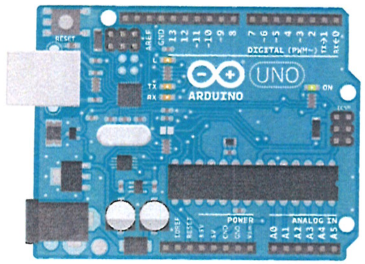
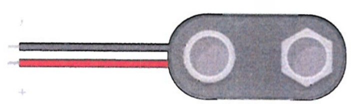
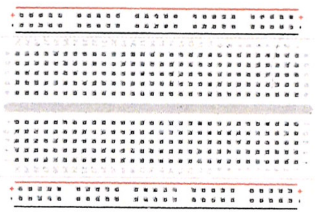
PARTS IN YOUR KIT



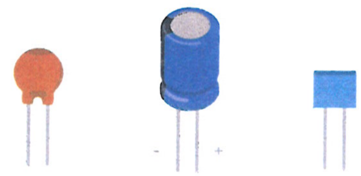
Arduino Uno - The microcontroller development board that will be at the heart of your projects. It's a simple computer, but one that has no way for you to interact with it yet. You will be building the circuits and interfaces for interaction, and telling the microcontroller how to interface with other components.



Battery Snap - Used to connect a 9V battery to power leads that can be easily plugged into a breadboard or your Arduino.



Breadboard - A board on which you can build electronic circuits. It's like a patch panel, with rows of holes that allow you to connect wires and components together. Versions that require soldering are available, as well as the solder-less type used here.

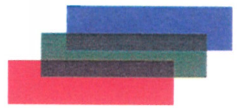


Capacitors - These components store and release electrical energy in a circuit. When the circuit's voltage is higher than what is stored in the capacitor, it allows current to flow in, giving the capacitor a charge. When the circuit's voltage is lower, the stored charge is released. Often placed across power and ground close to a sensor or motor to help smooth fluctuations in voltage.

DC motor - Converts electrical energy into mechanical energy when electricity is applied to its leads. Coils of wire inside the motor become magnetized when current flows through them.

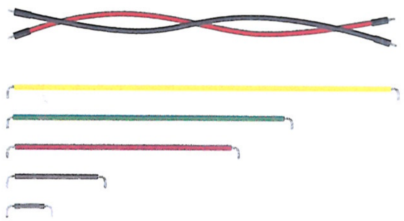
nets, causing the shaft to spin. If the direction of the electricity is reversed, the motor will spin in the opposite direction.

Diode - Ensures electricity flows only in one direction. Useful when you have a motor or other high current/voltage load in your circuit. Diodes are polarized, meaning that the direction that they're placed in a circuit matters. Placed one way, they allow current to pass through. Placed the other way, they block it. The anode side generally connects to the point of higher energy in your circuit. The cathode typically connects to the point of lower energy, or to ground. The cathode is usually marked with a band on one side of the component's body.

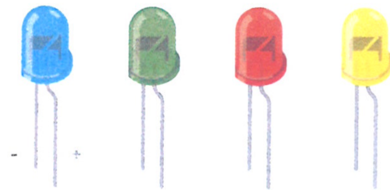


Gels (red, green, blue) - These filter out different wavelengths of light. When used in conjunction with photoresistors, they cause the sensor to only react to the amount of light in the filtered color.

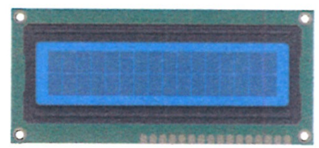
H-bridge - A circuit that allows you to control the polarity of the voltage applied to a load, usually a motor. The H-bridge in the kit is an integrated circuit, but it could also be constructed with a number of discrete components.



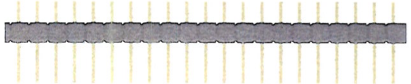
Jumper wires - Use these to connect components to each other on the breadboard, and to the Arduino.



Light Emitting Diodes (LEDs) - A type of diode that illuminates when electricity passes through it. Like all diodes, electricity only flows in one direction through these components. You're probably familiar with these as indicators on a variety of electronic devices. The anode, which typically connects to power, is usually the longer leg, and the cathode is the shorter leg.

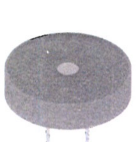


Liquid Crystal Display (LCD) - A type of alphanumeric or graphic display based on liquid crystals. LCDs are available in a many sizes, shapes, and styles. Yours has 2 rows with 16 characters each.



Male header pins - These pins fit into female sockets, like those on a breadboard. They help make connecting things much easier.

Optocoupler - This allows you to connect two circuits that do not share a common power supply. Internally there is a small LED that, when illuminated, causes a photoreceptor inside to close an internal switch. When you apply voltage to the + pin, the LED lights and the internal switch closes. The two outputs replace a switch in the second circuit.



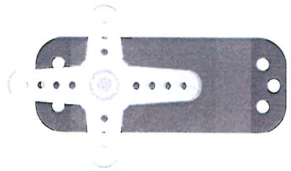
Piezo - An electrical component that can be used to detect vibrations and create noises.

