



**TUTORIAL DE USO DEL KIT DE 37
SENSORES
PARA UNO V2.0**

V2.0.19.01.23

Prefacio

Nuestra Empresa

Fundada en 2011, Elegoo Inc. es una compañía de tecnología en pleno crecimiento que se dedica a la investigación y desarrollo, producción, y mercadeo de hardware para aplicaciones de código abierto. Ubicada en Shenzhen, el Silicon Valley de China, hemos crecido hasta llegar a tener más de 150 empleados y una fábrica de 3280 metros cuadrados.

Nuestras líneas de productos incluyen cables DuPont, tarjetas UNO R3 y kits de principiantes diseñados para que los clientes de cualquier nivel puedan aprender Arduino. Además, comercializamos accesorios para Raspberry Pi tales como pantallas táctiles de tipo TFT. Adicionalmente, tenemos planes de expandir nuestras ofertas para incluir otras tecnologías tales como productos relacionados con impresión 3D. Todos nuestros productos cumplen con los estándares de calidad internacionales y han recibido el reconocimiento positivo de nuestros clientes en distintos mercados de todo el mundo.

Página web oficial: <http://www.elegoo.com>

Tienda virtual en Amazon EEUU:

<http://www.amazon.com/shops/A2WWHQ25ENKVJ1>

Tienda virtual en Amazon Canadá:

<http://www.amazon.ca/shops/A2WWHQ25ENKVJ1>

Tienda virtual en Amazon México:

<https://www.amazon.com.mx/shops/A2WWHQ25ENKVJ1>

Tienda virtual en Amazon Reino Unido:

<http://www.amazon.co.uk/shops/A1PA6795UKMFR9>

Tienda virtual en Amazon Alemania:

<http://www.amazon.de/shops/A1PA6795UKMFR9>

Tienda virtual en Amazon Francia:

<http://www.amazon.fr/shops/A1PA6795UKMFR9>

Tienda virtual en Amazon España:

<http://www.amazon.es/shops/A1PA6795UKMFR9>

Tienda virtual en Amazon Italia:

<http://www.amazon.it/shops/A1PA6795UKMFR9>

Packing list

 www.elegoo.com



JOYSTICK



RGB LED



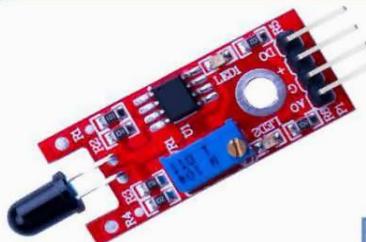
IR RECEIVER



AVOIDANCE



18B20 TEMP



FLAME



TRACKING

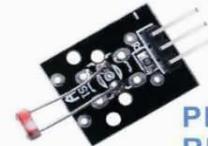
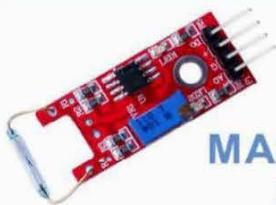


PHOTO RESISTOR

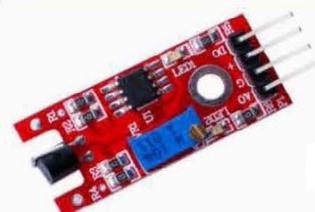


IR EMISSION

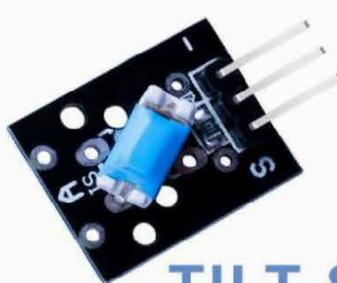
Contact us : service@elegoo.com



MAGNETIC
SPRING



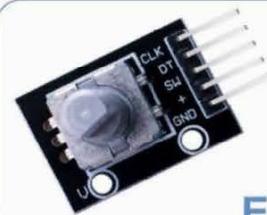
METAL
TOUCH



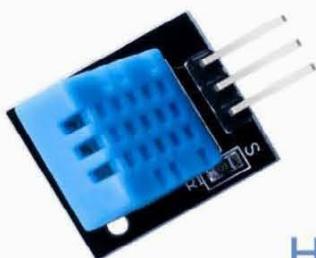
TILT-SWITCH



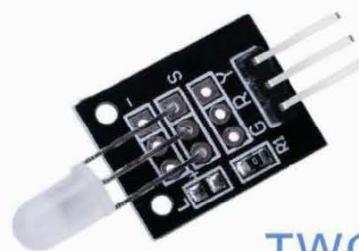
SMD
RGB



ROTARY
ENCODER

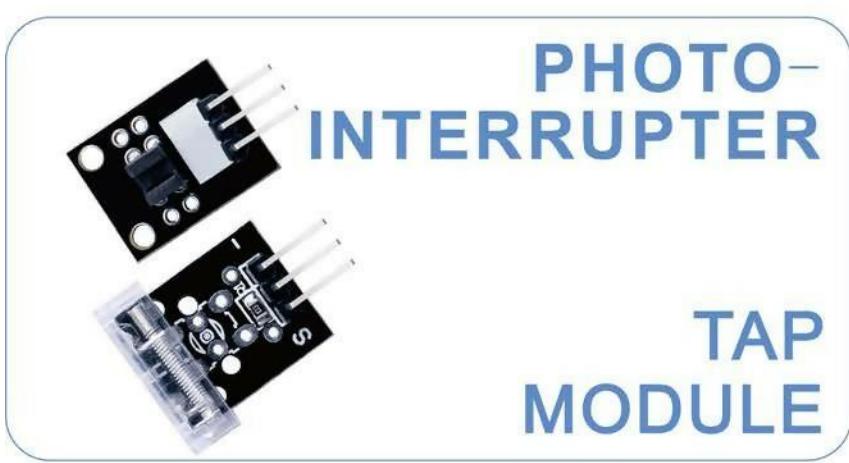


TEMP
AND
HUMIDITY

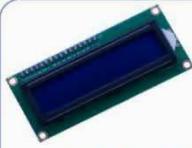


TWO-COLOR

Contact us : service@elegoo.com



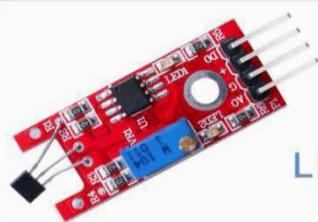
Contact us : service@elegoo.com



LCD 1602
MODULE
(WITH PIN HEADER)



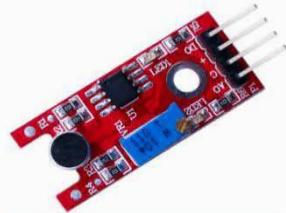
RELAY



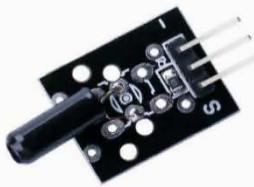
LINEAR
HALL



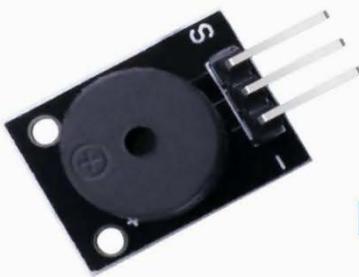
POWER
SUPPLY
MODULE



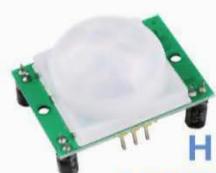
SMALL
SOUND



SHOCK



PASSIVE
BUZZER



HC-SR501 PIR
MOTION SENSOR

Contenido

| | |
|---|-----|
| Lesson 0 Installing IDE | 10 |
| Lesson 1 Add Libraries | 24 |
| Lesson 2 Open Serial Monitor | 37 |
| Lesson 3 Blink..... | 43 |
| Lesson 4 Temperature and Humidity Module | 58 |
| Lesson 5 DS18B20 Temperature Sensor Module..... | 72 |
| Lesson 6 Button Switch Module..... | 85 |
| Lesson 7 Switch Modules | 92 |
| Lesson 8 IR Receiver and Emission Module..... | 102 |
| Lesson 9 Active Buzzer Module..... | 113 |
| Lesson 10 Passive Buzzer Module..... | 121 |
| Lesson 11 Laser Module..... | 128 |
| Lesson 12 SMD RGB Module and RGB Module | 134 |
| Lesson 13 Photo-Interrupter Module | 143 |
| Lesson 14 Two Color LED Module..... | 150 |
| Lesson 15 Light Dependent Resistor Module | 156 |
| Lesson 16 Large Microphone Module and Small Microphone Module | 163 |
| Lesson 17 Reed Switch Module..... | 173 |
| Lesson 18 Digital Temperature Module | 182 |

| | |
|---|-----|
| Lesson 19 Linear Magnetic Hall Sensor Module | 195 |
| Lesson 20 Flame Sensor Module | 204 |
| Lesson 21 Touch Sensor Module | 213 |
| Lesson 22 Seven-Color Flash LED Module | 220 |
| Lesson 23 Joystick Module | 225 |
| Lesson 24 Line Tracking Module | 232 |
| Lesson 25 Obstacle Avoidance Module..... | 237 |
| Lesson 26 Rotary Encode Module | 246 |
| Lesson 27 Relay Module | 255 |
| Lesson 28 LCD Display | 261 |
| Lesson 29 Ultrasonic Sensor Module | 269 |
| Lesson 30 GY-521 Module | 277 |
| Lesson 31 HC-SR501 PIR Sensor | 289 |
| Lesson 32 Water Level Detection Sensor Module | 304 |
| Lesson 33 Real Time Clock Module | 313 |
| Lección 34 Keypad Module | 322 |

Lección 1 Instalación del IDE

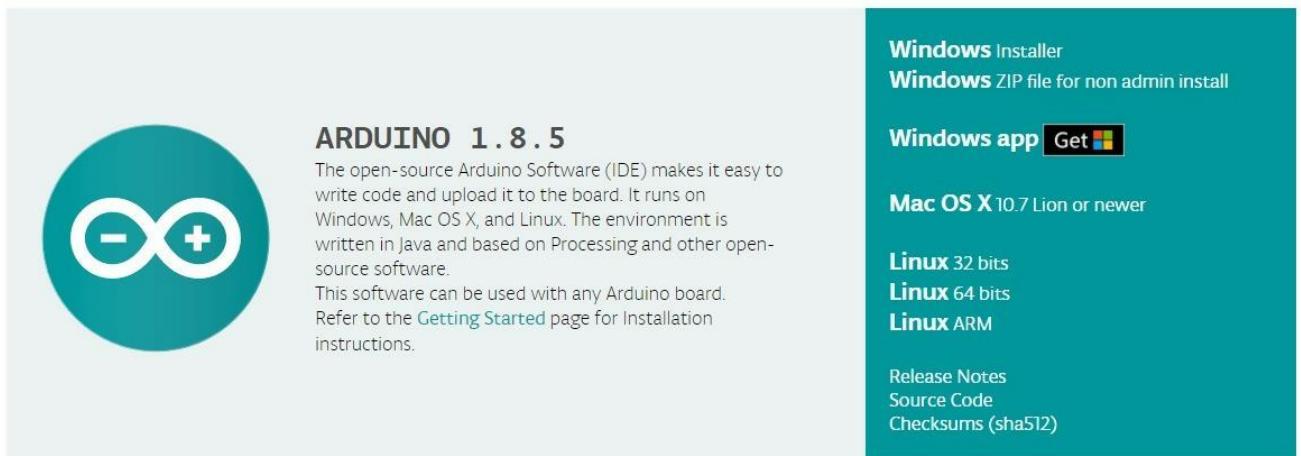
Introducción

El Entorno de Desarrollo Integrado o IDE (siglas en inglés de Integrated Development Environment) es el software que se utiliza en la plataforma de Arduino.

En esta lección, aprenderás como configurar tu computador para utilizar Arduino y poder trabajar de ahí en adelante con las demás lecciones que vienen posteriormente.

El IDE de Arduino que se utilizará para programar está disponible para Windows, Mac y Linux. El proceso de instalación es diferente para cada plataforma y se requiere cierto trabajo manual para instalar el software.

Paso 1: Descarga el IDE de Arduino desde: <https://www.arduino.cc/en/Main/Software>



La versión del IDE de Arduino disponible en esta página web es usualmente la más actualizada, por lo que es posible que el número de la misma sea diferente del que se muestra en la imagen anterior.

Paso 2: Descarga el IDE de Arduino compatible con el sistema operativo de tu computadora. Tomaremos como ejemplo el de Windows.



Haz clic en Windows Installer.

Support the Arduino Software

Consider supporting the Arduino Software by contributing to its development. (US tax payers, please note this contribution is not tax deductible). Learn more on how your contribution will be used.



Haz clic en JUST DOWNLOAD.

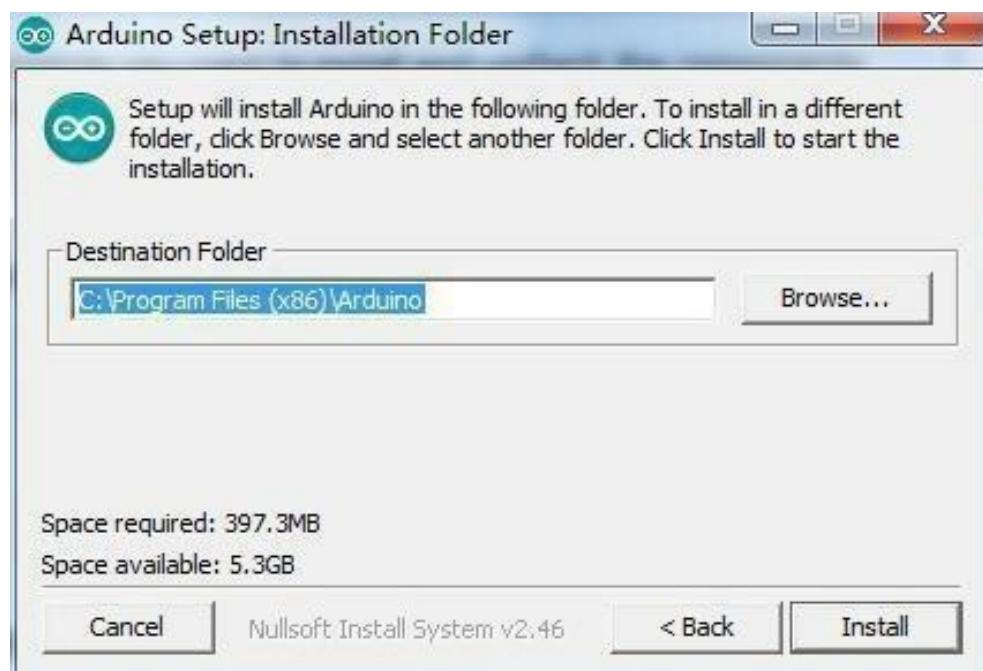
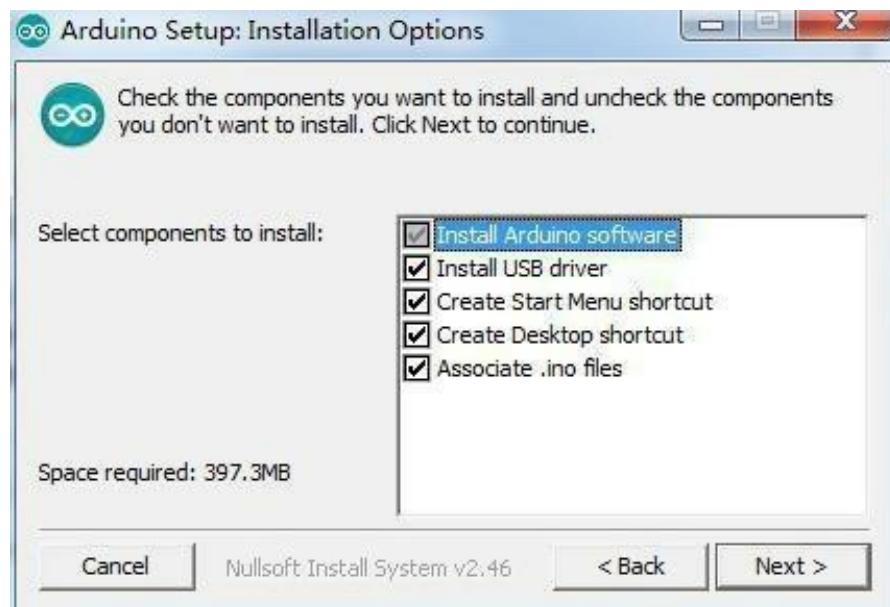
Por cierto, también puedes descargar el IDE de Arduino desde nuestra página oficial: <http://www.elegoo.com/download/>

Instalar el IDE de Arduino (para Windows)

Instala Arduino con el paquete de instalación .exe



Haz clic en / Agree para ver la siguiente interface

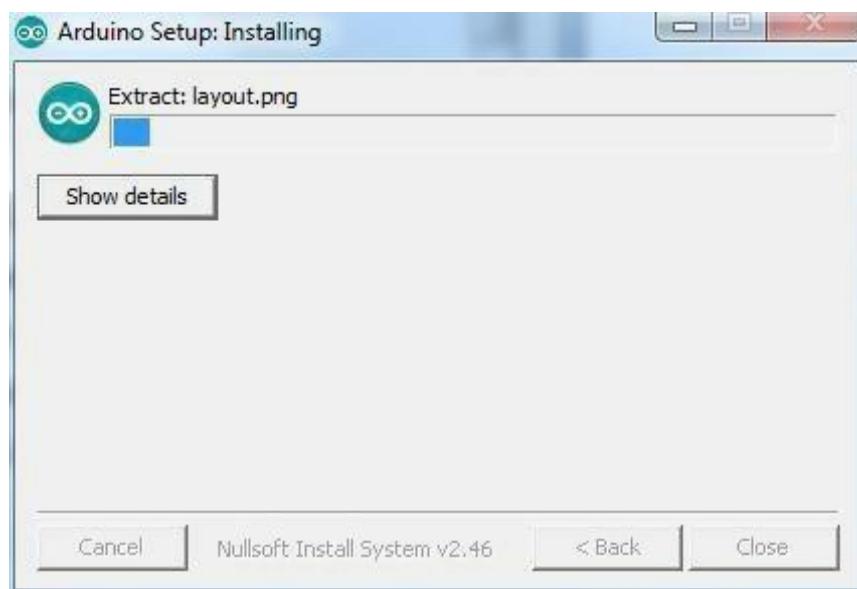


Haz clic en Next

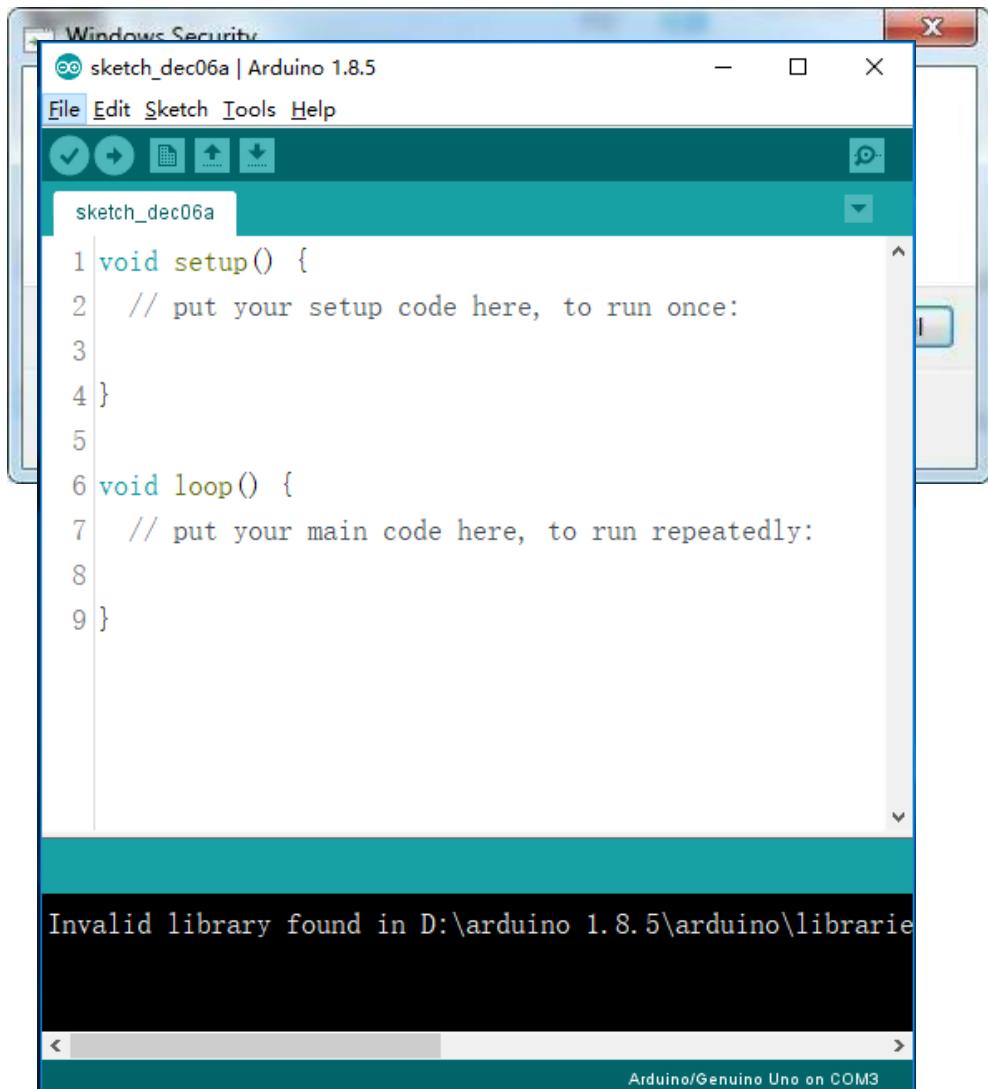
Puedes presionar **Browse...** para escoger la ruta en la cual quieras instalar el programa o escribir directamente el directorio.



Haz clic en *Install* para iniciar la instalación



Finalmente, aparecerá la siguiente interface. Haz clic en *Install* para finalizar la instalación.

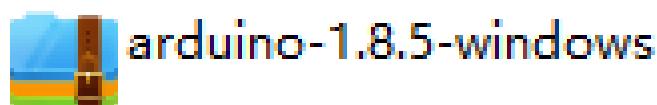


Después de eso, el siguiente ícono aparecerá en el escritorio

Haz doble clic sobre el para ingresar al IDE de Arduino

De esta forma, te hemos explicado como instalar el IDE de Arduino con el paquete de instalación .exe. A continuación te mostraremos algunas otras maneras de instalar el software, así que puedes saltar a la lección 1 directamente si así lo deseas instalar.

Descomprime el archivo zip descargado, haz doble clic para abrir el programa e ingresa en el entorno de desarrollo que deseas.



File Explorer Screenshot:

The File Explorer window shows the following file structure:

- Path: This PC > Storage (D:) > arduino 1.8.5 > arduino
- Items in 'arduino' folder:
 - drivers
 - examples
 - hardware
 - java
 - lib
 - libraries
 - reference
 - tools
 - tools-builder
 - arduino (selected)
 - arduino.ldj
 - arduino_debug
 - arduino_debug.ldj
 - arduino-builder
 - libusb0.dll
 - msvc100.dll
 - msvcr100.dll
 - revisions
 - uninstall
 - wrapper-manifest

Arduino IDE Screenshot:

The Arduino IDE window displays the following:

- Title bar: sketch_dec06a | Arduino 1.8.5
- Menu bar: File Edit Sketch Tools Help
- Toolbar icons: Checkmark, Refresh, Open, Save, Upload, Download, Print.
- Sketch code area:

```
1 void setup() {  
2     // put your setup code here, to run once:  
3  
4 }  
5  
6 void loop() {  
7     // put your main code here, to run repeatedly:  
8  
9 }
```
- Status bar: Arduino/Genuino Uno on COM3

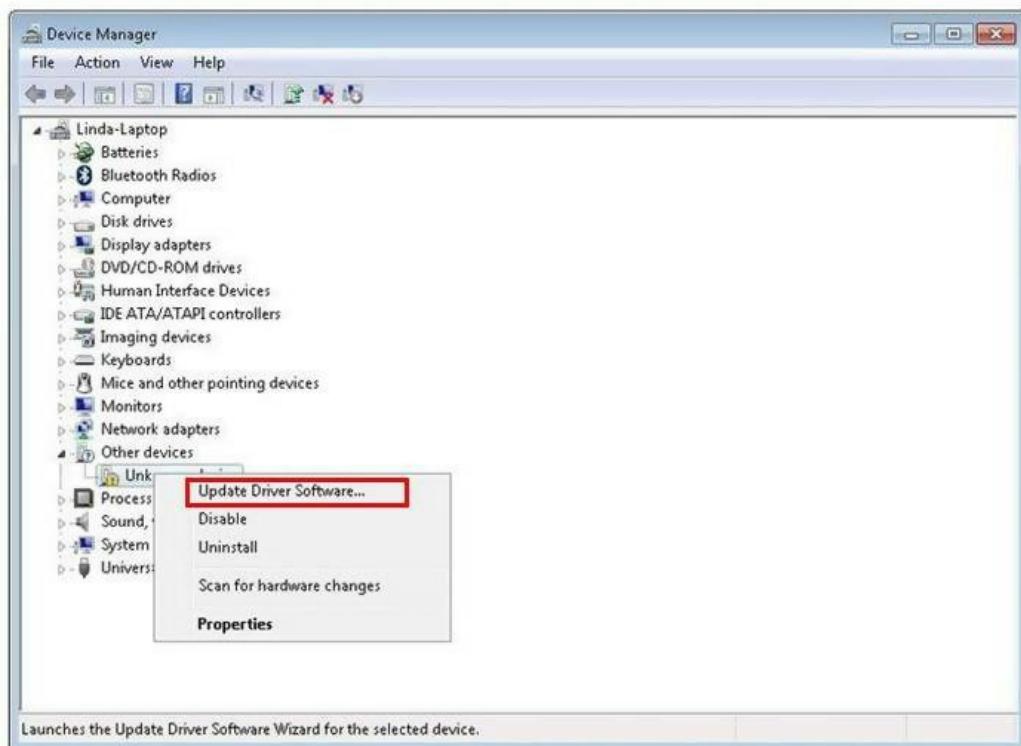
Es importante tomar en cuenta que este método de instalación requiere agregar otro driver adicional.

La carpeta contiene tanto el programa Arduino como los controladores (drivers) que le permiten a tu computador conectarse con Arduino a través del puerto USB. Antes de iniciar el software de Arduino, vas a configurar los drivers correspondientes.

- 1 Utiliza tu cable USB para conectar el Arduino con el puerto USB de tu computadora.
- 2 Verifica la indicación de la luz LED (una destellando y una estable)
- 3 Ignora el mensaje: "Nuevo hardware encontrado".
- 4 Cancela en caso que Windows trate de instalar los drivers de forma automática.

La manera más confiable de instalar los drivers USB es utilizando el Administrador de Dispositivos. Puedes acceder al Administrador de Dispositivos de diferentes formas dependiendo de tu versión de Windows. Por ejemplo, en Windows 7, necesitas abrir el Panel de Control, luego seleccionar la opción para ver los íconos y en la lista encontrarás el Administrador de Dispositivos.

En la categoría "Otros Dispositivos", deberías ver un ícono que diga "Dispositivo desconocido" con un pequeño triangulo amarillo como advertencia a su lado. Ese es tu dispositivo Arduino.





Haz clic derecho en el dispositivo y selecciona la opción en la parte superior del menú (Update Driver Software...). A continuación, se te pedirá seleccionar entre



'Search Automatically for updated driver software' (Buscar la actualización del software de manera automática) o 'Browse my computer for driver software' (Buscar en mi computadora el software controlador). Selecciona la segunda opción y navega a X\arduino1.8.5\drivers

Haz clic en 'Next'. Es probable que recibas un mensaje de seguridad, si es así, permite la instalación del software. Una vez que lo instales, recibirás un mensaje de confirmación.



Los usuarios de Windows pueden saltarse las instrucciones de instalación para Mac y Linux si lo desean, continuando a la lección 1. Los usuarios de Mac y Linux pueden continuar leyendo esta sección.

