical connection between the coil and the contacts, but no electrical connection.

Choosing a selection from a set of 8 buttons	
How many in the selection	How many ways to choose
1	8
2	28
3	56
4	70
5	56
6	28
7	8

Figure 20-2. The number of ways to choose a selection from eight buttons in an "n choose r" problem.

and does it need a little LCD screen to display an error message?

Perhaps you can see how a project like this tends to escalate. For this book, I need something simpler: A circuit that will be easy to build in an hour or so, using logic gates which fit on a breadboard. That is the mission.

Bearing this in mind, I have dumbed it down a bit. Instead of someone entering a series of digits, the person will hold down a selection of buttons simultaneously. A logic chip can deal with that by looking at the pattern, and if it's correct, maybe a 555 timer (as you might have guessed) will activate a relay.

A relay is necessary because I don't want the power supply for my circuit to have any electrical connection with the power supply of any other device that you are switching on and off. Relays are slow and clunky, but they are designed so that the coil only has a mechanical connection with the contacts, and no electrical connection. Figure 20-1 shows what I mean.

How Many Digits?

Pressing multiple buttons simultaneously is an unusual approach, but I think it can provide adequate security if you have enough buttons. Also it may be secure because people won't know how to use it. I'm imagining a scenario like this:

The intrepid system cracker is so desperate to investigate files on your hard drive, he figures he can bypass your system password by booting from a Unix flash drive that he carries with him—but wait, what's this? A numeric keypad!

He assumes that it is designed to enter a sequence of digits, like any other keypad. Okay, he sets to work. He tries all the significant numbers in your life. He enters the year or your birth, the date you graduated from high school, your phone number, or the three digits on your car license plate. And nothing works!

Even if he guesses somehow (perhaps from reading this book?) that multiple keys must be pressed together, there's no way of knowing how many. Surely it will be two or more—but suppose there are eight buttons to choose from. How many ways are there, to press a simultaneous selection?

That's an interesting question. In mathematics, this type of problem is known as "**n** choose **r**," where **n** is the number of objects in a set, and **r** is the number of them that you choose. (Why is the letter "r" used? I regret to tell you that I don't know.)

You can look up the "**n** choose **r**" formula online, where some web sites will even do the calculations for you. For instance, let's suppose I'm going to use eight buttons in this experiment. Figure 20-2 shows the number of ways to press a selection. If you only press two buttons, you can select 1 and 2, or 1 and 3, or 2 and 3, or 2 and 4 . . . and so on, and there will be 28 combinations altogether.

You'll notice in the table that the number of combinations increases and then decreases with the size of the selection that you choose. Do you see why this happens? It's because choosing some objects is the same process as not-choosing the remaining objects. That is, the number of combinations r is the same as n-r. But you aren't reading this book to have me talk about mathematics. Getting back to the point:

If you increase the value of n (the number of buttons) above eight, the total number of combinations for a selection escalates very quickly. There are 252 ways to choose five out of ten buttons, and 924 ways to choose six out of twelve . . . but I'm assuming you won't want to buy and wire more than eight buttons, so I'll settle for using eight and requiring the user to press four of them. This allows 70 combinations, which is not a very high number, but I still think my plan provides reasonable security, bearing in mind the intrepid-system-cracker scenario that I outlined above.

I have ideas for additional ways to increase the security of the Unlocker, but I'll get to them later.

The Logic Diagram

Somehow I need logic gates to verify that the correct four buttons out of eight have been pressed. But there's an additional requirement: The other four buttons must ***not*** be pressed. Otherwise, someone could just hold down all eight buttons at once, and the Unlocker would see the correct combination among them.

Suppose I number the buttons 1 through 8, and the secret combination consists of buttons 1, 2, 3, and 4. Of course in the final version of this security device, I wouldn't group the correct buttons together—I would scatter them in a nonintuitive pattern. But for demonstration and testing purposes, I'll put buttons 1 through 4 at the top of a breadboard together, and I'll put buttons 5 through 8 farther down.

Now let me summarize what needs to happen. For the system to unlock:

All of buttons 1, 2, 3, and 4 must be pressed together.

None of buttons 5, 6, 7, or 8 may be pressed.

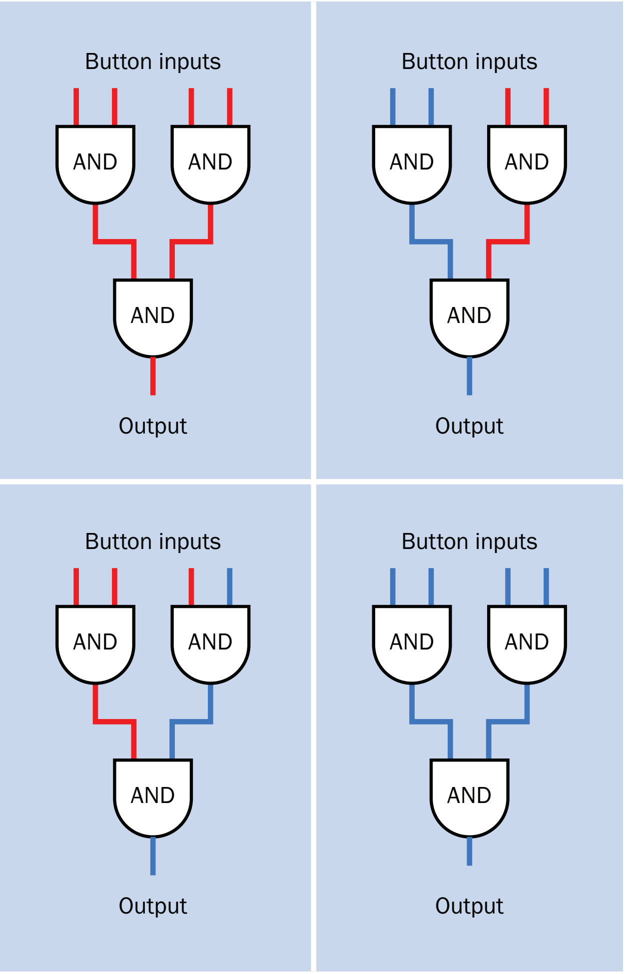


Figure 20-3. Logic diagram for three stacked AND gates.

What components should I use to do this? When I wrote that buttons "1, 2, 3, and 4. . . ." did you notice the word "and" in that phrase? Yes, I should use an AND gate.

If you turn back to Figure 19-23, you'll see that any 7421 chip contains a 4-input AND gate (actually, a pair of them). One of those gates would be ideal for my purposes, but I don't expect my readers to buy more parts than necessary. I already used a 2-input AND in the previous experiment, so maybe I can reuse that chip somehow. It actually contains four 2-input ANDs, so there should be a way for me to combine them, to process four inputs.

Indeed, there is, and Figure 20-3 shows how it's done. It shows a variety of button inputs, and I leave it to you to consider all the other combinations, if you wish.

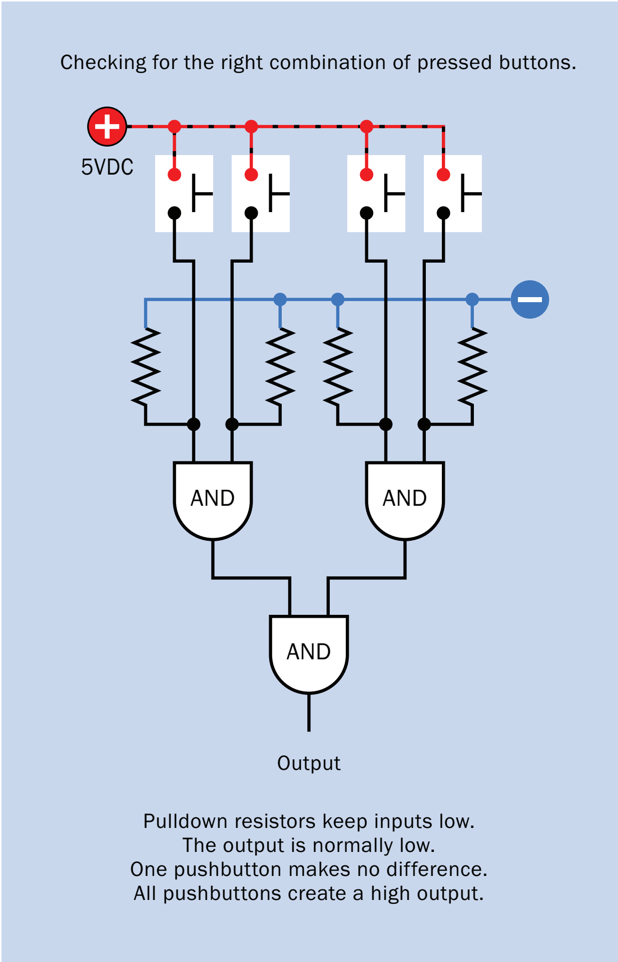


Figure 20-4. Wiring diagram for three stacked AND gates.

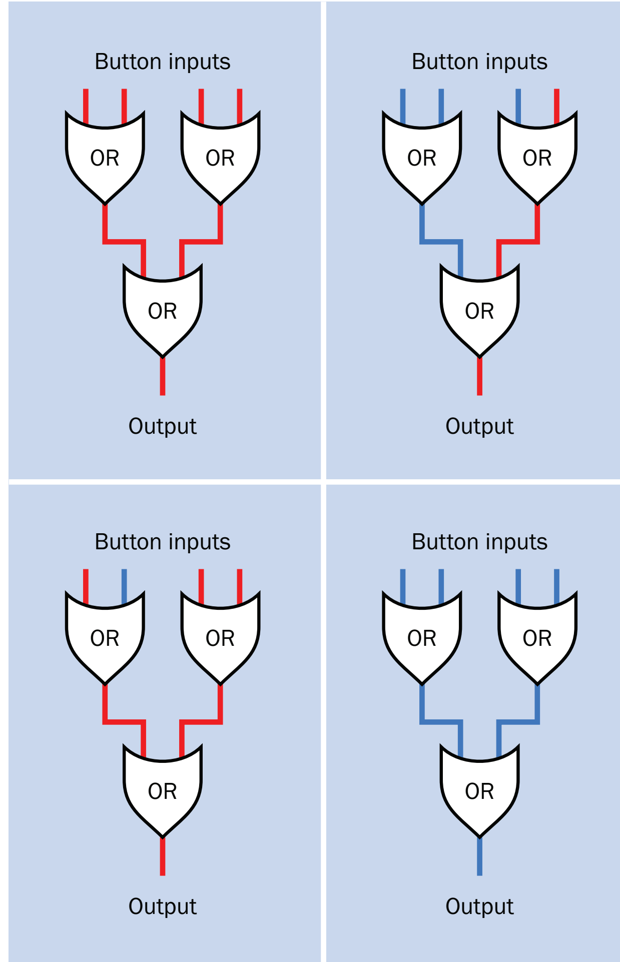


Figure 20-5. Logic diagram for three stacked OR gates.

Remember from Figure 19-16 that each AND gate has an output which is normally logic-low, and the only way to make it go high is if both of the inputs are high. After you examine Figure 20-3, I think you'll agree that to create a logic-high output at the bottom, ***all*** inputs at the top must be high. So, if buttons 1, 2, 3, and 4 deliver positive voltage to the inputs at the top level of the gates, the output at the bottom will only be positive if all of the buttons are pressed. Figure 20-4 shows how buttons and pulldown resistors would be added, with a summary at the bottom.

Incidentally, for the inputs of the third AND gate, no pulldown resistors are necessary, because the outputs from the gates above it are always logic-high or logic-low, and they never float.

- You can always add logic gates to a chain without using pullup or pulldown resistors.

Now, what about the buttons that must ***not*** be pressed? This sounds as if I need the opposite of an AND gate. Would it be an OR gate? Or a NAND gate? Or what?

Well, you could use a stack of OR gates, as in Figure 20-5. The output from each OR is logic-low if both of its inputs are logic-low. If either of its inputs goes high (or both of them), the OR output goes high. If I attach the pushbuttons to the positive supply, pressing any one of them will make the OR output go positive, and this can be a signal that the person trying to unlock the circuit has pressed a wrong button.

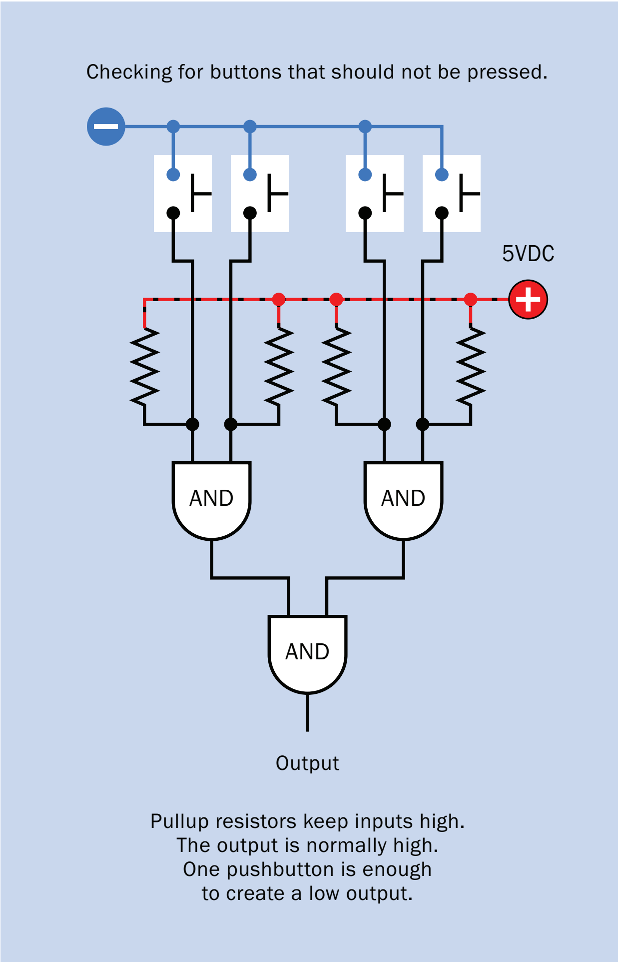


Figure 20-6. ANDing buttons that should not be pressed.

In fact when I was designing the Unlocker, my intuition told me to use OR gates, because as I wrote above, "None of buttons 5, 6, 7, or 8 may be pressed," and there is an "or" in that sentence.

No doubt, using ORs would work, except—how was I going to use the output?

My stack of AND gates would give a ***high*** output if the ***right*** buttons were pressed. The stack of OR gates would give a ***high*** output if one or more ***wrong*** buttons were pressed. But how was I going to put these factors together? I would need a circuit that would say, "If the AND gates give a high output, go ahead and unlock, unless the OR gates give a high output, in which case, don't."

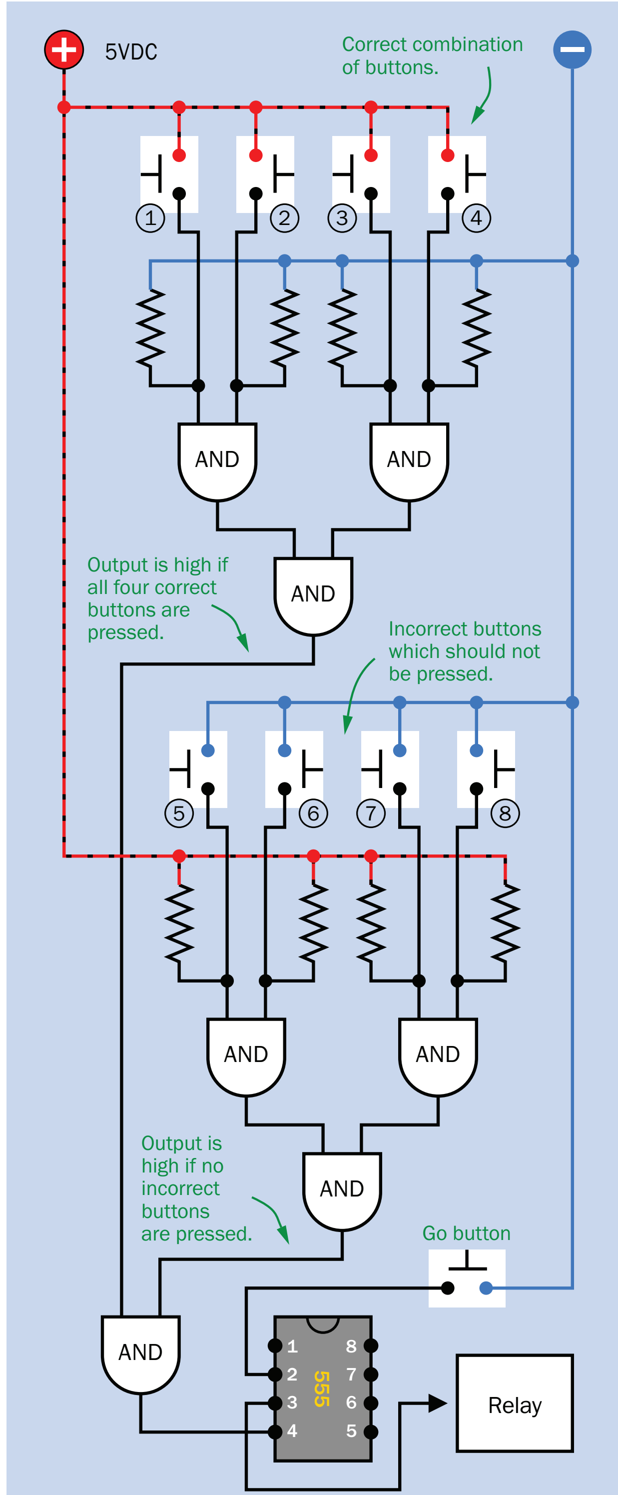


Figure 20-7. ANDing all the buttons.

Hmmm. I didn't see how to process this easily. But then I had an idea: Reverse the voltages! The "do not press" set of buttons could be processed through another stack of AND gates, if the voltages were reversed.

Take a look at Figure 20-6. The pullup resistors keep all the inputs to the AND gates high, so the output at the bottom will be high—unless a button at the top is pressed. That will cause the output to go low. Check back to Figure 20-3 if you need clarification.

Do you see why my new scheme is such an improvement? Now I have two stacks of gates, one with a ***high*** output when the ***right*** buttons are pressed, and the other with a ***high*** output so long as the ***wrong*** buttons are ***not*** pressed. When the first stack is high, and the second stack is high, it's okay to unlock the system.

Did you notice that word "and" again, in that last sentence? I could use one more AND gate to tie both of the stacks together, and a high output at the bottom means, "okay to unlock." Figure 20-7 shows what I mean. One additional AND gate connects with the Reset Pin of a 555 timer, and as I'm sure you remember, when the Reset pin goes high, it enables the timer.

At this point the person using the Unlocker presses one more button, which I call the "Go button." This causes the timer to generate a pulse, which will energize the relay. Of course, this is a simplified schematic: I have not shown the resistor and capacitor which will control the length of the pulse from the timer.

I was really surprised when I realized that this circuit could be built using seven AND gates and no other logic. Sometimes a project can be simpler than you expect.

The Unlocker on a Breadboard

Now I just had to fit all the components onto a breadboard. This is always a hassle when a lot of tactile switches are involved, because they tend to take up too much room, and you have to be careful that their output pins are all on separate rows of holes in the board. I tried to keep the layout simple, in Figure 20-8.

On the next page, you will find two more versions of the circuit: Components-only, in Figure 20-9, and the schematic in Figure 20-10.

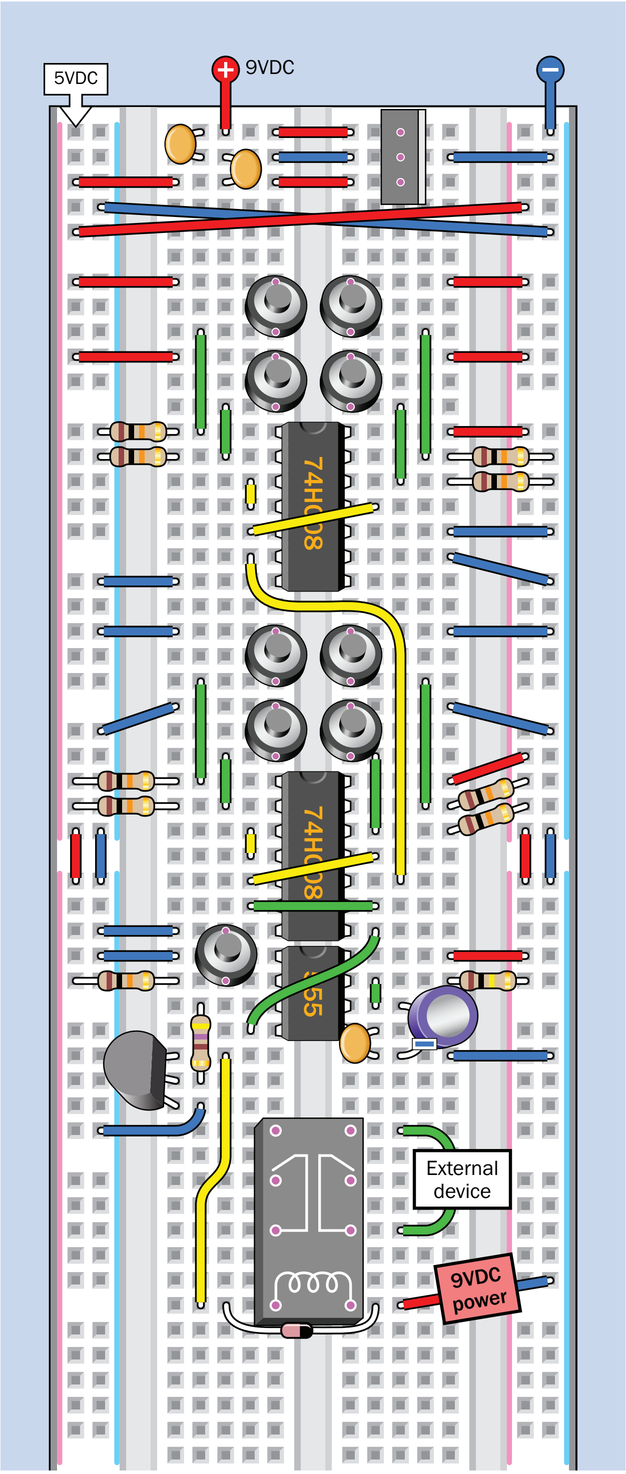
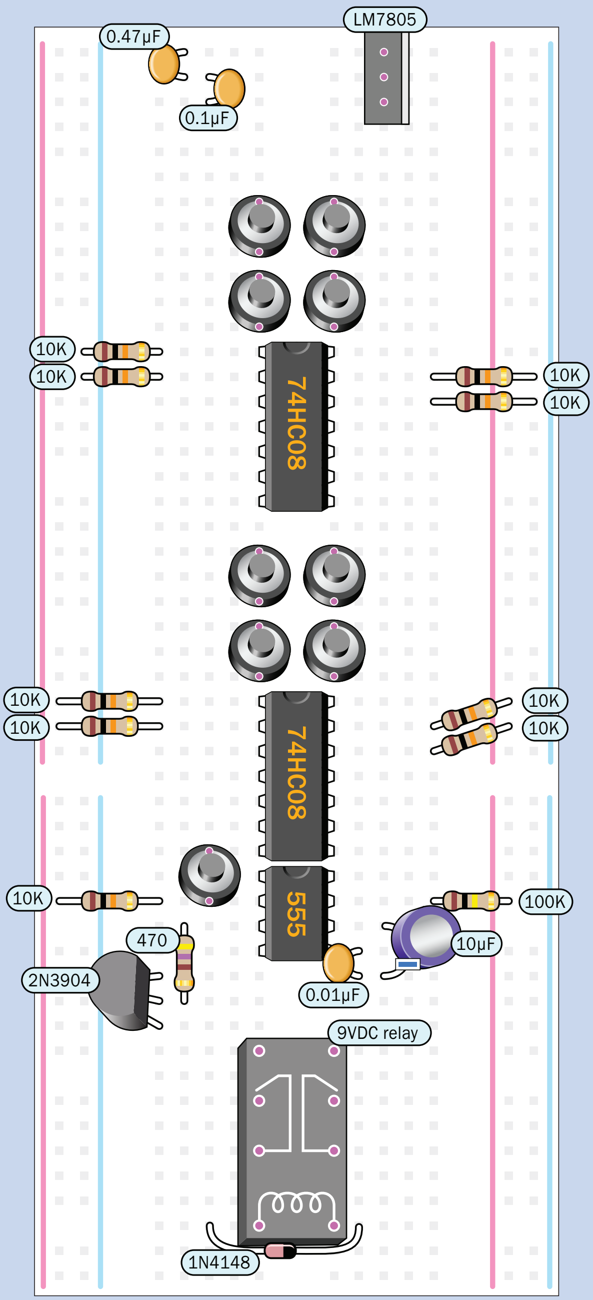


Figure 20-8. The complete Unlocker circuit.



Components

Resistors	Capacitors	
470 (1)	0.01µF (1)	Tactile sw (9)
10K (9)	0.1µF (1)	LM7805 (1)
100K (1)	0.47µF (1)	1N4148 (1)
	10µF (1)	74HC08 (2)
		555 timer (1)
		9VDC relay (1)

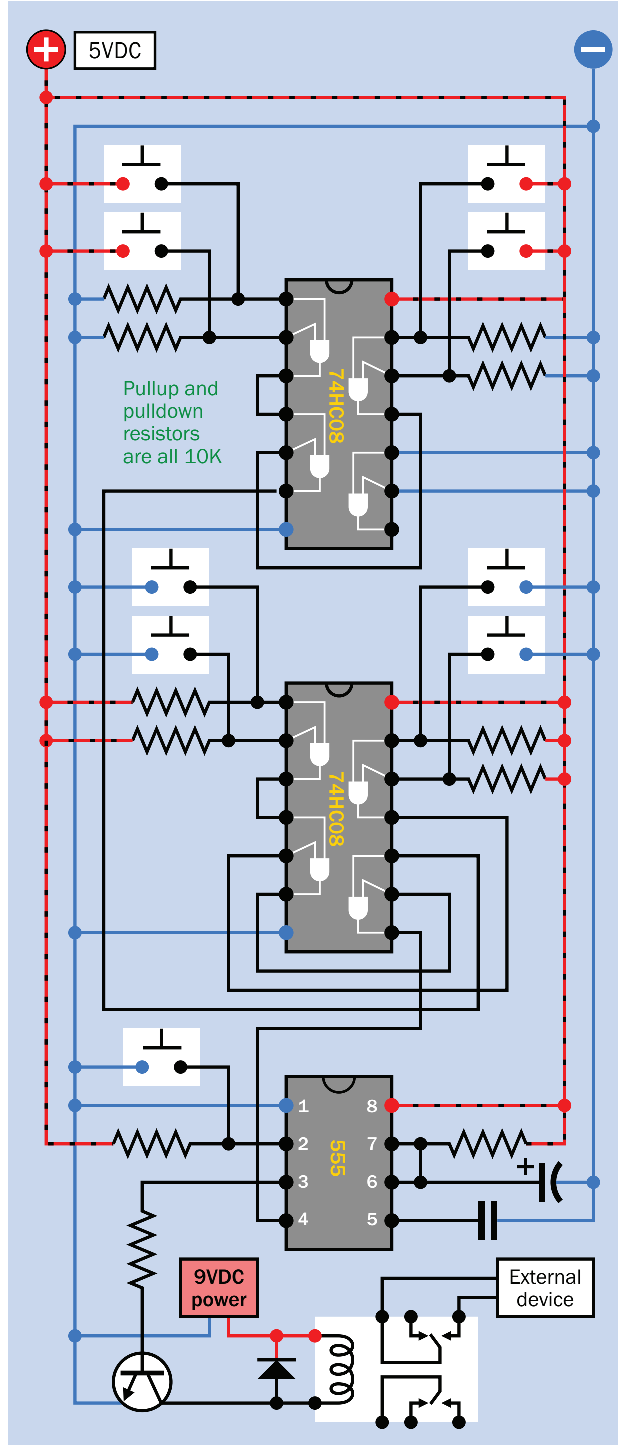


Figure 20-9 (left). Components-only version of the Unlocker.

Figure 20-10 (above). Schematic of the Unlocker.

Notice that I added a transistor to drive the relay. This is because I am assuming you don't want to buy a new 5V relay just for this circuit, and you already have a 9V relay left over from Experiment 7. A transistor can take a 5V output from the 555 timer and use it to switch 9V for the relay. (Of course, if you do happen to have a 5V relay, you can omit the transistor and wire the relay coil between the 555 output and ground.)

To test the circuit, you may want to add an LED indicator and a series resistor between the output of the 555 timer and negative ground, instead of "External device."

Now press all the buttons from 1 to 4 and none of the buttons from 5 to 8. That sends a high signal to the Reset Pin of the 555 timer, so that when you press the Go button, the timer will respond. If you don't press all of the top four buttons, or if you do press any of the lower four buttons, the "unlock" signal will not be created.

The 100K resistor and 10µF capacitor attached to the 555 timer will deliver a three-second pulse, which is all you need if you intend to use this circuit with the "on" button of a computer. If you are planning to use it for some other device, you may want to rewire the timer in bistable mode, so that its output will continue for as long as the Unlocker is powered up.

Be careful to keep the 9V power supply separate from the 5V supply which comes from the voltage regulator. I can guarantee that 5V logic chips won't like it if you try to power them with 9V. At the same time, though, your 9V supply must share the negative ground with the 5V supply—otherwise, the transistor won't work. What you can do is run a long wire all the way from the 9VDC input at the top of the board, down to the right end of the relay coil which I labeled appropriately.

You may notice that I added a diode below the relay coil, oriented against conventional current. This is sometimes known as a ***freewheeling diode***. When current is applied to a coil and then switched off, this causes a brief surge from the energy stored in the coil, and it can damage components such as the transistor which switches the relay in this circuit. The diode blocks forward current, but it allows the reverse surge to pass through it instead of hitting the transistor.

- Where sensitive electronic components are sharing a circuit with a device containing a coil, a freewheeling diode is a necessary precaution.

The Computer Interface

If you really want to use the Unlocker on a desktop computer, proceed at your own risk. I don't recommend it, but this is how it would be done.

First, make sure you have wired the combination lock circuit correctly. A single wiring error can cause your circuit to deliver 9VDC through the left-hand relay contacts instead of merely closing a switch. This is important!

Now let's consider how your computer normally functions when you want to switch it on.

Old computers used to have a big switch at the back, attached to the heavy metal box inside the computer that transformed house current to regulated DC voltages that a computer needs. Modern computers are not designed this way; you leave the computer plugged in, and you touch a little button on the box (if it's a Windows machine) or the keyboard (if it's a Mac), which is connected by an internal wire to the motherboard.

This is ideal from our point of view, because we don't have to mess with high voltages. Inside your computer, don't even think of opening that metal box with the fan mounted in it, containing the computer power supply. Just look for the wire (usually containing two conductors, on a Windows machine) that runs from the "power up" button to the motherboard.

First, ***make sure that your computer is unplugged***. Second, ground yourself if possible. Now find the wire. Very carefully snip just one of the two conductors in the wire. Now plug in your computer and try to use the "power up" button. If nothing happens, you've probably cut the right wire. (Even if you cut the wrong wire, it still prevented your computer from booting, which is what you want, so you can use it anyway.)

Remember, you are not going to introduce any voltage to this wire. You're just going to use the relay as a switch to reconnect the conductor that you cut. You should have no problem if you maintain a cool and calm demeanor, proceed methodically, and look for that single wire that starts everything.

After you find the wire and cut just one of its conductors, keep your computer unplugged during the next steps.

Strip insulation from the two ends of the wire that you cut, and solder an additional piece of two-conductor

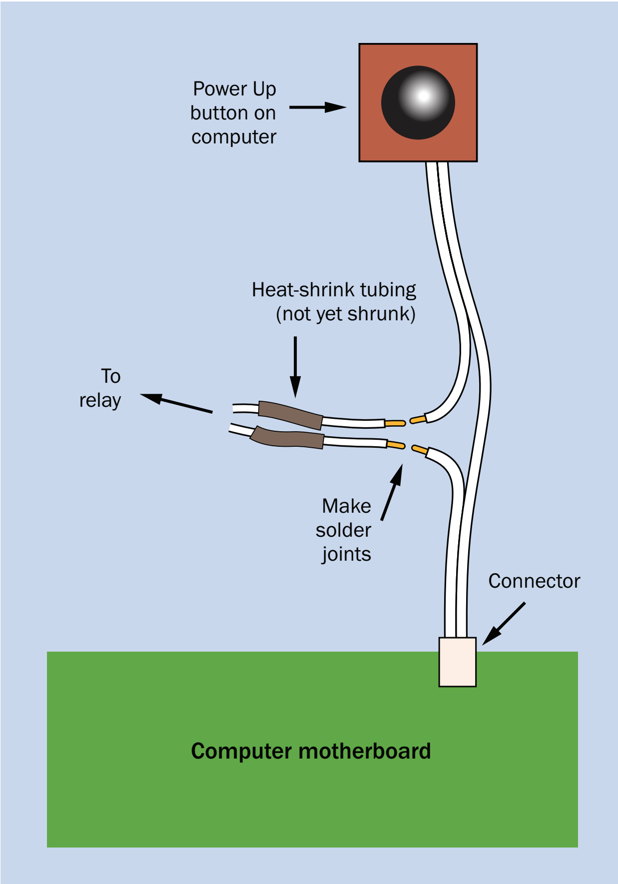


Figure 20-11. If you were planning to add a switch to your computer, this might be how you'd do it.

wire, as shown in Figure 20-11, with heat-shrink tube to protect the solder joints.

Run your new piece of wire to the relay, making sure you attach it to the pair of contacts which close when it is energized by the Unlocker. You don't want to make the mistake of unlocking your computer when you think you're locking it.

Plug in your computer, and press the computer's "start" button. If nothing happens, this is good! Now press the secret combination on your keypad and try the "start" button again during the three seconds in which the relay is closed. The relay is in series with the button on the computer, so they both have to close, for the button to work.

Installation

Having tested your circuit, the only remaining task is permanent installation. Just remember to remove the case completely from the rest of the computer if you are contemplating something along the lines shown in Figure 20-12.

Enhancements

I made this project as simple as possible, which means I can think of a lot of enhancements to complicate it.

Add more buttons. Assuming your secret combination consists of pressing half of the total number of buttons, the number of combinations will double, approximately, for each additional button. Of course, you'll have to buy some extra AND chips to process the extra buttons—or will you?

Include fake buttons. You can use buttons that are not connected to anything. This will still increase the number of possible combinations, although you won't be able to check if your new buttons have not been pressed.

Add a failure counter. You could use a counter chip to count the number of unsuccessful attempts to enter a password, and lock out subsequent attempts for a period of time. To do this, you would add a decade counter with ***decoded outputs***, meaning that it has 10 output pins which go logic-high one-at-a-time. When the third of fourth pin goes high, it could trigger a timer which generates a 30-minute pulse. You would rewire the circuit so that instead of powering the Go button from the negative bus, you power it from the output of your new, long-term timer. When the timer output goes high, the button stops working. The only difficult part would be to figure out how to reset the counter to zero. Possibly you could add a coupling capacitor between the output of the long-term timer and the reset pin of the counter, but this will depend whether the counter is reset by a low-to-high or high-to-low transition.

Buy a Keypad. In the first edition of this book, I suggested using a numeric keypad for this project. Some people felt that the keypad cost too much, and others had difficulty finding the right kind of keypad. It's still a possibility, so long as you are willing to shop around, and you buy a keypad where each key closes its own pair of contacts. If you buy a keypad that is ***matrix encoded***, it is really intended

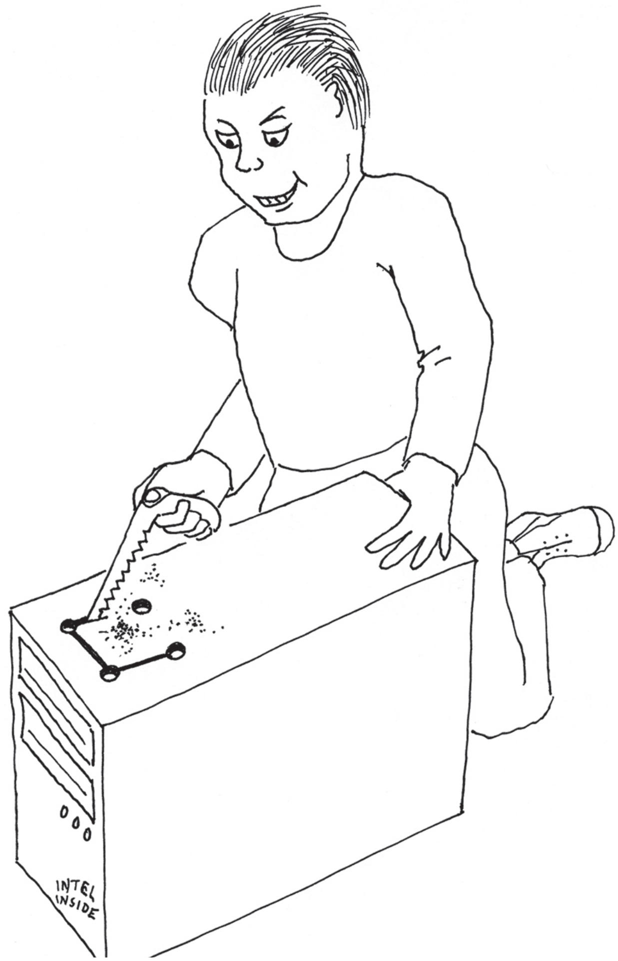


Figure 20-12. An option for installing your keypad (not necessarily recommended).

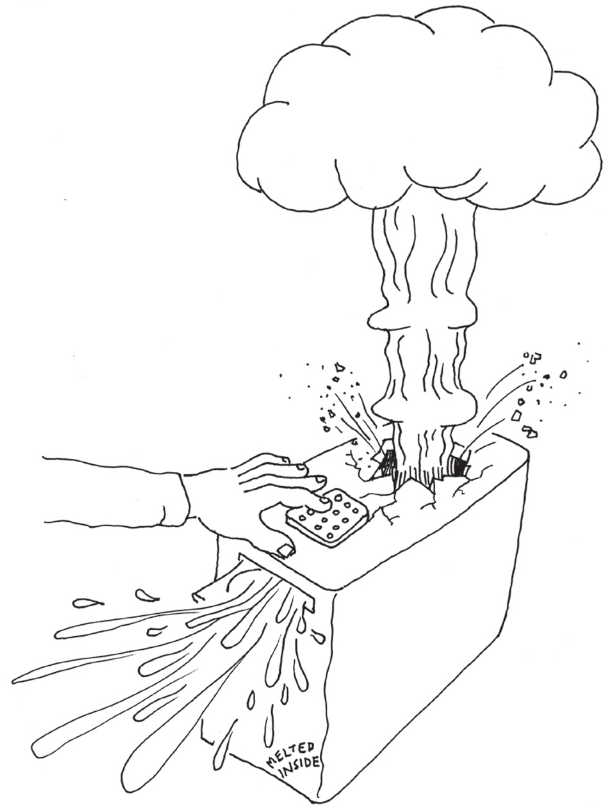


Figure 20-13. A meltdown/self-destruct system provides total protection.

for use with a microcontroller, and won't be compatible with your AND gates.

Protect the computer. To make this project more secure, you could add tamper-proof screws to the case. Naturally, you will also need the special tool that fits the screws, so that you can install them (or remove them, if your security system malfunctions for any reason).

Enable code updates. Another enhancement would be a way to facilitate changing your secret code if you feel the need. This will be difficult if you make a soldered version of the circuit, but you can install headers to allow you to swap the wires around. You might also consider adding *DIP switches* (dual-inline-pin switches . . . you can search online for this term, and see what you find).

Destructive Security. For those who are absolutely, positively, totally paranoid, you could fix things so that entering a wrong code flips a second high-amperage relay which supplies a massive power overload, melting your CPU and sending a big pulse through a magnetic coil clamped to your hard drive, instantly turning the data to garbage.

There's no doubt about it: Messing up hardware has major advantages compared with trying to protect data by using software. It's faster, difficult to stop, and tends to be permanent. So, when the Record Industry Association of America comes to your home and asks to switch on your computer so that they can search for illegal file sharing, just accidentally give them an incorrect unlocking code, sit back, and wait for the pungent smell of melting insulation—or a burst of gamma rays, if you go for the nuclear option (see Figure 20-13).

On a more realistic level, no system is totally secure. The value of a hardware locking device is that if someone does defeat it (for instance, by figuring out how to unscrew your tamper-proof screws, or simply ripping your keypad out of the computer case with metal shears), at least you'll know that something happened—especially if you put little dabs of paint over the screws to show that they've been messed with. By comparison, if you use password-protection software and someone defeats it, you may never know that your system has been compromised.

So much for computer security measures. The next experiment will be a little bit more practical.

Experiment 21
The Button Blocker

Here's another project using logic gates, this time to emulate a TV quiz show. The circuit only requires one OR chip and two timers, but it demonstrates the concept of feedback, and I think you'll find the concept challenging.

You Will Need:

- Breadboard, hookup wire, wire cutters, wire strippers, multimeter.
- 9V power supply (battery or AC adapter).
- 74HC32 quad 2-input OR chip (1).
- 555 timers (2).
- SPDT slide switch (1).
- Tactile switches (2).
- Resistors: 470 ohms (3), 10K (3).
- Capacitors: 0.1µF (3), 0.47µF (1).
- LM7805 voltage regulator (1).
- Generic red LEDs (3).

Button Blocking

Whenever I see a quiz show where contestants compete to be the first to answer a question, I wonder about the electronics. Somewhere hidden behind the scenery there has to be what I call a Button-Blocker circuit. When the fastest player presses a button, it blocks the other players. A minute later, the quizmaster resets the system for the next question, somehow. But how?

I've seen circuits online which do it, but some of them seem a bit too simple, while others seem a bit too complicated. I decided to build my own with additional "quizmaster control" to make a more realistic game. I also designed it to be easily expandable, serving almost any number of players.

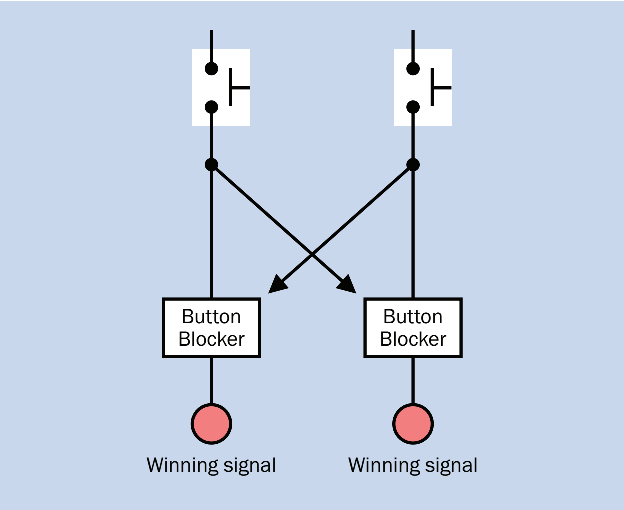


Figure 21-1. First player blocks the other player—somehow.

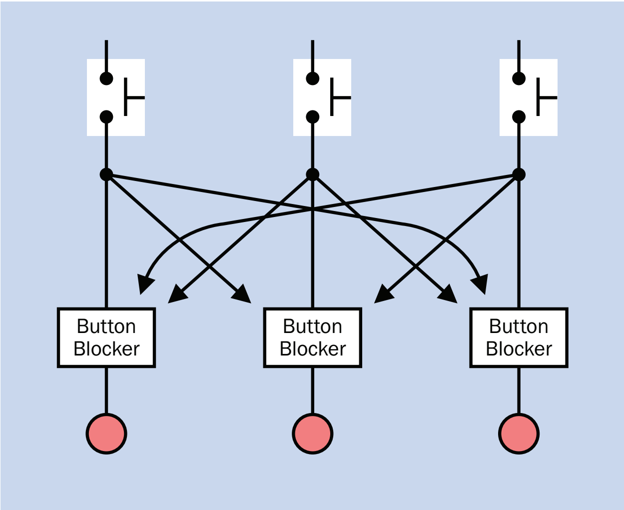


Figure 21-2. More players, more complicated connections.

A Conceptual Experiment

First let me restate the basic concept: Let's just have two players, each with a button to press, and whichever button is pressed first will block the other button.

The sketch in Figure 21-1 shows what I mean. The first person to press a button will light up a winning signal, but at the same time, the button will activate the other person's button blocker. I have used arrows to suggest some kind of connection which I'm not sure about, yet. This first step is simply to define the problem.

Now that I'm looking at it, I see something that I don't like. Suppose I want to expand this game to three players. Now each person's button has to activate the button blockers of two other people, and my nice simple diagram starts to get messy, as in Figure 21-2. Also, each button blocker now has to be able to process multiple inputs, and if I imagine a four-player game, the situation will be even more complicated. Anytime I see this kind of complexity, I think there has to be a better way.

Here's another thing to worry about. After a player lets his finger off the button, the other players' buttons will be unblocked again. Really, the button blocker should be a ***latch*** (also known as a flip-flop).

Hmm. How about if the latch does two things? It locks the output from the winning player's button in an "on" state, while suppressing new inputs from ***all*** the buttons.

I can summarize it like this:

- First player presses his button.
- His winning status is latched.
- The latched signal feeds back and blocks all the buttons.

Figure 21-3 illustrates this. Instead of a lot of connections, now there is one ***data bus*** serving all the players. This should be expandable without increasing the complexity. (A data bus is like a power bus, but for data.)

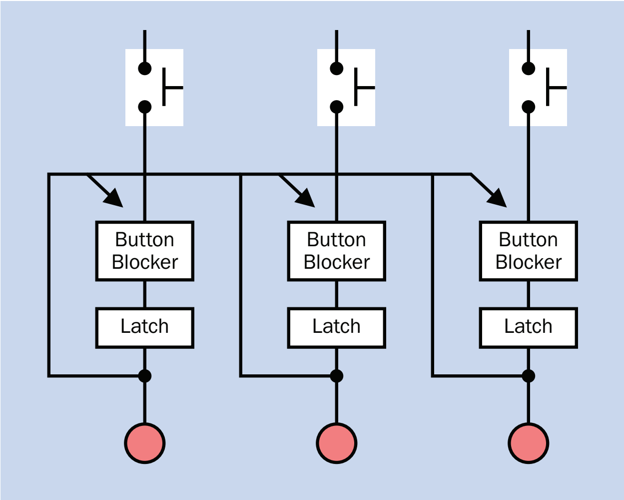


Figure 21-3. Any latch now blocks all the buttons.

