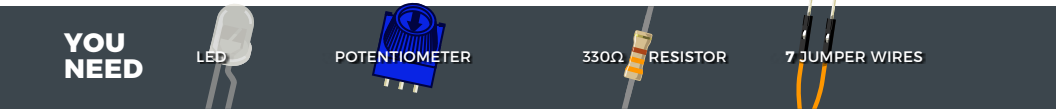
Circuit 1B: Potentiometer

Potentiometers (also known as “trimpots” or “knobs”) are one of the basic inputs for electronic devices. By tracking the position of the knob with your RedBoard, you can make volume controls, speed controls, angle sensors and a ton of other useful inputs for your projects. In this circuit, you’ll use a potentiometer as an input device to control the speed at which your LED blinks.



NEW COMPONENTS

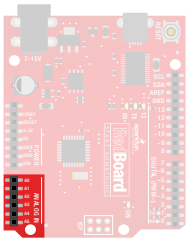
POTENTIOMETER: A potentiometer is a 3-pin variable resistor. When powered with 5V, the middle pin outputs a voltage between 0V and 5V, depending on the position of the knob on the potentiometer. Internal to the trimpot is a single resistor and a wiper, which cuts the resistor in two and moves to adjust the ratio between both halves.

NEW CONCEPTS

ANALOG VS. DIGITAL: We live in an analog world. There are an infinite number of colors to paint an object, an infinite number of tones we can hear, and an infinite number of smells we can smell. The common theme among these analog signals is their infinite possibilities.

Digital signals deal in the realm of the discrete or finite, meaning there is a limited set of values they can be. The LED from the previous circuit had only two states it could exist in, ON or OFF, when connected to a digital output.

ANALOG INPUTS: So far, we’ve only dealt with outputs. The RedBoard also has inputs. Both inputs and outputs can be analog or digital. Based on our previous definition of analog and digital, that means an analog input can sense a wide range of values versus a digital input, which can only sense two values, or states.



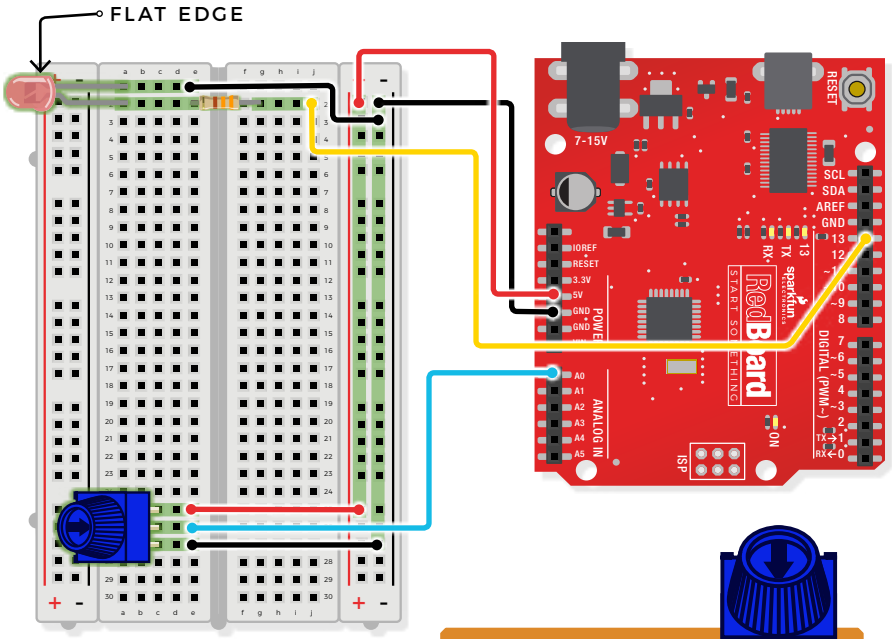
You may have noticed some pins labeled **Digital** and some labeled **Analog In** on your RedBoard. There are only six pins that function as analog inputs; they are labeled A0–A5.

VOLTAGE DIVIDER

VOLTAGE DIVIDERS are simple circuits that turn some voltage into a smaller voltage using two resistors. A potentiometer is a variable resistor that can be used to create an adjustable voltage divider. A wiper in the middle position means the output voltage will be half of the input. Voltage dividers will be covered in more detail in the next circuit.

HOOKUP GUIDE

READY TO START HOOKING EVERYTHING UP? Check out the circuit diagram and hookup table below to see how everything is connected.



NEW IDEAS

POTENTIOMETERS are not polarized and can be installed in either direction. Note that swapping the 5V and GND pins will reverse its behavior.

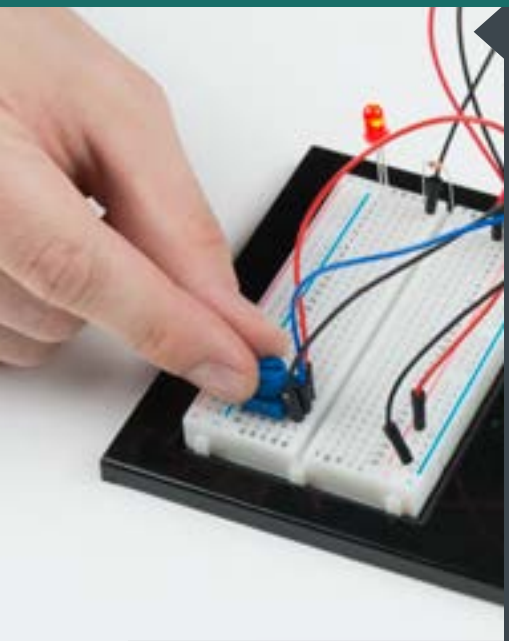
CONNECTION TYPES	◆ REDBOARD CONNECTION ■ BREADBOARD CONNECTION
JUMPER WIRES	◆ 5V to ■ 5V ◆ GND to ■ GND (-) ◆ A0 to ■ E26 ■ E25 to ■ 5V (+) ■ E27 to ■ GND (-) ■ E1 to ■ GND (-) ◆ D13 to ■ J2
LED	■ A1(-) to ■ A2(+)
330Ω RESISTOR (ORANGE, ORANGE, BROWN)	■ E2 to ■ F2
POTENTIOMETER	■ C25 + ■ C26 + ■ C27