Photoresistor - (also called a photocell, or light-dependent resistor). A variable resistor that changes its resistance based on the amount of light that falls on its face.

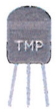
Potentiometer - A variable resistor with three pins. Two of the pins are connected to the ends of a fixed resistor. The middle pin, or wiper, moves across the resistor, dividing it into two halves. When the external sides of the potentiometer are connected to voltage and ground, the middle leg will give the difference in voltage as you turn the knob. Often referred to as a pot.

Pushbuttons - Momentary switches that close a circuit when pressed. They snap into breadboards easily. These are good for detecting on/off signals.

Resistors - Resist the flow of electrical energy in a circuit, changing the voltage and current as a result. Resistor values are measured in ohms (represented by the Greek omega character: Ω). The colored stripes on the sides of resistors indicate their value (see resistor color code table, p. 41).

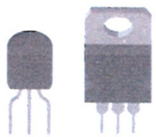


Servo motor - A type of geared motor that can only rotate 180 degrees. It is controlled by sending electrical pulses from your Arduino. These pulses tell the motor what position it should move to.



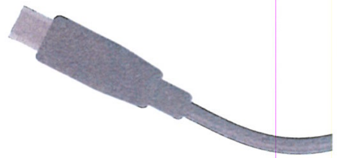
Temperature sensor - Changes its voltage output depending on the temperature of the component. The outside legs connect to power and ground. The voltage on the center pin changes as it gets warmer or cooler.

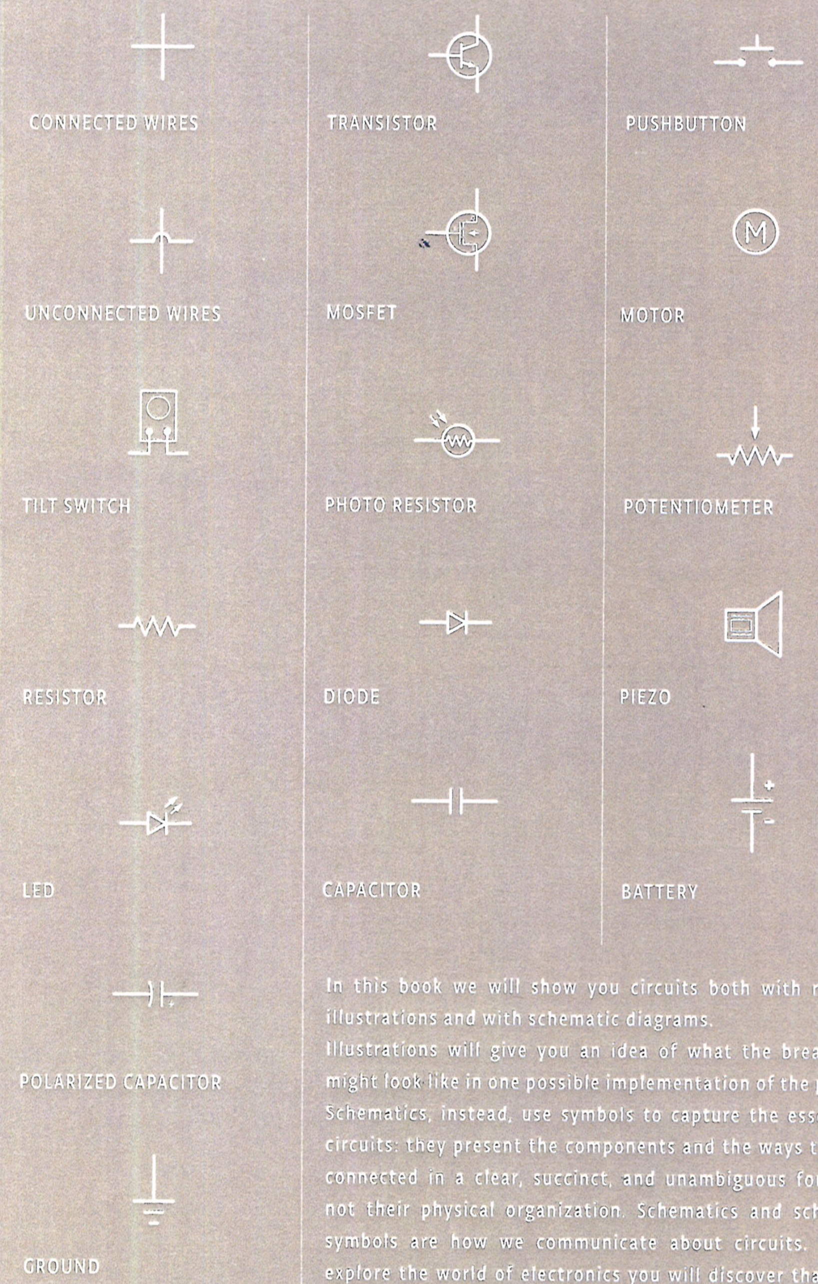
Tilt sensor - A type of switch that will open or close depending on its orientation. Typically they are hollow cylinders with a metal ball inside that will make a connection across two leads when tilted in the proper direction.