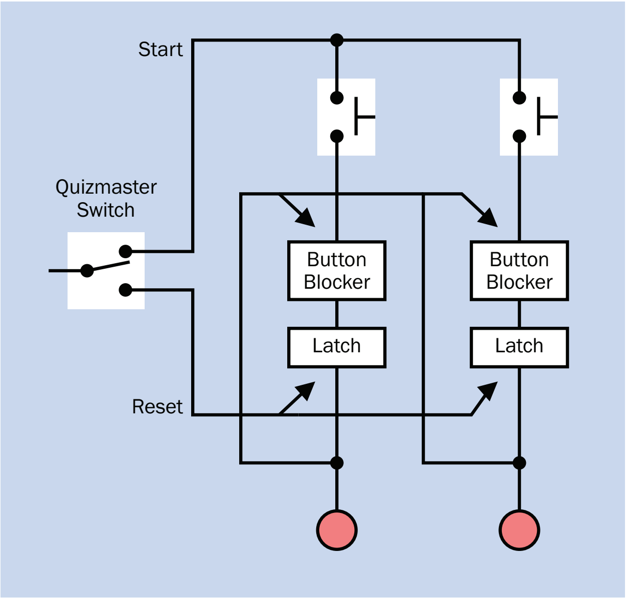


Figure 21-4. A Quizmaster Switch has been added to enable each game and reset each latch after a player has won.

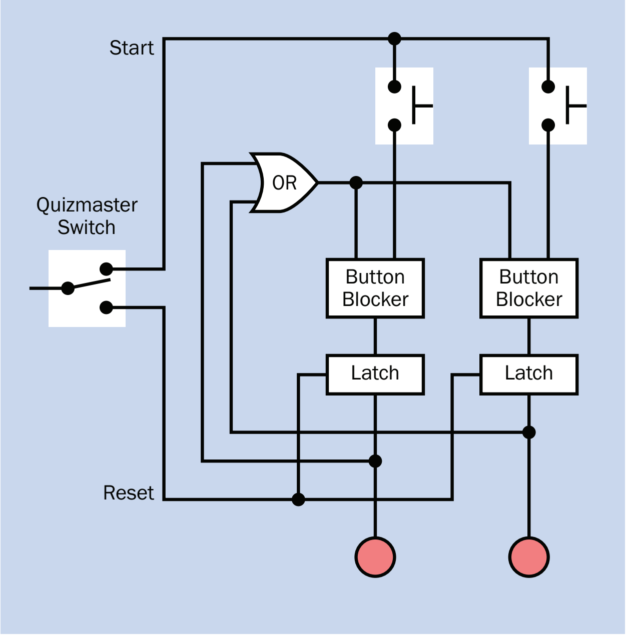


Figure 21-5. Adding an OR gate isolates one player's latch circuit from the other.

There's something important missing, though: a reset switch, to put the system back to its starting mode after a winner has been determined. The reset switch can also have a second position, the "start" position, to activate the players' buttons. This will prevent anyone from pressing a button to win the game before the quizmaster has finished asking the question.

To deal with these requirements, I added a double-throw Quizmaster Switch in Figure 21-4. Space limitations allow me room to show only two players, but the concept is still easily expandable.

Now it's time to be a bit more realistic and get rid of the arrows. The latches will all be reset together, so one connection from the Quizmaster Switch can be wired to all of them. The feedback to the Button Blockers may be a little more difficult, because I won't want the output from one latch to circle around and down, lighting the other player's winning signal. Therefore I'll add an OR gate, in which the outputs from the latches are separated from each other internally. See Figure 21-5.

The basic OR gate has only two logical inputs. Will this prevent me from adding more players? No, because you can buy an OR with up to eight inputs, or you can stack OR gates as shown in Figure 20-5 in the Unlocker project. All OR gates work the same way: If any of the inputs is high, the output is high.

Now I have to decide how the latch will work. I can buy an off-the-shelf flip-flop chip, which flips "on" if it gets one signal and "off" if it gets another, but chips containing flip-flops tend to have more features than I need for a simple circuit like this, and they have low-powered outputs, like most logic chips. I will introduce them to you in the next experiment, but for this project I'm going to use 555 timers yet again. They require very few connections, work very simply, and can deliver a good amount of current to drive bright LEDs.

Because this circuit will contain two 555s interacting with each other, I'll just remind you how each of them works in bistable mode. In Figure 21-6 you can see that when the Reset Pin is low, this creates a low output regardless of the state of the Trigger Pin. This is exactly what I want for the Quizmaster Switch in the Button Blocker circuit.

555 timer running in bistable mode		
Reset Pin	Trigger Pin	Output Pin
Low	Ignored	Low
High	High-to-low	High
High	Low-to-high	Unchanged

Figure 21-6. How a 555 timer behaves in bistable mode.

When the Reset Pin is high, it allows the Trigger Pin to take control. If the Trigger pin goes low, it triggers a high output. If the Trigger Pin then goes from low to high, the output remains unchanged—till the Reset Pin pulls it low again.

The only thing I don't like about this behavior is that both the Reset Pin and the Trigger Pin are active-low. They need a low input to make them respond. All right, then, in that case, each player's pushbutton will have to send a low signal. And when a pushbutton is not being pressed, it will have a pullup resistor. You can see the beginnings of this circuit in Figure 21-7, where the Quizmaster Switch is also active-low. I'm still using a simplified format, so I'm indicating the fixed high and low pin states of the timers with colored dots. Where a pin is not connected to anything, I'm not showing it.

I'm still using arrows where I'm sketchy about the details. For Player 1, I have labeled them A and B, and somehow they have to connect with point C, the Trigger Pin of the timer. I'm betting a logic gate will do the job, but which logic gate?

I'll have to think this through. When the Quizmaster Switch is in its reset position (pointing to the left), it resets the 555s and their outputs are forced low. Then the switch flips to the right, to start the game.

At this point, because the timer outputs are still low, the OR inputs are low. Therefore, at point A, the voltage is low.

Now I will just consider the situation for Player 1. The player's button has not been pressed, so the pullup re-

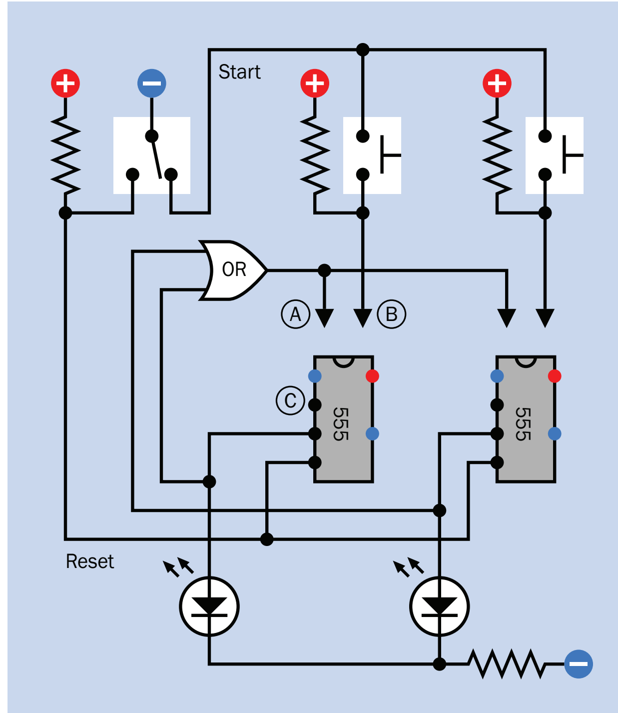


Figure 21-7. A simplified circuit using 555 timers.

sistor applies a high voltage to point B. Under these circumstances, I don't want the 555 to respond yet, so its input has to be high. Therefore:

A low, B high: C must remain high.

Now Player 1 presses the button. Its output goes low. This should trigger the timer by pulling C down to a low state. So:

A low, B low: C must go low.

The timer is triggered, so its output goes high. It circles back through the OR gate, so now the OR has one high input. Its output goes high, and the voltage at position A goes high. This must block any more button-presses. So:

A high, B low-or-high: C must be high.

Point A (from OR gate)	Point B (from pushbutton)	Point C (Trigger Pin of timer)
🔵	🔵	🔵
🔵	🔴	🔴
🔴	🔵	🔴
🔴	🔴	🔴

Figure 21-8. Truth table for the timer input in Figure 21-7.

Now I can create a truth table, which you can see in Figure 21-8.

Compare this truth table with the one on the left in Figure 19-17, and you'll see that I need another OR gate for each timer.

Because this circuit is a little tricky to understand, I will take you through it again in four steps, beginning with Figure 21-9. This shows the situation when the quizmaster has flipped the two-way switch to "reset" mode, forcing the Reset Pin of each timer into a low state. When the reset pins are low, they force the outputs low. The outputs circle around to the left-hand OR gate, which I have renamed OR1. The low output from OR1 is shared between OR2 and OR3, but because the pullup resistor on each player's pushbutton keeps the other pins in OR2 and OR3 high, their outputs cause the Trigger Pin on each timer to be high. Even if a player presses a pushbutton, the Quizmaster Switch is still in its reset position, so the pushbuttons are not energized and have no effect.

In Step 2, the quizmaster has asked a question and flipped her switch to the right, to supply (negative) power to the players' buttons. Remember, this is an active-low circuit. A low pulse from a player's pushbutton will trigger a timer. Neither of the players has responded yet, but the pullup resistor beside the quizmaster's switch is now energizing the Reset Pin on each timer, so they are not locked down anymore. Either of the timers can now be triggered, as soon as a player presses a button.

In Step 3, Player 1 has pressed a pushbutton, sending a low pulse to OR2. Now that OR2 has two low inputs, its output has gone low. The low pulse goes to the trigger pin of the left-hand timer—but components do not respond instantaneously, and the timer has not processed the signal yet.

In Step 4, a few microseconds later, the timer has processed the low input signal and created a high output pulse. This lights the LED and also circulates back to OR1. Now that OR1 has a high input, it also has a high output, because an OR output is always high when at least one input is high. The high output from OR1 goes to the inputs of OR2 and OR3, and so their outputs become high, too. These outputs go to the Trigger Pins of the timers, so any low signals from pushbuttons will be ignored now. The buttons have been blocked!

Remember, when a Trigger Pin on a 555 timer changes from low to high, its output remains unchanged. Consequently the Output Pin on timer 1 remains high, and the Output Pin on timer 2 remains low.

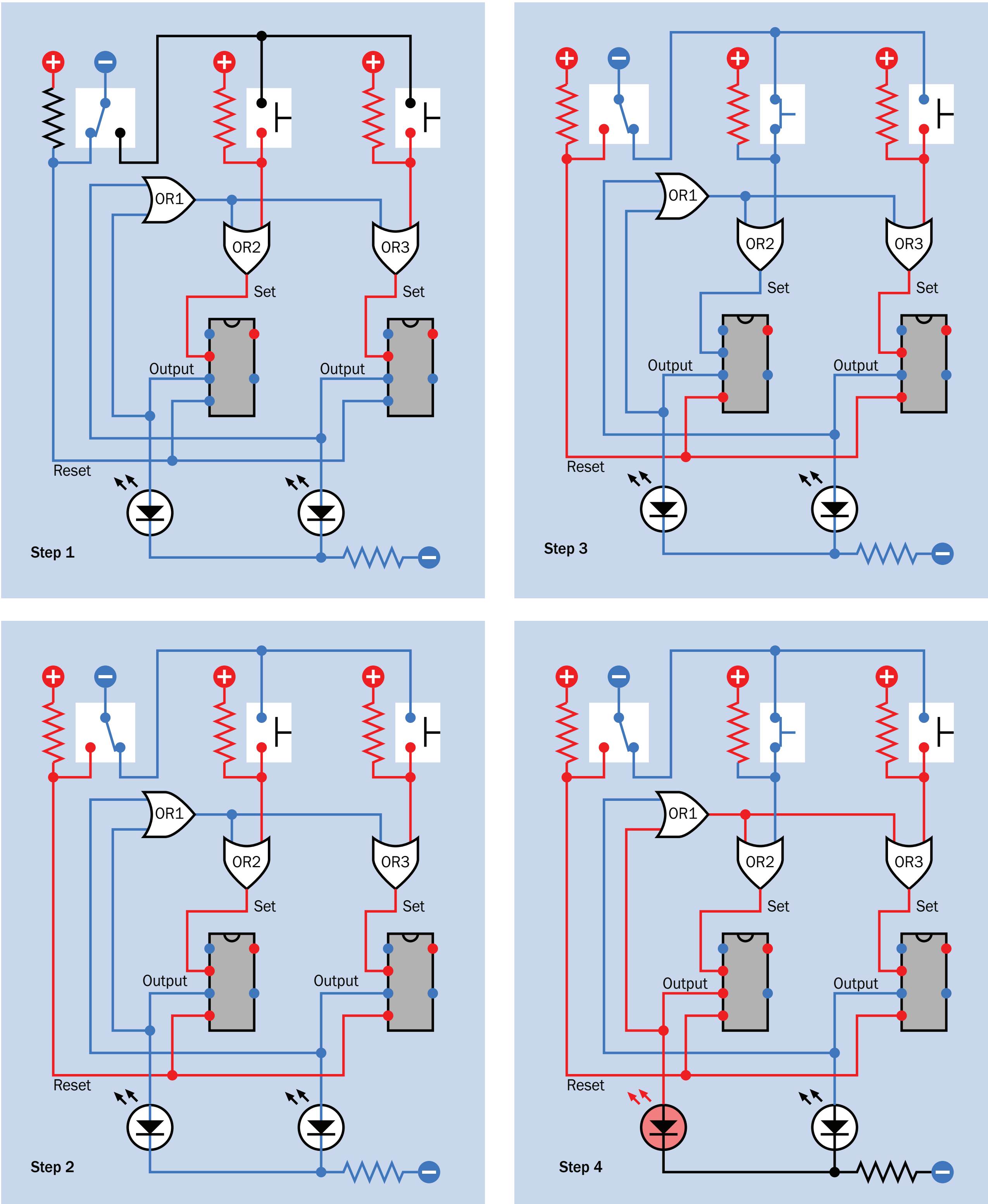
What Can Go Wrong?

There's only one situation that can defeat the Button Blocker. What if both players press their buttons almost at the same instant, so that the electronic components cannot react quickly enough to tell the difference? Both of the LEDs will light up before either of the timers can block the other.

On TV quiz shows, you never see this. Absolutely never! If the electronic system on a show registers a simultaneous response from two players, do you think it has a

Figures 21-9 through 21-12 (opposite), from top to bottom, left to right:

The sequence of events when no player can make a move (Step 1), buttons are activated (Step 2), Player 1 presses a button (Step 3), and the LED for Player 1 lights up, while Player 2 is blocked (Step 4).


OR1
OR2
OR3
Set
Set
Output
Output
Reset
Step 1
OR1
OR2
OR3
Set
Set
Output
Output
Reset
Step 3
OR1
OR2
OR3
Set
Set
Output
Output
Reset
Step 2
OR1
OR2
OR3
Set
Set
Output
Output
Reset
Step 4

feature to pick one of them at random? Just speculating, of course. But if I was building that system, I'd add that feature.

Breadboarding It

The breadboard diagram is shown in Figure 21-13, and because there are very few components, I haven't added a separate components-only diagram. The schematic is in Figure 21-14.

Because the only logic gates that I've used are OR gates, and only three of them are needed, you only need one logic chip: the 74HC32, which contains four 2-input ORs. The two OR gates at the top end of the chip have the same functions as OR2 and OR3 in my simplified schematic, and the OR gate at the bottom-left side of the chip works as OR1, receiving input from pin 3 of each 555 timer. If you have all the components, you should be able to put this together within an hour. To test it, start with the two-way switch with the actuator at the low end. This is the reset position. Now move the actuator up, and an LED lights to tell the players they can race each other to press their buttons. You can try to press the two buttons simultaneously, but I'll bet you the Button Blocker responds to just one of them, and blocks the other.

You may notice that I've added a 0.1µF capacitor between Pin 2 of each 555 timer (the Input Pin) and Pin 1 (which connects with negative ground). Why? Because when I tested the circuit without the capacitors, sometimes I found that one or both of the 555 timers would be triggered simply by flipping the Quizmaster Switch, without anyone pressing a button.

I wondered if the timers were responding to tiny and very rapid vibrations in the contacts when the switch was moved. This is known as ***contact bounce***, and sure enough, it turned out to be happening. The capacitors solved the problem. They may slow the response of the 555 timers fractionally, but not enough to interfere with slow human reflexes.

As for the player buttons, it doesn't matter if they "bounce," because each timer locks itself on at the very first impulse and ignores any more that follow. I'm going to explain more about switch bounce, and how to get rid of it, in the next experiment.

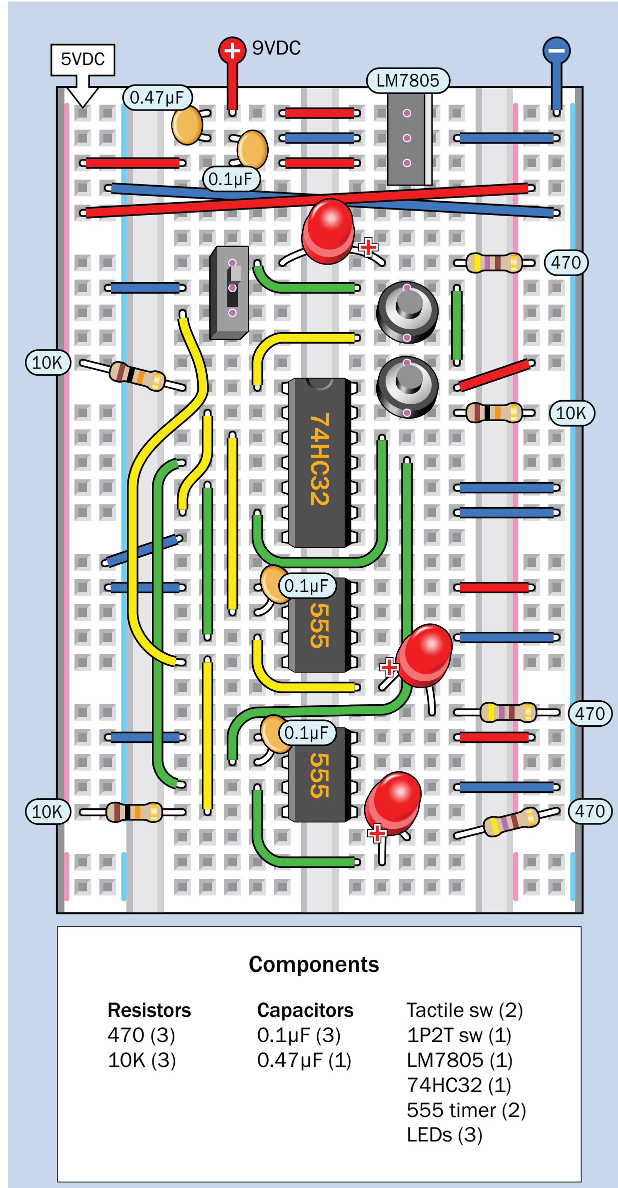


Figure 21-13. Complete breadboard circuit for the Button Blocker.

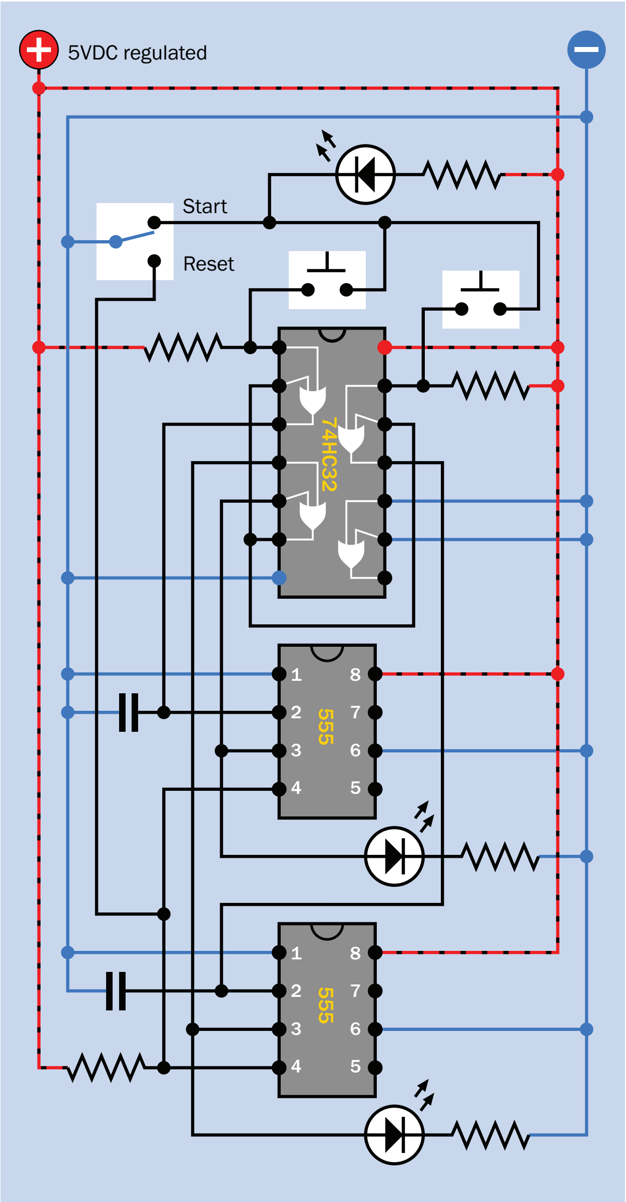


Figure 21-14. Schematic for the Button Blocker circuit.

Experiment 22
Flipping and Bouncing

In three experiments, now, I've used 555 timers in bistable mode. The time has come to deal with "real" flip-flops, including an explanation of how they work. I will also show how they can deal with the phenomenon that I mentioned briefly in the previous experiment: ***contact bounce***.

When a switch is flipped from one position to another, its contacts vibrate very briefly. This is the "bounce" of which I speak, and it can be a problem in circuits where digital components respond so quickly, they interpret every tiny vibration as a separate input. If you connect a pushbutton to the input of a counter chip, for instance, the counter may register ten or more input pulses from a single press of the button. A sample of actual switch bounce is shown in Figure 22-1.

There are many techniques for debouncing a switch, but using a flip-flop is probably the most fundamental.

Figure 22-1. Fluctuations created by vibrating contacts when a switch is closed. (Derived from a datasheet at Maxim Integrated corporation.)

You Will Need:

- Breadboard, hookup wire, wire cutters, wire strippers, multimeter.
- 9V power supply (battery or AC adapter).
- 74HC02 quad 2-input NOR chip (1).
- Optional quad 2-input 74HC00 NAND chip (1).
- SPDT slide switches (1).
- Generic red LED (2).
- Resistors: 1K (2), 10K (2).
- Capacitors: 0.1µF (1), 0.47µF (1).
- LM7805 voltage regulator (1).

Assemble the components on your breadboard, as shown in Figure 22-2. The 74HC02 is a quad 2-input NOR chip, and if you look back at Figure 19-19, you'll see that the NOR gates inside it are upside-down compared with the AND and OR chips which you have been using. You have to be careful about that when wiring it. In Figure 22-3, I've shown them in an x-ray view of the chip.

When you apply power, one of the LEDs at the bottom should be lit. When you move the double-throw switch to its opposite position, the other LED will be lit.

This doesn't seem very interesting, but now try something which may surprise you. Pull the switch out of the board—and whichever LED is on, stays on. Push the switch back into the board, move it to its opposite position to illuminate the other LED, and pull the switch out of the board again—and that LED stays on.

Here's the take-home message:

- A flip-flop requires only an initial input pulse—for example, from a switch.
- After that, it runs itself until it receives a different input.

Debouncing with NORs

The schematic in breadboard format is complicated when you're trying to understand this circuit, so I created a simplified, four-step sequence in figure 22-4 to show how the NORs affect each other. To refresh your memory, I've added a truth table in Figure 22-5 showing the logical outputs from a NOR gate for each combination of inputs.

In Figure 22-4, Step 1, the switch is supplying positive current to the left-hand side of the circuit, overwhelming the negative supply from the pulldown resistor, so we can be sure that the NOR gate on the left has one positive logical input. Because any logic-high input will make the NOR give a logic-low output (as shown in the truth table in Figure 22-5), the negative output crosses over to the

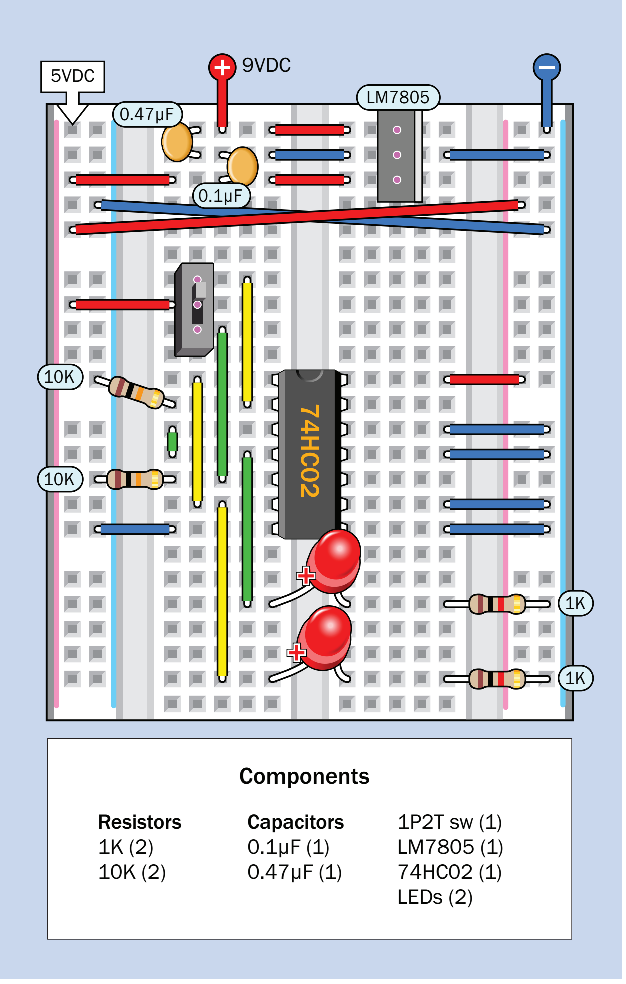


Figure 22-2. Breadboarded flip-flop circuit using NOR gates.

right-hand NOR, so that it now has two negative inputs, which make it give a positive output. This crosses back to the left-hand NOR gate. So, in this configuration everything is stable.

Now comes the clever part. In Step 2, suppose that you move the switch so that it doesn't touch either of its contacts. Or suppose that the switch contacts are bouncing, and failing to make a good contact. Or suppose you disconnect the switch entirely. Without a positive supply from the switch, the left-hand input of the left NOR gate goes from positive to negative, as a result of the pulldown resistor. But the right-hand input of this gate is still positive, and one positive is all it takes to make the NOR maintain its negative output, so nothing changes. In other words, the circuit has "flopped" in this state, regardless of whether the switch is disconnected.

(Where does the positive current come from, in Step 2? From the external power supply to the chip. Remember, logic gates always require a power supply.)

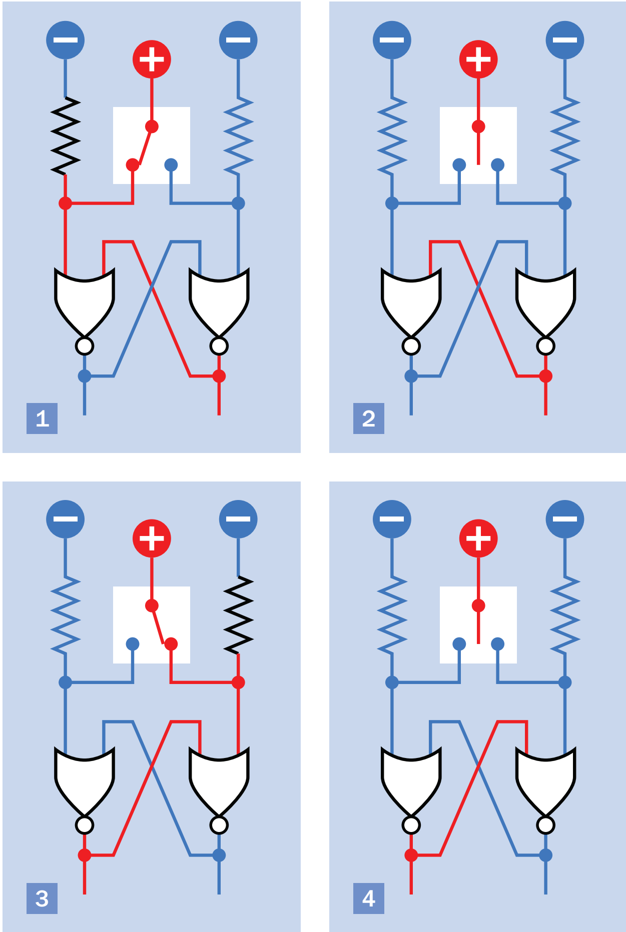


Figure 22-4. Using two NORs to make a flip-flop.

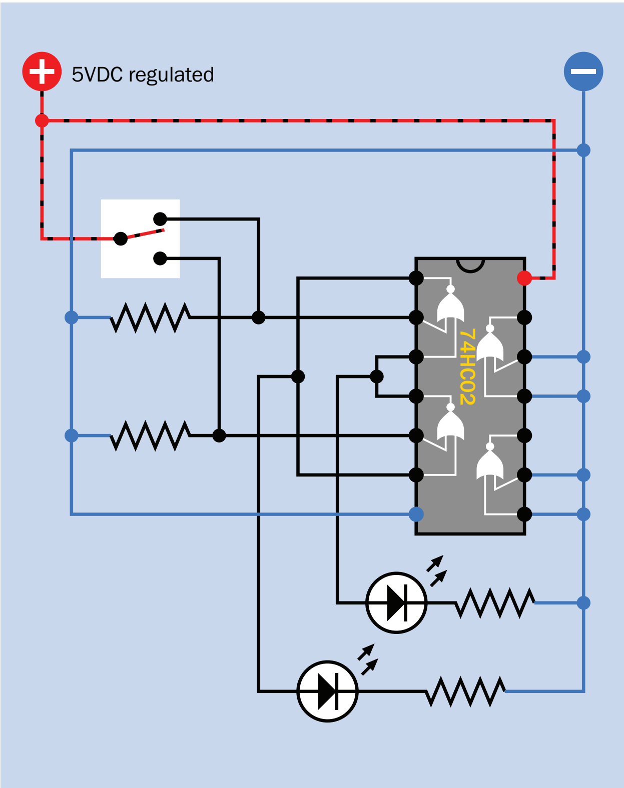


Figure 22-3. Schematic of the flip-flop circuit.

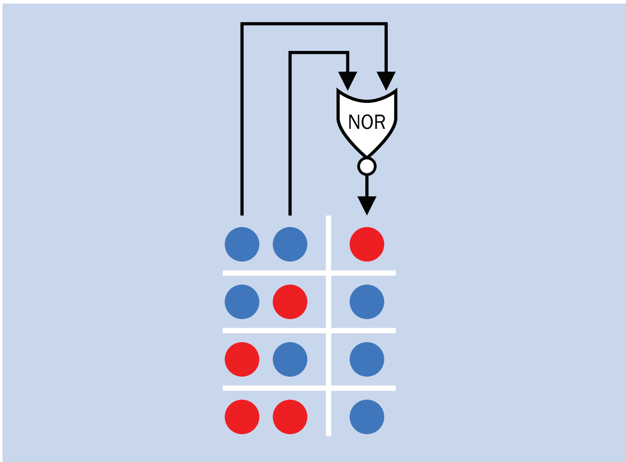


Figure 22-5. A reminder of the truth table for a NOR gate.

Referring to Step 3 in Figure 22-5, if the switch turns fully to the right and supplies positive power to the right-hand pin of the right NOR gate, that NOR recognizes that it now has a positive logical input, so it changes its logical output to negative. That goes across to the other NOR gate, which now has two negative inputs, so its output goes positive, and runs back to the right NOR.

In this way, the output states of the two NOR gates change places. They flip, and then flop there.

What would happen if you applied a positive input to both NORs at the same time? Both their outputs would go low, which would prevent the flip-flop from serving its purpose, which is to store one high output.

This is why I have shown the circuit being triggered by a double-throw switch. Initially, one side of the circuit has to have a logic-high state, while the other must be logic-low.

The introductory books that I have seen don't emphasize the need for a double-throw switch. When I first started learning electronics, I went crazy trying to understand how two NORs could debounce a simple SPST pushbutton—until finally I realized that they can't.

Debouncing with NANDs

The drawings in Figure 22-6 show a similar but opposite sequence of events if you use a negatively powered switch with two NAND gates. To refresh your memory of NAND behavior, I am including its truth table in Figure 22-7.

If you want to verify the function of the NAND circuit, you can use a 74HC00 chip, which I have included as an option in the parts list for this experiment. Be careful, though: the gates inside the NAND chip are not the same way up as in the NOR chip. The two chips are not swappable, so you will have to move some wires around on your breadboard.

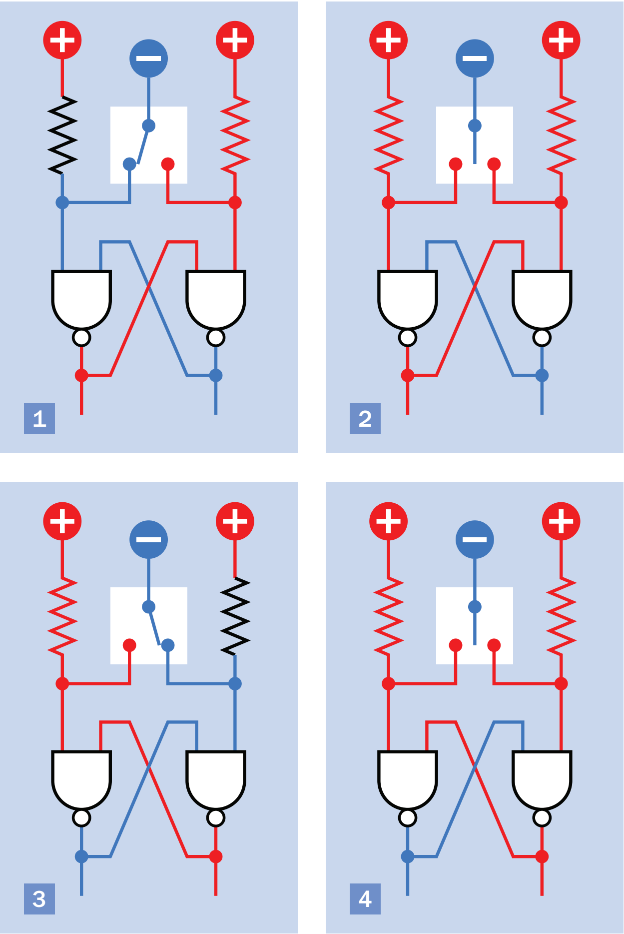


Figure 22-6. Two NAND gates can be used as a flip-flop.

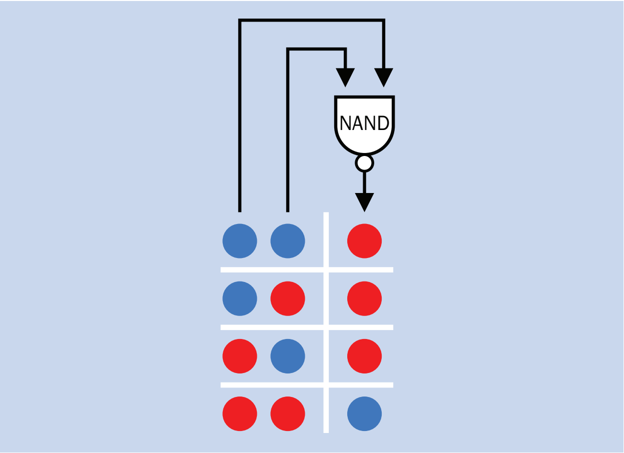


Figure 22-7. A reminder of the truth table for a NAND gate.

Jamming vs. Clocking

The NOR and the NAND circuits are examples of a ***jam-type*** flip-flop, so called because the position of the switch forces it to respond immediately, and jams it into that state. You can use this circuit anytime you need to debounce a switch (as long as it's a double-throw switch).

A more sophisticated version is a ***clocked flip-flop***, which requires you to set the state of each input first and then supply a clock pulse to make the flip-flop respond. The pulse has to be clean and precise, which means that if you supply it from a switch, the switch must be debounced—probably by using another jam-type flip-flop! Considerations of this type have made me reluctant to use clocked flip-flops in this book. They add a layer of complexity that I prefer to avoid in an introductory text. If you want to know more about flip-flops, I explore them in greater detail in *Make: More Electronics*. It's not a simple topic.

What if you want to debounce a single-throw button or switch? Well, you have a problem! One solution is to buy a special-purpose chip such as the 4490 "bounce eliminator," which contains digital delay circuitry. A specific part number is the MC14490 from On Semiconductor. This contains six circuits for six separate inputs, each with an internal pull-up resistor. It's relatively expensive, however—more than 10 times the price of a 74HC02 containing NOR gates.

Of course, you can always use a 555 timer wired in flip-flop mode. This debounces any type of input (from a pushbutton or a switch) because you can set it up with a pullup resistor on Pin 2, and a negative voltage supplied by the switch. The 555 responds to the very first voltage spike, and ignores all the bounces that may follow.

Now you can see why I like this option. It's so easy.

Experiment 23
Nice Dice

Electronic circuits to simulate the throw of one or two dice have been around for decades, but I have an ulterior motive for including one here: It provides an opportunity to explain binary code, the universal language shared by all digital computing devices.

You Will Need:

- Breadboard, hookup wire, wire cutters, wire strippers, multimeter.
- 9V power supply (AC adapter preferred, as this circuit draws more current than a battery may supply for long).
- 555 timer (1).
- 74HC08 quad 2-input AND chip (1).
- 74HC27 triple 3-input NOR chip (1).
- 74HC32 quad 2-input OR chip (1).
- 74HC393 binary counter (1).
- Tactile switches (2).
- Resistors: 470 ohms (6), 1K (4), 10K (2), 100K (1).
- Capacitors: 0.01µF (2), 0.1µF (2), 0.47µF (1), 10µF (1), 100µF (1).
- LM7805 voltage regulator (1).
- Generic red LEDs, 3mm preferred (14).

A Binary Counter

At the heart of every electronic dice circuit is a counter chip of some kind. Often it's a ***decade counter*** with ten "decoded" output pins that are energized one at a time, in sequence. A die only has six faces, but if you tie the seventh pin of the counter back to its reset pin, the counter will restart itself each time it tries to count beyond six. (Incidentally, "dice" is really a plural word that should only be used for two dice or more, while "die" is the singular word, although often it may not be used that way.)

I always like to do things differently, and I want to explain binary code, so I'll be using a binary counter in this circuit. The one that I've chosen is the 74HC393. The pinouts are shown in Figure 23-1, where you'll see that the chip actually contains two counters. I have used the prefix 1: to identify the features of the first counter and 2: to identify the second, although I won't use the second one until I add a second die to the circuit, near the end of this project.

Counter Testing

This chip has a few very simple functions, which you can observe if you bench-test it. Figure 23-2 shows the breadboard layout for this purpose, and Figure 23-3 shows the equivalent schematic.

- Remember, this is a 5V logic chip. Don't leave out the voltage regulator.

The capacitor and resistors that I have specified with the timer will run at about 0.5Hz, so that you can see the output pattern generated by the timer. If you watch carefully, you'll see that the ***end*** of the positive pulse on the LED attached to the timer is the ***start*** of each pulse on the bottom-right LED. This is because the counter chip is ***falling-edge triggered***, meaning that it responds when a logic-high input pulse drops to a low state.

Notice that I have added a 0.1µF capacitor between the power supply and negative ground, up near Pin 8 of the timer. This is to suppress little voltage spikes that the timer tends to generate, which exceed the 5V limit for 74xx chips and can shorten their life expectancy..

Wouldn't it make more sense to put the capacitor on the output of the 555 timer, if that's where the voltage spikes are? Yes, but if you try that, the counter will behave erratically, because the capacitor will also smooth the end of each pulse. Remember, this counter needs a falling edge to trigger it—but when you add a capacitor to the timer output, the falling edge turns into a gradual slope which the counter may not recognize properly. Place the capacitor on the power-supply pin for the timer, and it smooths the surge of current that the timer tries to take to create the spike at the output.

If your connections are correct, the four LEDs will run through the steps numbered from 0 through 15 in Figure 23-4 on the page after next, where a black circle indicates that an LED is not lit, and a red circle indicates that it is lit. The 1s and 0s have been added because a high state is often assigned a value 1 while a low state has a value 0 when a digital chip is doing arithmetic. These 1s and 0s are properly referred to as ***binary digits*** (which is where the abbreviation ***bit*** comes from).

I'm going to tell you some more, now, about binary and decimal arithmetic. Do you really need to know this? Yes, it's useful. Chips such as decoders, encoders, multiplexers, and shift registers use binary arithmetic, and so does the computer which I am using to write this text.

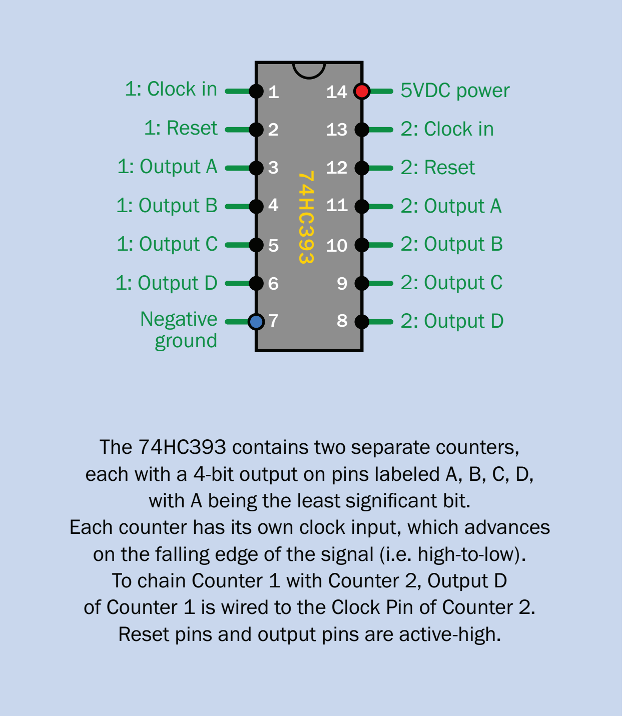


Figure 23-1. Pinouts of the 74HC393 chip. Prefix 1: identifies functions of the first counter in the chip, while 2: identifies the second.

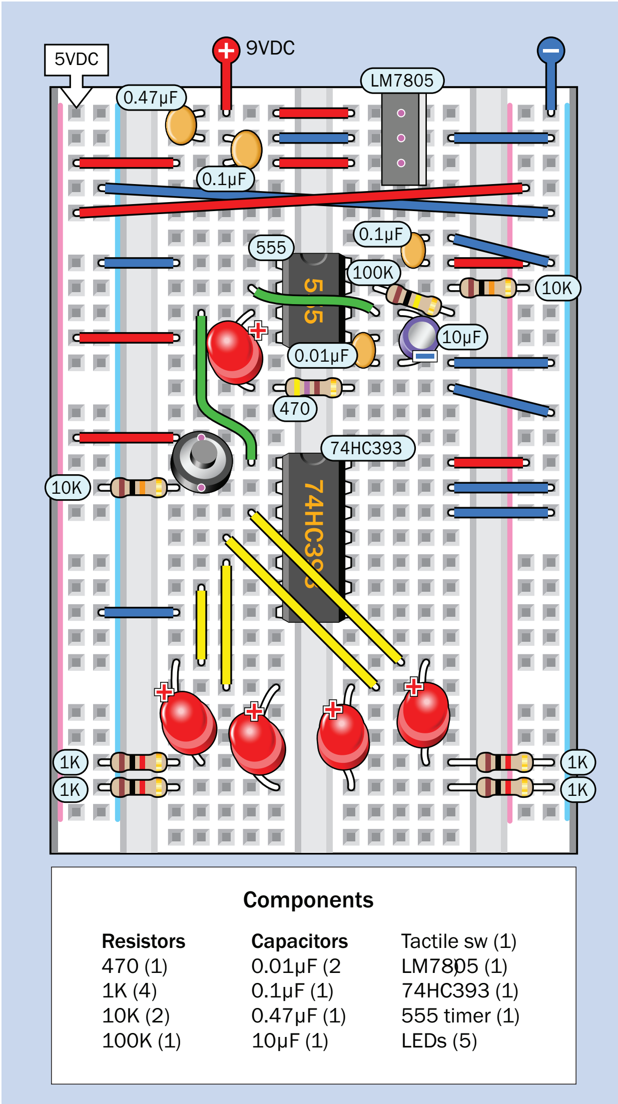


Figure 23-2. Breadboard layout for observing the output and reset function of the 74HC393 binary counter.

