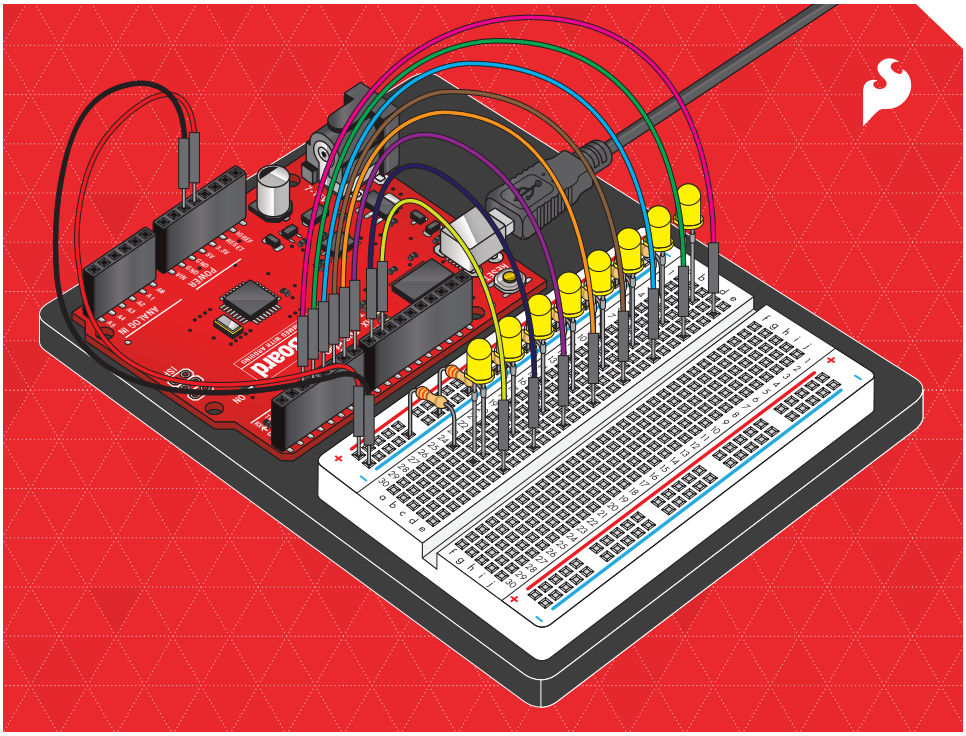


Guía SIK

Tu guía al “Sparkfun Inventor’s Kit” para Educadores

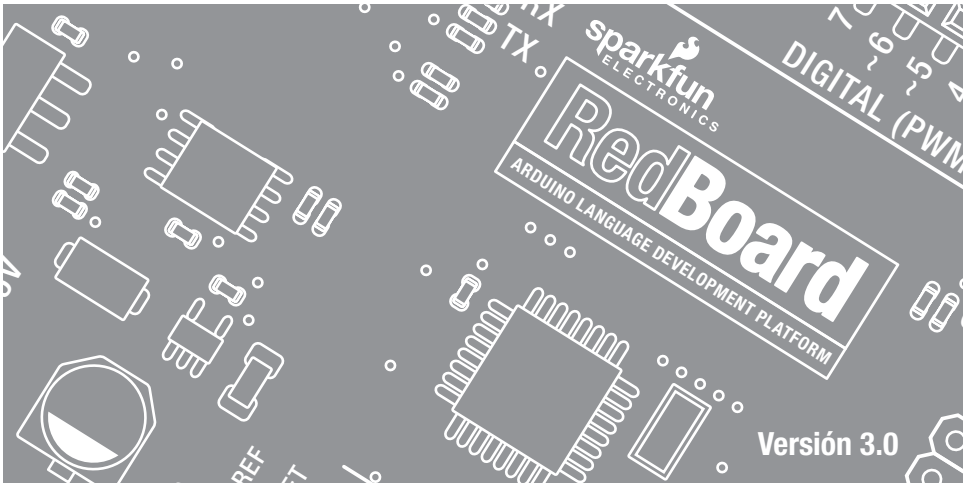


Tabla de contenidos

Bienvenido a la Guía de Inventores de SparkFun

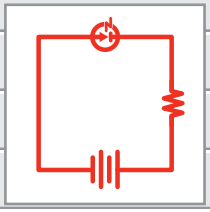
La Guía de Inventores de SparkFun es tu mapa para navegar en las aguas de la electrónica embebida para principiantes. Este folleto contiene toda la información que necesitarás para explorar los 15 circuitos del "SparkFun Inventor's Kit" para Educadores. Este manual se enfoca principalmente en una filosofía - que cualquiera puede (y debería) jugar con la electrónica. Cuando hayas acabado con esta guía, tendrás el conocimiento para poder empezar con tus propios proyectos y experimentos. Pero basta de hablar - ¡es hora de inventar!

www.sparkfun.com

Traducción al Español gracias a CRCibernetica.com y Alejandro Morales en Costa Rica

Sección 1: *Iniciando*

¿Qué es la plataforma RedBoard?	1
Descarga el Software de Arduino (IDE)	3
Instala controladores	4
Identifica tu Hardware	7
Descarga el “Código Guía del SIK”	8



Sección 2: *Iniciando con Circuitos*

El Mundo Funciona con Circuitos	9
Inventario de Partes	11
RedBoard	13
Protoboard	15
Circuito #1 - Tu Primer Circuito: LED Parpadeante	17
Circuito #2 - Potenciómetro	24
Circuito #3 - LED RGB	28
Circuito #4 - Múltiples LEDs	32
Circuito #5 - Botones presionables	36
Circuito #6 - Fotorresistencia	40
Circuito #7 - Sensor de Temperatura	44
Circuito #8 - Solo un Servo	48
Circuito #9 - Sensor Flexible	52
Circuito #10 - Potenciómetro Suave	56
Circuito #11 - Bocina	60
Circuito #12 - Rotando un Motor	64
Circuito #13 - Relé	68
Circuito #14 - Registro de Desplazamiento	72
Circuito #15 - LCD	76

¿Qué es la plataforma RedBoard?

La Revolución Hazlo Tú Mismo (Do It Yourself)

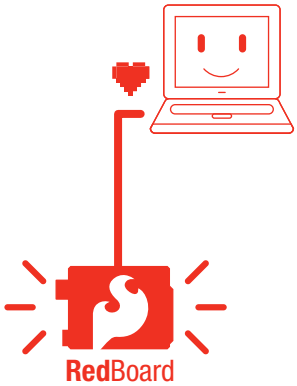
Vivimos en un tiempo único, en el que tenemos acceso a recursos que nos permiten crear nuestras propias soluciones e inventos. La revolución Hazlo tú Mismo está compuesta por inventores, "carpinteros electrónicos" y personas comunes que prefieren manufacturar sus propios proyectos antes que dejar que alguien lo haga por ellos.

www.sparkfun.com

Una Computadora para el Mundo Físico

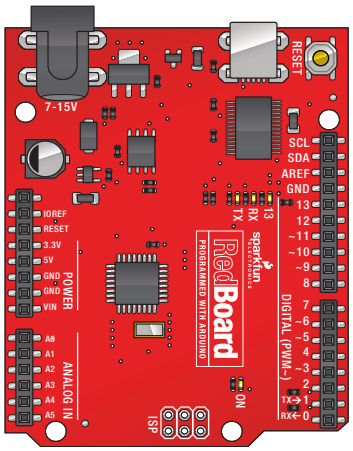
La RedBoard que tienes en tus manos (o en el escritorio) es tu plataforma de desarrollo. Por sí misma, la RedBoard es en esencia una pequeña computadora portátil. Es capáz de tomar entradas (tales como la señal de un botón al ser presionado o una lectura de un sensor de luz) e interpretarla esta información para controlar varias salidas (como el parpadeo de una luz LED o un motor eléctrico).

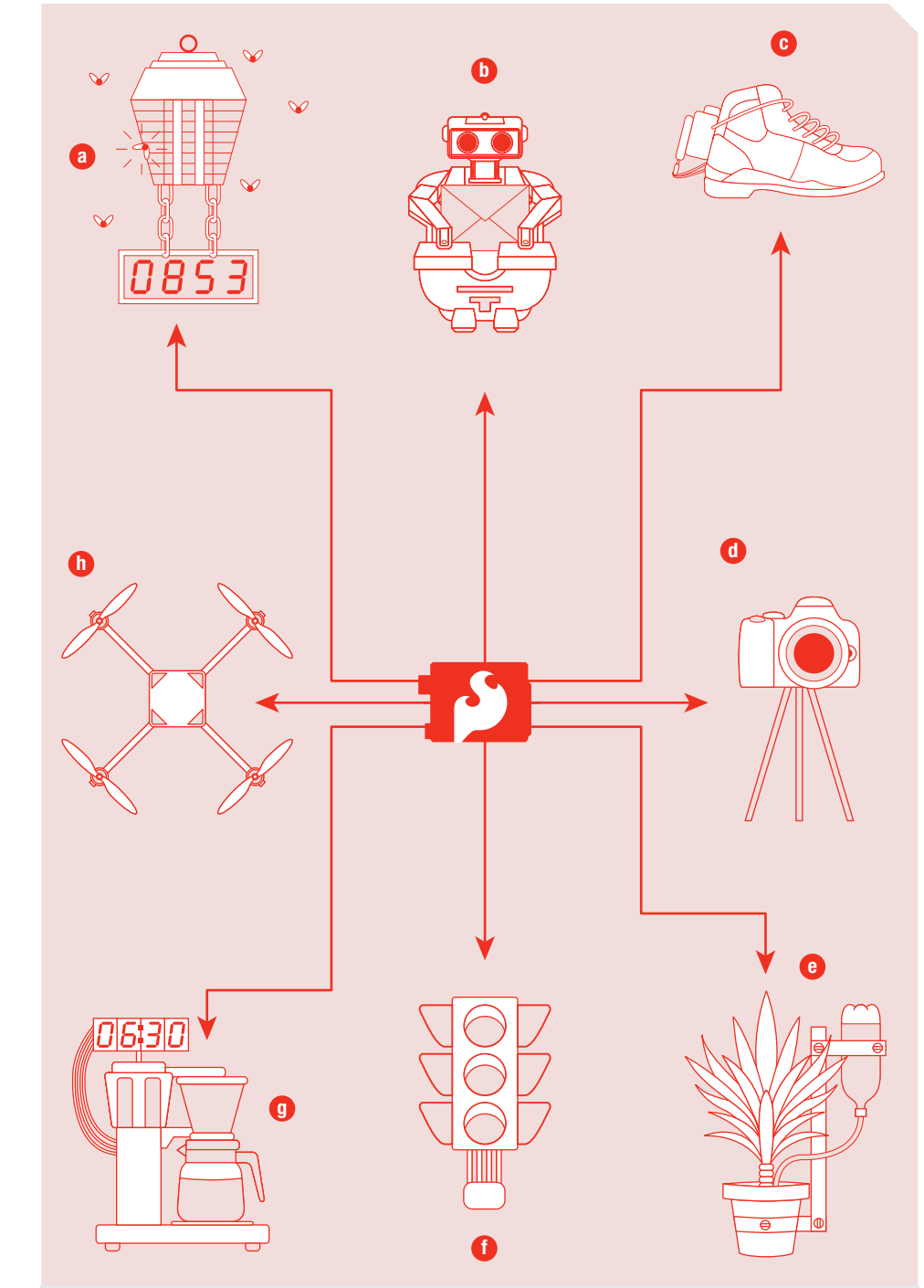
Aquí es donde nace el término "Computadora Física" - esta tarjeta es capaz de tomar el mundo de la electrónica y relacionarlo con el mundo físico en una forma real y tangible. Confía en nosotros - muy pronto esto tendrá más sentido.



// SparkFun RedBoard

La RedBoard de Sparkfun es una dentro entre multitudes de tarjetas que se desarrollan basadas en el ATmega328. Posee 14 pines de entrada/salida digital (de los cuales 6 pueden ser salidas PWM), 6 entradas analógicas, un oscilador de cristal de 16MHz, conexión USB, un Conector de alimentación, una entrada ISP, y un botón de reinicio. No te preocupes, aprenderás acerca de todos ellos más adelante.





a Contador para Lámpara Mata Insectos

b Notificador de correo para Juguetes Viejos

c Zapatillas Power-Lacing

d Operador de tiempo de espera para Cámaras

e Regador de plantas Automático

f Semáforo Reprogramable

g Coffee Maker Automático

h Quad-cóptero

Descarga el Arduino IDE (Ambiente de Desarrollo Integrado, por sus siglas en inglés)

Accede a internet

Para poder poner a funcionar tu RedBoard, es necesario que descargues primero la versión más actualizada del software de Arduino desde www.arduino.cc (¡es gratis!). Este software, conocido como Arduino IDE, te permitirá programar la tarjeta para que haga exactamente lo que tú quieres. Es similar a un procesador de palabras pero para escribir programas. Con una computadora con acceso a internet, abre tu navegador favorito y escribe el siguiente enlace en la barra de direcciones:

arduino.cc/en/main/software

1

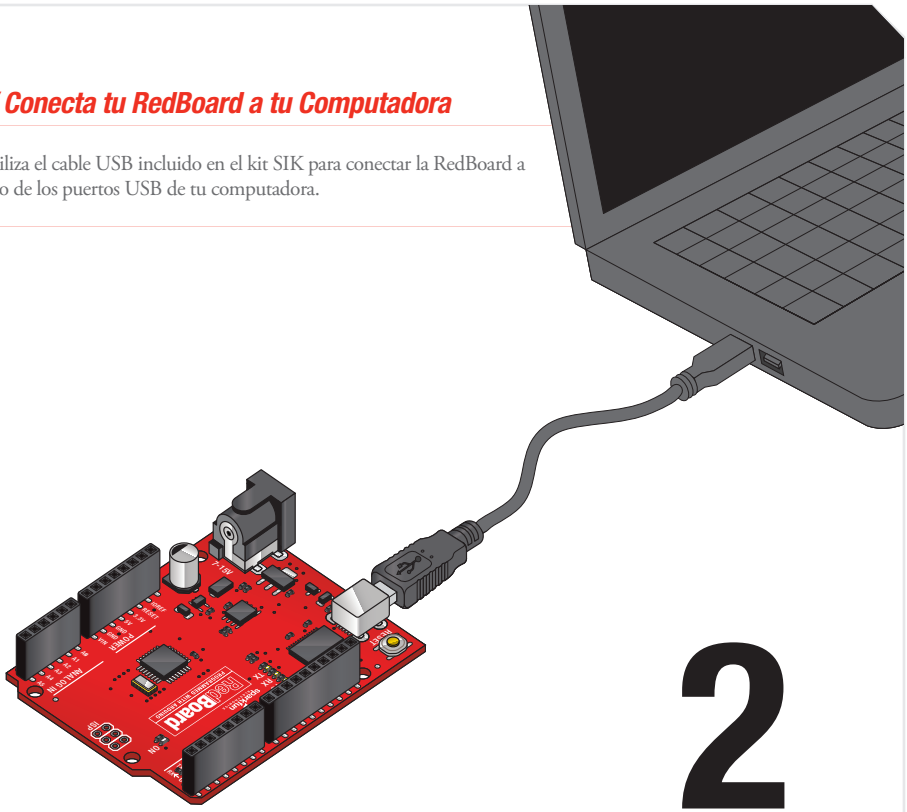
Descargar — Clic en tu sistema operativo apropiado, junto al signo de "+"

- + Windows
- + Mac OS X
- + Linux: 32 bit, 64 bit
- + Fuente

Elige el paquete de instalación apropiado para el Sistema Operativo de tu computadora.

// Conecta tu RedBoard a tu Computadora

Utiliza el cable USB incluido en el kit SIK para conectar la RedBoard a uno de los puertos USB de tu computadora.



2

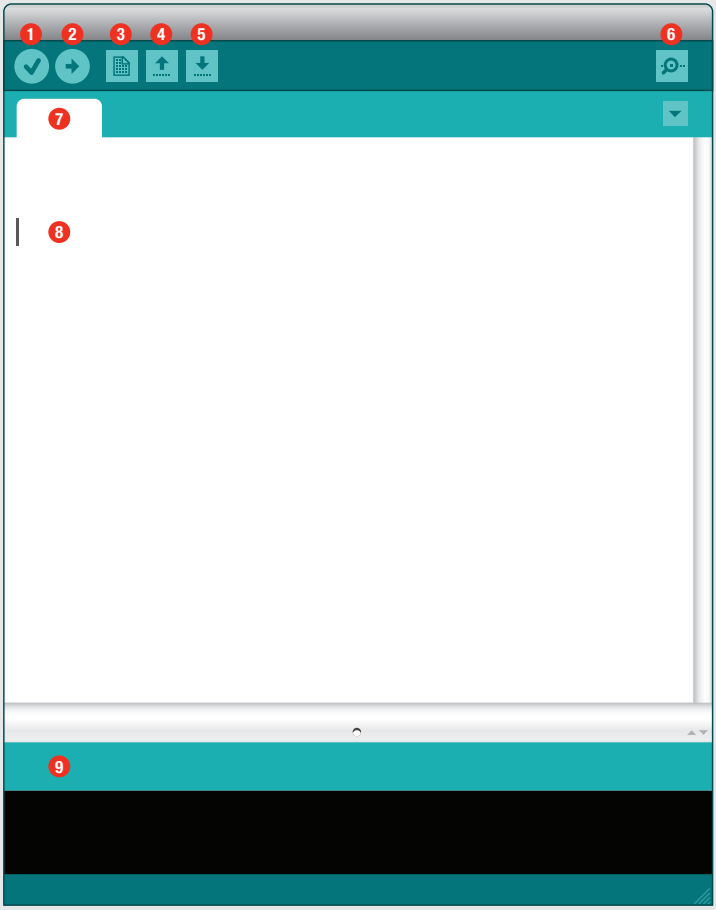
3

// Instala los drivers de Arduino

Dependiendo del sistema operativo de tu computadora, necesitarás seguir algunas instrucciones específicas. Por favor ingresa a **https://learn.sparkfun.com/tutorials/how-to-install-ftdi-drivers** para obtener instrucciones específicas acerca de cómo instalar los drivers FTDI en tu RedBoard.

// Abre el IDE de Arduino:

Abre el software del IDE de Arduino en tu computadora. Toca un poco la interfaz para que la vayas conociendo. No vamos a codificar nada en este momento, esto es solo una introducción. Debes realizar estos pasos para que ti IDE identifique tu RedBoard.



GUI (Interfaz Gráfica de Usuario)

1. **Verificar:** Compila y aprueba tu código. Encuentra errores de sintaxis (tales como paréntesis o signos de punto y coma faltantes). // Ver el siguiente diagrama
2. **Cargar:** Envía tu código a la RedBoard. Cuando le das clic, debes ser capaz de ver las luces en tu tarjeta parpadear rápidamente. // Ver el siguiente diagrama
3. **Nuevo:** Estos botones abren otra nueva pestaña de código.
4. **Abrir:** Este botón permite abrir un diseño existente. // Ver el siguiente diagrama
5. **Guardar:** Guarda el diseño actual.
6. **Monitor Serial:** Esto abre una ventana que muestra cualquier información serial que tu RedBoard esté transmitiendo. Es muy usado para tareas de depuración.
7. **Nombre del diseño:** Muestra el nombre del diseño en el que estás trabajando actualmente.
8. **Área de Código:** Esta es el área en donde escribes el código para tu diseño.
9. **Área de Mensajes:** Aquí es donde el IDE te dice si hay algún error en tu código.

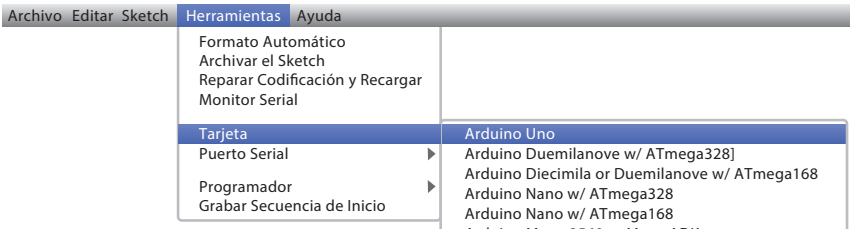
// Los tres comandos más importantes para esta guía se muestran a continuación:

Abrir

Verificar

Cargar

4 // Elige tu tarjeta: Arduino Uno

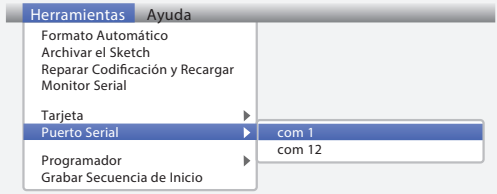


Nota:

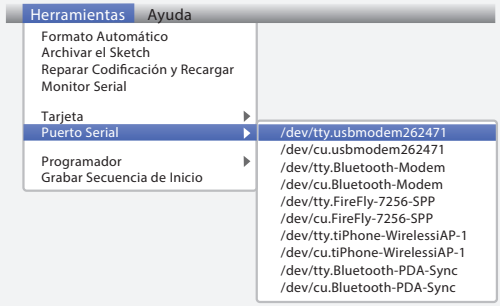
Tu RedBoard de SparkFun y el Arduino UNO son intercambiables pero no podrás encontrar la RedBoard listada dentro del Software de Arduino. En vez de esta selecciona “Arduino UNO”.

// Selecciona tu Dispositivo Serial

Selecciona el dispositivo serial de la RedBoard desde el menú de Herramientas | Puerto Serie. Probablemente sea **com3 o mayor** (COM1 y COM2 generalmente se reservan para puertos seriales de hardware). Para averiguarlo, puedes desconectar tu RedBoard y abrir de nuevo el menú; la entrada que desaparezca debe ser la del RedBoard. Vuelve a conectar la tarjeta y elige ese puerto serie.



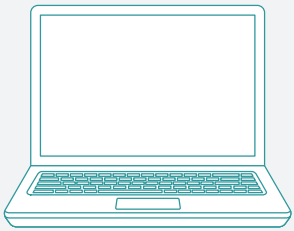
Selecciona el dispositivo serie del RedBoard desde el menú de Herramientas > Puerto Serial. En una Mac, esto debería ser algo que contenga **/dev/tty.usbmodem** o **/dev/tty.usbserial**.



http://www.arduino.cc/playground/Learning/Linux

Descarga código de Arduino (Utilizable con los circuitos de esta guía)

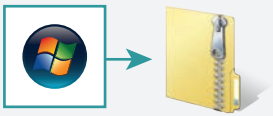
5



Ingresa al siguiente enlace para descargar el código:

sparkfun.com/sikcode

// Copia el archivo "SIK Guide Code" dentro de la biblioteca "Ejemplos" en la carpeta de Arduino



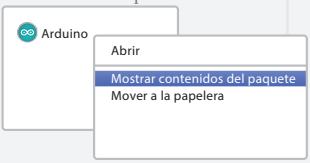
Descomprime el archivo "SIK Guide Code". Debería estar localizado en la carpeta de "Descargas" de tu navegador. Da clic derecho a la carpeta comprimida y elige la opción de "extraer".

Inicio → Programas → Arduino → Ejemplos

Copia la carpeta "SIK Guide Code" en la carpeta de Arduino llamada "Ejemplos".

Descomprime el archivo "SIK Guide Code". Debería estar localizado en la carpeta de "Descargas" de tu navegador. Da clic derecho a la carpeta comprimida y elige la opción de "extraer".

Busca "Arduino" en la carpeta de aplicaciones. Clic derecho (ctrl + clic) en "Arduino". Selecciona "Mostrar Contenidos del Paquete".



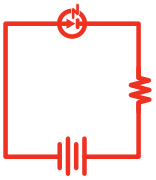
Contenidos
↓
Recursos
↓
Java
↓
Ejemplos

Copia la carpeta "SIK Guide Code" en la carpeta de Arduino llamada "Ejemplos".

http://www.arduino.cc/playground/Learning/Linux

¿QUÉ ES LO QUE SIGUE? Continúa leyendo para aprender más acerca de empezar a usar circuitos. ¡Luego puedes iniciar con tu primer circuito en la página 17!

Iniciando con Circuitos



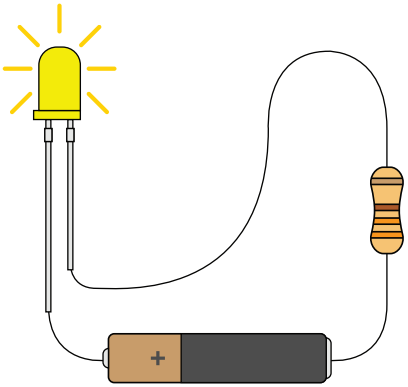
¿Qué es un circuito eléctrico?

Un circuito es básicamente un ciclo eléctrico con un punto de inicio y un punto final – con cualquier número de componentes entre dichos puntos. Los circuitos pueden incluir resistencias, diodos, inductores, sensores de todas formas y tamaños, motores, y cualquier otro tipo entre cientos de miles de componentes.

Los circuitos se dividen generalmente en tres categorías – circuitos analógicos, circuitos digitales o circuitos de señales mixtas. En esta guía explorarás los tres tipos de circuitos mencionados anteriormente.

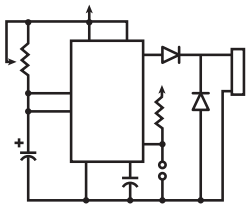
El Mundo Funciona con Circuitos:

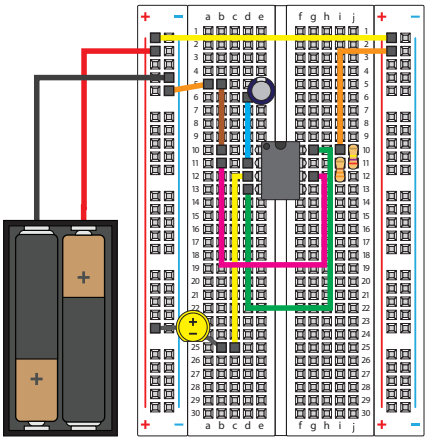
Donde sea que mires encontrarás circuitos. El teléfono celular que se encuentra en tu bolsillo, la computadora que controla el sistema de emisiones de tu automóvil, tu consola de videojuegos – todas estas cosas están completamente llenas de circuitos. En esta guía experimentarás con algunos circuitos y podrás aprender acerca de la esencia del mundo de la electrónica embebida.

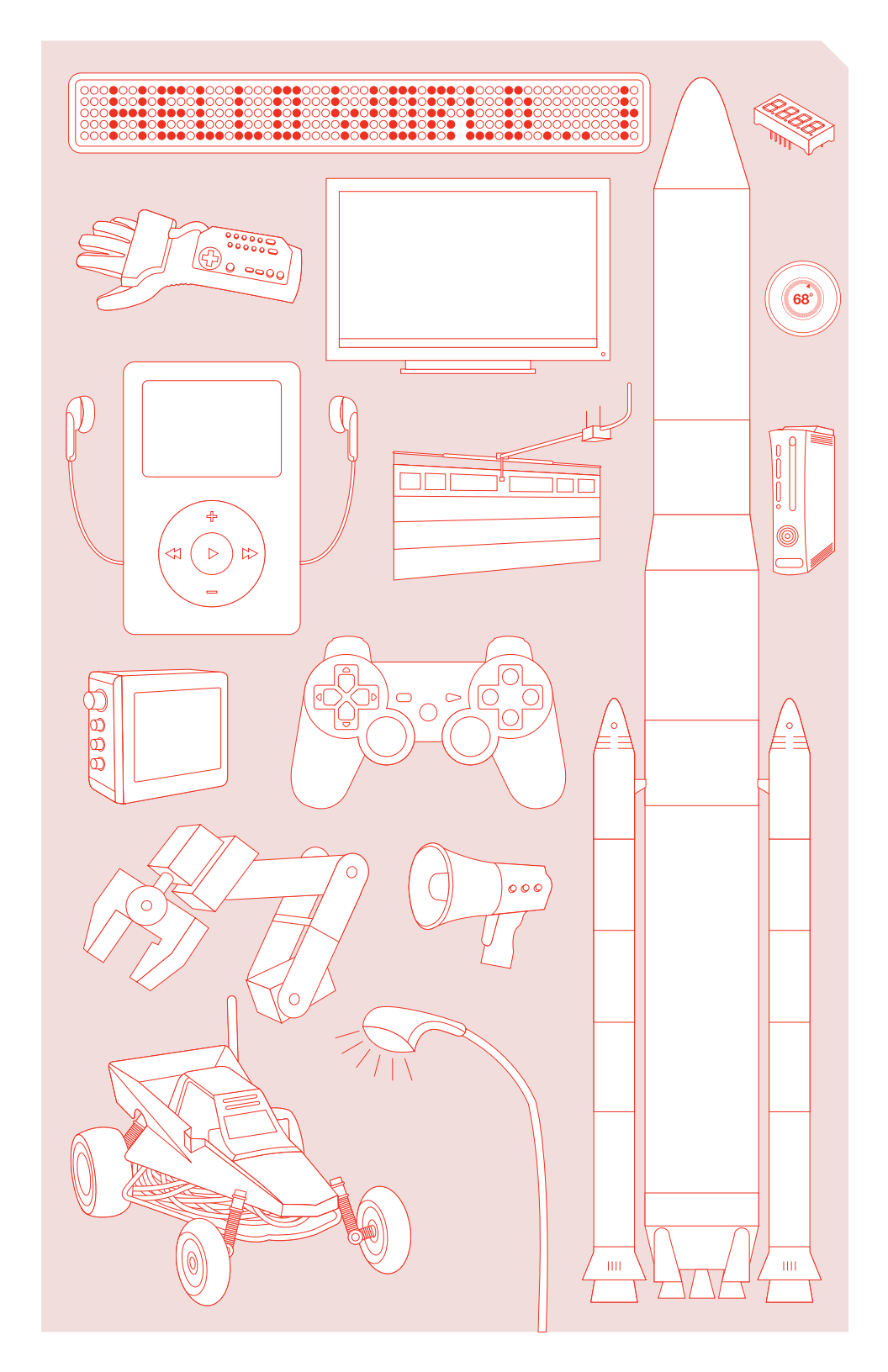


// Circuitos simples y complejos

En esta guía explorarás inicialmente circuitos simples - ¡pero eso no quiere decir que no puedas hacer cosas asombrosas con herramientas sencillas! Cuando hayas terminado con el SIK, tu conocimiento de circuitos te permitirá explorar proyectos asombrosos y desatar el potencial de tu imaginación.