Transistor - A three legged device that can operate as an electronic switch. Useful for controlling high current/high voltage components like motors. One pin connects to ground, another to the component being controlled, and the third connects to the Arduino. When the component receives voltage on the pin connected to an Arduino, it closes the circuit between the ground and the other component.

USB Cable - This allows you to connect your Arduino Uno to your personal computer for programming. It also provides power to the Arduino for most of the projects in the kit.





In this book we will show you circuits both with realistic illustrations and with schematic diagrams.
Illustrations will give you an idea of what the breadboard might look like in one possible implementation of the project. Schematics, instead, use symbols to capture the essence of circuits: they present the components and the ways they are connected in a clear, succinct, and unambiguous form, but not their physical organization. Schematics and schematic symbols are how we communicate about circuits. As you explore the world of electronics you will discover that some books and websites only provide schematic diagrams, so learning to read circuits this way is a valuable skill.
Here are the symbols we will be using throughout the book.

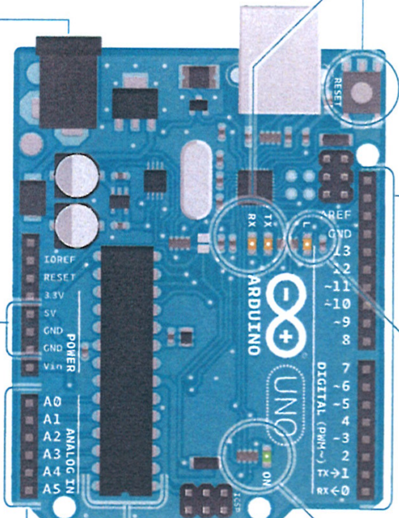
THE ARDUINO BOARD

Power connector
This is how you power your Arduino when it's not plugged into a USB port for power. Can accept voltages between 7-12V.

USB port
Used for powering your Arduino Uno, uploading your sketches to your Arduino, and for communicating with your Arduino sketch (via Serial.println() etc.)

Reset Button
Resets the ATmega microcontroller.

TX and RX LEDs
These LEDs indicate communication between your Arduino and your computer. Expect them to flicker rapidly during sketch upload as well as during serial communication. Useful for debugging.



Digital pins
Use these pins with digitalRead(), digitalWrite(), and analogWrite(). analogWrite() works only on the pins with the PWM symbol.

Pin 13 LED
The only actuator built-in to your Arduino Uno. Besides being a handy target for your first blink sketch, this LED is very useful for debugging.

GND and 5V pins
Use these pins to provide +5V power and ground to your circuits.

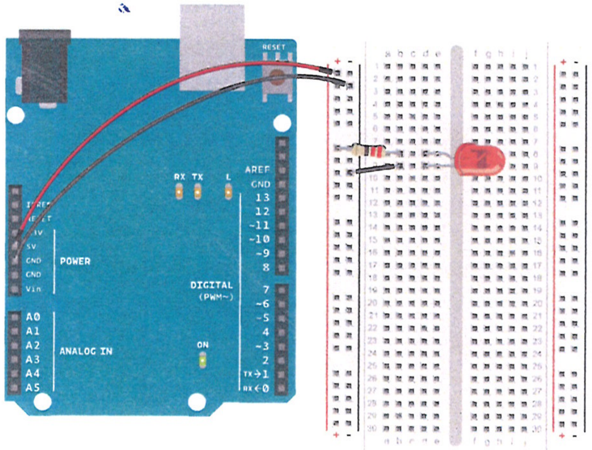
ATmega microcontroller
The heart of your Arduino Uno.

Analog in
Use these pins with analogRead().

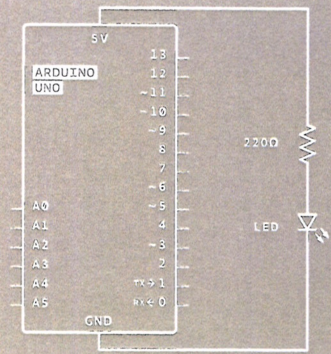
Power LED
Indicates that your Arduino is receiving power. Useful for debugging.

DRAWINGS

breadboard view (like in Fig. 5), that looks like the stuff in your kit. The other is a schematic view (like in Fig. 6), which is a more abstract way of showing the relationships between components in a circuit. Schematics don't always show where components are placed relative to each other, but they show how they are connected.

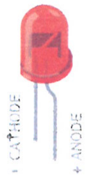


Circuit illustration.
Fig. 5



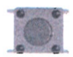
Schematic view
Fig. 6

COMPONENTS



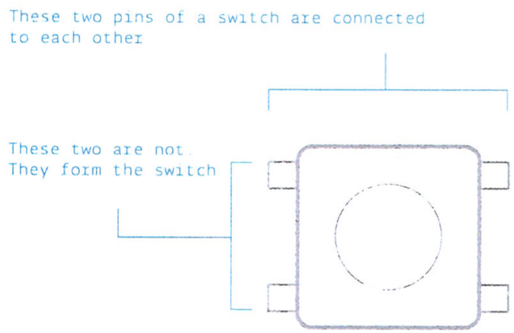
An ***LED***, or light-emitting diode, is a component that converts electrical energy into light energy. LEDs are polarized components, which means they only allow electricity to flow through them in one direction. The longer leg on the LED is called an anode, it will connect to power. The shorter leg is a cathode and will connect to ground. When voltage is applied to the anode of the LED, and the cathode is connected to ground, the LED emits light.