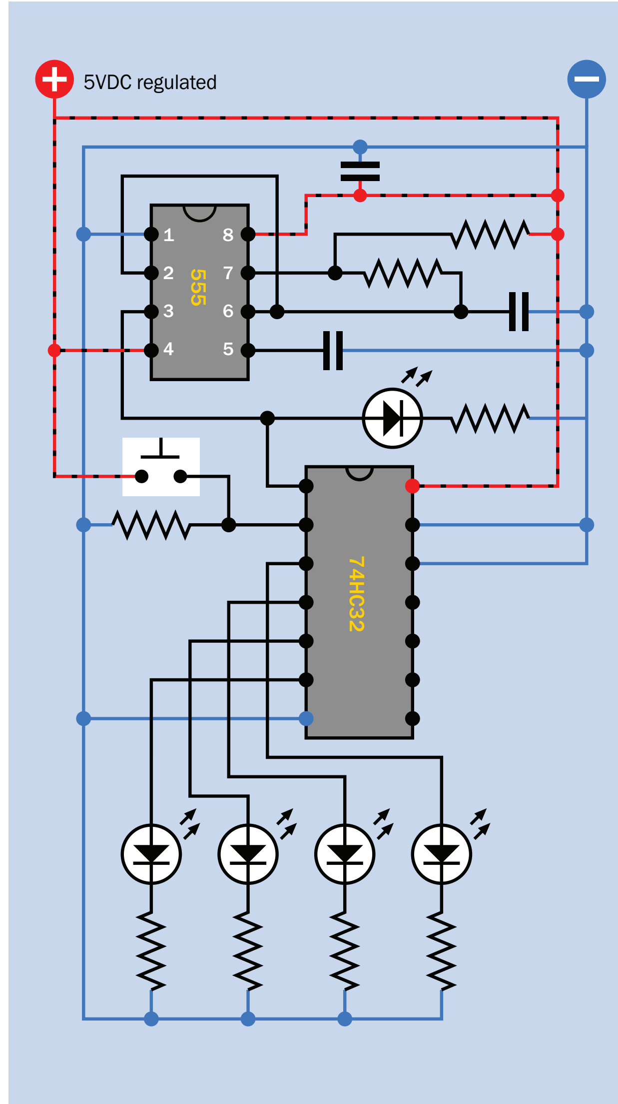


Figure 23-3. Schematic for the counter test circuit.

	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

Figure 23-4. The full sequence of outputs from a four-bit binary counter, reading from top to bottom.

Binary Code

Follow the steps from top to bottom in Figure 23-4, and see if you can figure out the logic. At first, output A changes from `0` to `1`, but when the counter needs to go through another cycle, it has a problem, because there is no number 2 in binary notation. We only have two electrical states: Low and high, representing 0 and 1. So, when the counter takes a step beyond line 1 of the diagram, it shifts `1` to the next column left, and reverts to `0`.

This process repeats indefinitely, and all the columns count in the same way. The rule can be stated like this:

Start with `0`.

Add `1` to `0` to make `1`.

If you add another `1`, move it to the left and go back to `0`. In binary arithmetic, `01 + 01 = 10`.

You can compare this to the way we count in decimal arithmetic: When you are counting upward and you reach `9`, and you want to add `1`, there is no digit higher than `9`, so you place a `1` in the next space to the left, and the rightmost digit goes back to `0`. In other words, in ten-based counting, `09 + 01 = 10`. If only computers had ten different states, they could count in tens like us; but they are not designed that way.

Each row of four digits in Figure 23-4 represents a four-bit *binary number*. The equivalent decimal number is shown in the black font on the left.

What happens after the 74HC393 reaches binary `1111`? It's a four-bit counter, so it rolls over automatically to `0000` and begins all over again.

The rightmost LED in each line represents the *least significant bit* of a four-bit binary number. The leftmost LED is showing the *most significant bit*.

Rising Edge, Falling Edge

I mentioned that the counter is falling-edge triggered: It counts to the next number when the voltage on its Clock Pin falls from logic-high to logic-low. (In Experiment 19, you used a counter that was rising-edge triggered. The type that you use depends on your application, and on which counters are available.

The 74HC393 counter also has a Reset Pin (just like the 4026B chip from Experiment 19).

- Some datasheets describe a reset pin as a "master reset" pin, which may be abbreviated MR.
- Some manufacturers call a "reset" pin a "clear" pin, which may be abbreviated CLR on a datasheet.

Whatever it's called, the reset pin will always have the same end result. It forces all the outputs of the counter to go low—which in this case means `0000` binary.

A reset pin requires a separate pulse. But is it falling-edge triggered, like the Clock Pin? Let's find out.

In Figure 23-3 you can see that Pin 2 of the timer, which is the Reset Pin, is held in a low state through a 10K pull-down resistor. But there is also a tactile switch which can connect the reset pin directly to the positive bus. This overwhelms the 10K resistor, and forces the reset pin to go high.

As soon as you press the tactile switch, all the LEDs go dark, and they stay dark until you let go of the tactile switch. The reset function of the 74HC393 is triggered and held by a rising edge leading to a high state.

The Modulus

Switch off the power, and I'll show you how to make the 74HC393 count to a lower number. I have a feeling that this will be useful in a dice simulation program.

First, disconnect the pullup resistor and the tactile switch from the Reset Pin (which is Pin 2 of the counter).

Now substitute a wire as shown in Figure 23-5, where the green jumper connects pins 2 and 6 of the counter. Figure 23-6 shows the same modification on the schematic.

Run the counter again. It counts from `0000` up to `0111`, as before. The very next binary output should be `1000`, but as soon as the fourth digit transitions from `0` to `1`, the high state is sensed by the reset pin, forcing the counter back to `0000`.

Can you see the leftmost LED flicker, before the counter resets? I doubt it, because the counter responds in less than a millionth of a second.

Because counting from `0000` through `0111` binary is equivalent to counting from `0` to `7` in decimal (that is, through eight steps), we now have a ***divide-by-eight*** counter. Previously, it was a ***divide-by-16*** counter. Bear in mind, the counter starts from `0`, so if it rolls over after it reaches `7` decimal, it's repeating after every 8 states.

Suppose you move the bottom end of your reset jumper from the fourth digit to the third digit. Now you would have a ***divide-by-4*** counter.

- You can easily wire almost any four-bit binary counter so that it resets after two, four, or eight pulses.

The highest number of pulses that a counter reaches before it repeats is known as the ***modulus***, often abbreviated as "mod." A mod-eight counter repeats after eight pulses.

Converting to Modulus 6

So far, so good, but for a dice simulation, wouldn't this counter be more useful if it could count up to six instead of eight?

Yes, definitely, but I had to show you the simple modification before I could get to the more complicated modification which will make it count to six.

In binary code, the first six outputs will look like this: `000`, `001`, `010`, `011`, `100`, `101`. (I can ignore the most significant bit, in column D, because I don't need it for just six states.) I need to make the counter reset after its output of `5` decimal, which is `101` binary.

(It would be more convenient in this project if the counter would start from `1` instead of `0`, but it doesn't do that.)

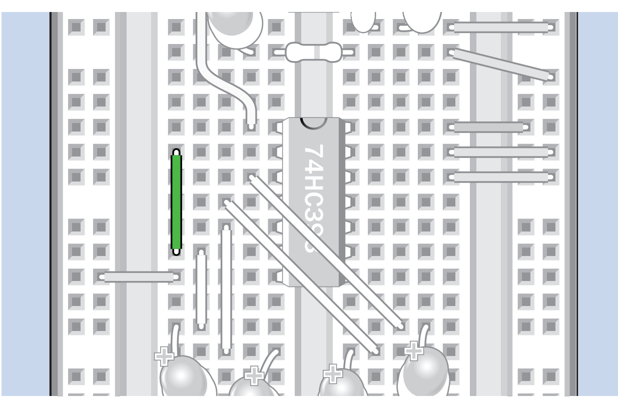


Figure 23-5. The green jumper has been added. The pulldown resistor and pushbutton have been removed.

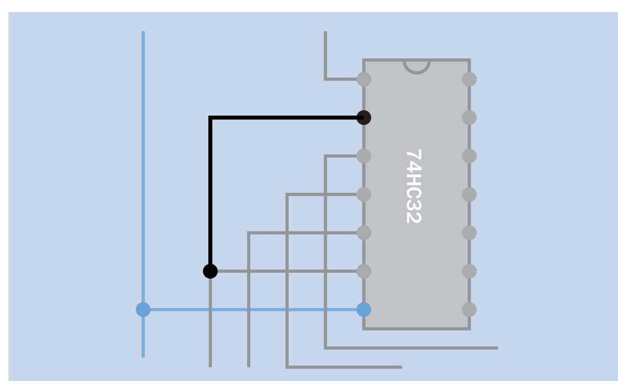


Figure 23-6. The same modification shown on the schematic.

What's the next output after `101` binary? The answer is `110` binary.

Is there something distinctive about `110`, which has not appeared previously in the count sequence? Check Figure 23-4 and you'll see that `0110` is the first in the series where Output C contains a `1` and Output B contains a `1`.

How can we tell the counter, "When you have a `1` from Output C, and a `1` from Output B, reset to `0000`?" The word "and" in the last sentence should give you a clue. An AND gate has a high output when, and only when, its two inputs are high. This is just what I need.

Can I drop an AND chip right in? Absolutely! All the members of the 74HCxx chip family are designed to talk to each other. This is the great pleasure in building logic circuits: You don't need many other components.

In Figure 23-7 you will see the concept of adding an AND gate. To do this on your breadboard, you'll have to add a 74HC08, which you may recall using previously. It contains four AND gates, of which only one is necessary—although actually I will use one more of them when I get to the point of adding the second die to the project.

Meanwhile, please remember this take-home message:

- You can use logic chip(s) with a counter to change the modulus of the counter by looking for a distinctive pattern in the output states, and feeding back a signal to the reset pin.

Not a 7-Segment Display

For the dice display, I could just use a 7-segment numeral that counts from `1` to `6`—but I don't want to do that, because it's not visually appealing. I prefer to use seven LEDs that emulate the pattern of spots on an actual die, in the sequence shown in Figure 23-8.

Now the big question: Is there an easy way to convert the binary output from the counter to illuminate the LEDs in those patterns?

Well, I don't know exactly how easy it is, but there has to be a way, using logic chips, and Figure 23-9 suggests a partial answer. On the left is the sequence of states on outputs A, B, and C as the chip counts upward from `000` to `101`. On the right is the sequence of spot patterns on a die.

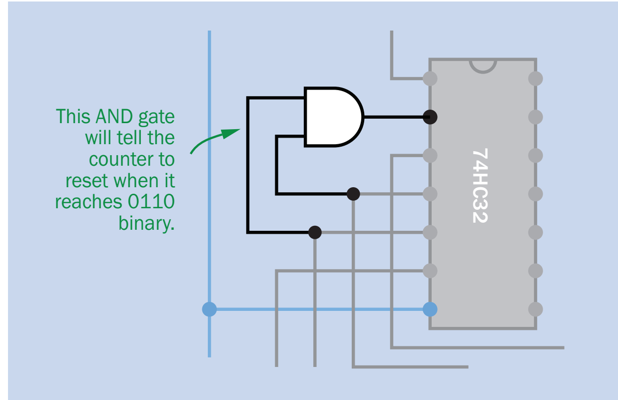


Figure 23-7. An AND gate in this location will make the counter cycle through six output states instead of its usual 16.

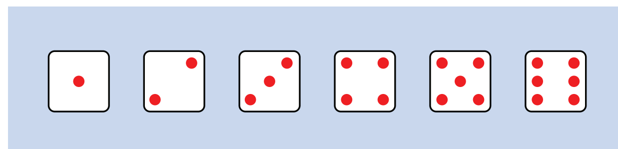


Figure 23-8. The series of spot patterns that must be reproduced by LEDs.

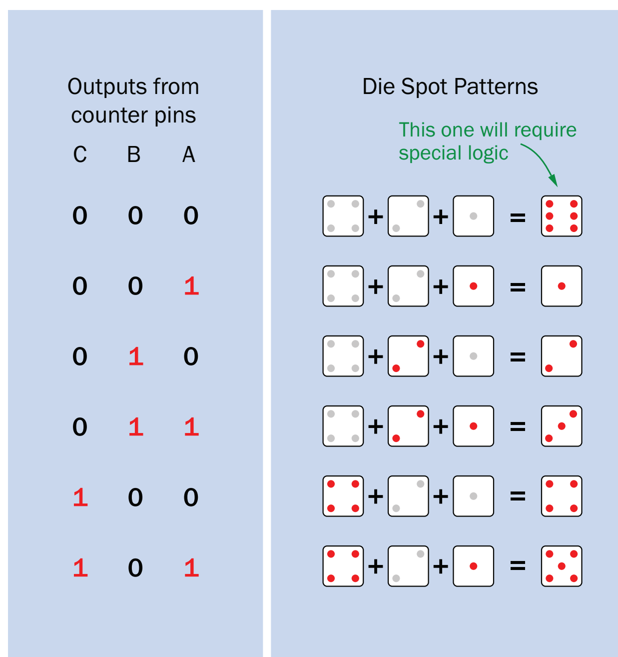


Figure 23-9. Figuring out how to translate counter output to spots on a die.

The first thing you may notice is that if you drive the center LED from Output A, the sequence fits precisely.

Now consider two diagonal LEDs. If you drive them from Output B, that pattern fits too—with the exception of **000**, which is a problem. I will get to that in a moment.

Now suppose you can drive the other two diagonally placed LEDs from Output C, and you can make Output C drive the first two diagonal LEDs as well. That takes care of spot-patterns four and five.

What about the spot patterns for six? The counter always starts from **000**, so I have to convert that to six LEDs somehow. It's a special case which will require special logic. This is annoying, but there's no way around it. I'm going to end up using a 3-input NOR gate to deal with that.

Figure 23-10 shows the complete logic diagram. I'll run through it from top to bottom.

Output A from the timer is simply connected with the center dot of the die. Easy.

Output B goes through OR2 to light two diagonal dots, and those LEDs are connected in series. Really? Can you connect LEDs in series? Yes, if you adjust the resistor for them appropriately. In fact, they are more efficient that way, because a lower-value resistor is sufficient.

But why is OR2 there?

Because the two diagonal dots are lit either by Output B OR by Output C.

Um—okay—but why does Output C connect through another OR, named OR1? Because all four corner dots in the display can be switched on by Output C, OR, in the special case, when the counter is starting out at **000** and needs to display six dots.

The 3-input NOR gate intercepts the **000** output from the counter, lights the two middle LEDs, and also tells OR1 to light two diagonal LEDs. OR1 then tells OR2 to light the other two diagonal LEDs as well, so I get the pattern for six dots.

Just in case this isn't entirely clear, I have created a snapshot sequence showing high and low states in the circuit as the counter increments from **000** through **101**. See figures 23-11, 23-12, and 23-13.

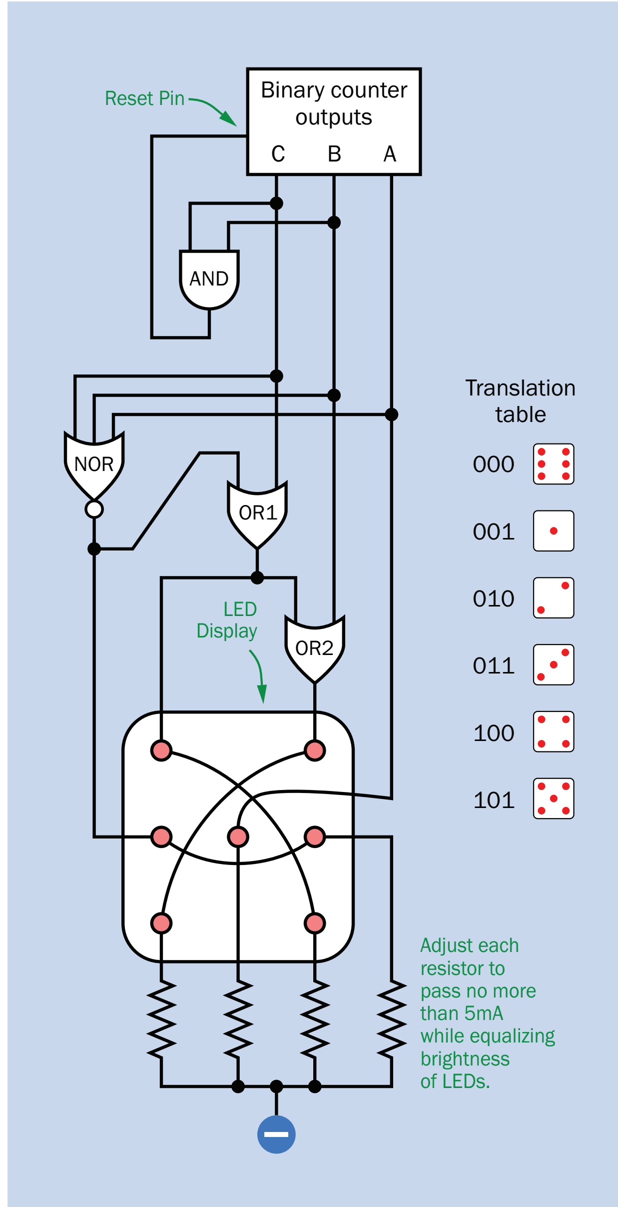


Figure 23-10. Logic diagram showing how counter outputs can be translated into spot patterns for a die.

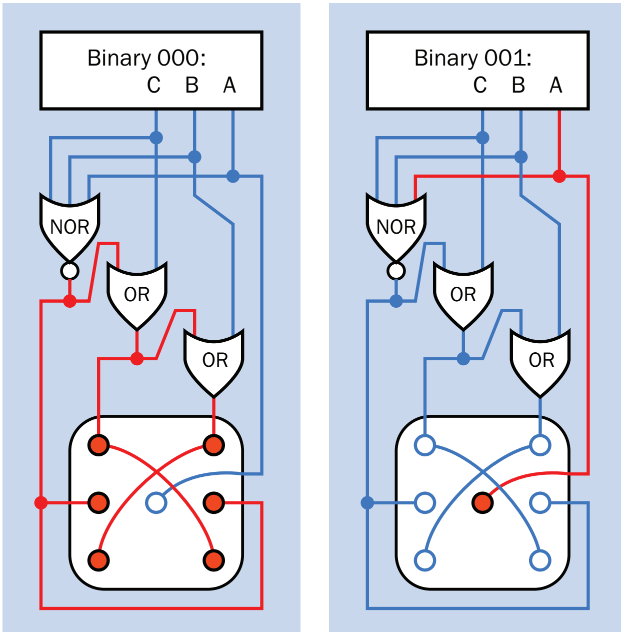


Figure 23-11. Counter outputs 000 and 001 translated into spots.

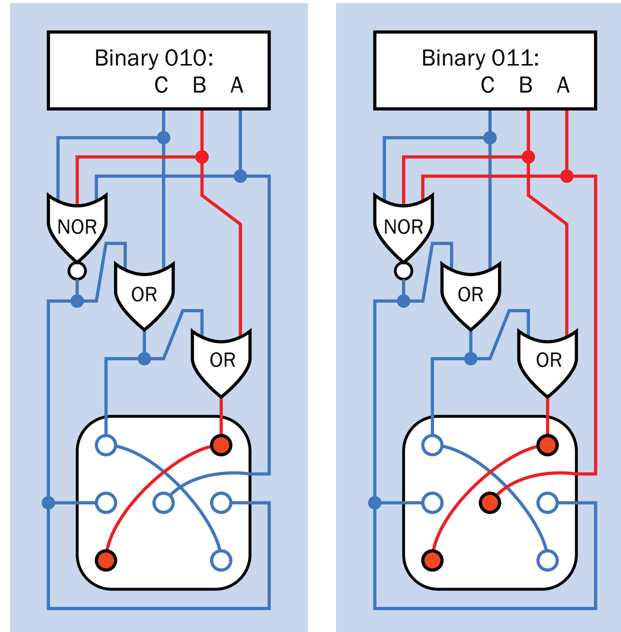


Figure 23-12. Counter outputs 002 and 003 translated into spots.

When people ask me the best way to figure out a pattern of logic gates to solve a problem, I never quite know what to say. For me, there's a certain amount of trial and error, and some intuitive guesses. I try something, trace all the different logic states, and then try something else, and sooner or later I get something that does what I want.

I know that formal systems exist to generate logic diagrams, but in my experience, the results tend use more gates and are less efficient. Therefore, I stick with my intuitive approach. I regret that I am not able to give better advice in this area.

The Second Die

I'm going to supply a schematic to simulate two dice, because the chips to create displays for one die contain enough spare AND, OR, and NOR gates to handle two—and as I already mentioned, the 74HC393 contains two counters. All you need for the second die are seven additional LEDs and a lot of extra hookup wire. The breadboarded circuit is shown in Figure 23-14, and the schematic in Figure 23-15, on the page after next.

Note that you cannot use exactly the same component layout as was shown in Figure 23-2; there are two new pushbuttons, the timing components have different values, and the pins of the 74HC393 are connected differently. You should build the circuit in Figure 23-14 starting from scratch.

This is the most complicated circuit in the book, partly because fitting 14 LEDs onto the breadboard was a challenge. I suggest you use 3mm LEDs rather than the 5mm type that I have been illustrating in most of the projects, but even they will be quite crowded.

You'll have to be extremely careful to place the LEDs correctly. Remember, three pairs of LEDs in each die are wired in series, so the current has to follow a zig-zag path across the breadboard. You may find that it's easier to work from the schematic when you are building this circuit. Either way, I strongly suggest scanning or photographing my diagrams and printing them at a larger size, if you can.

Ideally the current from each logic pin supplying the LEDs should not exceed 5mA, and you should use your meter to check this. The optimum value of the series resistors may vary depending on the LEDs that you use. I found that for each pair of LEDs wired in series, a 470-ohm resistor reduced the current to slightly more than 2mA. You can try a 330-ohm resistor to get closer to my ideal

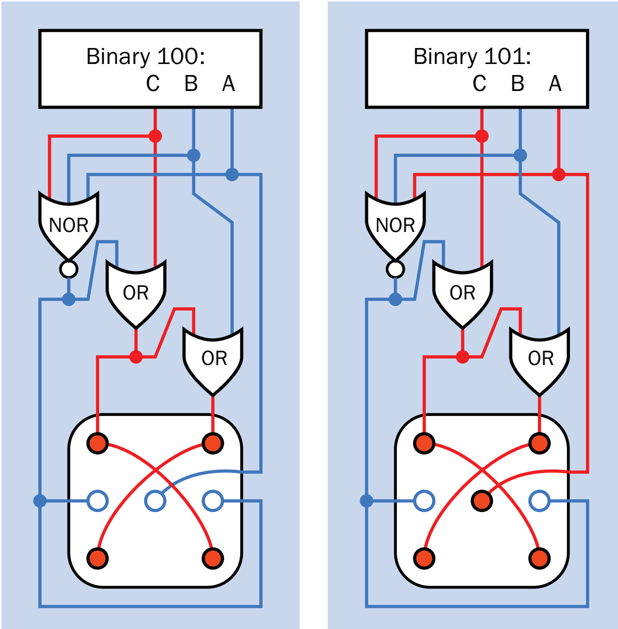


Figure 23-13. Counter outputs 004 and 005 translated into spots.

value of 5mA. For the center LED, which is not wired in series with any others, a 1K resistor maintained the current around 3mA. Even though these are low values, I found that the LEDs were acceptably bright.

I have labeled the three binary outputs from each side of the 74HC393 A, B, and C to help you to compare the circuit with the logic diagram in Figure 23-10.

Randomicity

Now, how does it work?

The most essential aspect of a dice simulation circuit is that it should create random results. The usual method to achieve this is by running a counter very fast, and allowing the player to stop the count at an arbitrary moment. That's exactly how this circuit works.

The right-hand pushbutton starts the timer, which runs very rapidly. You will see the LEDs dimly illuminated, as

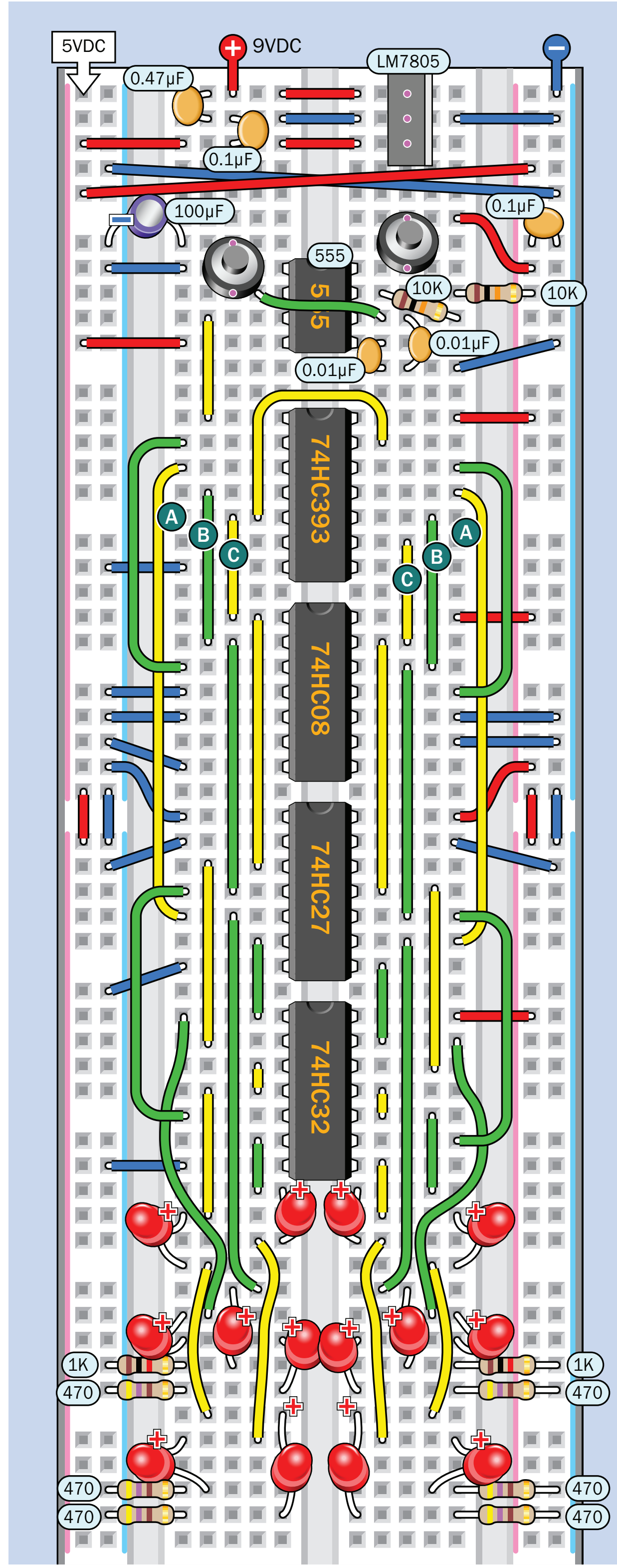
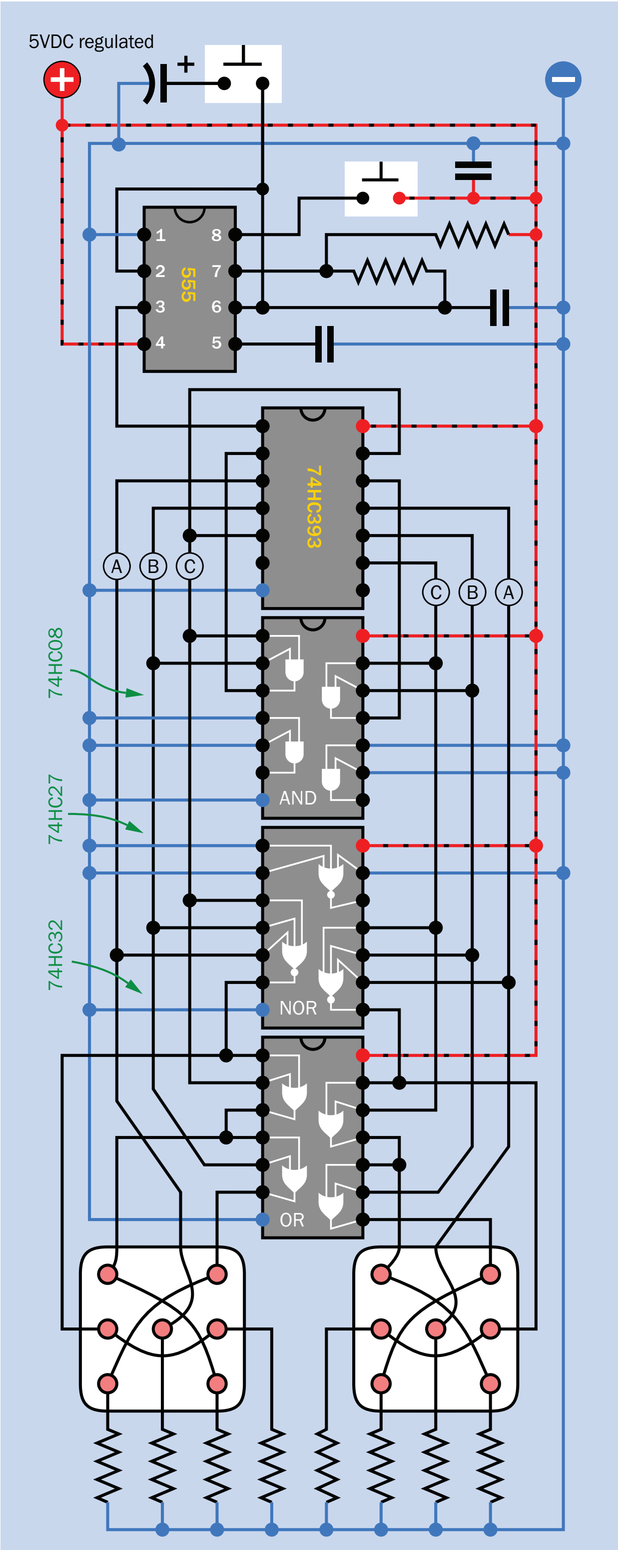


Figure 23-14 (right). Complete breadboard layout for Nice Dice. The LEDs are 3mm in size, to fit the limited space at the bottom of the board. Resistor values may be fine-tuned to equalize the light intensity of the center LEDs with the pairs wired in series.



they flash much faster than the persistence of vision. When you let go of the pushbutton, the timer stops and the counter displays a selection of dice values which will be random—provided each of the 36 possible combinations is generated for an equal length of time.

One way to achieve this would have been to add a second timer and run the two counters in the 74HC393 counter separately. I chose not to do this, for two reasons: First, I didn't have room for a second 555 timer, and second, I was concerned that the two counters would go in and out of phase, picking some dice combinations more often than others.

I chose to have one counter trigger the other. Here's how it works:

The first counter creates a sequence of six binary numbers from **000** through **101**, using the method that I explained previously. When it reaches **110**—just for an instant—Output B and Output C share their high state with Pin 1 and Pin 2 of the 74HC08 chip, containing AND gates. When Pin 1 AND Pin 2 are high, the 74HC08 delivers a high output on Pin 3. You can see a green jumper in the breadboard diagram connecting that pin back to Pin 2 of the counter, which is the Reset Pin. This makes the counter revert from **110** to **000**.

This transition causes Output C of the counter to go from high to low, and this output is connected with Pin 13. You can see this as a yellow jumper running around the top of the chip in Figure 23-14. Pin 13 is the Clock Pin for the second counter, and in this way, the high-low transition on Output C in the first counter clocks the second counter, making it advance from one binary number to the next. In other words, whenever the first counter completes

Components for Nice Dice

Resistors	Capacitors	Chips	Other
470 (6)	0.01µF (2)	555 timer (1)	Tactile sw (2)
1K (2)	0.1µF (2)	74HC393 (1)	LM7805 (1)
10K (2)	0.47µF (1)	74HC08 (1)	LEDs (14)
	47µF (1)	74HC27 (1)	
	100µF (1)	74HC32 (1)	

Figure 23-15 (left). Complete schematic for the Nice Dice game. LEDs are represented by red circles to fit the available space.

Figure 23-16 (above). Complete list of components.

the sequence of six dice values, it triggers the second counter to advance to its next value.

When the second counter reaches its `110` output, this is sensed at another AND gate, via pins 12 and 13 of the 74HC08 chip. The output from that AND goes from Pin 11 of the 74HC08 to Pin 12 of the 74HC393, resetting the second counter to `000`.

In this way, the pair of counters runs through all of the 36 dice combinations, and should allow equal time to each of them.

Just in case anyone doubts this system, I squeezed in an additional button and capacitor, right at the top of the circuit. If you hold this down while also holding down the right-hand button, you'll see the dice patterns counting at around one per second, so that the sequence becomes obvious. The second button simply adds a 100µF capacitor to the timing circuit of the 555 timer. When you don't press this button, the timer runs at about 5kHz.

If you really want to be certain that all numbers are equally weighted in this arrangement, the only way to find out is by using it repeatedly and noting how many times each number comes up. You might need to run it maybe 1,000 times to get decent verification. All I can say is, the result really ***should*** be random.

Enhancements

Could the circuit be simplified? As I mentioned at the beginning, a decade counter would allow simpler logic than a binary counter. You wouldn't need an AND gate to make it count with a modulus of six, because just the seventh output pin on the decade counter could be connected back to the reset.

However, if you wanted to run two dice, you would need two decade counters, and two chips to handle the logic for the two displays. To see why, search online for

```
electronic dice schematic
```

and I think you'll find that as soon as two dice are involved, my circuit is no more complicated than others which don't rely on a binary counter.

Needless to say, I am always open to being proven wrong, and if you come up with a neat dice circuit of your own, I'd love to see it.

The Slowdown Problem

In the first edition of ***Make: Electronics***, I included an extra feature in the Nice Dice circuit. When you took your finger off the "run" button, the die patterns gradually slowed before they stopped. This increased the suspense of waiting to see what the final number would be.

I enabled this feature by splitting the power supply to the 555 timer. The timer was "always on" but voltage to its RC network was shut off when the player stopped pressing the "run" button. At that point, a large capacitor slowly discharged into the RC network, and the timer slowed as the voltage to its timing capacitor diminished.

A reader named Jasmin Patry sent email giving me some bad news. He had found a way to run the circuit repeatedly while logging its results automatically, and the performance was not random. The value **1** came up much more often than other dice values, and he suspected that this had something to do with the slow-down feature.

Jasmin turned out to be a video game designer who understood much more about randomicity than I did. He had the polite, patient style of a man who really knew what he was talking about, and he seemed interested in helping to fix the problem that he had identified.

After he sent me graphs showing the relative frequency of each number in the simulation, I had to agree that the problem existed. I suggested many possible explanations, all of which turned out to be incorrect. In the end Jasmin successfully proved that the lower power consumption of a single LED, relative to the higher power consumption of six LEDs, allowed the timer to keep running for a little longer when the voltage was marginal. This made it more likely to stop while displaying a **1** dice pattern than a **6** dice pattern.

Eventually Jasmin suggested a substitute circuit, in which a second 555 timer was added and the outputs from the two timers were merged through an XOR gate. He successfully proved that this eliminated the bias toward any one number. I was delighted that one of my readers was able to improve the circuit that I had designed, but his circuit wouldn't fit on a single breadboard, which has been my rule for all the projects in the book. In fact, even two breadboards might have been insufficient. So, in the Second Edition, I simply omitted the slowdown capacitor that caused the trouble. I haven't entirely given up on it, though.

Slowdown Alternatives

You'd think there must be a simple way to make the display slow down without affecting the randomicity. When I looked online, I found that someone had used an NPN transistor with its emitter connected to Pin 7 of the timer, and a capacitor between its base and its collector, so that when power was disconnected, the transistor output would gradually diminish. Several other people had done the same thing in their dice circuits, but I suspect that this configuration is subject to the same problems that Jasmin found.

I built a completely different dice circuit using a microcontroller, where it was easy to implement a slowing feature by incrementing a variable in the code—but discovered that this had its own randomicity issues because of the imperfect random-number generator built into the chip.

I developed yet another dice circuit, using a different microcontroller—but here again, I had to depend on the built-in random-number generator, and I am skeptical that it creates an evenly distributed range of numbers.

The simple need for randomicity is really not simple at all. I became so interested in it after my emails with Jasmin Patry, I explored it at length in my book ***Make: More Electronics***, and also wrote a column about it in ***Make*** magazine (volume 45) in collaboration with Aaron Logue, who runs his own little web site describing projects that he builds. He introduced me to the concept of using a reverse-biased transistor to generate random noise, which was then processed with a clever algorithm attributed to the great computer scientist John von Neumann. This, I think, was as close to a perfect random-number generator as you could get—but the chip count was not trivial.

My collaborator Fredrik Jansson suggested fixing my simple slowdown circuit by using two power supplies, one exclusively for the LEDs, so that the current drain caused by different spot patterns would not affect the timer when it was slowed by a discharging capacitor. I think that should work, although I haven't tried it, because I felt that I couldn't expect my readers to buy a second power supply just for this feature.

Another possibility would be to use a Darlington array of transistors to amplify the outputs from the logic gates, so that the drain imposed by the LEDs should be reduced to a negligible value. But—would it really be negligible?

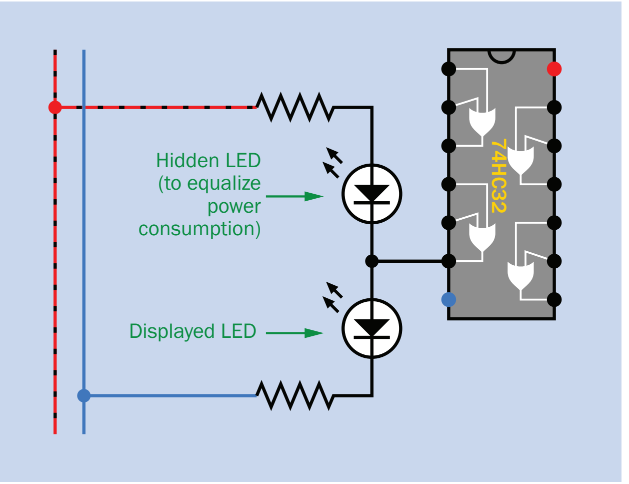


Figure 23-17. To equalize power consumption at all times, an extra hidden LED can sink current into each logic output.

For this edition of the book, I was tempted to try something totally different: Using a rotational encoder to replace the 555 timer as a pulse generator. An encoder contains small switches that open and close as the shaft is turned. I imagined driving it with a small motor, so that when the motor was switched off, it would gradually spin down (especially if it was fitted with a flywheel), causing the switches to generate pulses more slowly before stopping. However, I didn't want to include a project requiring the reader to buy a motor and an encoder just for one project, and I also wondered about the life expectancy of an encoder if the switches inside it were forced to cycle at 1kHz or more. It's possible to buy an optical rotational encoder, but that would be more expensive.

Another reader, Assad Ebrahim, has suggested using vibration sensors in conjunction with a microcontroller, so that the player could shake the assembly like dice in a cup. I had come up with a similar idea a few years ago, featuring neodymium ball magnets rattling in tubes fitted with Hall effect sensors—and I wrote a little piece about this as a "bonus project" for people who register their email addresses with me. It wasn't quite what I wanted, though. I just wanted a circuit that would run itself randomly while gradually slowing until it stopped.

One more reader, Frederick Wilson (who also helped me with fact-checking this book), suggested choosing a random number before the slowdown process began, and storing it temporarily somehow. The slowing displays

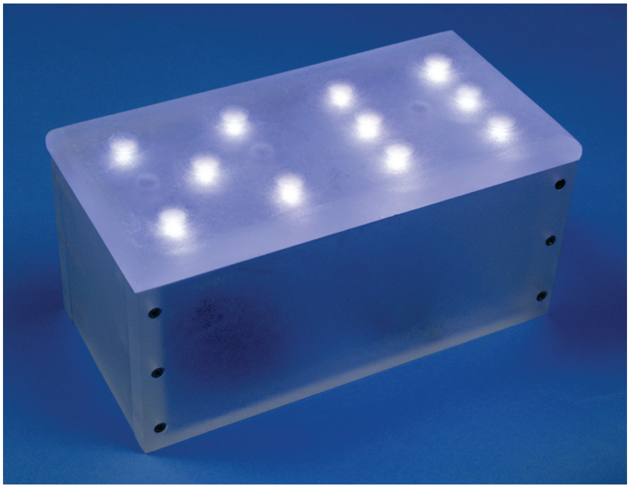


Figure 23-18. This electronic dice display measures about 3" x 6" and is driven by a microcontroller.

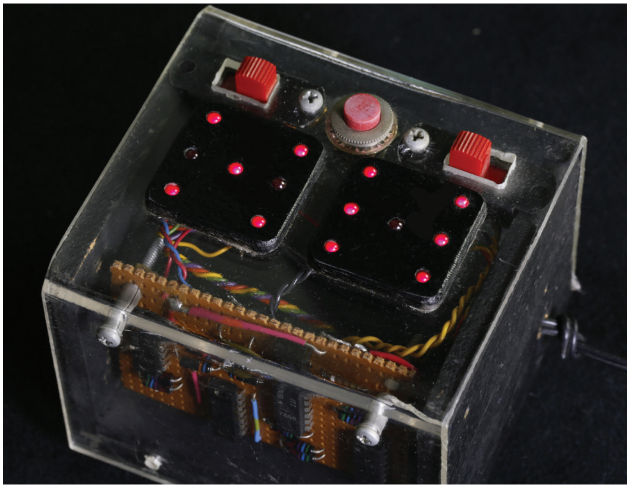


Figure 23-19. Electronic dice designed and built around 1975, in a box of 1/8" Lucite and black-painted plywood.

would then be for cosmetic purposes only. At the end, the stored number would be retrieved and displayed, and the observer would never know the difference. This idea would be easy to try with a microcontroller, but making it work in hardware would have required more components and techniques than I was willing to add to the circuit.

Just before this edition of *Make: Electronics* was ready to be printed, I received an idea from yet another reader, Jolie de Miranda, who came up with a very novel approach: An extra LED could be added to each center LED in each display. The extra LED would be hidden behind a piece of tape, and its only function would be to equalize current consumption with the other spot patterns.

Unfortunately this wouldn't address the greater power drain imposed when multiple LEDs are on. But when I mentioned the concept to Fredrik Jansson, he suggested equalizing the power consumption by adding a hidden duplicate of *every* LED (with series resistors).

A simplified schematic of this concept is suggested in Figure 23-17. Each extra, hidden LED is connected between the positive power supply and a logic-gate output, so that the LED will sink current into the chip whenever the output goes low, while the displayed LED will draw current from the chip whenever the output goes high.

A CMOS chip has outputs that are very symmetrical: They sink current when they are in a low state, just as easily as they source current when they are in a high state, so long as they are not overloaded. With Fredrik's scheme, each logic output would be either sourcing or sinking an equal amount of current, all the time, and the power consumption of each chip should be nearly constant. Of course this is a wasteful solution to the problem, but perhaps a few extra mA are a small price to pay for randomicity.

Meanwhile, I'm including a couple of photographs of finished electronic-dice projects. The one in Figure 23-18 was fabricated from half-inch polycarbonate, sanded to create a translucent effect, with 10mm LEDs embedded in cavities under the lid. The one in Figure 23-19 is an antique which I built with 74LSxx chips, after I learned about them by reading Don Lancaster's amazing *TTL Cookbook*. Almost 50 years later, the LEDs still light up randomly. (At least, I think they're random.)

Section Five

What Next?

At this point, I can branch out in numerous directions. Here are some possibilities:

Audio: This is a large field including hobby projects such as amplifiers and "stomp boxes," to modify guitar sound.

Electromagnetism: This is a topic that I haven't even mentioned yet, but it has some fascinating applications.

Radio-frequency devices: Anything that receives or transmits radio waves, from an ultra-simple AM radio onward.

Microcontrollers: These are tiny single-chip computing devices. You write a little program on your desktop computer, and load it into the chip. The program tells the chip to follow a sequence of procedures, such as receiving input from a sensor, waiting for a fixed period, and sending output to a motor.

I don't have space to develop all of these topics fully, so I'm going to introduce them briefly and leave you to decide which ones interest you. Then you may proceed beyond this book by reading other guides that have a narrower focus.

I'm also going to make some suggestions about setting up a productive work area, reading relevant books, and generally proceeding further into hobby electronics.

Tools, Components, and Supplies

No additional tools are needed for this final section of the book. You will need additional supplies for some of the projects, such as wire for investigating electromagnetism, but I will mention these items at the beginning of each chapter. The complete inventory of components is tabulated in Appendix A.

Storage

If you're getting hooked on the fun of creating circuits, but you haven't allocated a permanent corner for your new hobby, I have some suggestions. Your hobby can be easier and more convenient if you address the issue of storage.

Many electronic components are tiny, and require a system for labeling them, storing them, and finding them when you need them. Tools, on the other hand, may be mid-sized, such as a soldering iron, a heat gun, and a multimeter. You may also need a computer for quick online access to datasheets—and you may use it as an oscilloscope display.

Some parts you are likely to need often, such as resistors and capacitors. Others you may use maybe once a year, and some tools, likewise, are needed less often than others. Any system of organization has to take these requirements into account.

For components, you need a system of categories. Examples might be integrated-circuit chips, relays, sound-reproducing devices (beepers and speakers), breadboards and perforated board, battery carriers, inductors, project boxes, motors, trimmers, small switches, larger switches, and many types of wire. You may not own a lot of these items right now, but in any hobby, you tend to accumulate supplies, and it's sensible to plan ahead.

This leads me to three conclusions:

- You need storage around the workbench, within easy reach.
- Storage under a workbench is nice, too.
- Storage either side of you and behind you wouldn't be such a bad idea, either.

Most work benches don't make much provision for storage under them, so I'll deal with this first. I like a storage system which was perfected for office use all over the world, more than a century ago: The file cabinet.

The drawers of a file cabinet are space-efficient, and they keep out dust. I'm not suggesting that you use them for a lot of paper documents (although I do keep datasheets in printed form). The idea is that each drawer becomes a modular storage system containing plastic bins. These bins are for medium-sized items that you won't need very often, such as project boxes or heavier-gauge wire. The bins for least-frequently-used items go in the bottom of the drawer, while other bins stack on top of them.

Figure 24-1. An Akro-Grid storage box that will fit and stack in a file-cabinet drawer.

For this purpose, the ideal storage boxes measure around eleven inches long and eight inches wide (bearing in mind that a typical drawer is 11.5" wide and 25" long), so that three bins will fit in it sideways without wasting space. A drawer is usually about 10" high, so if your bins are 5" deep, you can stack them two-high.

The ones that I like for this purpose are Akro-Grids, made by Akro-Mils (see figure 24-1). These are very rugged, and are designed to be partitioned. They stack securely, and the bin at the top of the stack can have an optional transparent snap-on lid. You can download the full Akro-Mils catalog online and then search online for retail suppliers.

Another option, ideal for small tools and medium-sized components, is the type of wide, flat box such as the one shown in Figure 24-2, made by Plano. These can fit neatly in file-cabinet drawers if they are stacked on their long edges. You may find something like this at a crafts store, but it may not be as well constructed as the Plano brand. Many of their products are classified as fishing-tackle boxes, and the example in Figure 24-2 is also available with subdivisions.

