Transducers

A few different kinds of transducers will read an analog input. Some are very simple to work with, and others require a significant amount of extra circuitry to use. For this chapter, we'll concentrate on the simple ones: variable resistors.

Variable Resistors

The most common class of transducers for analog input are variable resistors. *Variable resistors* convert a change in mechanical, light, heat, and other forms of energy into a change in resistance. The most common variable resistor is a *potentiometer*, or pot for short. Pots are used in many everyday devices. Volume knobs are pots, varying the resistance and therefore the signal strength that reaches the speakers. Pots are great in situations where you need a simple variable control that's hand-operated, but that's not always what you need in physical computing projects. Fortunately, there are many other types of variable resistors.

Thermistors are variable resistors whose resistance changes with the ambient temperature. A photocell's resistance varies with the intensity of the light hitting it. *Force-sensing resistors*, or *FSRs*, have a variable resistance that depends on the amount of force exerted on the sensor. *Flex sensors* offer a varying resistance depending on how sharply they are bent. It may be that one of these is perfect for your needs. For example, a flex sensor could determine how much a person is bending her finger, or a force-sensitive resistor could measure how hard she is squeezing a ball. You may have to creatively contrive the situation so that the information you're after alters the energy on a transducer you want to use. For example, you might use a thermistor to sense how hard a person is blowing (because a person's breath will change the ambient temperature) or use a photocell to determine the distance of an object from the sensor (objects closer to the photocell will block more light, if the object is between the light source and the photocell).

Because microcontrollers are binary in nature, they can only read a high voltage or a low voltage. In order to read a changing voltage, certain accommodations are necessary. Many microcontrollers come with built in analog-to-digital converters. An *analog-to-digital converter* measures a range of voltages and converts the value of the voltage at any given moment to a digital value to be stored in the microcontroller's memory. Some microcontrollers, such as the BS-2 and certain PIC models, do not have built-in ADCs.

Inductive versus Resistive Loads

There are two general classes of devices you might control from a microcontroller: inductive loads and resistive loads. *Inductive loads* are devices that work by inducing a current in a wire using the current in another wire or by passing the wire through a magnetic field. Motors and solenoids are examples of inductive loads. Inductive loads produce blowback voltage. They should have a diode placed in parallel with them, or with the transistor controlling them, as in the motor circuit in Figure 6.6, to lessen the effects of the back voltage. You'll deal with them in depth in Chapter 10. *Resistive loads* are devices that work by resisting electrical current and converting it to some other form of energy. Light bulbs and heaters are resistive loads. Resistive loads don't create a blowback voltage, so no diode protection is necessary. A good rule of thumb is that if it creates motion of any sort, it's probably an inductive load.

For those microcontrollers, you use a resistor-capacitor circuit to “fake” an analog-to-digital conversion.

Variable Voltage Analog Input Transducers

Besides the simple analog transducers we’ve already mentioned, there are several more complex analog transducers that can be very useful. Many of these operate on 5-volt DC power already, making them convenient to interface with a microcontroller. Usually, complex analog transducers operate on a given power supply (for example, 5 volts) and produce a variable voltage as output. For example, the Sharp infrared proximity ranger (Sharp GP2D12) operates on 5-volt DC, and outputs a variable voltage between 0 and 5 volts depending on the proximity of an object in its field of view. The GP2D12 can sense objects in a range from about 10 to 80 cm. Other models in the same family can detect objects in different ranges. We’ll discuss these more in depth in Chapter 9, “Sensing Things.”

There are some devices that produce a varying voltage that’s not in the 0 to 5 volt range of a microcontroller’s ADCs. For example, microphones produce a varying voltage, but one that varies only by a few microvolts. In order to use microphones and other devices that produce microvolt changes, it’s necessary to use a circuit to amplify the voltage to the range that the microcontroller can read. One simple way to do this with a microphone is to pass the microphone’s signal to an audio amplifier, which raises the voltages to levels that are readable by a microcontroller. We’ll detail this in Chapter 13, “Controlling Sound and Light.”

Circuit

There are two basic circuits for reading an analog voltage on a microcontroller: the voltage divider and the resistor-capacitor circuit. Voltage divider circuits work only on microcontrollers that have analog-to-digital converters. R-C circuits work on all the microcontrollers we’re writing about.

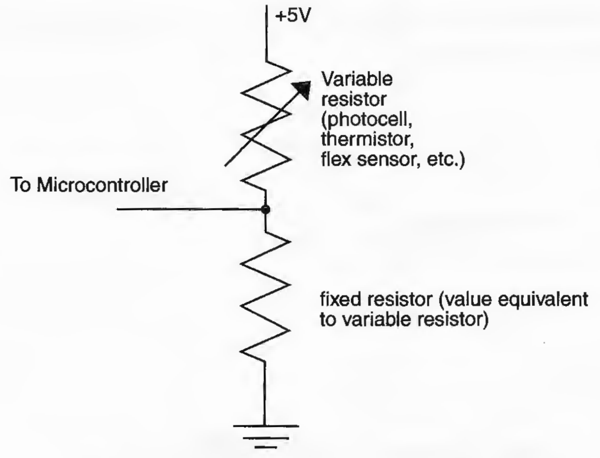
Voltage Divider

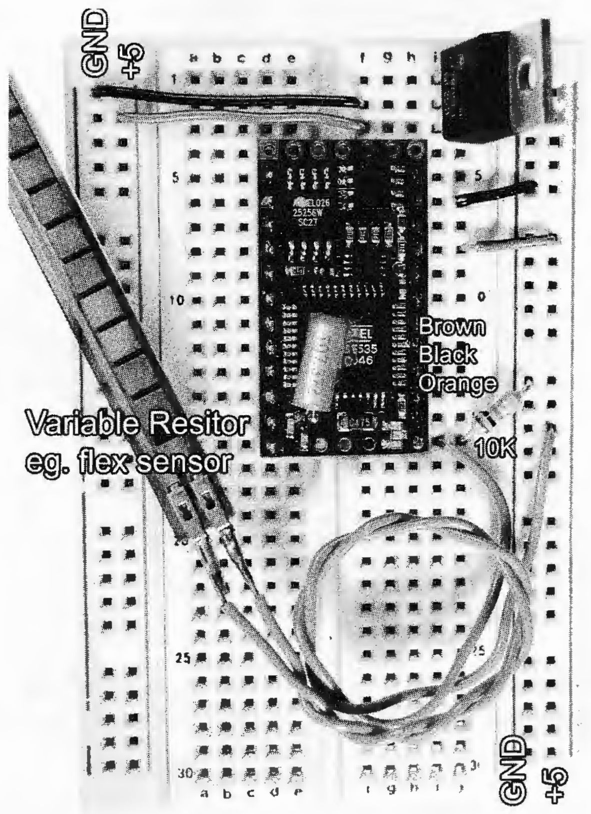
Even microcontrollers that support analog to digital conversion usually only do so on select pins. Before you get started with this circuit, you should identify the ADC-capable pins on your microcontroller. On the Basic Atom Pro24, they’re pins 0 through 4. On the BX-24, they’re pins 13 through 20. On the PIC 18F452, they’re pins RA0 through RA3, RA5, and RE0 through RE2. The BS-2 does not have any ADC pins.

You use variable resistors by passing a current through them and reading the voltage that comes out of them to determine how much they are resisting that current. In Figure 6.8, we connected a variable resistor (in this case, a flex sensor) from +5 volts to the microcontroller pin and a fixed resistor from the pin to ground. Use a fixed resistor that’s in the general range of the variable resistor’s range. For example, if your variable resistor varies from 10K to 100K, a 10K fixed resistor will suffice.

This is called a *voltage divider circuit* because the variable resistor and the fixed resistor divide the voltage into two parts. The variable resistor feeds a varying voltage to the microcontroller pin. The fixed resistor provides a path to ground. The voltage at the point where the resistors meet (namely, the pin) will vary with the ratio of the resistors’ values. For example, when the resistors are equal, the voltage on the pin will be exactly half of the supplied voltage, or 2.5 volts.

Figure 6.8
Analog input: a flex sensor.


+5V
Variable resistor (photocell, thermistor, flex sensor, etc.)
To Microcontroller
fixed resistor (value equivalent to variable resistor)



In the circuit seen in Figure 6.9, we used a potentiometer instead of the two resistors from the previous circuit. If you break open a potentiometer, you'll find a center slider touching into a single resistive strip. This makes the resistance of the two sides of the potentiometer perfectly complementary, and so it's perfect to replace the two resistors in our voltage divider circuit. If you connect one side of the pot to ground, the other to 5 volts, and the center to the microcontroller ADC pin, you will get a range of numbers from 0 to the maximum that the ADC can return.