A ***resistor*** is a component that resists the flow of electrical energy (see the components list for an explanation on the colored stripes on the side). It converts some of the electrical energy into heat. If you put a resistor in series with a component like an LED, the resistor will use up some of the electrical energy and the LED will receive less energy as a result. This allows you to supply components with the amount of energy they need. You use a resistor in series with the LED to keep it from receiving too much voltage. Without the resistor, the LED would be brighter for a few moments, but quickly burn out.

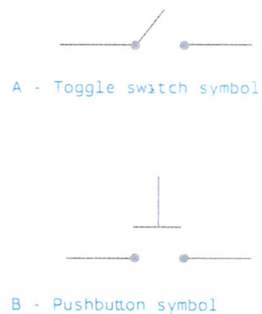


A ***switch*** interrupts the flow of electricity, breaking the circuit when open. When a switch is closed, it will complete a circuit. There are many types of switches. The ones in your kit are called momentary switches, or pushbuttons, because they are only closed when pressure is applied.

SWITCH CONNECTIONS



SWITCH SCHEMATIC VIEW



The switch
Fig. 7

BUILD THE CIRCUIT

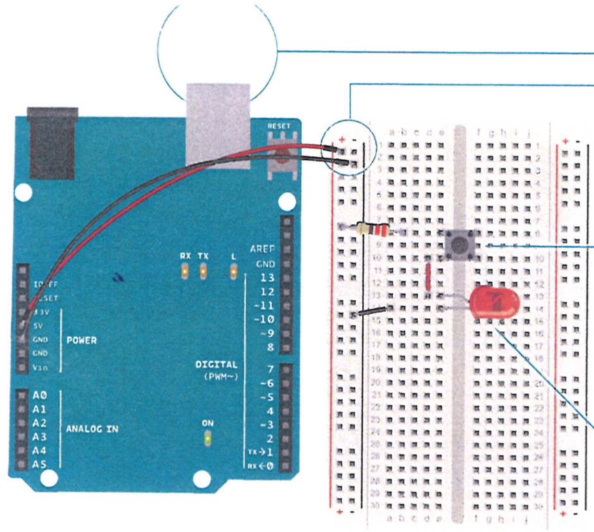


Fig. 8

Your first interactive circuit, using a switch, a resistor and an LED. Arduino is just the power source for this circuit; in later projects, you'll connect its input and output pins to control more complex circuits.

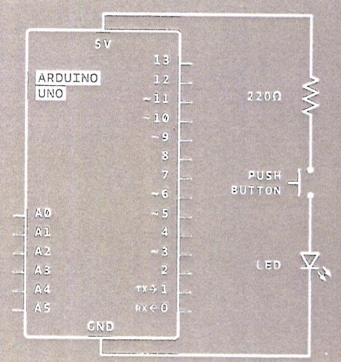


Fig. 9

You're going to use the Arduino in this project, but only as a source of power. When plugged into a USB port or a 9-volt battery, the Arduino will provide 5 volts between its 5V pin and its ground pin that you can use. 5V = 5 volts, you'll see it written this way a lot.

1. If your Arduino is connected to a battery or computer via USB, unplug it before building the circuit!

2. Connect a red wire to the 5V pin on the Arduino, and put the other end in one of the long bus lines in your breadboard. Connect ground on the Arduino to the adjacent bus line with a black wire. It's helpful to keep your wire color consistent (red for power, black for ground) throughout your circuit.

3. Now that you have power on your board, place your switch across the center of the board. The switch will sit across the center in one direction. The bend in the legs of the switch points to the center of the board.

4. Use a 220-ohm resistor to connect power to one side of the switch. The illustrations in this book use 4 bands. Your kit may have a mix of 4 and 5 band resistors. Use the illustration on the side to check for the right one for this project. Look at page 41 for a detailed explanation of the color codes for resistors.
On the other side of the switch, connect the anode (long leg) of the LED. With a wire connect the cathode (short leg) of the LED to ground. When you're ready, plug the USB cable into the Arduino.

USE IT

Once everything is set to go, press the button. You should see the LED light up. Congratulations, you just made a circuit! Once you've tired of pressing the button to turn the light on, it's time to shake things up by adding a second button.

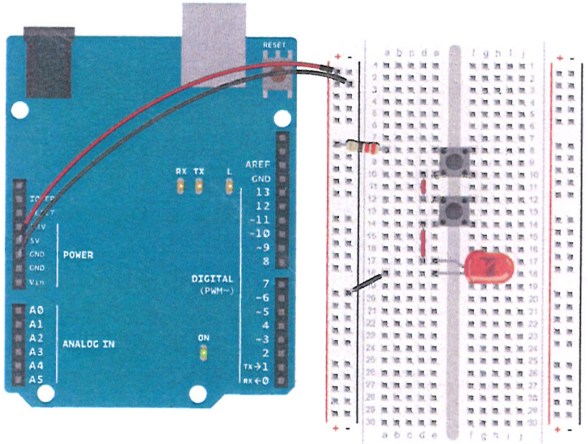
You'll be placing components on the breadboard in series and in parallel. Components in series come one after another. Components in parallel run side by side.