Figure 24-2. This Plano brand box is useful for storing spools of wire or medium-size tools. When stacked upright on its long edge, three will fit precisely within the width of a file-cabinet drawer.

Maybe you don't like the idea of digging things out of a file cabinet, and certainly I think parts and tools that you use more often are best stored on shelves—so long as everything is protected from dust. This means that open-topped parts bins are not appropriate. What else could be used?

You may be tempted by plastic units about 12" high and 18" wide containing 20 or 30 little plastic drawers, each drawer being maybe 2" wide and 6" deep. These units can hang on the wall, and your resistors and capacitors and semiconductors will be right there, readily available.

The trouble is, in my experience the drawers are either too big (for transistors or LEDs) or too small (for items such as control knobs). And, the storage units are not dust-proof.

Small storage boxes with movable interior dividers always look as if they will be useful, but in my experience, these dividers are not really secure, and if they come loose, parts can intermingle, requiring tedious resorting. After trying many alternatives, I ended up with Darice Mini-Storage boxes, shown in Figure 24-3. You can find these at crafts stores in small quantities, or buy them more economically in bulk online if you search for:

```
darice mini storage box
```

They come in four varieties, for different sizes of parts. The blue boxes are subdivided into five compartments

Figure 24-3. Darice Mini-Storage boxes are good for components such as resistors, capacitors, and semiconductors.

that are exactly the right size and shape for resistors with untrimmed leads. The yellow boxes are subdivided into ten compartments that are ideal for semiconductors and LEDs. The purple boxes aren't divided at all, and the red boxes have a mix of divisions. The lids have durable metal hinges, and they snap closed, so they don't fall open unexpectedly. The lids also have a raised edge, so that the boxes are stackable. The subdivided versions are shown in Figure 24-3.

Now, let's suppose you opt for Darice boxes or something similar. The next question is, where do you put them?

You can array them on shelves, although I prefer another modular system: The little boxes can be stored inside bigger boxes, to keep everything dust-free. After some searching, I found cheap plastic bins, with lids, measuring about 8" x 13" by 5" deep, each being just the right size to hold nine Darice parts boxes, as shown in Figure 24-4. The bins can then go on shelves.

Let's suppose you categorize each bin for a type of component, such as "light emitters." Then each of the little parts boxes can have its own sub-category, such as "high-intensity LEDs." Since each bin can hold nine boxes, I now have an organized system to hold up to 90 different types of LEDs and miniature light bulbs in a way that makes everything easy to find.

Figure 24-4. A plastic bin bought from a hardware store, just the right size to hold nine Darice parts boxes.

Does 90 sound a bit excessive? Well, it could be. Although—I mean—you never know! I certainly wouldn't want to run out of storage space.

To hold the plastic bins, I found steel shelves that could be adjusted to allow about 1/8" clearance. Now I have a wide variety of components within arm's length of my work bench, as shown in Figure 24-5.

Maybe this all sounds a bit obsessive-compulsive. Actually I tend to be disorganized, and have wasted a huge amount of time over the years, digging around for components that were never put away properly. I was like an overweight person who needed to go on a permanent

Figure 24-5. Diligent shopping should enable you to find shelves to hold plastic bins with minimal wasted space.

diet. It was in my own best interests to acquire some self-discipline.

Of course, every system should be modified if it doesn't satisfy real needs on an everyday basis. Experience showed me that the bin-and-box system wasn't ideal for retrieving parts that were very frequently used, such as resistors and capacitors.

Where resistors were concerned, I decided that the key was to trim and bend the leads so they would fit in little cups with screw-tops, measuring about 1-1/2" in diameter. You really don't need long leads on resistors; in all of the projects in this book, I think 3/4" leads have been sufficient. Ceramic capacitors, meanwhile, can fit in smaller, similar screw-top cups measuring 1" in diameter. The two sizes are shown in Figure 24-6.

Figure 24-6. Trimmed resistors will fit in containers about 1-1/2" in diameter, while ceramic capacitors can be stored in similar containers 1" in diameter.

Next I made a couple of vertical storage units as shown in Figure 24-7 (for the capacitors) and Figure 24-8 (for the resistors, on the next page). The circular recesses in the boards were drilled with Forstner bits, which just happened to be exactly the right size for a snug fit, and the storage units can sit on my work bench without taking up too much room. This is my ideal storage system for these components—so far.

Figure 24-7. Desktop storage unit for containers of capacitors.

You may be thinking that you have your own ideas for storage, and they conflict with mine. That's fine, because everyone needs a system reflecting personal preferences—but let me mention just one more factor. Spillage.

At my local crafts store, they sell boxes divided into 15 or 20 little compartments, for people who do beading work. Those boxes look ideal—but they are secured with just

Figure 24-8. Desktop storage unit for containers of resistors.

one lid. If that box falls on the floor with its lid open, the consequences are nightmarish. The stored components will scatter, and they will intermingle. If they happen to be ceramic capacitors, I'm going to have a very bad time scooping them off the floor and resorting them, using a magnifying glass to read their values.

The great advantage of little screw-top cups, for a clumsy person such as myself, is that you can knock everything onto the floor without any serious consequences—so long as you remember to keep the tops screwed on.

The Work Area

When I first started reading books about hobby-electronics, I was surprised to find some of them including fabrication plans for work benches. I really don't think anyone needs to build a special bench just to plug components into a breadboard, or solder them together.

My bare-minimum configuration would be a pair of two-drawer file cabinets with a section of kitchen counter-top placed across them. I like counter-top because it has a splash guard along the back edge, so things can't fall off—at least, not in that direction.

A more ideal area turned out to be an old-fashioned steel office desk—the kind of monster that dates back to the 1950s. They're difficult to move (because of their weight) and don't look beautiful, but you can buy them cheaply from used office furniture dealers, they're generous in size, they withstand abuse, and they last forever. The drawers are deep and usually slide in and out smoothly, like good file-cabinet drawers. Best of all, the desk has so much steel in it, you can use it to ground yourself before touching components that are sensitive to static electricity.

Figure 24-9. A benchtop power supply.

The real question is what tools and supplies you decide to keep on the work surface. Fortunately you don't need a large area when building electronics projects, so you can array frequently-used items around it.

A ***power supply*** for your projects will deliver smoothed current at a range of regulated and calibrated voltages which remain almost constant, regardless of how much current you are drawing. Your little wall-plug AC adapter cannot do this. On the other hand, as you've seen, a cheap 9V AC adapter is adequate for basic experiments—and

Figure 24-10. The Picoscope 2204A plugs into the USB port on a computer.

when you're working with logic chips, you need to mount a 5-volt regulator on your breadboard anyway. Bearing these factors in mind, I consider a good power supply desirable but optional. A relatively low-priced example is shown in Figure 24-9.

Another optional item is an ***oscilloscope***. This will show you, in realtime, the electrical fluctuations inside your wires and components, and will resolve mysteries that may remain baffling otherwise. To take one example, I learned a lot more about displacement current by studying it myself with an oscilloscope than I ever learned by reading about it.

Scopes are much more affordable now than when I started getting interested in electronics many years ago. In fact the Picoscope 2204A which I own costs one-tenth as much as the big, old oscilloscope with a cathode-ray tube which I bought back in 1999. The Picoscope also has better performance, higher resolution, and its electronics are contained a little box about 3" x 5", as shown in Figure 24-10. This plugs into the USB port of any computer, and the computer then does the work of displaying a trace on the screen. A cheap second-hand laptop and a Picoscope will give you performance that used to cost thousands of dollars, and the images are easily captured and saved onto your hard drive.

Pliers, wire strippers, wire cutters, and a magnifying lens of some kind should probably be accommodated on your work bench, along with a notebook to log your adventures into electronics. Figure 24-11 shows a storage solution I devised when I decided I really didn't like keeping gripping tools and cutting tools inaccessibly.

On top of my work area I have a two-foot square of half-inch plywood, not so much to protect the bench, but because I can clamp a miniature vise to one edge of the square. This is stable, yet I can turn the plywood if I need to access a project from a different angle. I also use a more modern version of Helping Hands.

In the past I used to cover the plywood with conductive foam, to reduce the risk of static discharge when working with sensitive components. Over the years, though, I realized that my particular combination of carpet, chair, shoes, and perspiration does not cause me to suffer from static, even though I live in an area of very low humidity.

Figure 24-11. A 7" piece of two-by-four is all you need to store pliers and wire cutters accessibly.

Static is an issue that you have to determine by experience. If you ever feel a little zap when you walk across the room and touch a metal door knob, you need to consider grounding yourself or using anti-static foam (or a piece of metal) whenever you build circuits using CMOS chips.

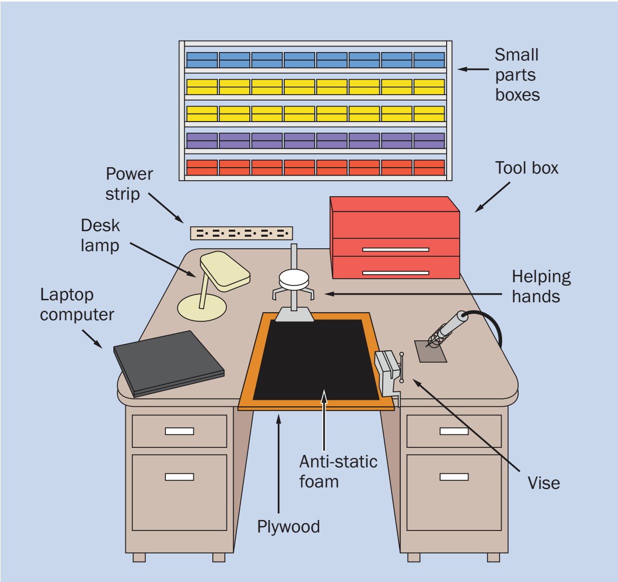


Figure 24-12. An old steel office desk can be as good as, if not better than, a conventional work bench when building small electronics projects.

Inevitably, during your work you'll create a mess. Little pieces of bent wire, stray screws, and fasteners can cause short circuits if they get into a project that you're building, so you need a trash container that has to be large enough to encourage you to use it.

A possible work bench configuration, using a steel desk, is shown in Figure 24-12. A more space-efficient configuration is suggested in Figure 24-13.

Online Sources

As you proceed further into electronics, what are some good web sites which will help you to learn more? This is a difficult question, because a lot of online sources provide incomplete coverage, and some are not entirely reliable. I like *Nuts and Volts* magazine, which has some excellent tutorials in its archives:

`www.nutsvolts.com/`

I think *Instructables* does a good job of explaining basic concepts:

`https://www.instructables.com/Basic-Electronics/`

Figure 24-13. For maximum utilization of available space, and to protect yourself from unwelcome intrusions, consider walling yourself in.

My publisher, *Make Community*, always comes up with a range of interesting projects, some of them using electronic components:

`https://make.co/`

I like some of the products that are sold online by *Adafruit*:

`www.adafruit.com/`

If you're hoping for the kind of old-school web site or blog that is run by a lone enthusiast, one of the last to survive is Don Lancaster's *Guru's Lair*:

`www.tinaja.com`

Lancaster's *TTL Cookbook* opened up electronics to at least two generations of hobbyists and experimenters. He knows what he's talking about, and isn't afraid of getting into some fairly ambitious areas such as writing his own PostScript printer drivers and creating his own serial-port connections. You'll find a lot of ideas on his site, while it lasts.

Here's a list of miscellaneous sites which can give you ideas for fun projects and ways to build them.

Mathematical Science and Technologies offers links to all kinds of products:

`http://mathscitech.org/articles/electronics#inspiration`

Hackaday is a mix of news and ideas:

`https://hackaday.com/`

Here are 30 fully documented Arduino projects:

`https://howtomechatronics.com/arduino-projects/`

The Turtlebot home page:

`https://www.turtlebot.com/`

Dozens more interesting Arduino projects:

`https://trybotics.com/project`

DF Robot:

`www.dfrobot.com/blog`

Garage projects from students at Carnegie-Mellon University:

`https://www.build18.org/garage/`

Techniques for soldering surface-mount components:

`https://schmartboard.com/`

Here's a breadboardable 8-bit computer (note, it requires more than one breadboard):

`https://eater.net/8bit/`

Some affordable gadgets:

`www.icstation.com/index.php`

DIY electronic music projects:

`https://diyelectromusic.wordpress.com/`

Naturally I can't guarantee how many of these sites will still be up and running by the time you read this.

Books

Do you need other books, in addition to this one? Well, I certainly hope you may consider buying some of mine! *Make: More Electronics* is a hands-on book which picks up where this one leaves off, and *Encyclopedia of Electronic Components* is a reference source. The Encyclopedia is in three volumes, some of the text written in collaboration with Fredrik Jansson.

Practical Electronics for Inventors by Paul Scherz is the book that I always recommend if you want a solid general reference volume. Despite its title, you will find it useful even if you never invent anything.

CMOS Sourcebook by Newton C. Braga is entirely devoted to the 4000 series of CMOS chips, not the 74HCxx series that I've used primarily in this book. The 4000 series is older and a little more vulnerable to static electricity, but the chips remain widely available, and will tolerate a wide voltage range, typically from 5V to 15V. This means you can set up a 9V circuit that drives a 555 timer, and use the output from the timer to go straight into CMOS chips—or vice-versa.

The Encyclopedia of Electronic Circuits by Rudolf F. Graf is a totally miscellaneous collection of schematics, grouped by function, with minimal explanations. This is a useful book to have around if you have an idea and want to see how someone else approached the problem. Many additional volumes in the series have been published, but start with this one, and you may find it has everything you need.

The Circuit Designer's Companion by Tim Williams contains much useful information about making things work in practical applications, but the style is dry and fairly technical. This may be useful if you've learned all the basics and are now interested in moving your electronics projects into the real world.

Any book by Forrest Mims will communicate on a level that is not too advanced, and will contain ideas for electronics projects. His slim volume *Getting Started in Electronics* is still fun. I have covered many of its topics here, but you may benefit by reading explanations and advice from a different source, and it goes a little farther than I have into some electrical theory, on an easy-to-understand basis, with cute drawings.

Experiment 24
Magnetism

Now that I have surveyed your future options, let me deal with a very important topic that has been waiting in the background: The relationship between electricity and magnetism. Quickly this will lead me into audio reproduction and radio, and I'll describe the fundamentals of self-inductance, which is the third and final basic property of passive components (resistance and capacitance being the other two). I left self-inductance until last because it has limited application to DC circuits; but as soon as you start dealing with analog signals that fluctuate, it becomes fundamental.

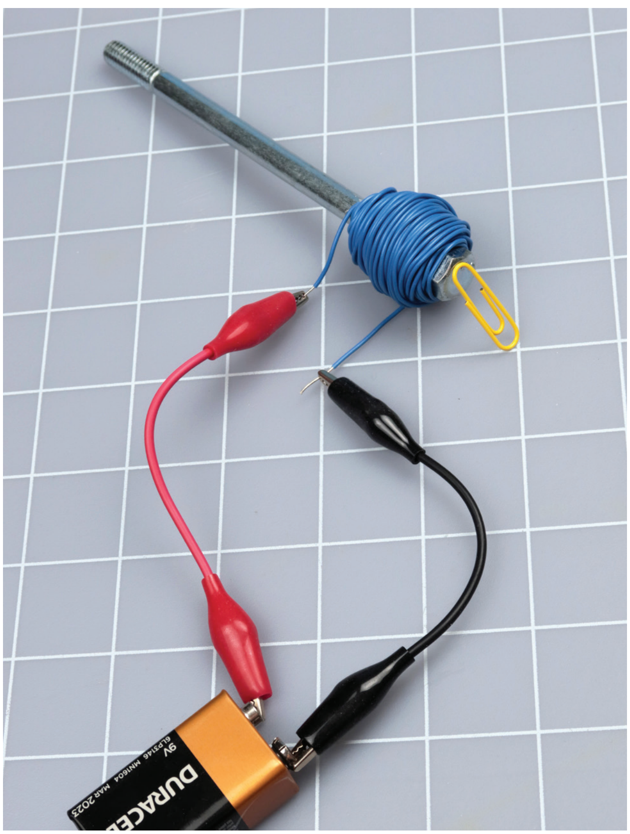


Figure 24-14. This most basic electromagnet using a steel bolt is just strong enough to pick up a paperclip.

You Will Need:

- Large screwdriver or steel bolt (1).
- 22-gauge wire (or thinner), 25 feet.
- 9V battery (1).
- Paperclip (1).

Procedure

This couldn't be much simpler. Wind about 25 feet of 22-gauge wire around the shaft of a screwdriver or some other steel object, such as a bolt. Don't use an object made of stainless steel; it tends to be nonmagnetic.

The turns should be neat and tight and closely spaced, and you'll need to make at least 100 of them, within a distance of less than 2". To fit the turns into this space, you'll have to layer them. If the final turn tends to unwind itself, secure it with a piece of tape.

Now apply a 9V battery. The coil will draw between 1A and 2A, so you should only connect it for a few seconds.

A coil of this number of turns should certainly generate enough magnetic attraction to lift a paperclip off the table.

Congratulations: you just made an electromagnet. The schematic for this circuit is shown in Figure 24-15.

A Two-Way Relationship

Electricity can create magnetism: When electric current flows through a wire, the electricity creates a magnetic field around the wire. This principle is used in almost every electric motor in the world.

Conversely, magnetism can create electricity: When a wire moves through a magnetic field, the field creates electromotive force in the wire, which causes current to flow, assuming that the wire forms a closed circuit.

This principle is used in power generation. A diesel engine, or a water-powered turbine, or a wind turbine, or some other source of energy can move coils of wire through a powerful magnetic field, which induces electricity in the coils.

With the exception of solar panels and batteries, all sources of electric power use magnets and coils of wire.

In Experiment 25, you'll be able to build a handheld power generator that can energize an LED. First, however, I have to provide a bit of theory.

Inductance

Because electricity ***induces*** a magnetic field, this effect is known as ***inductance***, which measures the power of a conductor to induce the field.

Even a straight piece of wire acquires a very weak magnetic field around it, as illustrated in Figure 24-16, where you have to imagine conventional current flowing from left to right.

If you bend the wire into a circle, the magnetic force starts to accumulate, pointing through the center of the circle, as shown in Figure 24-17. If you add more circles of wire to form a coil, the force accumulates even more. And if you put a steel or iron object in the center of the coil, it channels the magnetic field, and the effectiveness increases further.

In Figure 24-18, on the next page, you can see this graphically, along with a formula known as "Wheeler's approximation," which cannot be precise as it depends on the exact type of wire used. Still, it allows you to get an estimate of the inductance of a coil of wire, assuming you know the inner radius, the outer radius, the width, and the number of turns. (The dimensions must be in inches, not metric.) The basic unit of inductance is the Henry,

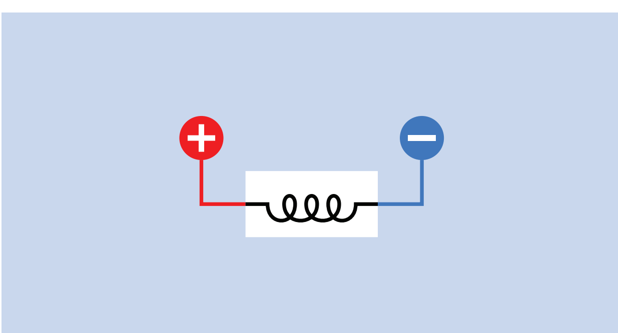


Figure 24-15. A schematic can't get much simpler than this.

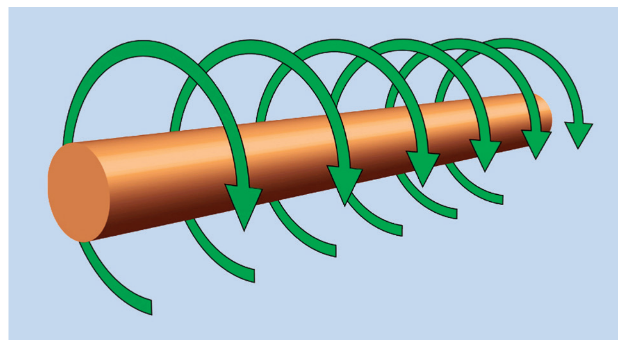


Figure 24-16. When the flow of electricity is from left to right along this conductor, it induces a magnetic force shown by the green arrows.

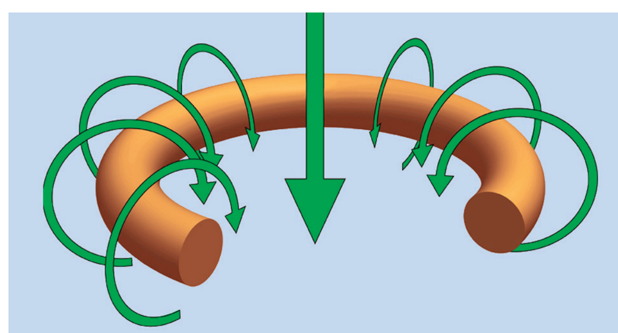


Figure 24-17. When the conductor is bent to form a circle, the cumulative magnetic force acts through the center of the circle, as shown by the large arrow.

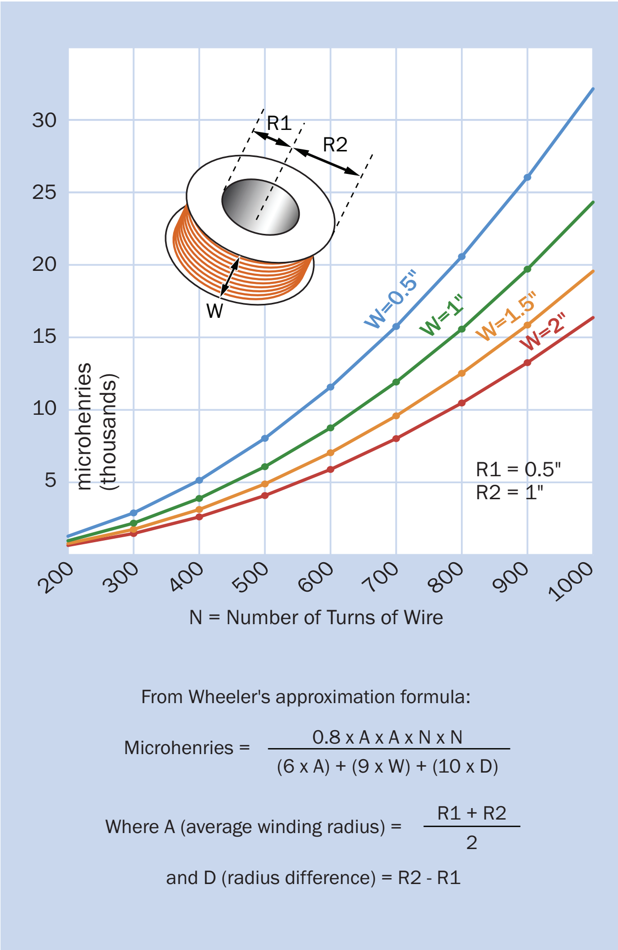


Figure 24-18 A graphical representation showing how the dimensions and number of turns in a coil affect its inductance, calculated approximately.

named after an American electrical pioneer named Joseph Henry. Because this is a large unit (like the Farad), the formula in Figure 24-18 expresses inductance in microhenries. As you might guess, there are 1,000,000 microhenries in a henry.

You'll see from the graphs that if you keep the basic size of a coil the same, and double the number of turns (maybe using thinner wire or wire with thinner insulation), the inductance of the coil increases by a factor of four. This is because the formula includes the factor N * N at the top. Here are some take-home messages:

- Inductance increases with the diameter of the coil.
- Inductance increases with the square of the number of turns. (In other words, three times as many turns create nine times the inductance.)
- If the number of turns remains the same, inductance is lower if you wind the coil to be slender and long, but is greater if you wind it so that it's fat and short.

Coil Schematics and Basics

Check the schematic symbols for coils in Figure 24-19. Either of the top two symbols represents a coil with an air core (the second symbol is older than the first). The lower-left symbol is for a coil wound around a solid iron core, while the lower-right symbol indicates a core composed of iron particles or ferrite.

Assuming you apply a positive DC power source at one end of a coil and negative ground at the other end, the magnetic field will reverse if you reverse the polarity of the power supply. You won't notice any difference, if you're picking up a paperclip, because the magnetic field actually works by inducing an opposite magnetic polarity in the object.

Perhaps the most widespread application of coils is in transformers, where alternating current in one coil induces an alternating magnetic field, which then induces alternating current in another coil. If the ***primary*** (input) coil has half as many turns as the ***secondary*** (output) coil, the voltage will be doubled, at half the current—assuming hypothetically that the transformer is 100% efficient.

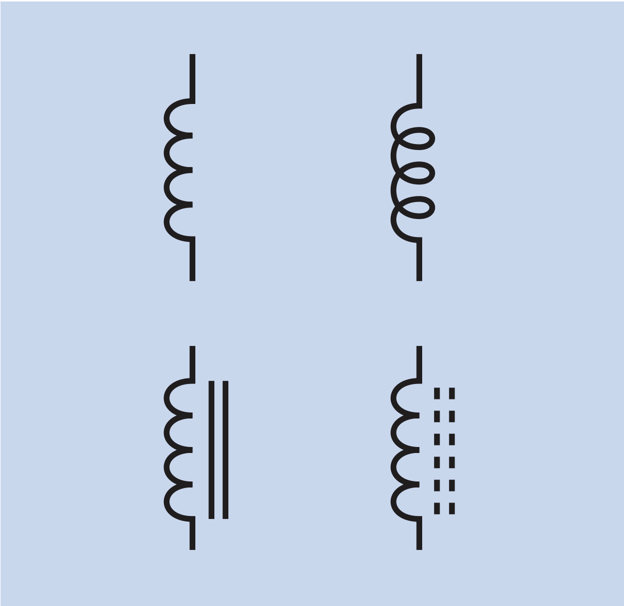


Figure 24-19. Schematic symbols to represent coils. See text for details.

Figure 24-20. Joseph Henry was an American experimenter who pioneered the investigation of electromagnetism.

Joseph Henry

Born in 1797, Joseph Henry was the first person to develop and demonstrate powerful electromagnets. He also originated the concept of "self-inductance," meaning a kind of "electrical inertia" which a coil possesses, when it resists a pulse of current.

Henry started out as the son of a day laborer in Albany, New York. He worked in a general store before being apprenticed to a watchmaker, and was interested in becoming an actor. Friends persuaded him to enroll at the Albany Academy, where he turned out to have an aptitude for science.

In 1826, he was appointed Professor of Mathematics and Natural Philosophy at the Academy, even though he was not a college graduate and described himself as being "principally self-educated." Michael Faraday was doing similar work in England, but Henry was unaware of it.

Henry was appointed to Princeton in 1832, where he received $1,000 per year and a free house. When Morse attempted to patent the telegraph, Henry testified that he was already aware of its concept, and indeed had rigged a system on similar principles to signal his wife, at home, when he was working in his laboratory at the Philosophical Hall.

Henry taught chemistry, astronomy, and architecture, in addition to physical science, and because science was not divided into strict specialties as it is now, he investigated phenomena such as phosphorescence, sound, capillary action, and ballistics. In 1846, he headed the newly founded Smithsonian Institution as its secretary.

Experiment 25
Tabletop Power Generation

In Experiment 24, you used electricity to generate a magnetic field. Now it's time to see how magnetism can generate electricity.

You Will Need:

- Wire cutters, wire strippers, test leads, multimeter.
- Cylindrical neodymium magnet, 3/16" diameter by 1.5" long, axially magnetized (1).
- Hookup wire, 26-gauge, 24-gauge, or 22-gauge, total 200 feet.
- LED (1).
- Capacitor, 1,000µF (1).
- Small signal diode, 1N4148 or similar (1).

Optional extras:

- Cylindrical neodymium magnet, 3/4" diameter by 1" long, axially magnetized (1).
- Half-inch diameter wooden dowel, 6" long (minimum).
- Steel screw, #6 size with flat head
- PVC water pipe, 3/4" internal diameter, 6" long (minimum).
- Two pieces of 1/4" plywood, each about 4" x 4". You will need a 1" hole saw or Forstner bit to drill a hole through the plywood. Corrugated cardboard may be strong enough instead; see text for details.
- Spool of magnet wire, quarter-pound, 26 gauge, at least 350 feet (1).

Procedure

First, you need a magnet. Neodymium magnets are the strongest available, and are not too expensive if you choose the small one specified in the parts list. Wrap about 10 turns of 22-gauge wire tightly around it, as shown in Figure 25-1. Now allow the wire to loosen slightly, so that the magnet can slide through the coils.

Set your meter to measure millivolts AC (not DC, because you're going to be dealing with alternating pulses of electricity). Strip a little insulation from each end of the coil, and use alligator test leads to attach the meter. Grasp the magnet between finger and thumb, and shuttle it quickly to and fro inside the coil. I'm guessing you'll see a value of 3mV to 5mV on your meter. Yes, this small magnet, and ten turns of wire, can generate a few millivolts.

If you're wondering if the magnet loses some of its power as a result of generating electricity—no, it doesn't. You are delivering some energy into the magnet, by moving it. The magnet passes the energy along to the wire and the meter.

Try winding a bigger coil, with the layers overlapping, as shown in Figure 25-2. Move the magnet quickly again. You should find that you are generating more voltage.

- In the previous experiment, I provided a formula that showed how passing electricity through more coils of wire would induce a stronger magnetic field. It works both ways:
- More coils of wire will generally induce a higher voltage, when a magnet moves through the coils.

This leads me to wonder—if we had a bigger, stronger magnet and a *lot* of turns of wire, could we generate enough electricity to power something, such as, maybe, an LED? I'm describing the next part of this experiment as "optional," as it will require extra supplies.

Powering an LED

Generally speaking, a coil is more powerful when it contains more turns of wire within the available space. A product known as ***magnet wire*** is optimized for this purpose, being coated with plastic or shellac insulation that is ultra-thin, so that the windings in a coil can be packed together as densely as possible.

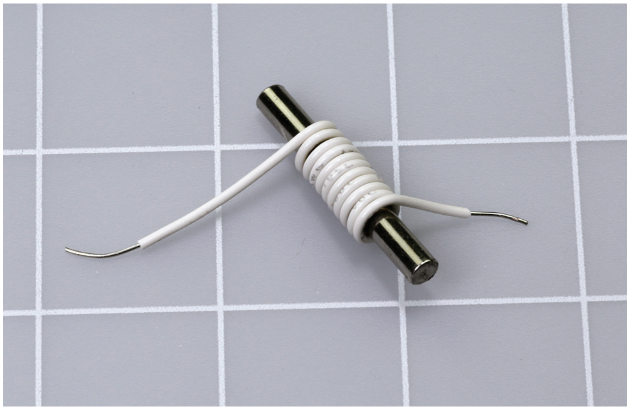


Figure 25-1. Just ten turns of wire can be sufficient to create a small electrical potential when a magnet moves through them.

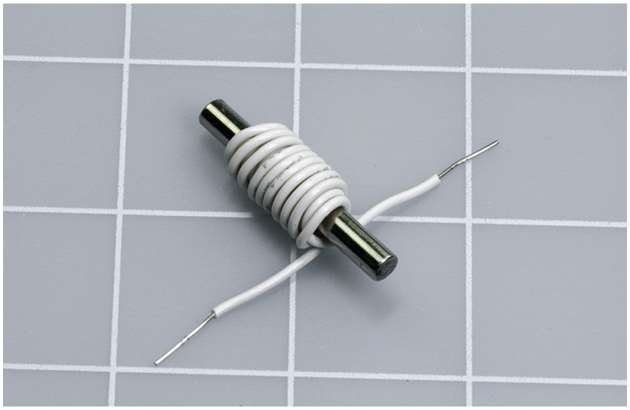


Figure 25-2. Adding more turns of wire will increase the measured voltage when the magnet moves through them.

Figure 25-3. Two hundred turns of 22-gauge wire on a homemade spool, with a magnet that attaches to a screw in a dowel.

Magnet wire would be ideal for this experiment, but I'm guessing that you may not want to buy 500 feet of it, which you will never use again.

Therefore, I'll start by using 22-gauge wire, because you can always reuse it later to make breadboard jumpers. This type of wire can create a coil which will generate enough power to light an LED, but only just. You have to follow the directions precisely.

You'll see from the specifications for this experiment, you need 200 feet of 22-gauge hookup wire. You can buy this in two spools of 100 feet each, and join them together to make 200 feet while you are winding a spool such as the one in Figure 25-3. Joining one piece of wire with the next piece can be done just by stripping the ends and twisting them together tightly. You don't need solder.

- When you want to create a magnetic field with a coil of wire, all the turns must be in the same direction.

To make an LED flash, you will need a more powerful magnet. The one that worked for me is cylindrical, measures 1" long and 3/4" diameter, and is axially magnetized, meaning that the north and south poles are at opposite ends of its axis. (The axis is an imaginary line that runs through the center of the cylinder. You can imagine the cylinder like a shaft rotating around its axis.)

You will need a spool with a central tube just big enough for the magnet to slide through. At your local hardware store, you can find 3/4" PVC water pipe with an internal diameter that is actually a fraction more than 3/4". This will allow the magnet to move freely, but the pipe is not manufactured to fine tolerances. Take the magnet to the store with you, to check. If one piece of pipe is a little too tight, try another. Note that gray electrical conduit is usually a fraction narrower than white PVC water pipe.

I used quarter-inch plywood to make the discs for a spool, but any other nonmagnetic material will do. If you use corrugated cardboard, it must be thick and rigid.

The dimensions of your coil are important. The discs must be at least 4" in diameter, and the gap between them must be a fraction more than 1". Wire that you wind between the discs will tend to push outward as it accumulates, so the discs must be firmly attached to the pipe. You cannot use metal screws or bolts for this purpose, because they will interact with the magnet. I used epoxy glue to join the discs with the pipe.

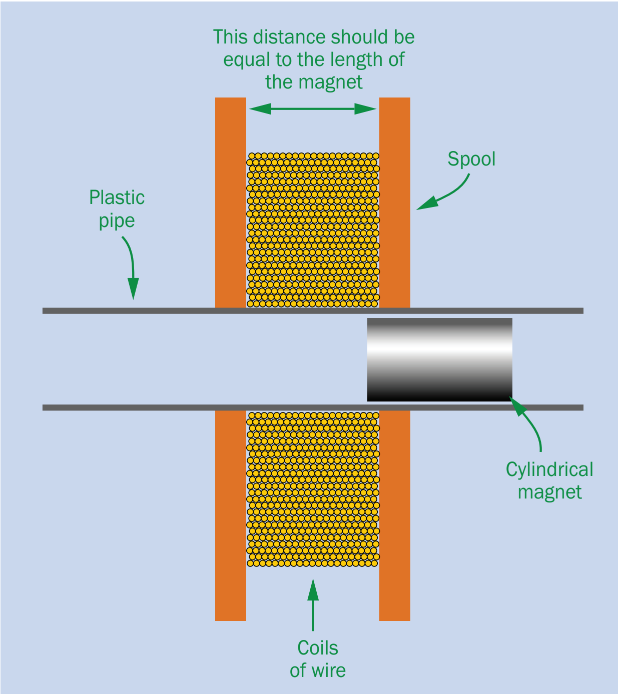


Figure 25-4. Setup for generating just enough power to light an LED.

Figure 25-5. A spool of magnet wire with the inside end accessible, circled in red.

When you are ready to start winding the coil, remember to drill a hole through one of the disks near the pipe, so that you can poke the end of the wire through it and get access to it later. Grip the spool firmly while you are winding the coil, and try to make the turns of wire neatly, so that they sit together with minimal wasted space.

The pipe must be at least 4" long, so that the magnet can slide through it in three phases:

1. When the magnet is at one end of the pipe, it is completely beyond the coil.

2. The magnet passes through the coil, which is about as thick as the length of the magnet.

3. At the other end of the pipe, the magnet is again completely beyond the coil.

To move the magnet conveniently, I drilled a hole in one end of a piece of 1/2" wooden dowel, and inserted a thin 1" flat-headed screw. I was then able to hold the dowel like a handle, while the magnet attracted itself to the screw. However, you can use the tip of a screwdriver to move the magnet, and the steel in the screwdriver should not interfere significantly with the experiment. Alternatively, you can pick up the coil assembly and shake it with the magnet shuttling back and forth in the tube.

Use a couple of alligator test leads to attach the ends of your coil to the probes of your meter, and set the meter to AC volts, as you did before. This time, though, set it to measure up to 2V. When you agitate the magnet as fast as possible, you should see a voltage of 0.5V to 0.7V.

You took all that trouble, and you got less than a volt?

Ah, but your meter is ***averaging*** the current. Each pulse is probably peaking at a higher voltage.

Disconnect your meter and use your test leads to attach an LED to the wires from your coil. A red LED will give you a better chance of seeing something than white or blue LEDs, which require more forward voltage. Now when you move the magnet vigorously, you should see the LED flash.

If it doesn't work, the problem is almost certainly that the magnet isn't moving fast enough. Try this: Grip the spool tightly, with your finger over one end of the pipe and your thumb over the other end, and move your arm as forcefully as if you were making a baseball pitch. Still nothing? Reverse the connection of the LED, because you may be able to move your arm more quickly in one direction than the other, and the magnet has polarity.

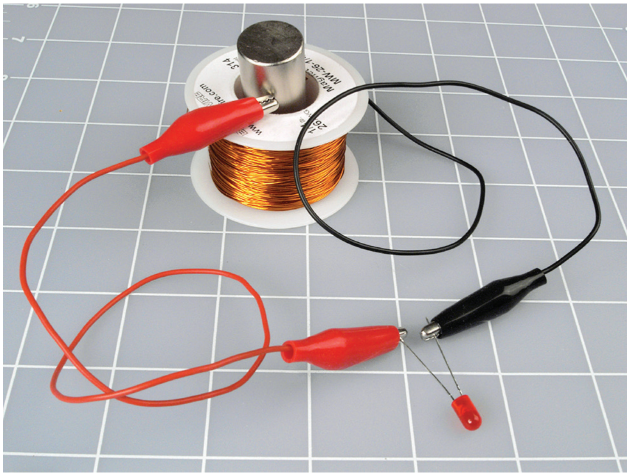


Figure 25-6. Ready for power generation, on a rather small scale.

Optional Enhancements

If you are willing to spend a little more money, you can get more impressive results.

First, use a bigger magnet. I got excellent results from one 2" long and 5/8" in diameter. Of course, you'll need a larger-diameter piece of PVC pipe to accommodate the magnet.

Second, there's the magnet-wire option. I used about 500 feet of 26-gauge wire. It's easy to find online; there are dozens of suppliers.

If you're fortunate, your magnet wire will be supplied on a plastic spool with a hole in the middle just a little bigger than the diameter of your magnet. And, better still, the spool will allow you access to the "tail" of the magnet wire sticking out in the center of the spool, as shown circled in red in Figure 25-5.

To remove the thin film of insulation from the ends of the magnet wire, you can rub them with fine sandpaper, or v-e-e-r-y carefully scrape them with the edge of a utility knife. Check with a magnifying glass to make sure that some insulation has been removed. You can apply your meter to check the resistance of the entire spool of wire, which should be less than 100 ohms.

Now you can attach an LED to each end of the magnet wire on the spool, and generate voltage by pushing your magnet in an out of the center of the spool, as shown in Figure 25-6. The LED will flash very easily.

Figure 25-7. Demonstration power-generation device capable of dazzling results.

If the spool is the wrong size, or if the tail of the wire isn't accessible, you'll just have to rewind the wire onto another spool. Suppose you have 500 feet; that will entail rewinding about 2,000 turns. If you can make four turns per second, you'll require 500 seconds—a little less than 10 minutes. That's doable.

Figure 25-7 shows a larger-scale device that I built for demonstration purposes. The coil of magnet wire is coated with epoxy glue, so that it won't unravel, and I mounted the pipe in a block of plastic that holds it securely. My neodymium magnet attracts itself to a piece of steel hammered into the end of an aluminum rod, also visible in the photograph.

I added two high-intensity LEDs to the coil, with their polarity in opposite directions. When the magnet shuttles

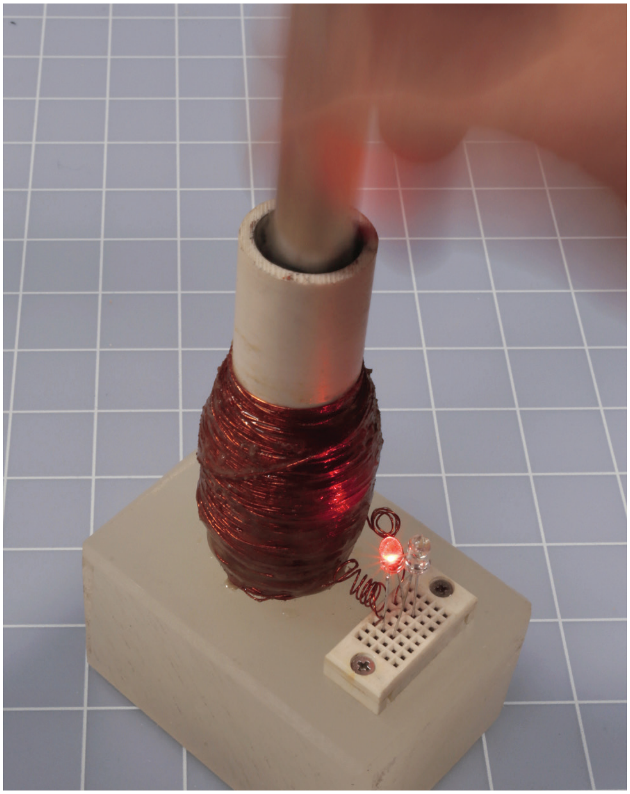


Figure 25-8. The LED generator in action.

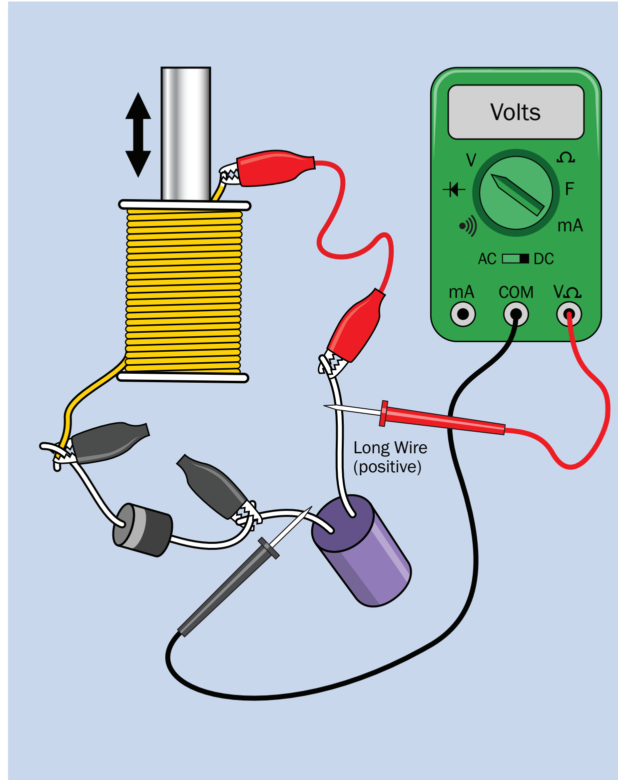


Figure 25-9. A diode enables you to accumulate voltage from your coil in a capacitor.

up and down, the LEDs light up the whole room. Also, because the LEDs are wired in parallel but with opposite polarity, they show that voltage travels through the coil in one direction on the up-stroke, and in the other direction on the down-stroke. See Figure 25-8.

Caution: Blood Blisters and Dead Media

Neodymium magnets are breakable. They're brittle and can shatter if they slam against a piece of magnetic metal (or another magnet). For this reason, many manufacturers advise you to wear eye protection.

Neodymium magnets can hurt. Because a magnet pulls with increasing force as the distance between it and another object gets smaller, it closes the final gap very suddenly and powerfully. You can easily pinch your skin and get blood blisters (or worse). Ouch!

Magnets never sleep. In the world of electronics, we tend to assume that if something is switched off, we don't have to worry about it. Magnets don't work that way. They are always sensing the world around them, and if they notice a magnetic object, they want it, *now*. Results may be unpleasant, especially if the object has sharp edges and your hands are in the way.

When using a magnet, create a clear area on a nonmagnetic surface, and watch out for magnetic objects underneath the surface. For example, my magnet sensed a steel screw embedded in the underside of a kitchen counter top, and slammed itself into contact with the counter top unexpectedly.

It's hard to take this seriously till it happens to you. But neodymium magnets are full of surprises, and they don't fool around. Proceed with caution.

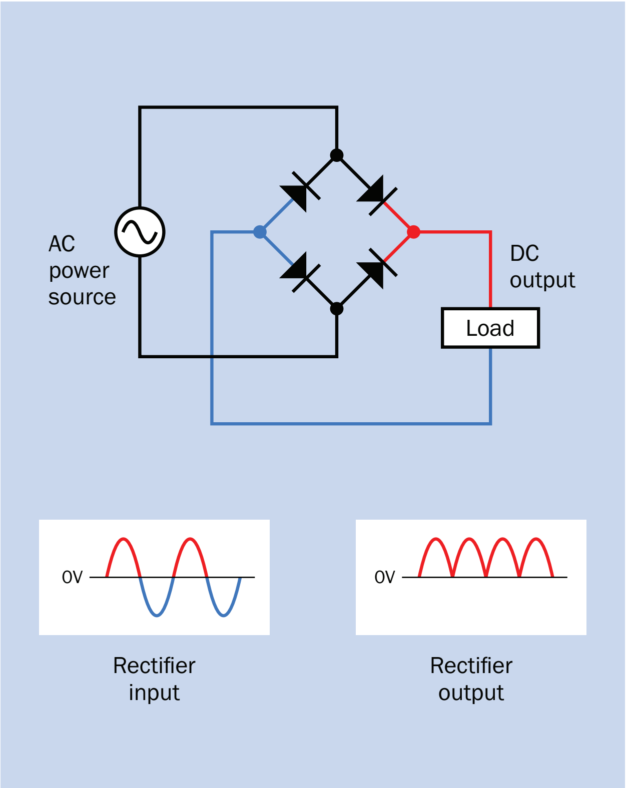


Figure 25-10. For serious real-world applications, four diodes can convert AC to DC.