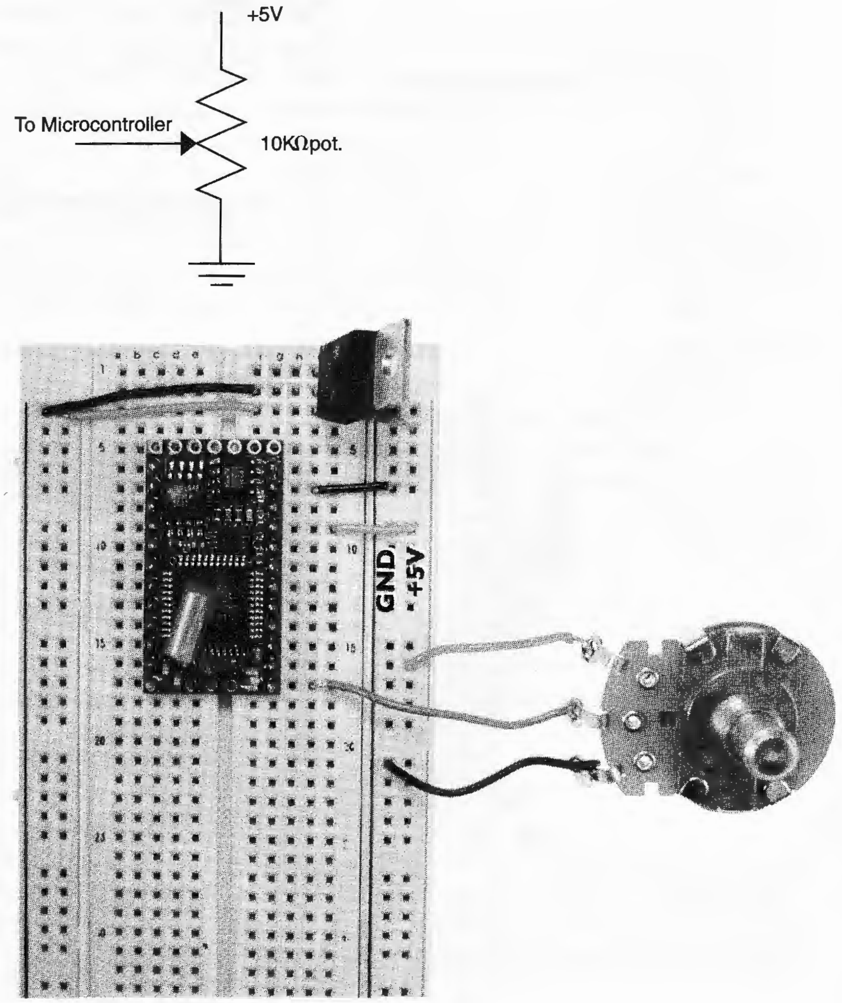


Figure 6.9
Analog input:
a potentiometer.

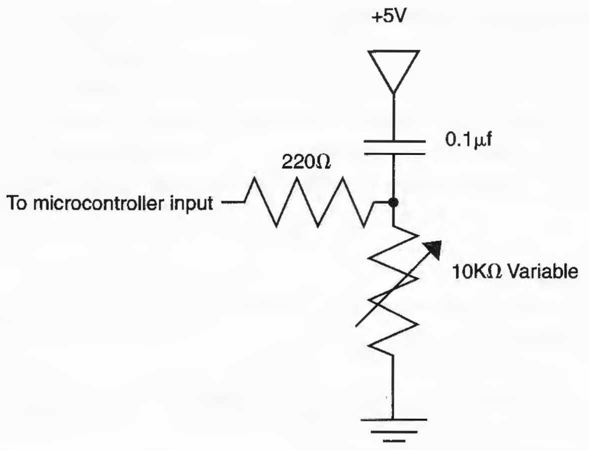
If you build these circuits as-is, place your multimeter leads at the point that would go to the microcontroller and at ground, and measure for voltage, you will see a changing voltage value.

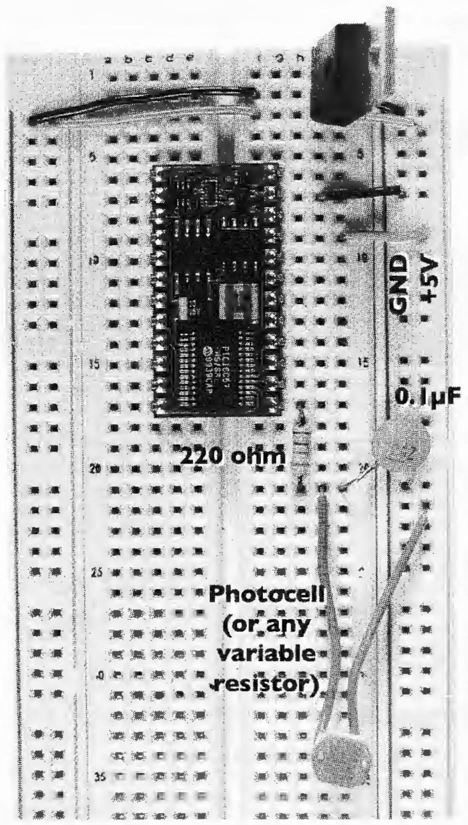
RC Circuit

If your microcontroller doesn't have any ADC pins on it, or not enough of them, there is another method you can use to measure a varying resistance using a capacitor. The method you'll be using is called `rctime`. The circuit for it looks like the one shown in Figure 6.10.

Here's what happens when you use the `rctime` command. Capacitors store charge when they're fed electricity, and release it when the feed is turned off. First, you take the input pin high. This lets all the charge out of the capacitor (since both sides of it are high). Then the microcontroller sets the voltage on the input pin low, causing the capacitor to start

Figure 6.10
Analog input:
RC circuit.





building up a charge. The microcontroller starts counting time, waiting for the capacitor to recharge. The charging happens in a matter of microseconds, but a microcontroller is fast enough to measure that time. The more resistance from the variable resistor, the longer it will take to charge. Once all the capacitor is charged, the microcontroller returns a value. The higher our variable resistor's resistance, the higher the value the `rctime` command returns. `rctime` is somewhat slower than using an ADC but it's handy when you don't have an ADC to use. For most variable resistor sensors, it will do the job just fine. You may also be able to use the `rctime` command with a sensor that produces an analog voltage in the 0 to 5 volt range, like the Sharp IR rangers mentioned above. Remove the resistor from the RC Circuit and attach the sensor's voltage output to the microcontroller pin. Leave the capacitor in your circuit, between the microcontroller pin and ground.

Programming

Here's the pseudocode describing what we're going to do in our program to make the microcontroller read an analog input:

```
Make a variable big enough to store the analog value that you'll be taking in.
Loop
    Read the analog value coming in
    Output the number
End loop
```

[redacted]

The basic four steps of edge detection come up constantly when designing any kind of interactive system. Edge detection routines enable the microcontroller to interpret changes in sensor readings as higher-level actions. You'll use these steps all the time, whether you're reading digital or analog sensors.

Analog Sensors: Thresholds, Edges, and Peaks

When you're working with analog sensors, the beginnings and endings of the events you might sense are slightly different than for digital sensors. We will talk about three kinds of changes that you might look for: thresholds, edges, and peaks.

Finding Thresholds in an Analog Signal

Sometimes you only care whether your sensor has passed a threshold. For example, if you're working with a photocell, and you want to sense a flashlight falling on the photocell but not the ambient light in the room, you might write a routine to filter out all readings below a certain threshold. Testing whether or not your reading is above or below a threshold essentially turns your analog sensor into a digital sensor.

You could detect a threshold with a routine like the following:

```
establish threshold from a reading of ambient conditions at start up
Loop:
Read sensor
If sensor reading is higher than threshold then
        React
End if
End loop
```

This code is simple but it all depends on having a good number for the threshold. If you are in a controlled setting, you may be able to use a fixed number for your threshold. But if your sensor were a photocell, for example, this threshold might work in the morning but not at night. You might need to calibrate the threshold to different ambient conditions. One quick way of doing this it to grab a sensor reading before you enter your main loop and use that for the threshold. You would just have to make sure that your microcontroller is powered up under normal ambient conditions to get a good threshold and restart the chip to recalibrate the threshold. Another approach is to continually read a *control* sensor that is situated in an area away from the area where you expect change. The reading from the control sensor gives you the current baseline of ambient conditions and is used to set the threshold for the sensor that is actually changing. For thresholds that automatically recalibrate over time, you will need to average your readings over time. See the section called "Smoothing, Sampling, and Averaging" for more on this.

Finding Edges in an Analog Signal

If you think about the threshold as an edge, the process of sensing when the sensor is activated is similar to the digital sensor example above. You take a reading, determine if it's above your threshold, and if the previous reading was below the threshold, the sensor has just been triggered. If the reading is below the threshold and the previous reading was above, then the sensor has just ceased to be triggered.

This is a very simple routine. It's much like the digital input reading routine before you added edge detection. A slightly more refined routine might incorporate an analog version of edge detection, like so:

```
Loop:
    Read sensor
    If sensor reading is higher than threshold then
        If previous sensor reading was below threshold then
            React
        End if
    End if
    Save the current sensor reading as the last reading
End loop
```

Since the logic of analog edge detection is the same as it is for digital edge detection, we'll leave it to you to write your own actual code for this.

You might find that find your readings are fluctuating slightly above and then below your threshold, giving you many apparent edges when you are really being still. For example, in the photocell graph in Figure 8.4, you can see lots of ripples in the edge of the curve. These could appear as multiple threshold crossings when in reality there's only one crossing that you care about. You can use some of the techniques in the section called "Smoothing, Sampling and Averaging" to reduce this problem.

Finding Peaks in an Analog Signal

Analog threshold detection is great for testing when an analog sensor crosses a threshold when it's rising or falling. Sometimes, however, you want to know when it's reached a peak and is headed back down again. For example, imagine the sensor in the graph on the right in Figure 8.4 is the sensor for a key on a piano keyboard. To know how loud the note is to be played, you would want to know the peak, which tells you how hard the key was hit. In this case, a slight variation on the edge detection routine would do the trick. Instead of an edge, you have a peak, where the sensor reaches its maximum value. In order to find the peak, you look for the sensor's reading to cross the threshold going up, then wait until it reaches a maximum and take that maximum as the peak.

In order to determine a peak, you have to set a threshold for when you start and stop looking for the peak. Figure 8.4 shows readings from two different sensors. The figure on the top shows what happens when you cover a photocell with your hand quickly. The figure on the bottom shows several taps on a force-sensitive resistor (FSR). You can see that the taps on the FSR happen much quicker than the covering of the photocell. There's a clear peak on each tap on the FSR. The readings start at zero (which is the threshold where you start and end looking for the peak) and end at zero. In the photocell graph on the left there is a peak, but the beginning and ending point is not as clear, so you need to set a threshold to determine which readings you care about and which you don't. With the photocell, you might only care when the light crosses the threshold, because the change is so gradual. On the other hand, because the change in the FSR readings is sudden, you might need to look for the peak.