COMPONENTS IN SERIES COME ONE AFTER ANOTHER

These two elements are in series

Once you've removed your power source add a switch next to the one already on your breadboard. Wire them together in series as shown in Fig. 10. Connect the anode (long leg) up the LED to the second switch. Connect the LED cathode to ground. Power up the Arduino again: now to turn on the LED, you need to press both switches. Since these are in series, they both need to be closed for the circuit to be completed.

ALWAYS REMOVE POWER BEFORE CHANGING ANYTHING IN YOUR CIRCUIT



The two switches are in series. This means that the same electrical current flows through both of them, so that they both have to be pressed for the LED to light up.

Fig. 10

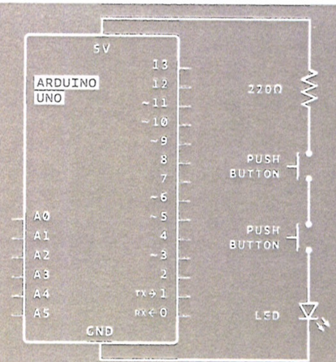
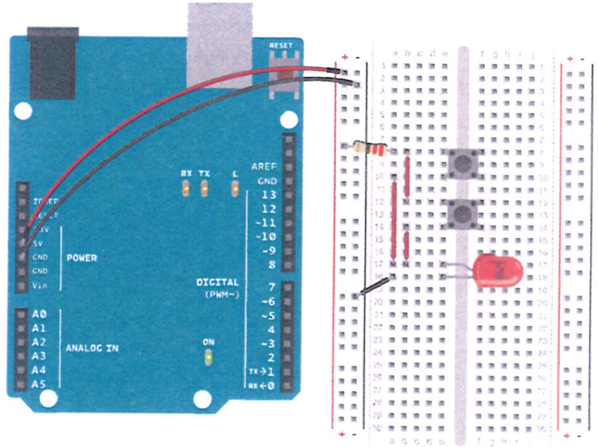


Fig. 11

COMPONENTS IN PARALLEL RUN SIDE BY SIDE

These two elements are in parallel

Now that you've mastered the art of things in series, it's time to wire up switches in parallel. Keep the switches and LED where they are, but remove the connection between the two switches. Wire both switches to the resistor. Attach the other end of both switches to the LED, as shown in Fig. 12. Now when you press either button, the circuit is completed and the light turns on.



These two switches are in parallel. This means that the electrical current is split between them. If either switch is pressed, the LED will light up.