Also, remember that ***magnets create magnets.*** When a magnetic field passes across an iron or steel object, the object can pick up some magnetism of its own. If you wear a watch, be careful not to magnetize it. If you use a smart phone, keep it away from magnets.

Likewise, any computer, disk drive, or video display is vulnerable to the strong field of a neodymium magnet.

Last but not least, powerful magnets can interfere with the normal operation of cardiac pacemakers.

Charging a Capacitor

Here's another thing to try. Disconnect the LED from whatever coil of wire you created, and connect a 1,000µF electrolytic capacitor in series with a signal diode, as shown in Figure 25-9. Attach your meter, measuring DC volts (not AC, this time), across the capacitor.

If your meter has a manual setting for its range, set it to at least 2VDC. Make sure the positive (unmarked) side of the diode is attached to the negative (marked) side of the capacitor, so that positive voltage will accumulate on one side of the capacitor while electrons can reach the other side, through the diode..

Now move the magnet vigorously up and down in the coil. The meter should show that the capacitor is accumulating charge. When you stop moving the magnet, the voltage reading may decline very slowly, mostly because the capacitor discharges itself through the internal resistance of your meter.

This experiment is more important than it looks. Bear in mind that when you push the magnet into the coil, it induces current in one direction, and when you pull it back out again, it induces current in the opposite direction. You are actually generating alternating current.

The diode only allows current to flow one way through the circuit. It blocks the opposite flow, which is how the capacitor accumulates its charge. If you jump to the conclusion that diodes can be used to convert alternating current to direct current, you're absolutely correct, and this process is known as ***rectifying*** the alternating current.

A diode only functions as a ***half-wave rectifier***, because it allows only half of the AC to pass through, while blocking the other half. You can use four diodes to make a ***full-wave rectifier***, as shown in Figure 25-10. For practical uses, the output has to be smoothed, and capacitors are usually added for that purpose.

Up Next: Audio

Experiment 24 showed that voltage can create a magnet. Experiment 25 has shown that a magnet can create voltage. We're now ready to apply these concepts to the detection and reproduction of sound.

Experiment 26

Speaker Destruction

You saw that electricity running through a coil can create enough magnetic force to pull a small metallic object toward it. What if the coil is very light, and the object is heavier? In that case, the coil can be pulled toward the object.

This principle is at the heart of a loudspeaker, which I am referring to throughout this book as a ***speaker***.

While you have used a speaker in previous experiments, I'd like you to see how it works; and for that purpose, there's really no better way than to disassemble it.

Maybe you'd prefer not to spend a few dollars on this destructive but educational process—in which case, you might consider picking up a piece of nonfunctional audio equipment at a yard sale, and pulling a speaker out of that. Or simply take a look at my photographs illustrating the process step by step.

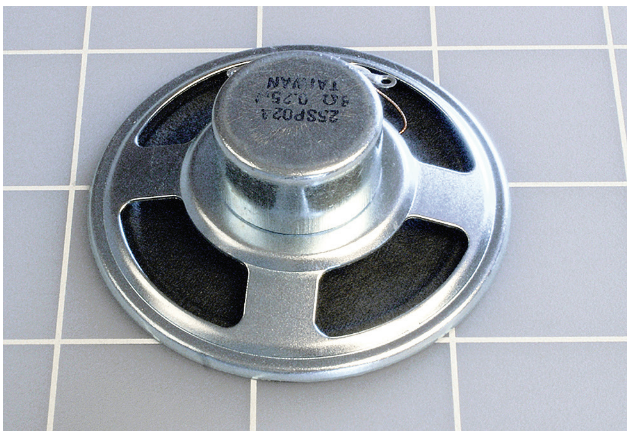


Figure 26-1. The back of a small loudspeaker.

Figure 26-2. A two-inch loudspeaker ready for its fate.

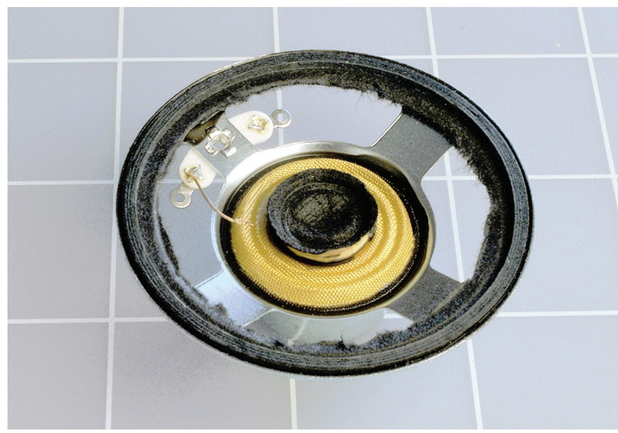


Figure 26-3. The loudspeaker with its cone removed.

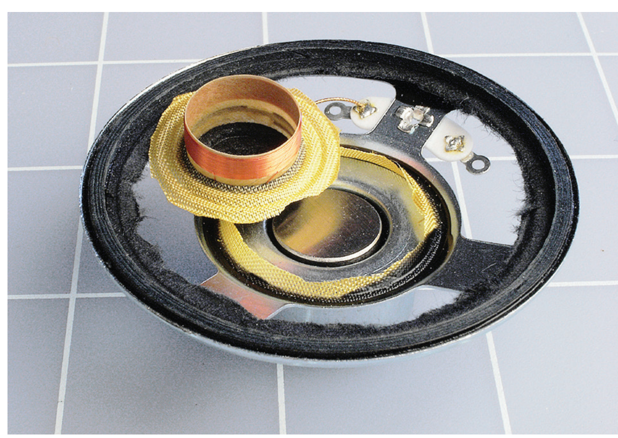


Figure 26-4. The copper coil is normally hidden inside the groove of the magnet, below it.

You Will Need:

- Cheapest possible speaker, working or not working, with 2" minimum diameter (1).
- Utility knife (1).

Procedure

Figure 26-1 shows a small speaker seen from the rear. A magnet is hidden in the sealed cap at the top of the picture.

Turn the speaker face-up, as shown in Figure 26-2. Cut around the outside edge of the cone with a sharp utility knife or X-Acto blade, then cut around the circular center and remove the O-shaped circle of black paper that you've created.

The unconed speaker is shown in Figure 26-3. The yellow weave at the center is a flexible section that normally allows the cone to move in and out, while preventing it from deviating from side to side.

Cut around the outside edge of the yellow weave, and you should be able to pull up a hidden paper cylinder, which has a copper coil wound around it, as shown in Figure 26-4. In the photograph, I've turned it over so that it is easily visible.

The two ends of this copper coil normally receive power through flexible wires from two terminals at the back of the speaker. When the coil sits in the groove visible in the magnet, the coil reacts to voltage fluctuations by exerting an up-and-down force in reaction to the magnetic field. This vibrates the cone of the speaker and creates sound waves.

Large speakers in a stereo system work exactly the same way. They just have bigger magnets and coils that can handle more power (typically, as much as 100 watts).

Whenever I open up a small component like this, I'm impressed by the precision and delicacy of its parts, and the way it can be mass-produced for such a low cost. I imagine how astonished Faraday, Henry and the other pioneers of electrical research would be, if they could see the components that we take for granted today. Henry spent days winding coils by hand to create electromagnets that were ten times as big but far less efficient than this cheap little speaker.

Figure 26-5. This beautiful Amplion horn illustrates the efforts of early designers to maximize efficiency in an era when the power of audio amplifiers was very limited.

Origins of Speakers

As I mentioned at the beginning of this experiment, a coil will move if its magnetic field interacts with a heavy or fixed object made of iron or steel. If the object is magnetized, the coil will interact with it more strongly, creating more vigorous motion. This is how a speaker works.

The idea was introduced in 1874 by Ernst Siemens, a prolific German inventor. (He also built the world's first electrically powered elevator in 1880.) Today, Siemens AG is one of the largest electronics companies in the world.

When Alexander Graham Bell patented the telephone in 1876, he used Siemens' concept to create audible frequencies in the earpiece. From that point on, sound-reproduction devices gradually increased in quality and

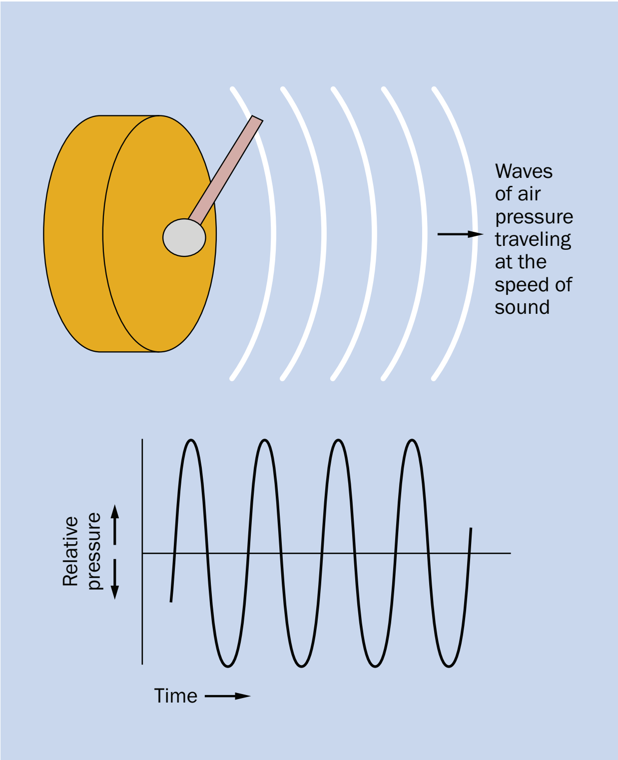


Figure 26-6. Striking a going makes its flat surface vibrate. The vibrations create waves of pressure in the air.

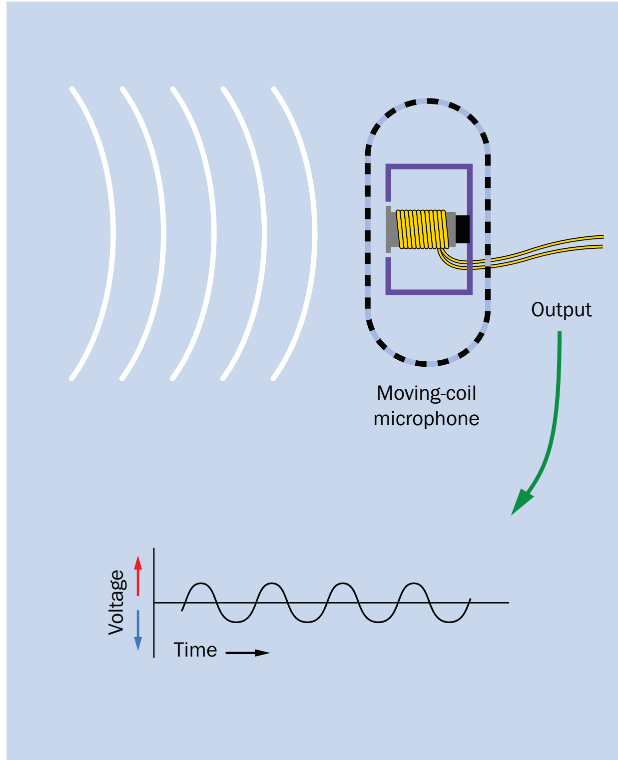


Figure 26-7. Sound waves entering a moving-coil microphone make a membrane vibrate. The membrane is attached to a coil on a sleeve around a magnet. Motion of the coil induces small currents.

power, until Chester Rice and Edward Kellogg at General Electric published a paper in 1925 establishing basic principles that are still used in speaker design a century later.

Online you can find photographs of very beautiful early speakers which used a horn design to maximize efficiency, as shown in Figure 26-5 on the previous page. A diaphragm was installed in the metal compartment at the base of the horn, and was caused to vibrate by current passing through copper coils.

As sound amplifiers became more powerful, speaker efficiency became less important compared with quality reproduction and low manufacturing costs. Today's speakers convert only about 1% of electrical energy into acoustical energy.

Sound, Electricity, and Sound

Time now to establish a more specific idea of how sound is transformed into electricity and back into sound again.

Suppose someone bangs a gong with a stick, as shown in Figure 26-6. The flat metal face of the gong vibrates in and out, creating pressure waves which the human ear perceives as sound. Each wave of high air pressure is followed by a trough of lower air pressure, and the wavelength of the sound is the distance (usually ranging from meters to millimeters) between one peak of pressure and the next.

The frequency of the sound is the number of waves per second, usually expressed as hertz.

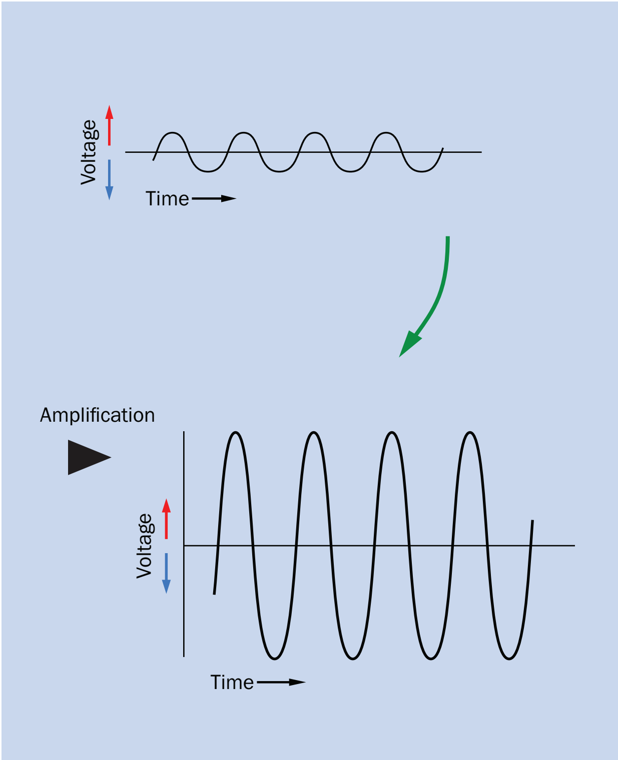


Figure 26-8. Tiny signals from the microphone pass through an amplifier, which enlarges their amplitude while retaining their frequency and the shape of their waveform.

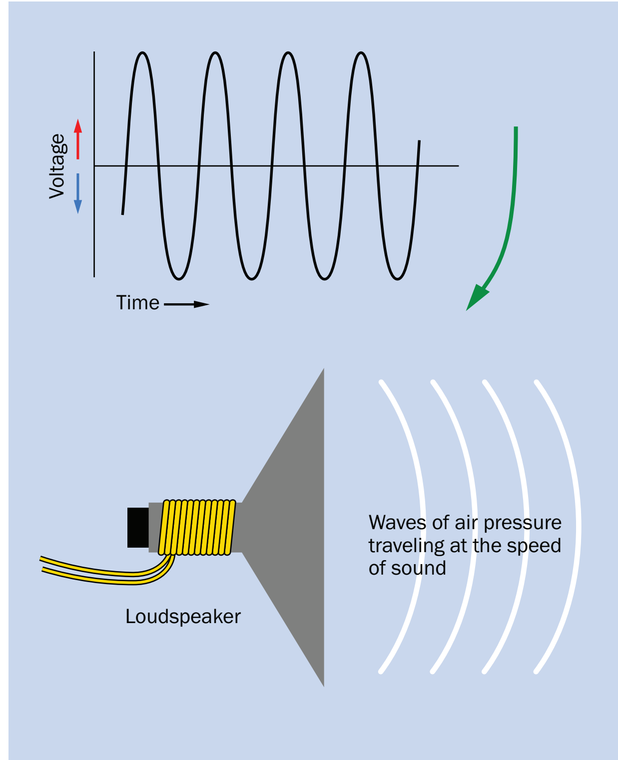


Figure 26-9. The amplified electrical signal is passed through a coil around the neck of a speaker cone. The magnetic field induced by the current causes the cone to vibrate, reproducing the original sound.

Suppose we put a very sensitive little membrane of thin plastic in the path of the pressure waves. The plastic will flutter in response to the waves, like a leaf fluttering in the wind. Suppose we attach a tiny coil of very thin wire to the back of the membrane so that it moves with the membrane. And let's position a stationary magnet inside the coil of wire. This configuration is like a tiny, ultra-sensitive speaker, except that instead of electricity producing sound, the sound will produce electricity. Pressure waves make the membrane oscillate along the axis of the magnet, and the magnetic field creates a fluctuating voltage in the wire. The principle is illustrated in Figure 26-7.

This is known as a ***moving-coil*** microphone. There are other ways to build a microphone, but this is the configuration that is easiest to understand. Of course, the voltage that it generates is very small, but we can amplify it using a transistor, or a series of transistors, as suggested in Figure 26-8.

Then we can feed the output through the coil around the neck of a speaker, and the speaker will recreate the pressure waves in the air, as shown in Figure 26-9.

Somewhere along the way, we may want to record the sound and then replay it. But the principle remains the same. The hard part is designing the microphone, the amplifier, and the speaker so that they reproduce the waveforms ***accurately*** at each step. It's a significant challenge, which is why accurate sound reproduction was only perfected over a period of decades.

Experiment 27
Making a Coil React

You've seen that when you pass current through a coil, the current creates a magnetic field. When you disconnect the current, what happens to the field that it created?

Some of the energy in the field is converted back into a brief pulse of electricity. We say that this happens when the field ***collapses***.

This experiment will enable you to see it for yourself.

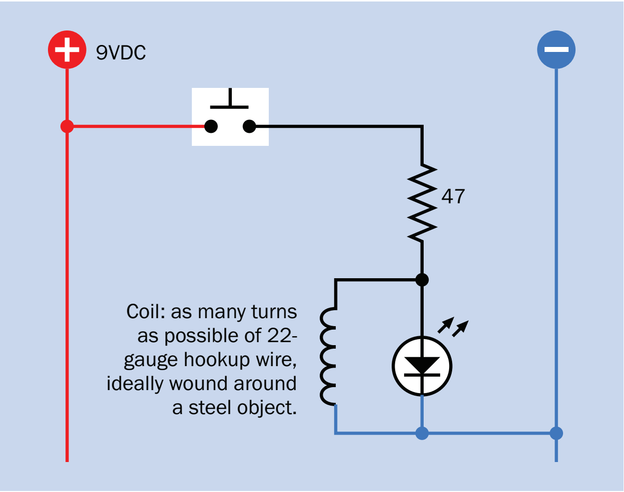


Figure 27-1. A simple circuit to demonstrate the self-inductance of a coil.

You Will Need:

- Breadboard, hookup wire, wire cutters, wire strippers, multimeter.
- Generic red LEDs (2).
- Hookup wire, 22 gauge (24 or 26 gauge okay), at least 25 feet (100 feet preferred).
- Resistor, 47 ohms (2).
- Capacitor, 1,000µF or larger (1).
- Tactile switch (2).

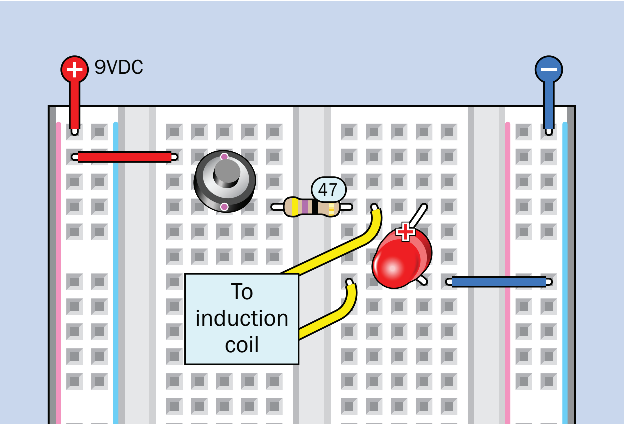


Figure 27-2. Breadboarded version of the self-inductance experiment.

Procedure

Take a look at the schematic in Figure 27-1, and the breadboarded version in Figure 27-2. For the coil, you can use as many turns as possible of hookup wire, wound around a screwdriver or other steel object. If you made the big coil in Experiment 25, that should work well.

When you look at the schematic, it doesn't seem to make much sense. The 47-ohm resistor seems too small to protect the LED—but why should the LED light up at all, when the electricity can go around it through the coil?

Now test the circuit, and I think you'll be surprised. Each time you press the button, the LED blinks briefly. Can you imagine why that should be?

Try adding a second LED, but be careful to turn it around, as in figures 27-3 and 27-4. Include the second 47-ohm resistor, press the button again, and the first LED flashes, as before. But now when you release the button, the second LED flashes. The LEDs are not very bright, but they are definitely flashing in response to some current passing through—first in one direction, then the other, even though you are only using a DC power supply.

Important: Don't press the button for more than a couple of seconds. You'll roast those little 47-ohm resistors, and if you are using a 9V battery, you'll run it down.

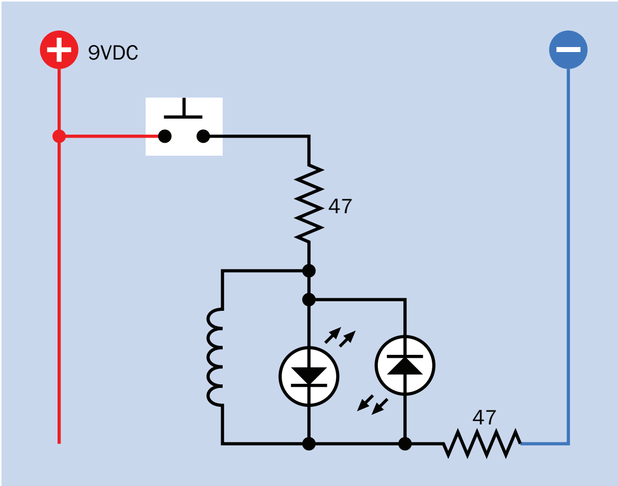


Figure 27-3. One LED flashes when a magnetic field is created; the other flashes when the field collapses.

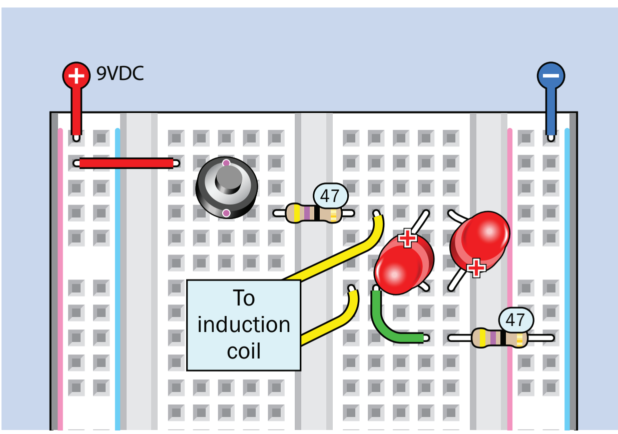


Figure 27-4. Breadboarded version of the two-LED demonstration circuit.

A Collapsing Field

When you apply voltage to a coil, initially the coil doesn't want to allow current to flow through it. Its uncooperative attitude is known as ***self-inductance***. While the coil is resisting current, the current bypasses it and flows briefly through the LED. Then the coil surrenders and starts to pass current, diverting it around the LED, which goes dark.

Sometimes people use the term ***inductive reactance***, or just ***reactance***, to describe the behavior of a coil.

While electricity is passing through the coil, it creates a magnetic field, which requires some energy. When you disconnect the power, the magnetic field collapses, and the energy is converted back into electric current. This caused the little pulse that you saw in the second LED when you let go of the button.

Perhaps you remember that in Experiment 20, I asked you to add a ***freewheeling diode*** in parallel with a relay coil, to absorb the pulse that the coil would create when the relay was switched off. That was exactly the same phenomenon that you're seeing here with a light-emitting diode. When power is disconnected and the magnetic field created by a coil collapses, you get ***back EMF***, where "EMF" is an acronym for ***electromotive force***. You have now seen for yourself that it exists.

Naturally, different sizes of coil store and release different amounts of energy.

Resistors, Capacitors, and Coils

The three primary types of passive components in electronics are resistors, capacitors, and coils. I can now list and compare their properties:

A ***resistor*** constrains current flow, and drops voltage.

A ***capacitor*** allows a pulse of current to flow initially, but blocks a continuing DC supply.

A ***coil*** (often referred to as an ***inductor***) blocks DC current initially, but allows a continuing DC supply.

In the circuit that I just showed you, I didn't use a higher-value resistor because I knew the coil would allow only a very brief pulse. The blinking LEDs would have been less easily visible if I had used a more usual 330-ohm or 470-ohm resistor.

Don't try to run the circuit shown in Figure 27-4 without the coil of wire included. You will quickly burn out one or both of the LEDs. The coil may look as if it isn't doing anything, but it is.

Here's one last variation on this experiment, to test your memory and understanding of electrical fundamentals. Build a new circuit shown in figures 27-5 and 27-6, using a 1,000µF capacitor instead of a coil (be careful to get its polarity the right way around, with the positive lead at the

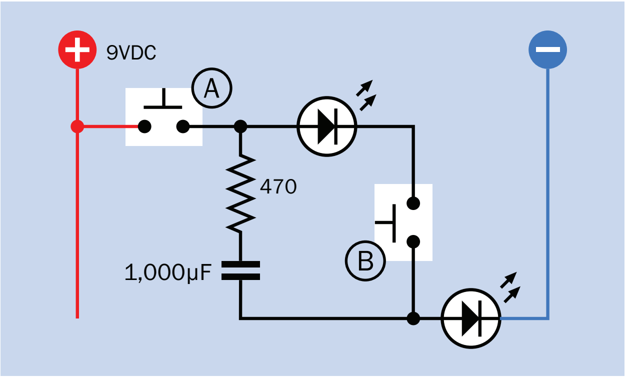


Figure 27-5. In many ways, the behavior of a capacitor is opposite to the behavior of a coil.

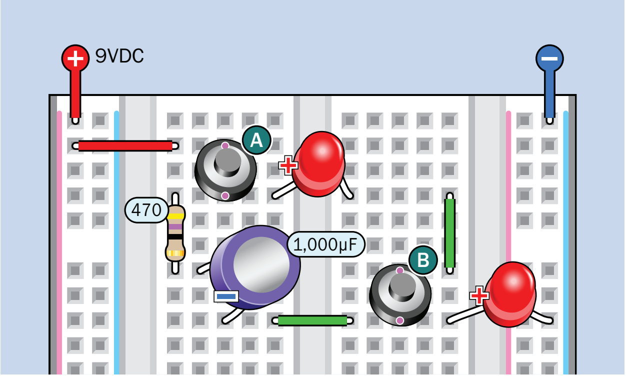


Figure 27-6. Breadboarded version of the capacitance demonstration circuit.

top.) Also, use a 470-ohm resistor, because the coil isn't there to divert current anymore.

First hold down button B for a second to make sure that the capacitor is discharged. Now, what will you see when you press button A? Maybe you can guess. Remember, a capacitor will pass an initial pulse of electricity known as displacement current. Consequently, the bottom LED lights up—and then gradually fades out as the capacitor accumulates a positive charge on its upper plate and a negative charge on its lower plate. As this occurs, the potential across the lower LED diminishes to zero.

The capacitor is now charged. Press Button B, and the capacitor discharges through the upper LED. You can see this as the opposite of the experiment in Figure 27-1, using a capacitor instead of a coil.

Capacitors and inductors both store power. You were able to see this more obviously with the capacitor, because a high-value capacitor is much smaller than a high-value coil.

Alternating Current Concepts

Here's a simple thought experiment. Suppose you set up a 555 timer to send a stream of pulses through a coil. This would be a form of alternating current.

Will the self-inductance of the coil interfere with the stream of pulses? That will depend on how long each pulse is, how rapidly they fluctuate, and how much inductance the coil has. If the frequency of pulses is just right, the self-inductance of the coil will reach a peak with each pulse, tending to block it. Then the coil will recover in time to block the next one. In this way, we can "tune" a coil to suppress some frequencies while allowing others to pass through.

If you have a stereo system with good speakers, each speaker cabinet probably contains two drivers: A small one for high-frequency sound, and a large one for low frequencies. Almost certainly there is a coil and capacitor inside the cabinet to prevent the higher frequencies from reaching the bigger speaker. This is known as a *crossover network*.

I don't have space in this book to get deeply into alternating current. It's a vast and complicated field where electricity behaves in strange and wonderful ways, and the mathematics that describe it can become quite challenging. However, I will show you how the concept of alternating current enables the transmission and reception of radio signals.

Experiment 28

One Radio, No Solder, No Power

This project describes a circuit to receive AM radio signals without a power supply. A device of this type used to be known as a ***crystal radio***, because it used a crystal of a mineral such as galena that functioned as a diode when the tip of a thin wire was pressed against it. The concept originated at the dawn of telecommunications, but if you've never tried it, you've missed an experience that is magical.

You Will Need:

- Smooth-sided glass or plastic cylindrical object, about three inches in diameter, such as a vitamin bottle or water bottle (1).
- Hookup wire, 22 gauge, 60 feet minimum.
- Heavier wire, 16 gauge preferred, 50 to 100 feet.
- Polypropylene rope ("poly rope") or nylon rope, 10 feet.
- Germanium diode (1).
- High-impedance earphone (1).
- Alligator test lead (1).
- Additional alligator clips (3) or use extra test leads.

Optional:

- Nine-volt power supply (battery or AC adapter).
- LM386 single-chip amplifier.
- Electrolytic capacitor, 10µF (1).
- Small speaker (2" acceptable).

The diode must be germanium, not the silicon diode of the type you have used previously. The earphone must be high-impedance, at least 2,000 ohms, not the type of modern earphone that you wear with a phone or MP3 player. For more details, see Appendix A.

Step 1: The Coil

You need to create a coil that will resonate with radio transmissions in the AM waveband. The coil will consist of 65 turns of wire, which can be 22-gauge hookup wire, if you have at least 60 feet available.

You can wind the wire around any empty glass or plastic container with smooth, parallel sides of a constant diameter close to 3". A water bottle will do, if it is made of plastic that is thick enough so that it won't be easily squashed under pressure.

I just happened to have a vitamin bottle that was exactly the right size. In the photographs, you'll notice that it has no label. I softened its adhesive with a heat gun (lightly, to avoid melting the bottle) and then peeled it off. Some

Figure 28-1. A suitable bottle for your coil. The holes will be used to anchor the wire.

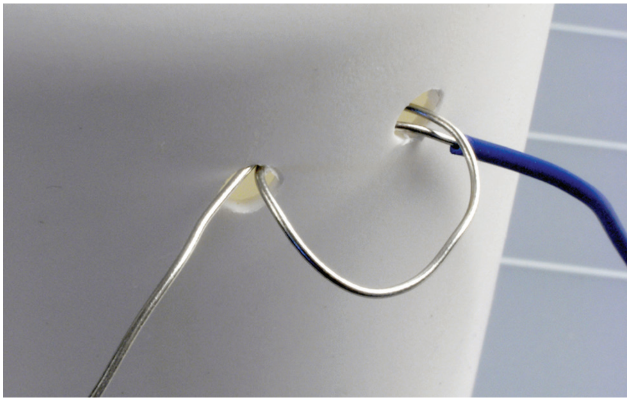


Figure 28-2. Anchor one end of your wire in a pair of holes.

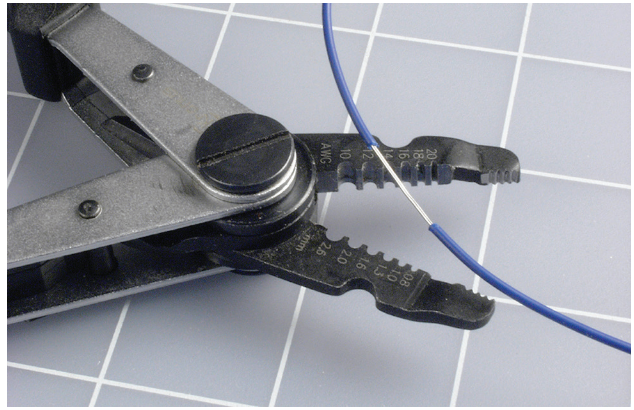


Figure 28-3. Use your wire strippers and your thumbnails to pull back about half-an-inch of insulation.

remaining adhesive residue was removed with a little xylene.

After you prepare a clean, rigid bottle, use a sharp object such as an awl or a nail to punch two pairs of holes in it, as shown in Figure 28-1. The holes will be used to anchor the ends of the coil.

Strip some insulation from the end of your hookup wire and anchor it in one pair of holes, as shown in Figure 28-2. Now wrap five turns of wire around the bottle, and stop it from unwinding itself by applying a small, temporary piece of tape. Duct tape is ideal, or regular Scotch tape will do. "Magic" tape isn't strong enough and will be difficult to remove.

Now you need to open a break in the insulation at the location in the wire just beyond the five turns that you wrapped around the bottle. You can do this by using your wire strippers to bite into the insulation and then push it away either side of the incision, as shown in Figure 28-3.

The next step is to twist the exposed wire into a loop, to make it easily accessible and prevent the insulation from closing up. See Figure 28-4.

You just made an access point into your coil. This is known as a ***tap***. Remove the piece of tape that you used to hold your first five turns temporarily, and wind another five turns around the bottle. Now create another tap. You'll need a total of twelve of them, altogether. It doesn't matter if they don't line up with each other precisely. When you have made the last tap, wind five more turns around the bottle and then cut the wire. Bend the end into a U shape about 1/2" in diameter, so that you can hook it through the pair of holes that you drilled at the far end of the bottle. Pull the wire through, then loop it around again to make a secure anchor point.

The coil that I wrapped around a vitamin bottle is shown in Figure 28-5.

Your next step is to set up an antenna, which will be a section of wire that is as thick as possible and as long as possible. If you live in a house with a yard outside, this is easy: Just open a window, toss out a reel of 16-gauge wire while holding the free end, then go outside and string up your antenna by using polypropylene rope ("poly rope") or nylon rope, available from any hardware store, to hang the wire from any available trees, gutters, or poles. The total length of the wire should be 50 to 100 feet. Where it comes in through the window, suspend it on another length of poly rope. The idea is that the rope is an insulator which will keep your antenna wire as far away from the ground or from any grounded objects.

If you don't have an accessible yard, you can string up your antenna indoors, hanging it with poly rope or nylon rope from window treatments, door knobs, or anything else that will keep it off the floor. The antenna doesn't have to be in a straight line; in fact you can run it all around the room.

Figure 28-4. Create a loop in the section of wire that you exposed.

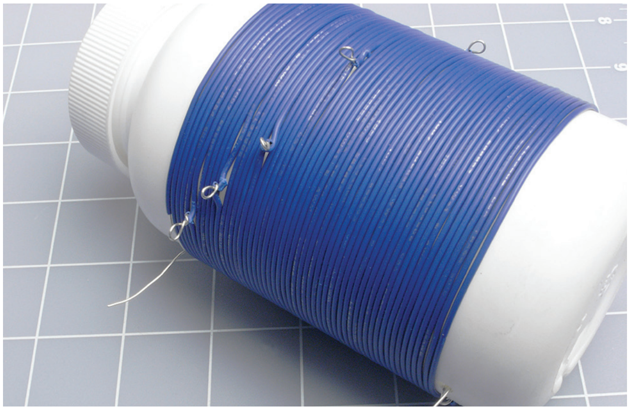


Figure 28-5. The completed coil, wrapped around the bottle.

Caution: High Voltage!

The world around us is full of electricity. Normally we're unaware of it, but a thunderstorm is a sudden reminder that there's a huge electrical potential between the ground below and the clouds above.

If you put up an outdoor antenna, don't use it if there is any chance of a lightning strike. This can be extremely dangerous. If a thunderstorm seems imminent, disconnect the indoor end of your antenna and drop it outside.

Antenna and Ground

Use an alligator test lead to connect the end of your antenna wire with the top end of the coil that you made.

Now you need to establish a ground wire. This literally has to connect with the ground. A cold-water pipe is often suggested for this purpose, because ultimately it goes into the ground somewhere, but (duh!) this will work only if the pipe is made of metal. Because a lot of plumbing these days is plastic, check under the sink to see if you have copper pipes before you try using a faucet for your ground.

Another option is to attach the wire to the screw in the cover plate of an electrical outlet, as the electrical system in your home is ultimately grounded. Be sure to anchor the wire securely, so there is absolutely no risk of it touching the sockets in the outlet.

The sure-fire way to get a good ground connection is to go outside and hammer a copper-clad grounding stake into reasonably moist earth. Any wholesale electrical supply house should be able to sell you a stake. They're commonly used to ground welding equipment. But try the easier options first.

Lastly you need a couple of slightly hard-to-find items: a germanium diode, which functions like a silicon-based diode but is better suited to the tiny voltages and currents that you'll be dealing with, and a high-impedance earphone. The kind of earphones or ear buds that you use with a media player will not work, here; this has to be an old-school item, like the one shown in Figure 28-6. If

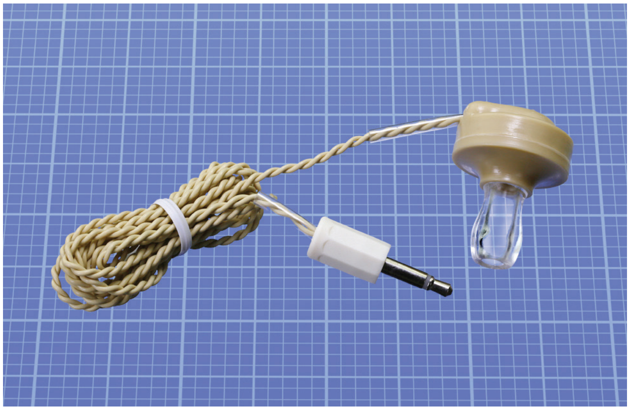


Figure 28-6. This is the type of earphone you need for your no-power radio. If you check it with your meter, you should find a resistance of about 2K.

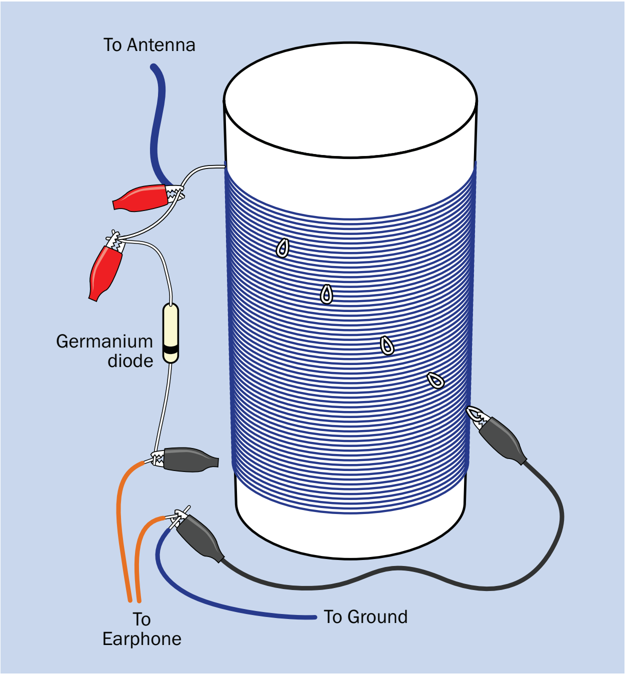


Figure 28-7. The assembled components.

Figure 28-8. The real-world version.

it has a plug on the end, you'll have to snip it off and then carefully strip insulation from the tip of each wire.

The parts are assembled with test leads and alligator clips, as shown in Figure 28-7. The real-world version that I built isn't as neat as the diagram, but the connections are still the same, as shown in Figure 28-8. Notice that the test lead on the right can latch onto any of the taps on your coil. This is how you will tune your radio.

If you followed the instructions, and you live within twenty or thirty miles of an AM radio station, and your hearing is reasonably good, you will be able to listen to the faint sounds of radio on your earphone—even though you are not applying any power to the circuit that you built. This project is more than a century old, but can still be a source of surprise and wonder, along the lines of Figure 28-9.

If you live too far from a radio station, or you can't put up a very long antenna, or your ground connection isn't very good, you may not hear anything. Don't give up; wait till sunset. AM radio reception changes radically when the sun is no longer exciting the atmosphere with its radiation.

To choose among radio stations, move the alligator clip at the end of your test lead from one tap to another on your coil. Depending on where you live, you may pick up just one station, or several, playing individually or simultaneously.

It may seem that you're getting something for nothing here, but really you are taking energy from the transmitter located at a radio station. A transmitter pumps power into a broadcasting tower, modulating a fixed frequency. When the combination of your coil and antenna resonates with that frequency, you're sucking in just enough voltage and current to energize a high-impedance earphone.

The reason you had to make a good ground connection is that the ground works with the antenna to provide capacitance when receiving the radio signal. The transmitter, also, has a ground connection. See Figure 28-10.

Figure 28-9. The pleasure of picking up a radio signal with ultra-simple components and no additional power.

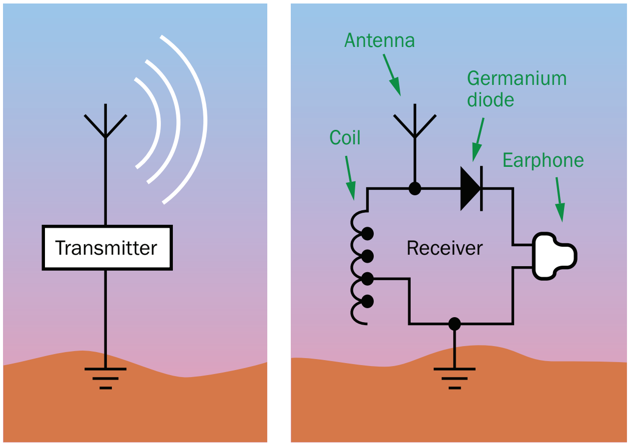


Figure 28-10. Your no-power radio takes just enough energy from a distant transmitter to create a barely audible sound in your earphone. Planet Earth completes the circuit.

Enhancements

If you have difficulty hearing anything through your earphone, try substituting a *piezoelectric transducer* of the type shown in Figure 28-11. Be careful to get the right thing: It should be a *passive* piezo speaker or audio alert. If it is described as a "beeper" or "buzzer" it may contain a circuit that makes a beeping sound when you apply power to it. The word "passive" means that it just reproduces whatever fluctuations in voltage you feed into it.

Press it tightly against your ear, and you may find that it works as well as an earphone, or better.

You can also try amplifying the signal. Ideally you should use an op-amp for the first stage, because it has a very high impedance. However, I decided to put op-amps in *Make: More Electronics*, where I had room to explore

Figure 28-11. A passive piezo speaker or audio alert should cost about as much as the 2-inch speaker you used in previous experiments, but has an effective internal resistance more suitable for your crystal radio.

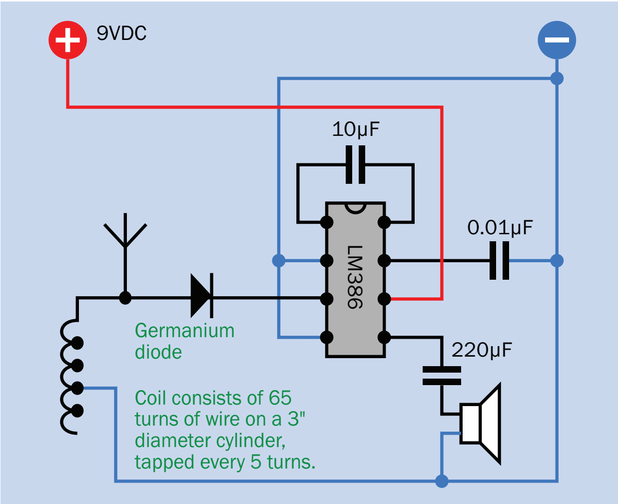


Figure 28-12. An LM386 single-chip amplifier can make your crystal-set radio audible through a loudspeaker.

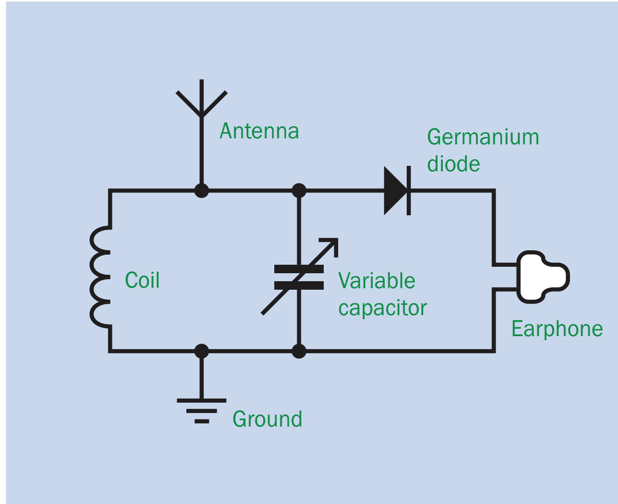


Figure 28-13. When a variable capacitor is added to the previous circuit, it enables better discrimination among different signals.

the topic more thoroughly. As a substitute, you can feed the signal directly into an LM386 single-chip amplifier. The output from this low-cost chip can be reproduced through a regular speaker.

Figure 28-12 shows the amplifier added to the version of the radio that you already built. The germanium diode can connect directly with the LM386 input, as I don't think you'll need a volume control. Be sure to include the 10µF capacitor between pins 1 and 8, as this works with the amplifier to increase its output. I live about 120 miles from Phoenix, Arizona, but was able to pick up a station broadcasting from the Phoenix area.

An alternative option is to improve the selectivity of your radio by adding a variable capacitor to tune the resonance of the circuit more precisely. Variable capacitors are not widely used today, but they are readily available on eBay. See if you can find one rated for 100pF or 200pF. You can buy one that's second-hand; they don't wear out. In figure 28-13, the variable capacitor is shown in the center of the circuit.

How Radio Works

High-frequency electromagnetic radiation can travel for many miles. To make a radio transmitter, I could use a 555 timer chip running at, say, 850 kHz (850,000 cycles per second), and would pass this stream of pulses through an extremely powerful amplifier to a transmission tower—or maybe just a long piece of wire. If you had some way to block out all the other electromagnetic activity in the air, you could detect my signal and amplify it.

This was more or less what Guglielmo Marconi did when he performed a groundbreaking experiment in 1901, except that he had to use a primitive spark gap, rather than a 555 timer, to create the oscillations. His transmissions were not very useful, because they had only two states: on or off. You could send Morse code messages, and that was all. Marconi is pictured in Figure 28-14.

Five years later, the first true audio signal was transmitted. Because a signal at audio frequency has insufficient energy to travel far from a transmitter, the audio wave has to be added to a high-frequency *carrier wave*.