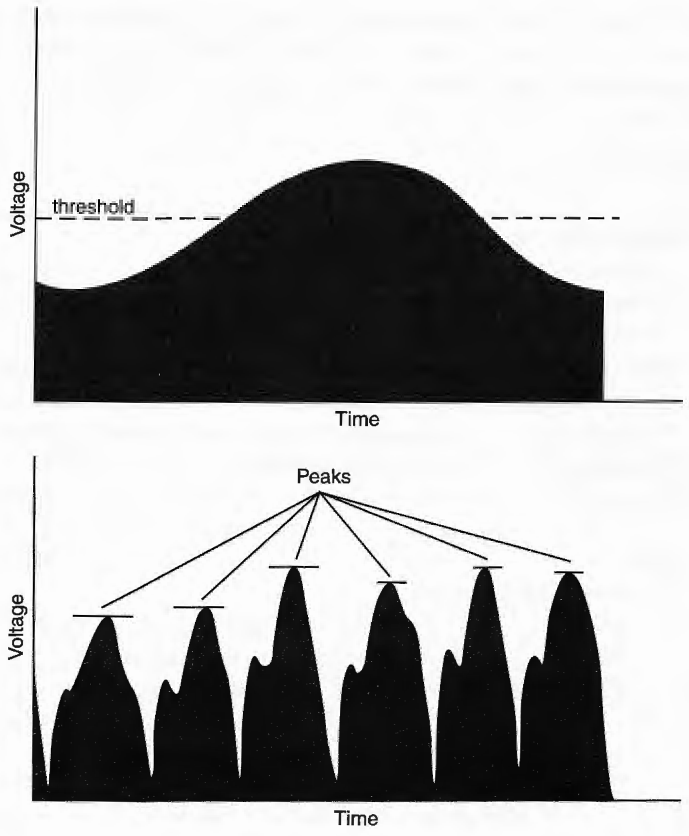


Figure 8.4
Analog sensor readings. On the top, a photocell being covered and uncovered quickly. On the bottom, several taps on a force-sensitive resistor.

In pseudocode, finding a peak looks like this:

```
Loop:
    Read sensor
    If sensorValue >= threshold then
        If sensorValue >= lastSensorValue then
            PeakValue = sensorValue
        End if
    Else
        If peakValue >= threshold then
            this is the final peak value; take action
        end if
        Set the peak value to 0 so we can start it rising next loop
    End if
End loop
```

You might have to set your threshold fairly high in order to eliminate false peaks like the ones you see at the beginning of each reading in the graph on the bottom in Figure 8.4.

9

Sensing Movement

Some of the most popular interactions in physical computing tap into the sometimes unnoticed but unavoidable tendency people have to express themselves through bodily movement. People move in order to position themselves best for a given activity, to get a better look at something they're interested in, to respond to music or some other stimulus, or just because they're uncomfortable and need to shift their weight. Body movement is exploited in some of the most common everyday applications, from the auto-flush mechanisms on public toilets to the light switches in our offices that turn on automatically when they detect movement. Position and motion sensors make it relatively simple to take advantage of body position and movement for a wide range of projects, from making ghosts in haunted houses leap out when a person passes a door threshold to changing the lighting and projections in a dance club based on the combined movement of bodies on a dance floor.

What all these interactions have in common is the ability of the computer to sense the position of objects in space. Given that the toilets at the airport know when you are standing in front of them,[1] you would think that this is an easy task. In fact, this can be a very difficult problem, depending on how much information you need and how much you can control the environment. In this chapter, we'll discuss a number of techniques for detecting position, motion, and orientation. We'll also discuss ways to constrain the environment to make the most effective use of the sensors described.

Assessing the Problem

In order to sense bodies in motion, you have to begin by determining how much you need to know about their motions. There are a few basic factors of motion to consider:

- *Position.* Do you need to know where something is within a space?
- *Orientation.* Do you need to know if they're rotating about an axis? Which way they're facing?
- *Velocity.* Do you need to know how fast they're moving, and in which direction they're moving?

[1] Bill Buxton, former chief scientist at Alias Research, is famous for exhorting computers manufacturers to make computers at least as smart as the toilets at O'Hare airport.

- *Absolute or Relative.* Do you need to know an object's absolute position in a space, or just the change from its previous position?
- *Identity.* Do you need to discern between multiple objects?

When talking about position and orientation, you have to consider the *degrees of freedom* the body has in which to move. The fewer degrees of freedom you are interested in, the easier the job. There are three degrees of freedom for position: left to right (X axis), up and down (Y axis), and front to back (Z axis). On top of these, there are three more degrees of freedom for orientation or rotation: rotation around the X axis (*tilt* or *pitch*), rotation around the Y axis (*pan* or *yaw*), or rotation around the Z axis (*roll*).

Range of sensitivity is one of the most important criteria when choosing a sensor. Many of the sensors discussed below are ranging sensors, meaning that they sense distance of a body from the sensor or movement in front of the sensor. The range of most sensors is constrained on the maximum end and on the minimum end. Most sensors will have an effective range of under 9 feet. Video tracking is an exception to this because you can adjust the range by adjusting the zoom on the lens. In addition, distance-ranging sensors will not operate uniformly at all distances, so you have to work with the zone in which they're most sensitive. Figure 9.1 illustrates the typical cone-shaped zone of sensitivity that most ranging sensors have.

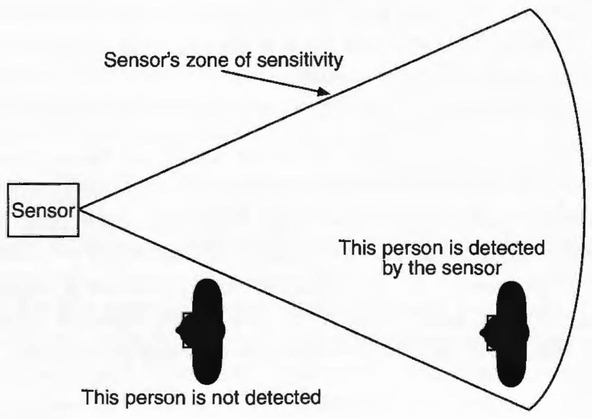

Sensor's zone of sensitivity
Sensor
This person is detected by the sensor
This person is not detected

Figure 9.1
Many of the sensors in this chapter have a cone-shaped field of sensitivity.

The best course of action in dealing with the various challenges of position tracking is not to apply more sophisticated technology. Better results usually come from constraining the environment to make your job easier. For example, ask yourself how many people should be able to experience your project at one time. Consider constraining the space or the entrances and exits so only one person at a time can fit through rather than dealing with identifying multiple targets. There will be times when this is inappropriate or limits the project too much, but if it doesn't make any difference, or if it improves your project to work with a constrained space, do so. Like a good magician, you should make your necessary constraints look like a perfectly natural part of your system.

How Ranging Sensors Work

Most sensors that read the distance from a target send out some form of energy (light, magnetism, or sound) as a reference signal. They measure the amount of energy that reflected off the target and compare it with the energy that went out. Then they convert the difference into an electrical voltage or digital signal that you can read on a microcontroller. For example, a very cheap low-tech solution for making your computer as smart as the toilets at the airport would be to place a photocell and a flashlight on your computer monitor facing the user (see Figure 9.2). As a person gets closer to the computer, more of the light from the flashlight will bounce off of her body back to the photocell. The photocell would then give you a reading of whether a person was sitting in front of the computer and roughly how close she was sitting.

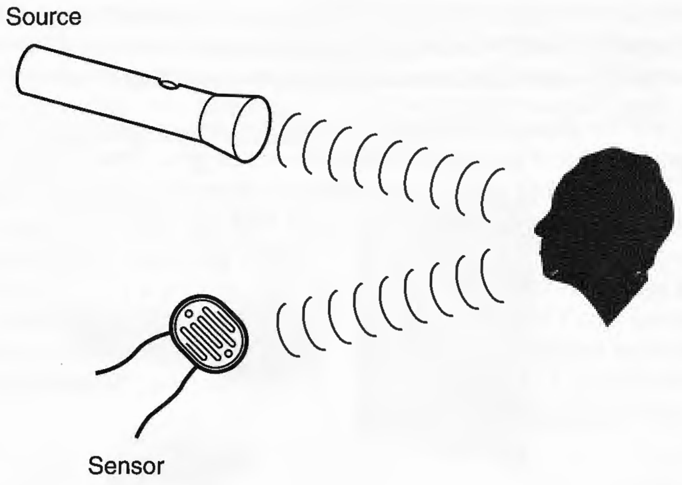


Figure 9.2
Many distance sensors work by sending a beam of energy (light, sound, and so on) out and bouncing it off the target.

This principle is common to many different sensors and across many scales. On a small scale, there are the finger sensors in virtual-reality gloves. In some of these gloves, light is sent down an optical fiber that rides along each finger.[2] The fiber is scratched at each knuckle so that a little light escapes when the fiber is bent. Bending a finger causes less light to reach the end of the fiber. A photocell at the end captures the light and reads the change as an analog value. On a large scale, airplane radar systems operate by sending out a radio signal and measuring the time it takes for the signal to bounce back from a target.