Fig. 12

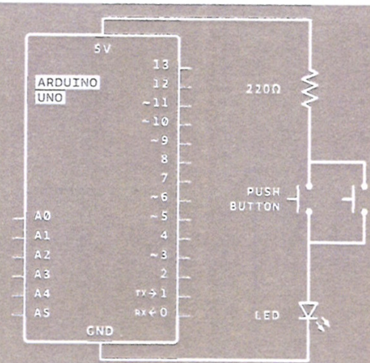


Fig. 13

CIRCUIT

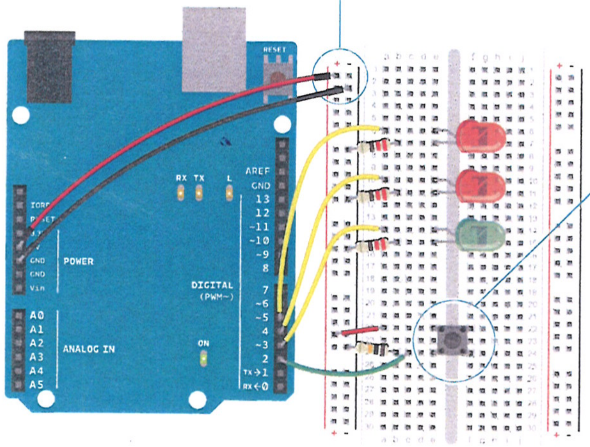


Fig. 1

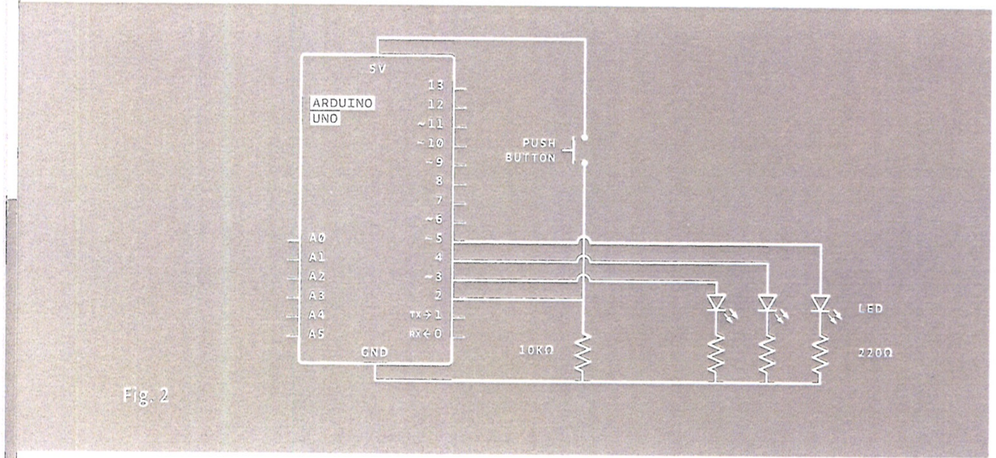
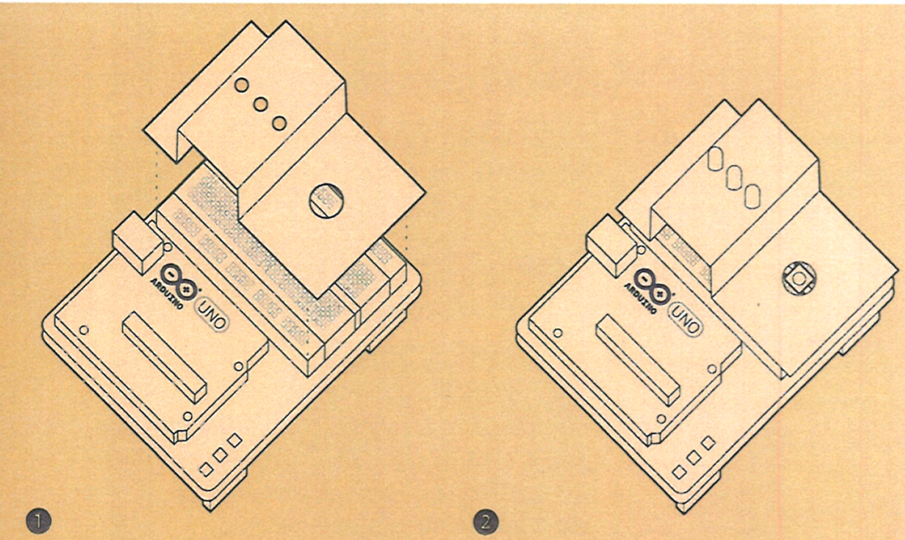


Fig. 2

1. Wire up your breadboard to the Arduino's 5V and ground connections, just like the previous project. Place the two red LEDs and one green LED on the breadboard. Attach the cathode (short leg) of each LED to ground through a 220-ohm resistor. Connect the anode (long leg) of the green LED to pin 3. Connect the red LEDs' anodes to pins 4 and 5, respectively.

2. Place the switch on the breadboard just as you did in the previous project. Attach one side to power, and the other side to digital pin 2 on the Arduino. You'll also need to add a 10k-ohm resistor from ground to the switch pin that connects to the Arduino. That pull-down resistor connects the pin to ground when the switch is open, so it reads LOW when there is no voltage coming in through the switch.

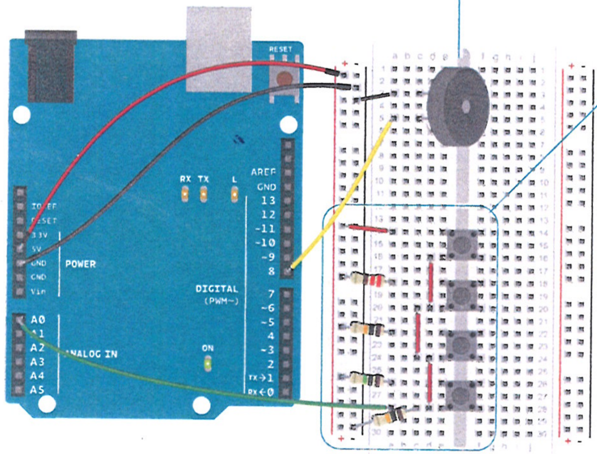
You can cover the breadboard with the template provided in the kit. Or you can decorate it to make your own launch system. The lights turning on and off mean nothing by themselves, but when you put them in a control panel and give them labels, they gain meaning. What do you want the green LED to mean? What do the flashing red LEDs mean? You decide!



1 Fold the pre-cut paper as shown.

2 Place the folded paper over the breadboard. The three LEDs and pushbutton will help keep it in place.

BUILD THE CIRCUIT



The arrangement of resistors and switches feeding into an analog input is called a resistor ladder.