OSX 10.7 Lion or newer

Instalar Arduino en Mac OS X

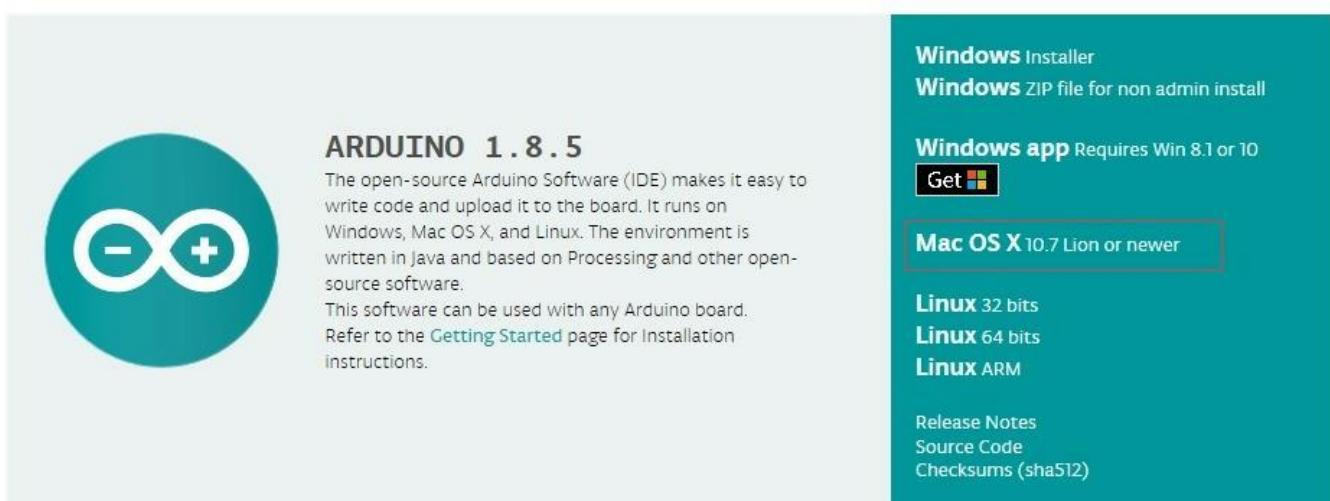
Paso Uno: Descargar el Software IDE de Arduino

Ve a la siguiente dirección:
[https://www.arduino.cc/e
n/Main/Software](https://www.arduino.cc/en/Main/Software)

Haz clic en JUST DOWNLOAD

Descarga y descomprime el archivo zip, haz doble clic en Arduino.app para ingresar al IDE de Arduino; el sistema

Te pedirá instalar la librería de Java runtime si no



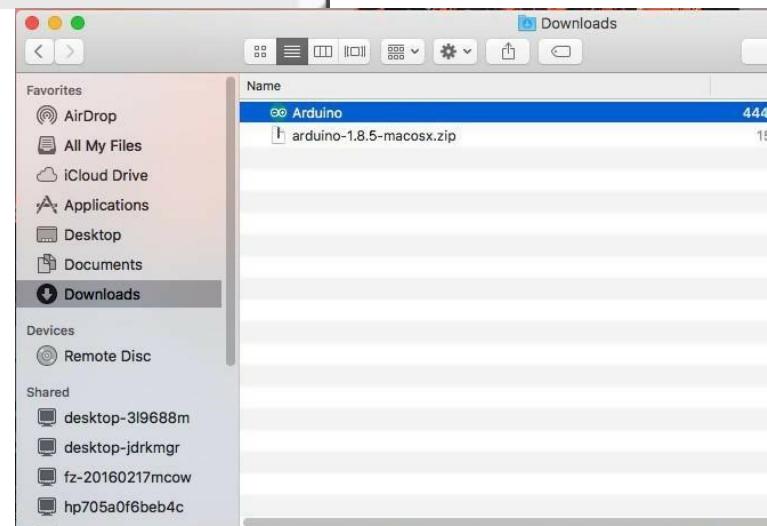
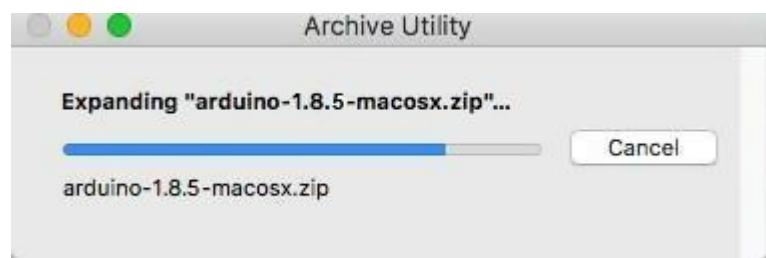
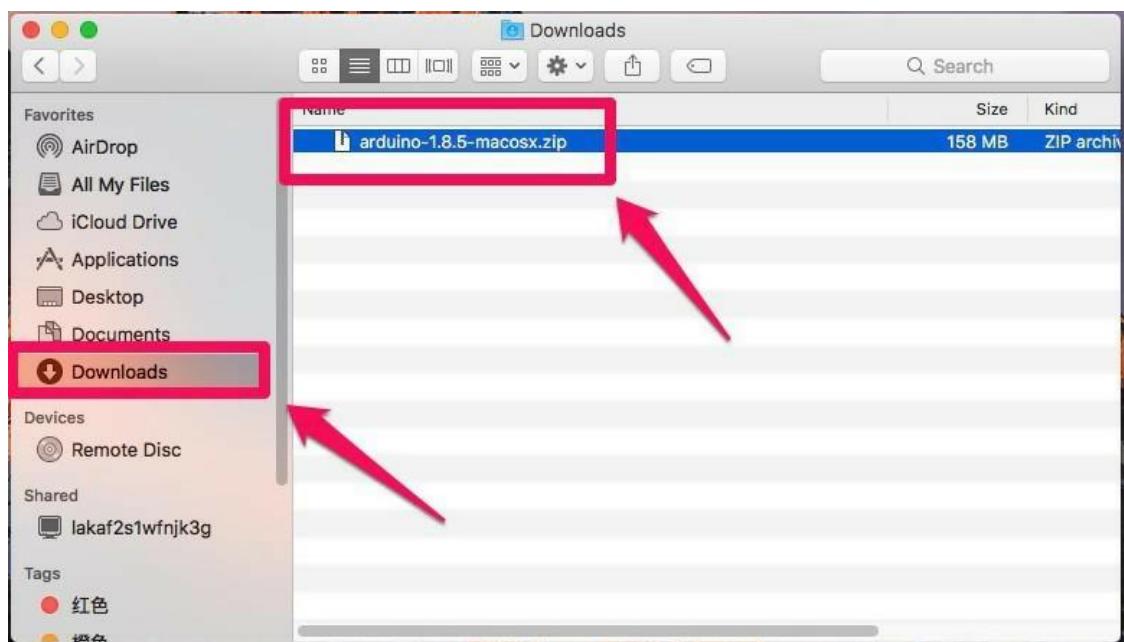
la tienes en tu computadora. Una vez que se termine de instalar, podrás correr el IDE de Arduino.

· Haz clic en Mac

Contribute to the Arduino Software

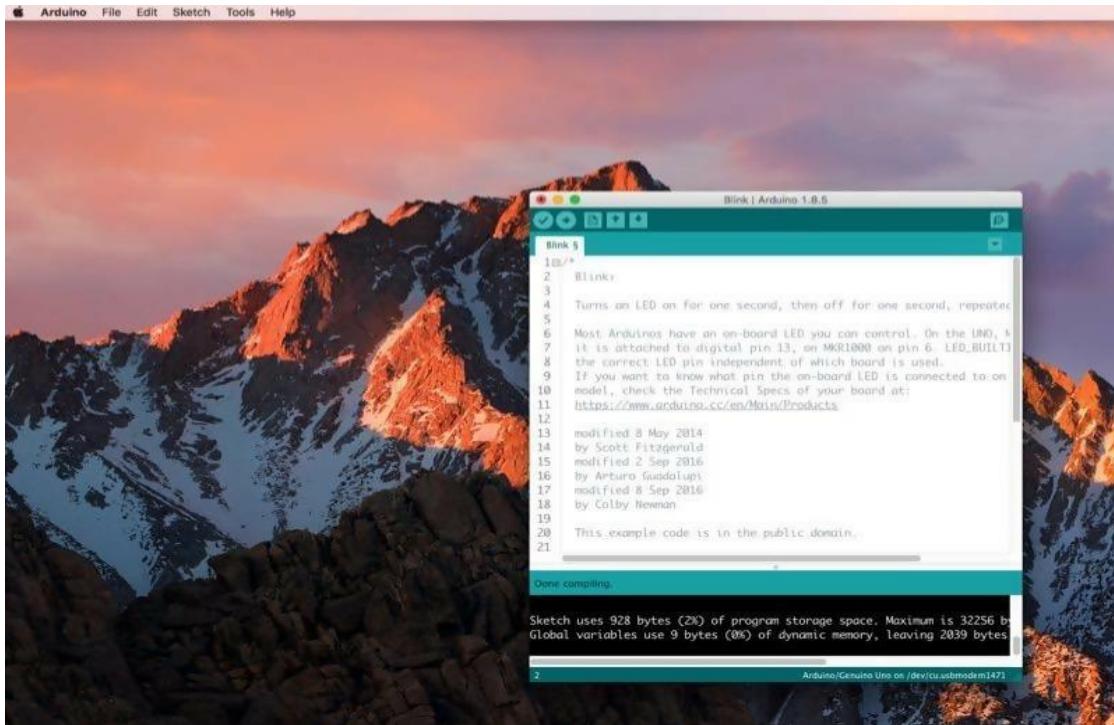
Consider supporting the Arduino Software by contributing to its development. (US tax payers, please note this contribution is not tax deductible). Learn more on how your contribution will be used.





- El archivo Arduino es la aplicación de Arduino, solo haz doble clic para abrirla.

- Arduino IDE



Instalar Arduino en Linux

Tendrás que utilizar el comando “make install”. Si utilizas Ubuntu, es recomendable instalar el IDE de Arduino desde el centro de software de Ubuntu.

Podrás obtener información específica de la instalación en el siguiente enlace:

<https://www.arduino.cc/en/guide/linux>

 [INSTALL ARDUINO ON LINUX.pdf](#)

 [arduino-1.8.5-linux32.tar.xz](#)

 [arduino-1.8.5-linux64.tar.xz](#)

Lección 1 Agregar Librerías

Una vez que tengas instalado el software de Arduino y hayas aprendido a usar las funciones integradas en el sistema por defecto, probablemente deseas expandir tus capacidades con las librerías adicionales de Arduino.

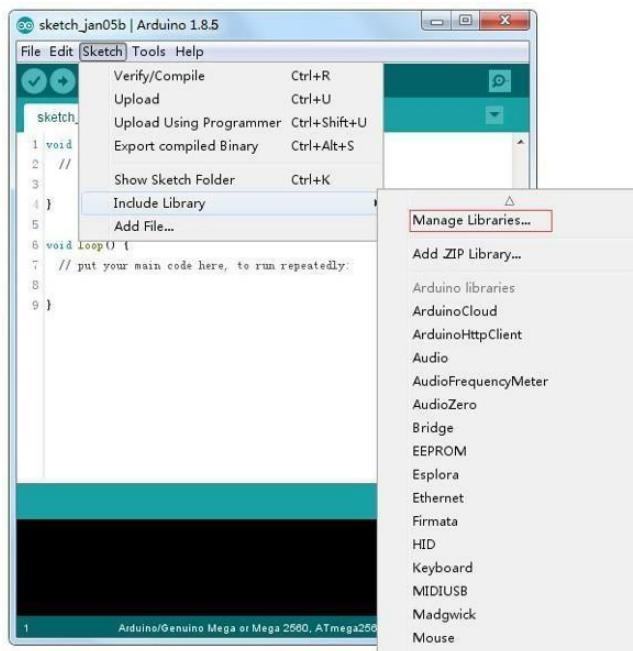
¿Qué son las librerías?

Las librerías son una colección de códigos que te facilitan conectarte a sensores, displays, módulos, etc. Por ejemplo, la librería integrada de Cristal Líquido te permite comunicarte en forma sencilla con los displays de LCD. Hay cientos de librerías adicionales disponibles para descargar en internet. Las librerías integradas y algunas de estas librerías adicionales pueden verse en las referencias. Para utilizarlas, es necesario que las instales primero.

Como instalar una librería

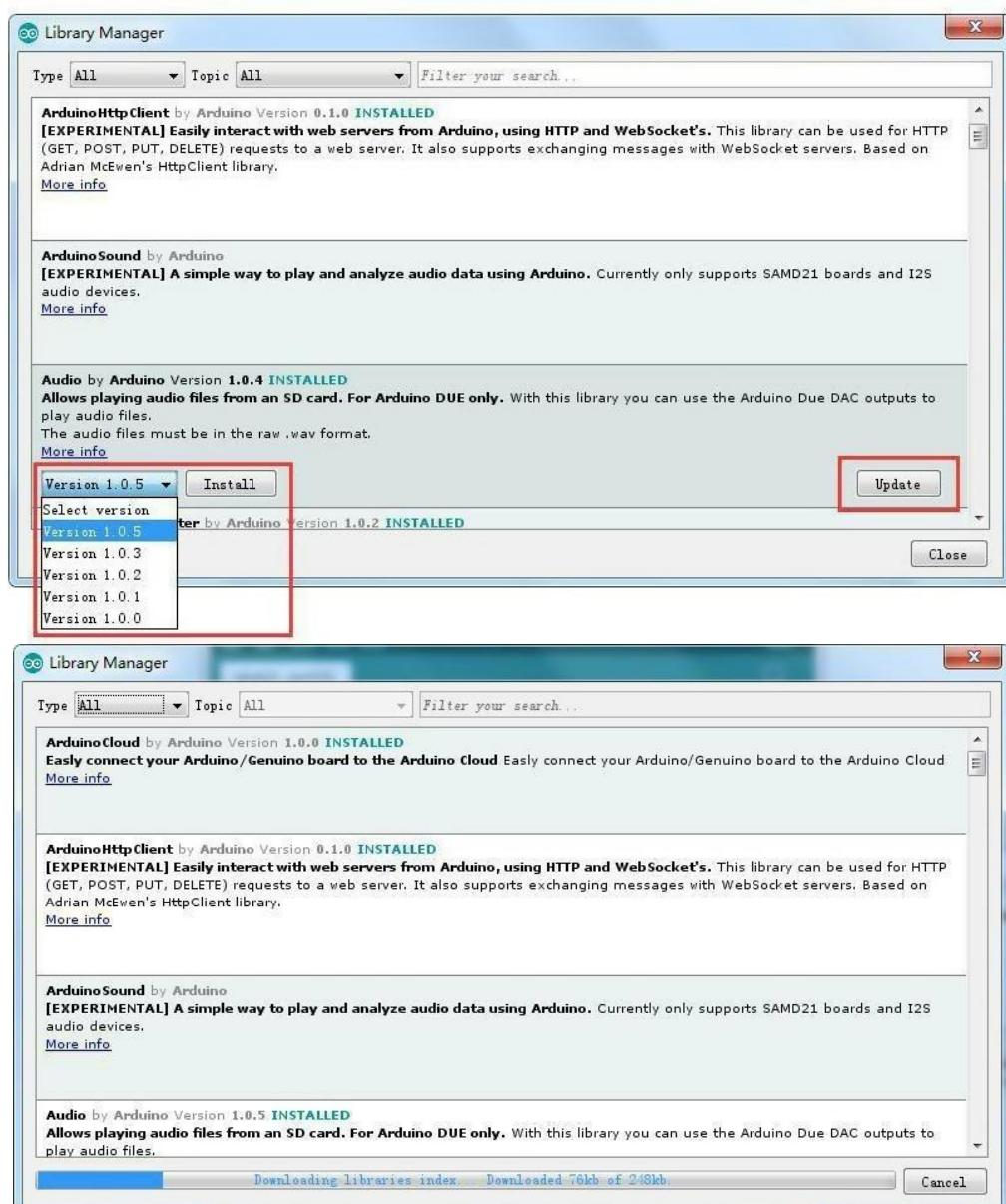
Utilizando el Library Manager

Para instalar una nueva librería en tu IDE de Arduino, puedes usar el Library Manager (disponible desde la versión 1.8.5 del IDE). Abre el IDE y haz clic en el menú "Sketch" y luego incluye las librerías en Library>Manage Libraries.



El Library Manager abrirá y en él podrás ver la lista de librerías instaladas o listas para instalar. En el ejemplo instalaremos la librería Bridge. Desplázate a lo largo de la lista para encontrarla y selecciona la versión de la librería a instalar. Algunas veces solo encontrarás una versión disponible. No te preocupes si el menú de selección de versiones no aparece, es normal que eso pase algunas veces

Trata de tener paciencia que se toma su tiempo; tal como aparece en la figura, refresca y espera un poco.



Para finalizar, haz clic en Install y espera que se incluya la nueva librería en el IDE. Descargarlo se toma cierto tiempo y depende de la velocidad de tu conexión a internet. Cuando se complete la instalación, una etiqueta de instalado aparecerá junto a la librería Bridge. Ya puedes cerrar el Library Manager para ese momento.

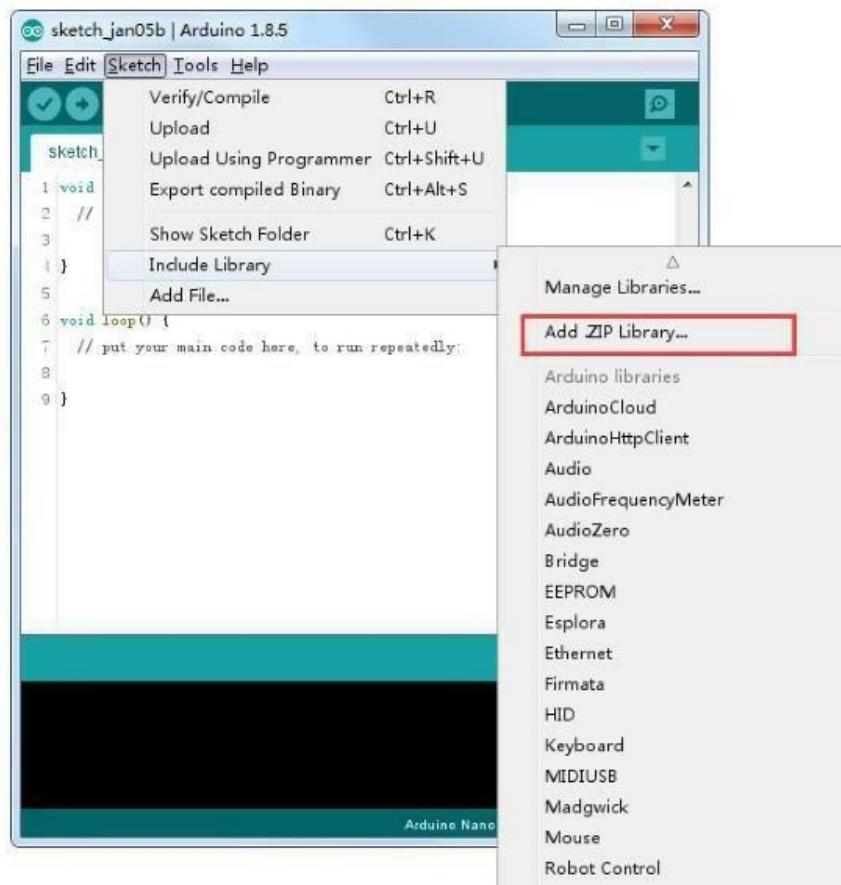


Ya tienes una nueva librería disponible en el menú incluir librerías. Si quieres agregar tu propia librería, abre un nuevo asunto en Github.

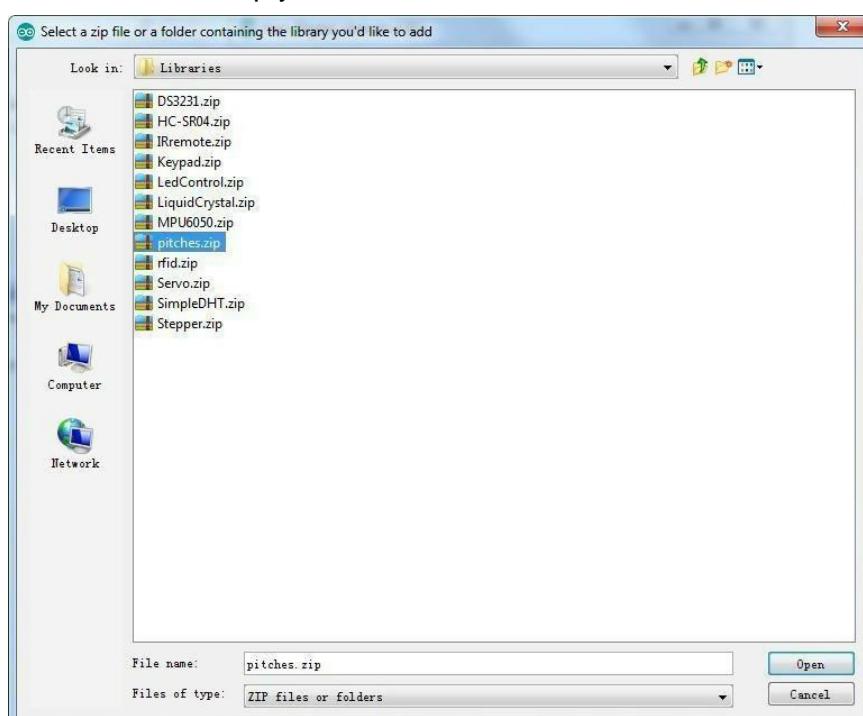
Importar una librería .zip

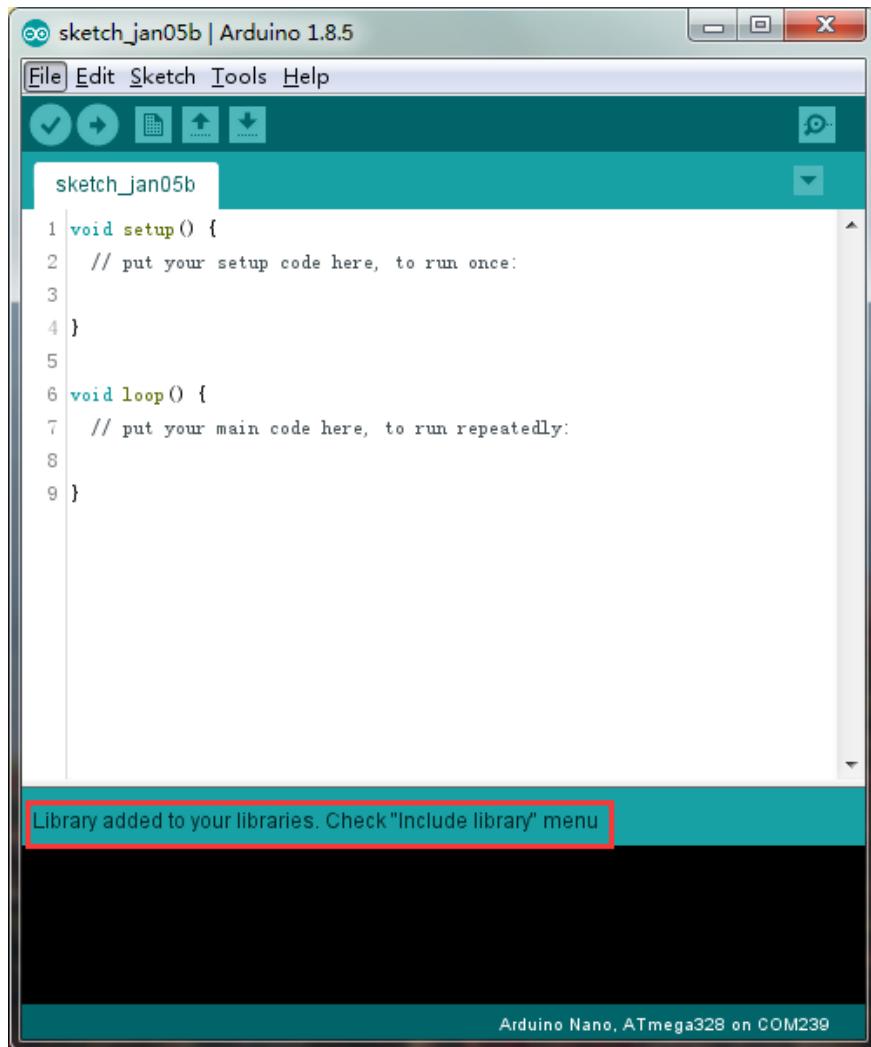
Las librerías se distribuyen normalmente mediante archivos o carpetas zip. El nombre de la carpeta casi siempre es el nombre de la librería. Dentro de la carpeta encontrarás un archivo .cpp, un .h, por lo general también uno llamado keywords.txt, una carpeta de ejemplos y otros archivos requeridos por la librería. A partir de la versión 1.0.5, puedes instalar librerías de terceros en el IDE. No descomprimas las librerías descargadas, déjala tal como está.

Navega a Sketch > Include Library en el IDE de Arduino. En la parte superior de la lista desplegable, selecciona la opción "Add .ZIP Library".



El sistema te solicitará seleccionar las librerías que quieras agregar. Navega hasta la ubicación de los archivos zip y ábrelos.





Vuelve al menú Sketch > Import Library. Deberías poder ver la librería al final de la lista desplegable. Ya está lista para usarse en tu Sketch. El archivo zip ya se habrá expandido en la carpeta librerías del directorio de tus sketches de Arduino. Importante: la librería estará disponibles para ser usada en los sketches pero no se expondrá en la carpeta [File>Examples hasta que no se haya reiniciado el IDE de Arduino](#).

Esos serían los dos métodos más comunes para instalar el IDE. Los sistemas Mac y Linux se pueden manejar de la misma manera. El proceso de instalación manual se muestra abajo aunque sea requerido en pocos casos. Los usuarios que no lo necesiten pueden saltarse esta parte.

Instalación Manual

Para instalar la librería, primero cierra la aplicación Arduino y descomprime el archivo zip que contiene la librería. Por ejemplo, si estás instalando una librería llamada

"Arduino Party", descomprime el archivo ArduinoParty.zip. Este debería tener una carpeta llamada ArduinoParty, con archivos como ArduinoParty.cpp y ArduinoParty.h adentro. (Si los archivos .cpp y .h no están en una carpeta, será necesario crearla. En ese caso, crea una carpeta de nombre "ArduinoParty" y coloca dentro de la misma todos los archivos que estaban en la carpeta zip, tales como ArduinoParty.cpp and ArduinoParty.h.)

Arrastra la carpeta Arduino Party a esta carpeta.

El directorio quedará más o menos así: "My Documents\Arduino\libraries".

Para usuarios Mac, será algo como: "Documents/Arduino/libraries".

Tu carpeta de librerías de Arduino debería verse así (en Windows):

MyDocuments\Arduino\libraries\ArduinoParty\ArduinoParty.cpp

MyDocuments\Arduino\libraries\ArduinoParty\ArduinoParty.h

My Documents\Arduino\libraries\ArduinoParty\examples

O así (en Mac y Linux):

Documents/Arduino/libraries/ArduinoParty/ArduinoParty.cpp

Documents/Arduino/libraries/ArduinoParty/ArduinoParty.h

Documents/Arduino/libraries/ArduinoParty/examples

Pueden haber más archivos que el .cpp y el .h, pero asegúrate que esos estén ahí (La librería no funcionará si colocas los archivos .cpp y .h directamente en la carpeta librerías o si se encuentran anidados en una carpeta extra.)

Por ejemplo:

Documents\Arduino\libraries\ArduinoParty.cpp y

Documents\Arduino\libraries\ArduinoParty\ArduinoParty.cpp no funcionarán)

Reinicia la aplicación Arduino.