This simple theory might inspire you to start making lots of homegrown sensors like the flashlight example above. Keep in mind that ambient conditions in the field might wreak havoc on a project that you've only tested in the lab. In the virtual-reality glove example above, both the light source and the sensor could be completely encased and never affected by ambient conditions. In real life situations, people with different wardrobes, hairstyles, and skin pigmentation will reflect light differently. Because there is no lens over the photocell, light coming from other sources and from every direction will affect the

[2] Unless you have a manufactured version of this, we recommend using flex sensors instead for most glove projects. Flex sensor finger sensors are much easier to build.

photocell. Changing ambient lighting conditions can interfere with your expectations of how much light should bounce back. Finally, your user may not enjoy having a flashlight shining in her face. We do not want to discourage your homegrown sensors, especially for very contained environments. However, it's generally worth the money to buy a commercial sensor module when it is available. These modules typically use lenses or filters to narrow the field of sensitivity. They use infrared light, ultrasonic sound, or other energy sources that are not perceptible by humans. In addition, they often send out energy that is modulated at a distinctive frequency to make it easier to distinguish it from other sources in the environment.

Detecting Presence

You can skip ahead to "Determining Position"

Sometimes all you want to know is whether someone is present or not. This only requires a simple digital input sensor. For example, if your haunted house multimedia extravaganza is started when a person enters a doorway, all you need is a digital sensor to tell when he enters.

One common use for distance sensors is to track a person moving in front of an object in order to trigger the object to action when the person gets close enough. This can be very effective, but keep in mind that being present and paying attention are not the same thing, as any parent or teacher can confirm. Imagine that you want to sense when a person is looking at your painting so that you can make the painting respond in some way. You could put a ranging sensor in front of the painting and look for a person to get close enough, but this sensor alone won't tell you whether she's got her back to the painting or not. Sensing attention is a more complex problem. The sensors we'll cover in this section are good for sensing presence only, but the sensors in the next section, "Determining Position," can be useful for detecting presence as well.

Foot Switches

Foot switches are the most straightforward means of detecting a person's presence, especially within a small area. The most common type is made of long strips of metal tape separated at intervals by foam tape. When someone steps on these sensors, the foam tape compresses and the metal strips touch each other like the contacts of any other switch. The biggest problem with these sensors is that they need to be robust enough to withstand the weight of people constantly stepping on them. For this reason, it's probably better to use a commercial version, even though they seem very simple to make. There are some good ones available at http://www.tapeswitch.com. For a higher-end analog version, see the TapTiles made for Infusion Systems' I-Cube (http:// www.infusionsystems.com). Floor pads tend to be ugly and should be concealed beneath some sort of carpet.

Photoelectric Switches

In a photoelectric switch, a light beam hits a target sensor. When the beam is broken by a body passing between the sensor and the light source, the switch is activated. These are the "electric eye" switches that have been used for decades in such things as automatic door openers. They're great for detecting passage through some threshold, such as a body through a doorway or a hand through a hole in a box. You always see these in movies about jewel theft, sending red beams of light across the room around the treasure. In reality, you seldom see the beam, and the jewel thief is never as attractive as the ones in the movies. Some photoelectric switches have an emitter on one side and a detector on the other. Others have the emitter and detector on one side and just a mirror on the other side. You can make your own with a cheap laser pointer on one side and a photocell or phototransistor on the other. Keep in mind that these sensors are digital, not analog. They won't tell you how far the body is from the light, just that the body crossed the beam of the light.

You can also buy photoelectric switches from any burglar alarm supply store (see Figure 9.3). These burglar alarm sensors can be effective across very wide areas and don't necessarily even need a microcontroller. They usually have two terminals, which get connected when the beam breaks. You can connect these as a switch into a digital input circuit or use it directly to turn on a relay or another low-amperage device. Quite often, they will delay their reset for some period of time after being tripped. There's usually a potentiometer in the device for lowering the delay. You may not be able to lower it to zero, though, depending on the sensor you buy. Highly Electric (http://www.highly.com) carries photoelectric switches if you're looking online for a source.

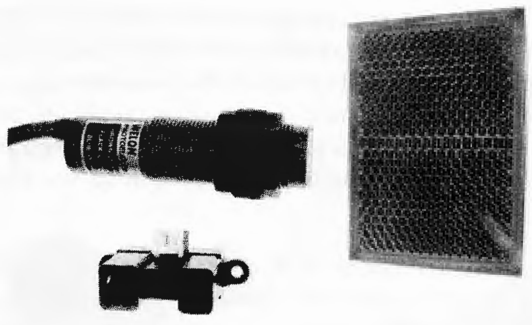


Figure 9.3
A photoelectric switch and its target, and an infrared sensor. These two sensors work on similar principles, but the photoelectric switch is digital (only detects if someone crosses the beam), and the ranging sensor is analog (measures the distance between the person and the sensor).

Motion Detectors

Motion detectors are those ubiquitous beige or black boxes with an LED that blinks when someone walks anywhere in the room (see Figure 9.4). These sensors respond to changes in the infrared light in the space. They only react to change, so will not tell you if someone is in the room and standing still, only if they are moving. The ones used for burglar alarms usually come with terminals that will connect as a switch into a digital input circuit. They offer a distinct advantage over photoelectric switches in that they are extremely easy to install, usually have a much wider field of sensitivity that can be adjusted by changing the lenses that usually come with them. Like photoelectric switches, these sensors will delay their reset for some period of time after being tripped. There's usually a potentiometer in the device for reducing the delay, but again, you may not be able to reduce it to zero, depending on your sensor. You can also hack into the motion sensors used for outdoor lights, which are cheaper and usually available in any hardware store. Though these

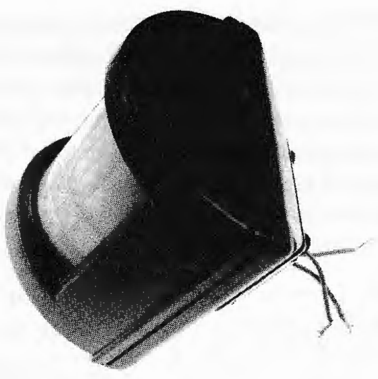


Figure 9.4
A motion detector.

sensors are usually designed to switch 120 volts AC, they usually operate well on as low as 5 to 12 volts DC, so they'll work with your microcontroller.

Magnetic Switches

Magnetic switches consist of a very thin pair of contacts in a protective housing. When exposed to a magnet they're drawn together, closing the switch. Sometimes you'll see the switches themselves (without the magnets) sold as *reed switches*. The magnetic switch is a favorite for sensing the position of moving objects because the magnet doesn't require any wires (see Figure 9.5). Suppose you want an object to be able to move unencumbered by wires, but you want to know when it is placed on a pedestal. You might simply put a magnet in the bottom of the object and the magnetic switch on the top of the pedestal.

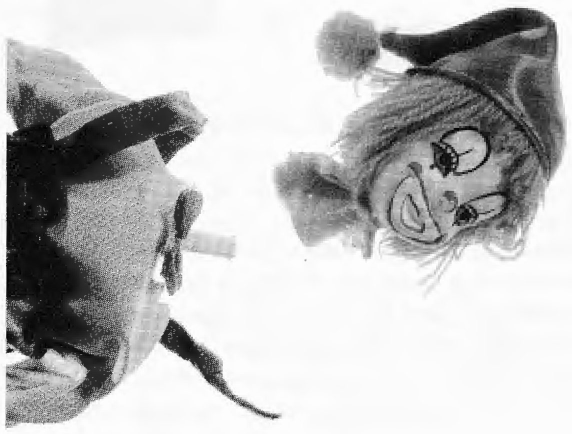


Figure 9.5
A creepy doll head with a magnetic switch inside the neck. The magnet is in the head.

Hall-effect sensors are similar to magnetic switches in that they detect a changing magnetic field. The simplest Hall-effect sensors are digital ones that change their output from low to high when the magnetic field around them changes. Analog Hall-effect sensors output a variable voltage as the strength of the magnetic field around them changes. Hall-effect sensors are very easy to use. Typically they have three leads: power, ground, and output. To use them, you give them voltage and ground, then connect the output to your microcontroller, reading it as digital or analog depending on the nature of the sensor. The range on a Hall-effect sensor is very short, a couple of inches at most, so they're really of most use for measuring very small changes between magnetic objects. Jameco and Digi-Key (http://www.jameco.com and http://www.digikey.com) both carry Hall-effect sensors.

Determining Position

The sensors we've covered so far allow you to tell whether a person is in a particular space or whether or not they've moved in the space, but they don't tell you where the person is or where she's moving. Following are a few sensors for determining that. Most of these will enable you to determine the person's or object's speed and velocity as well, if you keep track of the change in position.

It is common to equate digital sensors with detecting presence and analog proximity sensors with detecting distance. This is not always true. Given enough digital data points, you can create an analog feel from digital sensors. For example, if you put enough digital floor switches in a room, you would know roughly where in the room the person is standing by which floor switch they are standing on. Conversely, analog sensors can be used in a "binary" way by sensing if a particular threshold has been passed (see Chapter 8 for details on thresholds). You can change this threshold in software, giving you further flexibility for changing conditions.

IR Sensors

You really can't beat the Sharp GP2D family of IR sensors (carried by http://www.acroname.com, http://www.digikey.com, and other retailers) for determining short distance. They are cheap, relatively accurate, and easy to use. These sensors send out an infrared beam and read the reflection of the beam off a target (refer to Figure 9.2). Different models work at different ranges. The longest range sensor in the family, the GP2Y0A02YK, can sense from eight to 56 inches. The shortest range one, the GP2D120, can sense a range from about 1.5 to 12 inches. There are analog sensors in this family and digital sensors. The digital sensors trigger when a person or object moves within a given threshold distance of the sensor. The analog sensors output a variable voltage from 0 to 5 volts, which varies with the distance. We find the analog ones more useful because they can always be converted to digital use by measuring for a threshold. The standard analog input circuit (refer to Chapter 6) will work for these if your microcontroller has A/D conversion built in. Figure 9.6 shows the schematic and the wiring. If your microcontroller has no A/D converters, you can use a variation on the R-C circuit used with the `rctime` command, as shown in Figure 9.7.