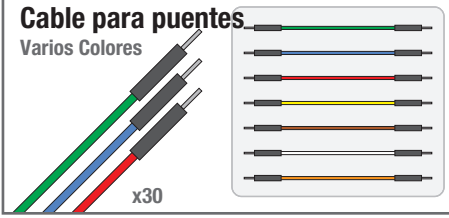





68°

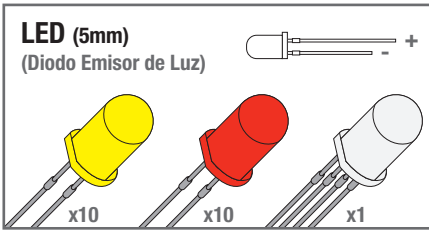
Inventario de Partes

Cable para puentes
Varios Colores



x30

LED (5mm)
(Diodo Emisor de Luz)

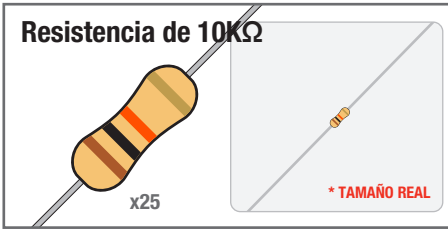


x10 x10 x1

Resistencia de 330Ω

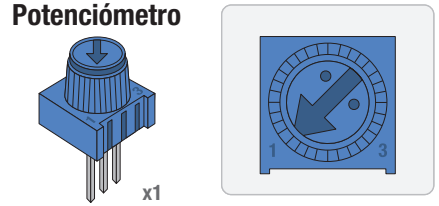
x25

Resistencia de 10KΩ



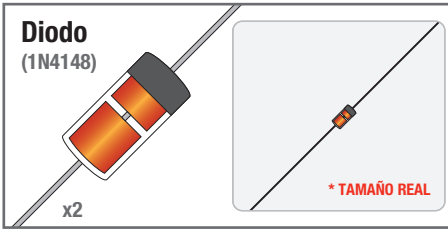
x25

Potenciómetro



x1

Diodo
(1N4148)

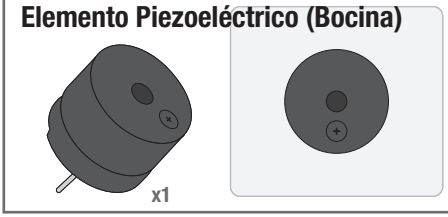


x2

Fotorresistencia

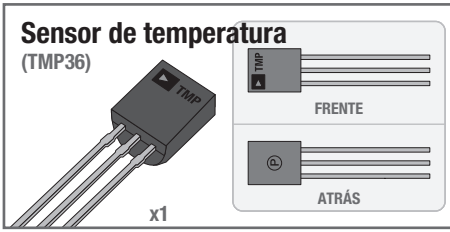
x1

Elemento Piezoeléctrico (Bocina)



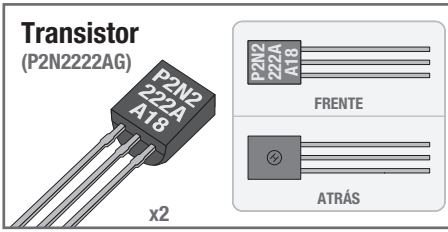
x1

Sensor de temperatura
(TMP36)



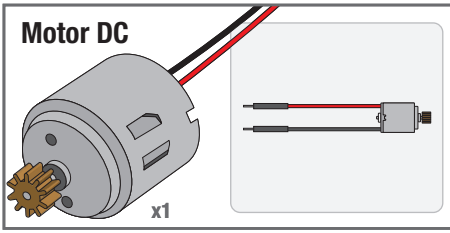
x1

Transistor
(P2N2222AG)



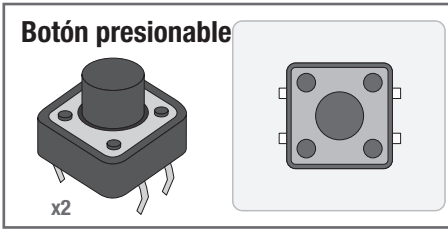
x2

Motor DC



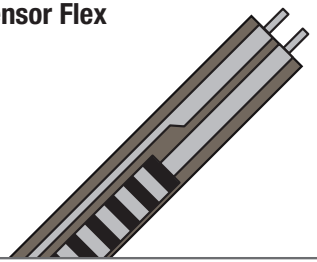
x1

Botón presionable



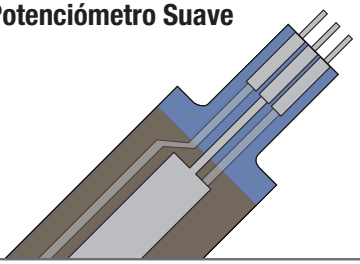
x2

Sensor Flex



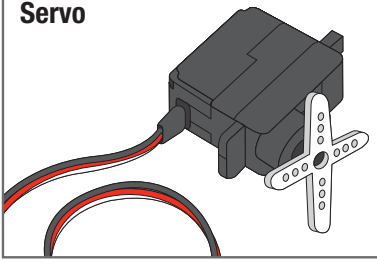
x1

Potenciómetro Suave



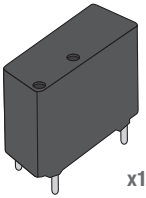
x1

Servo



x1

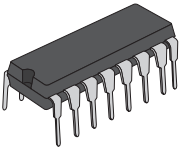
Relé



x1

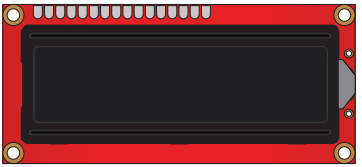
Circuito Integrado

(CI)



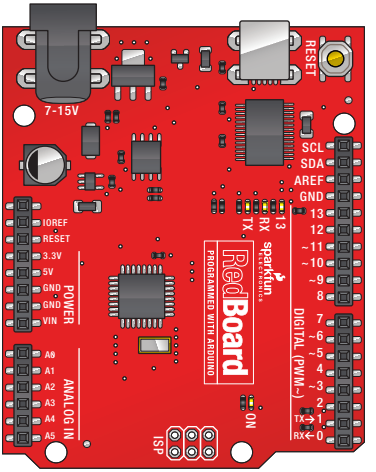
x1

LCD



x1

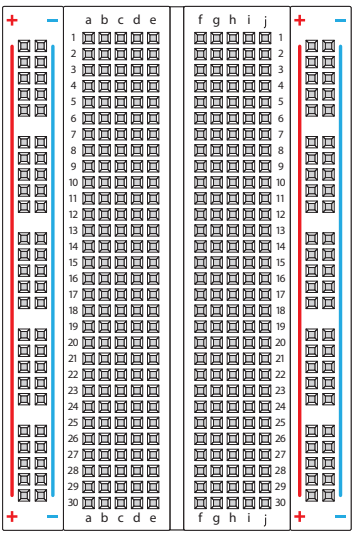
SparkFun RedBoard



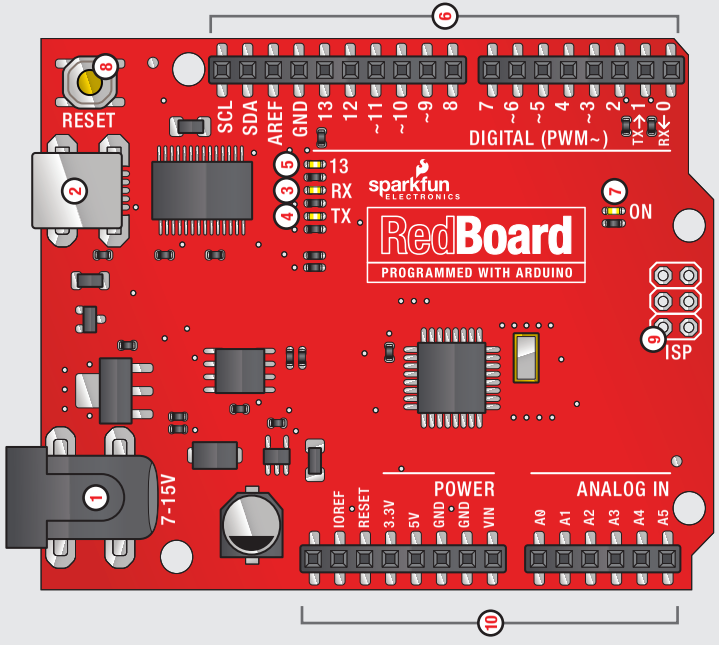
x1

Protoboard

Estándar sin soldadura (El color puede variar)



x1

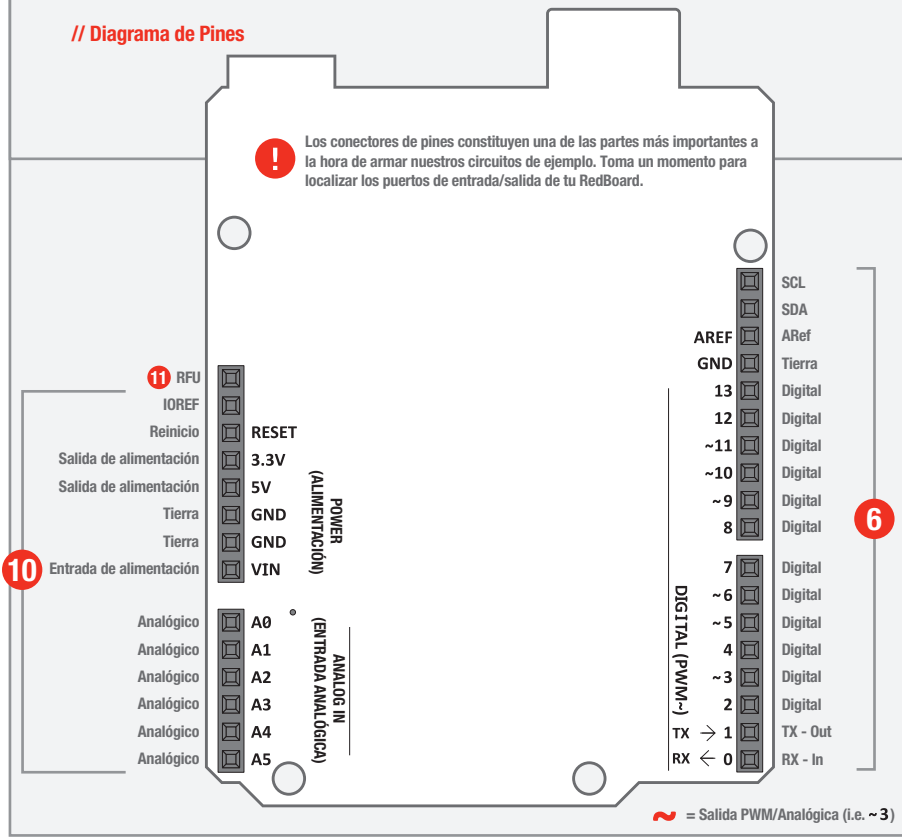

6
8
RESET
SCL
SDA
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
2
5
13
4
3
RX
TX
sparkfun
ELECTRONICS
RedBoard
PROGRAMMED WITH ARDUINO
7
ON
9
ISP
1
7-15V
IOREF
RESET
3.3V
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
10

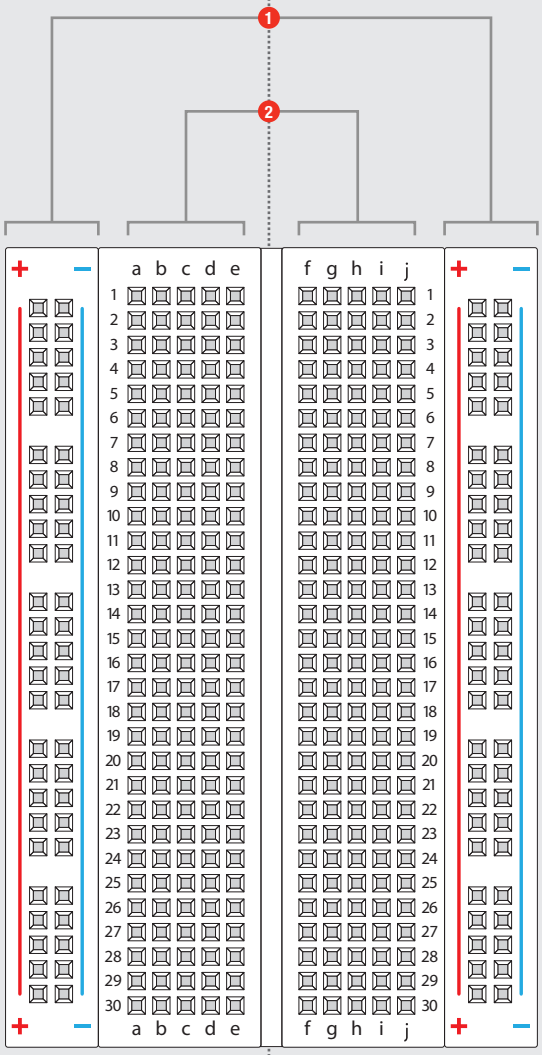
SparkFun RedBoard

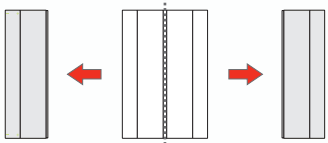
1. **Alimentación** (Conector "Barrel Jack") - puede ser usado tanto con un adaptador de 9V o 12V conectado a un enchufe como con una batería.
2. **Alimentación** (Puerto USB) - Provee alimentación y comunica tu tarjeta con tu computadora al conectarla por vía USB.
3. **LED** (RX: Recibiendo) - Muestra cuando el chip FTDI está recibiendo bits de información desde el microcontrolador. Esto sucede cuando el microcontrolador está enviando bits de información de vuelta a la computadora.
4. **LED** (TX: Transmitiendo) - Muestra cuando el chip FTDI está transmitiendo bits de información hacia el microcontrolador. Esto sucede cuando el microcontrolador está recibiendo esta información desde a la computadora.
5. **LED** (Pin 13: Indicador de problemas) - Este LED es incorporado a tu diseño para mostrar si tu programa está funcionando correctamente.
6. **Pines** (ARef, Ground (GND), Digital, Rx, Tx) - Estos pines pueden ser utilizados como entradas, salidas, alimentación y tierra. // Ver el siguiente diagrama
7. **LED** (Indica si la RedBoard está encendida) - Un simple LED indicador de encendido/apagado.
8. **Botón de Reinicio (Reset)** - Esta es una forma de reiniciar manualmente tu RedBoard, lo que hace que tu código vuelva a empezar desde cero.
9. **Pines ICSP** (Cargando código sin un cargador de inicio) - Se usa para "Programación Serial dentro del Circuito" ("In-Circuit Serial Programming"), requerida si quieres evitar el cargador de inicio o Bootloader.
10. **Pines** (Analog In, Power In, Ground, Power Out, Reset) - Estos pines pueden ser utilizados como entradas, salidas, alimentación y tierra. // Ver el siguiente diagrama
11. **RFU** - Este pin está reservado para un uso futuro.

// Diagrama de Pines

! Los conectores de pines constituyen una de las partes más importantes a la hora de armar nuestros circuitos de ejemplo. Toma un momento para localizar los puertos de entrada/salida de tu RedBoard.







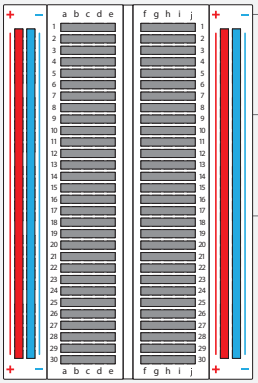
Esta línea divide la tarjeta por la mitad, restringiendo el flujo eléctrico hacia una de las dos mitades.

Protoboard

1 **Conexión vertical** ((+ Alimentación y - Tierra) - Buses de Alimentación // er el diagrama abajo

2 **Conexión Horizontal** (a-e & f-j) // Ver el diagrama abajo

¿Cómo está conectado todo?



+ Alimentación:

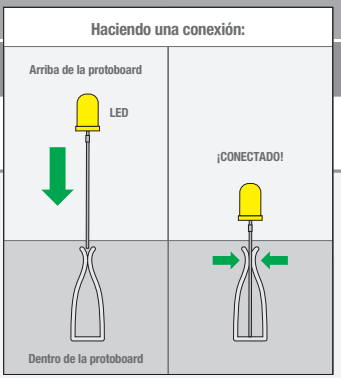
Cada signo de + indica que hay alimentación en cualquier lugar de la columna vertical.

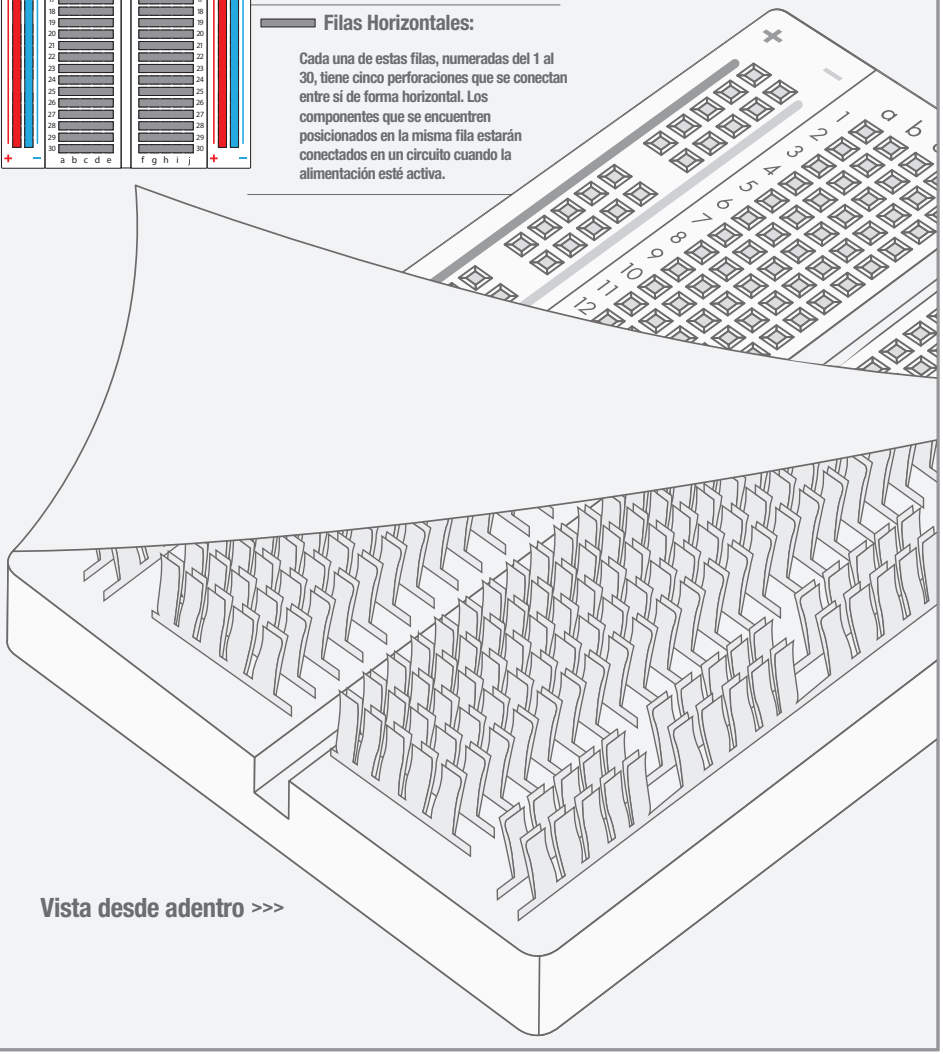
− Tierra:

Cada signo de – indica que hay tierra en cualquier lugar de la columna vertical.

Filas Horizontales:

Cada una de estas filas, numeradas del 1 al 30, tiene cinco perforaciones que se conectan entre sí de forma horizontal. Los componentes que se encuentren posicionados en la misma fila estarán conectados en un circuito cuando la alimentación esté activa.





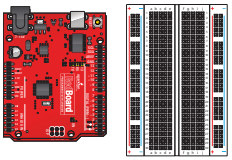
Vista desde adentro >>>

CIRCUITO #1 - *Tu primer Circuito*

Cómo funciona:

1 ENSAMBLA **2 ESCRIBE** **3 CARGA**

+ Asegúrate de que el texto tanto en la RedBoard como en la protoboard esté mirando hacia arriba para que puedas leerlo fácilmente.



+ Conectando el cable USB.

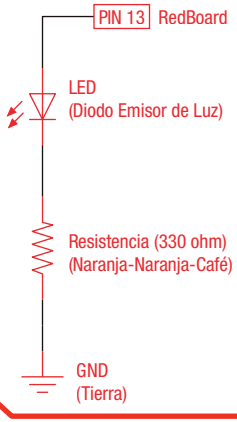
+ Atornilla la RedBoard en su lugar.

+ Retira la protección del pegamento en la parte posterior de la protoboard y pégala en su lugar.

Tu RedBoard trabaja con 5V. Esta es la alimentación que será suministrada por tu computadora vía USB y será la fuente de poder de todos los componentes que uses en tus circuitos. ¡Al conectar tu RedBoard a tu computadora, le estás suministrando el voltaje justo que necesita para funcionar! 5V no pueden hacerte daño, así que no tengas miedo de tocar cualquier cosa en tu circuito. Además puedes alimentar la RedBoard con el adaptador de barril. El regulador de voltaje incluido en la tarjeta puede manejar cualquier voltaje desde 7 a 15V en corriente directa.

1

LED Parpadeante



+ Este es un esquemático de tu circuito.

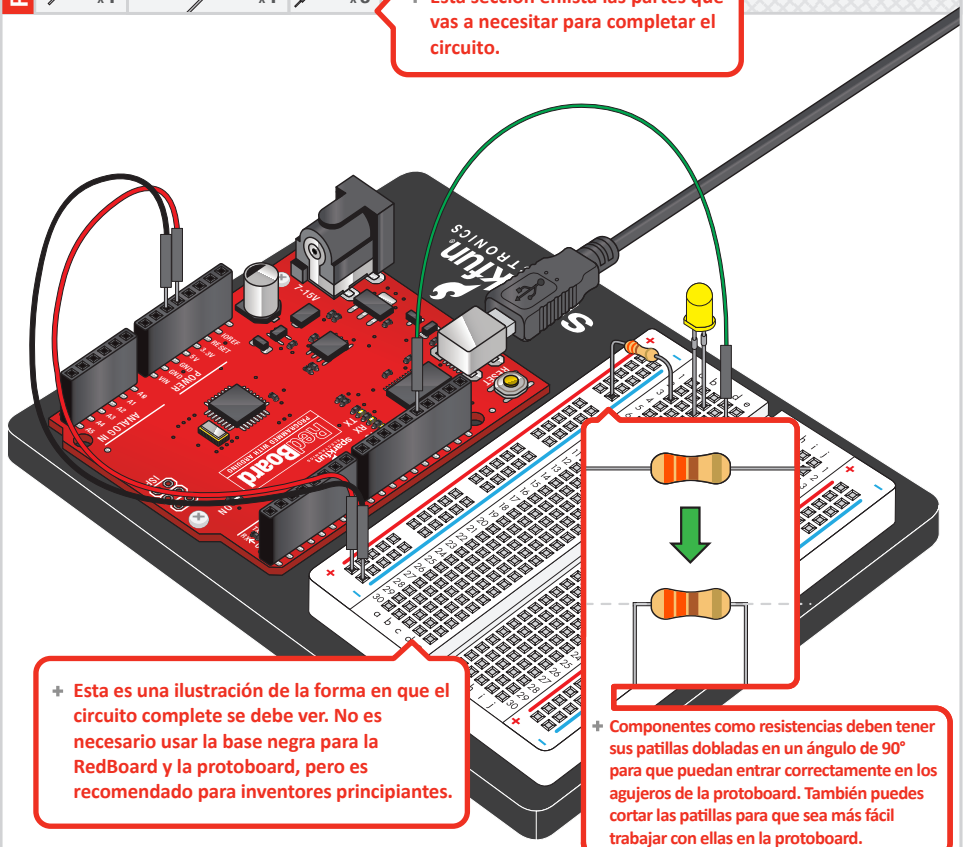
Los LEDs (Diodos Emisores de Luz) son luces pequeñas y poderosas que son utilizadas en muchas aplicaciones diferentes. Para empezar con el SIK, vamos a trabajar en hacer parpadear un LED. Correcto - es tan simple como encender y apagar una luz. Puede que no se vea como una gran cosa, pero estableciendo esta importante base tendrás fundamentos sólidos mientras trabajamos en ruta a experimentos más complejos.

+ Cada circuito empieza con una breve descripción de lo que estás a punto de armar y el resultado esperado.

PARTES:

LED	Resistencia de 330Ω	Cable
x1	x1	x3

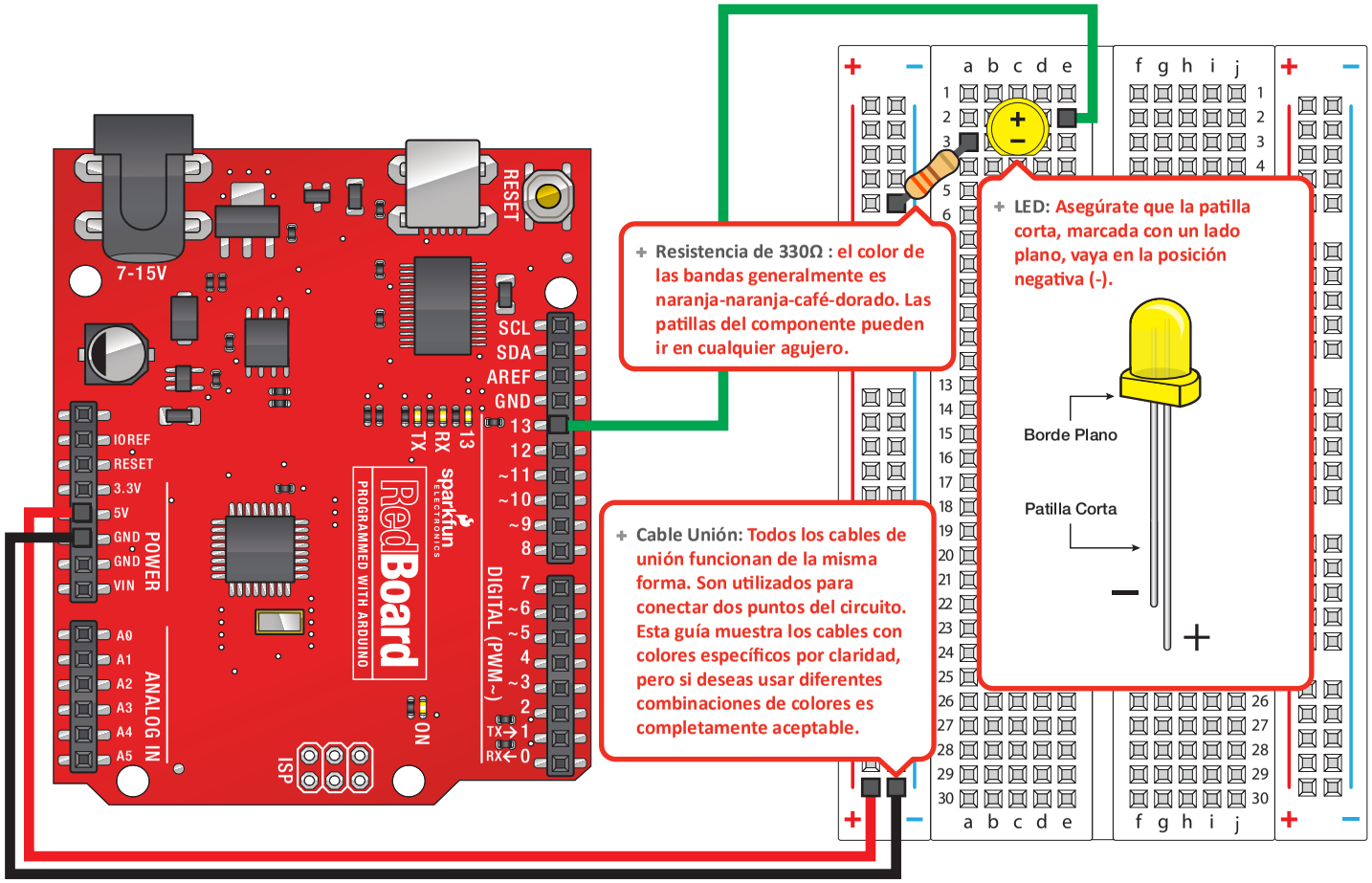
+ Esta sección enlista las partes que vas a necesitar para completar el circuito.

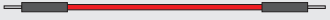
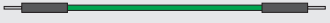


+ Esta es una ilustración de la forma en que el circuito complete se debe ver. No es necesario usar la base negra para la RedBoard y la protoboard, pero es recomendado para inventores principiantes.

+ Componentes como resistencias deben tener sus patillas dobladas en un ángulo de 90° para que puedan entrar correctamente en los agujeros de la protoboard. También puedes cortar las patillas para que sea más fácil trabajar con ellas en la protoboard.

Circuito 1: LED Parpadeante



Componente:	Imagen de Referencia:		
LED (5mm)			c2 - c3 (+ -)
Resistencia de 330Ω			a3 - -
Cable Conector		GND	-
Cable Conector		5V	+
Cable Conector		Pin 13	e2

+ Componentes como los LEDs son insertados en los agujeros c2(patilla larga) y c3(patilla corta) de la protoboard. Los pasos resaltados con el triángulo amarillo de precaución representan componentes polarizados. Presta atención especial a las marcas de los componentes pues indican cómo colocarlos en la protoboard.

+ Las resistencias son colocadas únicamente en los agujeros de la protoboard. El símbolo "-" representa cualquier agujero en la columna vertical del bus de Alimentación.

+ La tierra "GND" en la RedBoard debería estar conectada a la fila marcada con el "-" de la protoboard.

+ "5V" en la RedBoard se conecta con la fila marcada con el "+" en la protoboard.