Figure 28-14. Guglielmo Marconi, the great pioneer of radio (photograph from Wikimedia Commons).

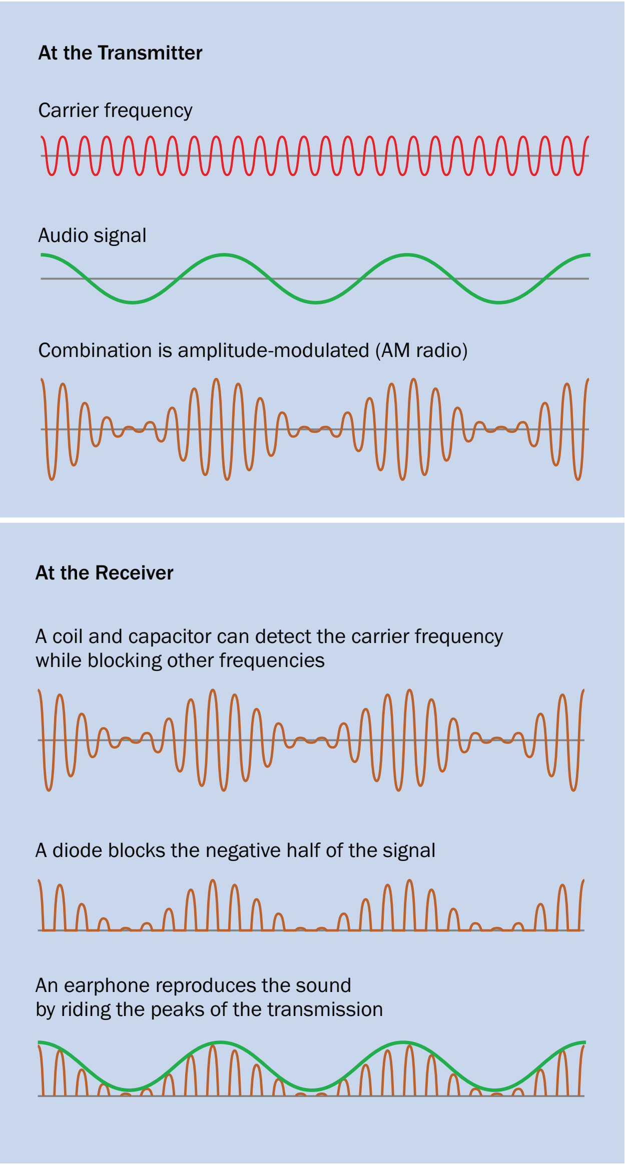


Figure 28-15. Using a carrier wave of fixed frequency to transmit an audio signal. In reality, the carrier is at a much higher frequency than shown here, relative to the audio signal; but the principle is the same.

The power of a carrier varies with the peaks and valleys of the audio, as shown in Figure 28-15. The correct way of describing this is to say that the audio signal ***modulates*** the ***amplitude*** (the size) of the carrier wave. The letters AM in "AM radio" are short for ***amplitude modulation***.

At the receiving end, a very simple combination of a capacitor and a coil can detect the carrier frequency among other noise in the electromagnetic spectrum. The values of the capacitor and the coil must be chosen so that their circuit will resonate at the same frequency as the carrier wave.

An earphone cannot possibly keep up with the frequency of the carrier wave. It would remain hesitating at the midpoint between the highs and lows, producing no sound at all. A diode solves this problem by subtracting the lower half of the signal, leaving just the positive voltage variations. Although these are still very small and rapid, they are now all pushing in the same direction, so that the earphone averages them out, approximately reconstructing the original sound wave.

Each incoming pulse of the carrier wave from the radio transmitter is initially blocked by the self-inductance of the coil, so the energy of the pulse charges the capacitor. If an equally negative pulse is received after an interval that is properly synchronized with the values of the coil and the capacitor, this will coincide with the capacitor discharging and the coil conducting. On this basis, the right frequency of carrier wave makes the circuit resonate. At the same time, audio-frequency fluctuations in the strength of the signal are translated into fluctuations in voltage in the circuit.

If you are wondering what happens to other frequencies pulled in by the antenna, the higher ones are blocked by the coil, while the lower ones pass through the coil to ground. They are just "thrown away."

The waveband allocated to commercial AM broadcasts in the United States is contained within limits of 525kHz to 1710kHz, with a minimum of 10kHz between each station and the next. This range of frequencies is known as the ***medium waveband***.

Many radio frequencies outside that range are allocated for special purposes, such as ham radio. It's not so difficult to pass the ham radio exam, and with appropriate equipment and a well-situated antenna, you can communicate with people all over the world—in realtime, instead of waiting for email.

Experiment 29
Hardware Meets Software

The remainder of this book will deal with microcontrollers, which you can think of as being chip-sized computing devices that are especially useful to control mechanical or electrical gadgets. When a microwave oven beeps to remind you that your food is cooked, a microcontroller is making it do that. In a car with antilock brakes, a microcontroller will be supervising them, while another microcontroller adjusts the fuel injection. Cameras, electronic blood-pressure monitors, washing machines—they all contain microcontrollers.

When you use a desktop or laptop computer, it receives input from a keyboard and displays output on a screen. A microcontroller works on a much lower level, often interacting with the components that you have been learning about in this book, as shown in Figure 29-1.

The success of microcontrollers led to the development of small boards such as the Raspberry Pi and the BeagleBone, each of which can be referred to as an ***SBC***, meaning a ***single-board computer***. Systems of that sophistication are beyond the scope of this book.

A microcontroller is basically a single-chip device. When you copy a program into the chip, instructions in the program can make output pins go high or low, much like the outputs from logic chips. The output can then trigger a transistor or solid-state relay to activate a gadget of your choice.

Other pins on the microcontroller can be configured to receive inputs, such as detecting if a button has been pressed. The microcontroller will also contain some ***analog-digital converters*** which convert a continuously varying input into a digital value that a program can evaluate.

Of course, to perform any of these tasks, a microcontroller needs a program to tell it what to do. In the early days of microcontrollers, the programs were written in assembly language; but in the early 2000s, that started to change.

Enter Arduino

The first Arduino was introduced in 2005, consisting of a little circuit board with a microcontroller mounted on it. Arduino didn't manufacture the chip; a company named Atmel took care of that. The value added by Arduino consisted mostly of some software which helped people to program the microcontroller more easily, using a limited version of a language known as C++ which I will refer to here as Arduino C. This hardware-software combination is now known as a *development system*, because you can use it to develop your own applications.

A lot of people in 2005 already knew how to write programs in C++, so they gravitated quickly to the Arduino,

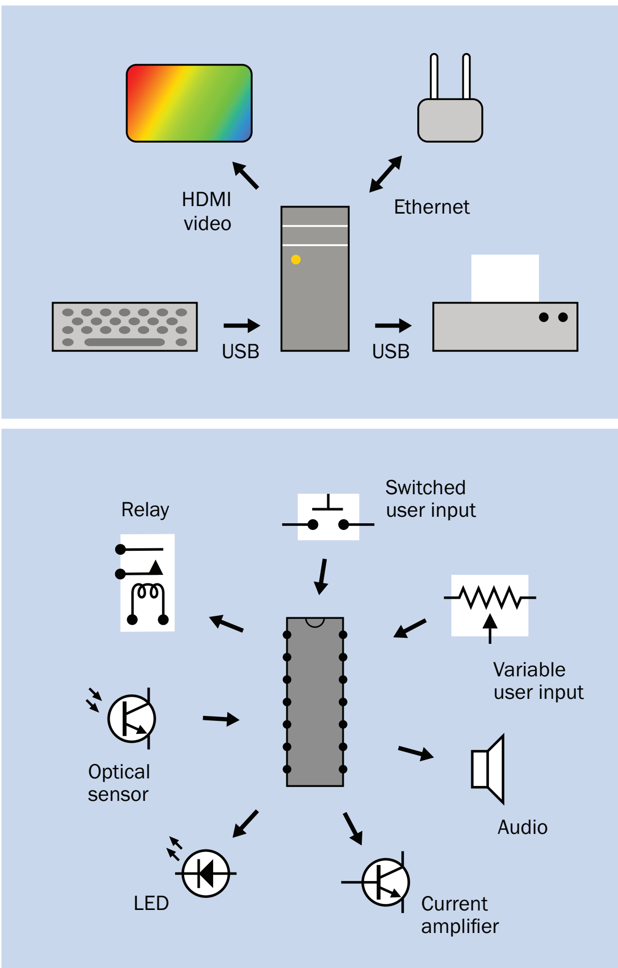


Figure 29-1. Comparing the input and output capabilities of a computer (top) with a microcontroller (bottom).

and a community evolved. People shared pieces of their programs online in *libraries*, and today, if you want to do something such as control a servo motor in a model aircraft, you might start by downloading code from a library which someone else wrote for exactly that purpose. You could then tweak it if it didn't do exactly what you wanted it to do.

Personally I have always preferred to follow my own path, and since this is a book about doing things yourself, I'm assuming that you may want to write your own programs, starting from scratch. With this in mind, I will take you through the initial steps, to give you an idea of how it's done. If you find it interesting, you will need to continue with a guide that is entirely devoted to programming.

First, I have to explain why I will use an Arduino system here instead of some other development system.

Which Microcontroller?

In the years since the first Arduino was marketed, competitors have proliferated. In Chapter 31 I will compare some of them, but in this experiment I am choosing an Arduino for four reasons:

- Arduino boards are still very popular.
- Because of the large user base, huge libraries of code are available. Even if you don't want to use these libraries as-is, you can adapt some of the routines for your own purposes.
- Although intro-level Arduino boards lack sophisticated features, this makes them easier to learn.
- The intro-level Arduino systems are in a stable state of development. They have become a fixture, like through-hole 74xx logic chips, which are still being sold for educational and development purposes fifty years after they were introduced. I believe people will still be using Arduinos for many years to come.

Core Concepts

Normally I like to dive straight in and suggest some experiments, but in this instance, you will need to know some terminology.

The list of instructions that you write for a microcontroller is a *program*. This is called *sketch* by the people who designed Arduino systems, perhaps because they thought that the word "program" sounded too technical. Really, though, it is a program, and if you look online, you'll find people using that word just as often as they talk about a sketch.

Typically you write the program on a desktop or laptop computer using a piece of software known as an ***IDE***, which is an acronym for ***integrated development environment***. You can think of it as being like a simple word processor which is specialized to deal with a programming language.

To obtain the IDE, you download it and install it, like any piece of software. After you use it to write a program, you ask it to tell you if you have made any mistakes. Then you tell the IDE to ***compile*** the program, which is a quick process to convert your instructions into machine language that the microcontroller can understand.

Finally, you tell the IDE to send your program into the microcontroller through a USB cable, at which point you will find out if the program does what you want it to do.

Not all development systems work exactly this way. Some languages don't require compiling, because the microcontroller contains a layer of code known as an ***interpreter*** which can convert your instructions into machine language immediately. However, Arduino C language has to be compiled.

Chip Choices

A photograph of an Arduino Uno board is shown in Figure 29-2, with some of its functions labeled. The rows of input and output sockets along each side of the board are known as ***headers***, which connect through the board with the pins on the microcontroller. You can push wires into the headers to connect the microcontroller with the outside world.

The board in Figure 29-2 contains an ATmega microcontroller as a full-size through-hole DIP chip, inserted in a socket. You can also buy another version of the Uno in which the chip is a surface-mount component which is not removable. This is very slightly cheaper, but I suggest you avoid it, because a socketed microcontroller allows you the option of prying it loose from the board and using it elsewhere, after you program it. This procedure is not quite as simple as it sounds, but can be done.

An alternative to the Arduino Uno is the Nano, which is almost identical in function, but all the components have been compressed to fit on a miniature board with 30 pins in DIP format. This is shown in Figure 29-3. The advantage of the Nano is that you can plug it straight into a breadboard and use it like an integrated circuit chip. I like that, myself, but for many people the Nano has a disadvantage: It cannot be used with ***shields***. This is an issue which I must explain so that you can make an informed buying decision.

A shield is an additional board that sits on top of an Arduino Uno (or similar board) and plugs into its headers. A shield can be like a miniature breadboard, allowing you to plug components into it, but these days most shields are circuit boards with components pre-installed.

Really this type of shield is a ***daughter board***, like the graphics card in a desktop computer. It provides some extra capabilities in hardware that can be controlled by the Arduino underneath. Some types of shields are designed so that you can stack them, like the ones in Figure 29-4 which are specifically designed to control a simple robot.

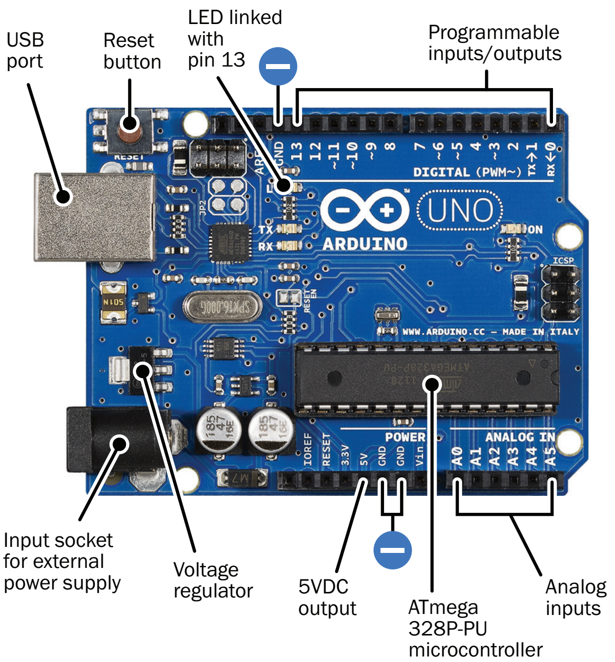


Figure 29-2. An Arduino Uno board featuring the ATmega microcontroller made by Atmel.

Hundreds of shields exist, marketed by many manufacturers. Now perhaps you can see how Arduino has evolved. Initially it was marketed for Makers who would write a few lines of software. The code could tell an Arduino to respond to a phototransistor by switching on some security lights at sunset, or maybe it could control the temperature in a greenhouse by monitoring a thermistor.

Then libraries of code became available online, so you didn't need to write so much of your own software anymore. Then shields were developed, so you didn't need to add your own hardware anymore.

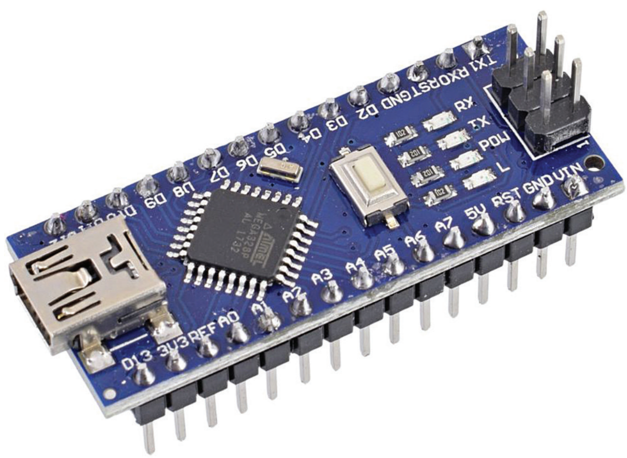


Figure 29-3. The Arduino Nano is breadboard-compatible.

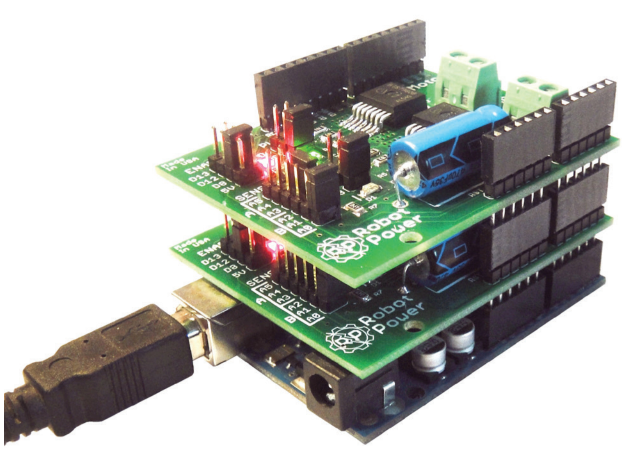


Figure 29-4. Stacked shields on an Arduino Uno.

This trend may seem contrary to the spirit of the maker movement, but it doesn't bother me. Some people want an easy path to a finished product which delivers a quick result; others want to build everything themselves. I'm happy that there's something for everyone. All that concerns me here is that you should understand your options before you buy an Arduino board. The comparison looks like this:

- The Uno can be bought with a socketed controller that can be transplanted later.
- The Uno is compatible with shields, for some additional functionality.
- The Nano is appropriate for breadboard development, in the same way as the circuits that I have described in this book.
- The same program that you write for an Uno will also work on a Nano (so long as it doesn't require a shield).

I'm going to be using the Uno, here, because it's more widely used than the Nano, and you may want to add shields to it in the future. But the very short program that I will show you will be compatible with either an Uno or a Nano.

Have I finished the explanations, now? Are we ready to get started?

Almost. I hate to prolong this introduction, but there is still one more issue to settle: Whether you should buy a clone.

Beware of Imitations?

The Arduino is ***open-source***, meaning that the company chose not to control and protect their intellectual property, because many people in the world of computers believe that the free sharing of information encourages innovation.

Unfortunately this means that anyone can make an "Arduino board," and if it doesn't work in exactly the right way, in every detail, the buyer has no recourse.

Sources such as Mouser, Digikey, Maker Shed, Sparkfun, and Adafruit all sell genuine Arduino products. On other sites, you may find ***unlicensed*** copies of the Arduino board more cheaply, sometimes described as ***clones***. These

Figure 29-5. Only the boards manufactured or licensed by Arduino are supposed to have this logo on them.

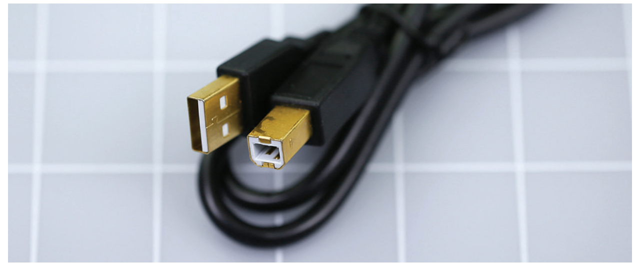


Figure 29-6. You will need this type of USB cable to attach your Arduino Uno board with a USB port on your computer.

imitations should not use the corporate logo, which is controlled by Arduino. To help you in distinguishing real boards from the imitations, you can look for the logo shown in Figure 29-5.

The unlicensed boards are completely legal, and they are probably reliable, but personally I bought a genuine Arduino board because I like to support the company, in the hope that it will continue to create new products.

Now, finally, the hands-on part.

You Will Need:

- Arduino Uno, preferably with socketed DIP chip, not surface-mount. (1).
- Optional: Arduino Nano instead of Arduino Uno (1).
- USB cable with A-type and B-type connector at opposite ends (1).
- Optional: USB cable with A-type connector at one end and mini connector at the other end, if you use the Nano. Note that this cable must have full functionality, for handling data; a phone recharging cable may not have this.
- A desktop or laptop computer with an available USB port and maybe 100MB of disk space (1).
- Generic LED (1).
- Series resistor, 470 ohms (1).

Setup

Arduino boards are not shipped with the necessary USB cable, so you have to buy your own, unless you already happen to have one. The type of cable for the Uno is shown in Figure 29-6. This configuration is often used as a printer cable, and they are easy to find. Even Walmart has them.

The Nano requires a mini-USB plug, and you must be careful that the cable will transmit data. A phone recharging cable is not going to work.

After you acquire the cable, you need the IDE software so that you can start writing programs. This presents you with yet another choice: You can download and install the IDE on your own computer, or you can use a "cloud version" that Arduino maintains online. Personally I prefer not to enter my birth date and email address so that I can use cloud-based software, so I downloaded a copy of the IDE.

Whichever option you prefer, you should find it at

`www.arduino.cc/`

By the time you read this, Arduino may have modified their web pages, but I'm betting you should still be able to find the IDE download without too much trouble.

If you use a Windows operating system, currently Arduino claims to support Windows 7 and upward, but I'm not entirely sure about that. When I tested the IDE installer on an old Windows 7 Dell laptop, I received some interesting error messages, the installation failed, and the board was bricked. I cannot generalize from a single bad experience, but I have a feeling that it's a good idea to use the IDE with a current operating system. (Versions are available for Windows, Mac, and Linux.)

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Figure 29-7. The programming template supplied by Arduino when you first use the IDE.

After you have downloaded an appropriate version of the IDE installer, you double-click its icon and follow installation instructions on the screen. Just in case you feel unsure about this, you can find the current procedure through a search engine. These search terms have worked for me:

```
install arduino ide windows

install arduino ide mac

install arduino ide linux
```

Some of the hits link with videos, which you may prefer to text. Don't plug in the Arduino board while you are downloading and installing the IDE.

After installation is complete, now you can plug the board into your computer using the USB cable. The board can power itself from the USB connection, so you don't need any other power source for it at this point. A green surface-mount LED should start glowing on the board, and a yellow LED should be flickering to indicate data transfer.

On your computer, you should find that the installer has placed an icon for the IDE software. Double-click it, and the IDE should open showing you a default program template which looks like Figure 29-7.

At the time when I'm writing this, a new version of the IDE is in development which may look a bit different. But the principles will remain the same.

The IDE has to recognize your Arduino board; otherwise, you won't be able to copy your program into it. At the top of the IDE window, you will see some menu options. Go to the **Tools** menu and find the **Board** option, which should show Arduino Uno as the board which you are using (or Nano, if you are using that). If necessary, open the submenu and click the correct option.

Also in the Tools menu, if you are using a Windows computer, verify that the word **arduino** appears beside a **COM** port. The concept of communications ports goes back to the early days of the MS-DOS operating system, but this legacy still exists under layers of Windows code. If you do not see the word **arduino** assigned to a port, allow the submenu to open, and see if **arduino** was assigned to a different port. If so, click that one.

If the IDE doesn't see your microcontroller, I regret that I won't be able to troubleshoot that for you here. I suggest you go online and use a search term such as

```
arduino cannot find board
```

to see a range of potential solutions.

The Old Arduino Blink Test

Now you are ready to give your Arduino some instructions. In the default template in Figure 29-7, you see:

```
void setup() {
```

This is a line of program code which the compiler and the microcontroller will understand, and it has to be at the beginning of every Arduino program.

The word `void` tells the compiler that this procedure won't generate any numerical result or output.

The term `setup()` is the name of a procedure that has to be done once only, at the beginning. Notice that there is a **{** mark following `setup()`, and a **}** mark further down.

- Every complete procedure in Arduino C should be contained within an opening `{` symbol and a closing `}` symbol.
- The { symbol and the } symbol are properly known as ***braces***.

The closing brace appears after some white space, but the compiler will ignore all extra white space and line breaks when it gets to work.

The line which reads

```
// put your setup code here, to run once.
```

is a *comment*, to tell you what's going on. The compiler ignores any line beginning with `//`.

Before you start writing new code to modify the template, I suggest you open the **File** menu in the IDE, go to Preferences, and look for the option to **Display line numbers**. Click the check-box beside it, because I will be referring to line numbers in the programming examples that follow.

Now in the editor window, you can delete the comment lines (but don't delete the other statements or the braces). Please write a little program which I have reproduced in Figure 29-8.

```
1 void setup() {
2   pinMode(13, OUTPUT);
3 }
4
5 void loop() {
6   digitalWrite(13, HIGH);
7   delay(1000);
8   digitalWrite(13, LOW);
9   delay(100);
10 }
```

Figure 29-8. The old blink test: A first step into Arduino programming.

If you have any prior familiarity with an Arduino, you'll be groaning as you say to yourself, "Oh no, it's the old blink test!" Yes, and that's why I subtitled this section, "The Old Arduino Blink Test." It's a program that almost everyone uses as a preliminary exercise, although I have changed the delay values. I will be getting to a more interesting project soon enough.

Note that in the Consola font that I have used for this program, a zero has a diagonal line through it to distinguish it from a capital letter O.

As you start typing, you'll find that the editor is helping you by checking for errors. A term such as `pinMode` is a *reserved word* for the Arduino, because it has a special meaning. All reserved words are *case-sensitive*, which means that pinmode or Pinmode are not the same as `pinMode`, and won't work. If you type a command correctly, the IDE displays it in a rusty-red color. If you don't type it correctly, the lettering will remain black.

The word `OUTPUT` is important, too. If you type it as output or Output, that's not the same as `OUTPUT`. When you get it right, the word will change from black to aqua-blue in color.

A semicolon marks the end of the instruction.

- A semicolon must be included at the end of each instruction. Always! Don't forget!

You can press Enter at the end of each line, to start a new line.

Your program won't be formatted exactly like mine, but you can create the formatting easily. Go to the **Tools** menu and choose **Auto Format**. As I mentioned before, the compiler will ignore extra white space, but indents are helpful to you, as they show the structure of a program.

Now, what do the statements in the program mean?

`loop()` on line 5 is the name of a section which follows the setup. As soon as the program is loaded into the Arduino, it will start to run automatically, and after the chip deals with the setup, it will repeat the `loop()` section indefinitely, until the power is disconnected.

On line 6, `digitalWrite` is a command to adjust the output of a pin. Which pin? I am specifying 13. If you check back to Figure 29-2, you'll see that small white numbers are printed on the board alongside the black header at the top of the photo. The programmable input and output pins are numbered 0 through 13, and I chose to use pin 13, which is right next to **GND**, the negative-ground pin.

The **GND** pin can be used in association with the digital pins. A program can tell the microcontroller to set a pin in a `HIGH` state, at which point it will supply positive voltage. Current will pass from the pin, through a small component such as an LED, and into the **GND** pin.

My program tells the Arduino that Pin 13 is going to be a digital output, and on line 6 the program says that the pin should go to a `HIGH` state.

`delay` tells the Arduino to wait and do nothing. For how long? `1000` means 1,000 milliseconds. There are 1,000 milliseconds in a second, so the Arduino is going to wait for a second. During that time, pin 13 will stay high.

I think you can figure out what lines 8 and 9 mean. Now you're ready for the next step.

Verify and Compile

Pull down the **Sketch** menu in the IDE and choose **Verify/Compile**. The IDE examines your code, and if it sees any problems, it complains about them in a black area at the bottom of the IDE window. You can go to the horizontal margin between the two areas and drag it upward with your mouse to make the black area bigger.

Let's suppose you made a mistake: You typed Digital Write instead of `digitalWrite`. When you try to compile the program, you'll get a message referring you to `6:3:error`. That means line six, beginning with the third character (the first two characters are spaces, assuming you used Auto Format).

You have to fix your program text till you can Compile/Verify without any errors.

Upload and Run

Now pull down the **File** menu, and choose **Upload**. Personally I think it makes more sense to imagine that I am downloading from my big computer into the little Arduino, but everyone calls it uploading, and that's just the way it is.

If the upload is successful, you'll see a tiny message saying **Done Uploading** just above the black error window.

Now look at the Arduino board, and your program is already running, making the onboard yellow LED blink on for a second and off for a tenth of a second, in accordance with the instructions in your program. The yellow LED just happens to be wired in parallel with Pin 13.

You're going to need to link a breadboard with your Uno board, and I'd like you to take care of that now. Figure 29-9 shows what I mean. This is one instance where those little flexible jumpers with a plug at each end are useful, but hookup wire will work just as well. You can see that a connection runs from Pin 13 on the Arduino header, through an LED and a series resistor, and back to **GND** on the header. Now your LED is flashing in sync with the yellow LED on the Arduino board.

This may seem a small achievement after taking so many steps, but we have to start somewhere, and a blinking

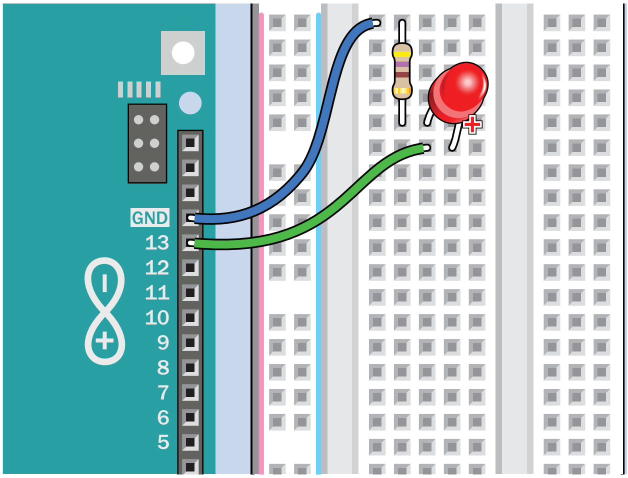


Figure 29-9. Linking the Arduino board with a breadboard.

LED is usually where microcontroller programming begins. Also, it enables you to experiment by changing parameters. In the IDE window, change the first delay statement on line 7 to

```
delay(100);
```

The blink rate of the LEDs doesn't change yet, because you still have to compile the code again and upload it to the Arduino. (Actually if you just choose the **Upload** option, the compiling will occur automatically.)

Now the LED is staying on for only one-tenth of a second. You see, now, that once you have gone through the hassle of writing a program, tweaking it is very easy. If you were using a 555 timer to do this, you'd be swapping resistors or capacitors and looking up the pulse durations in a table.

Here's a summary of what you may have learned so far, and what you had to do, to program the Arduino:

- Start a new program (or "sketch"). Select the **New** option from the **File** menu if necessary.
- Every program must begin with a `setup` function, which will run once only.
- Declare the number of a digital pin, and its mode, by using the `pinMode` command.
- The mode of a pin can be `INPUT` or `OUTPUT`.

- Some pin numbers are not valid. Look at your Uno board to see the numbering system that is used.

- A pair of braces must enclose every complete procedure in a program. They can be on separate lines, because the compiler ignores line breaks.

- Every instruction in a procedure must end in a semicolon.

- Every Arduino program must contain a `loop` function which will run repeatedly.

- `digitalWrite` is a command to make a pin that is set for output have a state that is specified as `HIGH` or `LOW`.

- `delay` makes the Arduino do nothing for a specified number of milliseconds (thousandths of a second).

- You can think of the numbers in parentheses after a command as *parameters* telling the Arduino how the command must be applied.

- Use the **Verify/Compile** option in the **Sketch** menu to check your program before you upload it to the Arduino.

- You must fix any errors found in the **Verify/Compile** operation.

- Reserved words are a vocabulary of commands that the Arduino understands. You have to spell them correctly. Uppercase is considered different from lowercase.

- After you upload your program, it will start running automatically, and will continue to do so until you disconnect power to the board or upload a new program.

- There is a Reset button (a tactile switch) beside the USB connector on the Uno board. When you press it, the Arduino restarts your program at the beginning.

Caution: Lost Code

If you modify your program and upload it to the microcontroller, the new version will *overwrite* the old version. In other words, the old version inside the microcontroller will be erased. If you didn't save the program on your computer under a different file name, it may be gone forever. Be very careful when uploading revised programs. Saving each version on your computer, under a new name, is a sensible precaution. Also, remember:

- After program instructions have been uploaded into the microcontroller, there is no way to read them back out again.

The IDE saves your program automatically in your computer, and gives it a default name derived from the date when you wrote it. You can of course pull down the **File** menu and choose **Save As...** to use your own filename.

Spaces are not allowed in filenames created on an Arduino.

Power and Memory

You may be wondering what happens to your program inside the microcontroller when you disconnect the Arduino board from your computer.

- The microcontroller needs power to run your program. But:

- The microcontroller does not need power to store your program. The program is stored automatically like data in a flash drive.

- If you want to run your program when the board is not connected with your computer, you have to provide power through the round black socket next to the USB socket on the board.

- The power supply can range from 7VDC to 12VDC. It does not have to be regulated, because the Arduino board contains its own regulator, just like the LM7805 that you used with logic chips. The regulator changes your power input to 5VDC for the microcontroller. (Some Arduinos use 3.3VDC, but not the Uno.)

- The power supply jack is 2.1mm in diameter, with center pin positive. You can buy a 9V AC adapter with that kind of plug on its output wire.

- If you connect external power while the Arduino is also connected with a USB cable, the Arduino automatically uses the external power.

- You can disconnect your Arduino from the USB port anytime, without bothering to use the "Safely Remove Hardware" option that exists in some versions of Windows.

Regarding the onboard memory: The ATmega chip uses the same kind of rewritable nonvolatile memory that you find in solid-state hard drives. While this is very reliable, there are uncertainties about its durability. Atmel guarantees it for 10,000 write operations, with provision for automatically locking out memory locations that go bad.

This would be ample in a memory card for your phone or camera, but a program in a microcontroller may make more active use of some areas of its memory. Whether this is a significant factor depends on the type of application and how frequently the chip will be used; but undeniably, a microcontroller does not have the extreme reliability of a traditional-style logic chip. Does this matter? I don't think so, but you have to make up your own mind about it.

Uno and Nano Issues

The Nano can draw power from a USB cable, just like the Uno, but it doesn't have a jack to receive power from an AC adapter. If you want to use the Nano when it is disconnected from your computer, you can supply any unregulated voltage from 6VDC to 20VDC to Pin 30 (labeled **VIN**), or 5VDC regulated power to Pin 27 (labeled **5V**).

What if you make a mistake? If you accidentally apply a higher voltage to Pin 27, you can hurt the chip. If you accidentally apply 5V to Pin 30, nothing will happen. Therefore, it's safer to use a 5V regulated supply, in the same way as if you were powering a logic chip.

What if you connect the Nano with the power supply the wrong way around? You will probably have a dead Nano. If you are using an Uno, it will be a dead Uno.

If you bought the version of the Uno which has a socketed microcontroller, you might imagine that after you program the microcontroller, you can just pull it out of its socket and transplant it anywhere you like. Unfortunately, there are some little snags associated with this, which you can learn about by reading a book titled *Make: AVR Programming* by Elliot Williams. If you are willing to deal with these issues, you will only need one Uno board, while you can buy a lot of Atmel chips very cheaply. In fact, each chip may cost one-tenth the price of an Uno board.

Put one chip into the board, program it, remove it, and use it in a standalone project. Put another chip into the board, transfer a different program into it, and use it in a different project. And so on.

If you use a Nano board, the whole thing has to stay in one piece. You can plug it into a breadboard, but if you want to make multiple copies of your microcontroller program, buying additional Nano boards will be far more expensive than buying multiple ATmega microcontroller chips.

Why Not Use Arduino for Everything?

Now that you have sampled the process of writing a program, and you can see that the Arduino can control many types of components, you may be thinking that you could have saved time by using a microcontroller in projects such as the Intrusion Alarm or the Unlocker.

I don't know about saving time, but you could have added more features to those circuits. For example, processing a passcode consisting of numbers in sequence would have been relatively easy, and you could have put little messages on an LCD screen, too.

This makes microcontrollers very tempting, but of course you don't get something for nothing. You have to learn and understand a programming language, after which you must take the extra steps of writing and debugging the code. Every language has some quirks and limitations, and gradually you bump into these, which can be frustrating. Language documentation may not be very good, and if you look for advice online, it may not be what you want, or it may be incorrect.

Eventually you may find that the microcontroller which you chose doesn't have all the capabilities that you really need, so you are tempted to upgrade to a more powerful chip that the manufacturer has introduced—or a chip by a different manufacturer. Now you have to learn a new version of the programming language, or a different language.

Many people are willing to deal with these issues because a microcontroller opens up so many possibilities. In fact, some tasks that would be impractical or almost impossible with old-school discrete components are relatively easy with a microcontroller, once you become fluent in a programming language. I'll make some comparisons on the next page.

To Code Or Not to Code

Advantages of Discrete Components:

- Simplicity.
- Instant results.
- No programming language needed.
- Cheap, for small circuits.
- Today's knowledge will be valid tomorrow.
- Better for analog applications such as audio.
- Some are still needed with microcontrollers.

Disadvantages of Discrete Components:

- Capable of performing one function only.
- Circuit design is challenging for applications involving digital logic.
- Not easily scalable. Large circuits are difficult to build.
- Revisions to a circuit may be difficult or even impossible.
- More components in a circuit generally require more power.

Advantages of Microcontrollers:

- Versatile, able to perform many functions.
- Additions or revisions to a circuit can be easy (just rewrite the program).
- Huge and diverse online libraries of applications, freely available.
- Ideal for applications involving complex logic.

Disadvantages of Microcontrollers:

- Significant programming skills required.
- Time-consuming development process.
- Evolving technology, requires a continuing learning process.
- Each microcontroller has its own features, requiring study and memorization.
- Greater complexity means more things that can go wrong.
- Requires a desktop or laptop computer, and data storage for programs.
- Data may be lost accidentally.
- Limited output current compared to TTL devices such as 555 timer.

Experiment 30
Nicer Dice

Now I'm going to go through the steps to build a slightly more ambitious program which will emulate the Nice Dice project that I described in Experiment 23. Instead of logic chips, I can use "if" statements with logical operators in a microcontroller program. I can replace several pieces of hardware with a dozen lines of computer code, and a microcontroller will eliminate the need for any other chips in this circuit. This is a great example of an appropriate application. (Of course, it does still require some LEDs and series resistors.)

You Will Need:

- Breadboard, hookup wire, wire cutters, wire strippers, test leads, multimeter.
- Generic LED (7).
- Resistor, 470 ohms (7).
- Arduino Uno board (1).
- Optional: Arduino Nano board instead of Uno (1).
- USB cable with A-type and B-type connector at opposite ends (1).
- Optional: USB cable with mini connector, for Nano board (1).
- Laptop or desktop computer with an available USB port (1).
- Tactile switch (1).

The Limits to Learning by Discovery

Learning by discovery works well when you're getting to know an electronic component. You can put it on a breadboard, apply power, and see what happens. Even when you're designing a circuit, you can use some trial and error and make modifications as you go along.

Writing programs isn't quite like that. If you just dive in without planning ahead, you can waste a lot of time—so in this final project, I'm going to describe the planning process.

Randomicity

The first question seems obvious: "What do I really want an electronic dice program to do?"

I want it to choose a random number from 1 to 6, and then display an appropriate pattern of LEDs. The random number in the chip version was chosen by the player pressing a button to interrupt some rapid counting at an arbitrary moment. Can an Arduino choose a random number for me, all on its own?

To answer a question like this, the first step is to check the Arduino web site, where you can find a language reference section. It is not as comprehensive as I would like, but is a starting point. Currently you can find it from the Arduino home page by clicking the **Documentation** tab and selecting **Reference**. Here you'll see a subhead for **Random Numbers**, using a `random()` function.

Almost every computer language includes a way to generate numbers in a way which seems random, even though it is really done with some mathematical formulas. If I use this feature in Arduino C language, the user can just press a button, the Arduino will select a number from 1 through 6, my program will convert it to the LED display, and that will be the whole story. Job done!

Then if the player needs to "throw the die" a second time, another button-press will do it.

That sounds very convenient—but now I'm wondering if it will look good. A number that pops up instantly may make people suspect that it isn't really random. Going back to the hardware version of this project, maybe the way it worked was a nice feature, like a Las Vegas slot machine, as it displayed a rapid blur of patterns and invited the player to stop the sequence.

A real slot machine, though, shows a random sequence that isn't a blur. The symbols appear just fast enough to be tantalizing, and you can try to stop the display when it shows you what you want.

I like that idea, so I'll use the `random()` function built into Arduino C to choose a series of numbers which appear to be random, and then the user can press a button to stop the sequence.

Okay. Then what?

How about another button which restarts the rapid number display? No, that's not necessary: the same button can do it. Press once to stop, press again to restart.

This is what I want the microcontroller to do. Now I have to figure out how.

Pseudocode

Before I write an actual program, I like to write *pseudocode,* which is a series of statements in English that will be easily converted into computer language. So, here is my pseudocode plan for the program which I will call Nicer Dice. Bear in mind that the microcontroller will execute these instructions very fast.

Step 1. Choose a random number.

Step 2. Convert it to a pattern of dice spots, and light up the appropriate LEDs.

Step 3. Check to see if a button has been pressed.

Step 4. If a button has not been pressed, go back to Step 1 to choose another number and repeat.

If a button has been pressed:

Step 5. Stop updating the display.

Step 6. Wait for the player to press the button a second time. Then go back to Step 1 and repeat.

Do you see any problems with this sequence? Try to visualize it from the microcontroller's point of view. Do you have everything you need, to get the job done?

No, actually not, because some instructions are missing. Step 2 says, "light up the appropriate LEDs," but—there is no instruction anywhere to switch them off!

- A computer *only* does what you tell it to do.

If I want the illuminated LEDs to be switched off before a new pattern is displayed, I have to include an instruction to do that.

Where should I put it? Maybe immediately before the new pattern is chosen and displayed. I'll include it as Step 0:

Step 0. Switch off any LEDs that are on.

But—how will the microcontroller know which LEDs are switched on? It would have to store the current pattern in its memory. That would add to the complexity of the program, so I think a better solution is to tell the microcontroller to switch off *all* the LEDs by making all the output pins go low, regardless of whether they are on or off right now.

The microcontroller will waste a little extra time obeying an instruction to switch LEDs off that are already off. Does this matter?

In the very early days of computing, when processors were slow, everyone had to optimize their programs to avoid the slightest delay. But those days are gone, and even a little microcontroller can waste a few processor cycles here and there. I'll switch off all the LEDs, regardless of their current states, and Step 0 will look like this:

Step 0. Switch off all the LEDs.

Button Inputs

Is anything else missing from the list of pseudocode instructions? Yes, and this will be more difficult. There is a button issue.

Once again, I need to visualize what I want the program to do, and some imagination is necessary. Picture this:

- The rapid display is cycling through numbers very quickly.
- The player presses a button to stop it. The display stops.
- The player releases the button and observes the display.
- Then the player presses the button again, and the rapid display resumes.

I see a problem here. The display will resume so quickly, the player's finger will still be on the button—so the display will stop again, because the button is still being pressed.

So—when the button is pressed the second time, to restart the display, maybe I won't actually restart it until the player lets go of the button. But that's not right, because it's counter-intuitive. People expect something to happen when they press a button, not when they release it.

Well—maybe the player can just learn that the button has to be released to make the display resume!

I don't think so. A program should never force a person to do something that doesn't come naturally. Programs serve people, not the other way around.

You may think this is getting a bit laborious, with so many details. In that case I have to say, "Programming is detail-oriented." That's just the way it is.

Of course you could just abandon the task and look online for programs that emulate dice. I'm sure you would find dozens of them. But then you wouldn't have the satisfaction of doing it yourself, and if anyone asked you "Did you write this program?" you'd have to admit that someone else wrote it.

In any case, the button problem is not so hard. I think what I need is for the display to resume, and during the first second or two, the program doesn't care if the player is still pressing the button, because it ignores the button completely. For this purpose, I will need the microcontroller to count a couple of seconds. Like this:

Before Step 0: Set an internal timer to zero, and start it running.

[other steps]

Step 4. If the button is pressed, AND the clock has counted past 2 seconds, then check the button.

But there is still one more issue: Contact bounce. The microcontroller is sufficiently fast and sensitive, it is liable to misinterpret contact bounce as multiple button presses.

The whole button-pressing thing is turning out to be more complicated than I expected, so I'm going to put it in its own little routine. The pseudocode will look like this (using **BR** to distinguish the button routine instructions from the main part of the program):

BR 1. Ignore the button for a brief moment while its contacts settle after being pressed.

BR 2. Now watch the button and wait until it is released.

BR 3. Now ignore the button for a brief moment while its contacts settle after being released.

BR 4. Watch the button and wait until it is pressed again.

BR 5. Now return to the main routine and resume the random display.

I think this should do the trick. Only one question remains: Can the microcontroller count seconds like a stopwatch?

The System Clock

If you check the language reference, you will find that there is a `millis()` function which measures milliseconds. It starts from zero each time a program begins, and is capable of counting to such a high number, it doesn't reach its limit until about 50 days have passed.

That's nice, but the Arduino doesn't allow a program to reset the system clock to zero. It starts automatically when the program starts, and it doesn't stop until the program ends.

How can I deal with this? In the same way that I would use the clock on my kitchen wall, in the real world, when I want to cook a hard-boiled egg.

First, I look at the clock and make a mental note of the time when the water starts boiling.

To that number, I add the number of minutes that I want, and I memorize the total. This will be my stop time.

When the clock reaches the stop time, I take the egg out of the water.

Let's suppose it is 5:02pm and I want to boil the egg for seven minutes. I say to myself, "5:02 plus seven minutes is 5:09, so I'll memorize 5:09 and take the egg out then."

The way to do this in the dice program is to use a little section of memory which is called a ***variable***, to memorize the stop time. You can think of a variable as being like a box with a label on the outside and a number on the inside. I can invent a name to go on the label, and in this case I will name the variable `ignore`, because it will specify how long the program should ignore the button.

At the start of the loop, I will tell the program to look at the current value of the system clock. I will add 2,000 to it (meaning, 2,000 milliseconds) and put the result inside the box labeled `ignore`. This will be my target time limit. Then while the program is running, it will keep checking the system clock to see if it has exceeded the limit.

Now I have a new Step 4:

Step 4. If the system clock exceeds the `ignore` limit, AND the button is pressed, visit the button routine.

Final Draft of Pseudocode

Bearing all of these issues in mind, here is the revised and, I hope, final sequence of events:

Setup: Set the `ignore` value to the timer value, plus 2,000.

Step 0. Switch off all the LEDs.

Step 1. Choose a random number.

Step 2. Convert it to a pattern of dice spots, and light up the appropriate LEDs.

Step 3. Check to see if the button has been pressed.

Step 4. If the button has been pressed AND the system clock has exceeded the `ignore` value, visit the button routine. Otherwise, go back to Step 0.

Button routine:

BR 1. Wait for a brief moment while the button contacts settle.

BR 2. Watch the button and wait until it is released.

BR 3. After it is released, wait for a brief moment while its contacts settle.

BR 4. Watch the button, wait until it is pressed again.

BR 5. Reset the `ignore` variable to the current time, plus 2,000.

BR 6. Return to the main routine and resume the random display.

Do you think it will work? Let's find out.

Hardware Setup

Figure 30-1 shows seven LEDs wired on a breadboard to display the spots on a die. The concept is the same as in the original Nice Dice project, except that the LEDs are all wired in parallel, now, instead of series, because the Arduino can source more current than a 74HCxx logic chip. Now I won't have to worry about equalizing the brightness of the LEDs.