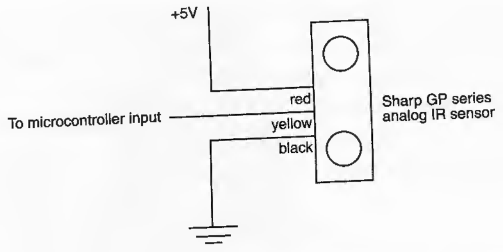
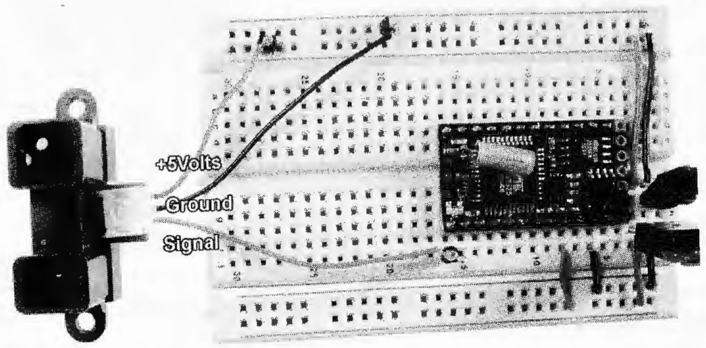


Figure 9.6
The Sharp GP2D12 infrared sensor connected to a microcontroller with built-in analog-to-digital conversion.



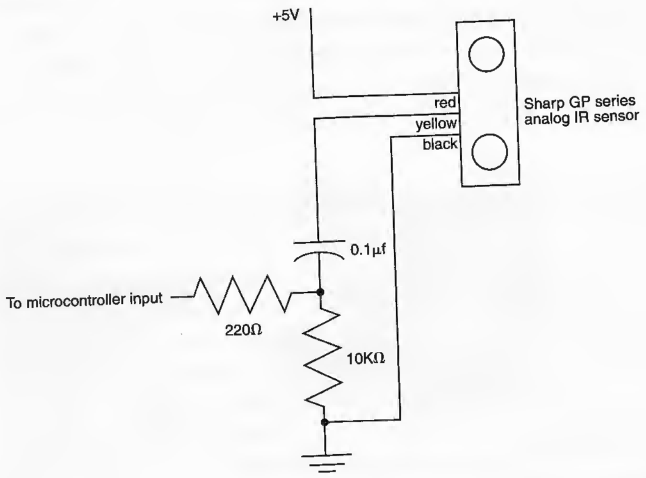
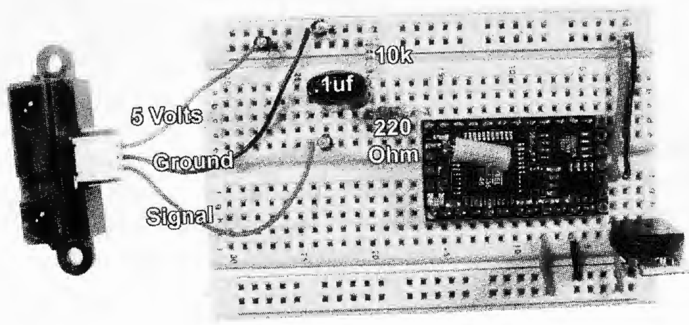


Figure 9.7
The RCTime circuit with a Sharp GP2D12 IR sensor.



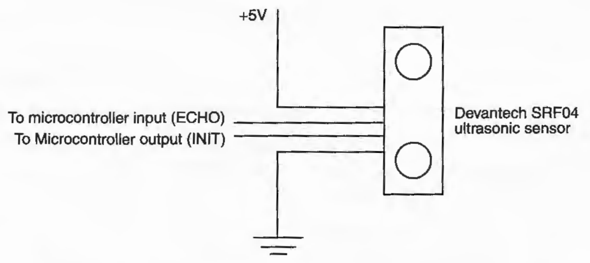
Ultrasonic Sensors

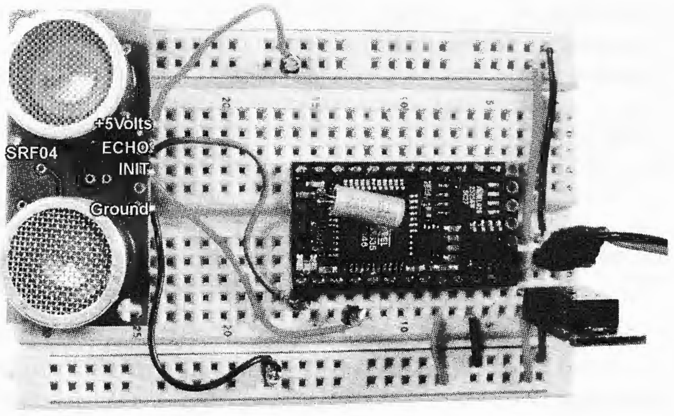
For longer ranges, ultrasonic ranging modules work well. Just like sonar devices, they send out a ping of ultrasonic sound, and then time how long it takes to bounce back. The longer it takes for the ping to return, the further away your target is. There are several different models. The SensComp/Polaroid sonar-ranging modules can read a range from six inches to 35 feet. They have an initiation pin and an echo pin. To use them, you set the initiation pin high, then use the `rctime` command on the echo pin to measure how long it takes to return a signal. These modules are useful but expensive, and they draw a significant current (up to two amps) when they send the ping.

The Devantech SRF family of ultrasonic sensors are a cheaper alternative to the Polaroids. The cheapest one, the SRF04, works in the same way as the SensComp/Polaroid model: you set an initialization pin high, then wait for a return on an echo pin. To determine the distance, you divide the time taken by the speed of sound in your microcontroller's software. More expensive models like the SRF08 allow you to just get the distance by communicating with the sensor using a protocol called I2C (for more on this protocol, see Chapter 12, "More Communication between Devices").

If you're using the Devantech SRF04 or the SenseComp/Polaroid module, you will need to do a little more programming. You can send out the ping with a simple `pulsout` command on the INIT pin of the module. You can time how long it takes the sound to bounce off the target and return using either a `pulsin` command or an `rctime` command. You will need a constant for the speed of sound to convert the microseconds returned by these commands into inches (73.746 microseconds per inch) or centimeters (29.033 microseconds per centimeter). Figure 9.8 shows the schematic and the wiring.

Figure 9.8
The Devantech SRF04 ultrasonic sensor.


+5V
To microcontroller input (ECHO)
To Microcontroller output (INIT)
Devantech SRF04 ultrasonic sensor



Here's the pseudocode for reading a Devantech SRF04 sensor:

```
Loop
    Pulse the init pin
    Time how long it takes to get a pulse back
    Divide the time by the speed of sound
    Pause to let the sensor reset
End loop
```

Because the minimum times on the `pulsout` and `pulsin` commands are not the same on the different microcontrollers we're using, you'll notice that the constant for the speed of sound changes from one microcontroller to the next. Here's the actual code:

[redacted]

[redacted]

Other Position Sensors

The infrared and ultrasonic modules are by far the easiest distance-ranging sensors and will probably take care of most of your distance measuring needs. For very close-range interactions, (less than the 1.5 inches that the Sharp GP2D120 is capable of measuring), consider analog Hall effect sensors. For very large-scale position sensing you can use GPS, which can pinpoint your position within a few meters almost anywhere on earth. GPS doesn't work indoors or in areas without a clear view of the sky, however. GPS is well covered in Chapter 12.

If you have very precise positioning requirements on the scale of the human body, consider magnetic motion trackers (http://www.polhemus.com or http://www.ascension-tech.com).

These give you extremely fast and accurate readings for all six degrees of freedom via RS-232 serial communication. The object that you're tracking has to be tethered to a small magnetic sensor. The range for the most accurate readings (less than 0.1") is about one yard radius, with a bigger range for less accurate readings. Large amounts of iron in the room will reduce the accuracy. These sensors are expensive. They cost thousands of dollars, but you can't beat their accuracy with any homegrown solution. They're great for applications where a person will stand still in one place and move parts of her body around or move objects around her body. Virtual reality designers love them. If you need precise positioning for these kinds of applications, then get one of these sensors; the investment will save you hundreds of hours trying to get precise six-degree tracking on your own.

In the $10,000 range, you can find a variety of sensors in the entertainment industry for motion capture for special effects and animation. Every year at the Association for Computing Machines' SIGGRAPH conference you will see new schemes for doing this. Often these systems require installation into a special purpose room. For example, Vicon (http://www.vicon.com) makes a system for motion capture that uses an array of cameras spread around the perimeter of an empty room. Retroreflective tape or balls (like the paint used in highway lines that reflects light very well) are attached to the body or object that you want to track. The cameras are designed to read only infrared light, which is reflected by the retroreflective tape. The data from all the cameras is combined to generate a three-dimensional track of the tape spots in space. To track a person, the spots are attached to all of the person's joints and whatever other body parts you want to track. Though motion capture systems are very accurate, they are also very expensive and require considerable computer expertise and horsepower to set up and maintain.

Determining Rotation

So far, we've introduced sensors that enable you to tell whether an object or person is present, and where they are in a space. In this section we'll talk about how to determine how the person or object is oriented with respect to other objects.

Potentiometers

The simplest rotation sensor is the ordinary potentiometer. For example, if you want your user to turn a wheel to adjust the rotation of an image or the pitch of a sound, you could attach a potentiometer to the pivot point of the wheel. The circuitry and programming for a potentiometer are very simple and are covered in the "Analog Input" section of Chapter 6. There are two challenges to using an ordinary potentiometer for sensing a full 360 degrees of rotation. First, most pots do not turn 360 degrees infinitely. Second, you have to find a way to attach them to the wheel that the user is supposed to rotate.