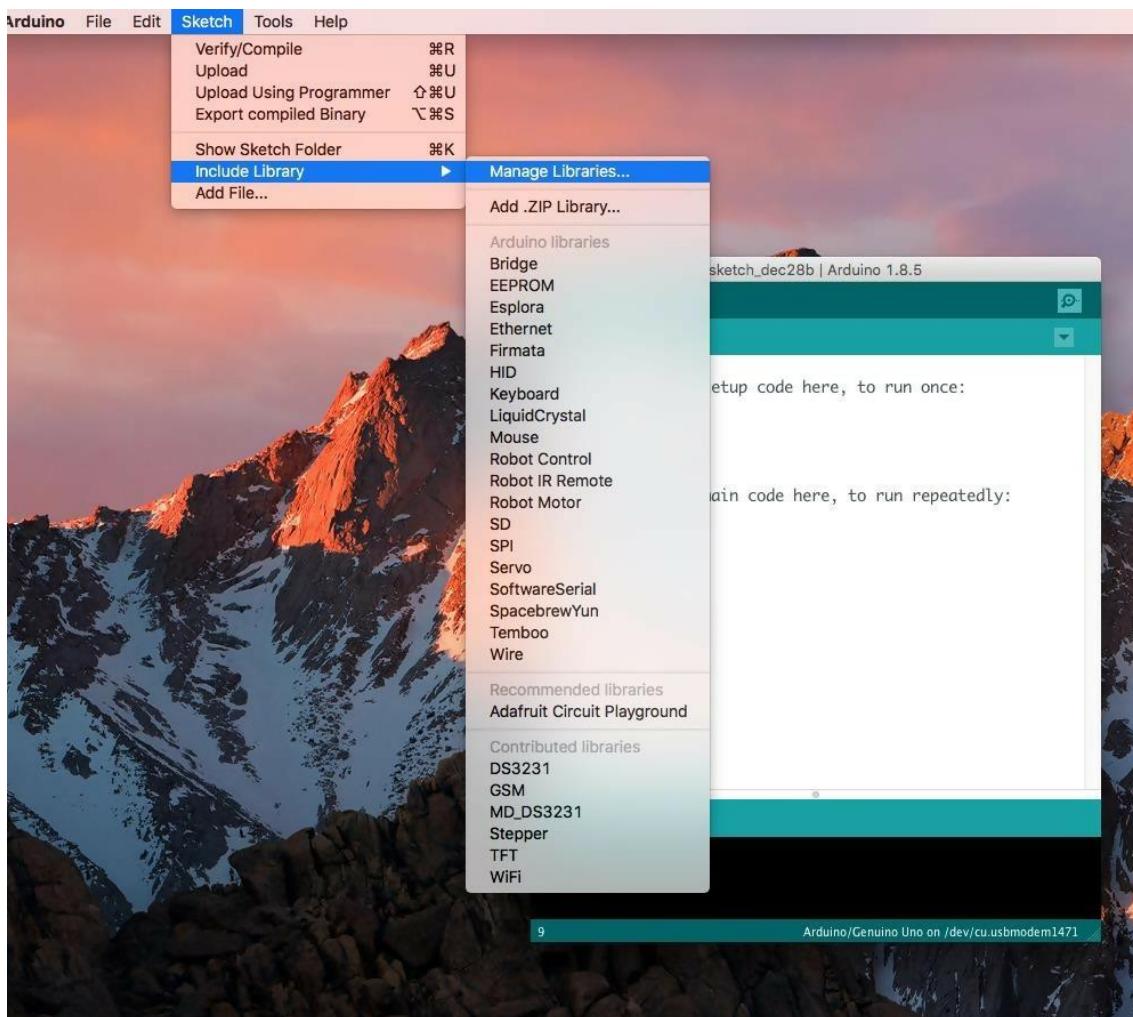
Asegúrate que la nueva librería aparezca en Sketch->Import Library en el software.

Listo! Ya has instalado una librería con éxito!

Como instalar librerías en el entorno Mac

Utilizando el Library Manager

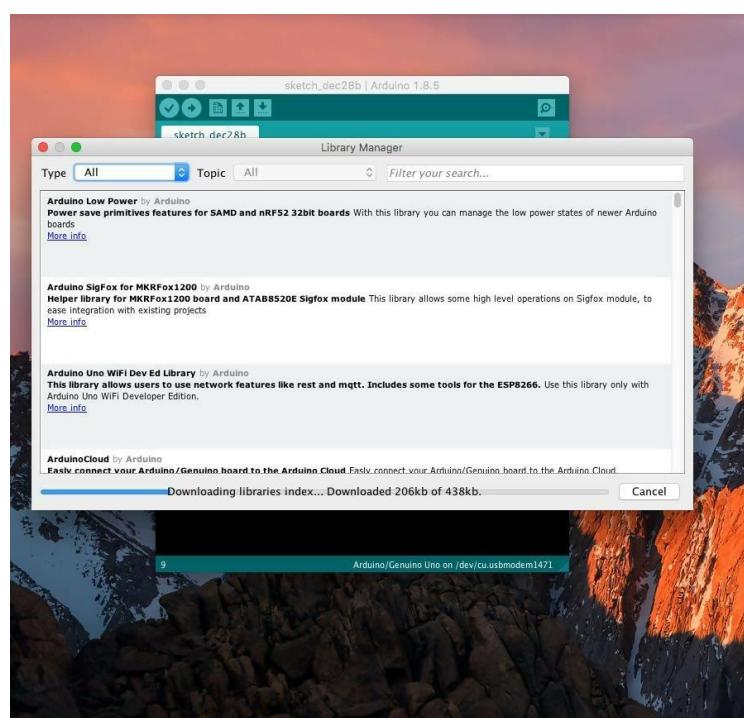
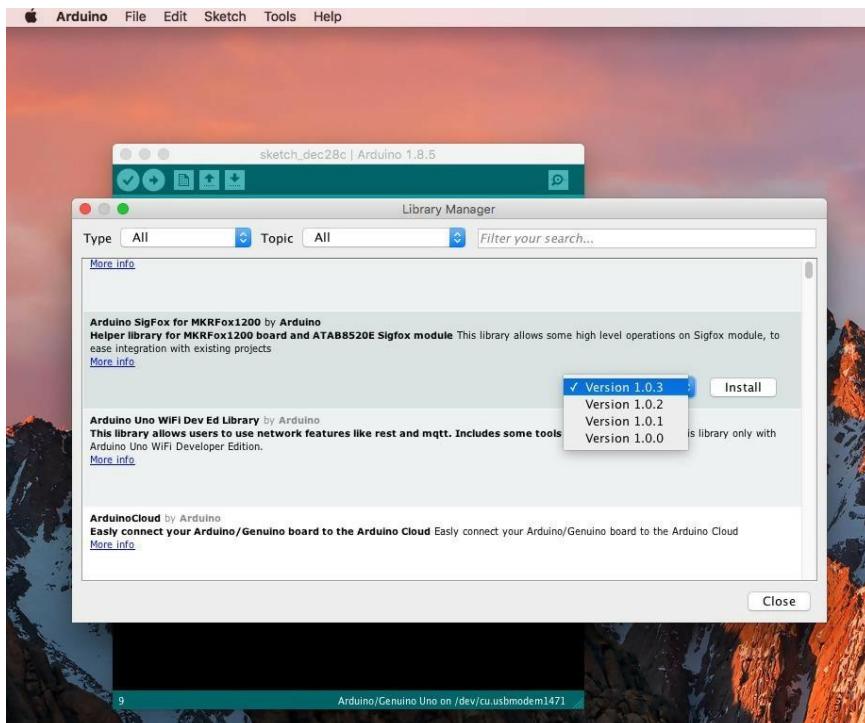
Para instalar una nueva librería en tu IDE de Arduino, puedes usar el Library Manager (disponible desde la versión 1.8.5 del IDE). Abre el IDE y haz clic en el menú "Sketch" y luego en las librerías Library > Manage Libraries.



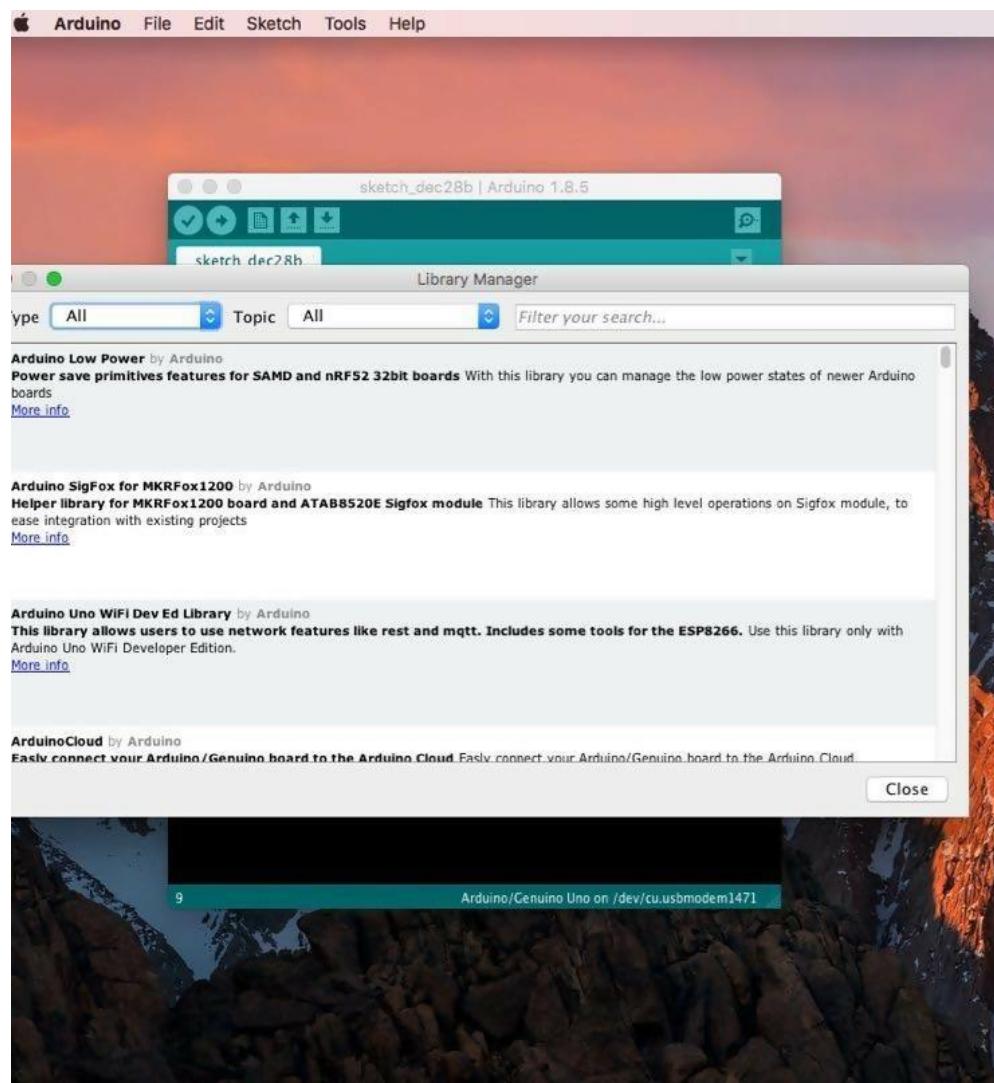
El Library Manager abrirá y en él podrás ver la lista de librerías instaladas o listas

para instalar. En el ejemplo instalaremos la librería Bridge. Desplázate a lo largo de la lista para encontrarla, y selecciona la versión de la librería que quieras instalar. Algunas veces solo habrá una versión de la librería disponible. Si no te aparece el menú de selección de versiones, no te preocupes, es algo normal.

Trata de tener paciencia que se toma su tiempo; tal como aparece en la figura. Por favor refresca y espera un poco.



Para finalizar, haz clic en Install y espera que se incluya la nueva librería en el IDE. Descargarlo se toma cierto tiempo y depende de la velocidad de tu conexión a internet. Cuando se complete la instalación, una etiqueta de instalado aparecerá junto a la librería Bridge. Ya puedes cerrar el Library Manager para ese momento.

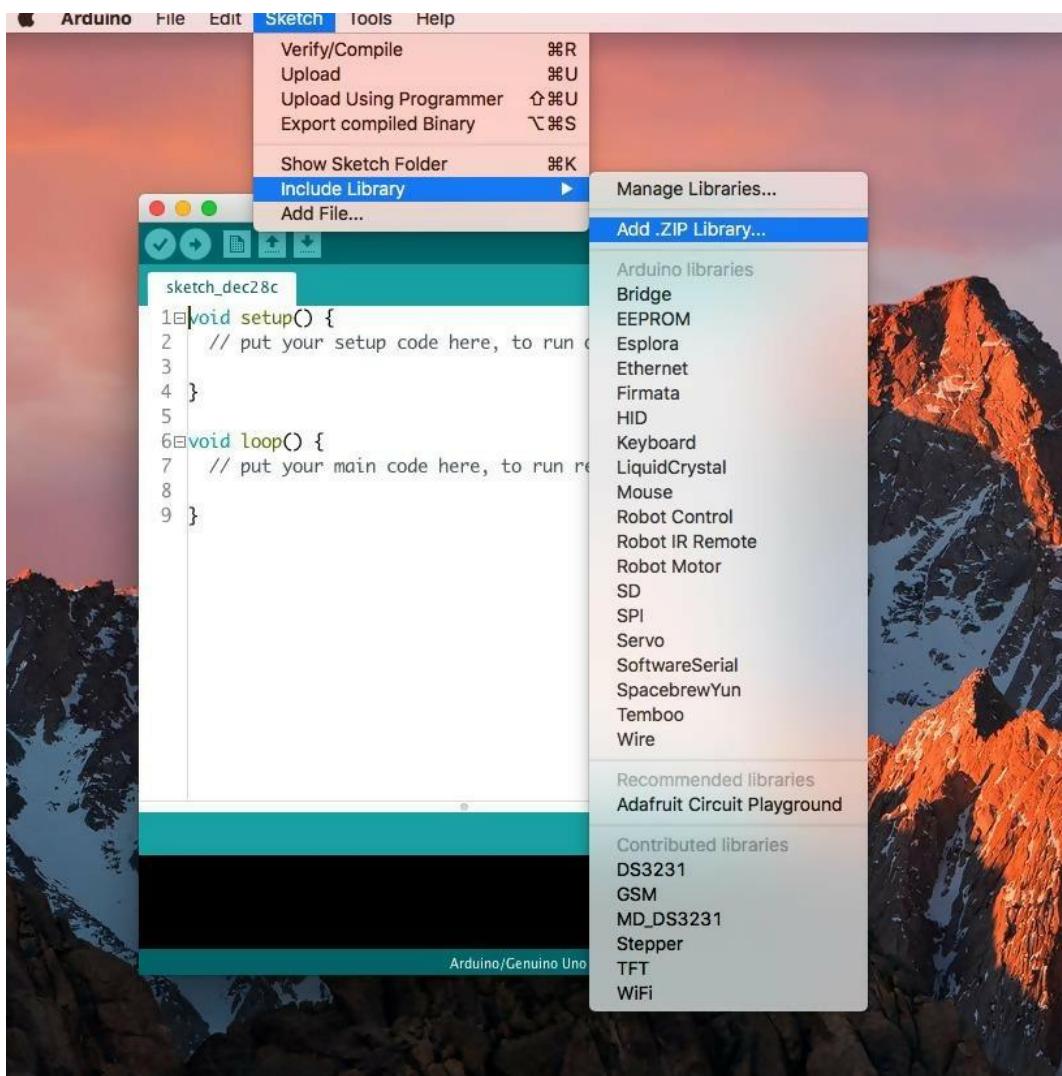


Ya tienes una nueva librería disponible en el menú incluir librerías. Si quieres agregar tu propia librería, abre un nuevo asunto en Github.

Importar una librería .zip

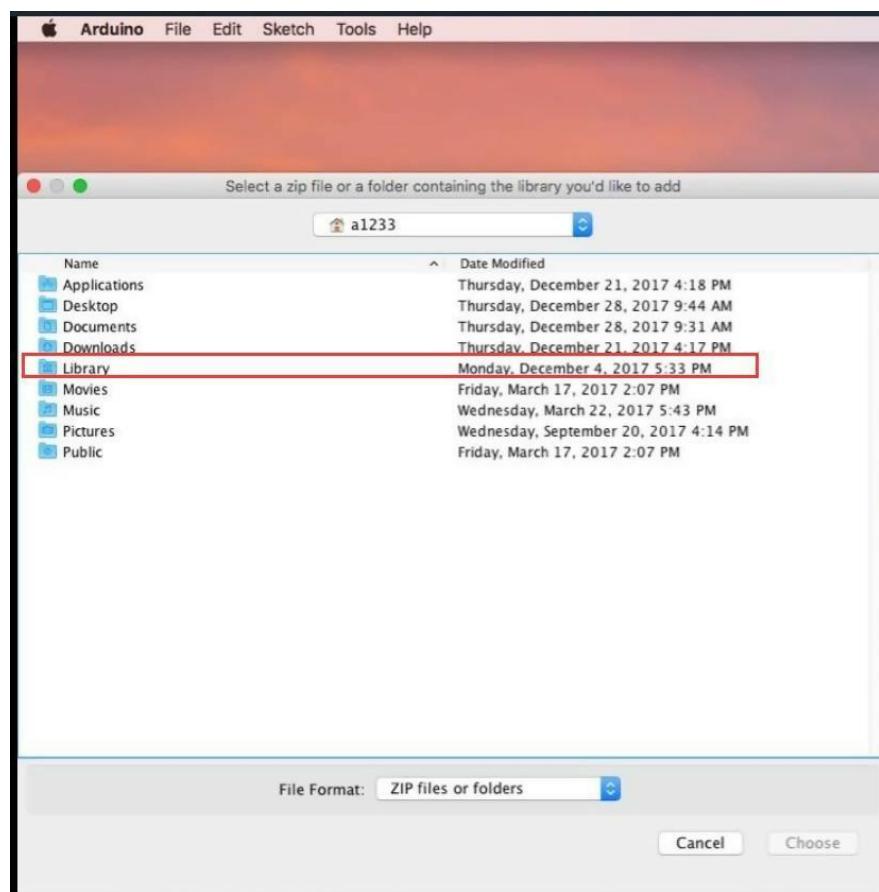
Las librerías se distribuyen normalmente mediante archivos o carpetas zip. El nombre de la carpeta casi siempre es el nombre de la librería. Dentro de la carpeta encontrarás un archivo .cpp, un .h, por lo general también uno llamado keywords.txt, una carpeta de ejemplos y otros archivos requeridos por la librería. A partir de la versión 1.0.5, puedes instalar librerías de terceros en el IDE. No descomprimas las librerías descargadas, déjala tal como está.

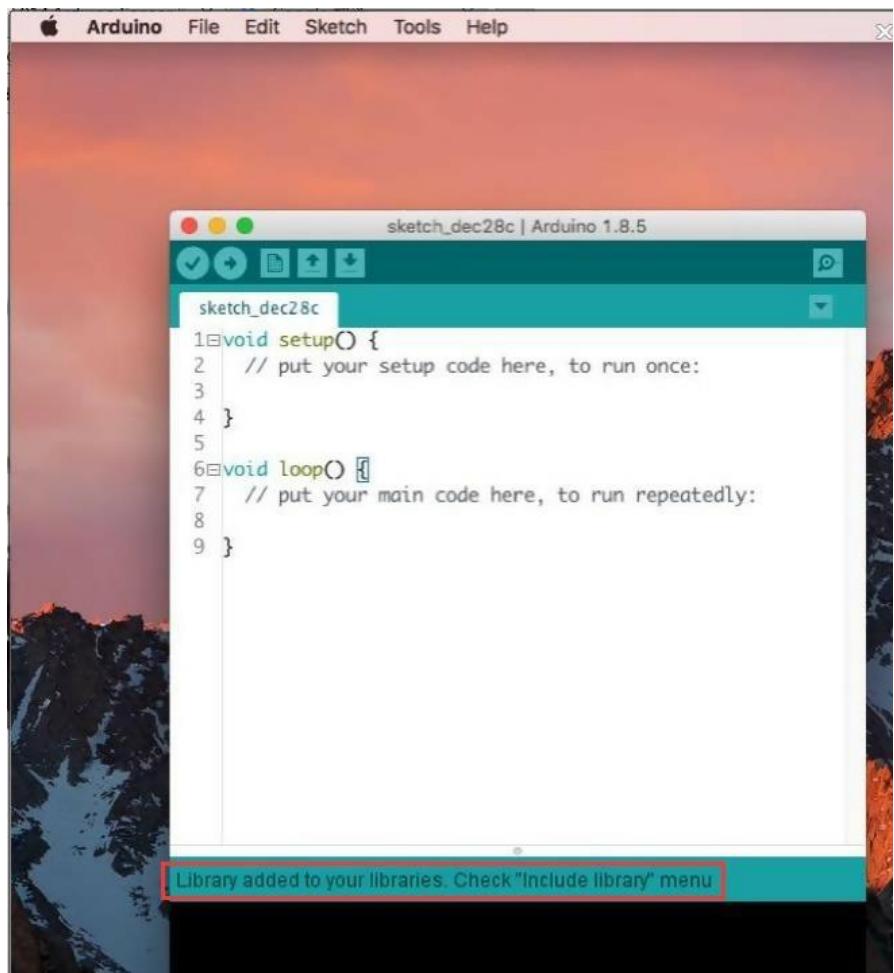
Navega a Sketch > Include Library en el IDE de Arduino. En la parte superior de la lista desplegable, selecciona la opción "Add .ZIP Library".



El sistema te solicitará seleccionar las librerías que quieras agregar. Navega

hasta la ubicación de los archivos zip y ábrelos.





Vuelve al menú Sketch > Import Library. Deberías poder ver la librería al final de la lista desplegable. Ya está lista para usarse en tu Sketch. El archivo zip ya se habrá expandido en la carpeta librerías del directorio de tus sketches de Arduino.

Importante: la librería estará disponibles para ser usada en los sketches pero no se expondrá en la carpeta <t125/> <t126/>File<t127/> <t128/>><t129/> <t130/>Examples<t131/> <t132/> hasta que no se haya reiniciado el IDE de Arduino.

Esos serían los dos métodos más comunes para instalar el IDE. Los sistemas Mac y Linux se pueden manejar de la misma manera. El proceso de instalación manual se muestra abajo aunque sea requerido en pocos casos. Los usuarios que no lo necesiten pueden saltarse esta parte.

Instalación Manual

Para instalar la librería, primero cierra la aplicación Arduino y descomprime el archivo zip que contiene la librería. Por ejemplo, si estas instalando a una librería llamada "ArduinoParty", descomprime el archivo ArduinoParty.zip. Debería contener una carpeta llamada Arduino Party, con archivos como ArduinoParty.cpp y ArduinoParty.h adentro (Si los archivos .cpp y .h no están en una carpeta, será necesario crearla. En ese caso, crea una carpeta de nombre "ArduinoParty" y coloca dentro de la misma todos los archivos que estaban en la carpeta zip, tales como ArduinoParty.cpp and ArduinoParty.h)

Arrastra la carpeta Arduino Party a esta carpeta. El directorio quedará más o menos así: "My Documents\Arduino\libraries". Para usuarios Mac, será algo como: "Documents/Arduino/libraries".<t32/> <t33/>En<t34/> <t35/>Linux,<t36/> será la carpeta "librerías" de tu sketchbook.

Tu carpeta de librerías de Arduino debería verse así (en Windows):
MyDocuments\Arduino\libraries\ArduinoParty\ArduinoParty.cpp
My Documents\Arduino\libraries\ArduinoParty\ArduinoParty.h

My Documents\Arduino\libraries\ArduinoParty\examples

O así (en Mac y Linux):

Documents/Arduino/libraries/ArduinoParty/ArduinoParty.cpp

Documents/Arduino/libraries/ArduinoParty/ArduinoParty.h

<t3>Documents/Arduino/libraries/ArduinoParty/examples

(La librería no funcionará si colocas los archivos .cpp y .h directamente en la carpeta librerías o si se encuentran anidados en una carpeta extra.)

Por ejemplo: Documents\Arduino\libraries\ArduinoParty.cpp y
Documents\Arduino\libraries\ArduinoParty\ArduinoParty\ArduinoParty.cpp no funcionarán)

Reinicia la aplicación Arduino. Asegúrate que la nueva librería aparezca en Sketch->Import Library en el software.

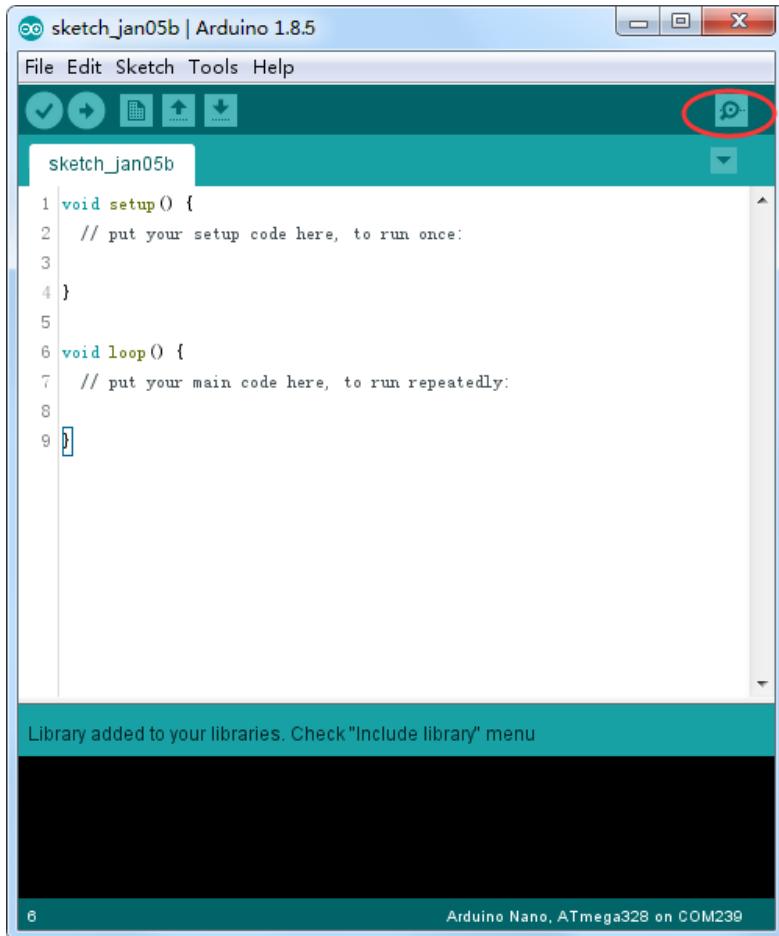
Listo! Ya has instalado una librería con éxito!

Lección 2 Abrir Monitor Serial de Arduino

El Entorno de Desarrollo Integrado o IDE (siglas en inglés de Integrated Development Environment) es el software que se utiliza en la plataforma de Arduino. Tomando en cuenta que usar un terminal es una parte muy importante para trabajar con Arduino y otros microcontroladores, se decidió incluir una terminal serial con el software. En el entorno Arduino se le llama Monitor Serial.