- A digital output pin on an Arduino Uno (or Nano) can easily source 20mA. The absolute maximum is 40mA, but you should avoid reaching that limit.

When I tested a standard red LED at 5V with a 470-ohm series resistor, it drew only 6mA, but was acceptably bright. When an output pin drives two of these LEDs in parallel, the total current will be 12mA, so the Arduino will have no problem with this.

In Figure 30-1, you can see that the left-right pair of LEDs will be powered by Pin 1 of the microcontroller, Pin 2 will power the center LED, Pin 3 will power two corner LEDs, and Pin 4 will power the other two corner LEDs. Those are arbitrary choices that I made. Meanwhile, Pin 0 will be an input pin, connecting with the pushbutton, and negative-ground will go to the **GND** pin on the timer.

After you install the LEDs in a breadboard, don't connect negative ground to the Uno board just yet. It's safer to upload the program first, because the Uno may have a different program already installed in it, and I don't know how it may have configured the input/output pins.

- You must be careful that a circuit does not to apply any voltage to a digital pin that is configured as an output.

I'm only using pins 0 through 4 in this circuit. Should I ground the unused pins of the Arduino, in the same way as if I were using a logic chip?

Absolutely not!

- Unused pins of the Arduino must be allowed to float.

- If a pin is configured as an output, never connect it directly with ground. This can cause permanent damage.

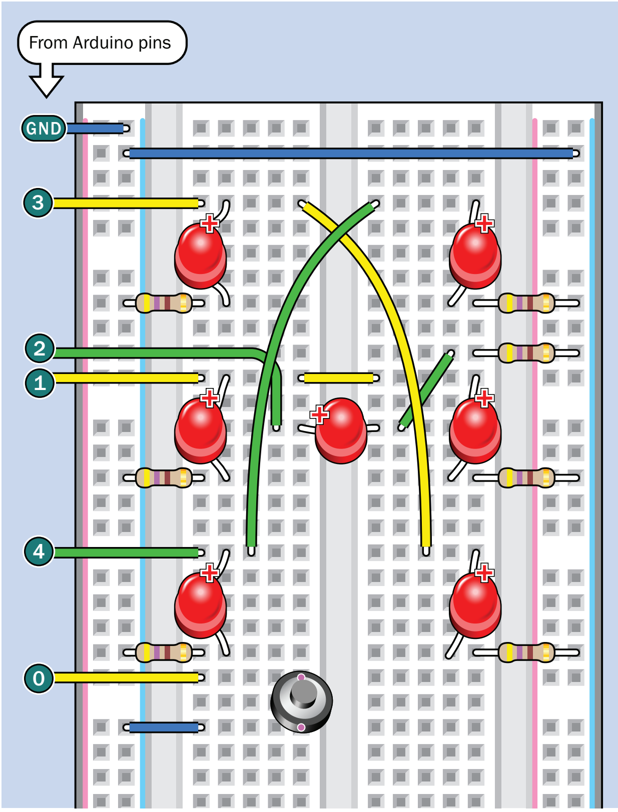


Figure 30-1. Seven LEDs wired on a breadboard to display the spot patterns of a die.

Now, the Program

Figure 30-2 shows the program that I wrote to match the pseudocode. First I will take you through the process of typing it, and then I will explain what the statements mean in Arduino C language. Of course, you need a comprehensive manual to learn the language properly; this is just a demo to suggest the possibilities.

Notice that on line 6, there is an underscore character between the word `INPUT` and the word `PULLUP`; you press a Shift key with the hyphen key to make an underscore.

On line 23 there are two pairs of vertical lines. Each line is sometimes called the *pipe* symbol. On a Windows keyboard, you are likely to find it above the Enter key. You create it by pressing Shift and the backslash \ character.

When you finish, choose the **Sketch > Verify/Compile** option in the IDE to see if you made any errors.

```
 1 int spots = 0;
 2 int outpin = 0;
 3 long ignore = 0;
 4
 5 void setup() {
 6   pinMode(0, INPUT_PULLUP);
 7   pinMode(1, OUTPUT);
 8   pinMode(2, OUTPUT);
 9   pinMode(3, OUTPUT);
10   pinMode(4, OUTPUT);
11   ignore = 2000 + millis();
12 }
13
14 void loop() {
15   for (outpin = 1; outpin < 5; outpin++)
16   { digitalWrite (outpin, LOW); }
17
18   spots = random (1, 7);
19
20   if (spots == 6)
21   { digitalWrite (1, HIGH); }
22
23   if (spots == 1 || spots == 3 || spots == 5)
24   { digitalWrite (2, HIGH); }
25
26   if (spots > 3)
27   { digitalWrite (3, HIGH); }
28
29   if (spots > 1)
30   { digitalWrite (4, HIGH); }
31
32   delay (20);
33
34   if (millis() > ignore && digitalRead(0) == LOW)
35   { checkbutton(); }
36 }
37
38 void checkbutton() {
39   delay (50);
40   while (digitalRead(0) == LOW)
41   { }
42   delay (50);
43   while (digitalRead(0) == HIGH)
44   { }
45   ignore = 2000 + millis();
46 }
```

Figure 30-2. Program listing for Nicer Dice.

Unfortunately Arduino C language uses a lot of punctuation which I find nonintuitive, and one extra brace symbol (or one too few) will prevent your code from working. You simply have to examine each line and compare it very carefully with my listing. The listing was copy-pasted directly from the Arduino IDE, immediately after I ran it successfully, so I feel confident that it's error-free.

When your **Verify/Compile** operation finds no additional error messages, "upload" the program. Now you can plug in the ground wire connecting your breadboard with the Uno board, and the LEDs should start flashing, because an Arduino always starts running its program automatically. Wait a couple of seconds, then press the button, and the display stops, displaying a random pattern. Press the button again, and the display resumes. Hold down the button, and after the two-second "ignore" period, the display stops again. The pseudocode has been successfully implemented!

Line by Line

The first three lines declare the variables that I invented for this program. The term `int` means *integer*, which is a whole number without any fractional value following a decimal point. In Arduino C, an integer must have a value in the range `-32,768` to `+32,767`.

Why such odd numbers? Because they are represented inside the microcontroller as 16-bit binary numbers, and that range happens to be the limit for 16 bits.

On line 1, `spots` is the name of a variable that will store a number from 1 to 6, representing the number of spots on the die.

On line 2, the `outpin` variable will store the value of each output pin that connects with one or two LEDs. When `outpin` has a value of 1, the microcontroller can switch the LEDs connected with Pin 1—and so on.

The word `long` means "long integer," which is stored in 32 bits, and can have a value from `-2,147,483,648` to `2,147,483,647`. I need a long integer to store the current value of the clock inside the microcontroller, because the clock uses a long integer, too, and its value may be greater than 32,767 milliseconds.

Why not use long integers for everything? Then I wouldn't have to worry about exceeding the limit for a regular integer. True, but a long integer takes twice as long to process (or more), and takes twice as much memory. We don't have a whole lot of memory on the Atmel microcontroller.

If you're wondering how I know all this, I read it in the language reference on the Arduino web site. You really have to read the documentation, to write your own programs—which is a topic I will discuss in the next chapter.

The `setup()` section, beginning on line 5, tells the microcontroller how I will be using each pin.

The word `PULLUP` tells the microcontroller to use an internal pullup resistor so that I don't have to install one with the button on the breadboard. The Arduino has pullup resistors internally, which is a nice feature, but it does not have pulldown resistors. This means that an input pin will be held high until the external button grounds it. A low state on the input pin means that the button has been pressed.

After the setup, you come to the word `for` on line 15. This is a very basic and convenient way to make the microcontroller count through a series of numbers, storing each new number in the `outpin` variable. The syntax works like this:

- The reserved word `for` tells the microcontroller to count from one value to another value, in steps that are specified.
- The first parameter following `for` is the initial value that will be stored in the `outpin` variable.
- The second parameter tells the microcontroller that it can continue counting, so long as it doesn't reach this limit. The < symbol means "less than," so this loop will continue counting while the outpin variable has a value less than 5. In other words, it will count from 1 through 4. Remember, I am using pins 1 through 4 to light the LEDs.
- The third parameter is the *increment*. That is, the amount that the loop adds to the `outpin` variable each time it cycles. Arduino C allows me to specify that by using two ++ symbols, so `outpin++` means, "add 1 to the outpin variable in each cycle."

A `for` loop allows you to specify all kinds of conditions. It is extremely flexible. This loop just counts from 1 through 4, but it could count as easily from 100 to 400, or to any range you like, limited by the type of the integer used in the loop (`int` or `long`).

After line 15 specifies the way in which the loop will run, a procedure follows on line 16, between a pair of braces, telling the microcontroller what to do during each cycle of the loop. There is only one operation in this procedure: write a `LOW` state to the pin specified by the variable `outpin`. Because `outpin` is going to count from one through four, the `for` loop is going to create a low output on digital pins 1 through 4.

Maybe now you can see what this is all about. The loop switches off all the LEDs by stepping through the output pins and making them go low.

Instead of a loop, I could have used these four separate commands:

```
digitalWrite (1, LOW);
digitalWrite (2, LOW);
digitalWrite (3, LOW);
digitalWrite (4, LOW);
```

But I wanted to introduce you to the concept of a `for` loop, because it's basic and important. What if you wanted to turn off nine LEDs? Or what if you wanted the microcontroller to flash an LED 100 times? A `for` loop is often the best way to make a procedure efficient, when it has to repeat itself.

After the `for` loop has zeroed the die display, on line 18 you get to the `random()` function, which chooses a number between the limits in the parentheses. I want a die value from 1 through 6, but the way the Arduino works is to choose a random number ranging from the lower value, to 1 less than the upper value.

Now that a random value has been copied into the `spots` variable, the program has to light an appropriate number of LEDs. The `if` statements deal with this.

On line 20, the first `if` is simple enough. If the die value that was chosen randomly is 6 spots, this will be the only occasion where a high value has to be sent out through Pin 1, which is connected with the middle-left and middle-right LEDs.

The double equals sign is used when the program has to make a comparison between two values. A single = is only used when a value is being assigned to a variable, as on line 18 in the listing.

Maybe you're wondering why I didn't write the program to light the corner spots as well as the middle spots when the spots value is 6. This is because the corner spots have to be switched on for other die values too, and it's more efficient to minimize the number of `if` tests. You'll soon see how this works.

The next `if`, on line 23, uses the pipe symbol that I mentioned previously. A pair of `||` symbols means `OR` in Arduino C. So, the function says that if we have a die value of 1, `OR` 3, `OR` 5, we light up the center LED, by putting a high state on Pin 2.

The third `if`, on line 26, uses the > symbol to mean that for any spots value greater than 3, light up one diagonal pair of spots.

On line 29, the program says that if the spots value is greater than 1, the other diagonally placed LEDs must be illuminated.

You can test the logic of these `if` tests by looking back at the spot patterns in the Nice Dice project, in Figure 23-10. The logic gates in that figure were chosen to fit the binary output from the counter chip, so they're slightly different from the logical operations in this program. Still, the LEDs are paired in the same way.

After the `if` functions, I inserted a delay of 20 milliseconds, because otherwise the LEDs flashed so rapidly, they looked as if they were on all the time. With the delay, you can see them flashing, but they're still too fast for you to stop at a number that you want—although, you can try!

You may wish to adjust the delay value to a number higher or lower than 20.

Now comes the important part: The `AND` test which can tell the microcontroller to visit the button routine. In Arduino C, two ampersands (written as &&) are interpreted as a logical `AND`. Line 34 can be written in plain English like this:

If (the system clock has advanced beyond the target value in the ignore variable, AND the button input on Pin 0 is low) . . . `checkbutton`.

What is "checkbutton"? It's the name I gave to the button routine. Where is it? I put it at the bottom of the program, on line 38, preceded by another `void` statement.

I didn't have to do it this way. I could have included the button routine in the main part of the program—but creating a separate section for each procedure is good practice, because it makes a program easier to understand. This will help other people to figure it out, and it will help you, too, when you take another look at it six months from now and can't remember how it works.

The concept of C language is that each part of a program is a separate block, and the program runs them by `calling` them when you want them. Think of each block of instructions as an obedient servant who only does one thing, such as washing the dishes or taking the garbage out. When you need that task to be executed, you just call the servant by name.

The blocks are properly known as `functions`, which you can write yourself. I decided that the correct way to structure this program was to split off the button routine into a function. I called it `checkbutton()` but I could have called it anything at all, so long as the word wasn't already being used for some other purpose.

`void checkbutton()` is the `header` for the function, after which the procedure is contained within braces, as usual.

Remember that `LOW` is the state on Pin 0 when the button is pressed; `HIGH` happens when it is released and the pullup resistor takes over. Also remember that the LEDs which have been illuminated will remain on while the microcontroller is dealing with the checkbutton function. The LEDs won't be switched off until the microcontroller goes back to the `for` loop on line 16, which switches them off.

I can describe the checkbutton function like this:

- Wait 50ms for the contact bounce in the pushbutton to stop after the player pressed the button.
- Now, while the button is being pressed, wait for it to be released.
- Wait another 50ms for the end of the contact bounce created by releasing the button.

(At this point the player is looking at the pattern of spots, and the program is waiting to start displaying more random patterns, as soon as the player presses the button again.)

- While the button is not being pressed, wait for it to be pressed again.
- Reset the `ignore` variable.

When the microcontroller gets to the end of the function, where does it go? Back to the line immediately after the one that called the function. Where is that? On line 36. The right-hand brace symbol on that line tells the microcontroller that this is the end of the `loop()` function, but the loop always repeats, so the action resumes on line 14, where the matching left-hand brace symbol is located. The LEDs are switched off, a new random number is chosen, and the loop function repeats.

The documentation for Arduino C, on the Arduino site, doesn't say anything about program structure, probably because it wants to get you started making things happen as simply as possible. So the Arduino simply forces you to use the mandatory **`setup()`** function, followed by the `loop()` function, and then it leaves you to learn about functions on your own. But structure is important, because as soon as a program begins to grow in size, you really need to subdivide it, to keep it from becoming a complicated mess. A standard C-language tutorial will explain this in more detail.

There is another advantage to dividing a program into functions. You can save the functions separately, and reuse them in other programs later. The `checkbutton` function could be reused in any game where you want to stop the action by pressing a button and then restart it by pressing the button a second time.

Likewise, you can use other people's functions in your own programs, provided the author doesn't restrict you from doing so by retaining copyright. You'll find them in the libraries that I have been referring to. Even though I think it's a great idea for people to write programs of their own, there's no shame in using a function that someone else wrote to do something boring and straightforward, such as copying text onto an LCD screen. There's a basic principle, here:

- It's good to be creative.

But:

- Don't reinvent the wheel.

Precautions

After you write a program and load it into your Arduino, you can continue to power it from the USB port, or you can set up the board with its own power supply. When taking this second option, I think it's a good idea to disconnect the USB connection before you add power from another source. If you apply voltage to the wrong pin, and your USB cable is still attached, there is a risk of damaging the USB interface chip inside your computer.

Some additional precautions may be worth your consideration. Search online for:

```
ten ways to destroy an arduino
```

—and you may find that it's easier to mess things up than you might imagine.

When using a Uno in conjunction with LEDs (or other components) on a breadboard, you'll have to deal with the issue of connecting everything together. The conventional arrangement is illustrated in Figure 30-3, using flexible jumper wires that have a plug at each end. You already know that I don't like them much, and this is another situation where they have potential to create problems. Plugging a jumper into a wrong socket on the Arduino board is just too easy, and creates the potential hazard of an output pin feeding back into another output, which your microcontroller would not like at all.

I believe that if you decide to write programs of your own, your best option is to buy a Nano, install it on a breadboard, and apply a 5VDC supply to it from an LM7805 regulator, exactly as if you were powering a logic chip. If you've built some of the chip-based circuits in Section Four of this book, you already know how to minimize the risk of wiring errors: Use red jumpers for positive power, blue or black for negative ground, and other colors elsewhere. These good habits should help you to avoid chip damage.

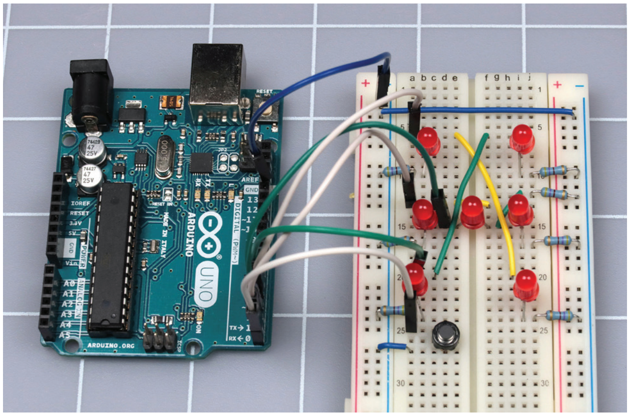


Figure 30-3. Wiring an Arduino to a breadboard with flexible jumpers is temptingly convenient, but does enable errors.

So much for my presentation on Arduino boards. Now, what if you want something different, such as a microcontroller that's more powerful, or a language that's easier to learn? I will try to address these options in the next and last chapter.

Chapter 31
The Learning Process

C language has never been regarded as the easiest way to learn programming. Fortunately, many alternatives exist, and in this chapter I'm going to suggest two of them. One is an obvious choice, while the other is not obvious at all, and may seem a bit odd. You may have noticed, though, that I don't mind being unconventional if I feel that something works.

I will also suggest two non-Arduino options for microcontroller hardware. One of those will be another odd choice—so let me deal with it first, before I end up with conventional choices that should make everyone happy.

Back to BASIC?

The BASIC computer language was introduced long, long ago, in 1964, to make programming more accessible to college students. Initially it included only a small set of commands, and every line had to begin with a line number, which you typed manually. Full-screen text editing did not exist, and the process of writing a program was archaic by today's standards.

Still, BASIC endured, because it had an easy learning curve and was quite versatile. Bill Gates liked it, so Microsoft developed a version. An extended version followed for the IBM-PC, and then in 1985 Microsoft released QuickBASIC, which was powerful enough for some business applications. Independent competitors such as PowerBASIC took the capabilities even further, including features that were useful in database programming.

During the 1990s, widespread adoption of Windows began to erode the market for dialects of BASIC that ran under MS-DOS. At this point the language might have faded into obscurity, except that Microsoft refused to give up on it. They introduced Visual BASIC to take full advantage of the Windows graphical user interface, and then they continued with substantial revisions that sustained a user base into the 21st century.

Today, many people seem unaware that BASIC still exists. It was never fashionable, partly because its name was a stigma. BASIC sounds—well—basic! And if you look up the acronym, it stands for Beginner's All-purpose Symbolic Instruction Code. It was designed for beginners, so why should we bother with it?

Nevertheless, advocates for the language have been quite stubborn, and in 1999 a British company calling itself Revolution Education created their own BASIC dialect for a range of microcontrollers. Their version included built-in features such as pulse-width modulation to control DC motors, I2C protocol for communication with other devices, analog-digital conversion, specific commands for servo-motor control, and of course management of high/low pin states.

Revolution Education named their development system PICaxe, because they chose the PIC series of microcontrollers as a platform. Perhaps they felt that a playful name was acceptable because they were marketing the modified chips for educational purposes, but the combination of "PICaxe" with "BASIC" didn't do much to erase the old stigma. Personally I didn't care about that; I just wanted something that worked.

In 2005 I was developing a fairly complicated rapid cooling device for a research lab in California. I needed a microcontroller to do the job, but I didn't want to program chips in assembly language, so my options at that time were limited. The Arduino products were of no use to me, as none of their boards had enough i/o pins for the six pumps, multiple switches and buttons, and more than a dozen temperature and pressure sensors that I was dealing with. Also, I knew that my clients would probably want me to modify the code on-site, and to do that using Arduino, I would have to move the microcontroller chip into and out of the development board.

The PICaxe system was the answer to all my requirements. One of their PICs had 40 pins, and all of their chips had a bootloader preinstalled so that you didn't need any ancillary hardware at all. Just install the PIC chip in a circuit board and use a custom USB cable to transfer the program into it, and you were good to go.

Revolution Education is still selling PICaxe products, and three of their chips are shown in Figure 31-1. I suppose I should mention that I am merely an end-user; I have no business or other relationship with the company or their employees.

My project was successful, but my clients asked for so many additional features, I had to build a much larger version. Could I still use PICaxe products? Yes, although I had to add another 40-pin chip and a lot more parts. I am including the board layout in Figure 31-2 just in case anyone is still skeptical that BASIC on a PIC microcontroller can be used for serious applications. Note that there are more than 50 i/o connections down the right-hand edge of the board.

An important factor enabling me to do this work was that the PICaxe documentation was excellent. My reasons for mentioning this will soon become obvious. Their web site provided:

- Complete language reference.
- A good and lengthy tutorial.
- Everything available in one place. No other web searches or browsing necessary.
- Tech support available with a 24-hour turnaround.

Their IDE also enabled me to run simulations of the code I wrote, before loading it into the microcontrollers. All these features are still available from Revolution Education at the time of writing.

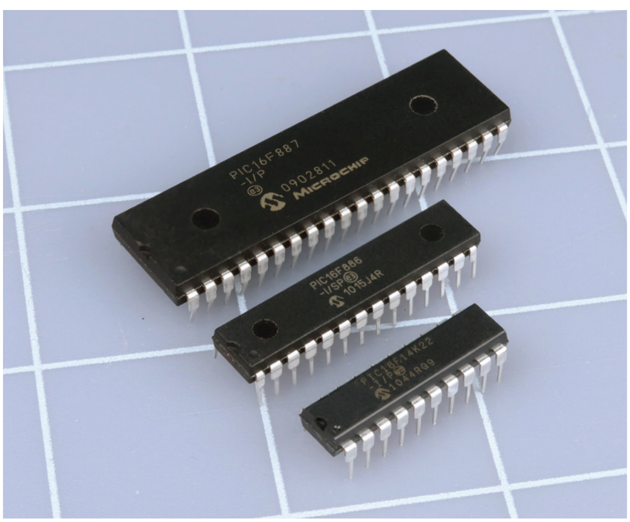
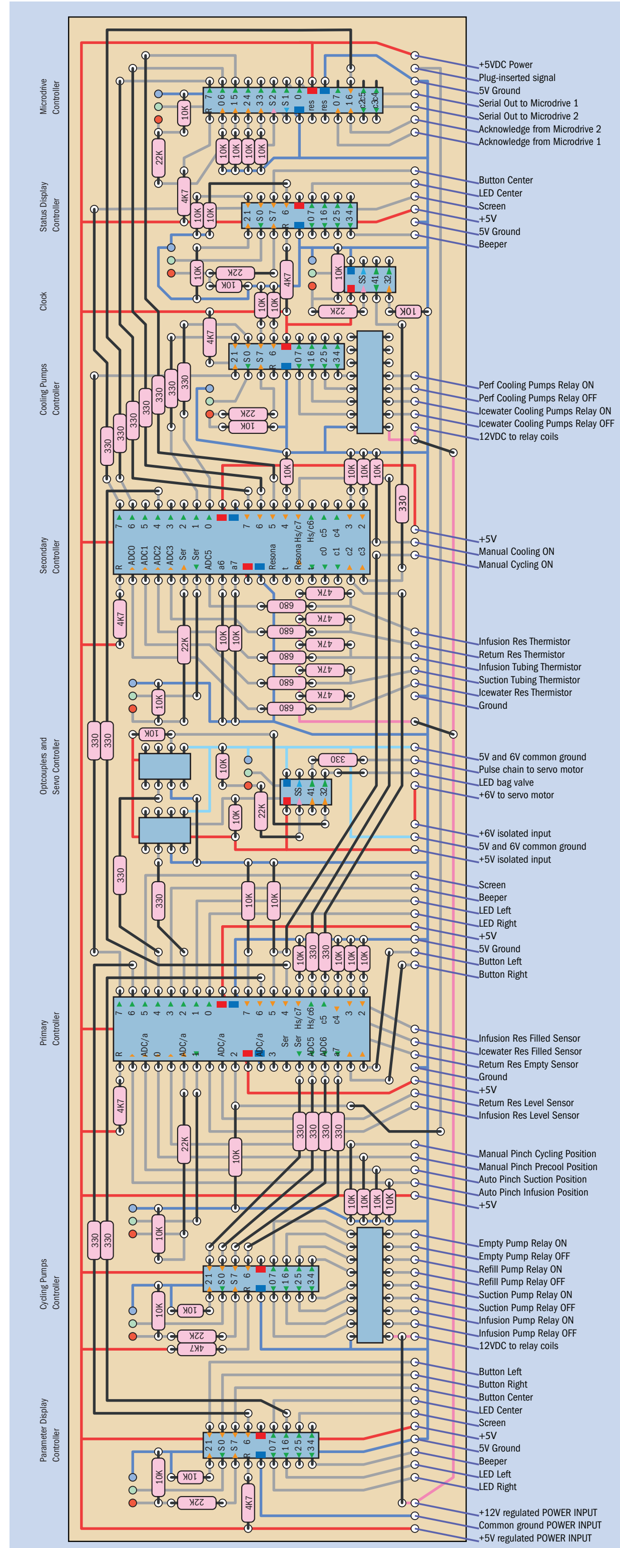


Figure 31-1 (above). With the PICaxe bootloader installed, these PIC microcontrollers can be programmed without using any hardware other than a special USB cable.

Figure 31-2 (right). A board layout using PICaxe chips to control a cooling system.



Limitations

Critics will argue that the old-school type of BASIC offered for the PICaxe is a dead end, and does not lend itself to well-structured code. I think you can write well-structured code in any language, if you try, but either way, I'm not sure that this is a big concern when you are programming a microcontroller. Typically, one block of code executes in a big loop. It reads sensor and button inputs, controls motors or LEDs, and then repeats. Maybe structure is not a very pressing issue when there isn't much of it.

What mattered to me when I started dealing with microcontrollers is that a language should be quick to write and easy to understand—especially if I might be coming back and looking at the code a year or two later. Here's a small snippet from my cooling-device controller:

```
checkinputs:

temp = icepin
if temp < icestatus then : outbits =_
  outbits or 1 : endif
if temp > icestatus then : outbits =_
  outbits or 2 : endif
icestatus = temp

if noperfcool = 1 then
  if perfstatus = 1 then
    perfstatus = 0
    outbits = outbits or 4
  endif
else
  temp = perfpin
  if temp < perfstatus then :_
    outbits = outbits or 4 : endif
  if temp > perfstatus then :_
    outbits = outbits or 8 : endif
  perfstatus = temp
endif
```

You won't know what the variable names mean or what the routines are for, but you can see that it's a series of simple instructions with minimal punctuation.

Critics may also complain that because the products from Revolution Education were not open-sourced, they never attracted a community of developers offering new, innovative products. This is true, but I saw it as an advantage. I liked dealing with a company which retained control over the hardware in addition the programming language. The documentation would not go out of date, and I would not have to worry about different versions evolving rapidly for different types of chips.

The old PIC chips are limited, by current standards. They don't have much memory, and they lack important features such as floating-point arithmetic (although a co-processor is available). Still, I think they remain a good choice if you are starting to explore this field, and you want to achieve something relatively simple without spending too much money. I don't think that the sophistication of microcontroller is very important in that situation.

Having stated my case, I'll leave it there, because programming languages are always a contentious topic, and I know that a lot of people will disagree with me. Let me move on to the most obvious alternative.

Python

The language named Python was created by a Dutch programmer named Guido van Rossum who happened to be a fan of the British comedy troupe, Monty Python's Flying Circus. Really it was another frivolous name, but Python filled some serious needs when it was released in 1991. According to the Python Institute, van Rossum wanted it to be:

- An easy and intuitive language, but just as powerful as those of the major competitors.
- Open source, so anyone can contribute to its development.
- Code that is as understandable as plain English.
- Suitable for everyday tasks, allowing for short development times.

In some ways this might have been a reaction against aspects of C++ language. Either way, the goals resonated with people, especially when Python acquired some advanced capabilities such as executing procedures via the Internet. The language became extremely popular and is widely taught in schools.

Python was written for desktop computers, and on its simplest level, it is as easy to understand as BASIC. Here's a snippet of code to count the number of words in a user input:

```
# Number of words
wds = input("Type some words, press Enter: ")
total = 1
for n in range(len(wds)):
  if(wds[n] == ' '):
    total = total + 1
print("Number of words = ", total)
```

The word "print" means "display on the screen," although it can be adapted to send output to a printing device. You can see that Python is informal and colloquial when you're using it to perform an elementary task like this.

There are obvious flaws in the routine that I have quoted, as it cannot deal with problems such as a person typing more than one space between each pair of words. Also, Python permits easier ways of accomplishing the same goal. I'm including the sample just for comparison purposes.

An equivalent routine in Microsoft BASIC shows an obvious family resemblance:

```
' Number of words
print"Type some words, press Enter:";
line input wds$
total = 1
for n=1 to len (wds$)
  if mid$(wds$,n,1)=" " then_
    total = total + 1
next
print"Number of words="; total
```

A longer sample of Python appears in Figure 31-3. This does exactly the same thing as the Nicer Dice program written in Arduino C which I reproduced in Figure 30-2. You can compare the two.

Really, when you are not writing a very long or complicated program, differences among languages can seem relatively small.

Your decision as to which language to use becomes more complicated if you are thinking that you may become serious about computer programming. In that case, learning Python would be a sensible career move.

But the biggest and most difficult question, assuming you have some interest in Python, is how can you learn

```
# Nicer Dice - Raspberry Pico MicroPython version

from machine import Pin
from utime import sleep, ticks_ms, ticks_add, ticks_diff
from urandom import randint

l1 = Pin(14, Pin.OUT) # middle, two leds board pin 19
l2 = Pin(15, Pin.OUT) # center, one led  board pin 20
l3 = Pin(13, Pin.OUT) # corner, two leds board pin 17
l4 = Pin(16, Pin.OUT) # corner, two leds board pin 21

button = Pin(21, Pin.IN, Pin.PULL_UP)

leds = [l1, l2, l3, l4]

ignore = ticks_add(ticks_ms(), 2000)

def checkbutton():
    global ignore
    sleep(0.050)
    while button.value() == 0:
        pass
    sleep(0.050)
    while button.value() == 1:
        pass
    ignore = ticks_add(ticks_ms(), 2000)

while True:
    spots = randint(1, 6)

    for l in leds:
        l.low()

    if spots == 6:
        l1.high()
    if spots == 1 or spots == 3 or spots == 5:
        l2.high()
    if spots > 3:
        l3.high()
    if spots > 1:
        l4.high()

    sleep(.020)
    if ticks_diff(ticks_ms(), ignore) > 0 and button.value() == 0:
        checkbutton()
```

Figure 31-3. The Nicer Dice program from Figure 30-2, where it was written in Arduino C, appears here rewritten in Python.

a version for microcontrollers in the most logical and time-efficient way, starting from zero experience of writing programs generally.

This is not as simple as it sounds.

In 2014, MicroPython was introduced specifically for microcontrollers, and I think most people would agree that it's easier to learn than any version of C. Therefore, I will start with that.

The micro:bit

A logical place to begin your learning process might be a development system intended purely for educational purposes, and the ***micro:bit*** satisfies that requirement. Developed in the UK by the BBC, it is described as a "tiny computer" but has the typical features of a microcontroller, such as inputs from pushbuttons and sensors.

The micro:bit can be programmed in three languages: Python, Scratch, and MakeCode. The Python is a version of MicroPython, with some features specific to micro:bit hardware, while Scratch (developed at MIT) is a visual language, and MakeCode is another visual language (developed by Microsoft). I'll deal with Scratch first, as it seems to be intended as the simplest possible entry point to programming.

Scratch displays code words as jigsaw pieces which you drag around on the screen of your desktop computer, to create the program that you will copy into your microcontroller. Because the pieces can only fit together in a limited number of ways, they prevent you from making errors. If you assemble the jigsaw, almost immediately you see animations moving on the screen. Figure 31-4 shows a sample Scratch screen of a program to control the movements of fish in the window.

My problem with Scratch is it seems to assume that a student has a short attention span and won't want to type anything. This strikes me as odd, in a world where people of all ages are busy sending text messages to each other during long periods of screen time.

More seriously, I think that purely visual languages underestimate the abilities of students, and they also create the misleading impression that programming can be a drag-and-drop operation with instant feedback. Moreover, a visual language does not enable an easy transition to more ambitious applications. It may be a good way to catch the attention of very young kids, but I don't think it's helpful as a step toward serious work.

With this in mind, I'll turn to the micro:bit implementation of Python, which is documented at microbit.org with a lot of tutorials. Mostly these consist of bite-sized code samples which you can copy-type or copy-paste to see the result. This approach may seem similar to my strategy in this book, where "Learning by Discovery" allows you to try something first before learning why it worked. However, the micro:bit materials tend to leave out the

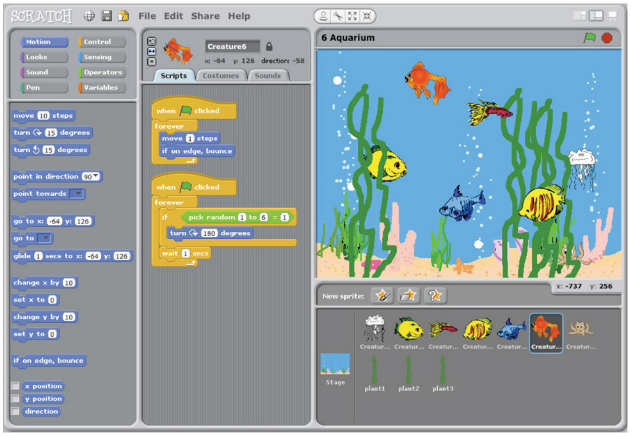


Figure 31-4. A screen shot of a program under construction in the Scratch language.

"why" part, which I think is a problem. To explain why, I'll use an example.

Many of the micro:bit samples use a `while True` loop. I managed to find only one tutorial which mentioned that statements below the first line of the loop have to be indented, but it didn't say why, and other aspects of the `while` instruction were not explained. For example, `True` has an initial capital letter, but `while` does not. Is the initial cap necessary? Is `True` some kind of special term, unlike the name of a variable? How can you tell the difference? What are the rules for choosing names of variables? What characters are permitted, and what are not? And how do you exit from a `while` loop, assuming you may want to?

These questions are not vital, so long as you just want to get a quick idea of what a program is; but as soon as you want to create your own code, the questions become very important indeed. To write software, you have to understand ***syntax***, which is the set of rules telling you how to compose instructions. You can think of syntax as being like grammar and spelling in a human language.

Suppose you were learning English for the first time, and you ran into words containing letters I and E such as ceiling, hierarchy, pier, piece, receipt, deceive . . . You would notice that sometimes the I precedes E, but sometimes it's the other way around, and you would wonder why.

There's a simple rule: "You write I before E, except after C." It's not always true, but it's better than having no guidance at all.

As for grammar in English, there are rules to guide you in using an apostrophe, or the difference between the verbs "lie" and "lay"—and so on.

If no one ever told you these rules, you could waste a lot of time inspecting samples of text and trying to figure everything out for yourself. Really, I think you would do better by learning in a structured, methodical way.

In a computer language, this is even more important. When you're writing a program, if you don't get everything exactly right, the code won't run, and you will be scolded with `syntax error` messages.

I don't think anyone can deny that syntax is essential, yet so far as I could tell, the introductory micro:bit tutorials don't mention it. In this respect, they are typical of many sources online which show you code samples but don't explain the rules you would have to follow to write your own.

An open-source advocate might say, "Oh, if you want to go beyond copying code samples, you can look up syntax easily enough. It's all freely available online."

True, but this brings me to another problem: There's a big gap between trivially simple code samples and formal documentation. Again and again, when you look for enlightenment on sites such as python.org, you find materials which suffer from a problem which I call "failing to explain the obvious."

I run into this myself, as a writer of educational texts. A writer has to be knowledgeable in the subject, but the more proficient you become, the harder it is to remember what it was like to know nothing at all. The result is that you forget to explain concepts which have become so familiar to you, they seem obvious.

If you already know some other programming language, or if you are accustomed to using Unix, the tutorials at python.org will be useful. If you don't have much prior knowledge, I think you may have to look elsewhere. That shouldn't be a problem, as there are so many other sources online; and yet in a way, this is itself a problem, because with such a large number of sites to choose from, how do you know which one to use?

Suppose you want the rules of syntax for a `while` loop. You could try this search term:

```
micropython while syntax
```

And you may end up, as I did, at

https://realpython.com/python-while-loop/

This actually turns out to be a very good and thorough summary, and one reason it is useful is that it's not bite-sized. However, it covers just that one topic, and because this is a Python site (even though I specified micropython as a search term), some topics specific to microcontrollers will be missing. For instance, the site won't tell you to make the state of an output pin change from low to high.

What I want (and perhaps what you want) can be summarized as "three ones":

- One well-organized resource, dedicated to one language, and specific to one piece of hardware.

The resource will contain code samples, but they will be followed by explanations which are organized so that they build a cumulative, structured understanding of the language. By the time you get to the end, you should be able to write the language yourself. Finding such a source for modern microcontrollers can be surprisingly hard.

My self-appointed role, in this chapter, is to serve as your guide if you want to learn to program microcontrollers, so I feel frustrated by having to tell you that I have not found an option that I can recommend wholeheartedly. I know people who have done it by looking at samples, comparing them, and trying to figure out the syntax that way, but it's not an efficient method.

Also, it has a significant down side: You end up with what I call knowledge voids.

I used to encounter this phenomenon when I taught introductory computer classes, where some of my students would arrive knowing bits of information that they had gleaned elsewhere. Inevitably, because they had gathered the fragments in a non-systematic way, they had missed some pieces. Their knowledge voids were like bubbles in concrete or holes in Swiss cheese, and the most troublesome aspect is that people are not aware of their own knowledge voids, because they cannot know what they don't know.

I spent three weeks researching materials relevant to Python and MicroPython, and I also bought hardware such as the pyboard, which is a MicroPython development platform. It arrived without any documentation at all, and the company that sold it to me had very little on their web site. They provided a link which took me to the MicroPython site, where I had some problems finding exactly what I needed to get the pyboard running, and—

I won't go down the list of all the sites I visited and products I tested, because this would serve no purpose. I'll just say that it was a frustrating experience.

However, along the way I decided that perhaps the RP2040 microcontroller may be a good option.

From Pi to Pico

In January 2021 the RP2040 was announced by Raspberry Pi, a British company known for a single-board computer that has sold millions of copies. They developed the chip in-house as a state-of-the-art microcontroller with an impressive specification, and they marketed it on a miniboard named the Pico, with a 264K of SRAM and 2MB of on-board flash memory.

The Pico looks approximately like the Arduino Nano but is much more powerful. The RP2040 chip is being licensed to other companies for their own products, but I will just deal with the Pico here.

The initial announcement seemed to acknowledge a certain unfulfilled need. It included the sentence: "Our ambition with RP2040 wasn't just to produce the best chip, but to support that chip with the best documentation."

The people at Raspberry Pi also published their own intro-level book, *Get Started with MicroPython on Raspberry Pi Pico*. Although the book features cartoon characters of small children, it does in fact enable you to get started.

A gap still remains between this ground-level book and the Raspberry Pi information pages, which tend to suffer from the "too obvious to explain" problem, but the company may rectify this shortcoming, and in the meantime the gap is being populated with books that are independently published. After sampling them in their Kindle editions, one in particular satisfies at least some of my requirements: *Programming the Raspberry Pi Pico in MicroPython* by Harry Fairhead and Mike James. (I have no contact or connection with the authors.)

This book uses all-MicroPython examples, and recommends downloading the most recent version of the language from a specific URL: "At the time of writing the file is called: rp2-pico-20210205-unstable-v1.14-8-g1f800cac3.uf2 but this is likely to change." I had to overcome my reluctance at installing a language with "unstable" in its filename. It's odd that electronics hardware is marketed in a finished state, while software seems to be perpetually under development.

The book gets very specific about everyday techniques to accomplish the most common tasks in microcontrollers: Receiving input, debouncing a pushbutton, using interrupts, pulse-width modulation, interfacing with other devices via an I2C bus (necessary for some types of sensors and text displays), and dealing with analog-digital conversion.

You may still want to know more about the Python language, but at least the code samples in this book are longer and have practical uses, while being thoroughly explained and easy to understand.

Here's a sample which receives a button press and determines if the user is holding it down for an extended period, which is a form of input that people have learned to use with mouse buttons and browsers:

```
s=0
while True:
  i=pinIn.value()
  t=time.ticks_add(time.
ticks_us(),1000*100)
  if s==0: #button not pushed
    if i:
      s=1
      tpush=t
    elif s==1: #button pushed
      if not i:
        s=0
        if time.ticks_diff(t, tpush) >
        2000000:
        print("Button held \n\r")
      else:
        print("Button pushed \n\r")
      else:
         s=0
  while time.ticks_us()<t:
    pass
```

By the time you read this, more MicroPython books should exist, specific to the Pico. In the meantime I will conclude tentatively that if you want a friendlier language than C, and you want to use a more powerful microcontroller, MicroPython on a Pico may have a promising future. Alternatively, you can still go back to the Arduino—and I retain my eccentric affection for the venerable PICaxe.

Beyond This Book

Turning away from microcontrollers, and going back to discrete components—if you've taken the time to work your way through the experiments in this book, what else is left for you to learn about hardware?

Here are some areas that I have not explored.

Mathematics. The process of Learning by Discovery is light on theory, and I have avoided most of the math that you'd be expected to learn in a more rigorous introductory course on electronics. If you have an aptitude for mathematics, it can provide a deeper insight into the way in which circuits work.

Alternating current. In this area, more math is involved. The way in which AC behaves is a fascinating topic, but it is not a trivial topic.

Surface-mount components. Some people enjoy the challenge of creating really tiny circuits using really tiny components. It's not something that I want to do, but if you have sufficient patience, the results can be amazing. Search online for

```
DIY surface-mount
```

and you'll find some instructional videos which almost make it look easy, if you have patience and steady hands (I have neither).

Vacuum tubes. At this point, they are mainly of historical interest. But there's something very special and beautiful about glass tubes containing heating elements that emit an orange glow, especially if you can enclose them in fancy cabinetwork. In the hands of a skilled craftsperson, tube amplifiers and radios become art objects. You have to deal with relatively high voltages, but if you are careful about that, I think the risk is tolerable.

Etched PCBs. After designing a circuit, you can create a layout of copper traces on a *printed circuit board*, often referred to as a *PCB*. You do this by executing drawings with computer software which saves in a format that PCB makers will recognize. You may even want to do your own etching, which is not especially difficult, but of course requires some specific materials and supplies.

Experimenting with high voltages. This is a topic that I didn't mention at all. Giant sparks of the type that Nikola Tesla liked to play with don't have any practical applications, and they entail safety issues—but they are stunningly impressive, and you can easily obtain the necessary information to build the equipment.

Op-amps and **MOSFET transistors** are important topics to explore. I regret that I didn't have space for them in this book.

Audio is an entire field with its own challenges, especially if you want to get into digital encoding of analog signals.

In Closing

I believe that the purpose of an introductory book is to open up a wide range of possibilities, leaving you to decide where you want to go.

Electronics allows you that choice, because almost any application—from robotics, to radio-controlled aircraft, to telecommunications, to computing hardware—can be pursued by just one person, working at home, with limited resources.

In the first edition of this book, my publisher was surprised that I wanted to include an email address where readers could contact me. I'm glad I did, because I have received testimonials from people who have done amazing things. Just recently, I was contacted by a reader in the UK named Assad Ebrahim who started reading *Make: Electronics* with very little knowledge of hardware, but has written a type of Forth language for controlling an Arduino interactively. He was able to devise a tiny Photo Frame program able to cycle through high-res pictures on a tiny OLED screen using less than 1KB of program memory. He demonstrated it to me during a Zoom call.

Personally, I'm not especially gifted when it comes to electronic hardware. I didn't learn it easily. My skill is to explain things by writing about them and making diagrams. If you happen to have a real aptitude for electronics, you should be able to go far beyond anything that I know how to do, just as Assad did. And and maybe some time you can tell me about your projects.

If this book opens up new possibilities for you, then it has fulfilled its purpose.

Appendix A

Specifications

This appendix provides detailed specifications and manufacturers' part numbers which will be useful if you decide to do your own shopping for tools, supplies, and electronic components. The tables here show the quantities of components and other consumable required to complete experiments 1 through 30 in the five sections of the book.

For recommendations regarding retail sources which are likely to stock the products, see Appendix B.

For photographs and introductory information, see the beginning of each section of this book. Items are listed here in the same sequence.

You Must Have

These items are essential throughout the book:

Multimeter, notebook, 9V power source (battery or AC adapter), alligator test leads, and basic tools.

If you wish to do soldering, as described in Section Three, you will need a soldering iron, solder, perforated board, and Helping Hands (or another device to hold your work). Magnifying glass, heat-shrink tube, and heat gun are recommended.

A breadboard will be essential from Section Two of the book onward, together with four colors of hookup wire (at least 10 feet of each color) to make jumpers.

Specifications for Section One

Multimeter

Manual-ranging recommended, able to measure voltage, current, resistance, and frequency, with continuity testing, transistor testing, and diode testing. Ability to measure capacitance is a plus. For a full discussion of meters, see page 2 onward.

Safety Glasses

Any manufacturer, lowest price available as the risk while doing electronics work is minimal. Quantity: 1 pair.

Test Leads

Each lead must have an alligator clip at each end. Short leads are preferred.

Example: Adafruit Short Wire Alligator Test Lead (set of 12), Adafruit product identifier 1592. Quantity: 5 minimum, ideally including 1 red, 1 blue or black, and 3 of some other color.

Battery

9V alkaline cell, any brand. Quantity: 2 minimum. Additional replacement batteries may be required depending on intensity and duration of use.

Battery Connector (optional)

For a 9V battery, this may be described as a *battery snap*. At one end of it are connectors to the terminals on a 9V battery, while the other end terminates in bared wires.

Example: Eagle Plastic Devices `121-0426/O-GR`, or Keystone Electronics `232`. Quantity: 1.

Fuse

Glass cartridge type, 2AG size (5mm approximate diameter), fast-blowing, 1A and 3A values, any voltage rating. You may substitute an automotive fuse from an auto parts store.