Fig. 2

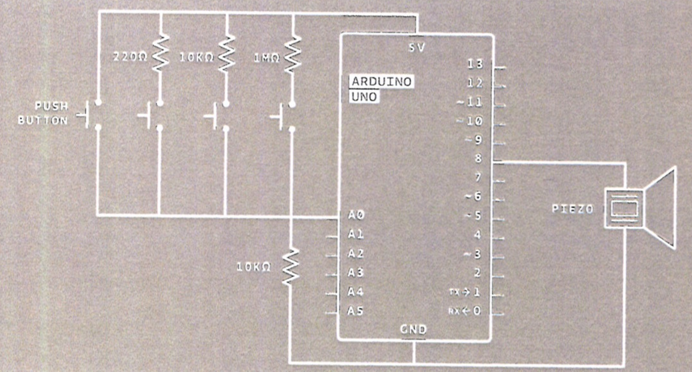
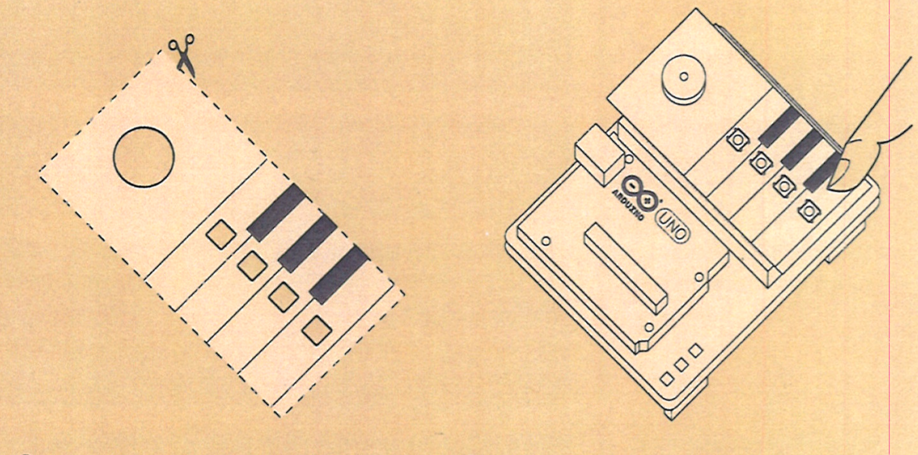


Fig. 3

1. Wire up your breadboard with power and ground as in the previous projects. Connect one end of the piezo to ground. Connect the other end to pin 8 on your Arduino.

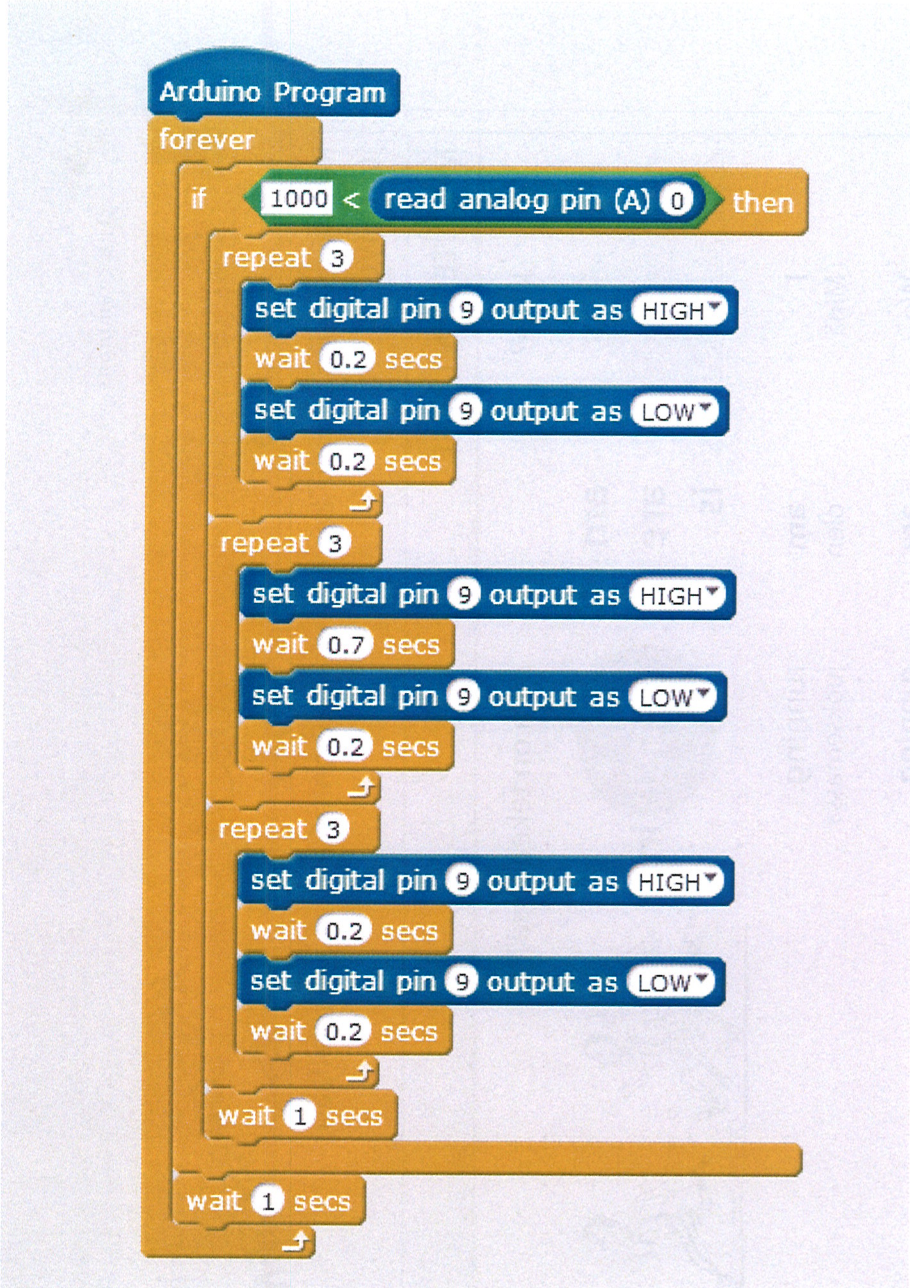
2. Place your switches on the breadboard as shown in the circuit. The arrangement of resistors and switches feeding into an analog input is called a resistor ladder. Connect the first one directly to power. Connect the second, third and fourth switches to power through a 220-ohm, 10-kilohm and 1-megohm resistor, respectively. Connect all the switches' outputs together in one junction. Connect this junction to ground with a 10-kilohm resistor, and also connect it to Analog In 0. Each of these acts as a voltage divider.