Most potentiometers don't turn a full 360 degrees. You can find multi-turn potentiometers, but eventually they have also a limit in both directions. You can pop the back off of a pot and eliminate the physical barrier to full rotation by cutting part of the metal casing off. However, a potentiometer's resistive band has a gap in it, positioned where the metal stops and keeps you from turning it further. You will get very erratic readings as the potentiometer's wiper moves over this gap. You might be able to live with this if the resolution that you need from the sensor is less than the error that the gap introduces. You can also buy dual potentiometers, or continuous rotation potentiometers, which contain two wipers, and rotate endlessly through 360 degrees. By reading the signal from both wipers, you get a continuously changing reading and can avoid the error caused by the gap. You do this by discarding the reading from whichever wiper is passing over the gap at any given moment.

Attaching a potentiometer to a larger wheel is usually a highly idiosyncratic problem, but there are a few guidelines that are helpful to remember. Generally, pots are too lightweight mechanically to take the force that an axle for a large wheel must take. Use a strong axle for the wheel, and then attach the pot to one of the axle's stationary supports. Next, couple the pot to the wheel itself to measure the rotation. Make sure there is a little give in the coupling so that the eccentricities of the wheel's movement don't produce wear on the pot. Figure 9.9 shows three different options for coupling a potentiometer to a wheel.

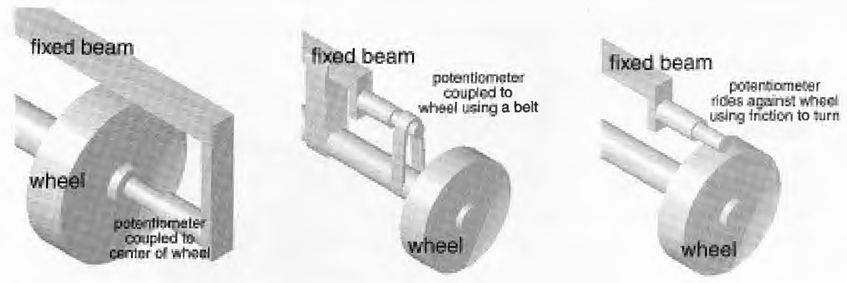

fixed beam
wheel
potentiometer coupled to center of wheel
fixed beam
potentiometer coupled to wheel using a belt
wheel
fixed beam
potentiometer rides against wheel using friction to turn
wheel

Figure 9.9
Three different methods for coupling a potentiometer to a wheel.

Accelerometers

Accelerometers measure the change in speed of movement, or acceleration. They typically have two (or sometimes three) axes of measurement. Accelerometers are commonly used in cars to measure acceleration and deceleration. If a car decelerates too fast, the accelerometer senses it and triggers the air bags to inflate. When the two axes are perfectly perpendicular to the earth, an accelerometer only measures changes from an outside force. For example, an accelerometer mounted on a toy car would measure the car's acceleration due to a push from you. When you tilt an accelerometer, however, it measures acceleration due to gravity. As a result, accelerometers will give you a relatively good measurement of the tilt of a body or object. Because they use gravity as a reference, they only need to be attached to the object you are sensing, thus avoiding all the mechanical hassles described above that come with attaching potentiometers to moving parts. Accelerometers will typically give you close to 90 degrees of sensitivity when it comes to measuring tilt in this way. Unless you're using a three-axis accelerometer, or two two-axis accelerometers mounted at 90 degrees relative to each other, you'll only get two directions of movement.

Analog Devices (http://www.analog.com) makes a number of good accelerometers, among them the ADXL202E. All of Analog's accelerometers are tiny chips, which are very difficult to solder by hand for the inexperienced solderer. However, they make a demonstration board, the ADXL202EB, that's very easy to integrate with a breadboard. The ADXL202E

Slip Rings

There is a problem that frequently accompanies sensing rotation. If you have electronics on a turning part that's wired to a non-turning part, the wires will get tangled and eventually limit the rotation. The simplest way to solve this problem is to make the rotating part's electronics self-contained, or to use wireless technologies. This means the rotating part will probably need batteries at least and much more supporting electronics at most. To avoid this, you can use slip rings as conductors for the wires on the rotating parts. *Slip rings* contain bands of metal conductors, typically on the shaft, to which the rotating electronics are attached. Springy conductors attached to the non-rotating part press against these bands to conduct without being attached. Slip rings introduce some electrical noise because the conductors are grinding against each other, so they're usually machined very precisely to reduce this. They're usually fairly expensive as a result. A quarter-inch stereo phono jack can work as a cheap and dirty slip ring for three conductors. Figure 9.10 shows a phono jack used as a slip ring and a professional slip ring.

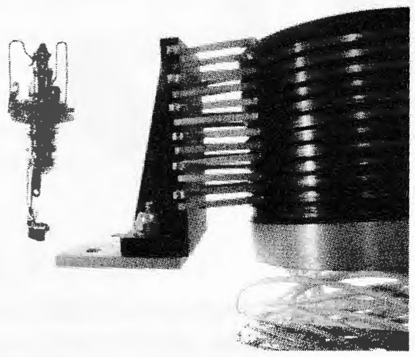
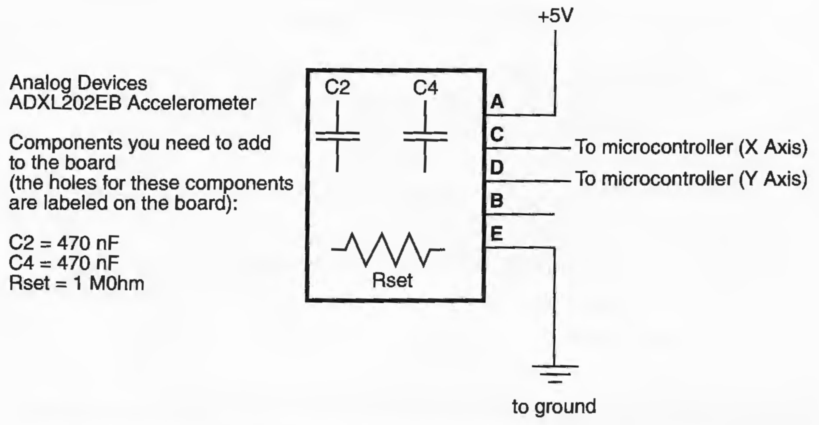


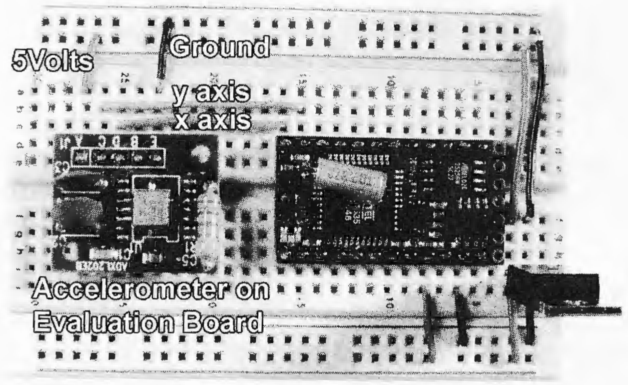
Figure 9.10
A phono jack used as a slip ring (left) and a professional slip ring (right).

has two output pins, one for the X axis and one for the Y axis. Both pins output a digital pulse whose width varies with the acceleration on that axis. You can measure the pulse width using the `rctime` command or the `pulsin` command. Analog also makes a second demonstration board, the ADXL202-EB-232, that gives you the X and Y accelerations via RS232 serial communication. This board costs over $200, though, so we don't recommend it. Figure 9.11 shows how to connect the ADXL202EB to a microcontroller.

To read the ADXL202E from a microcontroller, you simply listen for pulses coming in on the X pin and the Y pin. For most human-scale applications, just reading the varying numbers will give you a good enough resolution to do what you need to do. If you actually need to calculate the acceleration in meters per second squared, consult the data sheet available for download from Analog Devices (http://www.analog.com). The example below returns the raw numbers only.

Figure 9.11
An ADXL202EB accelerometer module connected to a microcontroller.


Analog Devices
ADXL202EB Accelerometer
Components you need to add to the board
(the holes for these components are labeled on the board):
C2 = 470 nF
C4 = 470 nF
Rset = 1 MOhm
+5V
C2
C4
Rset
A
C
D
B
E
To microcontroller (X Axis)
To microcontroller (Y Axis)
to ground



[redacted]

Compass

Accelerometers cannot read rotation around the vertical axis, or *pan* angle. For this you can use an electronic compass. Electronic compasses work just like conventional compasses, using the earth's magnetic field to determine rotation. Readings from a compass are not perfectly accurate (±5 degrees) and they don't react very quickly, but for rough readings, they're relatively easy to use. Reasonably priced models like the Devantech CMPS03 (available from http://www.acroname.com and others) send their reading back to your microcontroller using synchronous serial communication (see Chapter 12 for details on this method of communication).

Encoders

Encoders combine a rotating wheel and a light sensor to sense rotation. The heart of a common encoder is a wheel with slits cut in it. A phototransistor and an LED are mounted on either side of the wheel, aimed at it. The LED's light will pass through the slits and be received by the phototransistor, but it will be blocked by the spokes between the slits. Your microcontroller can then count pulses from the phototransistor to find out how many slits have passed and, therefore, how far the wheel has turned. This is a relative reading as opposed to the absolute reading that a pot gives you. If your microcontroller misses a pulse, your relative position shifts, and you have inaccurate information about the wheel's position. Generally, encoders are more effective for measuring speed than position. Encoders are common in older computer mice. Two encoders are positioned on two sides of a ball, which rotates both encoders at once. One encoder is used for reading movement along the mouse's long axis, and one measures movement laterally. If you've ever picked up your mouse to continue dragging across the screen, you've seen how encoders are relative and not absolute. The mouse's position on the pad isn't measured, but its speed across the pad is. This is why the cursor doesn't move much when you lift the mouse but does move when you drag the mouse across the pad. Figure 9.12 shows an encoder in a typical mouse.