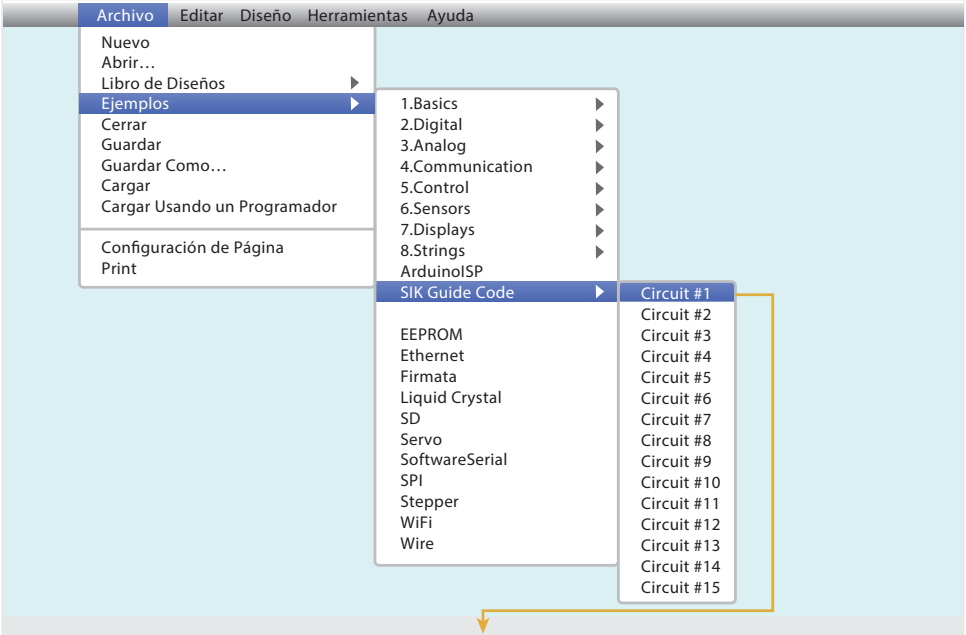
+ "PIN 13" en la ReadBoard se conecta al agujero "e2" de la protoboard.

+ **RedBoard:** El fondo rojo representa una conexión con uno de los pines principales de la RedBoard.

+ **Protoboard:** El fondo blanco representa una conexión con un agujero especificado por una coordenada letra-número como e2. Estas coordenadas son simples sugerencias que se alinean con la imagen gráfica.

Abre Tu Primer Diseño:

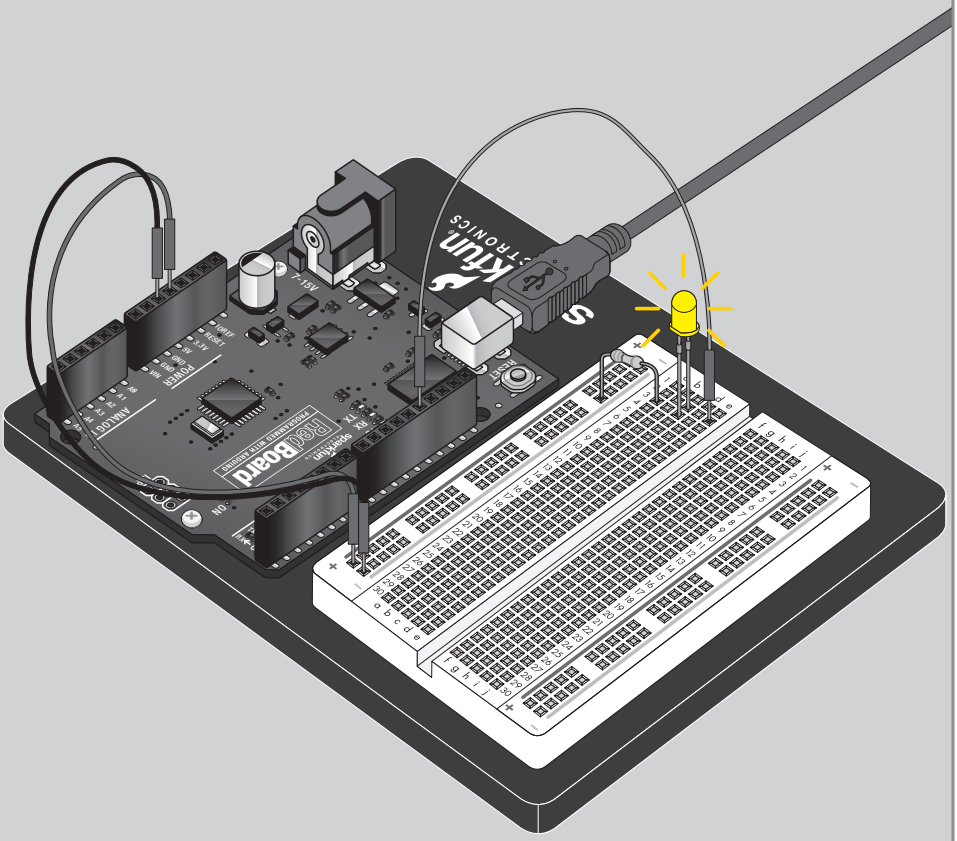
Abre el software Abre el software de Arduino IDE en tu computadora. Codificar en el lenguaje de programación de Arduino permitirá controlar tu circuito. Abre el código para el Circuito 1 accediendo al “Código Guía de SIK” que descargaste y colocaste en tu carpeta de “Ejemplos” previamente.



// Circuito #1

Verificar	Esto compila tu código. El IDE convierte el texto a instrucciones que la computadora pueda entender.
Cargar	Esto envía las instrucciones mediante el cable USB al chip computadora en la RedBoard. A continuación, la RedBoard empezará a correr tu código automáticamente.

// El resultado de un circuito completo con un código correcto luego de ser verificado y cargado.



1

Aquí es donde encontrará el código de Arduino para cada circuito.

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 1

Notas de Código:

+ Recuerda Verificar y Cargar tu código.

+ Empieza entendiendo cómo funciona el código de Arduino. Ver abajo.

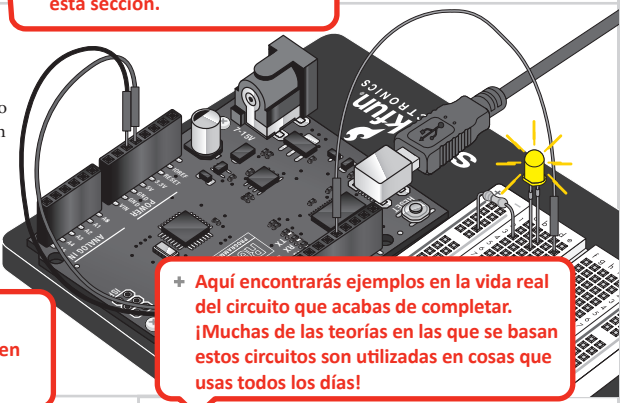
pinMode(13, OUTPUT); ⇨ Antes de que puedas usar alguno de los pines de la RedBoard, necesitas decirle a la RedBoard si es una ENTRADA o una SALIDA. Utilizamos una "función" propia del sistema llamada pinMode() para hacer esto.

digitalWrite(13, HIGH); ⇨ Cuando estás usando un pin como SALIDA, puedes ordenarle que esté en posición de ALTO voltaje (salida de 5 voltios), o en BAJO (salida de 0 voltios).

Lo que deberías ver:

+ Revisa si tu circuito está completo y funcionando en esta sección.

Deberías ver tu LED parpadear entre encendido y apagado. Si esto no funciona, asegúrate de que hayas ensamblado el circuito correctamente, verificado y cargado el código a tu tarjeta o puedes ver la sección de problemas comunes que se muestra abajo.



+ Aquí encontrarás ejemplos en la vida real del circuito que acabas de completar. ¡Muchas de las teorías en las que se basan estos circuitos son utilizadas en cosas que usas todos los días!

Esta es una sección dedicada a los problemas más comunes que se pueden dar mientras se ensambla el circuito.

Problemas Comunes:

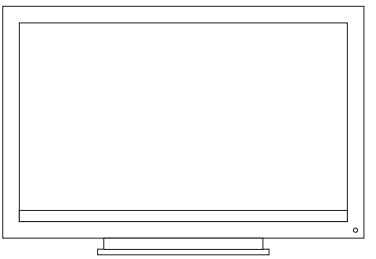
¿El LED no enciende?
Los LEDs trabajan en una sola dirección. Prueba quitarlo y rotarlo 180 grados (no hay de qué preocuparse, instalarlo al revés no provoca ningún daño permanente).

El programa no se Carga en la tarjeta
Algunas veces esto sucede, la causa más común se debe a una confusión con el puerto serial, puedes cambiar esto en Herramientas>Puerto Serial>

¿Aún no funciona?
Un circuito roto no es divertido, envíanos un correo electrónico y te responderemos tan pronto como sea posible:
techsupport@sparkfun.com

Aplicación en la vida real:

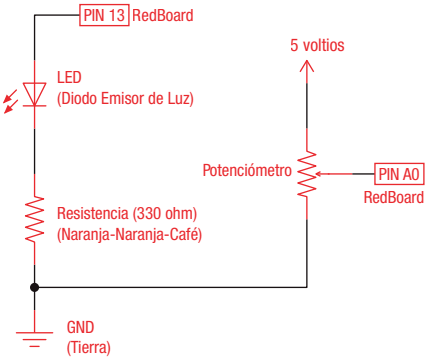
Casi todos los televisores modernos de pantalla plana y los monitores tienen luces LED indicadoras para mostrar si están encendidos o apagados.



CIRCUITO #2

2

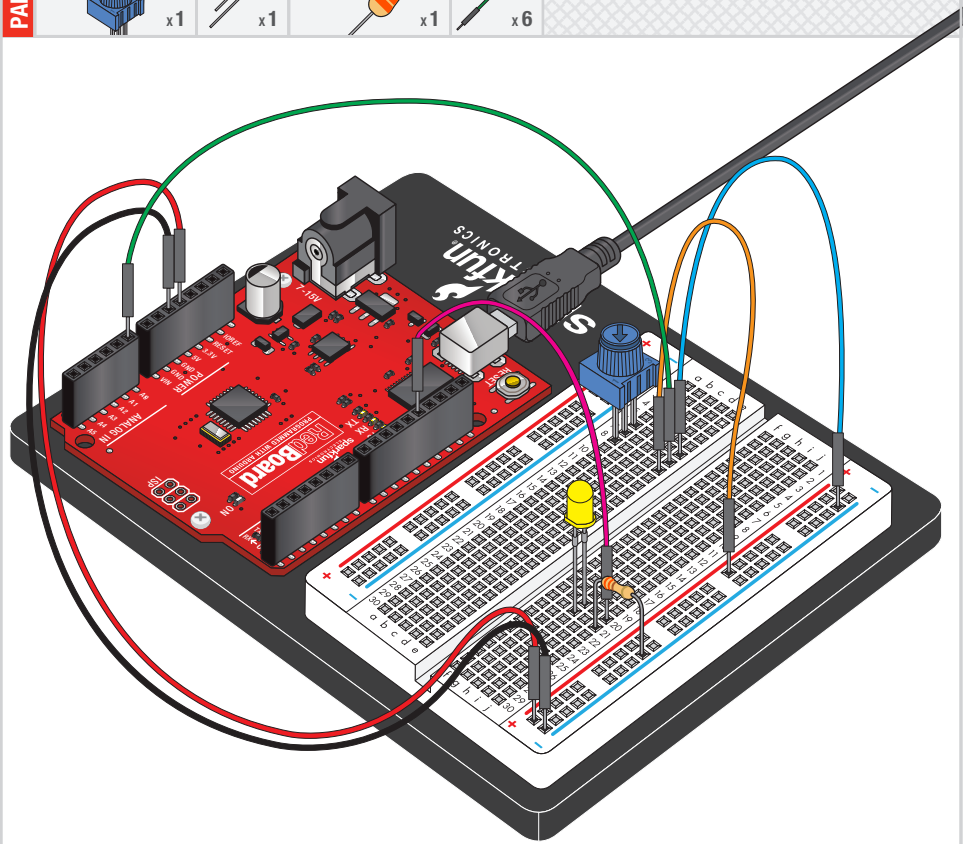
Potenciómetro



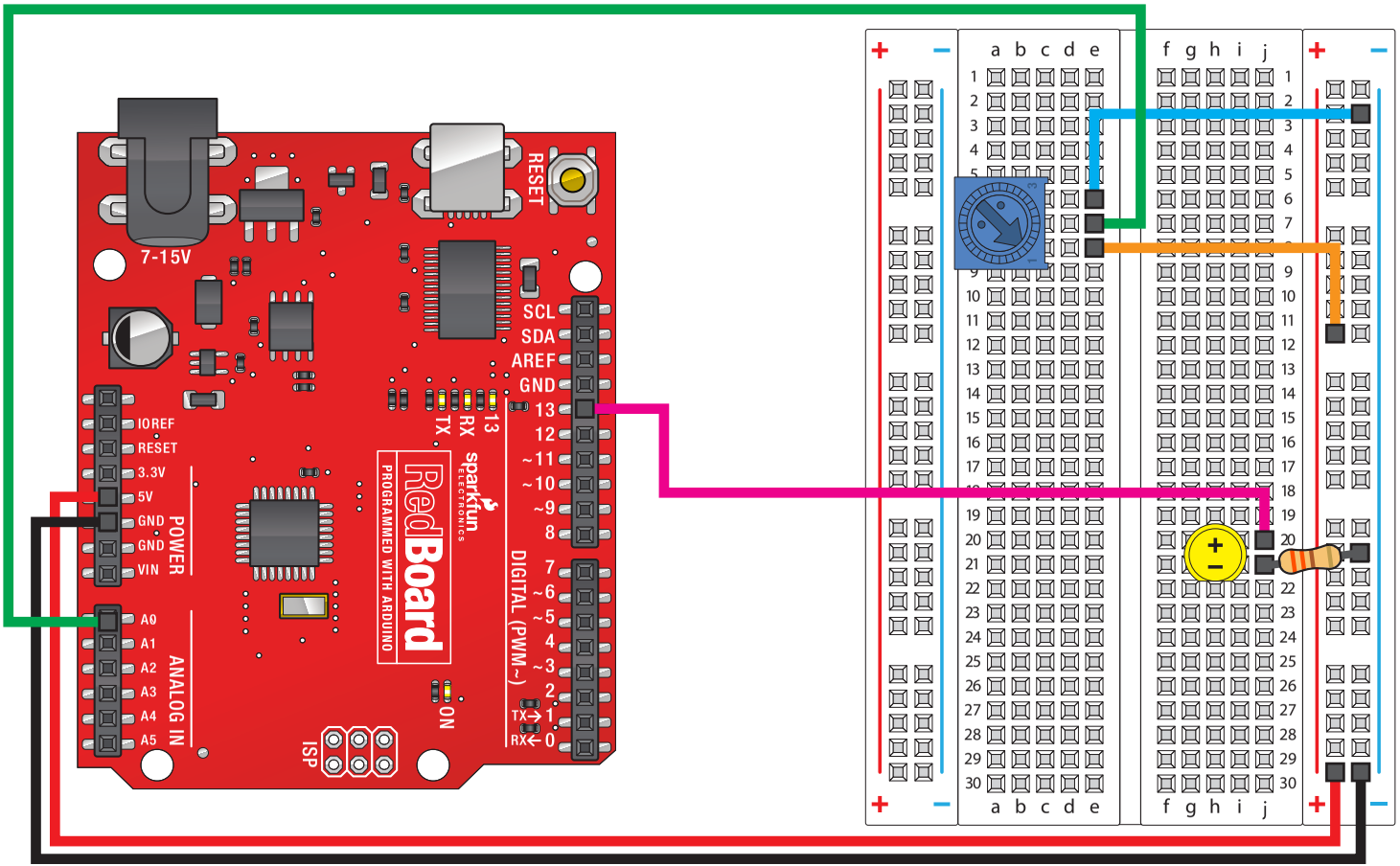
En este circuito trabajarás con un potenciómetro. Un potenciómetro es también conocido como una resistencia variable. Cuando está conectado con 5 voltios a través de sus dos pines exteriores, el pon del medio libera un voltaje entre 0 y 5V, dependiendo de la posición de la perilla en el potenciómetro. Un potenciómetro es una demostración perfecta de un circuito divisor de tensión con un voltaje variable. El voltaje está dividido proporcionalmente a la resistencia entre el pin del medio y el pin de tierra. En este circuito aprenderás cómo usar un potenciómetro para controlar la luminosidad de un LED.

PARTES:

Potenciómetro	LED	Resistencia de 330Ω	Cable
x1	x1	x1	x6



Circuito 2: Potenciómetro



Componente:	Imagen de Referencia:		
Potenciómetro			a6 a7 a8
LED (5mm) ⚠			h20 (+) h21 (-)
Resistencia de 330Ω			j21 -
Cable Conector			e6 -
Cable Conector		A0	e7
Cable Conector			e8 +
Cable Conector		Pin 13	j20
Cable Conector		5V	+
Cable Conector		GND	-

Digital versus Analógico:

Si miras con cuidado tu RedBoard podrás ver algunos pines etiquetados como "DIGITAL", y algunos etiquetados como "ANALOG". ¿Cuál es la diferencia?

Muchos de los dispositivos que vas a conectar, como los LEDs y los botones, solo tienen dos estados posibles: encendido y apagado, o como son conocidos en la RedBoard, "ALTO" (5 voltios) y "BAJO" (0 voltios). Los pines digitales en la RedBoard son excelentes para conectar estas señales desde y hacia el mundo real, e incluso pueden realizar trucos como una atenuación simulada (parpadeando entre encendido y apagado rápidamente), y comunicaciones seriales (transferir datos hacia otro dispositivo mediante patrones codificados de ALTOS y BAJOS).

DIGITAL

BAJO	o	ALTO
apagado		encendido
0 voltios		5 voltios

Sin embargo hay muchas cosas que no siempre están simplemente "encendidas" o "apagadas". Niveles de temperatura, perillas de control, etc. Todas tienen un rango continuo entre ALTO y BAJO. Para estas situaciones la RedBoard ofrece seis entradas analógicas que traducen un voltaje de entrada a un número en un rango de 0 (0 voltios) a 1023 (5 voltios). Los pines analógicos son perfectos para medir todos esos valores del "mundo real", y te permiten conectar tu RedBoard con todo tipo de dispositivos.

ANALÓGICO

0 voltios	a	5 voltios
0		1023

2

Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 2

Notas de Código:

int sensorValue;

Una "variable" es un valor guardado al que tú le has dado un nombre. Debes introducir, o "declarar" variables antes de usarlas; aquí estamos declarando una variable llamada sensorValue, de tipo "int" (integer o entero). ¡No olvides que los nombres de las variables son sensibles a las mayúsculas!

sensorValue = analogRead(sensorPin);

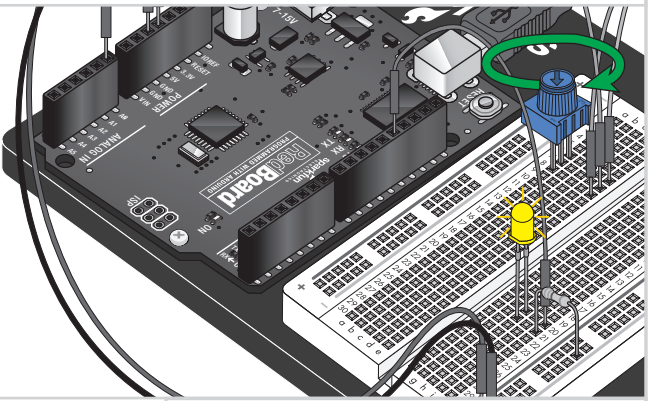
Utilizamos la función analogRead() para leer el valor en un pin analógico. analogRead() toma un parámetro, el pin analógico que quieres leer ("sensorPin"), y retorna un número ("sensorValue") entre 0 (0 voltios) y 1023 (5 voltios).

delay(sensorValue);

El Arduino es muy muy rápido, es capaz de correr miles de líneas de código cada segundo. Para hacerlo más lento, con el fin de que podemos ver lo que estamos haciendo, debemos insertar retardos dentro del código. La función delay() cuanta en milisegundos; hay 1000ms en un segundo.

Lo que deberías ver:

Deberías ver el LED parpadear más rápido o más lento de acuerdo con tu potenciómetro. Si esto no funciona, asegúrate de que hayas ensamblado el circuito correctamente, verificado y cargado el código a tu tarjeta o puedes ver la sección de problemas comunes que se muestra abajo.



Problemas comunes:

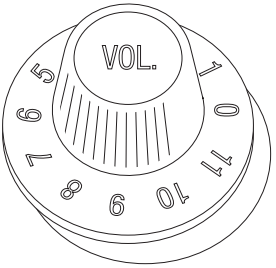
Funciona Esporádicamente
Es probable que esto se deba a una conexión inestable de los pines del potenciómetro. Esto puede ser solucionado presionando el potenciómetro hacia abajo.

No Funciona
Asegúrate de haber conectado la patilla controladora del potenciómetro al pin 0 digital en vez de al pin 0 analógico. (la fila de pines debajo de los pines de alimentación).

¿El LED no enciende?
Los LEDs trabajan en una sola dirección. Prueba quitarlo y rotarlo 180 grados (no hay de qué preocuparse, instalarlo al revés no provoca ningún daño permanente).

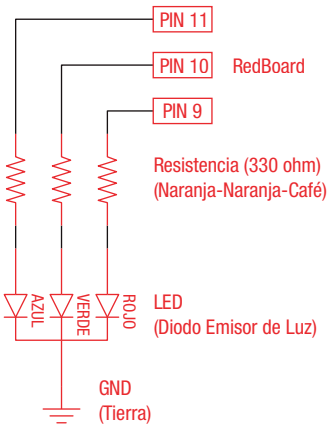
Aplicación en la vida real:

La mayoría de las perillas de volumen tradicionales emplean un potenciómetro.



CIRCUITO #3

3

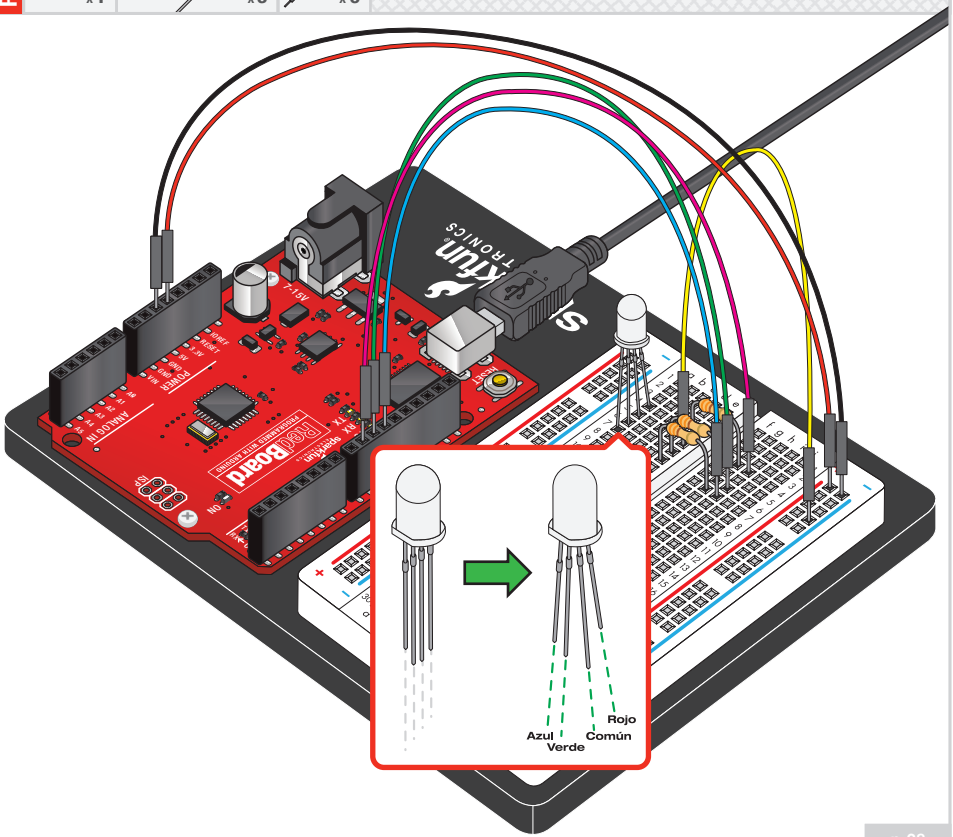


LED RGB

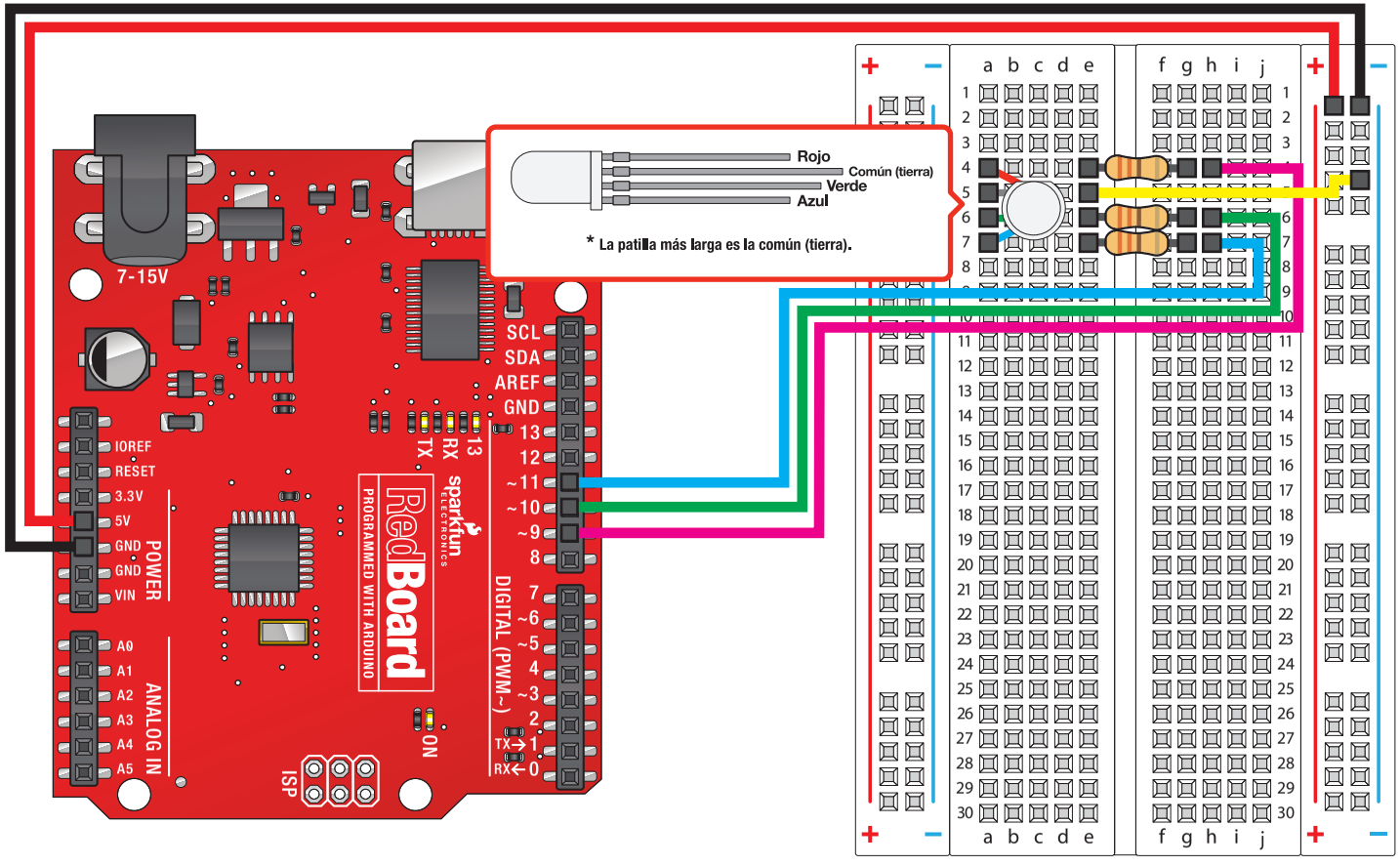
¿Sabes qué es aún más divertido que un LED parpadeante? Cambiar colores con un solo LED. Los LEDs RGB, o rojo-verde-azul por sus siglas en inglés, tienen tres diodos emisores de color que pueden ser combinados para crear todas clases de colores. En este circuito aprenderás a usar un LED RGB para crear combinaciones de colores únicas. ¡Dependiendo de qué tanto brilla cada diodo, casi cualquier color es posible!

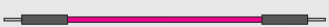
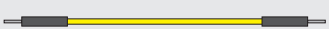
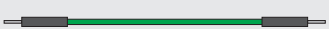
PARTES:

LED	Resistencia de 330Ω	Cable
x1	x3	x6



Circuito 3: LED RGB



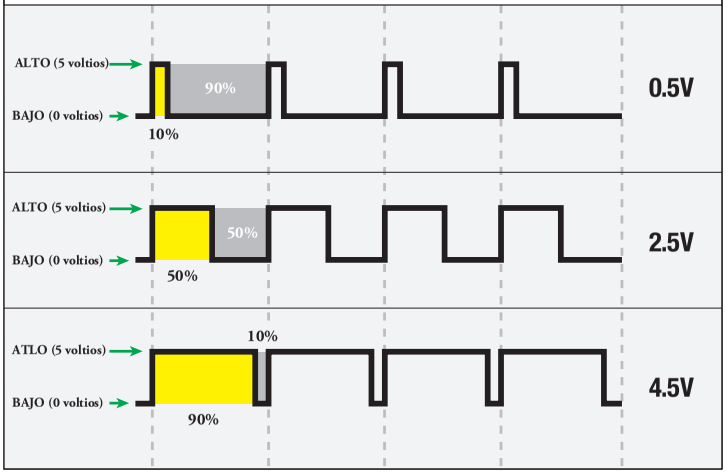
Componentes:	Image Reference:		
LED RGB (5mm) ⚠			
Resistencia de 330Ω			e4 - g4
Resistencia de 330Ω			e6 - g6
Resistencia de 330Ω			e7 - g7
Cable Conector		Pin 9	h4
Cable Conector			e5 - -
Cable Conector		Pin 10	h6
Cable Conector		Pin 11	h7
Cable Conector		5V	+
Cable Conector		GND	-

La asombrosa verdad detrás del analogWrite():

Hemos visto que el Arduino puede leer voltajes analógicos (voltajes entre 0 y 5 voltios) usando la función **analogRead()**. ¿Hay alguna forma de que la RedBoard pueda también emitir voltajes analógicos?