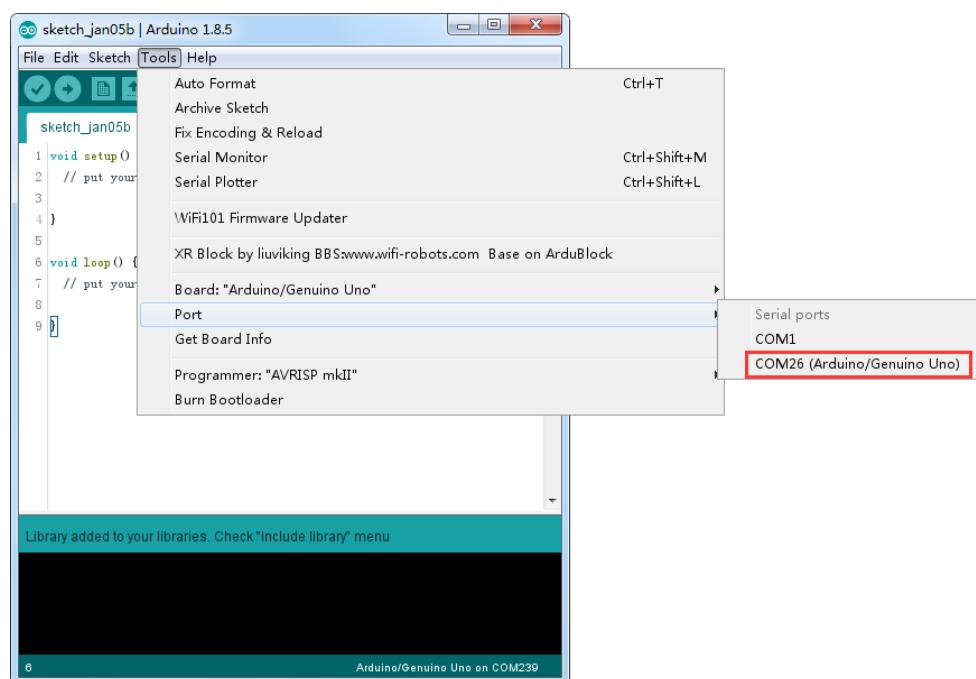
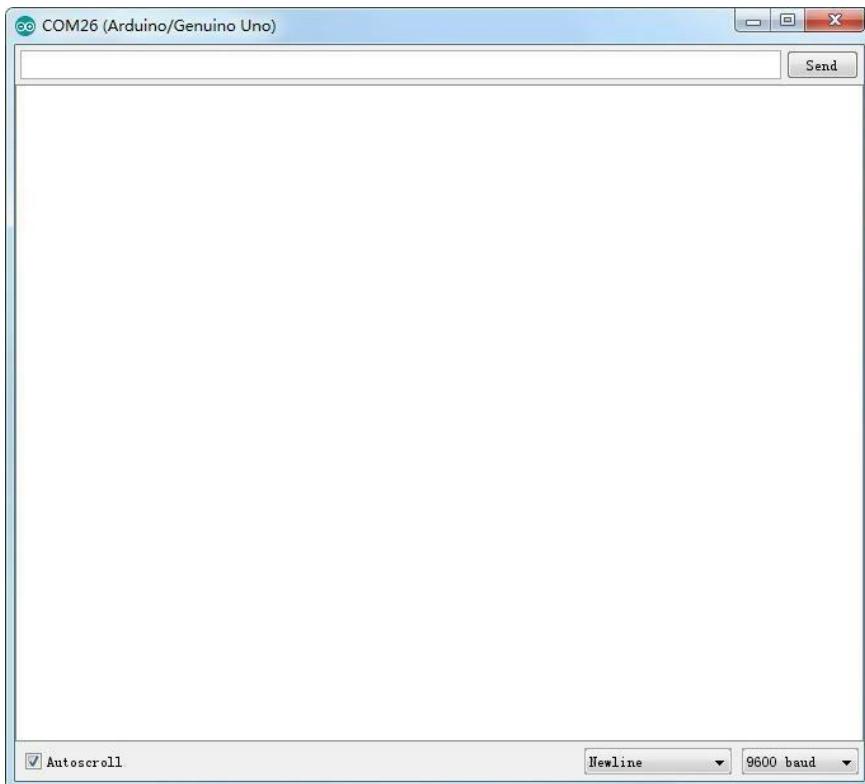
Realizar una conexión

El monitor serial viene con todas y cada una de las versiones del IDE de Arduino. Para abrirlo, simplemente haz clic en el ícono de monitor serial.



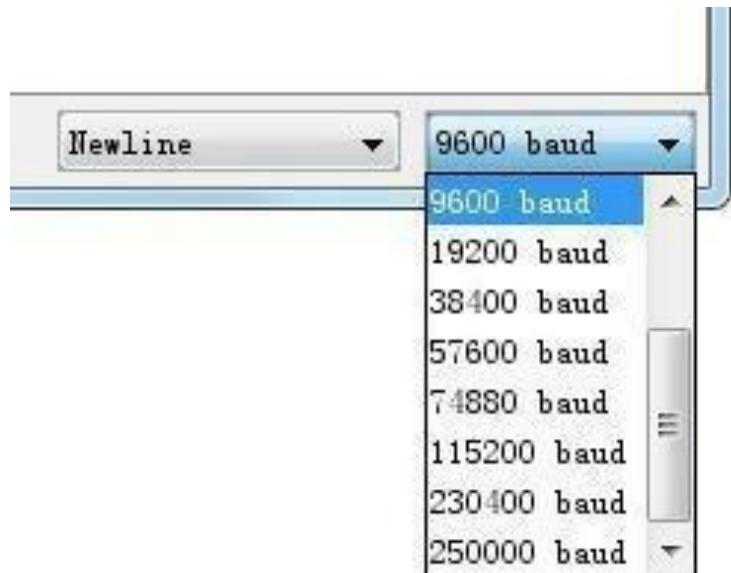
Seleccionar el puerto a abrir en el Monitor Serial equivale a seleccionar el puerto a través del cual se cargará el código Arduino. Ve a Tools -> Serial Port, y selecciona el puerto correcto.

Tips: Escoge el mismo puerto COM que se muestra en el Administrador de Dispositivos Una vez que lo abras, deberías ver algo como esto:



Configuración

El Monitor Serial tiene pocos ajustes, pero los suficientes para manejar la mayoría de tus necesidades de comunicación serial. El primer ajuste que puedes alterar es el baud rate o velocidad de transmisión. Haz clic en el menú desplegable de Baud rate para seleccionar la velocidad de transmisión correcta. (9600 baudios)



Por último, puedes configurar que el terminal haga Auto scroll o no, tildando la casilla de verificación ubicada en la esquina inferior izquierda.



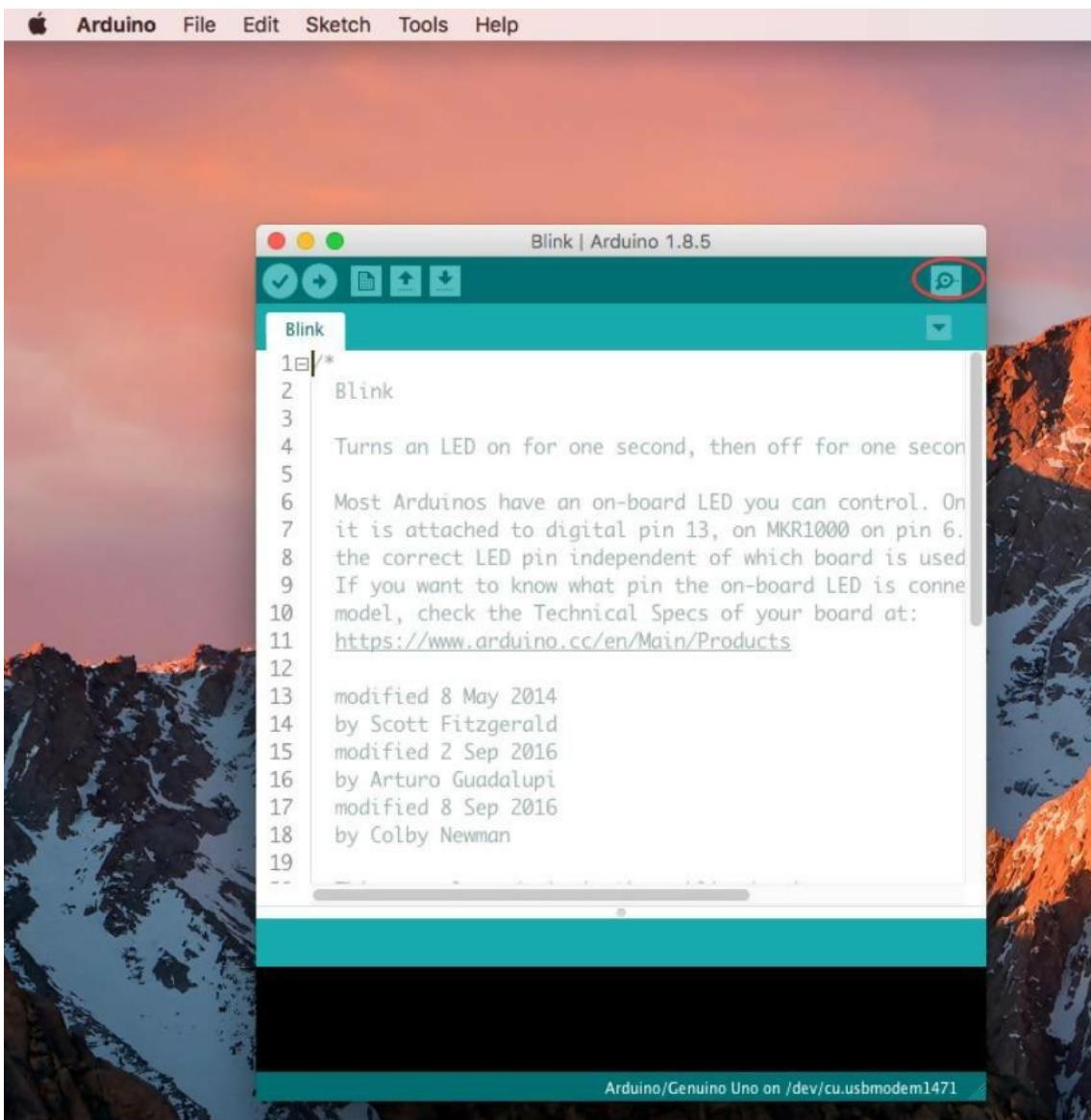
Pros

El Monitor Serial es una excelente manera de establecer una conexión serial con tu Arduino de forma rápida y sencilla. Si ya te encuentras trabajando en el IDE de Arduino, no hay necesidad de abrir una ventana separada del terminal para mostrar los datos.

Contras

Los escasos ajustes que se pueden configurar dejan mucho que desear en el Monitor Serial, por lo que para comunicaciones seriales avanzadas puede no funcionar como se esperaría.

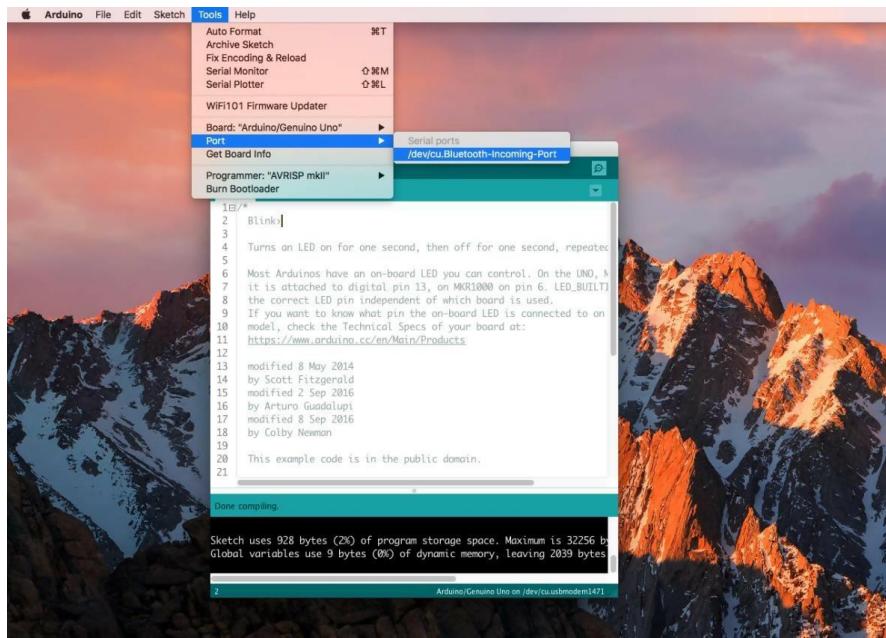
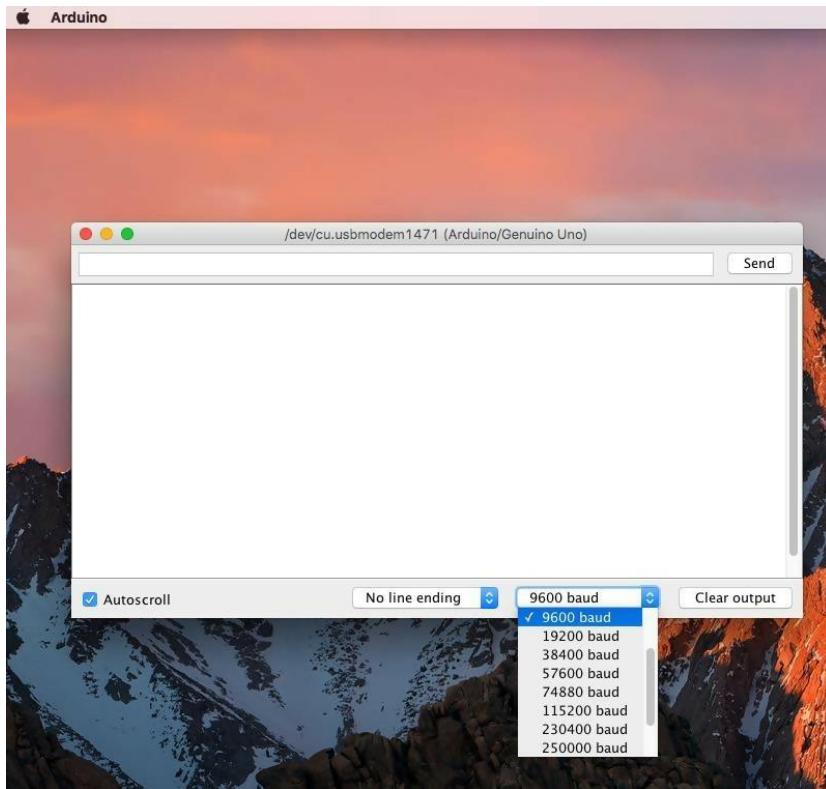
En el entorno Mac, abre el monitor de puerto serial:



Seleccionar el puerto a abrir en el Monitor Serial equivale a seleccionar el puerto a través del cual se cargará el código Arduino. Ve a Tools -> Serial Port, y selecciona el puerto correcto.

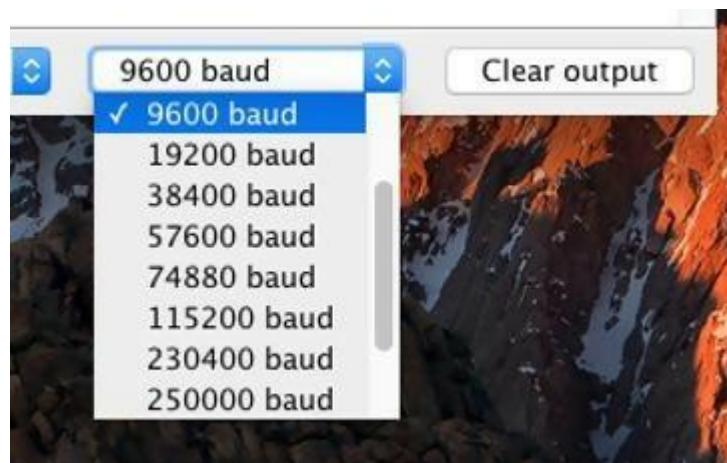
Tips: Escoge el mismo puerto COM que tienes en el Administrador de Dispositivos.

Una vez que lo abras, deberías ver algo como esto:

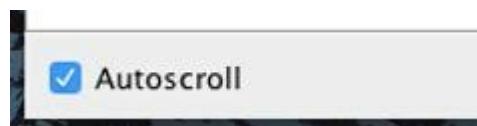


Configuración

El Monitor Serial tiene pocos ajustes, pero los suficientes para manejar la mayoría de tus necesidades de comunicación serial. El primer ajuste que puedes alterar es el baud rate o velocidad de transmisión. Haz clic en el menú desplegable de baud rate para seleccionar la velocidad de transmisión correcta. (9600 baudios)



Por último, puedes configurar que el terminal haga Auto scroll o no, tildando la casilla de verificación ubicada en la esquina inferior izquierda.



Pros

El Monitor Serial es una excelente manera de establecer una conexión serial con tu Arduino de forma rápida y sencilla. Si ya te encuentras trabajando en el IDE de Arduino, no hay necesidad de abrir una ventana separada del terminal para mostrar los datos

Contras

Los escasos ajustes que se pueden configurar dejan mucho que desear en el Monitor Serial, por lo que para comunicaciones seriales avanzadas puede no funcionar como se esperaría.

Lección 3 Destello

Resumen

En esta lección, aprenderás a programar tu tarjeta controladora UNO R3 para hacer destellar su LED integrado así como para descargar programas.

La tarjeta UNO R3 posee filas de conectores de ambos lados, las cuales se pueden usar para conectar varios dispositivos electrónicos enchufables conocidos como "shields" o tarjetas auxiliares que extienden sus capacidades.

Tiene también un LED individual que puedes controlar desde tus sketches. Este LED está incorporado directamente sobre la tarjeta UNO R3 y es conocido comúnmente como el LED "L" ya que así aparece marcado en la tarjeta.



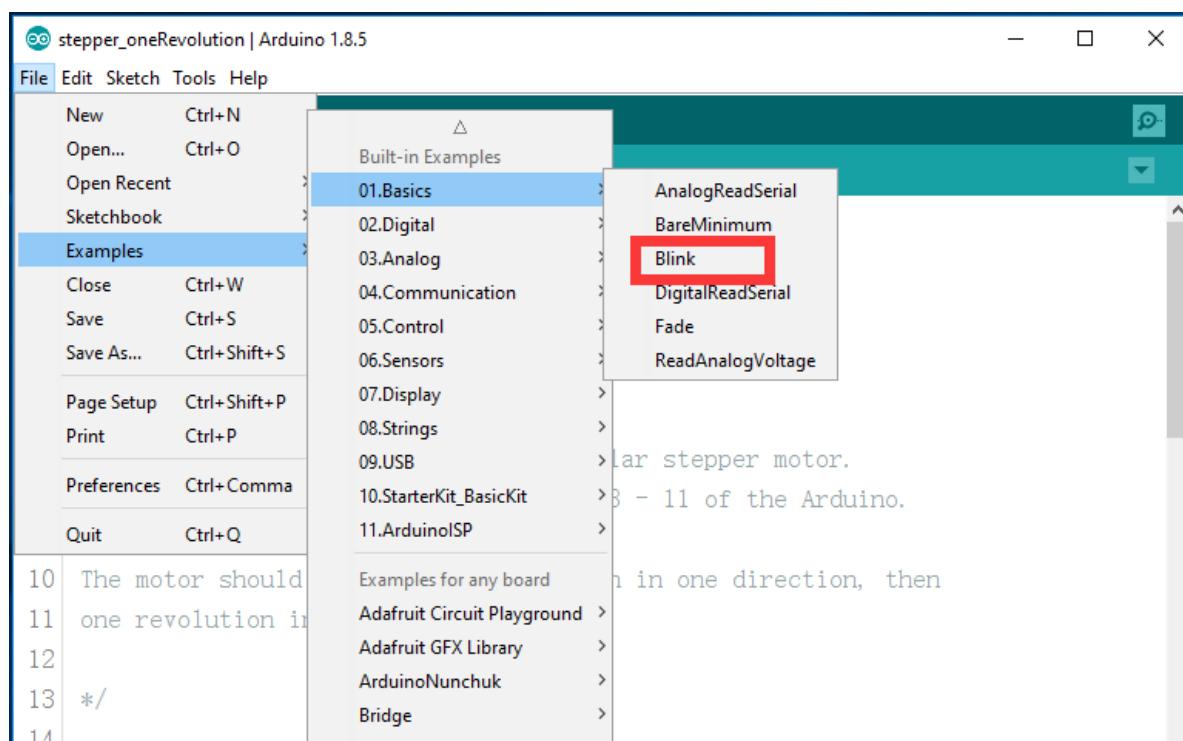
Como seguramente notarás, el LED "L" se ilumina cuando conectas la tarjeta a un puerto USB. Esto se debe a que las tarjetas se envían precargadas con el sketch "Blink" (o destello) instalado por defecto.

En esta lección, reprogramaremos la tarjeta UNO R3 con nuestro propio sketch de destello para después modificar la velocidad a la cual se ilumina.

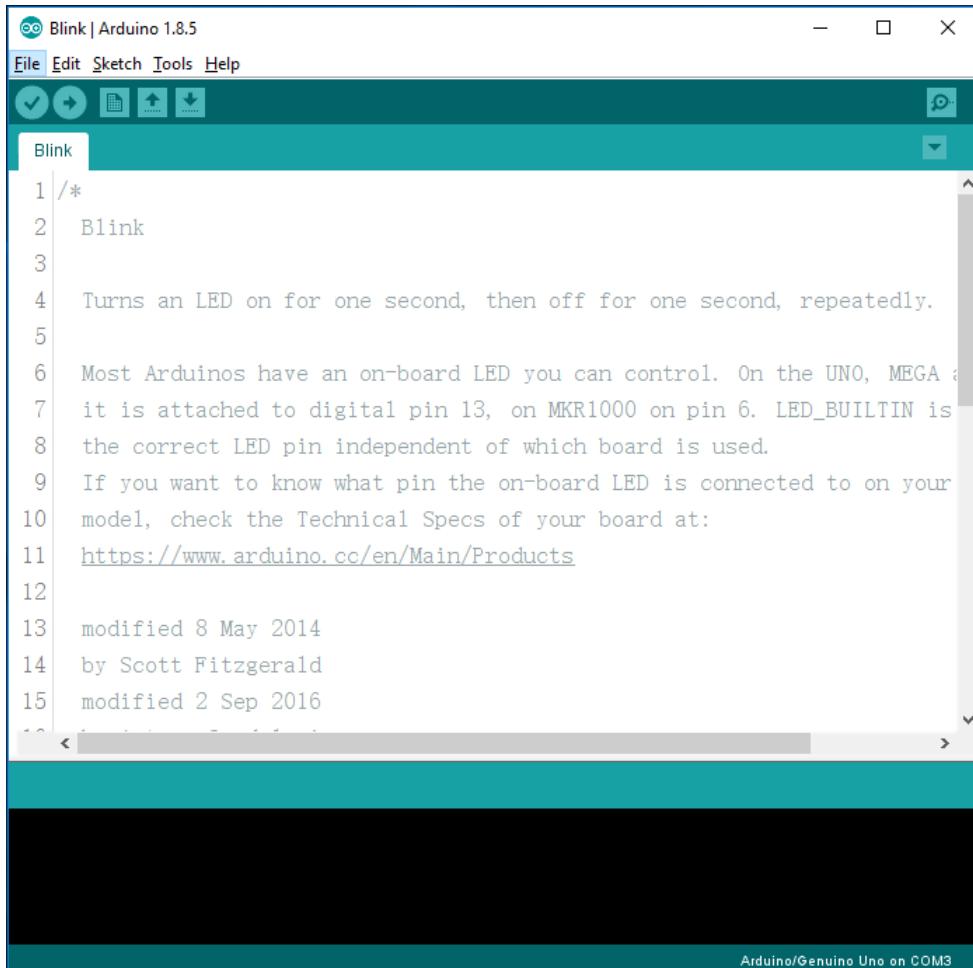
En la lección 0, configuraste tu IDE de Arduino y te aseguraste encontrar el puerto serial adecuado para conectarse con tu UNO R3. Pues ahora llegó el momento que pruebes esa conexión y programes tu tarjeta UNO R3.

El IDE de Arduino incluye una colección bastante grande de sketches de ejemplos para cargarlos y usarlos. Entre esos ejemplos está incluido un sketch para hacer destellar el LED "L".

Carga el sketch "Blink" que encontrarás en el siguiente menú del IDE: File > Examples > 01.Basics



Cuando abre la ventana del sketch, agrándala para que puedas ver el sketch por completo en la ventana.



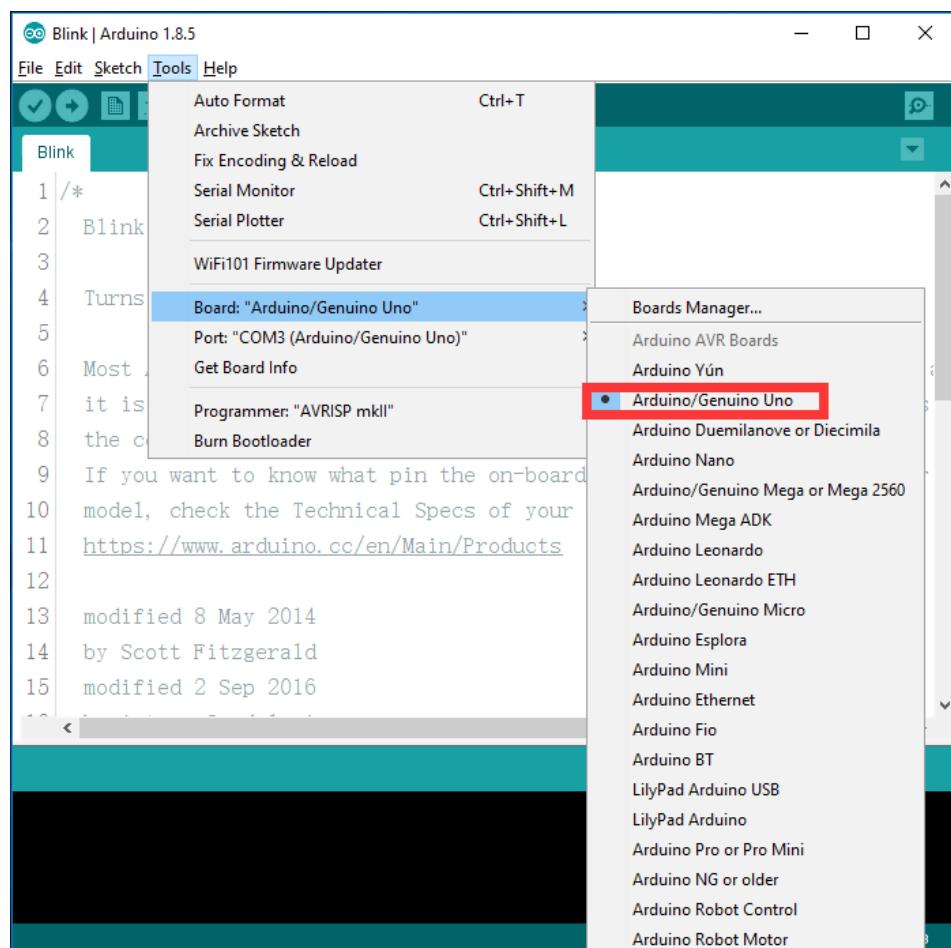
The screenshot shows the Arduino IDE interface with the title bar "Blink | Arduino 1.8.5". The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for Save, Run, Stop, and Upload. The main code editor window contains the "Blink" sketch. The code is as follows:

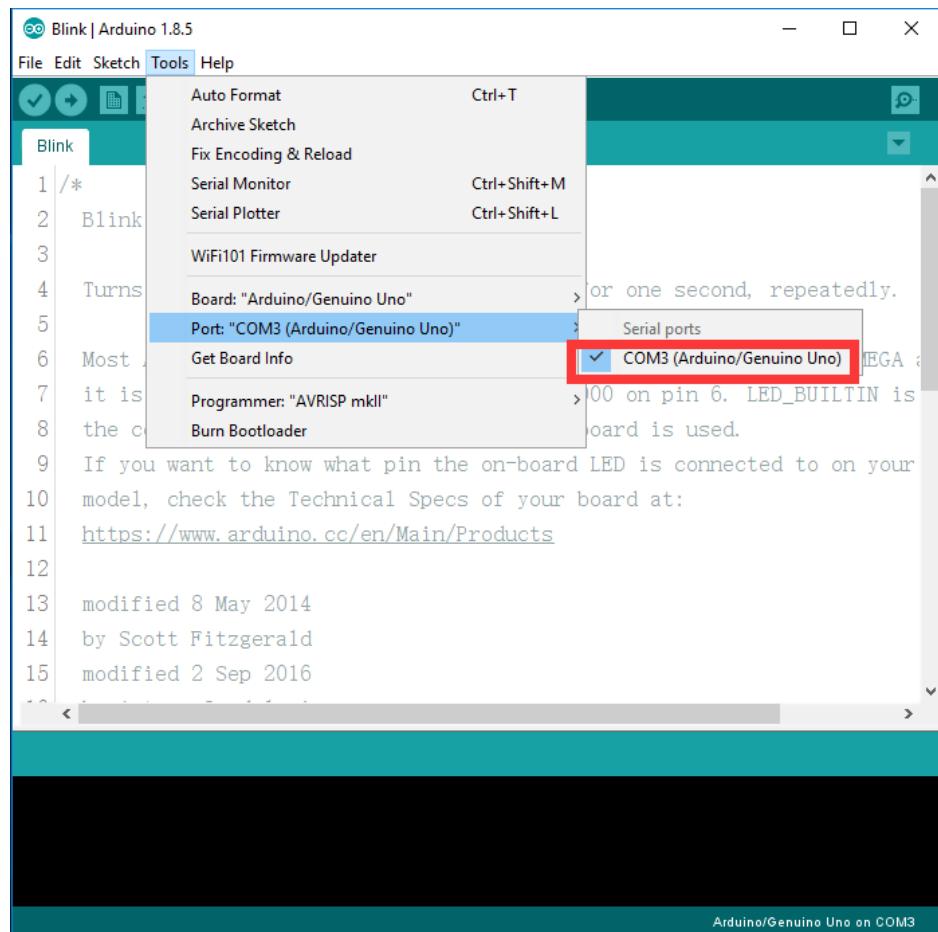
```
1 /*  
2  * Blink  
3  *  
4  * Turns an LED on for one second, then off for one second, repeatedly.  
5  *  
6  * Most Arduinos have an on-board LED you can control. On the UNO, MEGA and MKR  
7  * boards, it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is  
8  * the correct LED pin independent of which board is used.  
9  * If you want to know what pin the on-board LED is connected to on your  
10 * model, check the Technical Specs of your board at:  
11 * https://www.arduino.cc/en/Main/Products  
12 *  
13 * modified 8 May 2014  
14 * by Scott Fitzgerald  
15 * modified 2 Sep 2016
```

The status bar at the bottom right indicates "Arduino/Genuino Uno on COM3".