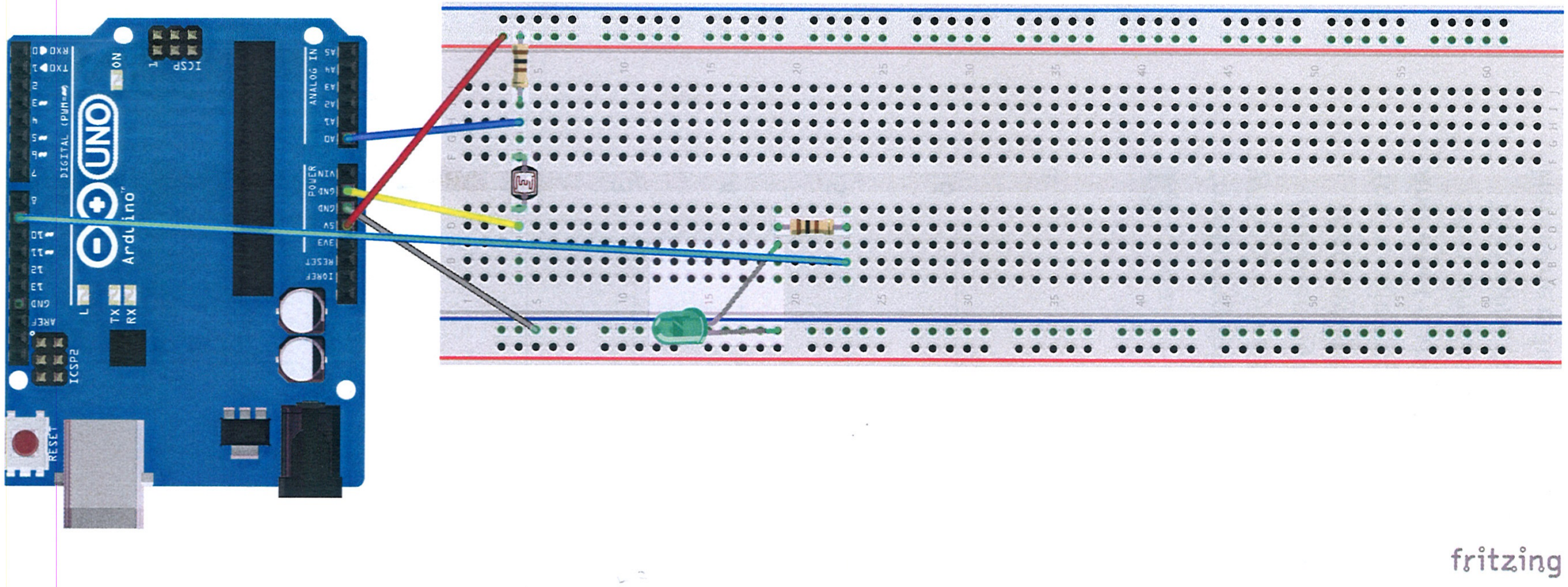
Think about an enclosure for the keyboard. While old analog synthesizers had wires poking out all over the place, your keyboard is sleek and digital. Prepare a small piece of cardboard that can be cut out to accommodate your buttons. Label the keys, so you know what notes are triggered by each key.



1. Draw and cut a piece of paper with holes for the four buttons and piezo. Decorate it to look like a piano keyboard.

2. Position the paper over the buttons and piezo. Enjoy your creation!


Arduino Program
forever
if 1000 < read analog pin (A) 0 then
repeat 3
set digital pin 9 output as HIGH
wait 0.2 secs
set digital pin 9 output as LOW
wait 0.2 secs
repeat 3
set digital pin 9 output as HIGH
wait 0.7 secs
set digital pin 9 output as LOW
wait 0.2 secs
repeat 3
set digital pin 9 output as HIGH
wait 0.2 secs
set digital pin 9 output as LOW
wait 0.2 secs
wait 1 secs
wait 1 secs



RESET
ICSP2
UNO
Arduino
DIGITAL (PWM~)
ICSP
POWER
ANALOG IN
fritzing