La respuesta es no… y sí. La RedBoard no tiene una salida real de voltaje analógico. Pero, como la RedBoard es tan rápida, puede fingirlo usando algo llamado **PWM ("Pulse-Width Modulation" o "Modulación de Ancho de Pulso")**. Los pines de la RedBoard que tienen un "~" junto a ellos son pines compatibles con salidas PWM/Analógicas.

La RedBoard es tan rápida que puede hacer parpadear un pin entre encendido y apagado casi 1000 veces por segundo. El PWM va un paso adelante al variar la cantidad de tiempo que el pin parpadeante pasa en ALTO vs el tiempo que pasa en BAJO. Si pasa la mayor parte del tiempo en ALTO, un LED conectado a ese pin se verá brillante. SI pasa la mayor parte de su tiempo en BAJO, el LED se verá opaco. Debido a que el pin está parpadeando mucho más rápido de lo que tus ojos pueden detectar, la RedBoard crea la ilusión de una salida analógica "real".



3 Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 3

Notas de Código:

```
for (x = 0; x < 768; x++)
{}
```

Un ciclo for() es usado para ir aumentando un número dentro de un rango y repetidamente correr el código que se encuentra dentro de las llaves {}. En este caso la variable "x" inicia en 0, termina en 767 e incrementa su valor en uno por cada iteración ("x++").

```
if (x <= 255)
{}
else
{}
```

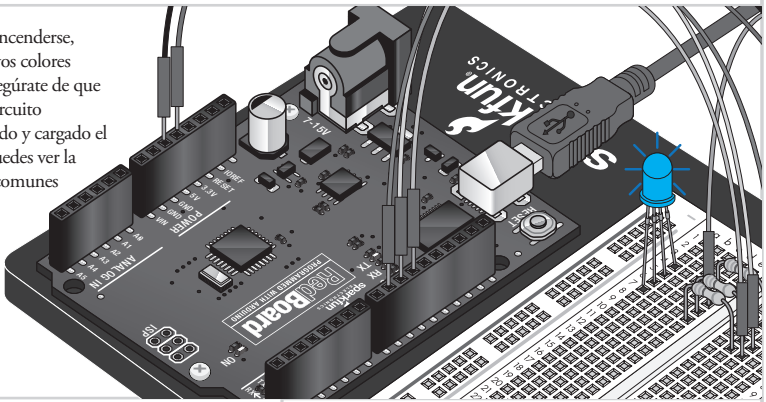
Las declaraciones "if / else" son utilizadas para tomar decisiones en tus programas. La condición dentro de los paréntesis () es evaluada; si es verdadera, se corre el código dentro de las primeras llaves {}. Si no es verdadera, se corre el código dentro de las segundas llaves {}.

```
delay(sensorValue);
```

El procesamiento de la RedBoard es muy muy rápido, es capaz de correr miles de líneas de código cada segundo. Para hacerlo más lento, con el fin de que podemos ver lo que estamos haciendo, debemos insertar retardos dentro del código. La función delay() cuanta en milisegundos; hay 1000ms en un segundo.

Lo que deberías ver:

Deberías ver tu LED encenderse, ¡pero esta vez con nuevos colores locos! Si no lo hace, asegúrate de que hayas ensamblado el circuito correctamente, verificado y cargado el código a tu tarjeta o puedes ver la sección de problemas comunes que se muestra abajo.



Problemas comunes:

El LED se Mantiene Oscuro o Muestra un Color Incorrecto
Con los cuatro pines del LED posicionados tan cerca unos de otros, a veces es fácil posicionar uno de manera incorrecta. Revisa que cada pon esté colocado donde debe ser.

Se ve todo Rojo
El diodo rojo del LED RGB puede ser un poco más brillante que los otros dos. Para hacer tus colores más balanceados, usa una resistencia con más Ohmios. O ajústalo en el código.

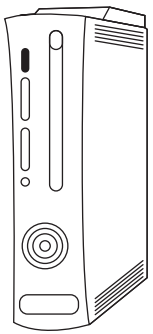
analogWrite(RED_PIN, redIntensity);

to

analogWrite(RED_PIN, redIntensity/3);

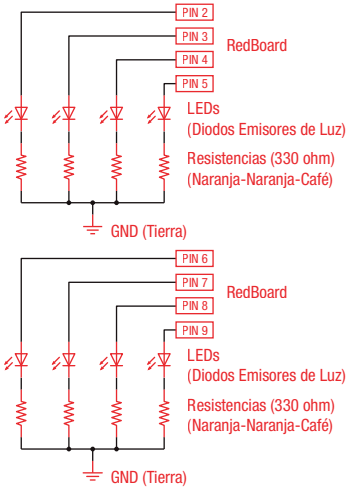
Aplicación en la vida real:

Muchos artículos electrónicos, como consolas de videojuegos, utilizan LEDs RGB para tener la versatilidad de mostrar diferentes colores en la misma área. Muchas veces los colores diferentes representan diferentes estados o condiciones de trabajo.



CIRCUITO #4

4



LEDs Múltiples

Así que ya hiciste que un LED parpadeara entre encendido y apagado – ¡fantástico! Es hora de levantar la barra un poco – conectando OCHO LEDS AL MISMO TIEMPO. Adicionalmente le haremos una pequeña prueba a nuestra RedBoard al crear varias secuencias de luces. Este circuito es un gran paso para empezar a escribir tus propios programas y para que vayas entendiendo la forma en que la RedBoard trabaja. Además de controlar los LEDs, aprenderás un par de trucos de programación para mantener tu código nítido y ordenado:

for() loops - usados cuando quieres correr una porción de código varias veces.

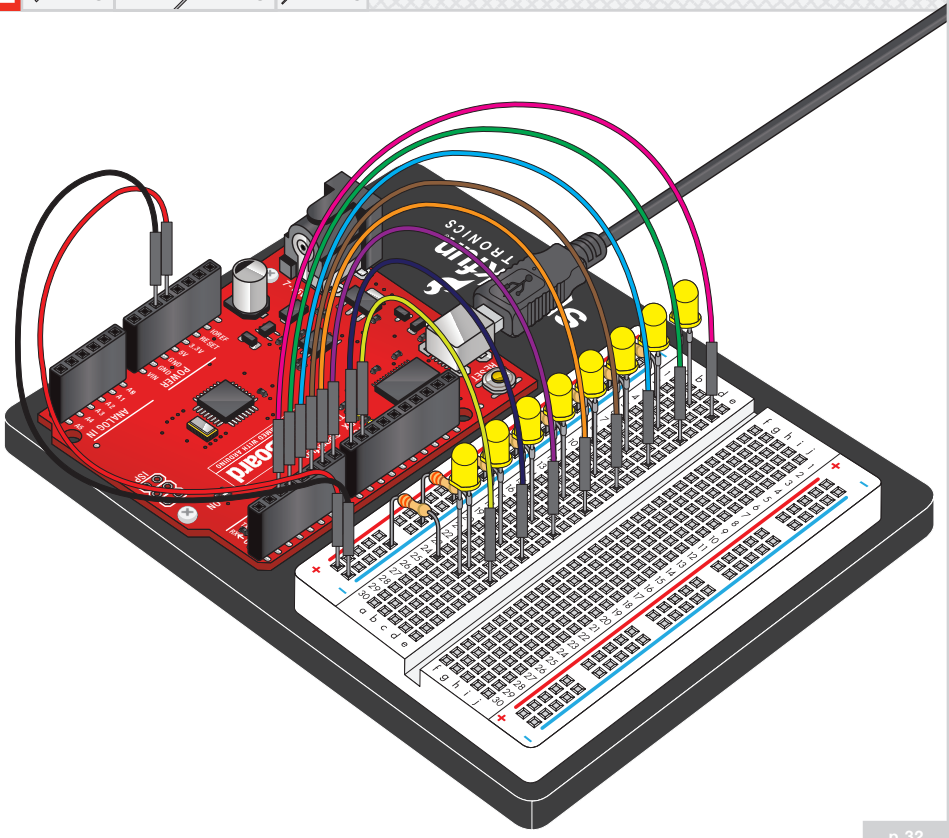
arrays[ ] - utilizados para volver el manejo de variables una operación más fácil, al agruparlas todas juntas.

PARTES:

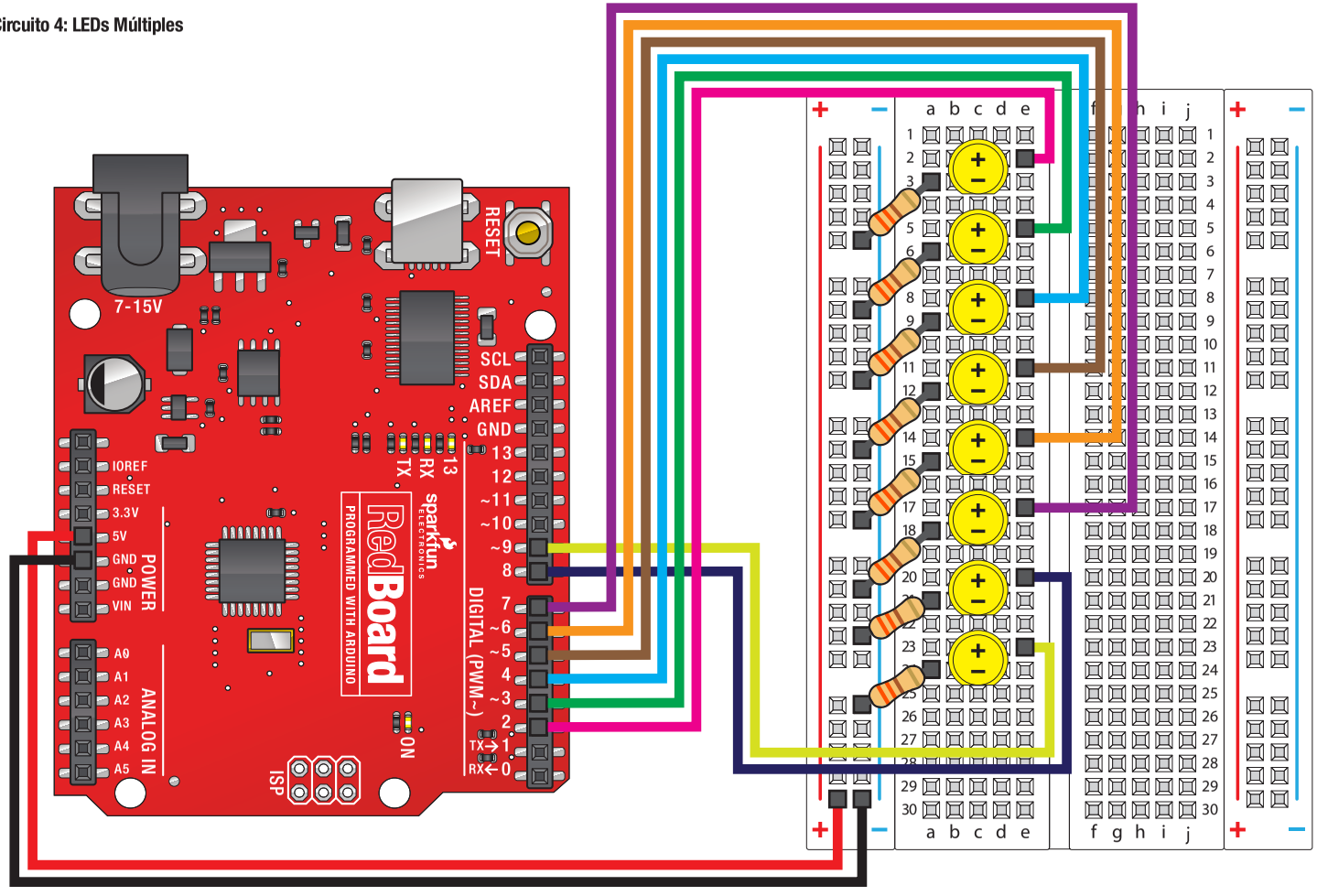

LED x8

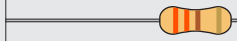
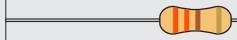
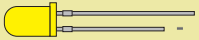

Resistencia de 330Ω x8


Cable x10



Circuito 4: LEDs Múltiples



Componente:	Imagen de Referencia:			Componente:	Imagen de Referencia:		
LED (5mm) ⚠			c2 (+) – c3 (-)	Resistencia de 330Ω			a18 – -
LED (5mm) ⚠			c5 (+) – c6 (-)	Resistencia de 330Ω			a21 – -
LED (5mm) ⚠			c8 (+) – c9 (-)	Resistencia de 330Ω			a24 – -
LED (5mm) ⚠			c11 (+) – c12 (-)	Cable Conector		Pin 2	e2
LED (5mm) ⚠			c14 (+) – c15 (-)	Cable Conector		Pin 3	e5
LED (5mm) ⚠			c17 (+) – c18 (-)	Cable Conector		Pin 4	e8
LED (5mm) ⚠			c20 (+) – c21 (-)	Cable Conector		Pin 5	e11
LED (5mm) ⚠			c23 (+) – c24 (-)	Cable Conector		Pin 6	e14
Resistencia de 330Ω			a3 – -	Cable Conector		Pin 7	e17
Resistencia de 330Ω			a6 – -	Cable Conector		Pin 8	e20
Resistencia de 330Ω			a9 – -	Cable Conector		Pin 9	e23
Resistencia de 330Ω			a12 – -	Cable Conector		5V	+
Resistencia de 330Ω			a15 – -	Cable Conector		GND	-

4

Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 4

Notas de Código:

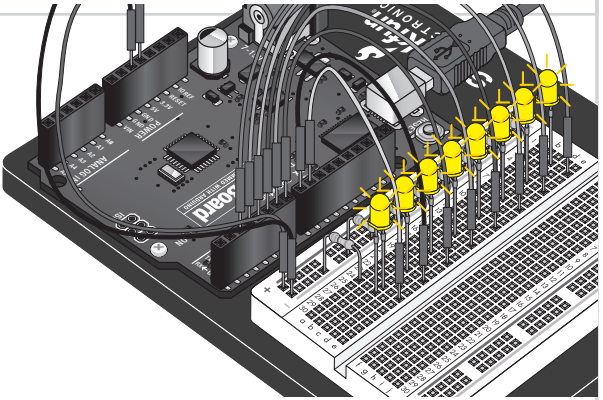
`int ledPins[] = {2,3,4,5,6,7,8,9};` ⇨ Cuanto tienes que manejar muchas variables, un arreglo es una forma útil para agruparlos en un solo lugar. Aquí estamos creando un arreglo de enteros, llamado ledPins, con ocho elementos.

`digitalWrite(ledPins[0], HIGH);` ⇨ Para obtener los elementos de un arreglo deber referirte a ellos por su posición. El primer elemento está en la posición 0, el segundo en la posición 1 y así sucesivamente. Para llamar a un elemento debes usar "ledPins[x]", donde x es la posición. Aquí estamos dando al pin digital 2 un valor de ALTO o **HIGH**, ya que el elemento en la posición 0 del arreglo es "2".

`index = random(8);` ⇨ Las computadoras les gusta hacer lo mismo cada vez que se ejecutan. Pero algunas veces tú quieres hacer algunas cosas de manera aleatoria, como simular el resultado de un dado al lanzarlo. La función random() es una gran forma de hacer esto. Visita **http://arduino.cc/en/reference/random** para más información

Lo que deberías ver:

Este es un circuito similar al número uno, pero en vez de un LED, debes ver todos los LEDs parpadear. Si no lo hacen asegúrate de que hayas ensamblado el circuito correctamente, verificado y cargado el código a tu tarjeta o puedes ver la sección de problemas comunes que se muestra abajo.



Problemas comunes:

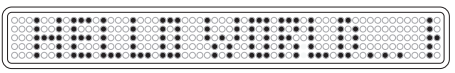
Algunos LEDs fallan al Encender
Es muy común poner un LED al revés. Revisa que los LEDs que no están funcionando estén conectados del lado correcto.

Operando fuera de secuencia
Cuando se tienen ocho cables es muy fácil que algunos se crucen entre ellos. Revisa que el primer LED esté conectado al pin 2 y cada uno de los pines que le siguen.

Empezar desde cero
Es muy fácil colocar un cable sin darse cuenta. Quitar todo y empezar desde cero a colocar los cables suele ser más fácil que rastrear el problema a través del circuito.

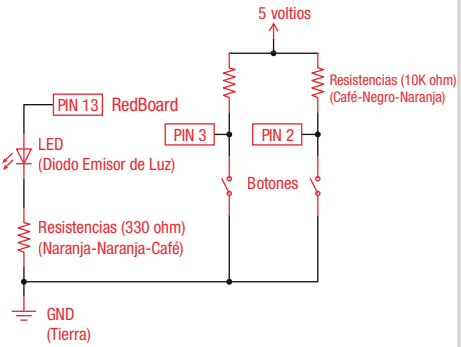
Aplicación en la vida real:

Los letreros de figuras cambiantes son utilizados generalmente para mostrar segmentos cortos de información importante. Estos son construidos a partir de muchos LEDs.



5

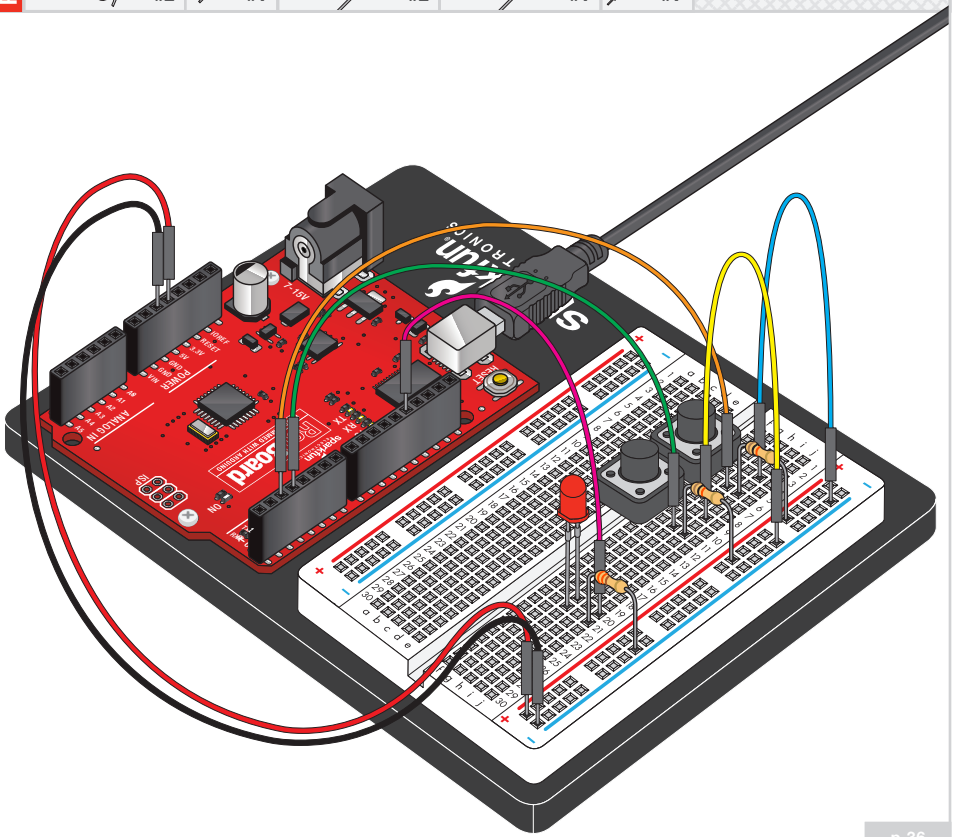
Botones Presionables



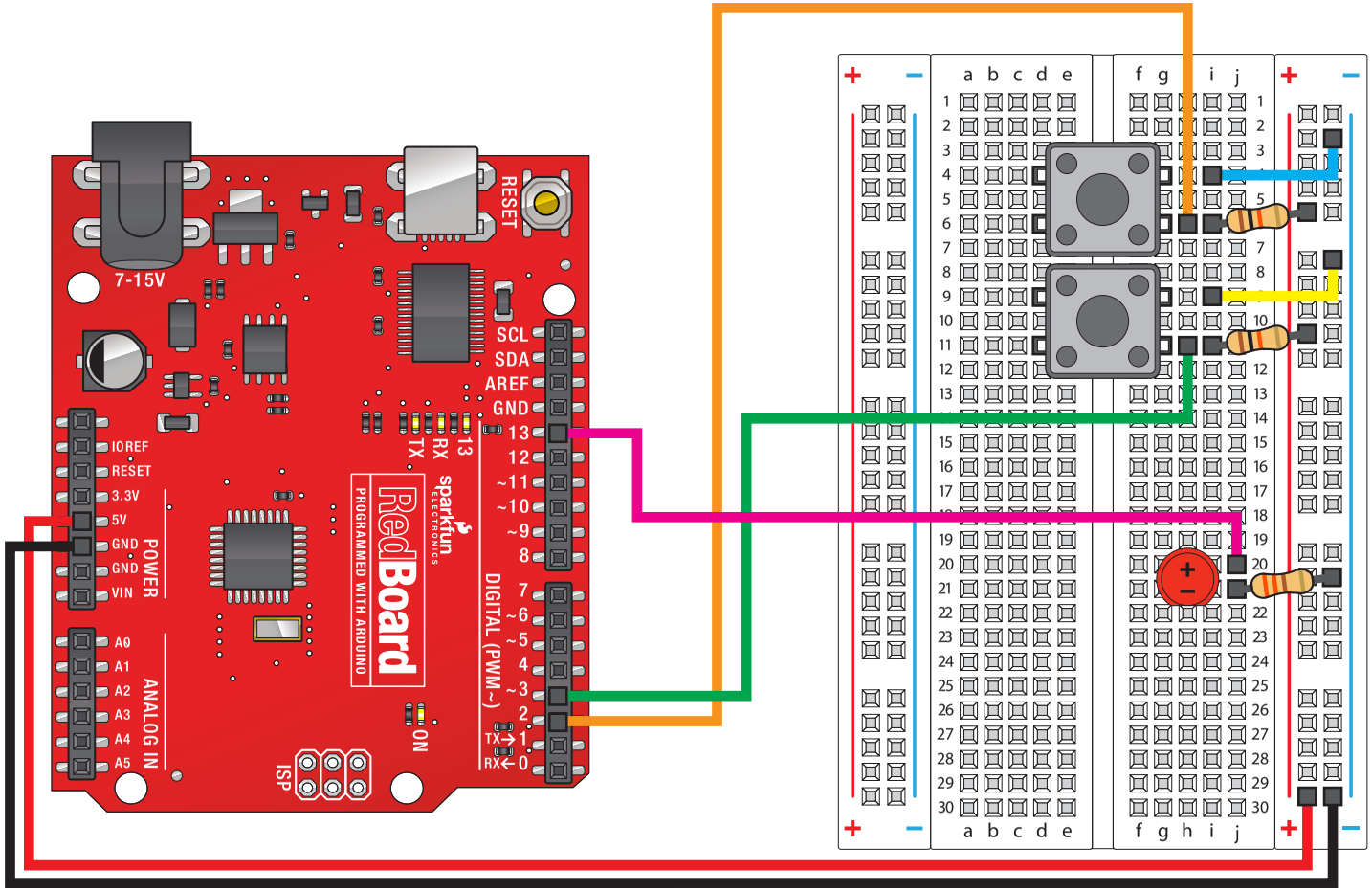
Hasta este momento solo nos hemos enfocado en salidas. Ahora vamos a ir al otro extremo del espectro y vamos a jugar con algunas entradas. En este circuito veremos una de las más comunes y simples de las entradas – un botón presionable. La forma en que un botón funciona con la RedBoard es la siguiente: cuando el botón es presionado, el voltaje marca un BAJO. La RedBoard lee esto y reacciona a partir de ello. En este circuito también usarás una resistencia "pull-up", la cual mantiene el voltaje en ALTO cuando no estás presionando el botón.

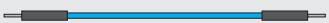
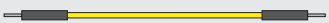
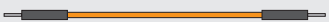
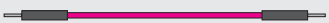
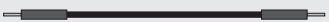
PARTES:

Botón Presionable	LED	Resistencia de 10KΩ	Resistencia de 330Ω	Cable
x2	x1	x2	x1	x7



Circuito 5: Botones Presionables



Componente:	Imagen de Referencia:		
Botón Presionable			d4 g4 d6 g6
Botón Presionable			d9 g9 d11 g11
LED (5mm) ⚠			h20 - h21 + -
Resistencia de 10KΩ			i6 - +
Resistencia de 10KΩ			i11 - +
Resistencia de 330Ω			j21 - -
Cable Conector			i4 - -
Cable Conector			i9 - -
Cable Conector		Pin 2	h6
Cable Conector		Pin 3	h11
Cable Conector		Pin 13	j20
Cable Conector		5V	+
Cable Conector		GND	-

Cómo usar la lógica como un Vulcano:

Una de las cosas que hace a la RedBoard tan útil es el hecho de que pueda realizar decisiones complejas basada en la entrada que está obteniendo. Por ejemplo, puedes hacer un termostato que encienda un calentador si se pone muy frío, un ventilador si se pone muy caliente, riegue tus plantas si se ponen muy secas, entre otras cosas.

Con el fin de hacer este tipo de decisiones, el ambiente de Arduino provee un grupo de operaciones lógicas que te permitirán construir declaraciones "if" complejas. Estas operaciones incluyen:

==	EQUIVALENCIA	**A == B** es verdadero si A y B son lo **MISMO.**
!=	DIFERENCIA	**A != B** es verdadero si A y B **NO SON LO MISMO.**
&&	Y	**A && B** es verdadero si **AMBOS,** A y B, son **VERDADEROS.**
||	O	**A || B** es verdadero si **A** o **B** o **AMBOS** son **VERDADEROS**
!	NEGACIÓN	**!A** es **VERDADERO** so **A** es **FALSO.** **!A** es **FALSO** si **A** es **VERDADERO.**

Puedes combinar estas funcionas para construir declaraciones if() complejas.
Por ejemplo:

```
if ((mode == heat) && ((temperature < threshold) || (override == true)))
{
digitalWrite(HEATER, HIGH);
}
```

…esto encenderá un calentador si estás en modo de calentamiento Y la temperatura es baja, O si enciendes un interruptor manual. ¡Usando estos operadores lógicos puedes programar tu RedBoard para que tome decisiones más inteligentes y tomar el control del mundo que te rodea!

5

Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 5

Notas de Código:

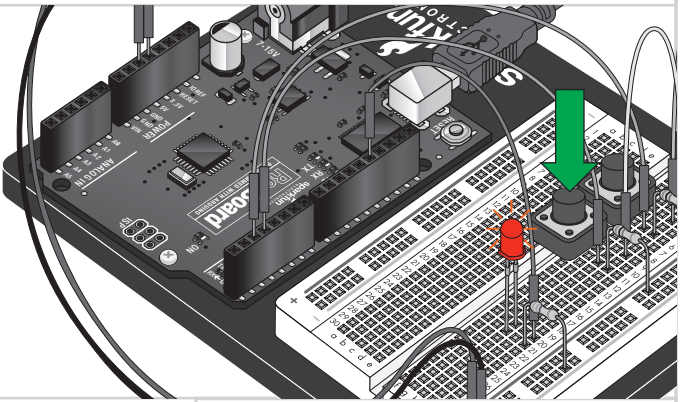
pinMode(button2Pin, INPUT); ⇨ Los pines digitales pueden ser usados tanto como entradas que como salidas. Entes de que uses alguno, necesitas decirle a la RedBoard en cuál dirección lo vas a utilizar.

button1State = digitalRead(button1Pin); ⇨ Para leer una entrada digital, debes usar la función digitalRead(). Esta retornará HIGH si hay 5V presentes en el pin, o LOW si hay 0V presentes en el pin.

if (button1State == LOW) ⇨ Ya que has conectado el botón e tierra (GND), este llevará a un estado de bajo (LOW) cuando sea presionado. Aquí estamos utilizando el operador de "equivalencia" ("==") para ver si el botón está siendo presionado.

Lo que deberías ver:

Debes ver el LED encenderse si presionas algún botón, y apagarse si presionas ambos botones. (¡Mira el código para averiguar por qué!) Si esto no funciona asegúrate de que hayas ensamblado el circuito correctamente, verificado y cargado el código a tu tarjeta, o puedes ver la sección de problemas comunes que se muestra abajo.



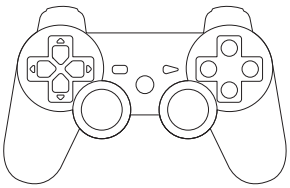
Problemas comunes:

La Luz no se enciende
El botón es cuadrado, y es por esto que es fácil ponerlo de manera incorrecta. Dale un giro de 90 grados y revisa si empieza a funcionar.

No estás satisfecho
No te preocupes, estos circuitos están hechos para crear una forma fácil de jugar con los componentes, pero una vez que lo unes todo, el cielo es el límite.

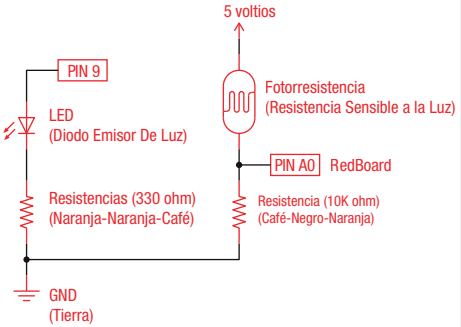
Aplicación en la vida real:

Los botones que usamos aquí son similares a los botones vistos en la mayoría de los controles de videojuegos.