Los sketches de ejemplo incluidos con el IDE de Arduino son de solo lectura. Por lo tanto, los puedes subir a una tarjeta UNO R3 pero si haces algún cambio no podrás salvarlo bajo el mismo nombre.

Conecta tu tarjeta Arduino a tu computadora con el cable USB y verifica que el tipo de tarjeta ('Board Type') y el puerto serial ('Serial Port') estén configurados correctamente.





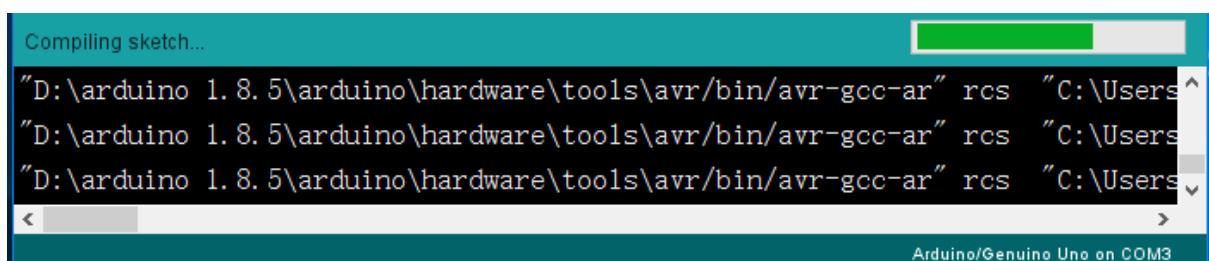
Nota: el tipo de tarjeta y puerto serial no son necesariamente los mismos que se muestran en la figura. Si usas la 2560, tendrás que seleccionar Mega 2560 como el tipo de tarjeta y las demás selecciones se pueden realizar de la misma forma. El puerto serial que se muestra es diferente en cada caso, a pesar de que se muestra el COM 3 aquí, podría ser el COM4 o COM8 en tu computadora. Un puerto COM correcto se supone que sea COMX (Arduino XXX) según el criterio de certificación.

El IDE de Arduino te mostrará la configuración actual de la tarjeta en la parte baja de la ventana.

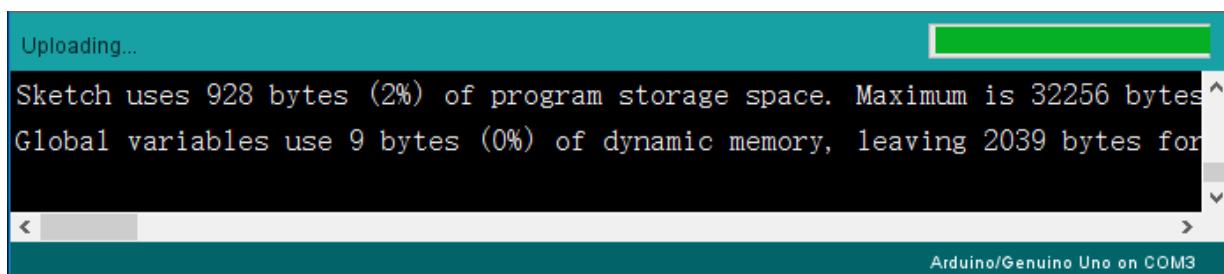


Haz clic en botón 'Upload'. En la barra de herramientas, es el segundo botón empezando desde la derecha, representado con una fecha que apunta hacia la derecha.

Si te fijas en el área de estatus del IDE, verás una barra de estado y una serie de mensajes. Al principio, dirá 'Compiling Sketch...'. Durante esta etapa se convierte el sketch a un formato adecuado para descargarlo en la tarjeta.



Después de eso, el estado cambiará a 'Uploading'. En ese punto, el LED del Arduino debería comenzar a encender de forma intermitente mientras se transfiere el sketch.

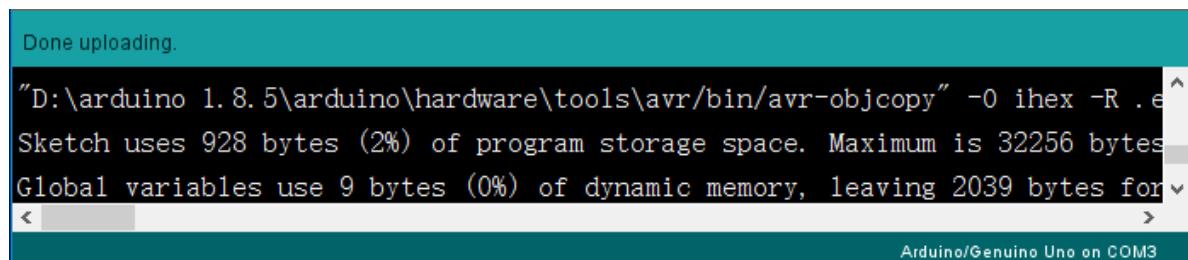


Uploading...

Sketch uses 928 bytes (2%) of program storage space. Maximum is 32256 bytes
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for

Arduino/Genuino Uno on COM3

Al terminar, el estatus cambiará a 'Done'.



Done uploading.
"D:\arduino 1.8.5\arduino\hardware\tools\avr/bin/avr-objcopy" -O ihex -R .e
Sketch uses 928 bytes (2%) of program storage space. Maximum is 32256 bytes
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for

Arduino/Genuino Uno on COM3

El otro mensaje nos dice que el sketch está utilizando 928 bytes de los 32,256 bytes disponibles. Si falla la carga, recibirás el siguiente mensaje después de 'Compiling Sketch...'!



Problem uploading to board. See <http://www.arduino.cc/en/Troubleshooting/ProgrammerNotResponding> Copy error messages
avrduude: stk500_recv(): programmer is not responding
avrduude: stk500_getsync(): attempt 10 of 10: not in sync: resp=0x22
Problem uploading to board. See <http://www.arduino.cc/en/Guide/Troubleshooting>

Arduino/Genuino Uno on COM1

Este es un error genérico y puede significar una de tres cosas: O tu tarjeta no está conectada, o no tienes instalados los drivers, o se seleccionó el puerto serial erróneo. Si te encuentras con esto, vuelve a la lección 0 y verifica tu instalación.

Una vez que la carga se haya completado, la tarjeta debería reiniciar y comenzar a destellar.

Instrucción: explica como trabaja el programa. Están allí para tu beneficio. Todo lo que aparece entre los símbolos /* y */ en la parte superior del sketch es un bloque de comentarios; explica para qué sirve el sketch.

Los comentarios de una sola línea comienzan con el símbolo // y todo lo que salga hasta al final de dicha

línea se considera parte del comentario.

Como se explica en el comentario de arriba, con esta línea se le está dando nombre al pin al cual se encuentra conectado el LED. Dicho pin es el 13 en la mayoría de los Arduino, incluyendo al UNO y el Leonardo.

Luego, tenemos la función de configuración o 'setup'. Nuevamente, tal como dice el comentario, esto se ejecuta cuando se presiona el botón reset. También se ejecuta cuando la tarjeta se reinicia por cualquier motivo, como cuando se enciende o cuando se ha terminado de cargar un sketch.

```
void setup() {  
    // inicializar el pin digital como una salida. pinMode(led,OUTPUT);  
}
```

Todo sketch de Arduino debe contar con una función de 'setup' y el sitio donde deberías colocar las instrucciones de la tuya es entre los símbolos { y }.

En este caso, solo hay un comando allí, el cual, de acuerdo a lo que dice el comentario le dice a la tarjeta de Arduino que vamos a usar el pin del LED como salida.

Es también obligatorio que los sketches tengan una función 'loop'. A diferencia de la función 'setup' que solo corre una vez. Después de reiniciar, la función 'loop' arrancará de nuevo de forma inmediata después de terminar de correr sus comandos.

```
void loop() {  
  
    digitalWrite(led, HIGH); delay(1000);  
    digitalWrite(led, LOW); delay (1000);  
  
    // enciende el LED (HIGH es el nivel de voltaje)  
    // espera un segundo  
    // apaga el LED llevando el voltaje a LOW  
    // espera un segundo  
  
}
```

Dentro de la función loop, los comandos encienden el LED primero que todo (HIGH), luego hacen un

'delay' (esperan) por 1000 milisegundos (1 segundo), para luego apagar el pin del LED y hacer una pausa de un segundo. Ahora harás que el LED destelle más rápido.

Como seguramente has adivinado, la clave es cambiar el parámetro dentro de los paréntesis () del comando 'delay'.

```
30 // the loop function runs over and over again forever
31 void loop() {
32   digitalWrite(LED_BUILTIN, HIGH);    // turn the LED on (HIGH is the volt
33   delay(500)                      // wait for a second
34   digitalWrite(LED_BUILTIN, LOW);   // turn the LED off by making the vo
35   delay(500)                      // wait for a second
36 }
```

Este periodo es en milisegundos, por tanto si quieras que el LED destelle dos veces más rápido, cambia el valor de 1000 a 500. Esto hace que el programa se pause solamente medio segundo entre cada cambio de estado.

Vuelve a cargar el sketch y verás como el LED comienza a destellar más rápidamente.

Bajo el entorno Mac

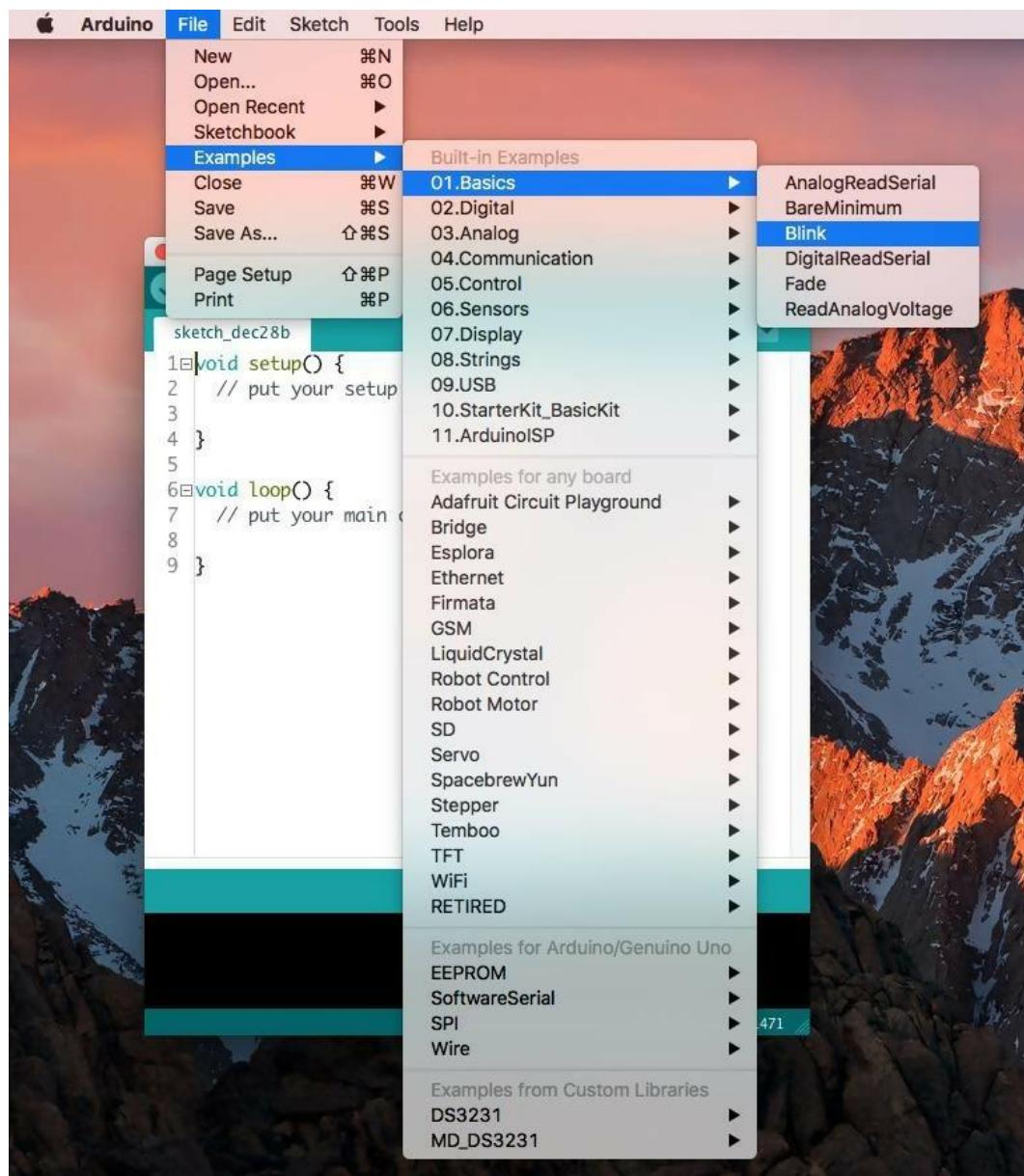
Como seguramente notarás, el LED "L" se ilumina cuando conectas la tarjeta a un puerto USB. Esto se debe a que las tarjetas se envían precargadas con el sketch "Blink" (o destello) instalado por defecto.

En esta lección, reprogramaremos la tarjeta UNO R3 con nuestro propio sketch de destello para después modificar la velocidad a la cual se ilumina.

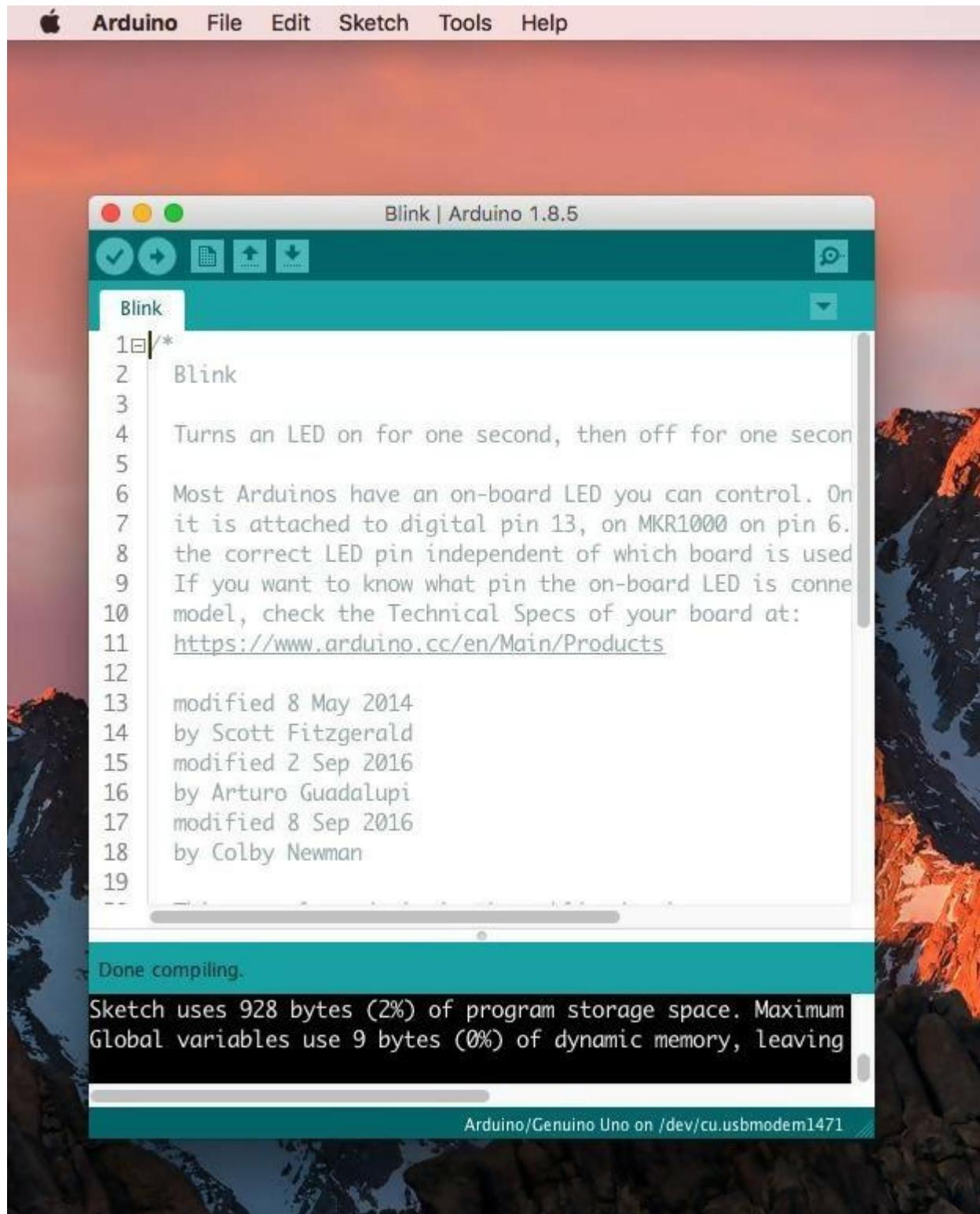
En la lección 0, configuraste tu IDE de Arduino y te aseguraste encontrar el puerto serial adecuado para conectarse con tu UNO R3. Pues ahora llegó el momento que pruebas esa conexión y programmes tu tarjeta UNO R3.

El IDE de Arduino incluye una colección bastante grande de sketches de ejemplos para cargarlos y usarlos. Entre esos ejemplos está incluido un sketch para hacer destellar el LED "L".

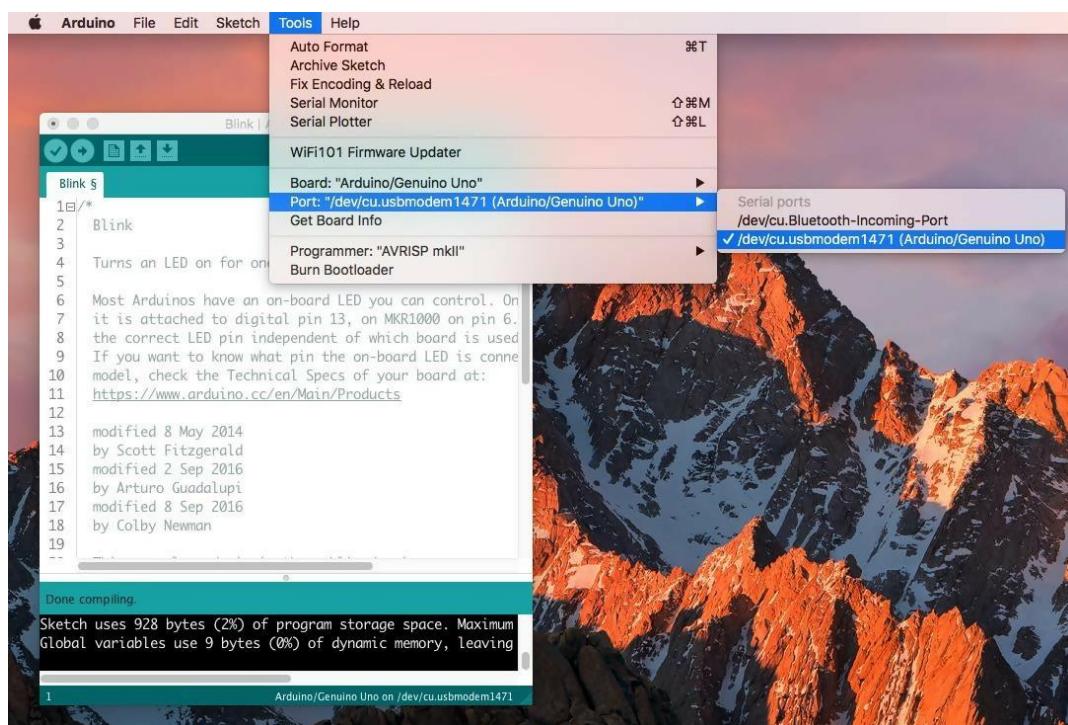
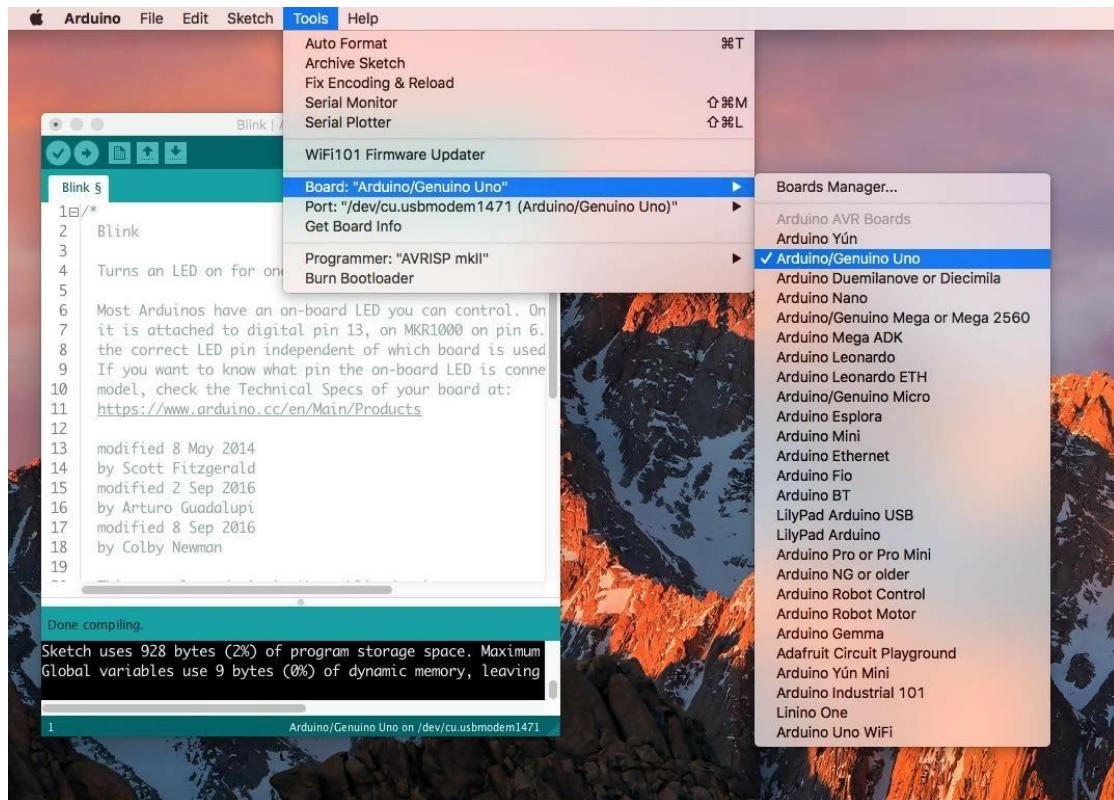
Carga el sketch "Blink" que encontrarás en el siguiente menú del IDE: File > Examples > 01. Basics



Cuando abre la ventana del sketch, agrándala para que puedas ver el sketch por completo en la ventana.

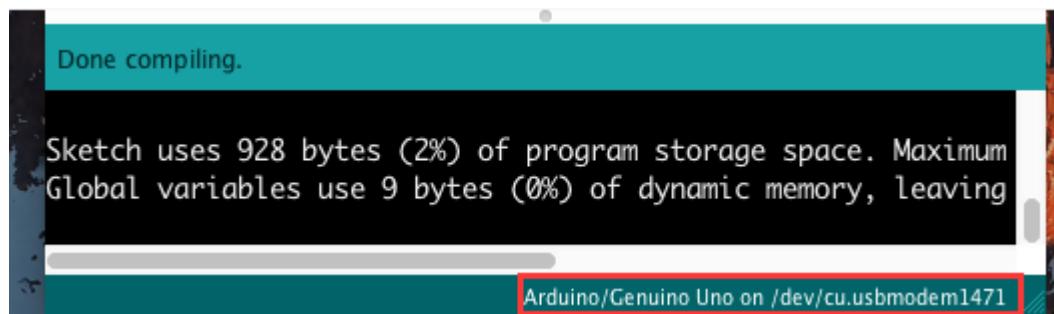


Conecta tu tarjeta Arduino a tu computadora con el cable USB y verifica que el tipo de tarjeta ('Board Type') y el puerto serial ('Serial Port') estén configurados correctamente.



Nota: el tipo de tarjeta y puerto serial no son necesariamente los mismos que se muestran en la figura. Si usas la 2560, tendrás que seleccionar Mega 2560 como el tipo de tarjeta y las demás selecciones se pueden realizar de la misma forma. Y el puerto serial se muestra para /dev/cu.usbmodem1471 (Arduino/Genuino Uno). Todos son diferentes, si no es el escogido acá, puede ser también /dev/cu.usbmodemfa121(Arduino Uno) en tu computadora. Un puerto COM correcto debe ser /dev/cu.usbmodemfaXX (Arduino XXX) , el cual corresponde con el criterio de certificación.

El IDE de Arduino te mostrará la configuración actual de la tarjeta en la parte baja de la ventana.

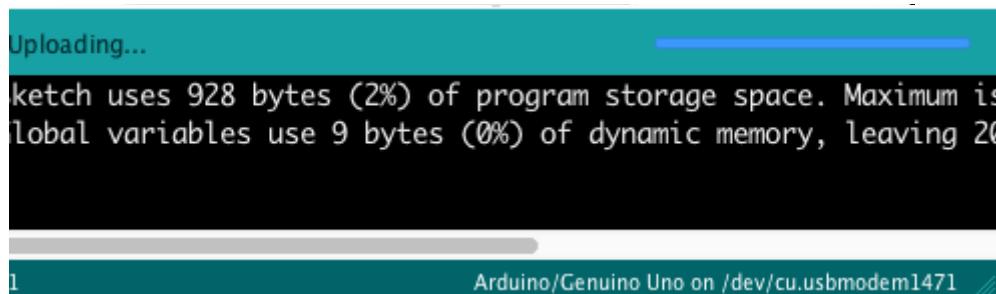


Haz clic en botón 'Upload'. . Es el segundo botón en la barra de herramientas, mirando desde la izquierda.

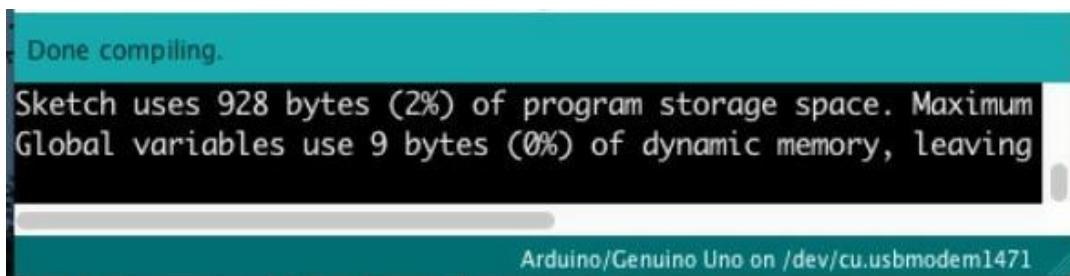


Si te fijas en el área de estatus del IDE, verás una barra de estado y una serie de mensajes. Al principio, dirá 'Compiling Sketch...'. Durante esta etapa se convierte el sketch a un formato adecuado para descargarlo en la tarjeta.