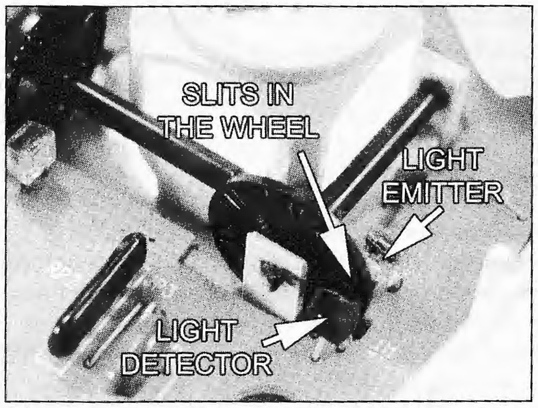


Figure 9.12
A rotary encoder in a mouse.

There is another type of encoder, called an *absolute encoder*, that does the work of counting the pulses for you and delivers an absolute position. Absolute encoders sound great, but they typically have less resolution than potentiometers. Both types of encoders can continuously move 360 degrees around a circle.

Speed of Rotation

Quite often, you need to know speed of rotation rather than distance of rotation. A tachometer is a typical device for measuring rotational speed. To build a tachometer, you only need one digital sensor on the side of the wheel. Magnetic switches work well for this because they require no physical connection to the wheel. You put the magnet on the wheel and the magnetic switch in a fixed position beside the wheel. With each rotation, the magnet turns on the switch. Count the on-off transitions per minute, and you've got the revolutions per minute (RPMs). Edge detection and debouncing (refer to Chapter 8) will be important for fast moving wheels. You will see this set up on most bicycle tachometers. To measure rotation speeds of less than a full rotation, you'll need to add more magnets at a regular spacing, or you'll need a potentiometer or an encoder.

Gyroscopes

Gyroscopes are sensors that measure angular acceleration. They're similar to accelerometers, except that they measure how fast the angle of rotation is changing, rather than measuring acceleration in a straight line. They can be very useful for measuring rotation around an axis and for measuring speed of rotation.

[redacted]

11

Touch Me

This chapter is about the point of physical contact between your body and your software. There are great opportunities for input and output through the huge sensitive membrane that is your skin and through the muscles under it. Two modes of sensation used in concert (for example, touch and hearing, or touch and sight) have much greater effect than either one alone, particularly when you're trying to build the illusion of immersive virtual reality. The study and implementation of techniques and psychological effects of this type of physical interface is called *haptics*. Some areas of your body come quickly to mind when imagining tactile interface, like the hands and feet, because we manipulate things with them, or the face and head because we're very sensitive to touch there. On the input side, there are some very easy-to-use items like force sensitive resistors, thermistors, and capacitance sensors. We will even touch on some techniques for sensing changes across your skin to read your unconscious reactions. On the output side, there are some easy devices that allow you to generate sensations of vibration and temperature. There are many other areas of the body worth considering as points of contact. Output that reaches your larger muscles falls into an area called *force feedback*, which is mostly by the use of larger motor and mechanical systems, covered in Chapter 10.

Force-Sensitive Resistors

The force-sensitive resistor (FSR) is one of the best sensors in the physical computing tool kit. FSRs convert mechanical force into electrical resistance. Like any other variable resistor, they generally have two leads and fit very easily into the analog input circuit shown in Chapter 6. They come in a variety of forms, as shown in Figure 11.1, but all are generally small and flat. They require a firm yet slightly pliable backing to work properly, so even if you are embedding them in something amorphous like a pillow, be sure to mount them on a flat surface first. The leads can be quite flimsy, and their movement will add error to the sensor's reading. To prevent this, mount everything, including your connection to the FSR, on a stiff backing and encase the contact end with heat shrink or hot glue (don't encase the whole sensor in hot glue, or it won't work!).

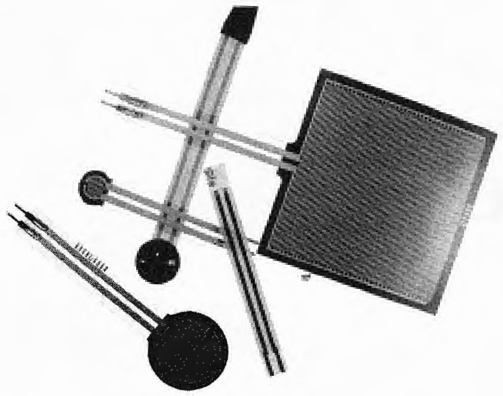


Figure 11.1
Force-sensitive resistors come in various shapes and sizes.

FSRs are typically designed to sense small amounts of force, such as the force of your finger pressing a button or keypad. The force of your body weight will quickly bring these sensors to their peak reading. This is useful if you only want to use them as a digital input to tell if a person is stepping on them or not. Digi-Key carries a range of good FSRs made by CUI Incorporated. You can find them on Digi-Key's site listed as force sensors. Look for part number 102-1212-ND thorough 102-1227-ND. Tekscan's FlexiForce sensors are also popular. Parallax (http://www.parallax.com), Images SI (http://www.imagesco.com), and others carry them. Interlink Electronics sells a design kit (part number 50-76247 at http://www.interlinkelec.com) with a variety of FSR forms, including long strips and small circle pads. They sell these sensors individually as well, at about $3.00 per sensor, but only in lots of about $60.00 or more. If you need a large number of FSRs, it's a good deal. You can also get FSR pads that sense position, like the Infusion Systems' SlideLong and SlideWide (http://www.infusionsystems.com).

Homegrown FSRs

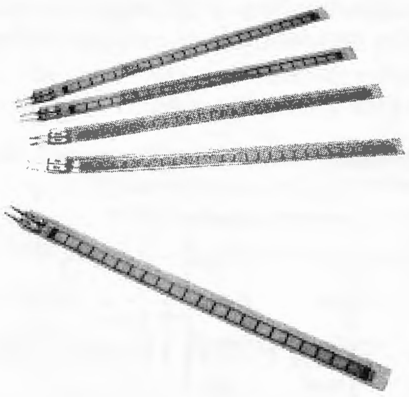
FSRs have a conductive form of rubber inside that lessens its resistance to current flow the more it is compressed. You can build your own FSRs using materials like conductive foam or conductive rubber (Zoflex) that behave this same way. You probably have some conductive foam because chips like your microcontroller are generally shipped in a small piece of it. You can also buy larger pieces of it (Radio Shack part number 276-2400, or Jameco part number 13864). You simply sandwich the conductive foam between two conductors and read the resistance between the conductors. Any copper foil or copper mesh that solder will stick to will work for the conductors. Metal screen door mesh works well. Building your own FSR allows you to vary the sensitivity by adding more material between the sensors. Creating your own sensor might also be necessary if you need to fit it into an unusual space like a small nose on a doll. Conductive foam will tend to lose some of its sensitivity over time, and you will have to recalibrate. You can buy Zoflex conductive rubber (http://www.irmicrolink.com) in sheets, or your can form it yourself. It works similarly to conductive foam.

Force sensors are inherently subject to wear and tear, and getting consistent values out of them is difficult. Unless you are a skilled fabricator, you're better off buying the professionally produced FSRs.

Flex Sensors

Flex sensors, shown in Figure 11.2, look and work much like FSRs, but they vary resistance based on how much you bend them instead of how much you press on them. They come in the form of a flat plastic strip that can be bent up to 180 degrees, increasing resistance from about 10 kilohms to 40 kilohms. They have two leads and connect easily into an analog input circuit. The obvious application is for sensing things that bend like fingers and hinges. They show up in a number of different virtual reality control gloves. They are so easy to work with you can consider them for sensing other types of rotary or linear movement. For example, a flex sensor could be used to sense how far out a small drawer has been pulled or how far a door has been opened. You can find flex sensors at Infusion Systems, Jameco, and Images SI.

Figure 11.2
Flex sensors.



Pressure Sensors

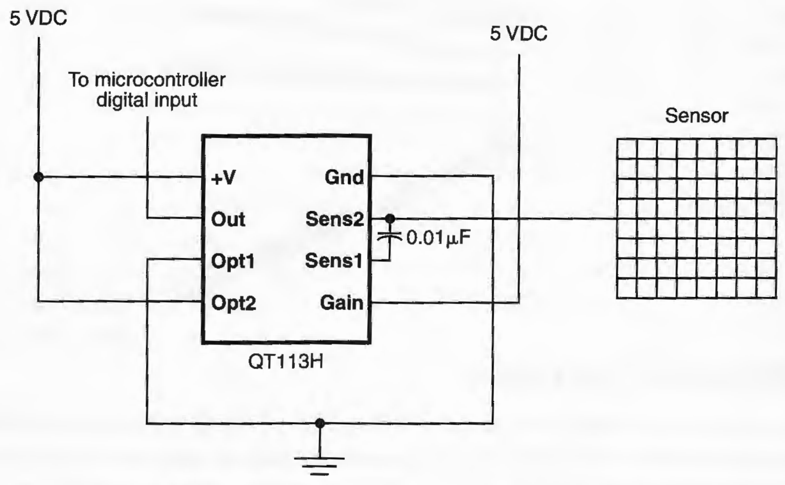
One common mistake people make when they're looking for force sensors is to search for pressure sensors. Pressure sensors measure pressure exerted by a gas or fluid. They're most commonly used in hydraulic or pneumatic applications, where you need to know the pressure of a gas or a liquid as it moves through a tube or a valve. Sometimes they're used in meteorological applications too, for example, to take measurements of atmospheric pressure (barometers). These sensors often have a small tube attached in order to couple with a valve. They can also be used as breath sensors, such as Infusion Systems' Air sensor. Digi-Key and Jameco also carry a number of pressure sensors that produce a varying voltage or resistance in response to changing pressure. Look for one that operates at the voltage or resistance levels that your microcontroller can sense.