6

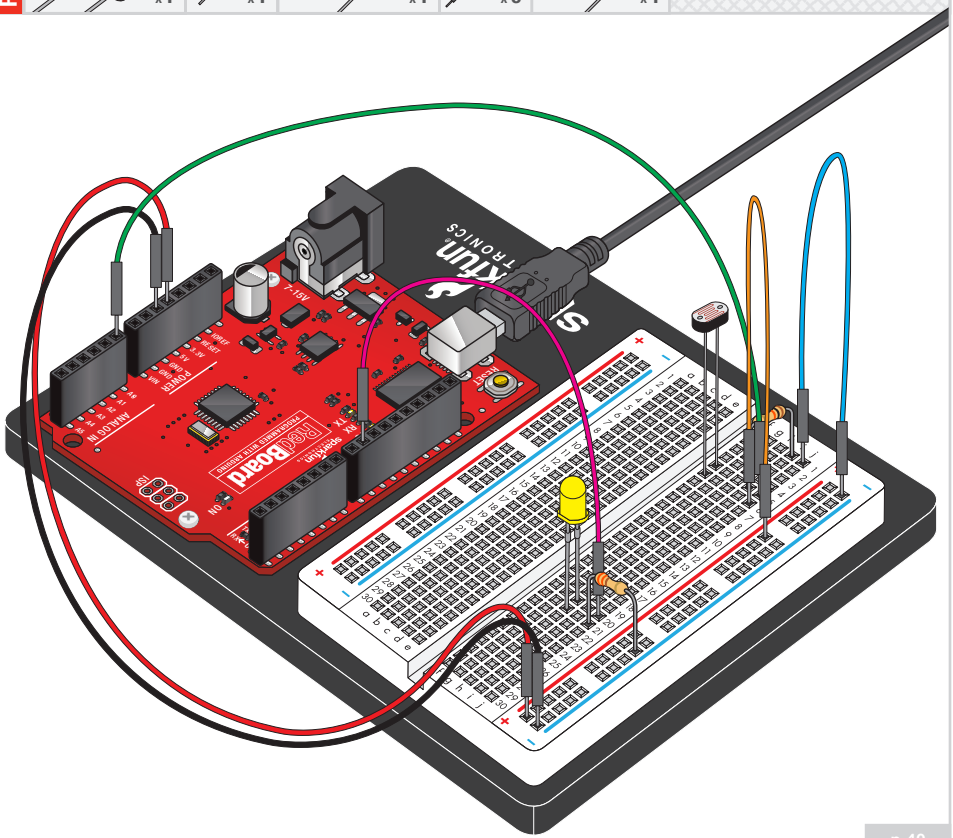
Fotorresistencia



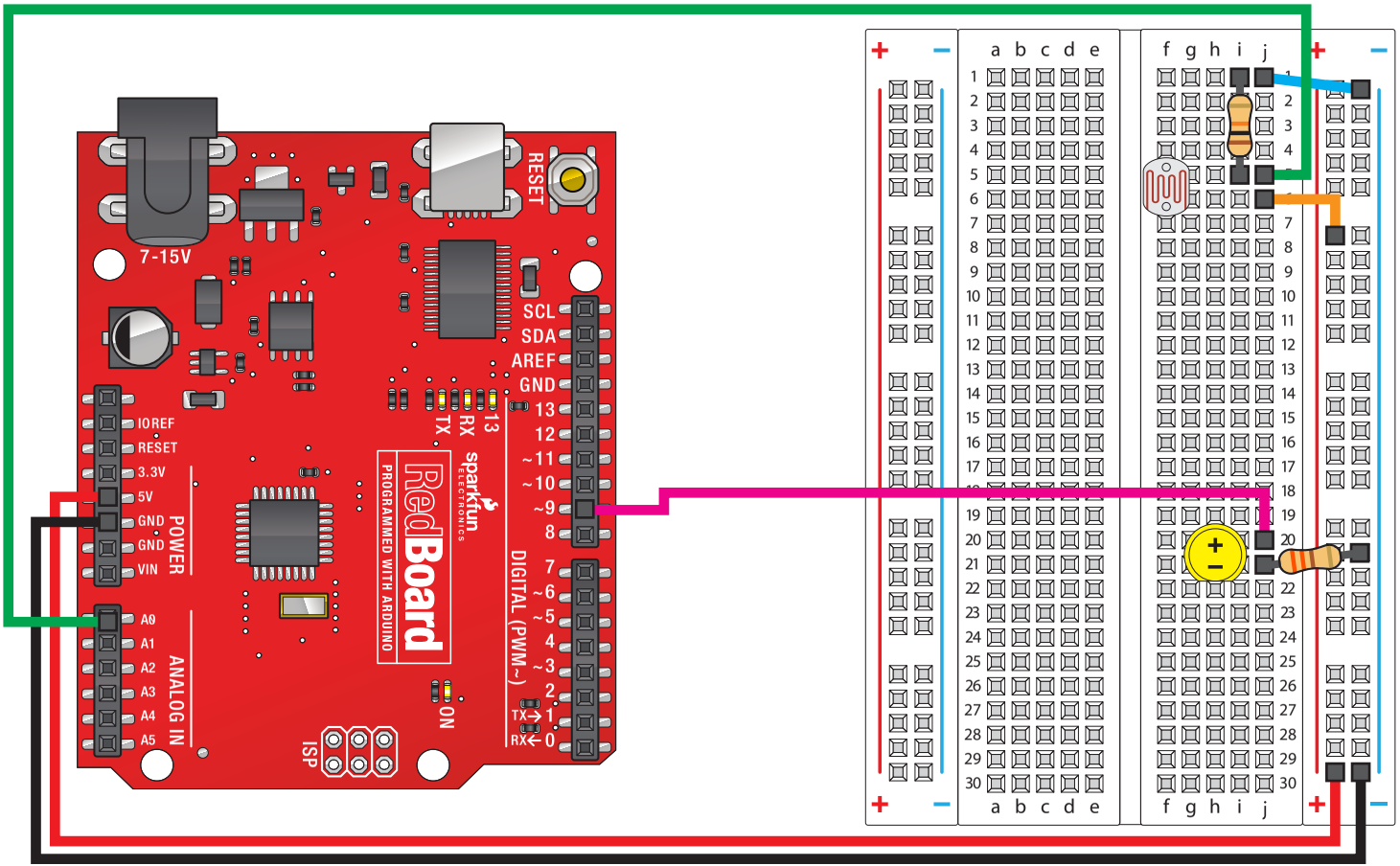
Así que ya has jugado con un potenciómetro, cuya resistencia varía según el movimiento de una perilla. En este circuito estarás usando una fotorresistencia, la cual cambia su resistencia basada en la cantidad de luz que recibe el sensor. Ya que la RedBoard no puede interpretar directamente la resistividad (en vez de esto, lee el voltaje), utilizamos un divisor de voltaje para usar nuestra fotorresistencia. Este divisor de voltaje dará como salida un alto voltaje cuando esté percibiendo mucha luz y un bajo voltaje cuando no lo haga.

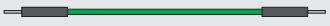
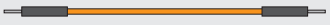
PARTES:

Fotorresistencia	LED	Resistencia de 330Ω	Cable	Resistencia de 10KΩ
x1	x1	x1	x6	x1



Circuito 6: Fotorresistencia

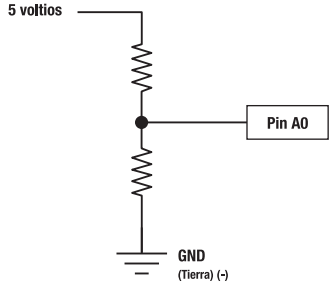


Componente:	Imagen de Referencia:		
Fotorresistencia ⚠			f5 - f6
LED (5mm) ⚠			h20 - h21 (+ -)
Resistencia de 330Ω			j21 - -
Resistencia de 10KΩ			i1 - i5
Cable Conector			j1 - -
Cable Conector		A0	j5
Cable Conector			j6 - +
Cable Conector		Pin 9	j20
Cable Conector		5V	+
Cable Conector		GND	-

Midiendo sensores resistivos:

Muchos de los sensores que utilizarás (potenciómetros, fotorresistencias, etc.) son resistores disfrazados. Sus resistencias cambian en proporción a lo que estén detectando (nivel de luz, temperatura, sonido, etc.).

Los pines de entrada analógica de la RedBoard miden voltaje, no resistencia. Pero podemos usar fácilmente sensores resistivos con la RedBoard incluyéndolos como parte de un "divisor de voltaje".



Un divisor de voltaje consiste en dos resistores. El resistor de "arriba" es el sensor que utilizarás. El de "abajo" es un resistor normal fijo. Cuando conectas el resistor de arriba a 5 voltios, y el de abajo a tierra, el voltaje en el medio debe ser proporcional al resistor de abajo relativo al total de resistividad (resistor de arriba + resistor de abajo). Cuando uno de los resistores cambia (como lo hacen tus sensores cuando detectan algo), ¡el voltaje de salida cambia también!

Aunque la resistencia de los sensores va a variar, los sensores resistivos (sensor flex, sensor de luz, potenciómetro suave y trimpot) utilizados en el SIK son de alrededor de 10Kohms. Generalmente queremos que el resistor fijo esté cerca de este valor, por lo que usar un resistor de 10K es una gran opción para el resistor fijo de "abajo". Puedes notar que el resistor fijo no necesariamente debe ser el de abajo. Hacemos esto solo con el fotodiodo pues más luz = más voltaje, pero pueden ser intercambiados y obtendríamos la respuesta opuesta.

6

Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 6

Notas de Código:

lightLevel = map(lightLevel, 0, 1023, 0, 255);

Parámetros
map(value, fromLow, fromHigh, toLow, toHigh)

value: el número a mapear
fromLow: el límite inferior del rango actual del valor
fromHigh: el límite superior del rango actual del valor
toLow: el límite inferior del rango objetivo del valor
toHigh: el límite superior del rango objetivo del valor

Cuando leemos una señal analógica usando analogRead(), esta lectura será un número de 0 a 1023. Pero cuando queremos manejar un pin PWM usando analogWrite(), este requiere un número de 0 a 255. Podemos "encoger" el mayor rango dentro del menor usando la función map().

Visita **http://arduino.cc/en/reference/map** para más información.

lightLevel = constrain(lightLevel, 0, 255);

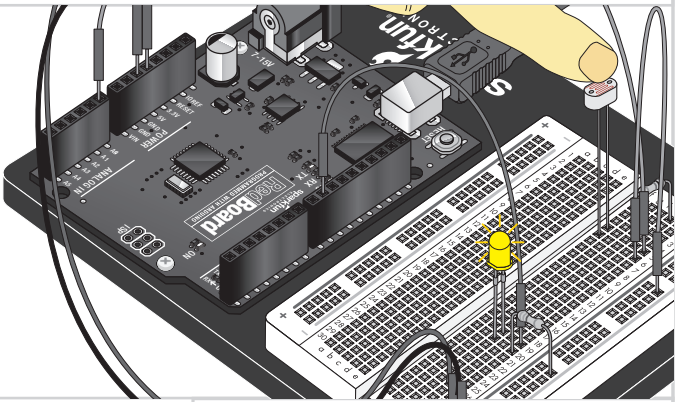
Parámetros
constrain(x, a, b)

x: el número a restringir, todo tipo de dato
a: : el límite inferior del rango, todo tipo de dato
b: el límite superior del rango, todo tipo de dato

Ya que la función map() aún puede retornar números fuera del rango objetivo, utilizamos también una función llamada constrain() la cual "restringirá" los números dentro de un rango. Si el número está fuera del rango se convertirá en el mayor o menor número. Si está dentro del rango se quedará igual.
Visita **http://arduino.cc/en/reference/constrain** para más información.

Lo que deberías ver:

Deberías ver el LED brillar más o menos de acuerdo a la cantidad de luz que el Fotorresistor está leyendo. Si esto no funciona asegúrate de que hayas ensamblado el circuito correctamente, verificado y cargado el código a tu tarjeta, o puedes ver la sección de problemas comunes que se muestra abajo.



Problemas comunes:

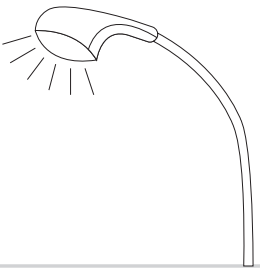
El LED Permanece Oscuro
Este es un error que seguimos cometiendo una y otra vez, si tan solo pudieran fabricar un LED que funcione de ambas formas. Sácalo y dale un giro.

No está Respondiendo a los Cambios de Luz
Dado que el espaciado de los cables en el Fotorresistor no es estándar, es fácil colocarlo erróneamente. Revisa que esté colocado en el lugar correcto.

Aún no funciona del todo
Es probable que estés en un cuarto muy claro o muy oscuro. Prueba encendiendo o apagando la luces para ver si esto ayuda. O si tienes una linterna cerca de ti inténtalo con eso.

Aplicación en la vida real:

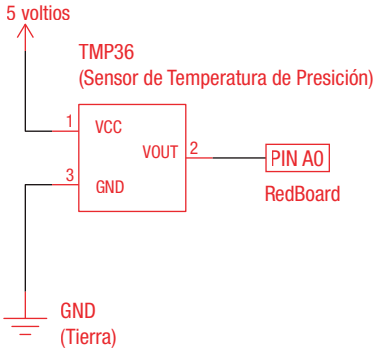
Una lámpara colocada en un camino usa un pequeño sensor para detectar cuando encender las luces en la noche.



CIRCUITO #7

7

Sensor de Temperatura

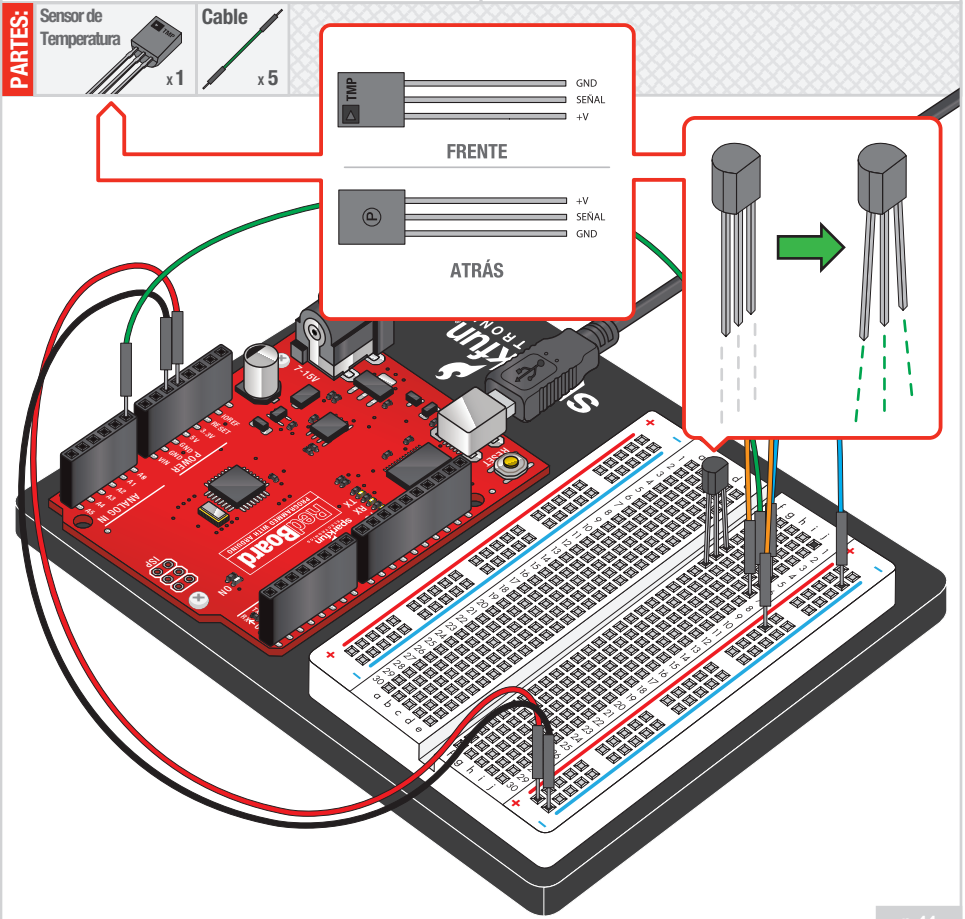


Un sensor de temperatura es exactamente lo que suena – un sensor usado para medir la temperatura del ambiente. Este particular sensor tiene tres pines – un positivo, una tierra y una señal. Este es un sensor de temperatura lineal. Un cambio en la temperatura de un grado centígrado es igual a un cambio de 10 milivoltios en la salida del sensor. El sensor TMP36 tiene un valor de 750mV a 25°C (temperatura ambiente). En este circuito, aprenderás como integrar el sensor de temperatura con tu RedBoard y usar el monitor serial del Arduino IDE para mostrar la temperatura.

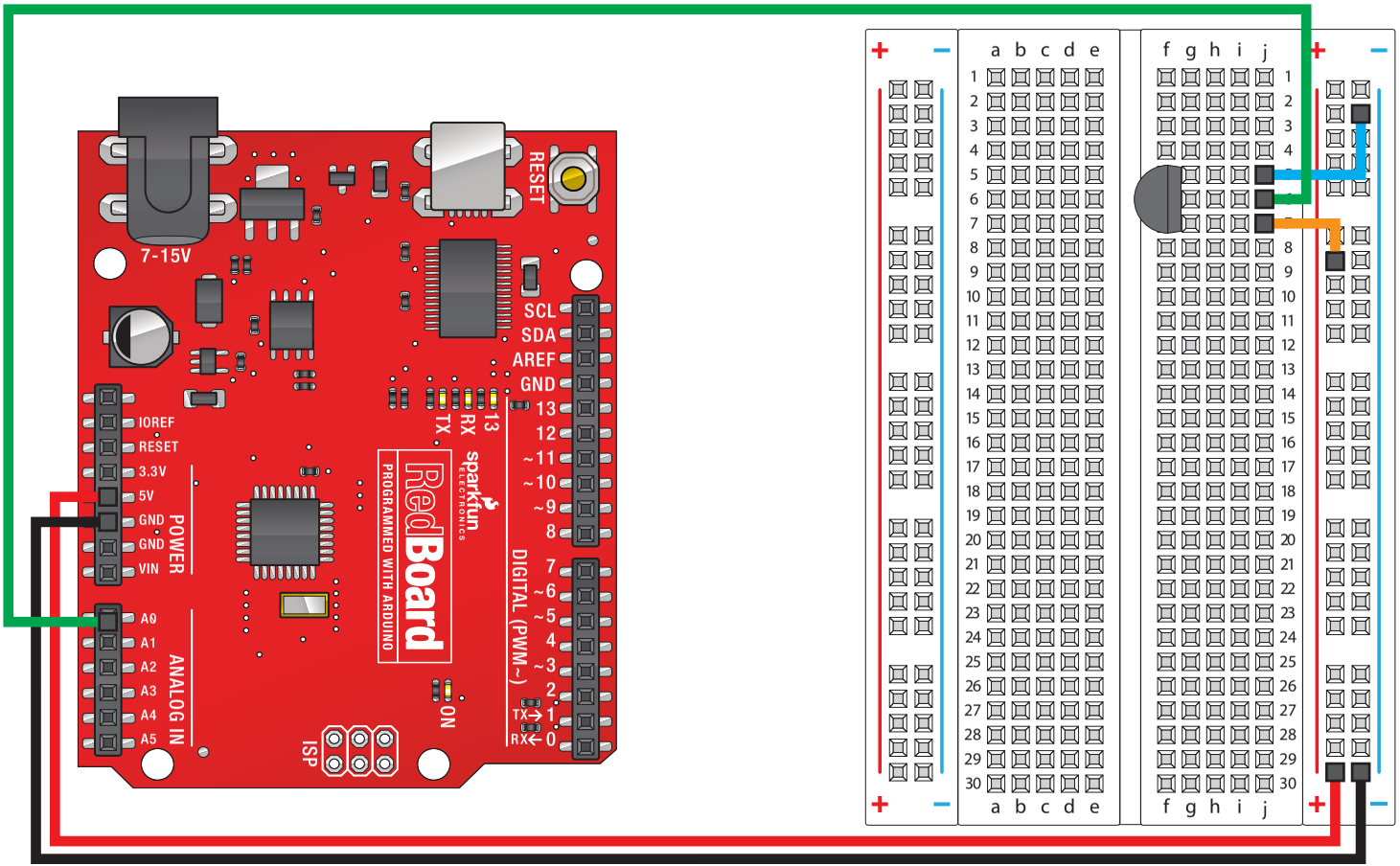
> **!** Cuando estés construyendo el circuito ten cuidado de no confundir el sensor de temperatura con el transistor, son casi idénticos. Busca la etiqueta "TMP" en el cuerpo del sensor de temperatura.

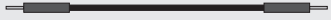
PARTES:

Sensor de Temperatura	Cable
x1	x5



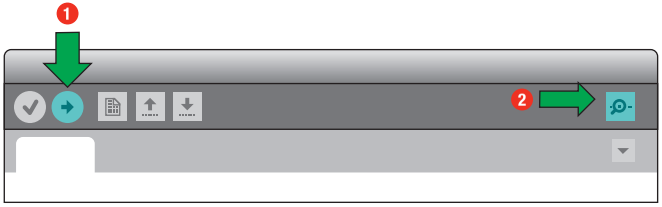
Circuito 7: Sensor de Temperatura

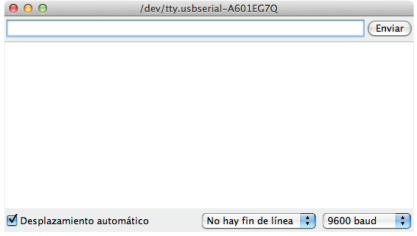


Componente:	Imagen de Referencia:		
Sensor de temperatura			f5 f6 f7 - V_{OUT} +
Cable Conector			j5 -
Cable Conector		A0	j6
Cable Conector			j7 +
Cable Conector		5V	+
Cable Conector		GND	-

Abriendo tu monitor serial:

Este circuito usa el monitor serial del IDE de Arduino. Para abrirlo primero carga el programa, luego haz clic en el botón que se ve como una lupa en un cuadrado. Para que el monitor serial opere correctamente debe estar configurado al mismo baud rate (velocidad en bits por segundo) que el código que estás corriendo. Este código corre a 9600 baud; si la configuración del baud rate es diferente de 9600, cámbiala a 9600.





7

Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 7

Notas de Código:

Serial.begin(9600);

Antes de usar el monitor serial debes llamar la función Serial.begin() para inicializarlo. 9600 es el "baud rate", o la velocidad de comunicaciones. Cuando dos dispositivos se comunican entre ellos, ambos deben tener la misma velocidad.

Serial.print(degreesC);

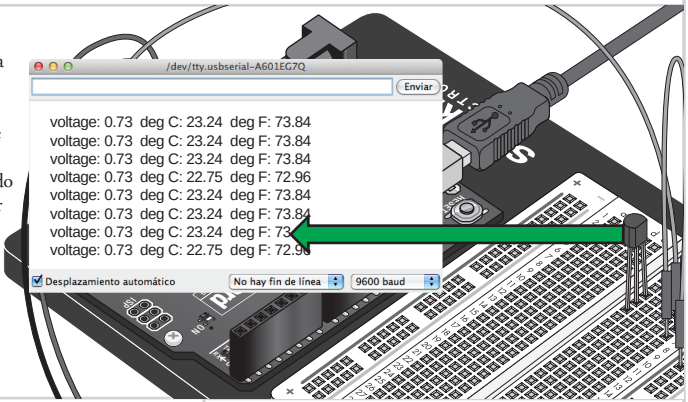
El comando Serial.print() es muy inteligente. Puede imprimir casi cualquier cosa que le puedas tirar, incluyendo variables de todo tipo, texto entre comillas (conocido como "strings"), etc. Visita **http://arduino.cc/en/serial/print** para más información.

Serial.println(degreesF);

Serial.print() imprimirá todo en la misma línea. Serial.println() moverá el cursor a la línea siguiente. Utilizando estos comandos juntos puedes crear impresiones de texto y datos fáciles de leer.

Lo que Deberías ver:

Debes ser capaz de leer en el monitor serial del Arduino IDE la temperatura que tu sensor de temperatura está detectando. Si esto no funciona asegúrate de que hayas ensamblado el circuito correctamente, verificado y cargado el código a tu tarjeta, o puedes ver la sección de problemas comunes que se muestra abajo.



Problemas comunes:

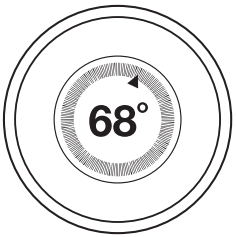
Nada Parece estar Ocurriendo
Este programa no tiene ningún indicador externo que diga que está funcionando. Para ver resultados debes abrir el monitor serial del IDE de Arduino (instrucciones en páginas anteriores).

Se Muestran Caracteres sin Sentido
Esto sucede porque el monitor serial está recibiendo datos a una velocidad diferente de la esperada. Para solucionar esto haz clic en la caja que dice "*** baud" y cámbiala a "9600 baud".

El Valor de Temperatura no está Cambiando
Trata prensando el sensor con tus dedos para calentarlo o presionando una bolsa de hielo en él para enfriarlo.

Aplicación en la Vida Real:

Edificios con sistemas de control de clima usan sensores de temperatura para monitorear y mantener sus configuraciones.

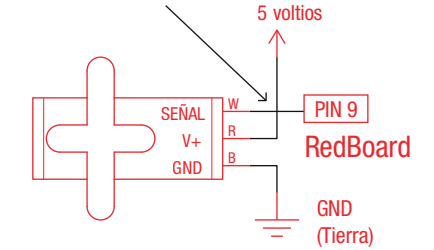


8

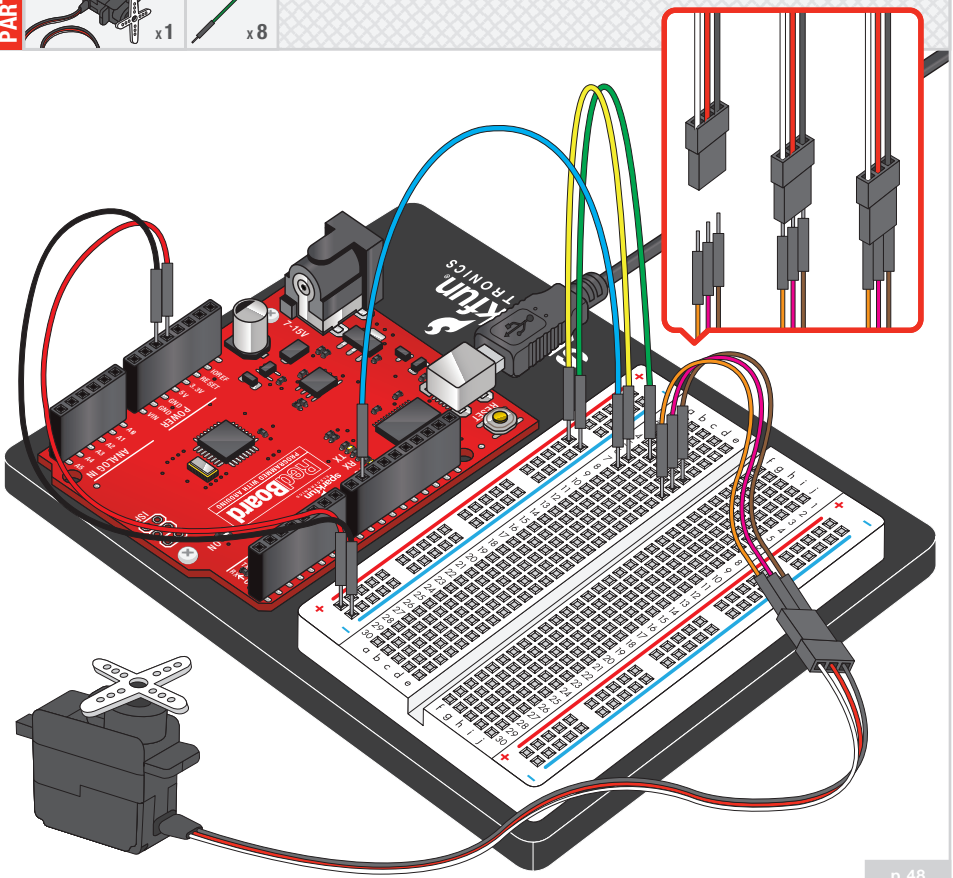
Solo un Servo

Los servos son ideales para aplicaciones electrónicas pues pueden hacer muy bien algo que los motores no – pueden moverse a una posición determinada de forma precisa. Variando el ancho de pulso del voltaje de salida hacia un servo puedes moverlo a una posición específica. Por ejemplo, un pulso de 1.5 milisegundos moverá el servo 90 grados. Es este circuito aprenderás cómo usar PWM (modulación de ancho de pulso) para controlar y rotar un servo.