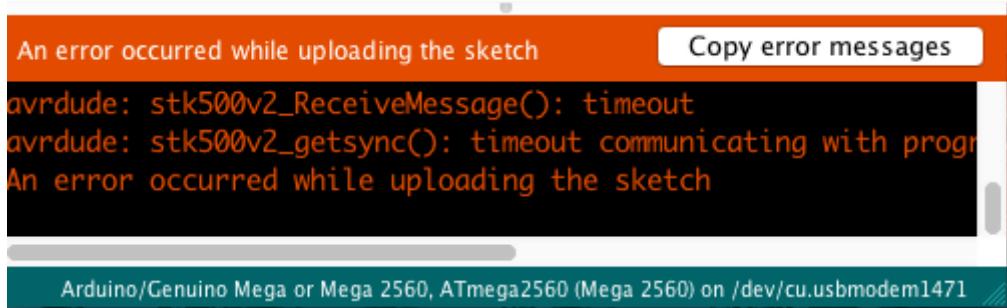
Después de eso, el estado cambiará a 'Uploading'. En ese punto, el LED del Arduino debería comenzar a encender de forma intermitente mientras se transfiere el sketch.



Al terminar, el estatus cambiará a 'Done'.



El otro mensaje nos dice que el sketch está utilizando 928 bytes de los 32,256 bytes disponibles. Podrías recibir el siguiente mensaje de error después de la fase 'Compiling Sketch...'.



Puede significar que la tarjeta no está conectada, que no se han instalado los drivers (de ser necesarios) o que se ha seleccionado el puerto serial errado.

Si te encuentras con esto, vuelve a la lección 0 y verifica tu instalación.

Una vez que la carga se haya completado, la tarjeta debería reiniciar y comenzar a destellar. Abre el código

Nota que una gran parte de este sketch está compuesto por comentarios. Los mismos no son parte del programa propiamente dicho; solo explican cómo funciona el programa. Están allí para tu beneficio. Todo lo que aparece entre los símbolos /* y */ en la parte superior del sketch es un bloque de comentarios; explica para qué sirve el sketch.

Los comentarios de una sola línea comienzan con el símbolo // y todo lo que salga hasta al final de dicha línea se considera parte del comentario.

La primera línea de código es: int led = 13;

Como se explica en el comentario de arriba, con esta línea se le está dando nombre al pin al cual se encuentra conectado el LED. Dicho pin es el 13 en la mayoría de los Arduino, incluyendo al UNO y el Leonardo.

Luego, tenemos la función de configuración o 'setup'. Nuevamente, tal como dice el comentario, esto se ejecuta cuando se presiona el botón reset. También se ejecuta cuando la tarjeta se reinicia por cualquier motivo, como cuando se enciende o cuando se ha terminado de cargar un sketch.

```
void setup() {  
// inicializar el pin digital como una salida. pinMode(led, OUTPUT);  
}
```

Todo sketch de Arduino debe contar con una función de 'setup' y el sitio donde deberías colocar las instrucciones de la tuya es entre los símbolos { y }.

En este caso, solo hay un comando allí, el cual, de acuerdo a lo que dice el comentario le dice a la tarjeta de Arduino que vamos a usar el pin del LED como salida.

Es también obligatorio que los sketches tengan una función 'loop'. A diferencia de la función 'setup' que solo corre una vez (después de reiniciar) la función 'loop' arrancará de nuevo de forma inmediata después de terminar de correr sus comandos.

```
void loop() {
```

```
digitalWrite(led, HIGH); delay(1000);
digitalWrite(led, LOW); delay(1000);
}

// enciende el LED (HIGH es el nivel de voltaje)
// espera un segundo
// apaga el LED llevando el voltaje a LOW
// espera un segundo
```

Dentro de la función `loop`, los comandos encienden el LED primero que todo (HIGH), luego hacen un 'delay' (esperan) por 1000 millisegundos (1 segundo), para luego apagar el pin del LED y hacer una pausa de un segundo.

Ahora harás que el LED destelle más rápido. Como seguramente has adivinado, la clave es cambiar el parámetro dentro de los paréntesis () del comando 'delay'.

```
30 // the loop function runs over and over again forever
31 void loop() {
32   digitalWrite(LED_BUILTIN, HIGH);    // turn the LED on (HIGH is the volt
33   delay(500);                      // wait for a second
34   digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the vo
35   delay(500);                      // wait for a second
36 }
```

Este periodo es en milisegundos, por tanto si quieres que el LED destelle dos veces más rápido, cambia el valor de 1000 a 500. Esto hará que cada retardo sea de medio segundo y no de un segundo entero.

Vuelve a cargar el sketch y verás como el LED comienza a destellar más rápidamente.

Lección 4 Módulo de Temperatura y Humedad

En este tutorial aprenderemos a usar un sensor de temperatura y humedad DHT11. Es lo suficientemente preciso para la mayoría de los proyectos que requieren registros de lecturas de temperatura y humedad.

Utilizaremos una librería específicamente diseñada para estos sensores a fin de que nuestro código sea corto y fácil de escribir.

Temperatura y Humedad

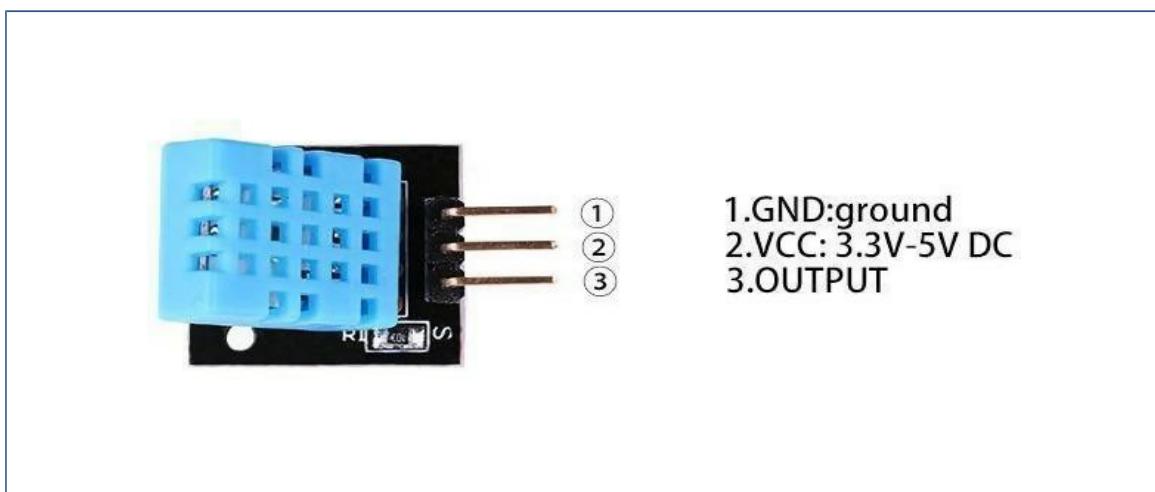
Un módulo con un sensor de temperatura/humedad tipo DHT1.

Rango de temperatura: -20 - 60°C (+/-1°C), Humedad Rel. : 5-95% (+/5%),

Voltaje de alimentación: 3 a 5.5V.

Con un resistor tipo pull up de 10 K ohm

Librería: DHT.h



Componentes requeridos:

1 x Elegoo Uno R3

1x Módulo de Temperatura y Humedad

3 x cables F-M (Cables DuPont Hembra a Macho)

Introducción del componente sensor de temperatura y humedad:

El sensor digital de temperatura y humedad DHT11 es un sensor de tipo compuesto que contiene una señal digital de salida calibrada que muestra la temperatura y humedad. Los módulos digitales especialmente diseñados con tecnología de recolección de datos y la tecnología de medición de temperatura y humedad se aplican para que el producto tenga alta confiabilidad y una excelente estabilidad a largo plazo. El sensor está formado por componentes húmedos y dispositivos de medición de temperatura tipo NTC, conectados con un microcontroladores de 8-bits de alto desempeño.

Aplicaciones: HVAC, deshumidificadores, equipos de medición e inspección, bienes de consumo, automotrices, control automático, registradores de datos, estaciones meteorológicas, electrodomésticos, reguladores de humedad, medición y control de humedad para aplicaciones médicas y generales, etc.

Parámetros del producto

Humedad Relativa:

Resolución: 16 Bit

Repetibilidad: $\pm 1\%$ RH

Precisión: A $25^{\circ}\text{C} \pm 5\%$ RH

Intercambiabilidad: Totalmente intercambiable

Tiempo de respuesta: 1 / e (63%) of 25°C 6s / 1m / s air 6s

Histéresis: $<\pm 0.3\%$ RH

Estabilidad a largo plazo: $<\pm 0.5\%$ RH / yr in

Temperatura:

Resolución: 16 Bit

Repetibilidad: $\pm 0.2^{\circ}\text{C}$

Rango: A $25^{\circ}\text{C} \pm 2^{\circ}\text{C}$

Tiempo de respuesta: 1 / e (63%) 10S

Características eléctricas

Fuente de poder: DC 3.5 ~ 5.5V

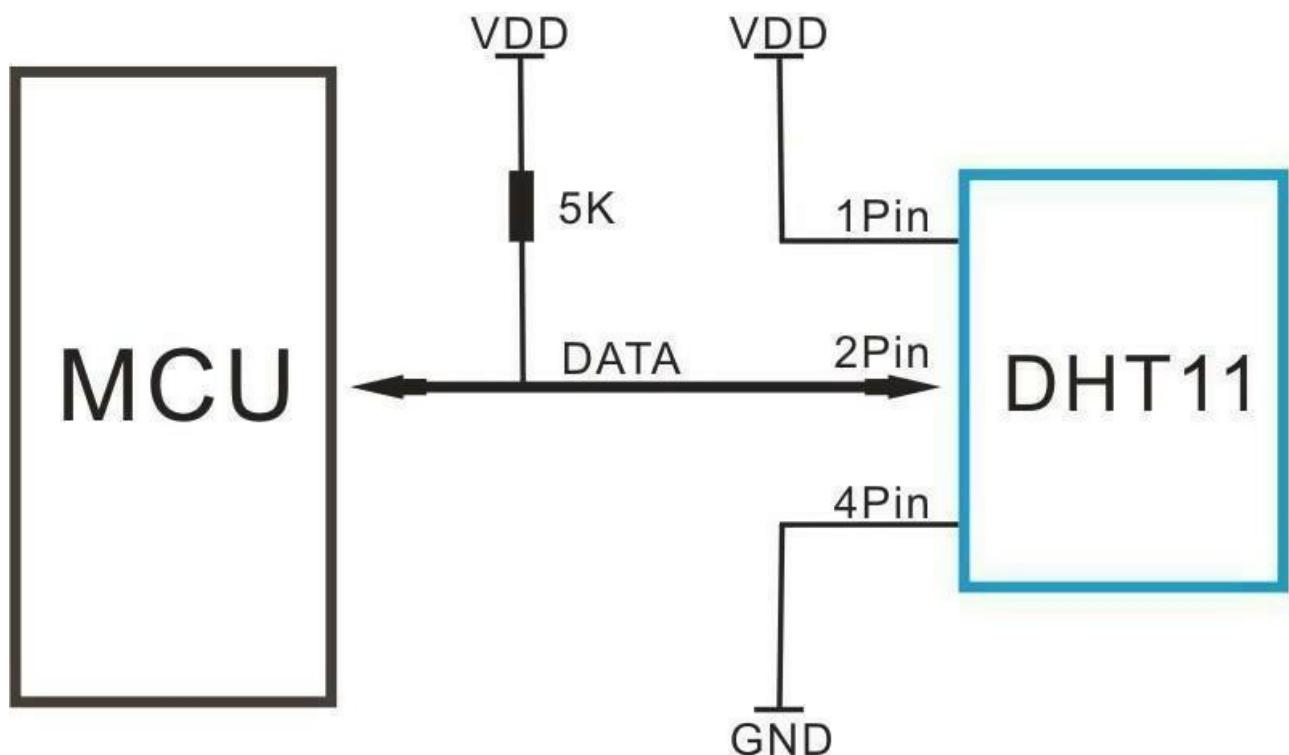
Corriente de suministro: en medición 0.3mA en espera 60 μ A

Periodo de muestreo: más de 2 segundos

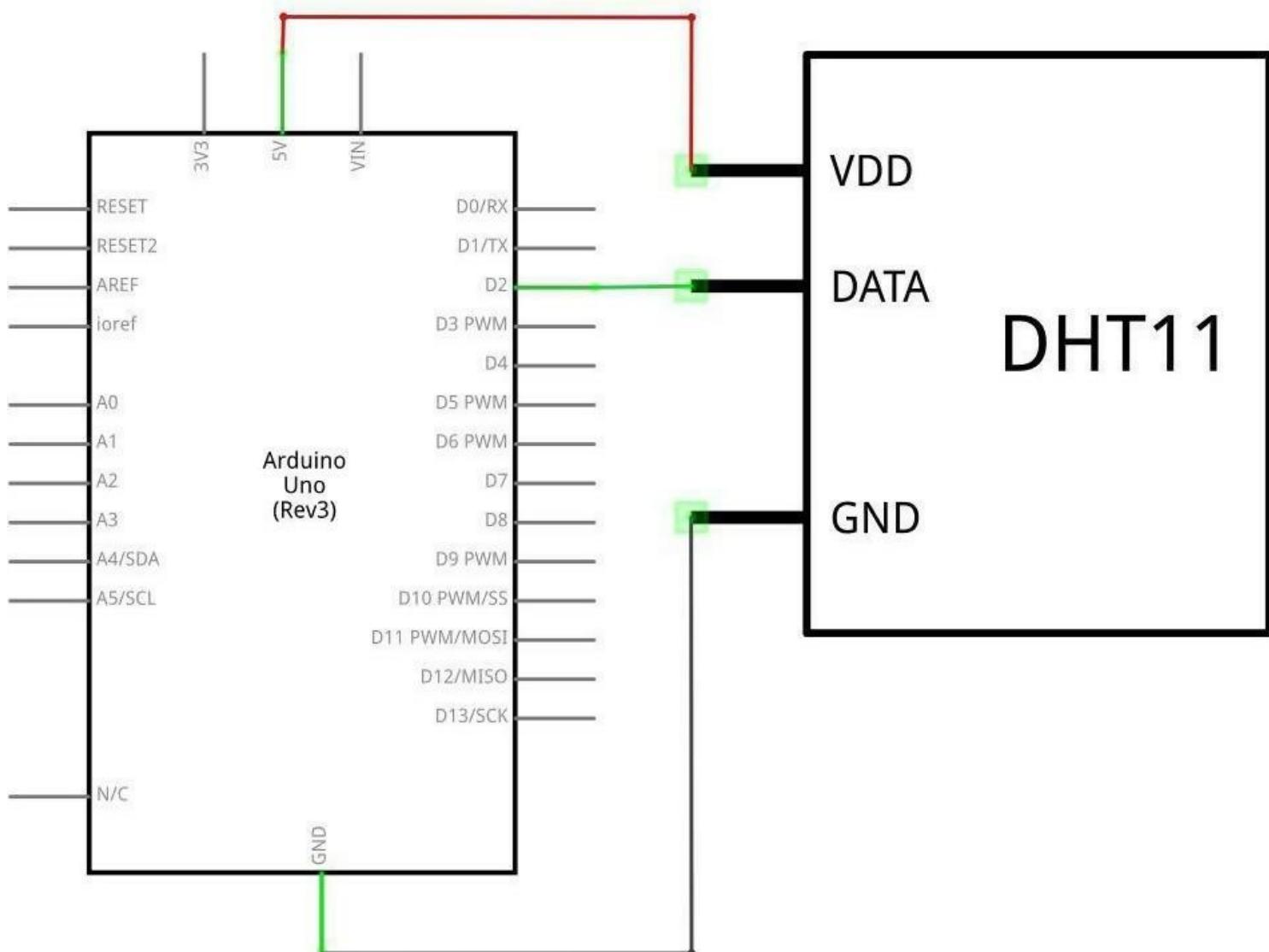
Descripción de Pines:

1. La fuente de poder VDD 3.5~5.5VDC
2. DATA datos seriales, un bus sencillo
3. NC, pin vacío
4. GND tierra, punto negativo de la fuente

Aplicaciones típicas



Esquema de conexiones



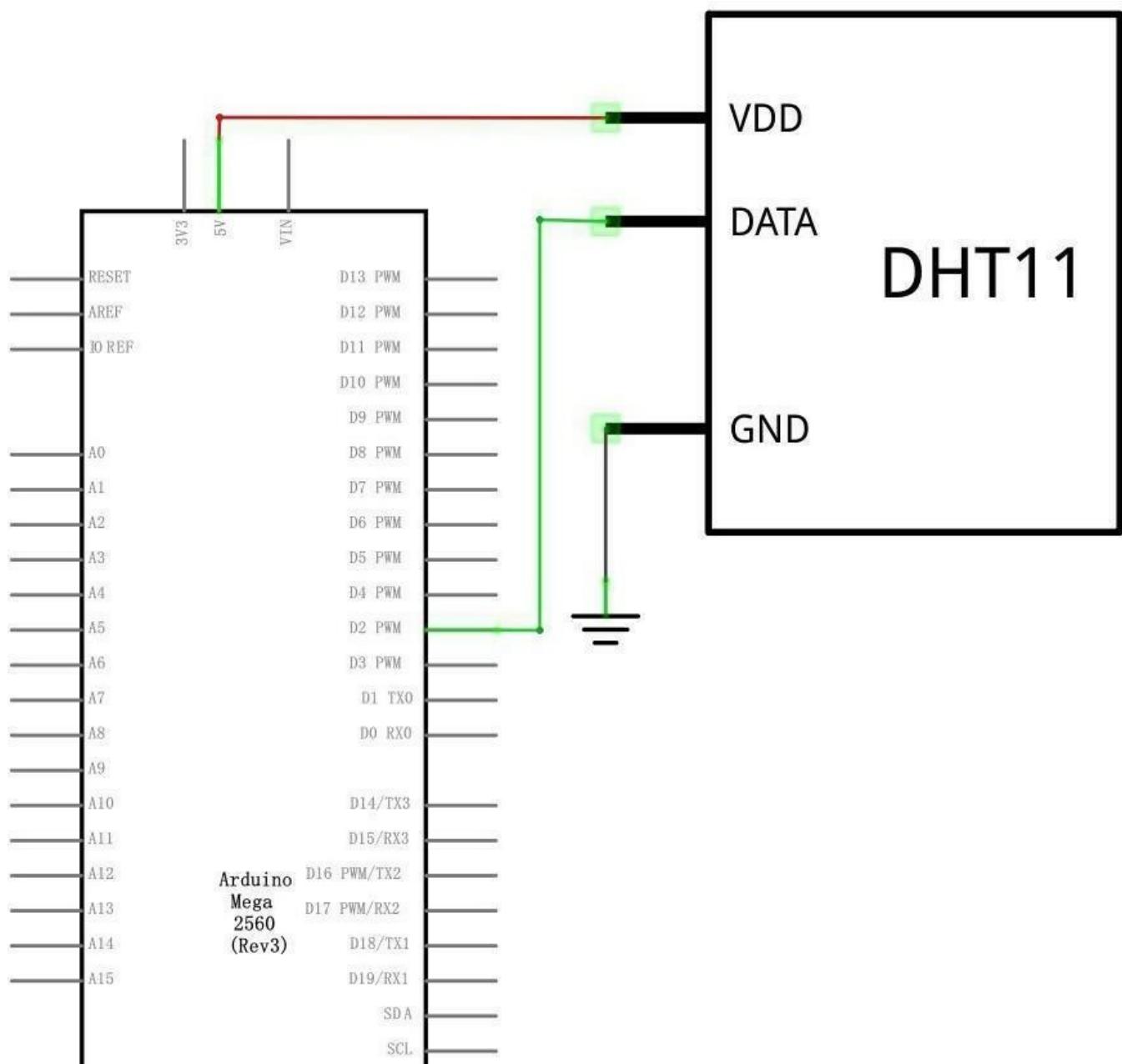
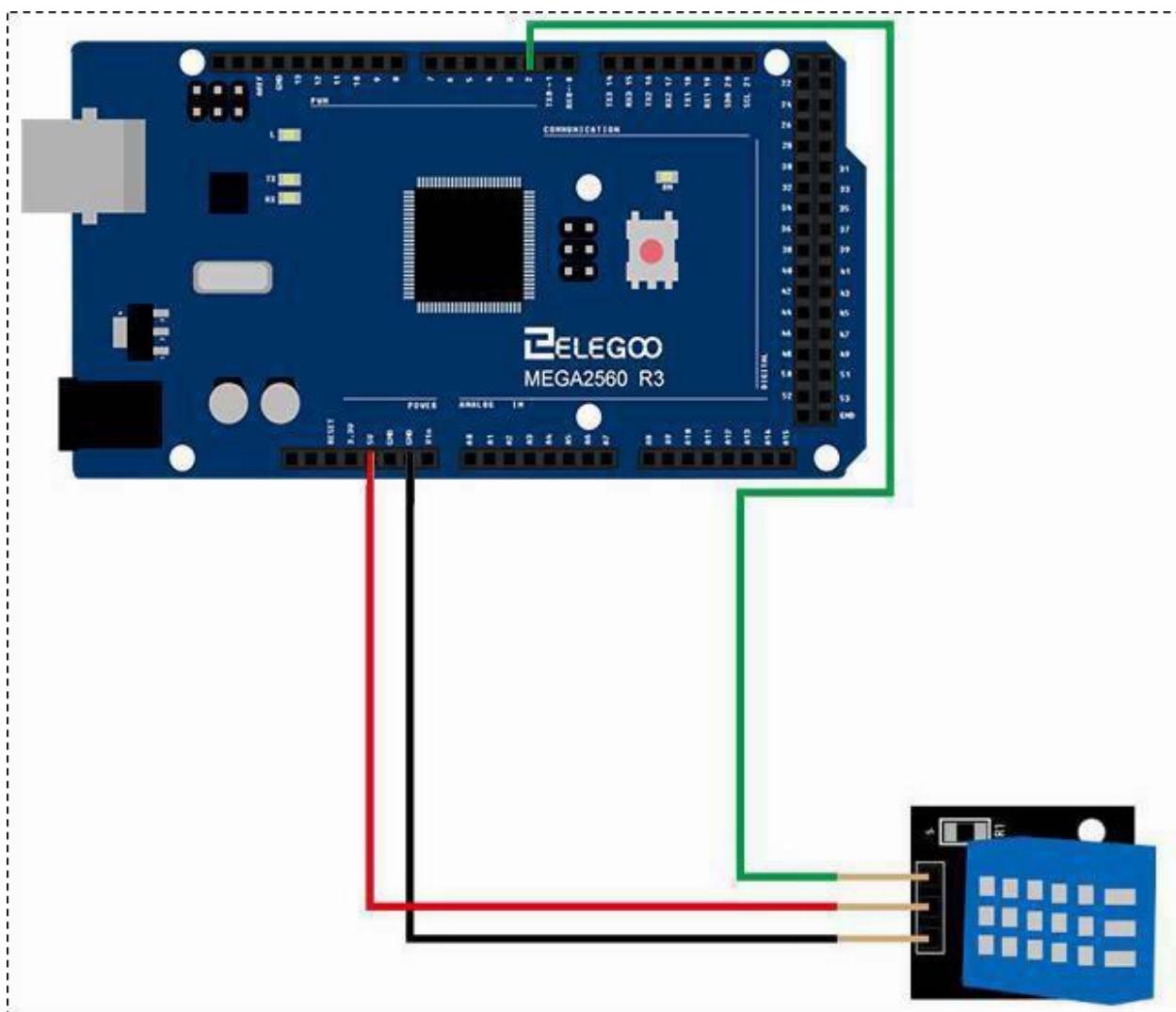
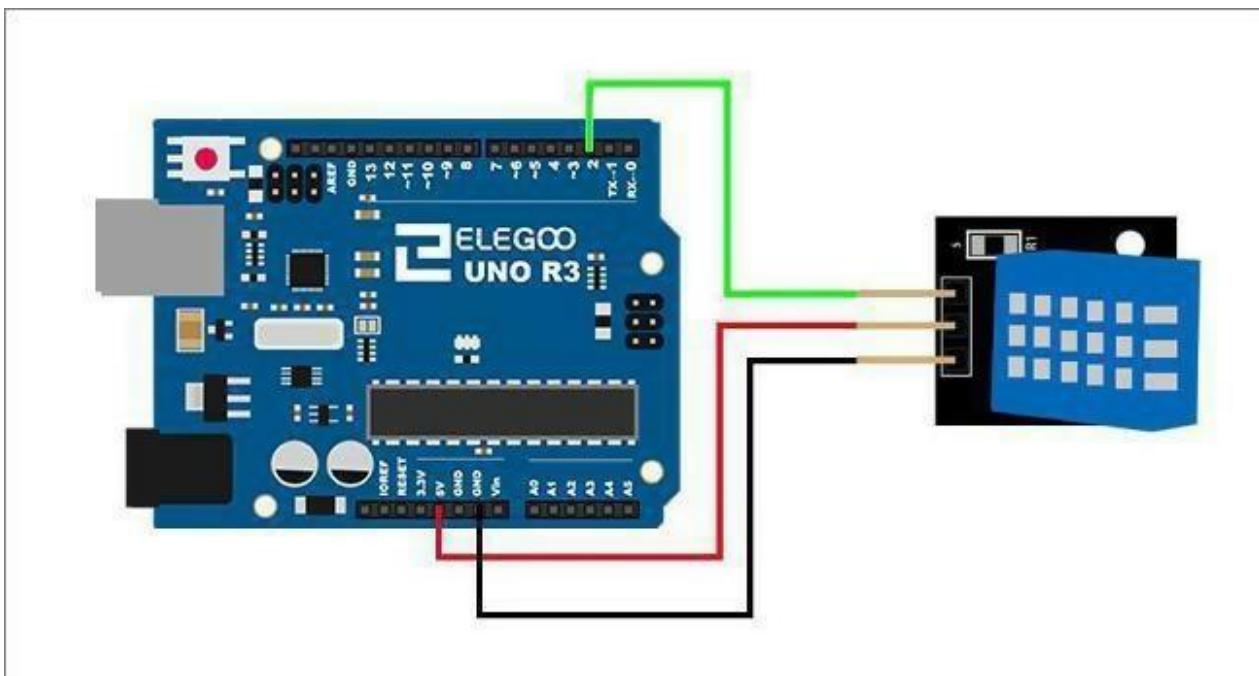


Diagrama de cableado



Tal como puedes ver, solo necesitamos 3 conexiones al sensor, debido a que uno de los pines no se utiliza.