Open the Arduino IDE

Connect the RedBoard to a USB port on your computer.

Open the Sketch:
File > File > Examples > SIK_Guide_Code-V_4 > **CIRCUIT_1B-POTENTIOMETER**

Select **UPLOAD** to program the sketch on the RedBoard.



WHAT YOU SHOULD SEE

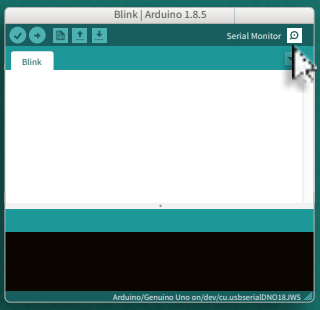
You should see the LED blink faster or slower in accordance with your potentiometer. The delay between each flash will change based on the position of the knob. If it isn't working, make sure you have assembled the circuit correctly and verified and uploaded the code to your board. If that doesn't work, see the Troubleshooting section.

PROGRAM OVERVIEW

1	Read the position of the potentiometer (from 0 to 1023) and store it in the variable **potPosition**.
2	Turn the LED on.
3	Wait from 0 to 1023 milliseconds, based on the position of the knob and the value of **potPosition**.
4	Turn the LED off.
5	Wait from 0 to 1023 milliseconds, based on the position of the knob and the value of **potPosition**.
6	Repeat.

ARDUINO PRO TIP

ARDUINO SERIAL MONITOR: The Serial Monitor is one of the Arduino IDE's many great included features. When working with embedded systems, it helps to see and understand the values that your program is trying to work with, and it can be a powerful debugging tool when you run into issues where your code is not behaving the way you expected it to. This circuit introduces you to the Serial Monitor by showing you how to print the values from your potentiometer to it. To see these values, click the Serial Monitor button, found in the upper-right corner of the IDE in most recent versions. You can also select **Tools** > **Serial Monitor** from the menu.



Serial Monitor button in the upper-right of the Arduino IDE.

Serial Monitor printout and baud-rate menu.

You should see numeric values print out in the monitor. Turning the potentiometer changes the value as well as the delay between each print.

If you are having trouble seeing the values, ensure that you have selected 9600 baud and have auto scroll checked.

CODE TO NOTE

INTEGER VARIABLES:

```
int potPosition;
```

A variable is a placeholder for values that may change in your code. You must introduce, or "declare," variables before you use them. Here we're declaring a variable called **potPosition** of type **int** (integer). We will cover more types of variables in later circuits. Don't forget that variable names are case-sensitive!

CODE TO NOTE

SERIAL BEGIN:

```
Serial.begin(9600);
```

Serial commands can be used to send and receive data from your computer. This line of code tells the RedBoard that we want to "begin" that communication with the computer, the same way we would say "Hi" to initiate a conversation. Notice that the baud rate, 9600, is the same as the one we selected in the monitor. This is the speed at which the two devices communicate, and it must match on both sides.

ANALOG INPUT:

```
potPosition =
analogRead(A0);
```

We use the `analogRead()` function to read the value on an analog pin. `analogRead()` takes one parameter, the analog pin you want to use, `A0` in this case, and returns a number between 0 (0 volts) and 1023 (5 volts), which is then assigned to the variable **`potPosition`**.

SERIAL PRINT:

```
Serial.
println(potPosition);
```

This is the line that actually prints the trimpot value to the monitor. It takes the variable **`potPosition`** and prints whatever value it equals at that moment in the `loop()`. The `ln` at the end of `println` tells the monitor to print a new line at the end of each value; otherwise the values would all run together on one line. Try removing the `ln` to see what happens.

CODING CHALLENGES

CHANGING THE RANGE: Try multiplying, dividing or adding to your sensor reading so that you can change the range of the delay in your code. For example, can you multiply the sensor reading so that the delay goes from 0–2046 instead of 0–1023?

ADD MORE LEDS: Add more LEDs to your circuit. Don't forget the current-limiting resistors. You will need to declare the new pins in your code and set them all to `OUTPUT`. Try making individual LEDs blink at different rates by changing the range of each using multiplcation or division.

TROUBLESHOOTING

The potentiometer always reads as 0 or 1023	Make sure that your 5V, A0 and GND pins are properly connected to the three pins on your potentiometer. It is easy to misalign a wire with the actual pot pin.
No values or random characters in Serial Monitor	Make sure that you have selected the correct baud rate, **9600**. Also ensure that you are on the correct serial port. The same serial port you use when uploading code to your board is the same serial port you use to print values to the Serial Monitor.

You've completed Circuit 1B!

Continue to circuit 1C to learn about photoresistors and analog to digital conversion.