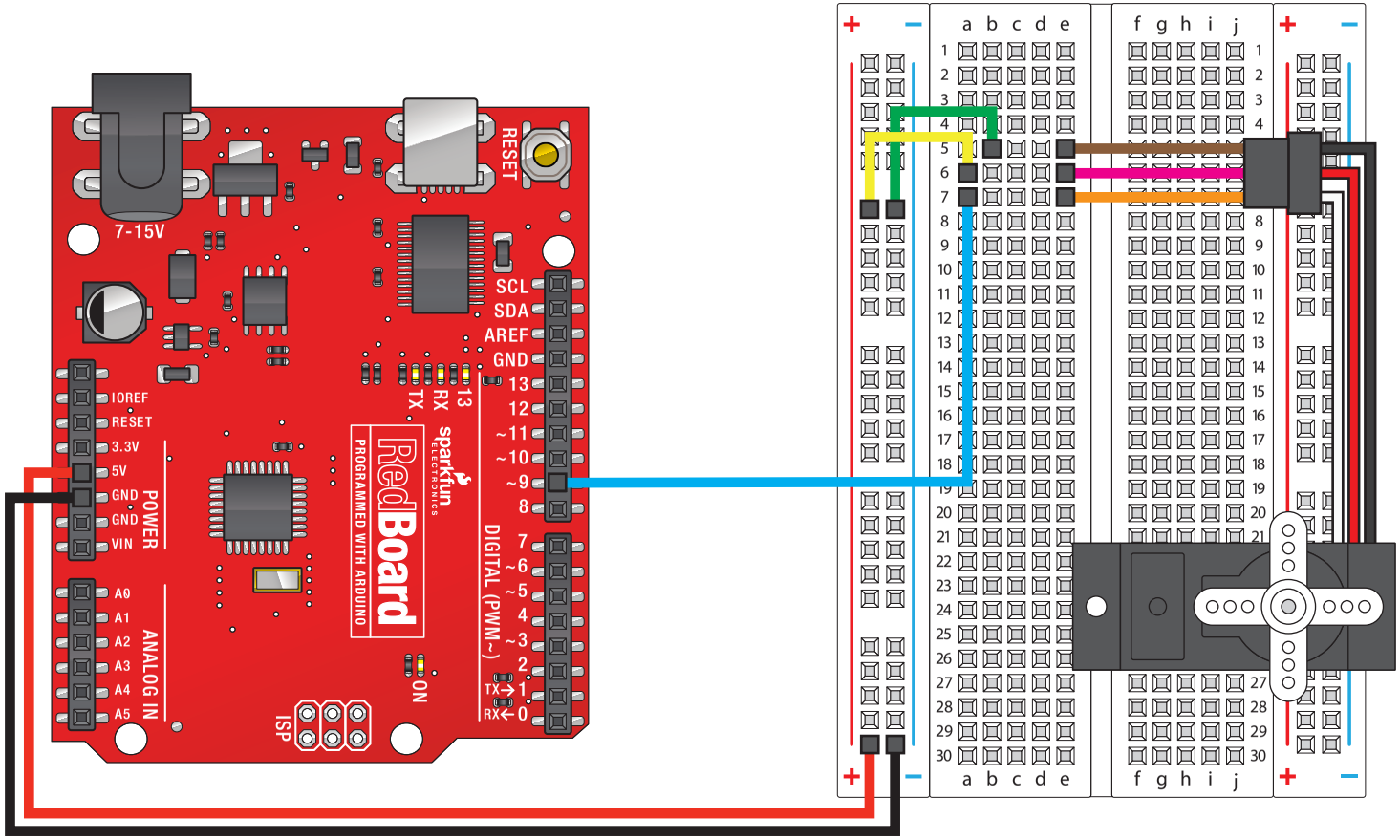
Si no hay punto de unión significa que no hay conexión

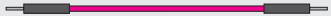
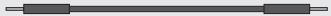


PARTES:



Circuito 8: Solo un Servo



Componente:	Imagen de Referencia:		
Servo			e5 e6 e7
Cable Conector			e5 ●
Cable Conector			e6 ●
Cable Conector			e7 ○
Cable Conector		Pin 9	a7
Cable Conector			b5 -
Cable Conector			a6 +
Cable Conector		5V	+
Cable Conector		GND	-

Expande tus horizontes usando Bibliotecas:

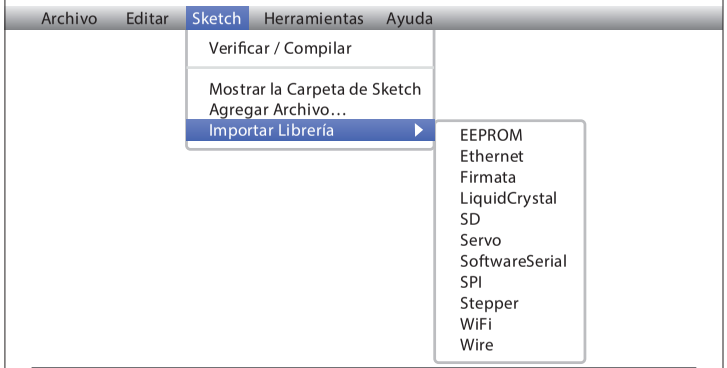
El ambiente de desarrollo de Arduino te da un set de comandos incorporados para manipular entradas y salidas básicas, tomar decisiones usando lógica, resolver problemas matemáticos, etc. Pero el poder real del Arduino es la gran comunidad que lo usa y sus deseos de compartir su trabajo.

Las bibliotecas son colecciones de comandos nuevos que han sido empaquetadas juntas para que sea fácil incluirlas en tus diseños. Arduino viene con un puñado de bibliotecas útiles, como la biblioteca servo utilizada en este ejemplo, las cuales pueden ser utilizadas para conectar dispositivos más avanzados (pantallas LCD, motores "stepper", puertos Ethernet, etc.)

Visita **http: //arduino.cc/en/reference/libraries** para Encontrar la lista de las bibliotecas estándar e información sobre cómo usarlas.

Pero cualquiera puede crear una biblioteca, y si quieres usar un nuevo sensor o dispositivo de salida, es probable que alguien ya haya escruto una que conecte ese dispositivo con la RedBoard. Muchos de los productos de SparkFun vienen con bibliotecas de Arduino y puedes encontrar aún más usando Google y el "Arduino Playground" en **http://arduino.cc/playground/**. ¡Cuando TÚ pongas a trabajar la RedBoard con un nuevo dispositivo, considera hacer una biblioteca para este y compártela con el mundo!

Para usar una biblioteca en un diseño, selecciónala en **Sketch > Importar Librería**.



Después de importar la biblioteca en tu código vas a tener acceso a un número de comandos y funciones previamente escritas. Para más información acerca de cómo usar las funciones de las bibliotecas estándar puedes acceder a: **http://arduino.cc/en/Reference/Libraries.**

8

Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 8

Notas de Código:

#include <Servo.h>

#include es un comando "preprocesador" especial que inserta una biblioteca (o cualquier otro archivo) en tu diseño. Puedes escribir este comando tú mismo, o escoger una biblioteca desde el menú "Sketch / Importar Librería".

Servo servo1;

servo1.attach(9);

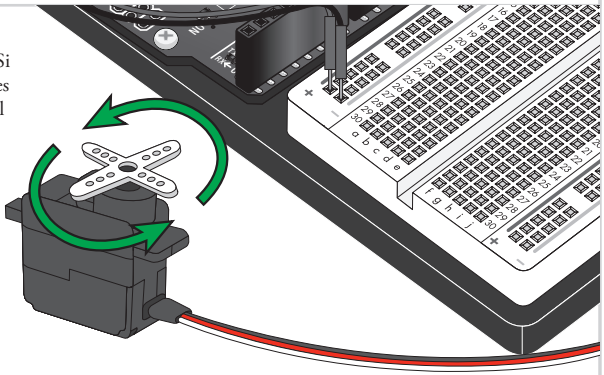
La biblioteca servo añade nuevos comandos que te permiten controlar un servo. Para preparar el Arduino para que controle un servo debes crear primero un "objeto" Servo por cada servo (aquí lo llamamos "servo1"), y luego hacerle un "attach" a un pin digital (aquí estamos usando el pin 9).

servo1.write(180);

Los servos en este kit no giran una vuelta completa, pero pueden ser ordenados para que se muevan a una posición específica. Usamos el comando write() de la biblioteca servo para mover un servo a un número específico de grados (0 a 180). Recuerda que el servo requiere tiempo para moverse, así que dale un pequeño delay() si es necesario.

Lo que deberías ver:

Deberías ver tu motor servo moverse a varias posiciones a diferentes velocidades. Si el motor no se mueve, revisa tus conexiones y asegúrate de haber verificado y cargado el código, o mira la sección de problemas comunes que se muestra más abajo.



Problemas comunes:

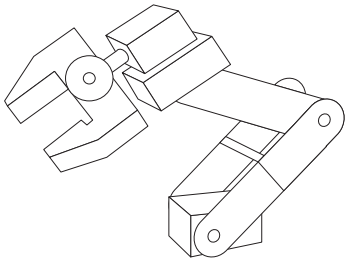
El Servo no Gira
Aún con cables de colores es sorprendentemente fácil conectar un servo al revés. Este podría ser el problema.

Aún no funciona
Un error que hicimos una o dos veces fue simplemente olvidar conectar la alimentación (los cables rojo y café) a +5 voltios y tierra.

Conecta y Empieza
Si el servo se comienza a mover, luego se detiene y hay una luz parpadeante en tu RedBoard, la fuente de poder que estás usando no está dando la talla. Usar un adaptador de pared en lugar del conector USB debería resolver este problema.

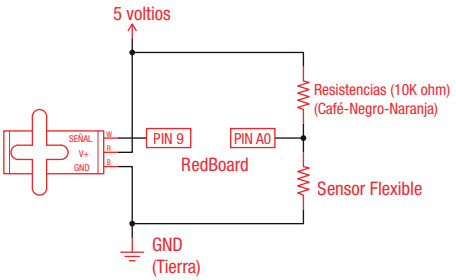
Aplicación en la vida real:

Los brazos robóticos que podrías ver en líneas de ensamblaje o en películas de ciencia ficción tienen servos dentro de ellos.



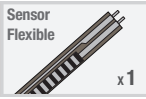
9

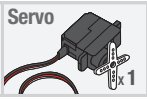
Sensor Flexible



En este circuito utilizaremos un sensor flexible para medir, bueno, ¡flexibilidad! Un sensor flexible usa carbono en una banda plástica para actuar como una resistencia variable, pero en lugar de cambiar la resistencia girando una perilla, la cambias al doblar el componente. De nuevo utilizaremos un "divisor de voltaje" para detectar este cambio de resistencia. El sensor se dobla en una dirección y cuanto más se doble, más alta es la resistividad que adquiere; tiene un rango entre 10Kohm a 35Kohm. En este circuito utilizaremos el doblamiento del sensor flexible para controlar la posición de un servo.

PARTES:

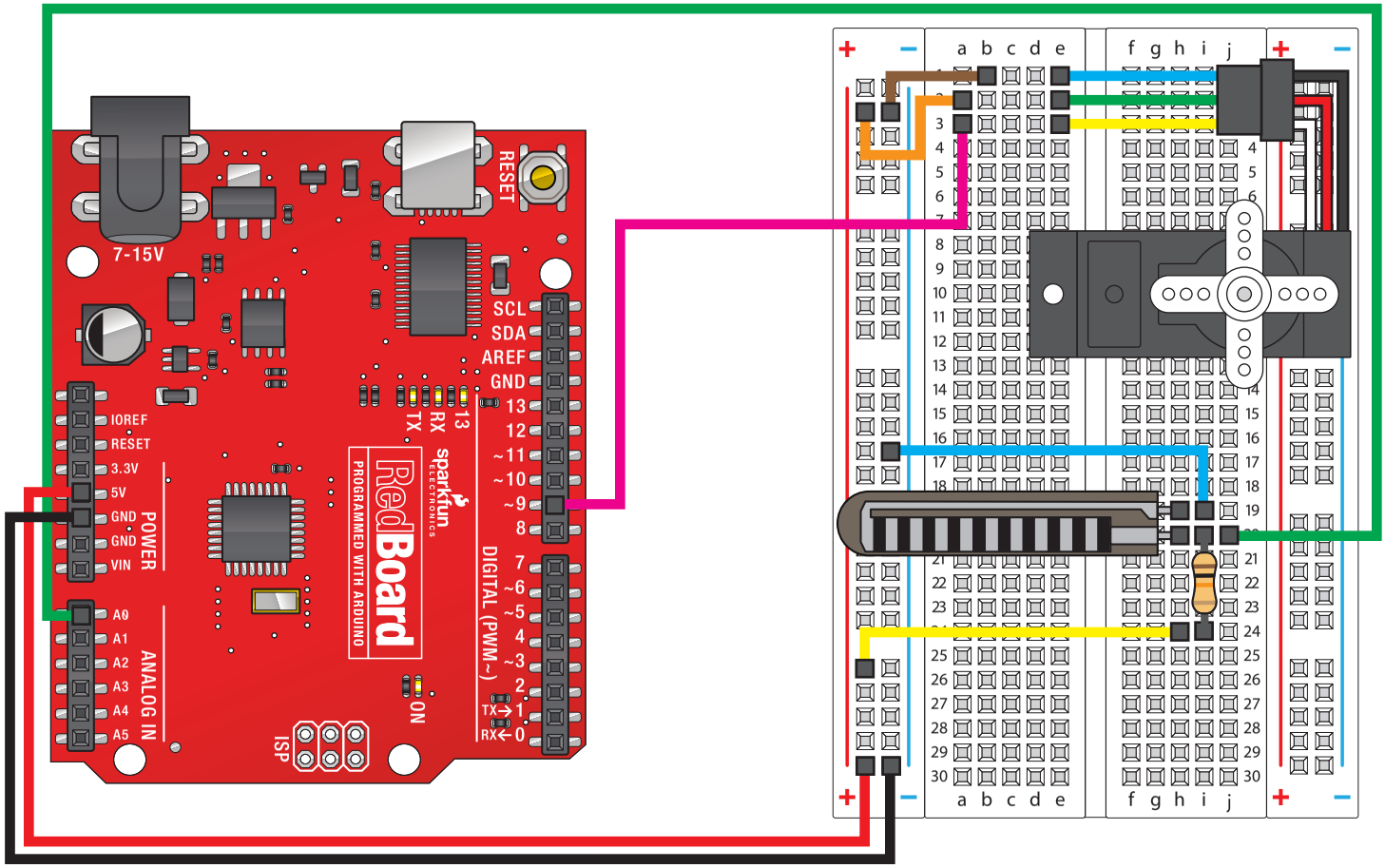

Sensor Flexible x1

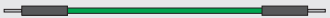
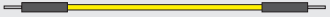
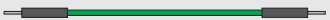
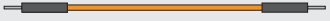
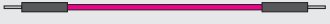
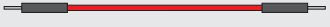

Servo x1


Resistencia de 10KΩ x1


Cable x11

Circuito 9: Sensor Flexible



Componente:	Imagen de Referencia:		
Servo			e1 - e2 - e3
Cable Conector			e1 -
Cable Conector			e2 -
Cable Conector			e3 -
Sensor Flexible			h19 - h20
Resistencia de 10KΩ			i20 - i24
Cable Conector			i19 -
Cable Conector		A0	j20
Cable Conector			h24 - +
Cable Conector			b1 -
Cable Conector			a2 - +
Cable Conector		Pin 9	a3
Cable Conector		5V	+
Cable Conector		GND	-

Depurando tus diseños con el Monitor Serial:

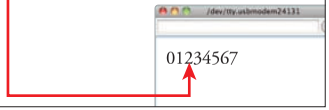
Nos pasa a todos – escribes tu código, compila y carga exitosamente, pero no puedes entender por qué no está haciendo lo que quieres que haga. Las computadoras más grandes tienen pantallas, teclados y ratones que puedes usar para depurar tu código, pero computadoras pequeñas como la RedBoard no tienen esas cosas.

La clave para tener visibilidad con un microcontrolador son las salidas. Estas pueden ser casi cualquier cosa, incluyendo LEDs y bocinas, pero una de las herramientas más útiles es el monitor serial. Usando Serial.print() y println() puedes imprimir fácilmente texto y datos entendibles por los humanos desde la RedBoard a una ventana en tu computadora. Esto es genial para el resultado final de tu diseño, pero además es increíblemente útil para depurar.

```
for (x = 0; x < 8; x++)
{
    Serial.print(x);
}
```

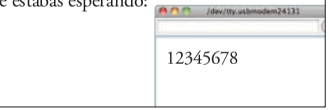
Digamos que necesitas un ciclo for() de 1 a 8, pero parece que tu código no está funcionando bien. Solo agrega **Serial.begin(9600)**; a tu función **setup()**, y agrega Serial.print() o **println()** a tu ciclo:

Querías 1 a 8, pero el ciclo te está dando 0 a 7. ¡Ups! Ahora necesitas arreglar el ciclo.




```
for (x = 1 ; x < 9 ; x++)
{
    Serial.print(x);
}
```

Y si corres el código de nuevo, obtendrás la salida que estabas esperando:



9

Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 9

Notas de Código:

`servoposition = map(flexposition, 600, 900, 0, 180);`
map(value, fromLow, fromHigh, toLow, toHigh)

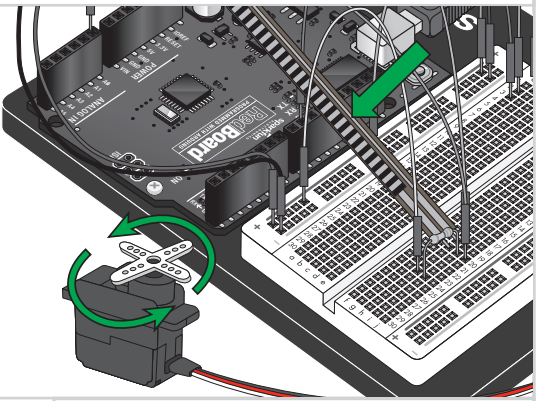
Ya que la combinación sensor flexible / resistencia no nos entrega un rango completo de 0 a 5V, estamos usando la función map() como una forma de reducir ese rango. Aquí le decimos a la función que espere valores de 600 a 900, en vez de 0 a 1023.

`servoposition = constrain(servoposition, 0, 180);`
constrain(x, a, b)

Ya que la función map() aún puede retornar números fuera del rango objetivo, utilizamos también una función llamada constrain() la cual "restringirá" los números dentro de un rango. Si el número está fuera del rango se convertirá en el mayor o menor número. Si está dentro del rango se quedará igual.

Lo que deberías ver:

Deberías ver el motor servo moverse de acuerdo a cuanto está siendo doblado el sensor flexible. Si esto no está funcionando asegúrate de que hayas ensamblado el circuito correctamente, verificado y cargado el código a tu tarjeta, o puedes ver la sección de problemas comunes que se muestra abajo.



Problemas comunes:

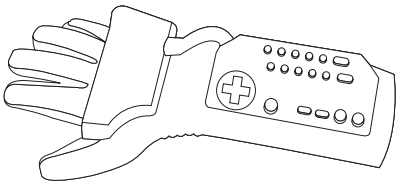
El Servo No Gira
Aún con cables de colores es sorprendentemente fácil conectar un servo al revés. Este podría ser el problema.

El Servo no se Mueve como se Espera
El sensor está diseñado para funcionar en una sola dirección. Prueba doblarlo hacia el otro lado (donde la cara rallada quede hacia afuera haciendo una curva convexa).

El Servo no se Mueve muy Lejos
Necesitas modificar el rango de valores en la llamada a la función map().

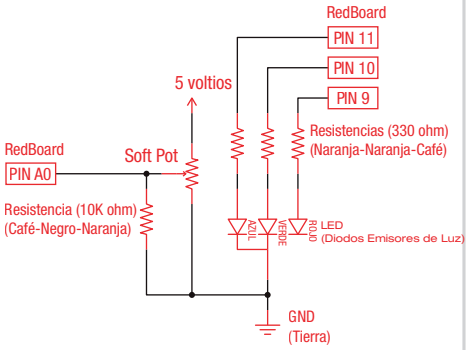
Aplicación en la vida real:

Accesorios controladores para consolas de videojuegos como el "Power Glove" de Nintendo usan tecnología sensible a flexibilidad. Este fue el primer control de videojuego que intentó simular el movimiento de la mano en una pantalla en tiempo real.



10

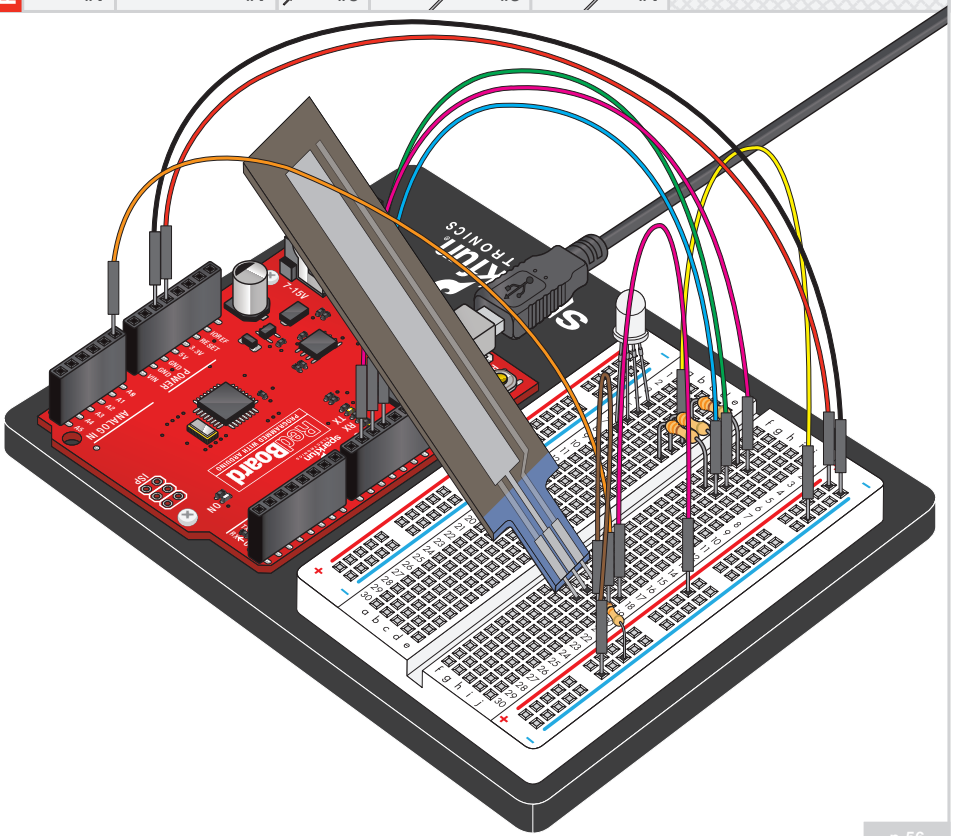
Potenciómetro Suave



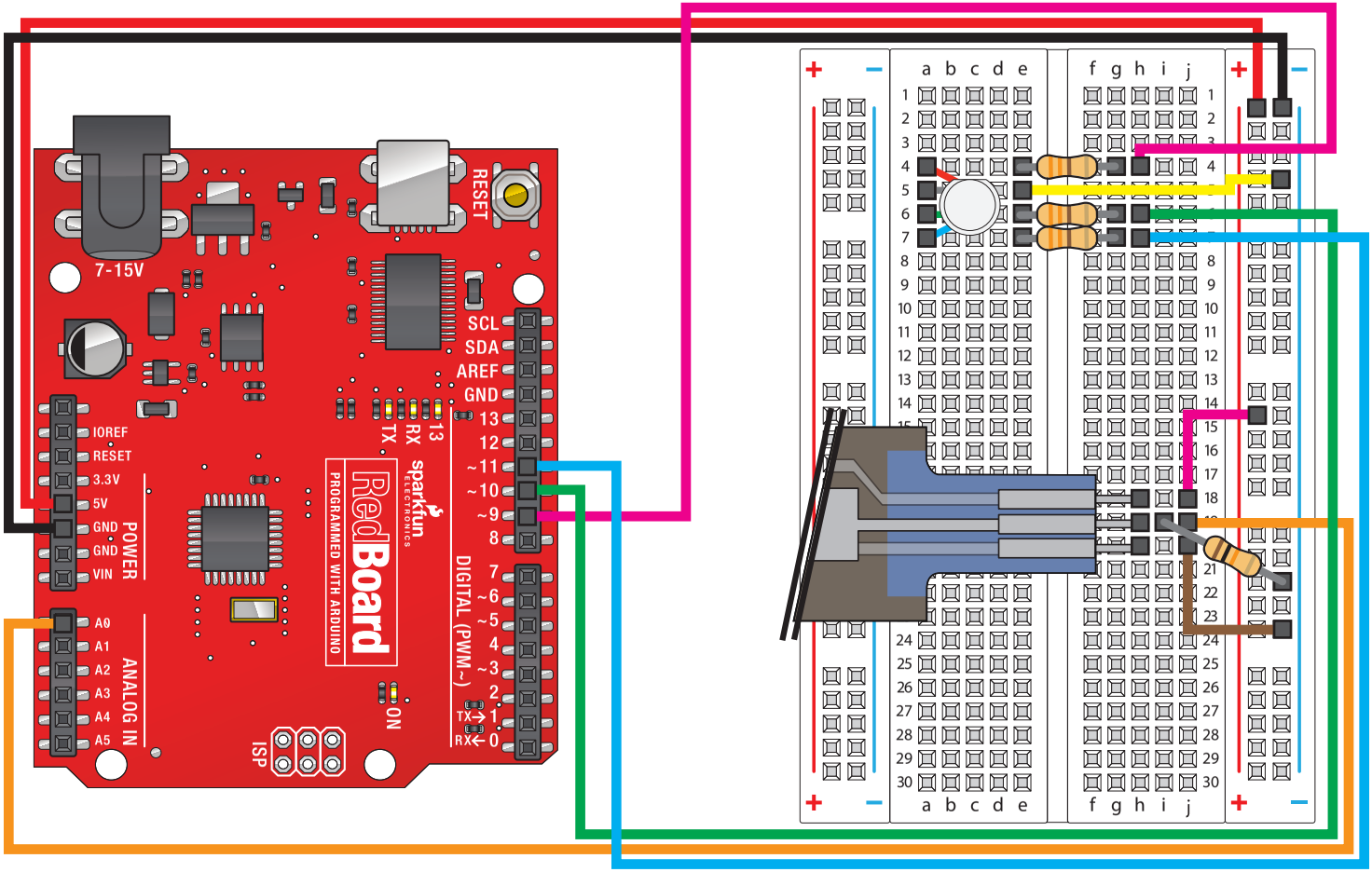
En este circuito vamos a usar otro tipo de resistencia variable – esta vez es un potenciómetro suave (o soft pot). Este está constituido por una banda delgada y flexible que puede detectar dónde se le está aplicando presión. Al presionar en varias partes de la banda puedes variar la resistencia de 100 a 10Kohmios. Puedes usar esta habilidad para rastrear movimiento en el soft pot o simplemente utilizarlo como un botón. En este circuito conectaremos el potenciómetro suave y lo usaremos para controlar un LED RGB.

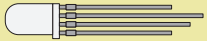
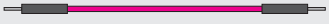
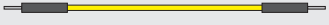
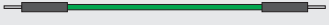
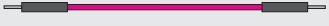
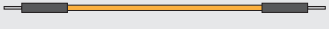
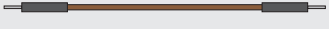
PARTES:

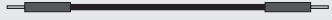
LED	Potenciómetro Suave	Cable	Resistencia de 330Ω	Resistencia de 10KΩ
x1	x1	x9	x3	x1



Circuito 10: Potenciómetro Suave



Componente:	Imagen de Referencia:		
LED RGB (5mm) ⚠			a4 a5 a6 a7
Potenciómetro Suave			h18 h19 h20
Resistencia de 330Ω			e4 - g4
Resistencia de 330Ω			e6 - g6
Resistencia de 330Ω			e7 - g7
Resistencia de 10KΩ			i19 - -
Cable Conector		Pin 9	h4
Cable Conector			e5 - -
Cable Conector		Pin 10	h6
Cable Conector		Pin 11	h7
Cable Conector			j18 - +
Cable Conector		A0	j19
Cable Conector			j20 - -

Componente:	Imagen de Referencia:		
Cable Conector		5V	+
Cable Conector		GND	-

10 Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 10

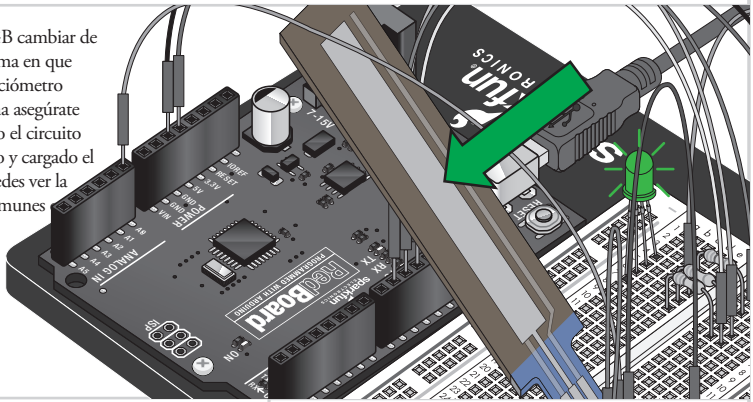
Notas de Código:

```
redValue = constrain(map(RGBposition, 0, 341, 255, 0), 0, 255)
 + constrain(map(RGBposition, 682, 1023, 0, 255), 0, 255);

greenValue = constrain(map(RGBposition, 0, 341, 0, 255), 0, 255)
 - constrain(map(RGBposition, 341, 682, 0,255), 0, 255);

blueValue = constrain(map(RGBposition, 341, 682, 0, 255), 0, 255)
- constrain(map(RGBposition, 682, 1023, 0, 255), 0, 255);
```

Estas funciones grandes y tenebrosas toman un solo valor (RGBposition) y calculan los tres valores RGB necesarios para crear un arcoíris de colores. Estas funciones crean tres "picos" para los valores de rojo, verde y azul, los cuales se enlazan para mezclar y crear nuevos colores. ¡Mira el código para más información! Aún si no estás 100% seguro de cómo funciona, puedes copiar y pegar esta (o cualquier) función en tu código y usarla tú mismo. Si quieres saber más acerca de crear tus propias funciones – echa un vistazo al circuito #11.

Lo que deberías ver:

Deberías ver el LED RGB cambiar de color de acuerdo a la forma en que interactúas con tu potenciómetro suave. Si esto no funciona asegúrate de que hayas ensamblado el circuito correctamente, verificado y cargado el código a tu tarjeta, o puedes ver la sección de problemas comunes muestra abajo.



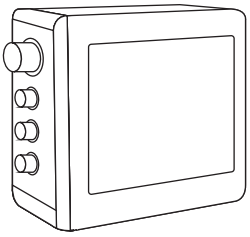
Problemas comunes:

El LED Permanece Oscuro o Muestra un Color Incorrecto
Con los cuatro pines del LED posicionados tan cerca unos de otros, a veces es fácil posicionar uno de manera incorrecta. Revisa que cada pon esté colocado donde debe ser.

Resultado Extraños
La causa más probable es que estés presionando el potenciómetro en más de una posición. Esto es normal y puede ser utilizado para crear resultados estupendos.

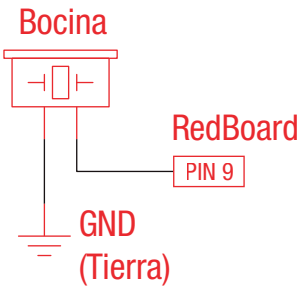
Aplicación en la vida real:

Las perillas en muchos objetos, por ejemplo en un radio, usan conceptos similares al que acabas de completar para este circuito.