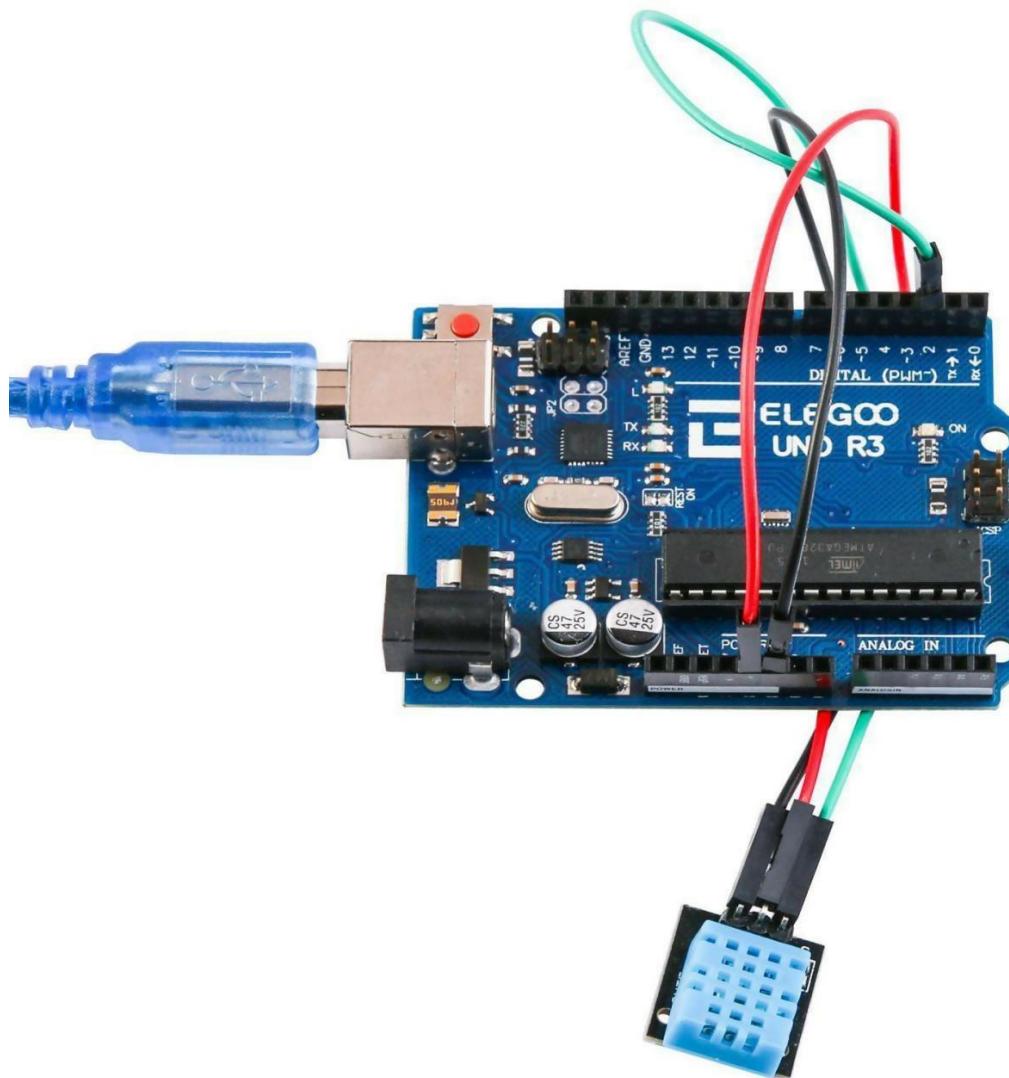
Las conexiones son: voltaje, tierra y señal, la cual puede ser conectada a cualquier pin de nuestro UNO.

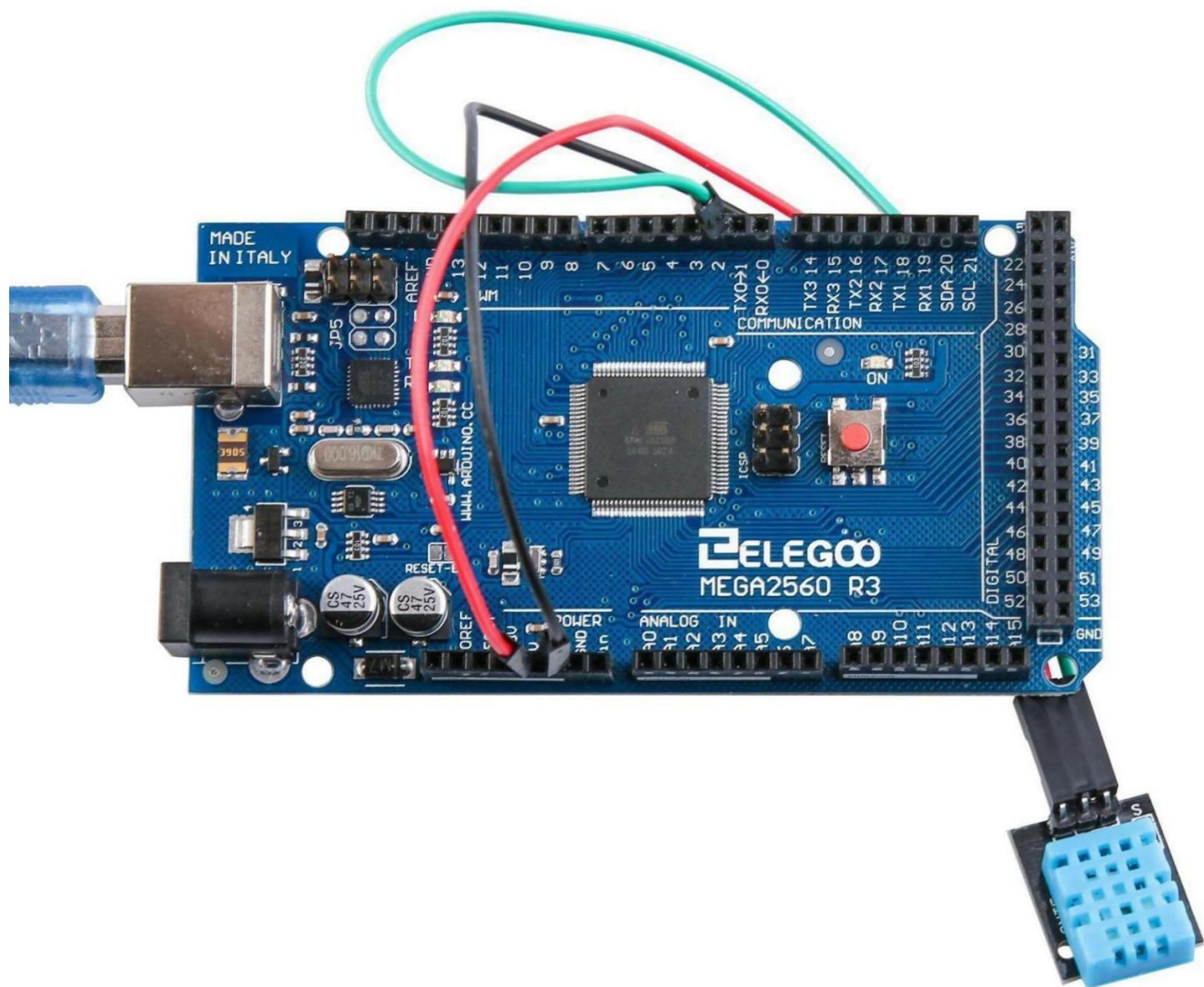
Código

Después de realizar el cableado, por favor abre el programa en la carpeta de código (Lección 4 Módulo de Temperatura y Humedad) y haz clic en UPLOAD para cargar el programa. Si hay algún error, consulta los detalles en la lección 2 para cargar el programa.

Antes de poder correrlo, asegúrate de tener la librería < DHT> instalada, o reinstálala de ser necesario. De otra forma, tu código no funcionará.

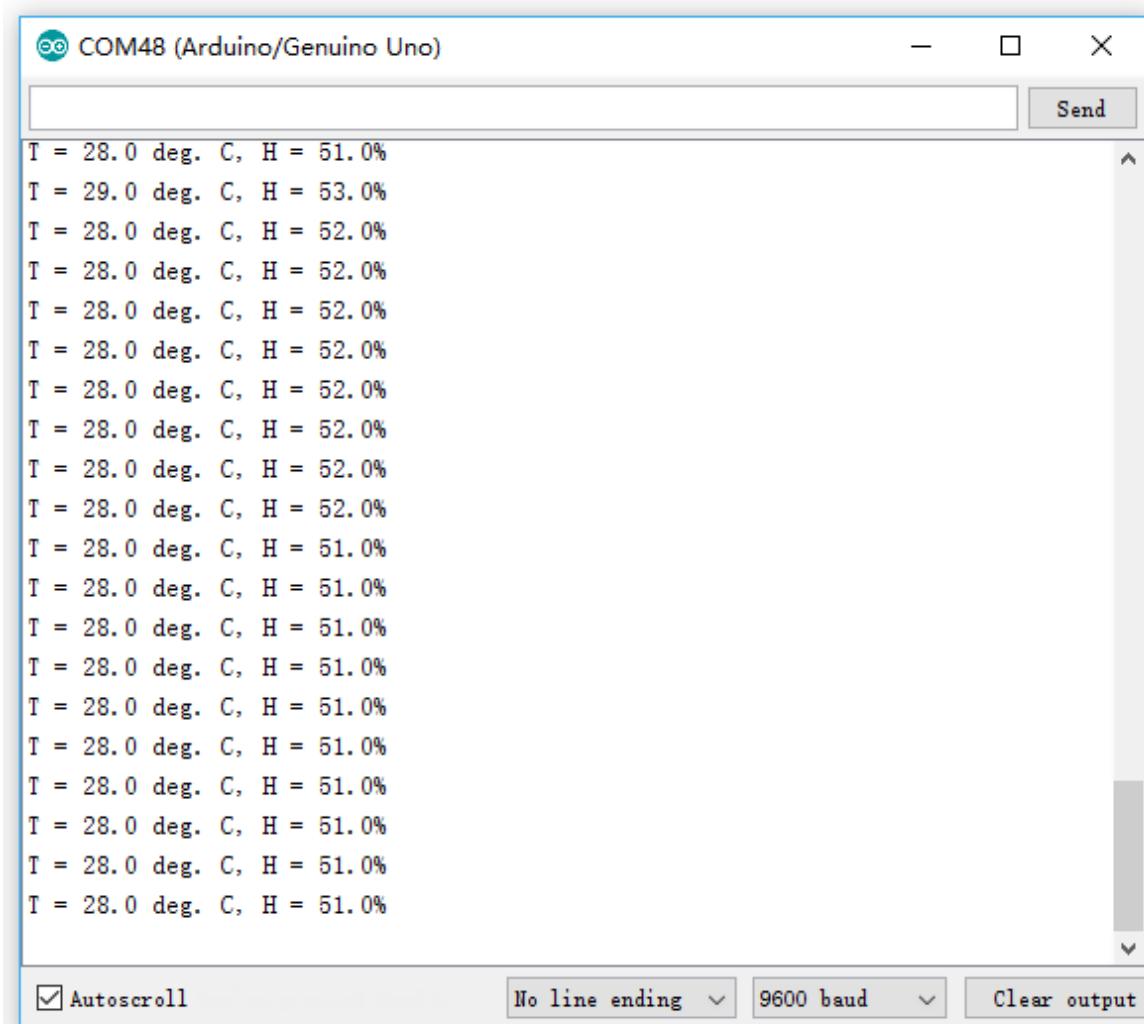
Para detalles del tutorial acerca de cómo cargar los archivos de librerías, consulta la lección 1.





Carga el programa y abre el monitor, se podrán ver los datos como aparecen abajo: (se muestra la temperatura ambiental, en este caso de 28 grados)

Haz clic en el botón del Monitor Serial para encenderlo. En la lección 2 se explica en detalle como utilizarlo.



A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
#include<dht_nonblocking.h>

#define DHT_SENSOR_TYPE DHT_TYPE_11
//#define DHT_SENSOR_TYPE DHT_TYPE_21
//#define DHT_SENSOR_TYPE DHT_TYPE_22

static const int DHT_SENSOR_PIN = 2;
DHT_nonblocking dht_sensor( DHT_SENSOR_PIN, DHT_SENSOR_TYPE );

void setup( )
{
    Serial.begin( 9600 );
}

void loop( )
{
    if(dht_sensor.measure(&temperatu
        re, &humidity)){ Serial.print( "T
        = " );
        Serial.print( t
        emperature,
        1
        );
        Serial.print(
        " deg.C,H="
```

```
 );
Serial.print(  
    humidity,  
    1      );
Serial.printl  
    n( "%" );
}  
}
```


A partir del programa mostrado arriba, es posible aprender la sintaxis de la estructura de control de la programación.

(1) if

if prueba si cierta condición se ha alcanzado, como por ejemplo si una entrada está por encima de cierto número. El formato para una prueba if es como sigue:

```
if (algunaVariable > 50) {      // Haz algo aqui }
```

El programa prueba si el valor de algunaVariable es mayor que 50. Si es así, el programa ejecuta una acción particular. Para decirlo de otra forma, si lo que está entre paréntesis (el enunciado) es verdadero, se ejecutará entonces lo que aparece entre corchetes. Si no, el programa se salta el código.

Los corchetes después de un enunciado if pueden ser omitidos. Si se omiten, la siguiente línea (definida por el punto y coma) se convertirá en el único enunciado condicional.

```
if (x > 120) digitalWrite(LEDpin, HIGH);  
if (x > 120) digitalWrite(LEDpin, HIGH);  
if (x > 120) {digitalWrite(LEDpin, HIGH);} //
```

Todos son correctos

Los enunciados que se evalúan dentro de los paréntesis requieren del uso de uno o más operadores:

Operadores:

| | |
|--------|----------------------------|
| x == y | (x es igual a y) |
| x != y | (x es diferente de y) |
| x < y | (x es menor que y) |
| x > y | (x es mayor que y) |
| x <= y | (x es menor o igual que y) |
| x >= y | (x es mayor o igual que y) |

Advertencia

Ten cuidado de no utilizar accidentalmente el símbolo de igualdad sencillo (ejemplo: if ($x = 10$)). Este símbolo es un operador de asignación, es decir, determina que el valor de x es una constante, en este caso 10. En su lugar, utiliza el símbolo de igualdad doble (ejemplo: if ($x == 10$)), que es el operador de comparación, para probar si la variable x es igual a 10 o no. El último enunciado es verdadero sí y solo si x es igual a 10; el anterior siempre será verdadero. Esto debido a que C evalúa el enunciado if ($x=10$) de la siguiente manera: 10 está asignado a x , así que x contiene 10. Luego, el "if" condicional evalúa 10, dándolo siempre como verdadero, ya que todo número diferente de cero es verdadero. En consecuencia if ($x = 10$) siempre será evaluado como TRUE, lo que no es deseable cuando se requiere utilizar un enunciado "if". Adicionalmente, la variable x será definida como una constante de valor 10, lo que tampoco es una acción deseable.

if también puede formar parte de una estructura de control entrelazada, utilizando la construcción if...else].

(2) Enunciado "for"

El enunciado "for" se utiliza para repetir un bloque de enunciados encerrados entre llaves. Un contador de incremento se usa por lo general para monitorear los incrementos y terminar el lazo.

El enunciado "for" es útil para cualquier operación repetitiva, y se usa frecuentemente en combinación con arreglos para operar recolección de datos desde los pines. El encabezado del lazo "for" consta de tres partes: for (inicialización; condición; incremento) { //enunciado(s); }

La inicialización sucede al principio y solo una vez. Cada vez que se cumple una vuelta del lazo se evalúa la condición; si es verdadera, se ejecuta el bloque de enunciado y el incremento, para volver a probar la condición al continuar. Cuando la condición se hace falsa, se termina el lazo.

Ejemplo

```
// Baja la intensidad de un LED utilizando un pin PWM pin int PWMPin = 10; // LED en serie con resistor de
1k en pin 10 void setup()
{
    // no es necesario configurar } void loop()
{
    for (int i=0; i <= 255; i++)
    {
        analogWrite(PWMPin, i);
        delay(10);
    }
}
```

Tip de programación

El loop for de lenguaje C es mucho más flexible que los de otros lenguajes de programación, incluyendo a BASIC. Cualquiera de los tres elementos del encabezado puede ser omitido, aunque se requiere utilizar punto y coma. De igual manera, los enunciados para inicialización, condición e incremento pueden ser cualquier enunciado válido para C con variables no relacionadas. Estos raros tipos de enunciado "for" inusuales pueden proporcionar soluciones a extraños problemas de programación.

Lección 5 Módulo Sensor de Temperatura DS18B20

En este experimento, aprenderemos a utilizar el módulo DS18B20 para medir la temperatura ambiental y hacer un termómetro.

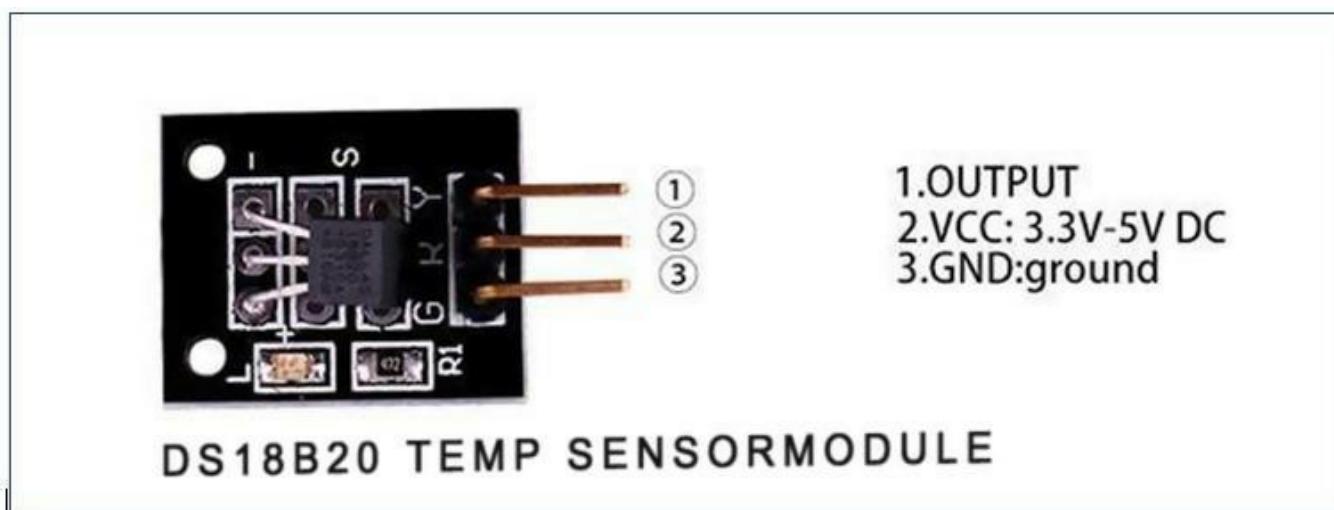
Debido a que la salida del anterior sensor de temperatura es analógica, será necesario agregar un chip A/D y D/A al circuito. Esto será todo un desafío. Así que vamos a crear el módulo Ds18b20

El nuevo módulo sensor de temperatura DS18B20 es muy bueno para solucionar el problema. es barato, tiene un bus de un solo cable y es totalmente compatible con la plataforma Arduino. Los usuarios pueden hacer una red de manera sencilla con este módulo.

Sensor de Temperatura DS18B20

Es un módulo con un sensor de temperatura digital tipo 'One Wire' (DS18B20). Un resistor tipo pullup de 4.7K ohm se incluye para la señal del bus. Se pueden agregar sensores adicionales al bus y direccionarlos de forma individual. Solo se debe conectar un sensor tipo pullup al bus, sin importar la cantidad de sensores que se conecten al mismo.

- Rango de temperatura: -55 a +125°C
- Precisión típica: 0.5°C
- Resolución: 9-12Bit, dependiendo del programación

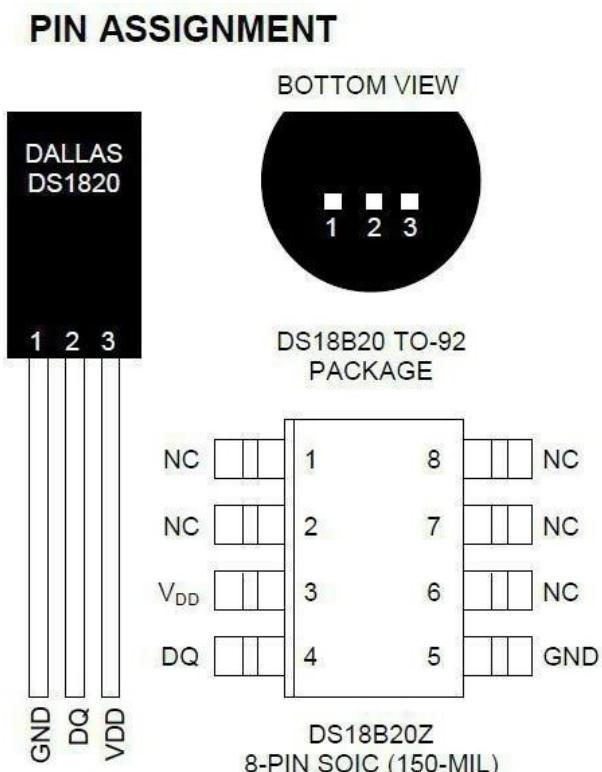


Componentes requeridos:

1 x Elegoo Uno R3 1 x cable USB

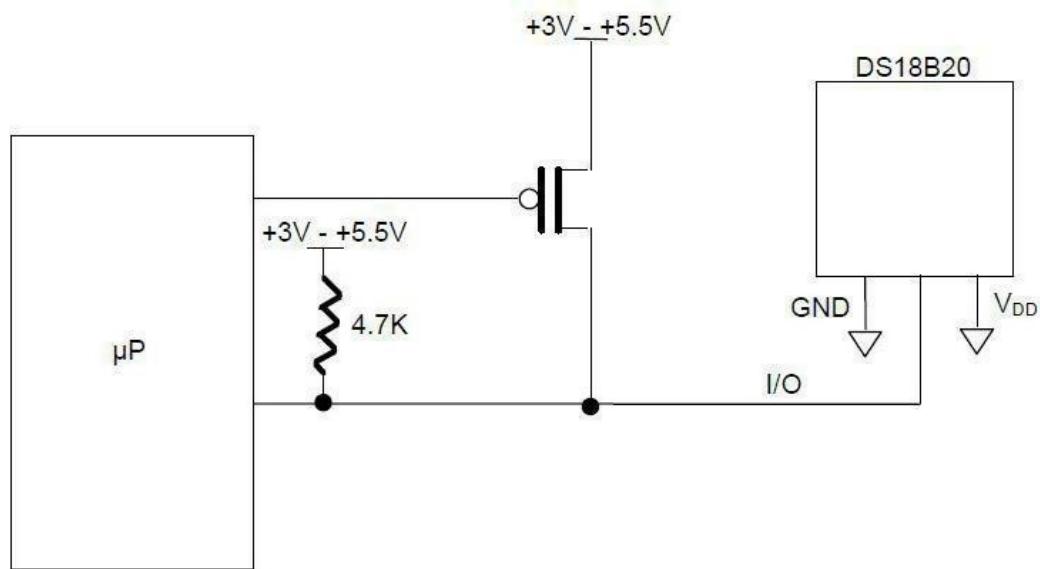
1 x módulo DS18B20 3 x cables F-M

Introducción del componente DS18B20:



PIN DESCRIPTION

| | |
|-----------------|------------------------|
| GND | - Ground |
| DQ | - Data In/Out |
| V _{DD} | - Power Supply Voltage |
| NC | - No Connect |



Principio

El módulo DS18B20 utiliza un bus sencillo. El rango de la fuente de poder de 3.0 V a 5.5 V sin fuente de poder para tiempo de espera.

Puede medir temperatura en un rango desde -55 grados a +125 grados, con una precisión de +/-0.5°C.

El sensor de temperatura posee un DPI programable puede ajustarse entre 9 y 12. La temperatura propiamente dicha se representa mediante 12 bits y puede registrarse a una tasa máxima de una vez cada 750 milisegundos.

Cada DS18B20 contiene un número de serie único, de forma que puede haber múltiples chips DS18B20 en un bus.

Esquema de conexiones

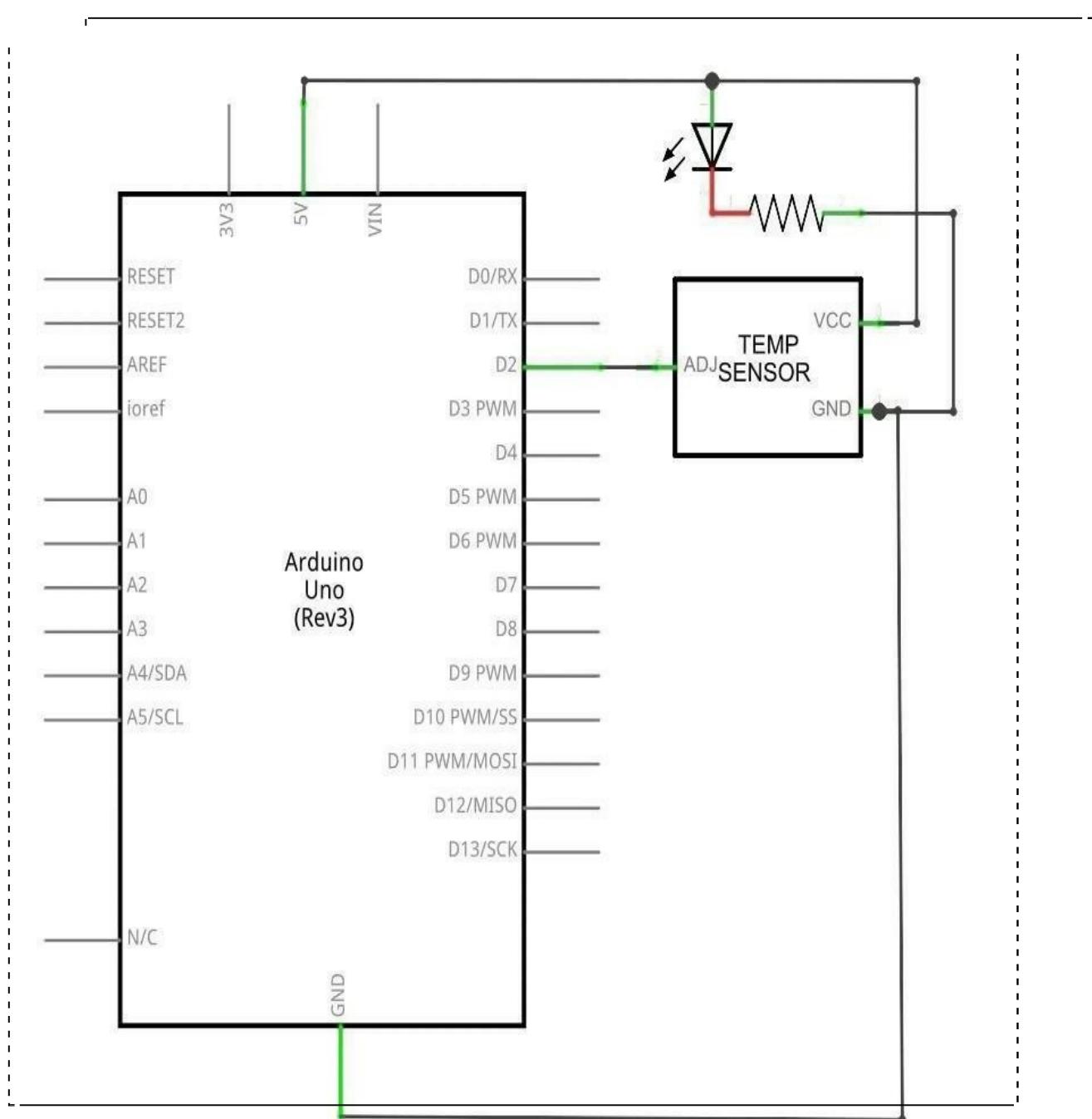
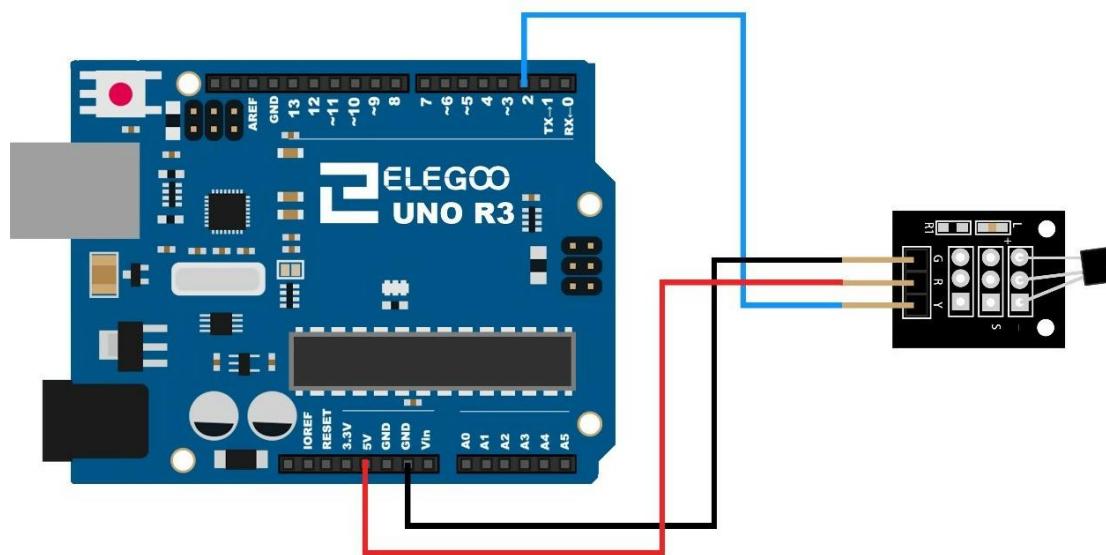
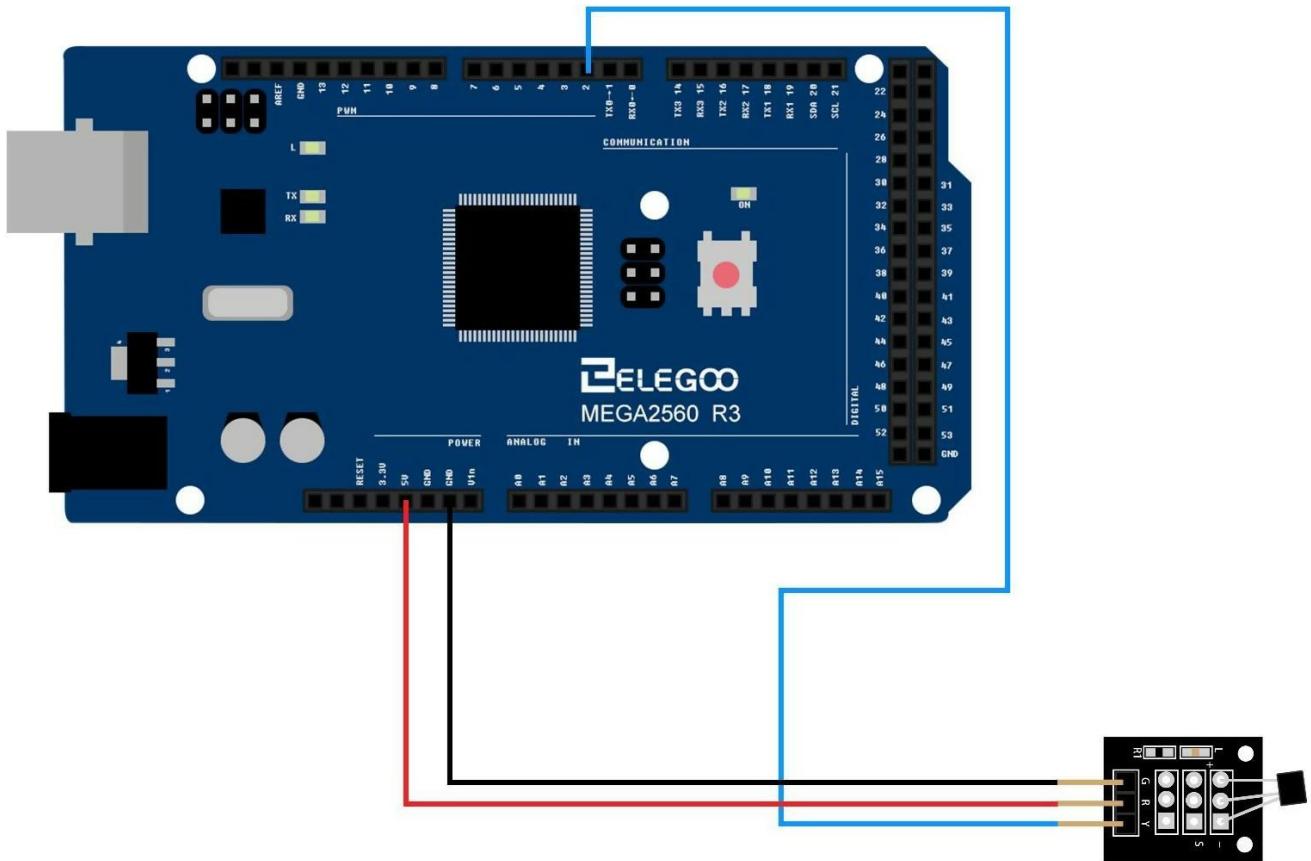


Diagrama de cableado





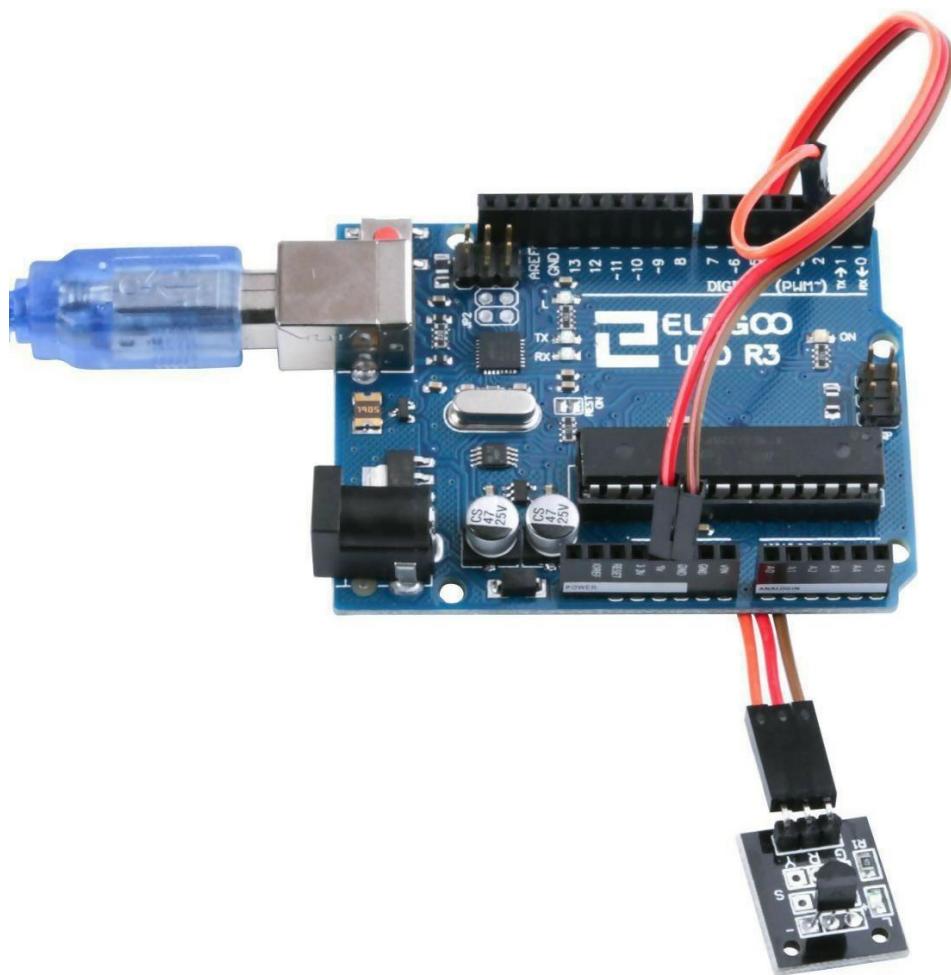
Código

Después de realizar el cableado, por favor abre el programa en la carpeta de código Lección 5 Módulo de sensor de temperatura digital DS18B2) y haz clic en UPLOAD para cargar el programa. Si hay algún error, consulta los detalles en la lección 2 para cargar el programa.

Antes de poder correrlo, asegúrate de tener la librería <Dallas Temperature> y la <OneWire> instaladas, o reinstálalas de ser necesario. De otra forma, tu código no funcionará.

Para detalles del tutorial acerca de cómo cargar los archivos de librerías, consulta la lección 1.

Resultados



Este sensor de temperatura puede ejecutar mediciones en diferentes lugares al mismo tiempo. Carga el programa y abre el monitor, se podrán ver los datos como aparecen abajo:

A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
/* Incluye las librerías que necesitamos*/ #include <OneWire.h>
#include <DallasTemperature.h>

/* El cable de datos esté enchufado al puerto 2 del Arduino*/ #define ONE_WIRE_BUS 2

/* Configura una instancia oneWire para comunicarte con dispositivos del tipo OneWire (No
solo con Ics de temperatura Maxim/Dallas)*/
One Wire one Wire(ONE WIRE BUS);

/*Pasa nuestra referencia oneWire a Dallas Temperature. */ Dallas Temperature sensors
(&one Wire);

/*ç
* La función de configuración. Solo arrancamos el sensor desde aqui
*/
void setup (void)
{
/* start serial port*/ Serial. begin(9600);
Serial. println("Dallas Temperature IC Control Library Demo");

/*Arranca las librerías*/ Sensors .begin ();
}

/*ç
* Función principal, obtiene y muestra la temperatura
*/