Examples of glass fuse: Littelfuse `0225001.MXP` or `0225003.MXP`. Quantity: At least 1 of each value.

Light-Emitting Diode: Generic Red

Any manufacturer. For sections one and two of the book, 5mm diameter is recommended, as it is easier to handle (also sold as `T1-3/4` size). For sections three, four, and five, 3mm are recommended to fit circuit layouts. Red is preferred because this wavelength requires lower forward voltage and current than some other colors, which is a useful attribute in circuits where LEDs are driven directly from logic chips.

Look for LEDs with high light output, measured in *millicandela*, abbreviated *mcd*. A value of `400` mcd is good. *Diffuse* LEDs may be more pleasant to look at than those encased in *water-clear* plastic. Examples: Kingbright `WP710A10SRD/D` or `/E` or `/F` (3mm), Kingbright `WP7113SRD/D` or `/E` or `/F` (5mm), or Lite-On `LTL-4263` (5mm).

Resistors

One-quarter-watt (250mW), 5% tolerance preferred, any lead length, assorted values. See the components listed in figures A-1, A-2, A-4, A-5, and A-6. Any manufacturer.

Mending Plates or Brackets

Size 1/2" x 1" minimum, must be described as *galvanized* or *zinc plated* (not brass, not stainless). Example: National Hardware 4-pack model `N226-761`. Quantity: 4 plates or brackets.

Components for Section One	Experiments					If re-used	If not re-used
	1	2	3	4	5		
9V battery	1	1	1	1		1	4
Generic red 5mm LED		2	1		1	2	4
Cartridge fuse rated 1A				1		1	1
Cartridge fuse rated 3A				1		1	1
Lemons (or juice)					2	2	2
Galvanized bracket					4	4	4
Copper object					4	4	4
15-ohm resistor		1		1		1	2
150-ohm resistor		1				1	1
470-ohm resistor		1	1			1	2
1K resistor			1			1	1
1.5K resistor		1	1			1	2
2.2K resistor			1			1	1
3.3K resistor			1			1	1

Figure A-1. This table shows quantities of components for experiments 1 through 5 in Section One of the book. The column headed "If re-used" assumes that you will be willing to re-use the parts from each previous experiment, within Section One only. The column headed "If not re-used" assumes that you will not re-use any component from any previous project.

Copper-Plated Coins

If copper-plated coins are unavailable where you live, any other copper objects will do, so long as their surface area is about the same as that of a small coin. In a crafts store, you'll find decorative objects; in a hardware store, you may be able to buy a short piece of copper pipe and cut it into sections with a hacksaw; and in an auto parts store, you may find copper-plated alligator clips. Bear in mind that the copper will be discolored by chemical reactions in this experiment.

Notebook

Any notebook with at least 50 blank unruled pages, large enough to enable diagrams as well as notes.

Specifications for Section Two

Miniature Screwdrivers

A set such as Stanley part number `66-052` containing Phillips and straight blades. Quantity: 1.

Small Long-Nosed Pliers

No longer than 5 inches. Cheapest available, any manufacturer. Quantity: 1.

Wire Cutters

No longer than 5 inches. Cheapest available, any manufacturer. Consider buying them in a set that also contains long-nosed pliers. Quantity: 1.

Flush Cutters (Optional)

Cheapest available, any manufacturer. Quantity: 1.

Sharp-Nosed Pliers (optional)

Typically 4 inches long. May be found in a set with pliers, wire cutters, and flush cutters. Often sold for jewelry making.

Example: Amazon product `B07QVPGX7H`. Quantity: 1.

Wire Strippers

Must be able to strip 22-gauge wire (also referred to as 22 AWG wire).

Example: Irwin wire stripping tool, manufacturer product ID `2078309`. Quantity: 1.

Breadboard

This may be described as ***solderless breadboard***, ***PCB board***, or ***prototyping board***. Must be dual-bus, with 800 holes (tie-points) minimum. A red stripe and a blue stripe must be printed alongside each bus to differentiate them clearly.

Example: Elegoo `MB-102` at Amazon, or `PRT-00112` at Sparkfun, or `2157706` at Jameco. Quantity: 1, although extras will be useful if you want to preserve a circuit that you have built instead of disassembling it to make room for the next one.

Components for Section Two	Experiments						If re-used	If not re-used
	6	7	8	9	10	11		
Alligator test leads	5	3					5	5
SPDT slide switch 0.2" pins	2						2	3
SPDT slide switch 0.1" pins					1		1	1
Generic red 5mm LED	1		2	1	1	2	2	7
DPDT 9VDC relay		2	1		1		2	4
Tactile switch		1	2	2	2	2	2	9
Transistor, 2N3904					1	6	6	7
Trimmer, 10K					1		1	1
Speaker, 8-ohm						1	1	1
100-ohm resistor			1	2	1	1	2	5
470-ohm resistor	1		1	1	2	1	2	6
1K resistor			2	2	1	1	2	6
4.7K resistor						4	4	4
10K resistor			1	2		1	2	3
33K resistor					1		1	1
47K resistor						2	2	2
100K resistor					1		1	1
330K resistor					1		1	1
470K resistor						2	2	2
10nF capacitor						3	3	3
1µF capacitor			1			2	2	3
47µF capacitor						1	1	1
100µF capacitor			1	1			1	2
1,000µF capacitor			1	1	1		1	3

Figure A-2. This table shows quantities of components for experiments 6 through 11 in Section Two of the book. The column headed "If re-used" assumes that you will be willing to re-use the parts from each previous experiment, within Section Two only. The column headed "If not re-used" assumes that you will not re-use any component from any previous project.

Hookup Wire

Sometimes referred to as ***bulk wire***. Any manufacturer, 22 gauge (also known as 22 AWG), copper or tinned copper, solid conductor, any voltage rating, minimum 10 feet each of red, blue, yellow, and green. Other colors can be substituted but won't match the diagrams in the book.

Example: Adafruit Hook-up Wire Set, item `1311`.

Stranded Wire (optional)

Any manufacturer, 22 gauge (also known as 22 AWG), copper or tinned copper, any voltage rating, minimum 25 feet, any color.

Jumpers (optional)

You may buy pre-made jumpers instead of making your own from hookup wire, although the colors will not match the diagrams in this book. If you buy a boxed assortment, one box will be sufficient. Any manufacturer. Use this search term, including the quote marks:

```
"jumper wire" assorted box
```

Slide Switch

For Experiment 6, used with alligator jumpers: Single pole, double throw (SPDT), ON-ON, any voltage rating, any current rating, any type of contacts, any type of terminals, **ideal terminal spacing 5mm or 0.2".**

Example: E-Switch `EG1201` or `EG1201A`. Quantity: 2.

For all other experiments: Single pole, double throw (SPDT), ON-ON, any voltage rating, any current rating, any type of contacts, solder-pin or through-hole terminals with **spacing of 0.1" or 2.5mm to fit breadboards**. Quantity: 3.

Tactile Switch

Miniature round pushbutton with two solder pins spaced 0.2". Any voltage, any current.

Examples: Alps `SKRGAED-010`, or TE Connectivity `FSM2JART`, or Panasonic `EVQ-PV205K`, or Eagle/Mountain Switch `TS7311T1601-EV`.

You may substitute rectangular (not square) tactile switches with two kinked pins spaced 6mm or 6.5mm if you are willing to straighten the pins with pliers, to insert them in a breadboard.

Examples: Panasonic `EVQ-PE605T` or C&K `PTS635SH50LFS`. Small variations in part numbers may refer to the color of switches or other unimportant attributes. See renderings in Figure 6-22. Quantity: 10.

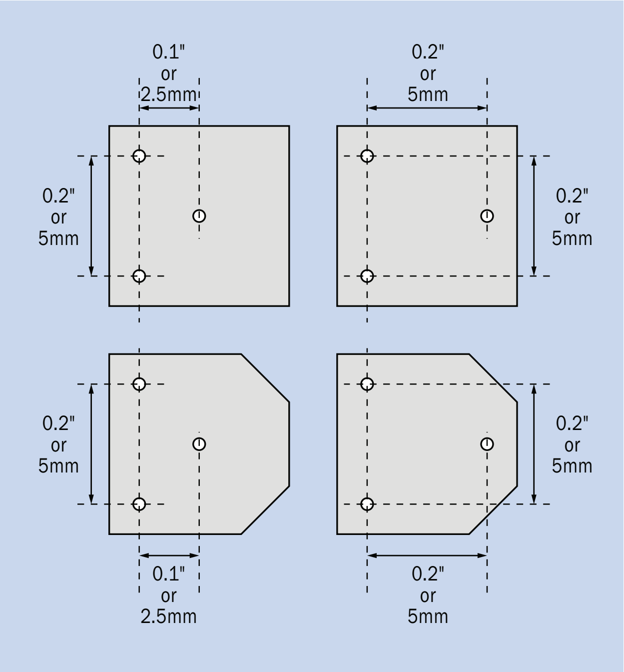


Figure A-3. Pin spacing requirements for trimmers.

Relay

DPDT, "2 Form C" type, with a coil rated for 9VDC, any contact rating, any switching current, non-latching, with solder pins.

Examples: Omron `G5V-2-H1-DC9`, or Omron `G5V-2-DC9`, or Fujitsu `RY-9W-K`, or Axicom `V23105-A5006-A201`.

If you substitute a different relay, you run the risk of its pin layout being incompatible. You will need to read the datasheet for the relay and compare the layout and the functions shown on page 63.

Trimmer Potentiometer

Sometimes these are listed as *trimpots*, although that is actually a trademark owned by Bourns. They may also be described as *trimmer resistors* or simply *trimmers*. For experiments in this book, here are the primary requirements for a trimmer that will fit the space I allocated on a breadboard:

Physical size: Diameter (if round) or edge length (if square) must be between 6mm and 8mm. It may also be classified as 1/4".

Single-turn, top-adjusted (not side-adjusted).

Acceptable pin layouts are shown inside the footprints of some trimmers in Figure A-3. Some trimmers have pins that are straight and rounded, while others are flat and kinked (see Figure 6-23). The straight pins slide more easily into a breadboard and may remain more secure. The kinked pins may be acceptable if you flatten the kinks a bit with some pliers.

Examples include TT Electronics `36F` or `36PR` series, Amphenol `N-6L50` series, Vishay `T7` series, and Bourns `3306F, K, P, W` series, or Bourns `3362F, H, P, R` series. Additional numbers and letters in a part number will vary depending on the resistance value and other factors.

Transistor

The `2N3904` NPN bipolar transistor is used throughout this book. Letters appended to the part number may describe its packaging, and may be ignored, so long as you buy a through-hole version, not surface-mount.

Example: On Semiconductor `2N3904BU`. Quantity: 10.

Capacitors

Ceramic capacitors are recommended for values up to and including 1μF, electrolytics above 1μF. You may substitute a ceramic capacitor instead of an electrolytic, but avoid using an electrolytic capacitor instead of a ceramic, unless you are sure that the direction of current will not reverse. The exact type of ceramic capacitor is not crucial, but look for 25VDC working voltage. For electrolytics, look for 12VDC working voltage (higher rating is okay, but the component will be larger and may cost more). See figures A-1, A-2, A-4, A-5, and A6 for values and quantities.

Loudspeaker

Also described as a *speaker*. If you are asked to specify a type, choose electromagnetic. Size 40mm to 50mm diameter (about 2"), rated 250mW (0.25W) or higher, 8 ohms impedance, ideally terminating in wire leads.

Example: CUI Devices `GF0501`. Quantity: 1.

Specifications for Section Three

AC Adapter

May be described as an *AC-DC Adapter*, but if not, check that it has a DC output, not AC. Avoid *universal* adapters with multiple selectable outputs, which may not be as well-regulated as single-output adapters. Output 9VDC, minimum 300mA (0.3A). The output wire can terminate in any plug, since you will remove it. Quantity: 1.

Low-Wattage Soldering Iron

For electronics work, should be rated 15W. Buy the smallest you can find. You may look for one bundled with a stand, so long as it is a real, heavy stand, not just a stamped piece of metal. The tip of the iron should be small enough to solder components in perforated board with hole spacing of 0.1".

Example: Weller `SP15NUS`. Quantity: 1.

Soldering Iron Stand

To hold a soldering iron safely. This may be incorporated with Helping Hands (see below).

Example: Weller `PH70`, although other brands may be cheaper. Quantity: 1.

Solder

Must be *electronic solder* with a *rosin core*. The thickness may range from 0.02" to 0.04" (0.5mm to 1mm). Quantity: If you only want to solder a couple of projects, three feet of solder will be sufficient.

Medium-Wattage Soldering Iron (optional)

For larger parts that absorb a lot of heat. Example: Weller Therma-Boost `TB100`. Quantity: 1.

Magnifying Lens

Any handheld magnifier approximately 1" diameter, so that it can be held close to the eye. A jeweler's loupe is acceptable.

Components for Section Three	Experiments			If re-used	If not re-used
	12	13	14		
Generic red 3mm LED		1	2	2	3
Transistor 2N3904			2	2	2
470-ohm resistor		1		1	1
4.7K resistor			2	2	2
470K resistor			2	2	2
1μF capacitor			2	2	2

Figure A-4. This table shows quantities of components for experiments 12 through 14 in Section Three of the book. The column headed "If re-used" assumes that you will be willing to re-use the parts from each previous experiment, within Section Three only. The column headed "If not re-used" assumes that you will not re-use any component from any previous project.

Helping Hands (optional)

Many variants exist. To avoid irrelevant hits, use a search term with quote marks, such as

```
"helping hands" electronics
```

Choose whichever version you like the look of. Sometimes a soldering iron stand is included, which is useful. A magnifying lens may be included, but is not so useful, as it cannot be hand-held and provides less than x2 magnification.

Plywood (optional)

To protect your work surface when soldering. Any piece of plywood or similar composition board, at least 1/4" thick.

Copper Alligator Clips (optional)

Search for the three words, inside quote marks. Copper sinks heat more effectively than steel, but if you snip an alligator clip from one of your test leads it should be reasonably effective.

Perforated Board (unplated)

Only required in Experiment 14. Do not use plated board for this experiment. If your search is unproductive, try:

```
unplated prototyping board
```

It may also be called ***phenolic board***. Unplated board is also known as ***unclad*** or ***untraced***. Vectorboard by Vector Electronics is an example, but relatively expensive. Quantity: One piece approximately 4" x 8" will be enough for three small projects.

Perforated Board (plated)

This type of board is used for the finished version of Experiment 18, but you can use it for other projects where you want to make a permanent duplicate of a breadboarded circuit. For convenience, use the type that has copper traces in the same pattern as connections inside a breadboard.

Try BusBoard `SB830`, GC Electronics `22-508`, or go to Adafruit and look for ***Perma-Proto***. Quantity: 1 for each project that you want to make permanent.

Heat-Shrink Tubing

Prices vary widely; eBay may be your best source for an assortment. Use 1/4" and 3/8" tubing for most small projects (it shrinks to approximately 50% of its cold diameter). Quantity: About 24" of each size (may be sold in small pieces). Any colors.

Heat Gun (optional)

You can use a hair dryer to activate heat-shrink tube, especially if you add a cone-shaped paper nozzle to direct the air, but a heat-gun works better. Buy the cheapest, smallest you can find, for electronics work. I don't recommend cordless heat guns, as they are heavier and cost more.

Examples: NTE `HG-300D`, or Wagner Furno `300`. Quantity: 1.

Hooked Meter Leads (optional)

For your multimeter, each lead has a tiny spring-loaded hook at one end and a plug for the meter at the other end. Search for ***mini-grabbers*** but be careful not to buy test leads that have hooks at both ends. Quantity: 1 black, 1 red, usually sold as a pair.

Machine Screws (bolts) (optional)

To attach perforated board to the inside of a project box, use #3 or #4 size flat-headed bolts in lengths of 3/8" and 1/2". My favorite source for this kind of hardware is McMaster-Carr. Buy an equal number of nylon-insert locknuts.

Project Boxes (optional)

Also known as ***plastic enclosures***. To narrow your search, I suggest this term, including quote marks:

```
"project box" electronics
```

The cheapest and easiest to use are made of ABS plastic. Quantity: 1 for each permanent project.

Headers (optional)

Strips of tiny plugs and sockets. Snap off as many as you need.

Examples: Mill-Max `800-10-064-10001000` and `801-93-050-10-001000` from large electronics suppliers, or 3M `929974-01-36RK` and `929834-01-36-RK`. Quantity: 1 strip of 64 plugs, 1 strip of 64 sockets.

Specifications for Section Four

Integrated Circuit Chips

See the first pages of Section Four for a discussion of chips. While all the chips you will need are listed in Figure A-5, it's a good idea to buy an extra chip of each type, as they can be damaged by incorrect voltage, reversed polarity, overloaded outputs, or static electricity.

Any manufacturer is acceptable. The ***package*** of a chip refers to its physical size, and may also be referred to as its ***form factor***. This attribute should be checked carefully when ordering, to avoid buying surface-mount chips by mistake. All logic chips must be in a ***through-hole DIP*** package (meaning a dual-inline package with two rows of pins that have 0.1" spacing). This may also be referred to as ***PDIP*** (meaning a plastic dual-inline package). The DIP and PDIP descriptors may be appended with the number of pins, as in DIP-14 or PDIP-16.

Surface-mount chips will usually have packaging descriptors beginning with S, as in SOT or SSOP. Do not buy any chips with "S" type packages.

Two chip families are used in this book: the 4xxx family and 74HCxx family. Part numbers will have additional letters or numbers added by individual manufacturers as prefixes or suffixes, as in `SN74HC00DBR` (a Texas Instruments chip) or `MC74HC00ADG` (from On Semiconductor). These versions are functionally identical. Look carefully, and you will see the `74HC00` generic number embedded in these proprietary numbers. When searching catalogs, use the part number without prefix and suffix abbreviations; for example, `74HC08` instead of `SN74HC08N`.

Old TTL logic chips, such as the `74LSxx` series, have compatibility issues. They are not used or recommended for any of the projects in this book.

555 Timer

Be careful to get the ***TTL*** version (also known as the ***bipolar*** version), not the CMOS version, for projects in this book. Here are some guidelines:

The TTL version often states "TTL" or "bipolar" in its datasheet, specifies a minimum power supply of no less than 4.5V or 5V, specifies an inactive current consumption of at least 3mA, and will source or sink 200mA. Part numbers often begin with `LM555`, `NA555`, `NE555`, `SA555`, or `SE555`. If you search by price, the TTL versions of the 555 timer are usually half the price of the CMOS versions.

If in doubt, check the datasheet. The CMOS versions always state "CMOS" on the first pages of their datasheets, and they allow a lower minimum power supply of 2V in most cases. They claim an inactive current consumption in microamps (not milliamps), and will not source or sink more than 100mA. Part numbers include `TLC555`, `ICM7555`, and `ALD7555`.

Seven-Segment Display

The display used in Experiment 19 must be an LED device, height 0.56", low-current red preferred, with common cathode, able to function at 2.2V forward voltage and 5mA forward current.

Examples include Broadcom/Avago `HDSP-513E`, or Lite-On `LTS-547AHR`, or Inolux `INND-TS56RCB`, or Kingbright `SC56-21EWA`. See Figure 18-1 for dimensions and pin spacing.

Components for Section Four	Experiments									If re-used	If not re-used
	15	16	17	18	19	20	21	22	23		
555 timer	1	2	3	3		1	2		1	3	13
SPDT slide switch	1		1				1	1		1	4
Trimmer, 500K	1	1								1	2
Tactile switch	2		1	3	2	9	2		2	9	21
Generic red 3mm LED	1		4	3	1	1	3	2	14	14	29
Speaker, 8-ohm		1								1	1
Trimmer, 10K		2		1						2	3
Diode, 1N4148		1	1							1	2
Counter, 4026B				3						3	3
7-segment LED digit				3						3	3
Regulator, LM7805					1	1	1	1	1	1	5
AND chip, 74HC08					1	2			1	2	4
OR chip, 74HC32					1		1		1	1	3
9V battery						1				1	1
DPDT 9V relay						1				1	1
Transistor, 2N3904						1				1	1
NOR chip, 74HC02								1		1	1
Counter, 74HC393									1	1	1
NOR chip, 74HC27									1	1	1
100-ohm resistor		1								1	1
470-ohm resistor	1		4	2		1	3		6	6	17
1K resistor		3		3	1			2	4	4	13
4.7K resistor				1						1	1
10K resistor	3	3	5	5	2	9	3	2	2	9	34
47K resistor			4	1						4	5
100K resistor						1			1	1	2
470K resistor		1	2	2						2	5
0.01µF capacitor	1	2	2	2		1			2	2	10
0.047µF capacitor				1						1	1
0.1µF capacitor					1	1	3	1	2	3	8
0.47µF capacitor		1	2	2	1	2	1	1	1	2	11
10µF capacitor	1	1	3	1		1			1	3	8
100µF capacitor		1	2	1					1	2	5

Figure A-5. This table shows quantities of components for experiments 15 through 23 in Section Four of the book. The column headed "If re-used" assumes that you will be willing to re-use the parts from each previous experiment, within Section Four only. The column headed "If not re-used" assumes that you will not re-use any component from any previous project.

Components for Section Five	Experiments							If re-used	If not re-used
	24	25	26	27	28	29	30		
22 gauge hookup wire, 25 feet	1			1				1	1
Paperclip	1							1	1
Cylindrical neodym magnet 3/16" x 1.5"		1						1	1
22/24/26 gauge wire, 100 ft spool		2		1	1			2	4
Generic red 3mm LED		1		2		1	7	7	11
1,000µF electrolytic capacitor		1		1				1	2
Diode, 1N4148		1						1	1
Optional cylindrical magnet 3/4" x 1"		1						1	1
Optional magnet wire 26 gauge 1/4 lb		1						1	1
Speaker, cheapest available, 2"			1		1			1	2
47 ohms resistor				2				2	2
470 ohms resistor				1		1	7	7	9
Tactile switch				2				2	2
High-impedance earphone					1			1	1
16 gauge wire, 50 to 100 ft					1			1	1
Poly rope, 10 ft package					1			1	1
Germanium diode					1			1	1
Alligator test lead					4			4	4
Optional LM386 amplifier chip					1			1	1
Optional passive piezo speaker or alert					1			1	1
Arduino Uno or Nano						1	1	1	2
USB cable to fit Uno or Nano						1	1	1	2

Figure A-6. This table shows quantities of components for experiments 24 through 30 in Section Five of the book. The column headed "If re-used" assumes that you will be willing to re-use the parts from each previous experiment, within Section Five only. The column headed "If not re-used" assumes that you will not re-use any component from any previous project.

Specifications for Section Five

Neodymium Magnets

I suggest K&J Magnetics as a source of supply, as the site has many variants and maintains a very informative primer on magnets. The link:

`www.kjmagnetics.com/neomaginfo.asp`

16-Gauge Wire

This is only required for the antenna in Experiment 31. If the cost is prohibitive, try 50 or 100 feet of 22-gauge wire. If you live relatively close to an AM radio station, it should be adequate.

High-Impedance Earphone

Only required for Experiment 28, this must be high-impedance type and can be ordered from sources such as

`www.scitoyscatalog.com`
`www.mikeselectronicparts.com`

Germanium Diode

Available from the same sources as the high-impedance earphone, above. Some may also be available from large suppliers Digikey, Mouser, or Newark. Ideally it will be described as "suitable for crystal radio receiver."

Arduino Uno Board or Arduino Nano Board

Widely available from sources ranging from Mouser to Amazon.

Appendix B

Sources

Kits

At this time, I am aware of two suppliers in the United States who have stated that they offer kits specifically for the experiments in this book:

protechtrader

`www.protechtrader.com/Make-Electronics-Kits-3rd-Edition`

Chaney Electronics

`www.goldmine-elec-products.com/make3/`

I will be verifying the contents of kits from these suppliers, but I have no financial or other relationship with them, and if you have questions about the kits you should contact the suppliers directly.

I regret that I don't know of any kit suppliers in countries outside of the United States.

WARNING: Any kits sold for the first edition of *Make: Electronics* will not be entirely compatible with this third edition. Some parts that you need may not be included, and some parts that you don't need may be included. Please check for the words "Third Edition" when you are shopping for kits.

Additional Parts

Protechtrader is now offering parts that you can buy individually for the specific projects in this book:

`https://www.protechtrader.com/electronic-components-kits`

Scroll down to the numbers at the bottom, to see additional pages.

Protechtrader also sells the bare-bones meter shown in Figure 1-1 which I think is good value and should be sufficient for the projects in this book, although I have not tested it over a prolonged period.

The company has introduced its own brand of AC adapters, including one that has a fixed output of 9V.

Chaney Electronics sells a very wide range of components, including some surplus and discounted items, and their own kit projects:

`www.goldmine-elec-products.com/chaney-electronics/`

For kits and parts of various kinds, you may also visit

`www.makershed.com/collections/electronics`

Neighborhood Shopping

All of the *components and supplies* for the experiments in this book are best acquired online, as I will describe below. You can buy all the *equipment* online, too, but in some cases, especially tools, you may prefer to shop in a physical store where you can handle and inspect the merchandise.

I suggest three types of stores: Hardware, auto parts, and crafts. Maybe you have never been into a crafts store—or an auto parts store, if you don't maintain a car. They can be useful options when you don't want to wait for a mail-order delivery.

The table in Figure B-1 suggests the types of bricks-and-mortar stores which I think are the most promising sources for the items listed. In cases where I have not listed a product, I believe you are more likely to find it online. For example, I have listed stranded 22-gauge wire, which may be found in a hardware store, but I have not listed solid 22-gauge wire, because it is an uncommon item in retail outlets.

Neighborhood sources for equipment and supplies	Hardware store	Auto parts store	Crafts store
Multimeter	●	●	
Safety glasses	●		
1A or 3A fuse		●	
Mending plate or bracket	●		
Miniature screwdrivers		●	●
Small long-nosed pliers	●	●	●
Sharp-nosed pliers			●
Wire cutters	●	●	●
Wire strippers	●	●	
Stranded 22AWG wire	●	●	
30W soldering iron	●	●	
Soldering iron stand	●		
Rosin-core solder	●	●	
Wearable magnifying lens			●
Heat gun	●	●	
Wire nuts	●		
Deburring tool	●		
Reamer	●		
Heat-shrink tube	●	●	
Plywood	●		●
Machine screws	●	●	
Project box			●

Popular chains of stores in the United States		
Hardware	**Auto parts**	**Crafts**
Home Depot Lowe's Ace Hardware True Value Harbor Freight Tools	Autozone O'Reilly NAPA Advanced Carquest	Michael's Hobby Lobby Jo-Ann Spencer Gifts A. C. Moore

Figure B-1. Retail outlets where you may find items listed.

Online Sources for Tools and Supplies

If you are shopping online for tools and supplies, I leave it to you to do a basic search for each product and decide where you want to place an order. There are hundreds of sites to choose from, and I don't have any specific preferences—with one exception, which is McMaster-Carr. They have an immense range of every conceivable type of hardware, including hard-to-find items, raw materials, and tools that you have probably never heard of:

`www.mcmaster.com`

Their site also features dimensioned CAD drawings of most of their products, and excellent tutorials. The company maintains instantly available and fully-informed telephone support, and it is fanatical about prompt delivery. Prices may be a bit higher than you will find at their competitors, but are not excessive for bulk items.

Online Sources for Electronic Components

Major Suppliers

Three major suppliers offer huge selections of components, and do not generally require you to buy minimum quantities. You really can buy a single transistor, if you so wish, and it will come in its individually labeled plastic bag. The names are:

Mouser

`www.mouser.com`

Digikey

`www.digikey.com`

Newark

`www.newark.com`

Shipping is not free, so try to buy as many different components as possible at one time. Most parts are so small and light, a lot of them will fit into one minimum-size box.

Another useful source is eBay, which can save you money by connecting you directly with Asian suppliers. This is especially useful when you want assortments. For example, I buy assorted resistors and capacitors on eBay. The site is also useful for supplies such as wire or solder, or components that are relatively obscure or have become obsolete.

eBay

`www.ebay.com`

At the time of writing, eBay is encountering competitors which may eat into their market share. You'll have to check into this yourself, as I don't know if they'll exist in a year or two. Try this search term:

```
alternatives to ebay
```

Smaller Suppliers

Smaller sellers offer unique advantages. Searching their inventory is quicker and easier, as there are fewer products. They are more oriented toward the needs of hobbyists and makers, so you won't have to dig for what you want among items that are inappropriate. Some small vendors offer older, surplus items far more cheaply than you can find them from the big suppliers. Here are the sites that I use fairly often:

Adafruit

`http://www.adafruit.com/`

Parallax

`www.parallax.com/`

Sparkfun

`www.sparkfun.com/`

Robot Shop

`www.robotshop.com/`

Pololu

`www.pololu.com/`

Jameco

`www.jameco.com/`

For surplus products and bargain bundles, try:

All Electronics

`www.allelectronics.com/`

Electronic Goldmine

`www.goldmine-elec-products.com/`

Asian Suppliers

The advantage of ordering directly from vendors in China or Vietnam is obvious: Low prices. You will have to wait 10 to 14 days for the package to arrive, but in my experience, the packages always do arrive.

If you check AliExpress or Utsource, each is a gateway to many different sellers:

AliExpress

`www.aliexpress.com/category/515/electronics-stocks.html`

Utsource

`www.utsource.net/category/elec-component-1.html`

Online Search Strategies

Searching for components can be a little more challenging than everyday searches. I'll begin with three examples of searches, and then I will summarize some general principles.

I will be pasting in dollar amounts from 2021 for demonstration purposes; the amounts may have changed significantly by the time you read this.

First Search

I'll start with an easy one. Suppose you want a 2N3904 transistor

You have two options: (1) Begin your search from a general-purpose search engine such as Google, or (2) Go to a particular supplier and search inside their site.

I'll try Google, using this search term:

```
buy 2N3904
```

The word "buy" is necessary to filter out all the sites online which just want to provide information about the transistor.

The first hit is an ad from the Utsource web site, wanting to sell me 10 transistors for 3 cents each. Can this be correct? Yes, but read the fine print: Shipping is free by FedEx, but only if I buy $150 of merchandise. Well, now I'm in Utsource, I'll use their search window to find more sources of 2N3904 transistors. And—there are hundreds of them. Literally. It's going to take a while to do comparison shopping here. Also, even if I specify postal service instead of FedEx, it will cost more than $10. And of course it will take a while.

I will back up now and go to Mouser. In the search box at the top of their home page, I type in

```
2N3904
```

The site offers me a lot of part numbers with extra letters on the end. They are the `2N3904BU`, `2N3904TA`, `2N3904TF`—and many more. This happens a lot when searching for components of this type. Here's a simple rule:

- If there is a search result without extra letters at the end, you can choose that one.

So, click on the first hit, `2N3904`, and the site finds 17 variants for me. Now I have to narrow the search via a series of choice-boxes asking me which attributes I want. These are known as "filters," and I will go through them from left to right.

Manufacturer? I don't care.

Mounting style? This is important. Most electronic components are now sold for automated assembly using ***surface mount*** technology. Fortunately, most abbreviations for surface-mount components begin with letter S. So now we have another simple rule:

- Don't select any mounting style or package beginning with S.

For the transistor, we have "through-hole" as a style. Is through-hole okay? Yes, almost every project in this book will entail poking the leads of components into little holes in some kind of board. So, another rule:

- If you have the choice, always select ***through-hole*** components.

Alternatively:

Through-hole components may be described as having ***solder pins***. This is another acceptable option, as pins will be okay for a breadboard.

Click the through-hole option, and then click the button that says Apply Filters. Now I'm down to 10 hits. Do I care about the remaining boxes? No, no, no . . . until I get to "Series." I want `2N3904`, not `2N39`, so I click `2N3904` and then, once again, Apply Filters.

Now I have just seven hits to choose from, showing fractionally different transistors. Do I care whether the leads are straight or kinked? No. So I look at the column headed Pricing, and I click the icon beside it, which sorts the list from low price to high price. The lowest price is 20 cents, which is a whole lot more than on Utsource, but I will receive the parts much more quickly. I enter the quantity I want, I click the Buy button, and the transistors are added to my cart.

There are other alternatives. For instance, I can go to eBay and search for 2N3904. The first hit is 20 transistors for $1.88 with free shipping, from within the United States. So that's much cheaper than Mouser.

However, if I want more components, and I use eBay, I will probably end up buying them from multiple sellers, and some will take longer to show up than others. This becomes confusing, and shipping charges may be involved.

Well, how about if I go to Amazon and search for `2N3904.` Here I get a hit selling 200 pieces for $5.80, with free shipping. That's only 3 cents each! What a deal! Of course I have to buy 200, and in my whole life I may only use maybe 10 of them. Maybe not such a deal after all.

So what do I do? Personally I do most of my buying on Mouser (or Digikey, or Newark) because ultimately it saves time, and I am very familiar with their user interface, and my order shows up in one package, always within three days. It's a matter of personal preference.

You may be wondering: How do you know that the transistors offered on Utsource, eBay, or Amazon will be exactly the right ones? At Mouser I had to reject surface-mount versions. Will I run the risk of getting tiny, unusable versions from the discount sources?

This is extremely unlikely. Another simple rule for you:

- Discount component sources don't usually offer surface-mount components. If they do, they will probably say so.

From discount sources, you will usually get the version of each component most commonly used by individuals in hobby-electronics, and they will be what you want.

Search Two: More Complexities

The first search was easy because I had a part number, and the part number allowed very few variations. Life is not always so simple. Here's an example of a real-life search that I had to pursue: I wanted a counter with a 3-bit output for use in the "Nice Dice" circuit in Experiment 23. (If you don't know what a counter is, that's okay. I just want to demonstrate the search process.) First I went to Mouser and entered my search term:

```
counter
```

While I was typing it, Mouser suggested an autocomplete: "Counter ICs."

An IC is an integrated circuit, which is the same thing as a chip. So I clicked the autocomplete suggestion, which took me to a page suggesting 821 hits.

The Mounting Style filter gave me the same options as when I was looking for a transistor: "SMD/SMT" (which are surface mount chips) and "Through-Hole." I clicked "Through Hole" and clicked the Apply Filters button. This reduced my options to 177 hits.

All the logic chips in this book are `HC` type in the `7400` family. I went to the Logic Family filter, and clicked `74HC`. But, not so fast! There's another little rule you should know:

- Component suppliers often list the same thing under different names.

I think this is because the vendors employ people to transcribe manufacturers' datasheets into the company database, and these people don't know a whole lot about electronics.

So, I scrolled down through more options in the Logic Family filter, sure enough I found HC listed separately from 74HC. Now what? Easy:

Hold down Ctrl and click additional items in a list to select more than one.

Of course on a Mac, you may Command-click.

Now I had 52 HC chips to choose from, so I had to do some more filtering. In the list headed "Counter Type," I selected "Binary," because I wanted a binary output. This left me with 33 remaining matches.

There were no 3-bit chips, but I could use a 4-bit chip and ignore the highest bit. I saw two options in Number of Bits: 4 and `4-bit`. Another example of two terms meaning the same thing. I Ctrl-clicked to select both.

Counting sequence could be "Up," or "Up/Down." I only wanted to count up, so I clicked that. Now only 9 matches left! Time to click Apply Filters and inspect the results.

For this book, I always try to use the most popular components, because they have a lower chance of becoming obsolete. I can make this choice by picking the chip which has the largest inventory at Mouser. I saw more than 7,000 of the SN74HC393N by Texas Instruments, so I chose that. Note that the letters preceding the chip number merely identify the manufacturer, and are not important. Was I done? Not quite! There is one more rule:

- When you don't have a part number, always check the datasheet.

Initially I didn't have the part number for this chip, and I had no experience in using it, so I clicked the datasheet link to make sure it would do what I wanted. A 14-pin chip providing maximum continuous output current of plus-or-minus 4mA with a nominal 5-volt supply—blah, blah, blah—oh, but wait! This chip actually contained two 4-bit counters, and I only needed one. Well, I wasn't going to quibble over that, and in fact I realized I could make use of the second counter in the chip if I enlarged the scope of my project.

The SN74HC393N would cost me about 50 cents. Might as well put six of them in the shopping cart. That's only $3, so maybe I should look for something else, reasonably small and light, so that I could add it without paying any additional shipping charge. First, however, I printed the datasheet for the chip and added it to my paper-based file-folder system.

You can see that this process entailed a lot of clicking. But it took me less than ten minutes, and I found exactly what I wanted.

I could have followed a different path. I knew I wanted a chip in the 74xx family, so I could have gone to this URL, which I keep bookmarked for easy reference:

```
www.wikipedia.org/wiki/List_
of_7400_series_integrated_circuits
```

This includes all the 74xx logic chips that have ever been made. If you go to this page, you can press Control-F to search the text, and then type in:

```
4-bit binary counter
```

It has to be an exact match, which means you must type 4-bit, not 4 bit. The search yields 13 hits, and you can compare the features of the chips. After you choose one, you can copy its serial number and paste it into the search field of a site like Mouser, which takes you straight to that one component, with much less clicking.

I could have used yet another approach, doing a Google search for people discussing and advising each other about counter chips. But you get the general idea. You don't need a part number, to find what you want.

Search Three: Aggravation

Now suppose you want to buy a slide switch for use in Experiment 6. You have read my instructions, and they tell you not to worry about the voltage or current capacity, because you're only going to use it for demo purposes with tiny current and small voltage. But it has to be an SPDT switch (you may not know what that means yet, but still, you have to include it in your search), and it has to be ON-ON type, and the terminals should be 0.3" or 7mm apart so that you can grab them easily with your alligator test leads. That should be simple enough, shouldn't it?

The problem is that for a component such as a switch, there will be tens of thousands of varieties available from the big three suppliers. This will entail a lot of search filtering, so I will start with a Google search:

```
buy "slide switch" spdt
```

The first hit takes me to a page on Pololu, where there's a little picture, much like the picture that I helpfully provided at the beginning of Section Two. Just the thing!

Or maybe not. Don't forget:

- When you don't have a part number, always check the datasheet.

Fortunately Pololu includes a little diagram on the same page, showing the distance between pins on the switch, which seems to be 2.5 although Pololu doesn't say what this means. Well, it will either be millimeters or inches, and it certainly isn't 2.5 inches, so it must be 2.5 mm, which is 0.1".

Component dimensions are sometimes provided in millimeters, sometimes in inches, or sometimes both. Where both are provided, one unit may be in parentheses, but you won't know whether it is the inches or the millimeters. Check my conversion charts on pages 134 and 135 or refer to a ruler which has both units printed on it. Bear in mind, 0.1 inch = 2.54mm.

This switch was smaller than I wanted. I would not be able to grab the center pin easily with an alligator clip, without touching the other pins.

Unsuccessful searches are very aggravating. Still, since I was already on Pololu's site, I used their search window to see if they had other SPDT slide switches—and no, they didn't have any others.

My next hit on Google was for the Sparkfun site, but their switches had 0.1" (2.5mm) pin spacing, just like on Pololu. You see, a lot of people want tiny switches that plug into a breadboard, where holes are 0.1" apart.

Well, maybe I could specify my pin spacing in a search term. Like this:

```
buy "slide switch" SPDT 0.3in
```

Ah, no. That dumped me into Search Engine Hell, where the number "3" caused me to be offered switches that had an actuator (the little button) that is 0.3" high, or switches sold in quantities of 3, or 0.3" wide . . . and nothing was what I wanted.

I realized I would have to go back to a big component supplier where the switch I wanted was sure to be in stock—if I could find it.

At Mouser I typed "slide switch" into the main search box at the top of the home page, and got 3,868 hits. Oh dear. What now?

I went through the filters. Didn't care about the manufacturer. Illuminated? I clicked "non-illuminated." Contact Form? I clicked "SPDT." Then "ON-ON." Skipped the rest and clicked Apply Filters. Now I had 363 to choose from.

So now all I had to do was find the filter which would let me choose pin spacing. And—there was no such filter! Pin spacing is fundamental, but they didn't let me specify it. And when I went to Digikey and Newark, they didn't let me choose it either.

Now I was stuck. Datasheets always show pin spacing, but I didn't want to check 363 datasheets.

It was time for creative thinking. This required standing up, going to the kitchen, and eating a snack. If you just continue a search single-mindedly, you'll get tunnel vision and you will not think creatively.

I returned to my computer with an idea. The termination style options on Mouser included Quick Connect. I happened to know, Quick Connect tabs are large, so they have to be well spaced. Aha! Click that option, click Apply Filters. Only 1 switch now qualified. But—it was non-stocked!

I didn't give up. I clicked the link to the datasheet for that component, because manufacturers always offer many variants of their switches, and the size of a switch in the series tends to be fixed within those variants. If one variant was big enough for Quick Connect terminals, they all should be. Sure enough I got to a datasheet for the C&K company's `1000` series of switches. All of them had pin spacing of 0.185" or 4.7mm. Less than I hoped, but I figured that if I bent the terminals outward, it will work.

From the datasheet, I selected a SPDT switch and copied the part number. Then I went back to Mouser and pasted in the part number. It would cost $3.70, which was more than I wanted to pay, but I decided to buy it anyway.

At this point you may be thinking that you never want to do a search like that. Well, neither do I! But this was a worst-case scenario. Most of the time, searches are not so frustrating.

If you have a part number, searches should be especially easy, and in this book, I always supply part numbers, unless a component is utterly generic, like an alkaline battery. But what if a part becomes obsolete? Then the part number won't work anymore!

Fear not. First, a lot of obsolete parts are still available through eBay. Second, I usually supply part numbers for two components, in case one disappears.

When all else fails:

If a part number cannot be found, email me, and I will resolve the problem. I will distribute the information to all readers who have registered with me, and a substitute part number will be provided in the next printing of the book.

You Can Always Try Voice

There is one more option when you are using a big component vendor. The phone! Each of the big three has sales representatives who can assist you. It doesn't matter that you are an individual buying small quantities; the representative doesn't know that, and doesn't care, and will get paid the same amount for helping you, no matter who you are. If you have a technical question (such as, "I want a slide switch with at least 0.2" pin spacing") they can bump the call up to someone who may know the answer without looking it up.

Another option is to open a chat window. This will allow you to copy-paste a part number into the window and get a fairly quick answer advising you of similar options if the part is unavailable.

General-Purpose Search Engines

A general-purpose search can guide you to special-purpose components if you do it right. Suppose, for instance, you are looking for a toggle switch.

A search term that is brief and vague will not be successful. If you want a DPDT toggle switch rated for 1 amp, just say so:

```
"toggle switch" dpdt 1a
```

Note the use of quote marks to nail down a specific phrase, discouraging the search engine from showing near-miss results that are not quite what you asked for. Also note that search terms are not case-sensitive; there's no advantage in putting a term such as dpdt in caps.

You can narrow your search even further by naming a source, such as:

```
"toggle switch" dpdt 1a amazon
```

Why mention Amazon, if you can go to amazon.com and do your searching there? Because I don't think the search capability at amazon.com is quite as good as a search engine such as Google or Bing.

Exclusions

When using a general search engine, use the minus option to avoid items that you don't want. For instance, if you are only interested in a full-size toggle switch, you could try this:

```
"toggle switch" dpdt 1a amazon
-miniature
```

Alternatives

Don't forget the OR logical operator. If a single-pole double-throw switch will work just as well for you as a double-pole double-throw switch, you could try this on a search engine:

```
"toggle switch" dpdt OR spdt 1a
-miniature
```

View Images

If you don't want the chore of typing an elaborate search term, you have other options. One is to click the word "Images" which a search engine displays immediately above each set of search results. This will show you pictures of every conceivable kind of switch, and because our brains are well equipped to recognize images quickly, scrolling through a lot of pictures can be a more efficient way to find what you want than scrolling through a lot of text.

Smart Datasheet Viewing

You can view the datasheet of a component by using a general search, but you need to do it right. You should *not* use this as your search term:

```
datasheet 74HC08
```

This will take you to an annoying third-party site which stores datasheets and shows them to you one page at a time, interspersed with ads. You will do better if you type a general search term including the manufacturer of the part. For instance, the `74HC08` chip is made by Texas Instruments (among other manufacturers), so you can type, as a search term:

```
texas datasheet 74HC08
```

The first hit takes you to Texas Instruments, where they maintain their datasheets in a very civilized manner, with no ads. Since their company manufacturers thousands of chips, you can use this search technique for almost any part number.

More About eBay

If you're in the habit of buying from eBay, you can skip this information. But if you don't use eBay often, these tips can make your experience easier.

First, don't hesitate to click the little "Advanced" option just to the right of the Search button on eBay's home page. This will allow you to specify attributes such as the country of origin (if you want overseas suppliers, or if you want to avoid them), and you can limit your search to Buy it Now items. You can also specify a minimum price, which can be useful to eliminate stuff that is too cheap to be any good. Then, before starting the actual search, I usually click the display option for Price + Shipping: Lowest First.

Once you find what you want, it's time to check the seller's feedback. For sellers within the United States, I want 99.8% or better. I've never had a problem with sellers rated 99.9%, but I have been disappointed sometimes with service from sellers rated 99.7%.

If a supplier is in an Asian nation such as China, Thailand, or others, you can be less fussy about feedback, because a lot of buyers give bad feedback when they don't receive something as quickly as they expect. Overseas sellers will warn you that a small packet will take 10 to 14 days in transit, but buyers complain anyway, and this unfairly drags down the feedback rating. In reality, in my experience, every item that I have ordered from overseas sources has always turned up, and has always been what I wanted. You just need to exercise a little patience.

After you find what you want on eBay, you may want to click the Add to Cart button, rather than the Buy it Now button, because you can look for additional items from the same seller, and you'll save time by grouping them into one shipment. This should also reduce shipping costs.

Click the Visit Store option in the Seller Information window, or if the seller doesn't have an eBay store, click to See Other Items. You then have the option to search within that seller's list of products. After you add as many as you want to your cart, it's checkout time.

Revisiting the Kits Option

My descriptions of searches may make you feel that they aren't worth the trouble. Why not just buy a kit, make a single payment, and have everything you need delivered to you a few days later?

Yes, this is an attractive option. But what if you decide you want to modify one of my projects? Or what if you want to build a circuit that isn't described in this book? As soon as you take that step, you're going to have to go shopping. With this in mind, I suggest buying as wide a range of components as possible, all at one time.

The idea, of course, is to enable you to have as much fun as possible when pursuing an interest in electronics.

Index

C

D

E

F

G

N

O

P

T

U

V

W

Z