11

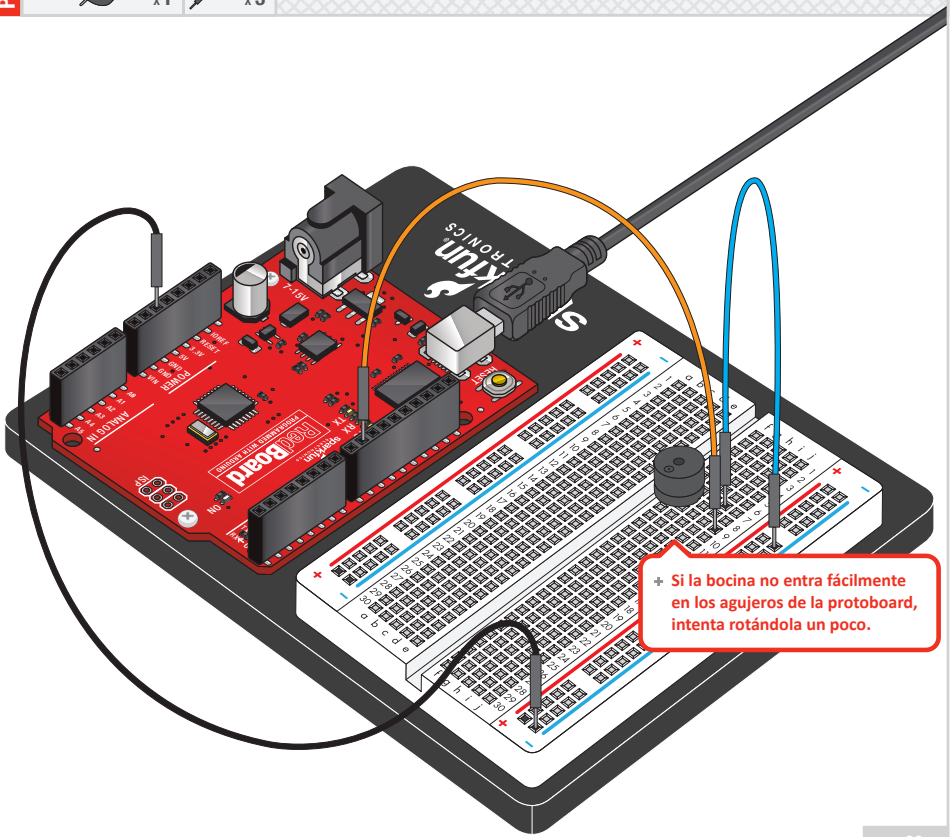
Bocina



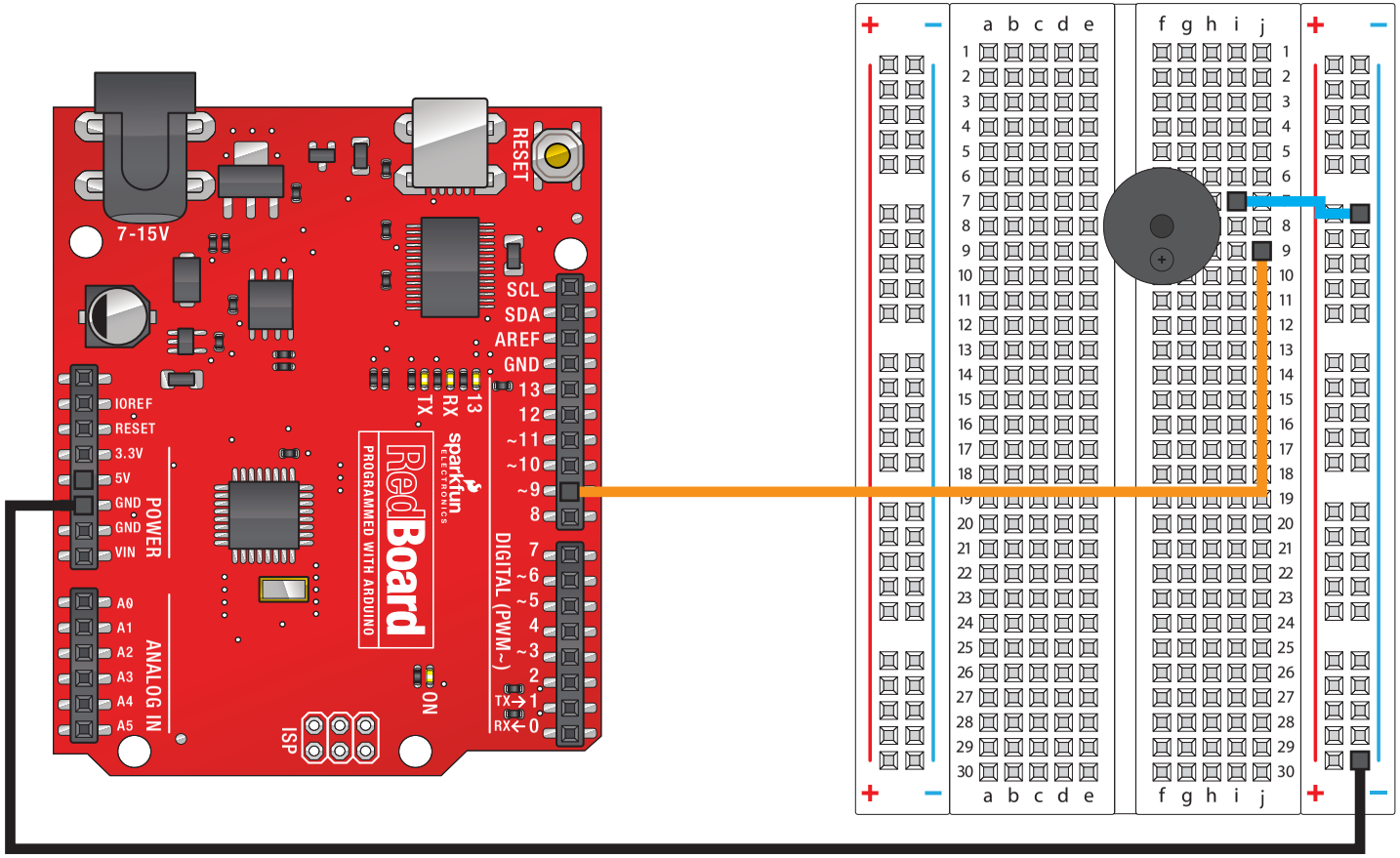
En este circuito haremos de nuevo un puente entre el mundo digital y el mundo analógico. Estaremos usando una bocina que hace un pequeño "clic" cuando le aplicas voltaje (¡pruébalo!). Por sí solo no es extremadamente excitante, pero cuando enciendes y apagas el voltaje cientos de veces en un segundo, la bocina producirá un tono. ¡Y si unes unos cuantos tonos uno tras otro, tendrás música! Este circuito y diseño jugarán con un tono clásico. ¡Nunca te vamos a decepcionar!

PARTES:

Bocina	Cable
 x1	 x3



Circuito 11: Bocina



Componente:	Imagen de Referencia:		
Bocina			f9 (+) f7 (-)
Cable Conector			i7 - -
Cable Conector		Pin 9	j9
Cable Conector		GND	-

Creando tus propias funciones:

Arduino contiene una variedad de funciones incluidas que son útiles para todo tipo de cosas (visita **http://arduino.cc/en/reference** para ver la lista). Pero también puedes crear tus propias funciones fácilmente. Primero, necesitamos declarar una función. Aquí tenemos un ejemplo simple llamado “suma”, la cual suma dos números y retorna el resultado. Vamos a explicarla bien.

```
int add(int parameter1, int parameter2)
{
        int x;

        x = parameter1 + parameter2;

        return(x);
}
```

Tus funciones pueden recibir valores ("parametros") y retornar un valor como esta lo hace.

Nota: el lenguaje de programación de Arduino no admite tíldes por lo que debes procurar no usarlas o el programa mostrará un error de sintaxis.

Si vas a enviar parámetros a tun función, colócalos (y a sus tipos) en el paréntesis que va luego del nombre de la función. Si tu función no usa ningún parámetro, tan solo deja el paréntesis vacío () luego del nombre.

Si tu función retorna un valor como resultado, escribe el tipo del valor de retorno en frente del nombre de la función. Luego, en tu función, cuando estés listo para retornar el valor, escribe una declaración de return(valor). Si no vas a retornar ningún valor, coloca “void” al frente del nombre de la función (similar a la declaración de las funciones setup() y loop()).

Cuando escribes tus propias funciones haces tu código más nítido y fácil de reutilizar. Visita **http://arduino.cc/en/Reference/FunctionDeclaration** para más información acerca de las funciones.

11

Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 11

Notas de Código:

```
char notes[] = "cdfda ag cdfdg gf ";

char names[] = {'c','d','e','f','g','a','b','C'};
```

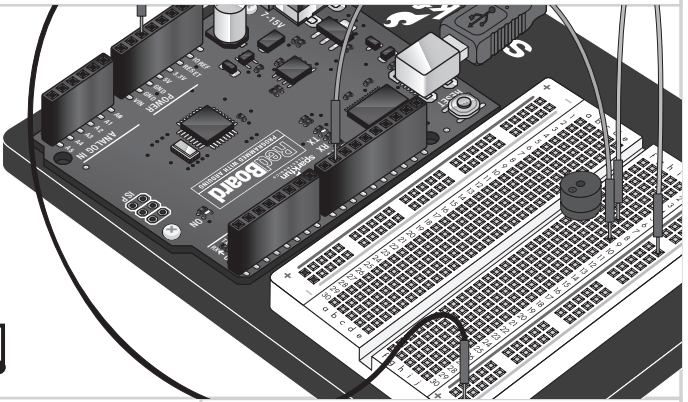
Hasta ahora hemos estado trabajando únicamente con datos numéricos, pero el Arduino también puede trabajar con texto. Los caracteres (sencillos, imprimibles, letras, números y otros símbolos) tienen su propio tipo, llamado “char”. Cuando tienes un arreglo de caracteres puedes definirlo entre comillas dobles (también llamado un “string”), O como una lista de caracteres entre comillas simples.

```
tone(pin, frecuencia, duracion);
```

Uno de los comandos propios de Arduino más útiles es la función tone(). Esta función maneja un pin de salida a cierta frecuencia, haciéndolo perfecto para controlar bocinas y altavoces. Si le das una duración (en milisegundos), reproducirá el tono y luego se detendrá. Si no le das una duración se mantendrá reproduciendo el tono para siempre (pero puedes detenerlo con otra función, noTone()).

Lo que deberías ver:

Deberías ver – bueno, ¡nada! Pero deberías ser capaz de escuchar una canción. Si esto no sucede asegúrate de que hayas ensamblado el circuito correctamente, verificado y cargado el código a tu tarjeta, o puedes ver la sección de problemas comunes que se muestra abajo.



Problemas comunes:

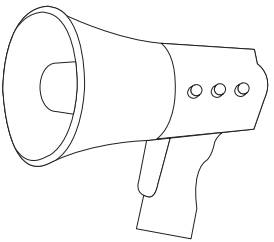
No hay Sonido
Dada la forma y tamaño del elemento piezoeléctrico es fácil errar los agujeros en la protoboard. Revisa que esté colocado de forma correcta.

No Puedo Pensar Mientras la Melodía está Sonando
Tan solo retira el elemento piezoeléctrico mientras piensas, carga tun programa y luego conéctalo de nuevo.

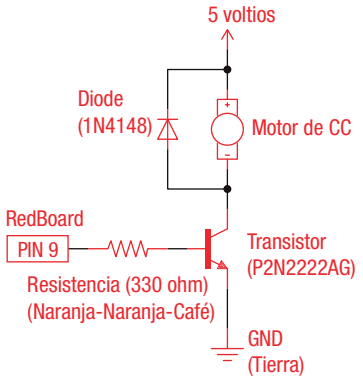
Te Sientes Decepcionado y Desolado
Este código está escrito para que puedas añadir fácilmente tus propias canciones.

Aplicación en la vida real:

Muchos megáfonos modernos tienen configuraciones que usan una bocina amplificadora. Generalmente son muy ruidosas y muy buenas para atraer la atención de la gente.



12 Rotando un Motor

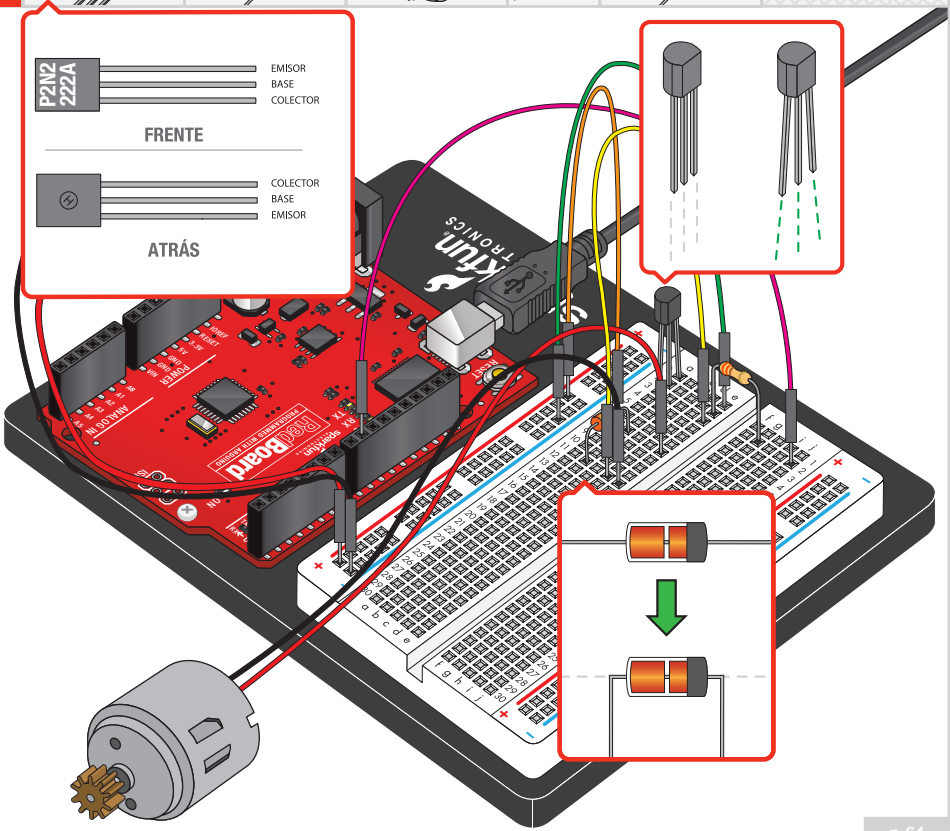


¿Recuerdas cuando jugaste con un motor servo? Ahora vamos a pasar a hacer un motor girar. Esto requiere el uso de un transistor, el cual puede entregar una mayor cantidad de corriente de lo que puede la RedBoard. Cuando usamos un transistor, debemos asegurarnos de que sus especificaciones máximas son suficientemente altas para tu uso. El transistor que vamos a usar para este circuito tiene valores máximos de 40V y 200miliamperios - ¡perfecto para tu motor de juguete! Cuando el motor esté girando y es apagado de repente, el campo magnético dentro de él colapsa, generando un pico de voltaje. Esto puede dañar el transistor. Para evitar que eso ocurra, usamos un "diodo de retorno", el cual desvía el pico de voltaje alrededor del transistor.

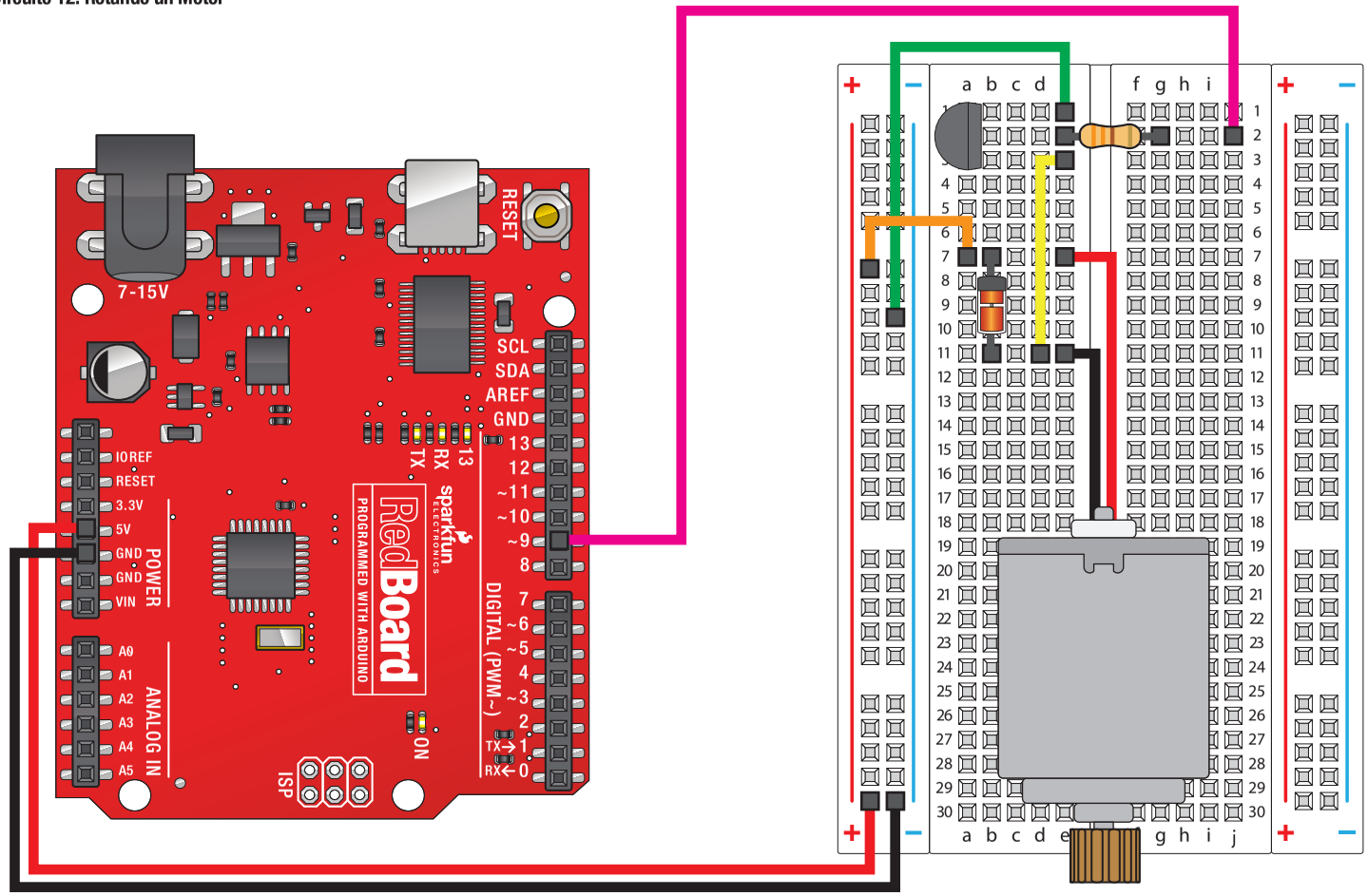
! Cuando estás construyendo el circuito ten cuidado de no confundir el transistor con el sensor de temperatura, pues son casi idénticos. Busca la etiqueta "P2N2222A" en el cuerpo del transistor.

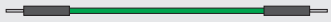
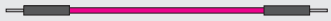
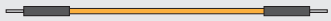
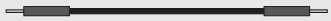
PARTS:

Transistor	Diodo	Motor de CC	Cable	Resistencia de 330Ω
P2N2222AG x1	1N4148 x1	x1	x6	x1



Circuito 12: Rotando un Motor



Componente:	Imagen de Referencia:		
Transistor P2N2222AG			a1 (E) - a2 (B) - a3 (C)
Diodo 1N4148			b7 - b11
Motor de CC			e7 - e11
Reistencia de 330Ω			e2 - g2
Cable Conector			e1 - -
Cable Conector		Pin 9	j2
Cable Conector			a7 - +
Cable Conector			e3 - d11
Cable Conector		5V	+
Cable Conector		GND	-

Armándolo todo:

En este punto probablemente estés empezando a tener tus propias ideas de circuitos que hagan cosas divertidas, o ayuden a resolver un problema real. ¡Excelente! Aquí tienes algunos consejos acerca de programación en general.

La mayoría de los diseños que escribas serán un ciclo de algunos o todos estos pasos:

1. Reciben alguna clase de entrada
2. Realizan algún tipo de cálculos o decisiones
3. Entregan alguna clase de salida
4. ¡Todo se repite! (¡O no!)

Ya te hemos enseñado a usar un puñado de diferentes sensores de entrada y dispositivos de salida (y aún nos faltan unos cuantos). Siéntete libre de hacer uso de los ejemplos en tus propios diseños – esta es la principal idea detrás del movimiento del "Software Libre" o "Open Source".

Generalmente es muy fácil colocar porciones de diferentes diseños juntas, tan solo ábrelas en dos ventanas, copia y pega lo que necesites entre ellas. Esta es una de las razones por las que estamos promoviendo "buenos hábitos de programación". Cosas que usen constantes para referenciar números de pin, y organizar tus diseños en funciones, hacen más fácil poder reutilizar tu código en nuevos diseños. Por ejemplo, si unes dos porciones de código que usan el mismo pin, puedes cambiar fácilmente una de las constantes a un nuevo pin. (No olvides que no todos los pones soportan la función **analogWrite();** los pines compatibles están marcados en tu tarjeta).

Si necesita ayuda, existen foros en internet donde puedes hacer preguntas. Prueba el foro de Arduino en arduino.cc/forum, y el foro de SparkFun en **forum.sparkfun.com**. Cuando estés listo para pasar a temas más avanzados, echa un vistazo a la página de tutoriales de Arduino en **arduino.cc/es/tutorial**. Muchos de los productos más avanzados de SparkFun son programados con Arduino, (permitiéndote modificarlos fácilmente), o tener ejemplos de Arduino para ellos. Mira nuestras páginas de productos para más información.

Finalmente cuando crees algo realmente genial, considera compartirlo con el mundo para que otros puedan aprender de tu ingenio. ¡Asegúrate de hacérnoslo saber en **https://www.sparkfun.com/project_calls** para que podamos ponerlo en nuestra página principal!

12

Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 12

Notas de Código:

while (Serial.available() > 0)

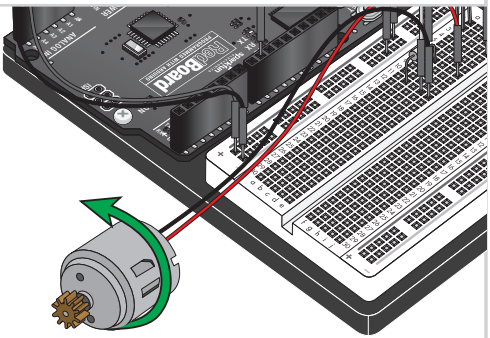
El puerto serial de la RedBoard puede ser usado tanto para recibir como para enviar datos. Ya que los datos pueden llegar en cualquier momento, la RedBoard guarda los datos entrantes en el puerto hasta que estés listo para usarlos. El comando Serial.available() retorna el número de caracteres que el puerto ha recibido, pero no han sido usados por tu diseño aún. Cero significa que no ha llegado ningún dato.

speed = Serial.parseInt();

Si los datos del puerto han estado esperándote, hay varias formas para que los puedas usar. Ya que estamos escribiendo números en el puerto, podemos usar el comando Serial.parseInt() para extraer o "parsear" números enteros de los caracteres recibidos. Si escribes "1" "0" "0" en el puerto, esta función retornará el número 100.

Lo que deberías ver:

El motor de CC debería girar si haz ensamblado los componentes del circuito correctamente y haz verificado/cargado el código correcto. Si el circuito no funciona revisa la sección de problemas comunes que se muestra más abajo.



Problemas comunes:

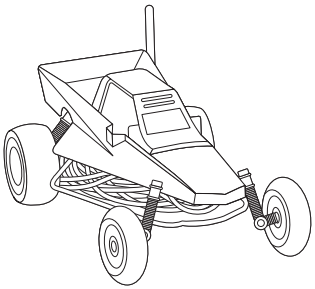
El Motor No Gira
Si estás usando un transistor diferente al incluido en el kit, revisa en la hoja de datos que sus diagrama de pines sea compatible con el del P2N2222AG (muchos se encuentran al revés).

Aún no hay Suerte
Si usaste tu propio motor, revisa que este funcione con 5 voltios y que este no consuma mucha potencia.

Aún No Funciona
Algunas veces la RedBoard se desconectará de la computadora. Intenta desconectar y reconectarla a tu puerto USB.

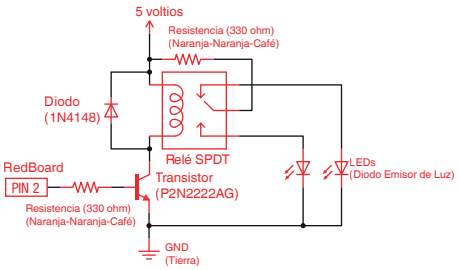
Aplicación en la vida real:

Los carros a control remoto usan motores de Corriente Continua (CC) para hacer girar sus ruedas.



13

Relés

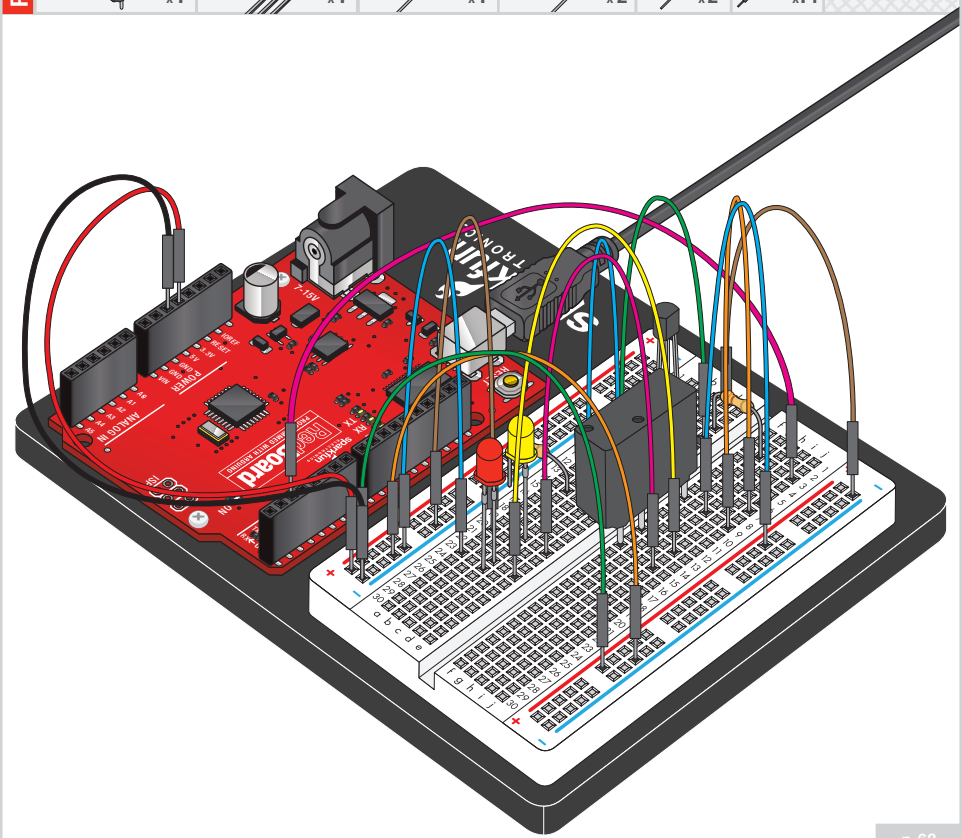


En este circuito vamos a usar algunas de las lecciones que aprendimos en el circuito 12 para controlar un relé. Un relé es básicamente un interruptor mecánico controlado electrónicamente. Dentro de esa caja plástica de apariencia inofensiva hay un dispositivo electromagnético que, cuando recibe una carga de energía, causa que se dispare un interruptor. En este circuito aprenderás como controlar un relé como los profesionales – ¡dándole a tu RedBoard habilidades aún más poderosas!

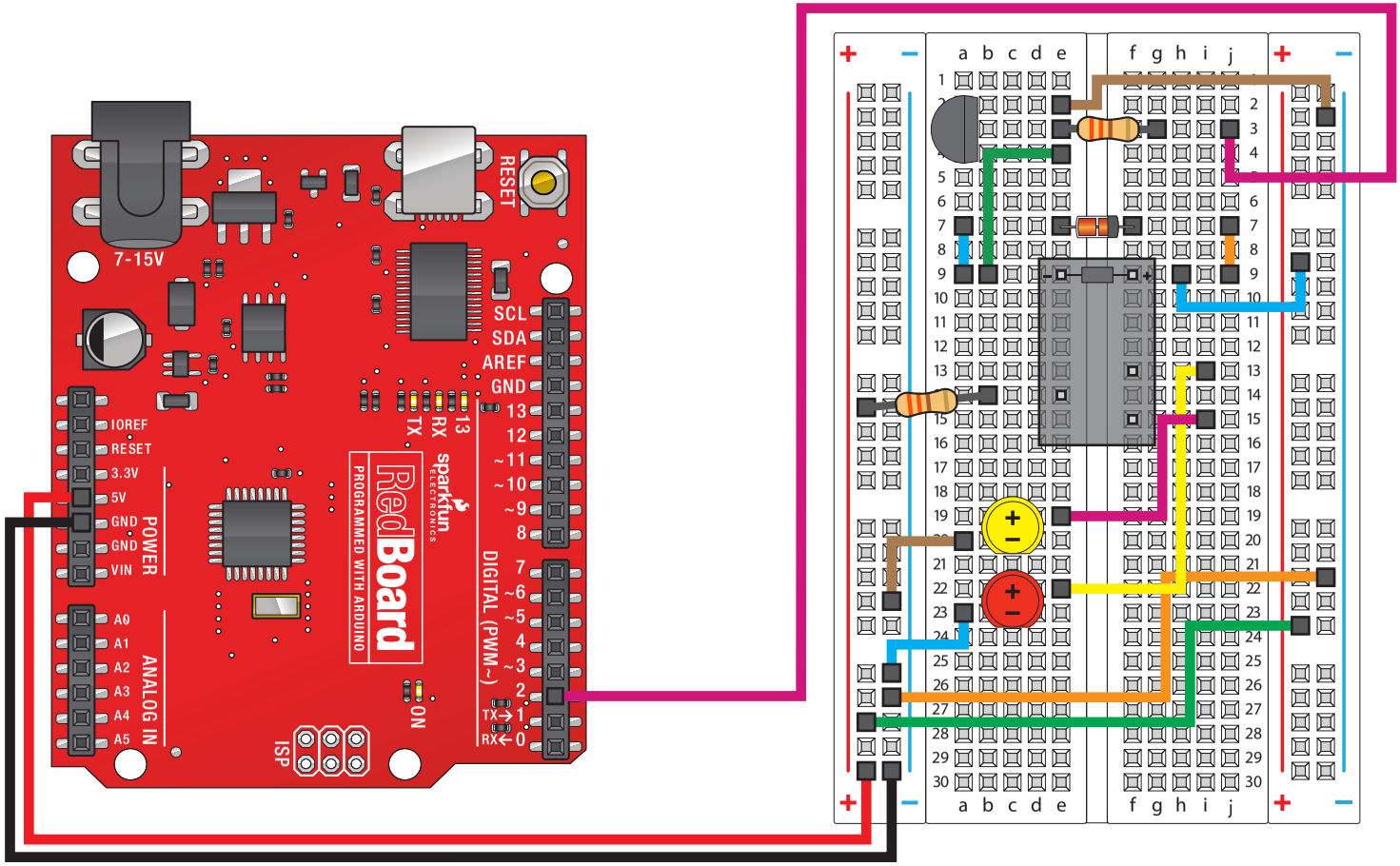
> **!** Cuando el relé está apagado, el pin COM (común) estará conectado al pin NC (Normalmente Cerrado). Cuando el relé esté encendido, el pin COM (común) estará conectado al pin NO (Normalmente Abierto).