Sensing Touch Using Capacitance Sensors

For sensing the very lightest touch, you should look into capacitance sensors. Your body is something of a capacitor, always storing a small electric charge. The shock you get when you touch a grounded object after walking across a carpet is the discharging of your body's stored charge. Some capacitance sensors can detect your body's charge from distances of up to a meter and are used to measure analog distance. This is the technique used by the

Theremin musical instrument (http://www.thereminworld.com). Most capacitance distance sensors require quite a lot of additional circuitry, so they aren't as precise or reliable as the infrared or ultrasonic sensors mentioned in Chapter 9. Capacitance sensors are most useful as touch sensors or near-touch sensors. You can create buttons that work with only the slightest touch of a finger. Capacitance sensors can be adjusted to detect your finger just before it actually touches (within a few millimeters), which can give an object a magical feeling. Because they have no moving parts and can be hidden below thin conductive or non-conductive surfaces like cloth or cardboard, they offer many options for making your buttons not look like buttons. Capacitance sensors are not limited to sensing your finger or your body. These sensors can detect any object that carries a static charge.

Quantum Technologies' Qtouch sensor ICs put all the circuitry for capacitive sensing into a chip for you. Digi-Key sells many models, for example the QT113H, part number 427-1012-ND. The QT113H, shown in Figure 11.3, produces a digital output at 5 volts when a person touches the sensor. A microcontroller can read this as a simple digital input.



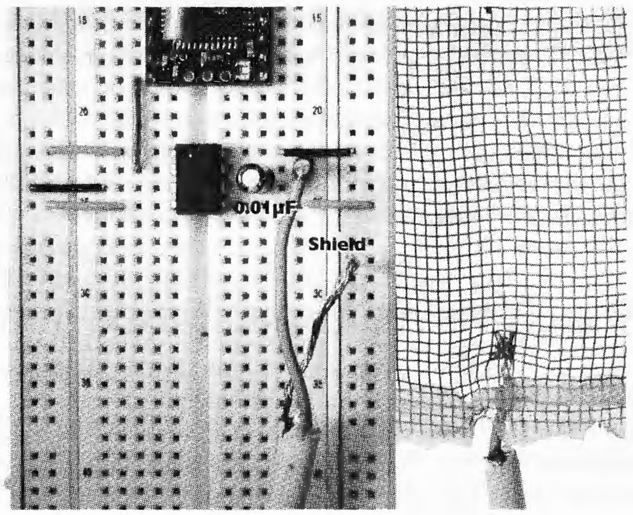


Figure 11.3
The QT113H capacitive sensor schematic (top) and circuit (bottom). Note that the shield foil from the cable is connected to ground.

You have to connect a sensor, which does not come with the chip. We've found copper mesh or copper foil works nicely as the sensor. The connection between the sensor chip and the sensor itself also reacts to changes in capacitance, so you'll find that even touching the wire between the two will trigger a reaction. This can be a problem if you only want a very localized area to be sensitive, or if the sensor wire has to run near other current-carrying wires. To get around this, use a shielded conductor between the chip and the sensor. A *shielded conductor* is a wire or a group of wires wrapped in a foil or mesh wrapping. Many video and speaker cables and computer cables are shielded. If you strip back the outer insulation jacket of these cables, you'll find the foil or mesh wrapping surrounding the inner wires. Don't remove it. Connect it to the ground of your circuit to make the conductors insensitive to capacitive changes.

The important pins on this chip are power and ground, the sense pins, and the output. You can use either of the sense pins for your sensor, then put a capacitor in between them, as shown in the schematic in Figure 11.3. The option pins on the Qtouch ICs allow you to set various options like the sensitivity, the reset time (how long the sensor takes to go low again if no change is detected), and more. Read the data sheet for all the options.

Quantum makes a whole family of these chips (http://www.qprox.com) with various options, including having multiple switches per chip. They also make a few analog capacitance sensors, like the QT300, which give you a changing value when they detect movement within a centimeter or so of the sensor.

Off-the-Shelf Touch Interfaces

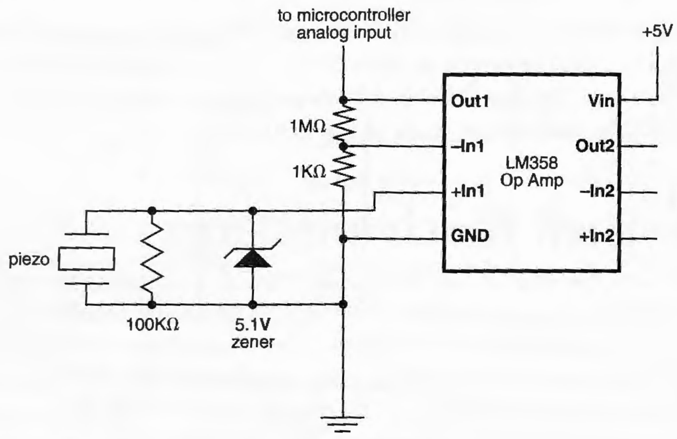
Touch screens are the easiest way to connect touch with computer graphics. There are many suppliers that integrate touch screens into ordinary monitors at increasingly affordable prices. In addition to the usual video connection these will have a serial or USB connection for the touch information. After you install the drivers, your software can get the touch location by simply reading the mouse coordinates. You can also buy screens that can be attached to conventional monitors. Elo TouchSystems (http://www.elotouch.com) makes a number of both integrated touch screen monitors and add-on touch screens. You can usually find these in surplus at very reasonable prices. All Electronics (http: //www.allcorp.com) frequently has a number of different models in stock. You may even consider using these as sensitive touch panels without attaching them to a monitor.

The disadvantage of touch screens is that they generally will only read contact at one point at a time. This means that if you put all five fingertips on the screen, you won't get five discrete locations. Instead, the screen will report several different locations in succession in an inconsistent order. If you want a multitouch touchpad, your options are more limited. Tactex Systems made a wonderful multitouch sensor board, the MTCExpress, but it has been discontinued. Used models show up on eBay frequently and are coveted by musicians.

Sensing Vibrations Using Piezoelectric Sensors

Piezoelectric sensors are often used to detect strain or very slight force changes. These sensors produce a varying voltage when they are bent. They respond very quickly, and to slight changes. Microphones are made of piezoelectric film. In fact, if you connected a piezo to the microphone amplifier circuit in Chapter 13, "Controlling Sound and Light," it would work well as a knock detector. With a fast processor like a PIC, you could use two piezos on a table and time the arrival of their signals to determine the position of a knock on the table. Piezos are used for all kinds of sensors beyond microphones, from sensing strain on bridges to the hitting of a drum to the pitter-patter of children's footsteps in flashing sneakers.

The voltage range that a piezo sensor can produce can be up to a few thousand volts, and it can generate changes as small as a few microvolts. Because of this, piezos can be difficult to read across their entire sensitivity range. One common way to read miniscule changes from a piezo is to use an *operational amplifier*, or *op amp*. Op amps take a very small voltage signal and amplify it to a range that's readable. They can be used for many other functions as well, like reading the difference between two voltages, the sum of two voltages, and more. They've got a reputation among physical computing hobbyists for being difficult to use, but they don't have to be.[1] Figure 11.4 shows one very simple-to-use op amp used to measure the changes from a piezo.



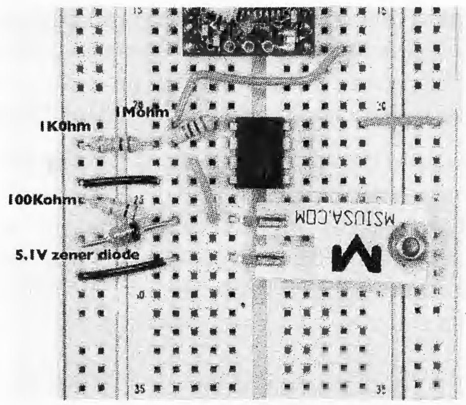


Figure 11.4
A piezo sensor and an LM358 op amp to amplify the voltage changes.

[1] There's much more to be said about op amps. For a more lengthy and technical introduction, see *Practical Electronics for Inventors* by Paul Scherz (McGraw-Hill/TAB Electronics, 2000). If you're really itching to dive into the subject in depth, see *The IC Op Amp Cookbook* by Walter C. Jung (Prentice Hall PTR, 1986).

This op amp, the LM358, is called a *single-supply* op amp because it only needs a positive voltage and ground. Many op amps are *dual-supply*, meaning that they need a positive voltage, ground, and a negative voltage. The output from this circuit will vary between 0 and 5 volts, depending on your piezo and what it's physically mounted to. You can read it with an analog-to-digital converter. If you're using the BS-2, you'll find that the RCTime circuit doesn't catch all the changes from this circuit because the piezo changes voltages very fast. The more extreme the piezo is bent, the more voltage it will produce.

The sensor in the circuit in Figure 11.4 is a piezo film sensor with vibrating mass from Measurement Specialties (http://www.msiusa.com/sensors). It's available from many other suppliers, including Digi-Key as part number MSP-1007-ND. You can use almost any piezo element, though. A piezo speaker cut out of a child's toy used with the same circuit is sensitive enough to respond to a very delicate touch, even to blowing.

You can change the amplification factor of the op amp by changing the two resistors attached to the negative input pin. In this circuit, they're 1 kilohm and 1 megohm, giving an amplification of 1000:1.

The unusual diode attached to the piezo is called a *zener diode*. You'll see it again in Chapter 13, when you build a telephone line interface. A zener diode allows electrical energy to pass up to a certain voltage and cuts the rest off. In this circuit, the zener diode is rated for 5.1 volts, and it's used to send to ground any voltage that the piezo produces above 5 volts.

[redacted]