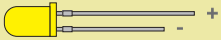
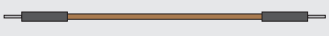
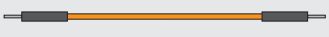
PARTES:

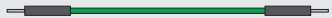
Relé	Transistor P2N2222AG	Diodo 1N4148	Resistencia de 330Ω	LED	Cable
x1	x1	x1	x2	x2	x14



Circuito 13: Relés



Componente:	Imagen de Referencia:		
Relé			e14 e9 / f15 f13 f9
Transistor P2N2222AG			a2 a3 a4 (E B C)
LED (5mm)			c19 - c20 (+ -)
LED (5mm)			c22 - c23 (+ -)
Diodo 1N4148			e7 - f7
Resistencia de 330Ω			e3 - g3
Resistencia de 330Ω			b14 - +
Cable Conector			e2 - -
Cable Conector		Pin 2	j3
Cable Conector			j7 - j9
Cable Conector			h9 - +
Cable Conector			i13 - e22
Cable Conector			i15 - e19

Componente:	Imagen de Referencia:		
Cable Conector			- -
Cable Conector			+ +
Cable Conector			a23 - -
Cable Conector			a20 - -
Cable Conector			a7 - a9
Cable Conector			e4 - b9
Cable Conector		5V	+
Cable Conector		GND	-

13

Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 13

Notas de código:

digitalWrite(relayPin, HIGH);

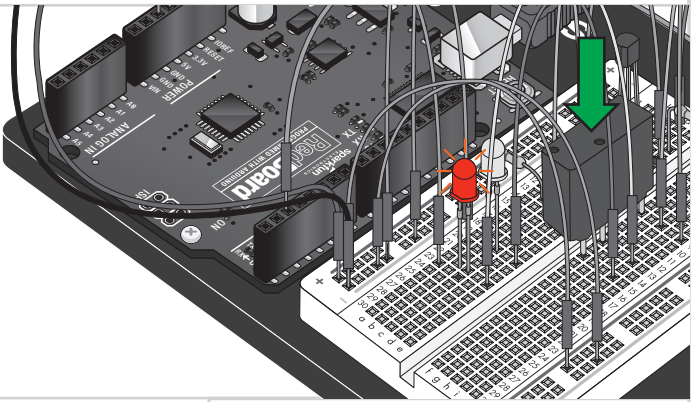
Cuando encendemos el transistor, que a su vez energiza la bobina del relé, los contactos del interruptor del relé están cerrados. Esto conecta el pin COM del relé con el pin NO (Normalmente Abierto por sus siglas en inglés). Cualquier componente que esté conectado a estos pines se encenderá. (En este caso utilizamos LEDs, pero puede ser casi cualquier cosa).

digitalWrite(relayPin, LOW);

El relé posee un contacto adicional llamado NC (Normalmente Cerrado). El pin NC se conecta al pin COM cuando el relé está apagado. Puedes usar cualquiera de estos pines dependiendo del comportamiento deseado, ya sea normalmente encendido o normalmente apagado. Además, se pueden usar ambos pines para alternar la alimentación entre dos dispositivos, similar a las luces de precaución en un paso de ferrocarril.

Lo que deberías ver:

Debes ser capaz de escuchar un clic proveniente de los contactos del relé, y ver los dos LEDs alternar su iluminación en intervalos de 1 segundo. Si esto no ocurre, revisa que el circuito esté ensamblado correctamente y esté cargado el código correcto en la tarjeta. Adicionalmente, puedes consultar la sección de problemas comunes que se muestra más abajo.



Problemas comunes:

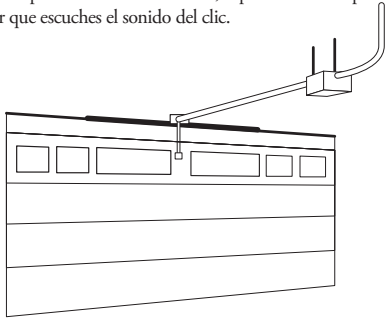
Los LEDs no se iluminan
Revisa que estos estén bien conectados. La patilla más larga (con el borde plástico no plano) es la patilla positiva.

No se escucha el sonido del clic
La sección del circuito que contiene el transistor o la bobina no está funcionando. Revisa que el transistor esté conectado de manera correcta.

El circuito no funciona del todo
Los relés incluidos en la tarjeta están diseñados para ser soldados a un circuito, no para ser utilizados en una protoboard, por lo que puede ser que necesites presionarlo un poco para asegurar que funcione bien (ocasionalmente puede salirse de nuevo). Cuando estés construyendo el circuito ten cuidado de no confundir transistor con el sensor de temperatura pues son casi idénticos.

Aplicación en la vida real :

Los portones eléctricos utilizados en los garajes usan relés para operar. Pruébalo tú mismo, si pones atención puede ser que escuches el sonido del clic.

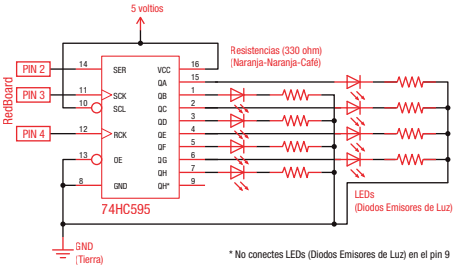


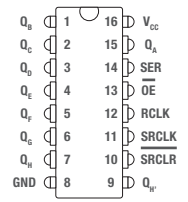
CIRCUITO #14

14

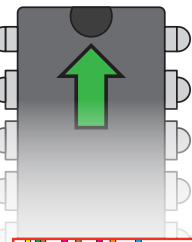
Registro de Desplazamiento

Ahora vamos a adentrarnos en el mundo de los CIs (Circuitos Integrados). En este circuito aprenderás todo acerca del uso de un registro de desplazamiento (también llamado conversor serie a paralelo). El registro de desplazamiento le dará a tu RedBoard ocho salidas adicionales, usando solamente tres pines de tu tarjeta. Para este circuito practicarás usando el registro de desplazamiento para controlar ocho LEDs.



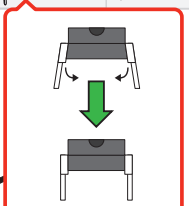


Alinea la muesca en la parte superior, en medio de "e5" y "f5" en la protoboard. La muesca indica donde se encuentra el pin 1.



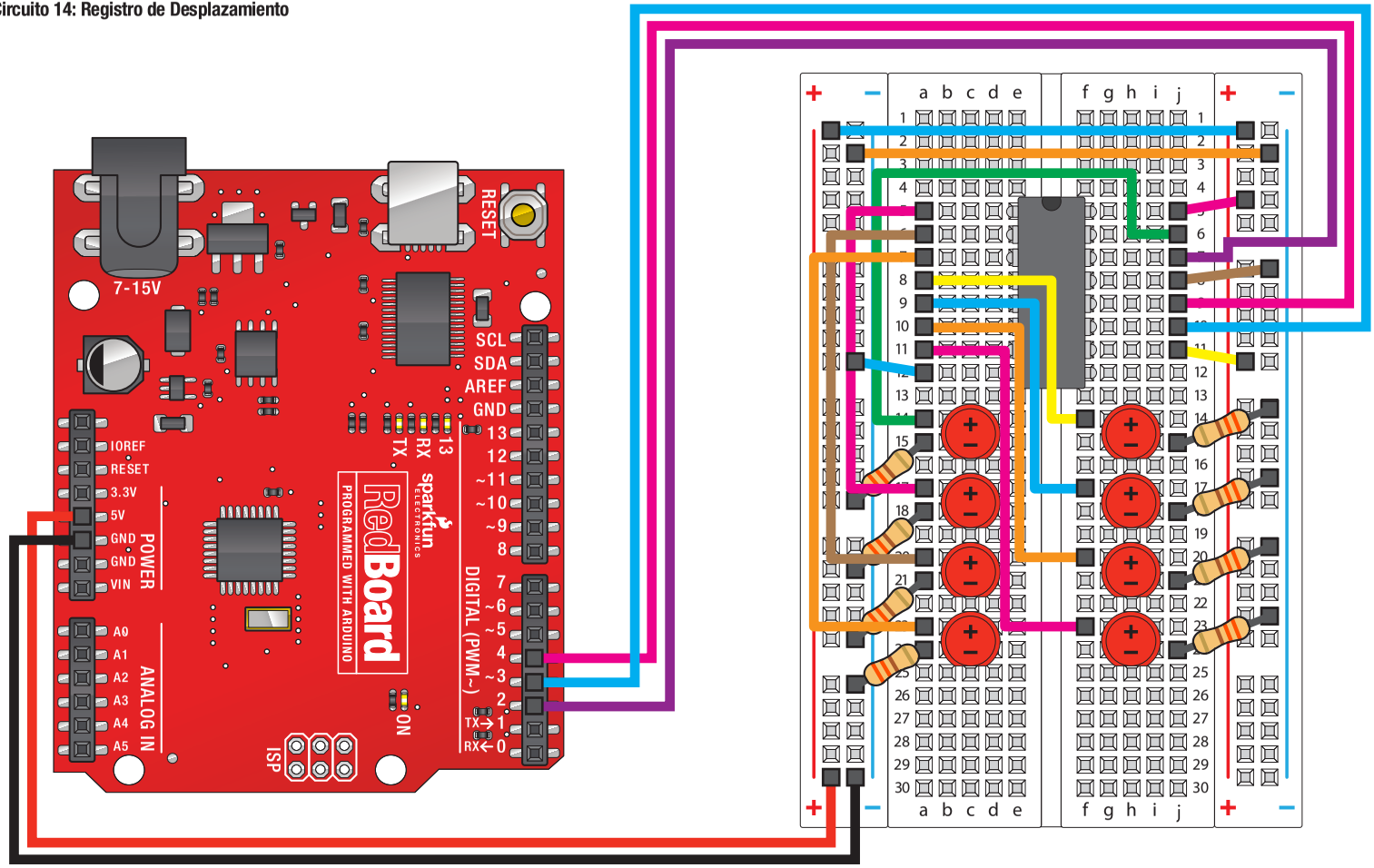
PARTES:

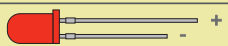
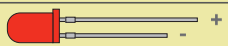
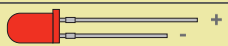
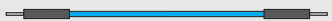
CI	LED	Resistencia de 330Ω	Cable
x1	x8	x8	x19

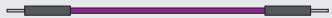
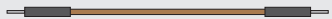
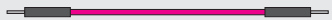
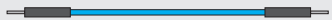
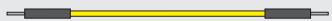
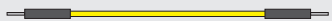
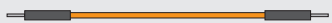
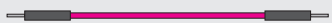
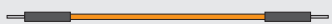
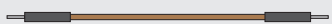
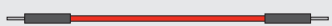


Dobla las patillas en un ángulo de 90°.

Circuito 14: Registro de Desplazamiento



Componente:	Imagen de Referencia:		
CI		e5 e6 e7 e8 e9 e10 e11 e12 f5 f6 f7 f8 f9 f10 f11 f12	
LED (5mm)		c14 c15 + -	
LED (5mm)		c17 c18 + -	
LED (5mm)		c20 c21 + -	
LED (5mm)		c23 c24 + -	
LED (5mm)		h14 h15 + -	
LED (5mm)		h17 h18 + -	
LED (5mm)		h20 h21 + -	
LED (5mm)		h23 h24 + -	
Resistencia de 330Ω		- a15	
Resistencia de 330Ω		- a18	
Resistencia de 330Ω		- a21	
Resistencia de 330Ω		- a24	
Resistencia de 330Ω		j15 -	
Resistencia de 330Ω		j18 -	
Resistencia de 330Ω		j21 -	
Resistencia de 330Ω		j24 -	
Cable Conector		+ +	

Componente:	Imagen de Referencia:	
Cable Conector		- -
Cable Conector		j5 +
Cable Conector		j6 a14
Cable Conector		Pin 2 j7
Cable Conector		j8 -
Cable Conector		Pin 4 j9
Cable Conector		Pin 3 j10
Cable Conector		j11 +
Cable Conector		f14 a8
Cable Conector		f17 a9
Cable Conector		f20 a10
Cable Conector		f23 a11
Cable Conector		a23 a7
Cable Conector		a20 a6
Cable Conector		a17 a5
Cable Conector		5V +
Cable Conector		GND -

14 Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 14

Notas de código:

shiftOut(datapin, clockpin, MSBFIRST, data);

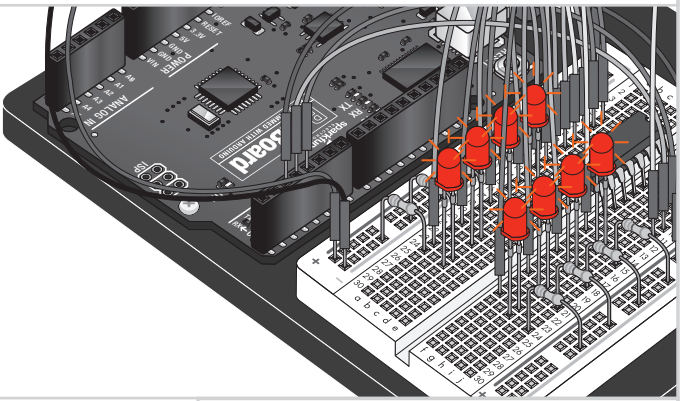
Te comunicarás con el registro de desplazamiento (y con muchos otros componentes) usando una interfaz llamada SPI, o "Serial Peripheral Interface". Esta interfaz utiliza líneas separadas de datos y de reloj que trabajan juntas para mover datos desde y hacia la RedBoard a alta velocidad. El parámetro MSBFIRST especifica el orden en que se envían los bits individuales, este caso se envía el Bit Mas Significativo de primero.

bitWrite(data, desiredPin, desiredState);

Los bits son la porción de memoria más pequeña en una computadora; cada uno puede guardar ya sea un "1" o un "0". Números más grandes son guardados como arreglos de bits. Algunas veces queremos manipular estos bits directamente, por ejemplo ahora, estamos enviando ocho bits al registro de desplazamiento y queremos que se conviertan en 1 o 0 para encender o apagar los LEDs. La RedBoard tiene diversos comandos, como bitWrite(), que hacen esto fácil de realizar.

Lo que deberías ver:

Deberías ver los LEDs encenderse de forma similar al circuito 4 (pero esta vez, estás usando un registro de desplazamiento). Si no lo hacen asegúrate de haber ensamblado el circuito correctamente, verificado y cargado el código en tu tarjeta. Mira la sección de problemas comunes más abajo.



Problemas comunes:

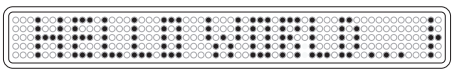
El LED de alimentación de la RedBoard se Apaga
Esto nos sucedió en un par de ocasiones, sucede cuando el chip es insertado al revés. Si lo arreglas rápidamente nada saldrá dañado.

No Funciona del Todo
Sentimos sonar como un disco rayado pero esto se debe probablemente a un cruce de cables.

Frustración
Envíanos un correo electrónico, este circuito es simple y complejo al mismo tiempo. Queremos escuchar los problemas que tienes para poder dirigirnos a ellos en ediciones futuras: **techsupport@sparkfun.com**

Aplicación en la vida real:

De forma similar al circuito #4, un letrero de figuras cambiantes muestra un mensaje con múltiples LEDs. Esencialmente el registro de desplazamiento realiza la misma tarea en este circuito #14.

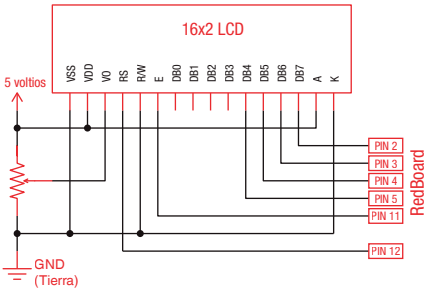


CIRCUITO #15

15

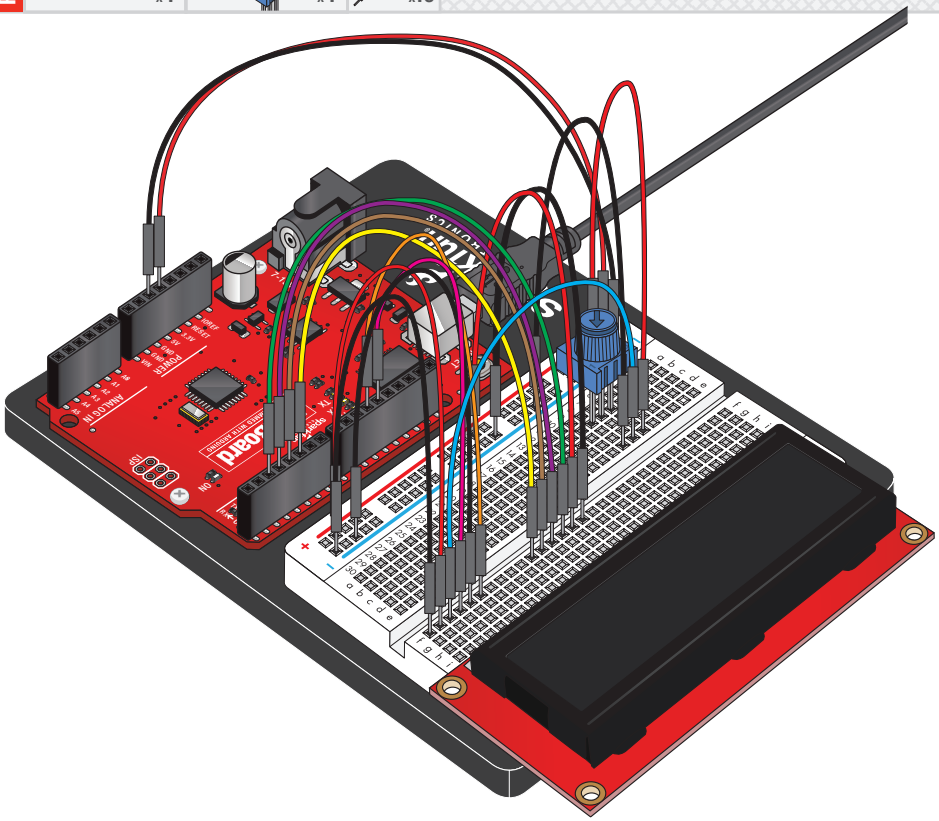
LCD

En este circuito aprenderás acerca de cómo usar un LCD. Un LCD, o visualizador de cristal líquido, es una simple pantalla en donde puedes visualizar comandos, bits de información o lecturas de tu sensor – todo dependiendo de la forma en que programes tu tarjeta. En este circuito aprenderás lo básico para incorporar un LCD en tu proyecto.

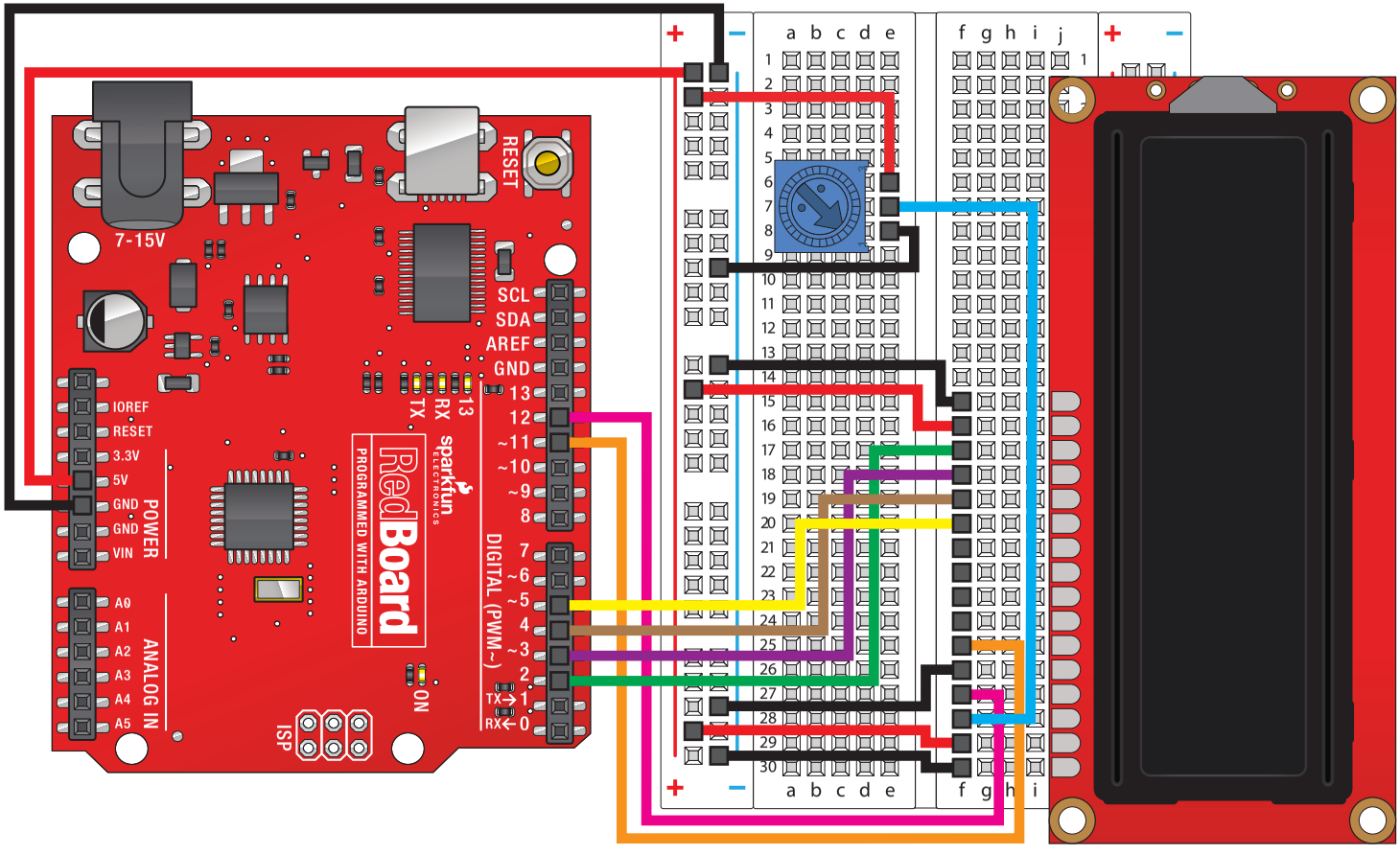


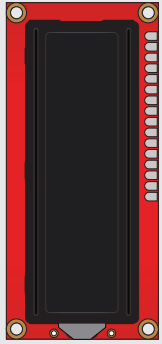
PARTES:

LCD	Potenciómetro	Cable
 x1	 x1	 x16



Circuito 15: LCD



Componente:	Imagen de Referencia:		
LCD			j30 j29 j28 j27 j26 j25 j24 j23 j22 j21 j20 j19 j18 j17 j16 j15
Potenciómetro			b8 b7 b6
Cable Conector		5V	+
Cable Conector		GND	-
Cable Conector			e6 +
Cable Conector			e7 f28
Cable Conector			e8 -
Cable Conector			f15 -
Cable Conector			f16 +
Cable Conector		Pin 2	f17

Componente:	Imagen de Referencia:		
Cable Conector		Pin 3	f18
Cable Conector		Pin 4	f19
Cable Conector		Pin 5	f20
Cable Conector		Pin 11	f25
Cable Conector			f26 -
Cable Conector		Pin 12	f27
Cable Conector			f29 +
Cable Conector			f30 -

15

Código de Arduino:

Open Arduino IDE // Archivo > Ejemplos > SIK Guide > Circuit # 15

Notas de código:

`#include <LiquidCrystal.h>` ⇨

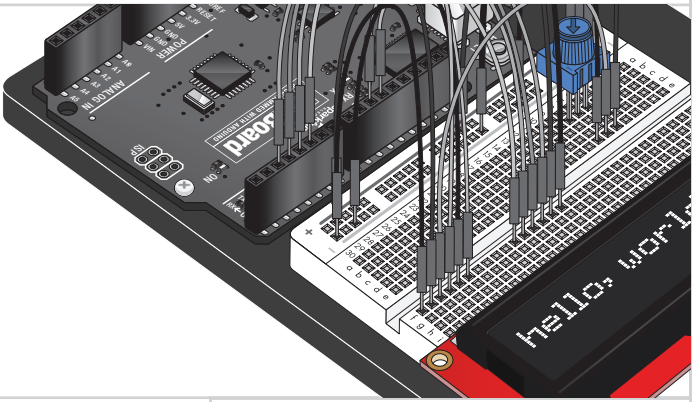
Este bit de código le dice a tu Arduino IDE que incluya la biblioteca para una pantalla LCD simple. ¡Sin esto ninguno de los comandos funcionará, así que asegúrate de incluirlo!

`lcd.print("hello, world!");` ⇨

Esta es la primera vez que encenderás algo en tu pantalla. Debes ajustar el contraste para hacerlo visible. ¡Gira el potenciómetro hasta que puedas ver el texto claramente!

Lo que deberías ver:

Inicialmente, debes ver las palabras "hello, world!" aparecer en tu LCD. Recuerda que puedes ajustar el contraste usando el potenciómetro si no puedes ver las palabras claramente. Si tienes algún problema asegúrate de que el código sea correcto y revisa tus conexiones.



Problemas comunes:

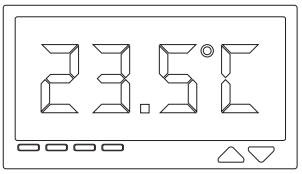
¿La pantalla está en blanco o completamente encendida?
Juega con el contraste girando el potenciómetro. Si está ajustado incorrectamente no serás capaz de leer el texto.

¿No Funciona del Todo?
Revisa de nuevo el código, específicamente chequea que hayas incluido la biblioteca LCD.

La Pantalla está Parpadeando
Revisa tus conexiones en la protoboard y el Arduino.

Aplicación en la vida real:

¡Los LCD están en todas partes! Desde LCDs avanzados como tu televisor, hasta simples pantallas de notificación, este es un visualizador muy común y útil.



Para aprender más

Visítanos online:

Este es solo el inicio de tu exploración en el mundo de la programación y los circuitos embebidos. Nuestro sitio web tiene gran variedad de tutoriales para saciar tu apetito de conocimiento. Además tenemos una comunidad de hackers, programadores, ingenieros y otros usuarios de nuestros productos en nuestros foros. ¡Así que ingresa a nuestra página web para más información acerca de Arduino, o para planear tu próximo proyecto!

sparkfun.com

NOTAS:

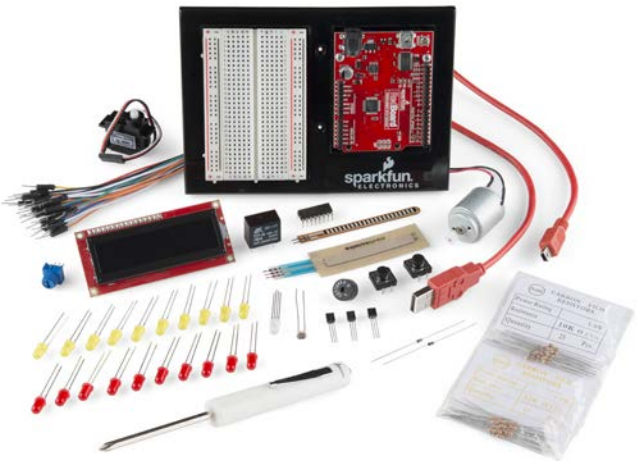
Inicia tu Viaje en el Mundo de la Electrónica

¡Este kit te guiará a través de experimentos de variadas dificultades mientras aprendes todo acerca de los sistemas embebidos, computación física, programación y más! Este kit es perfecto para cualquiera que quiera explorar el poder de la plataforma RedBoard.

Además aprenderás a ensamblar físicamente 15 circuitos electrónicos básicos, pero no se requiere ningún punto de soldadura. ¡No es necesaria ninguna experiencia previa!

El SparkFun Inventor's Kit enseña programación básica, para la cual necesitas tener tanto una computadora como una conexión a internet.

EL KIT INCLUYE



- RedBoard de SparkFun
- Protoboard
- Folleto de Instrucciones
- Relé sellado
- Servo pequeño
- LEDs rojos y amarillos
- LED RGB
- Sensor de Temperatura
- Motor CD
- Registro de desplazamiento de 8bits
- Interruptores de botón presionable
- Potenciómetro
- Fotorresistencia
- Transistores
- Cables de Unión
- Cable USB
- Diodos de señal
- Resistencias de 10K ohms
- Resistencias de 330 ohms
- Bocina piezoeléctrica
- Sensor flexible
- Potenciómetro suave
- Plataforma base
- LCD