```

```
void loop(void)
{
/* Llama sensores. Solicita las temperaturas () para promediar un total */
/*request to all devices on the bus*/ Serial.print("Requesting temperatures...");
sensors.requestTemperatures(); /*Envía el comando para obtener las temperaturas*/
Serial.println("DONE");
/* Después de recibir las temperaturas, las imprimimos acá.*/
/*Usamos la función ByIndex como ejemplo, solo obtenemos la temperatura del primer
sensor.*/ Serial.print("Temperature for the device 1 (index 0) is: ");
Serial.println(sensors.getTempCByIndex(0));
}
```

A partir del programa mostrado arriba, es posible aprender la sintaxis de la estructura de control de la programación.

(1) begin ()

Descripción

Configura la velocidad de transmisión de datos en bit por segundos (baudios). Para comunicarte con la computadora, utiliza cualquiera de estas velocidades: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, o 115200. Puedes también especificar otras velocidades para, por ejemplo, comunicarte a través de los pines 0 y 1 con un componente que requiera una velocidad de transmisión particular.

Un segundo argumento opcional configura los bits de datos, paridad y parada. Por defecto la configuración es 8 bits de datos, sin bit de paridad ni parada.

Syntax `Serial.begin(speed)`

`Serial.begin(speed, config)` Arduino Mega only: `Serial1.begin(speed)` `Serial2.begin(speed)`
`Serial3.begin(speed)` `Serial1.begin(speed, config)`

`Serial2.begin(speed, config)` `Serial3.begin(speed, config)`

Parámetros

Velocidad: en bit por segundos (baudios)

config: configura los bits de datos, paridad y datos. Son valores válidos: SERIAL_5N1
SERIAL_6N1 SERIAL_7N1
SERIAL_8N1 (por defecto) SERIAL_5N2
SERIAL_6N2 SERIAL_7N2 SERIAL_8N2 SERIAL_5E1 SERIAL_6E1 SERIAL_7E1 SERIAL_8E1
SERIAL_5E2

SERIAL_6E2 SERIAL_7E2 SERIAL_8E2 SERIAL_5O1 SERIAL_6O1 SERIAL_7O1 SERIAL_8O1
SERIAL_5O2 SERIAL_6O2 SERIAL_7O2 SERIAL_8O2

No retorna nada, ejemplo: void setup() {

```
Serial.begin(9600); // Abre el puerto serial, configurando la velocidad de transmisión de datos en 9600  
bps  
}
```

```
void loop() {} [Get Code]
```

Arduino Mega ejemplo:

```
// Arduino Mega utilizando sus cuatro puertos seriales  
// (Serial, Serial1, Serial2, Serial3),  
// con diferentes velocidades de transmisión: void setup(){  
Serial.begin(9600); Serial1.begin(38400); Serial2.begin(19200); Serial3.begin(4800); Serial.println("Hello  
Computer");  
  
Serial1.println("Hello Serial 1");  
Serial2.println("Hello Serial 2");  
Serial3.println("Hello Serial 3");
```

}

void loop() {}

(2) **println()**

Imprime la data en el puerto serial en la forma de un texto ASCII entendible por el ser humano, seguido de un carácter de corte de línea (ASCII 13, o '\r') y uno de línea nueva (ASCII 10, o '\n'). Este comando toma la misma forma que Serial.print().

Sintaxis

Serial.println(val) Serial.println(val, format)

Parámetros

val: El valor a imprimir - cualquier tipo de datos

Formato: especifica la base numérica (para tipos de datos integrales) o el número de decimales (para tipos de punto flotante)

Retornos

size_t (long): println() muestra el número de bytes, aunque leer dicho número es opcional

Example:

```
/*
```

Entrada analógica

Lee una entrada analógica en analog in 0, e imprime el valor.

```
*/
```

<http://www.elegoo.com>

int analogValue = 0; void setup() {

// variable que almacenará el valor analógico

// abre el puerto serial a 9600 bps: Serial.begin(9600);

}

void loop() {

// Lee la entrada analógica en pin 0: analog Value = analog Read(0);

// La imprime en muchos formatos: Serial.println(analogValue); Serial.println(analogValue, DEC);
Serial.println(analogValue, HEX); Serial.println(analogValue, OCT);

// La imprime como un decimal codificado mediante ASCII

// La imprime como un decimal codificado mediante ASCII

//La imprime como un hexadecimal codificado mediante ASCII

// La imprime como un octal codificado mediante ASCII

Serial.println (analog Value, BIN); // La imprime como un binario codificado mediante ASCII

// retardo de 10 milisegundos antes de la nueva lectura: delay(10);

}

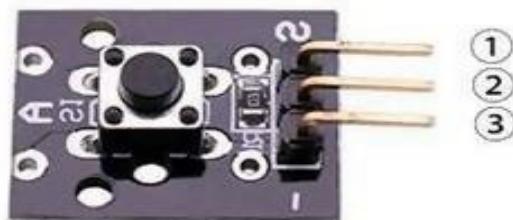
Lección 6 Módulo de Botón Interruptor

Resumen

En este experimento, aprenderemos a utilizar el botón interruptor

Módulo de Botón Interruptor

Un resistor de 10 K ohm incorporado se conecta entre el pin central y el pin "S" para ser usado como resistor pull up o pull down según se requiera. El pulsador conecta los dos pines externos.



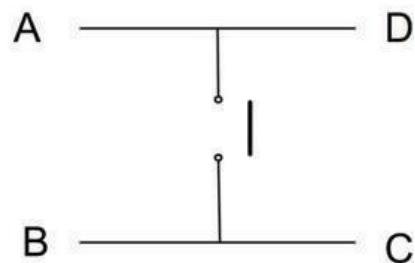
- 1.OUTPUT
- 2.VCC: 3.3V-5V DC
- 3.GND:ground

Componentes requeridos:

- 1x Elegoo Uno R3 1x cable USB
- 1x Modulo de botón 3x cables F-M

Interruptores pulsadores

Los interruptores son componentes realmente simples. Cuando presionas un botón o mueves una palanca, conectan dos contactos de forma que la electricidad fluya entre ellos. Los pequeños interruptores táctiles que se usan en esta lección tienen cuatro conexiones, por lo que pueden ser algo complicados.



En realidad, solo hay dos conexiones eléctricas, ya que dentro del interruptor B y C están conectados, al igual que A y D

Principio

El interruptor de botón y el puerto 13 forman parte de un simple circuito con el LED integrado. Para hacer destellar el LED usando el botón, podemos conectar el puerto digital número 13 al mismo y el interruptor al puerto 3 de la tarjeta Elegoo Uno. Cuando el interruptor sense, el LED parpadeará de acuerdo a su señal.

Esquema de conexión

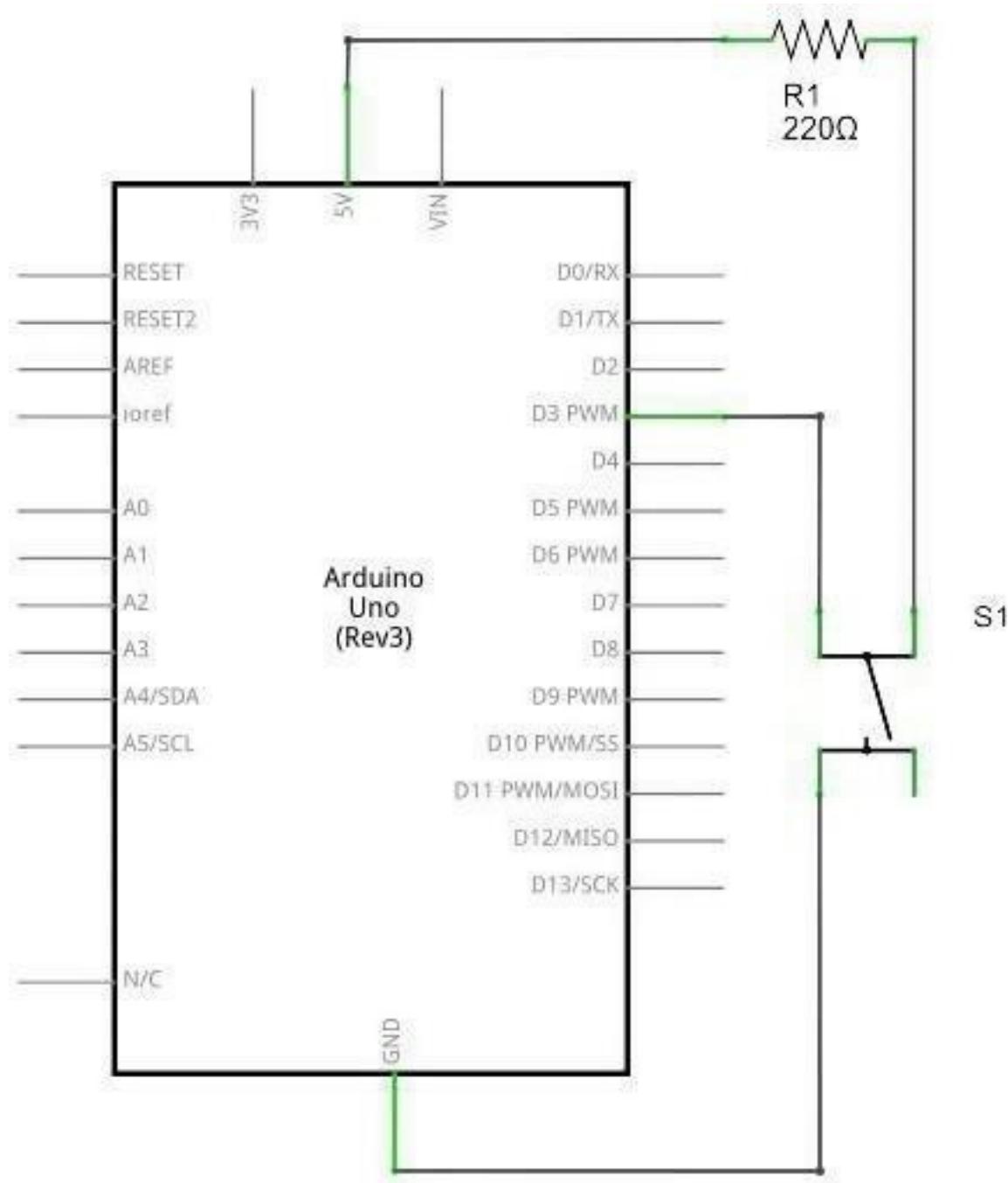
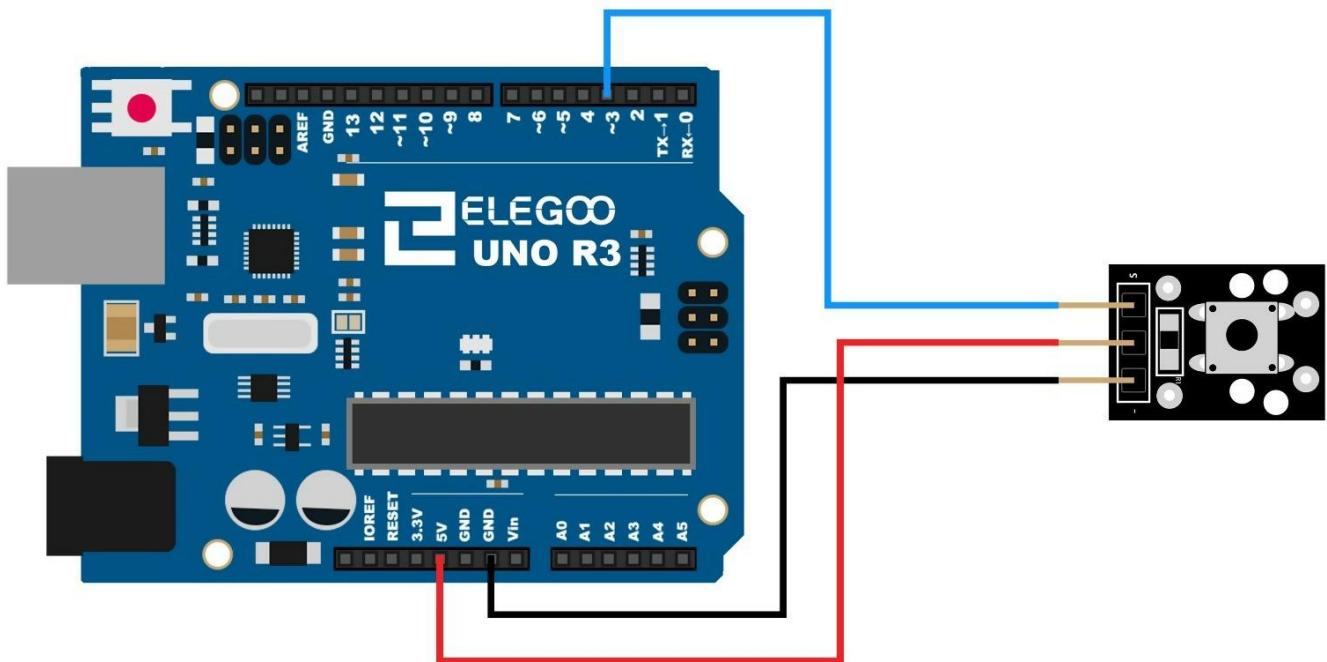


Diagrama de cableado



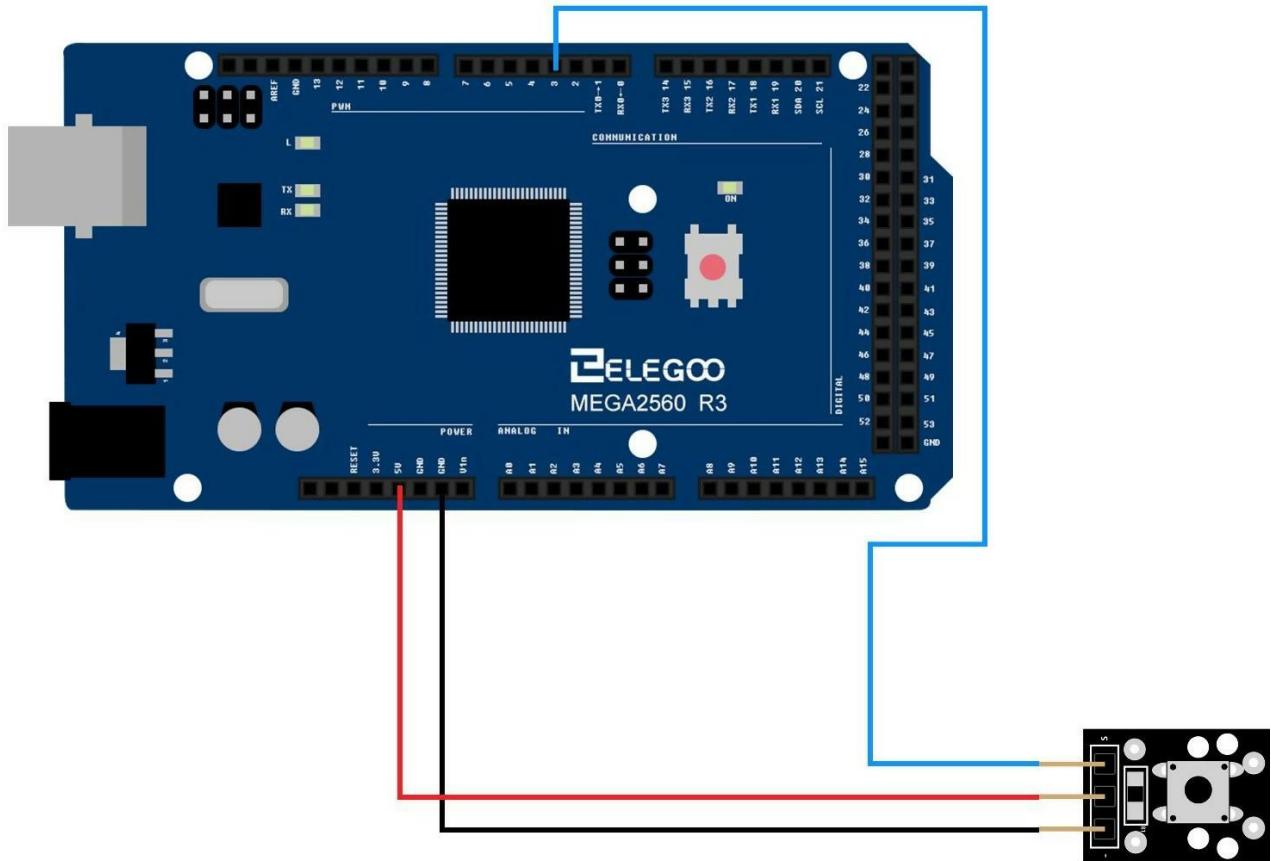
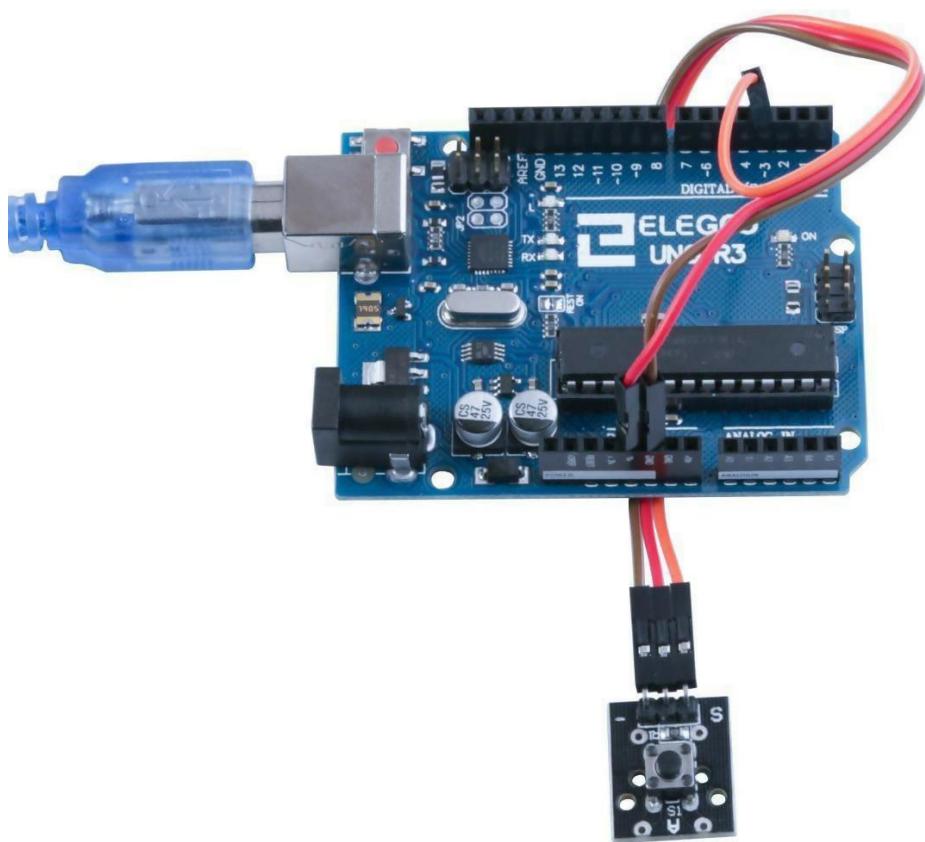


Imagen ejemplo



Código

Después de realizar el cableado, por favor abre el programa en la carpeta de código (Lección 6 Módulo de botón interruptor) y haz clic en UPLOAD para cargar el programa. Consulta la lección 2 para más detalles acerca de la carga del programa en caso que haya algún error. Luego presiona el botón y podrás ver como el LED enciende y apaga.

A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
//define el puerto del LED int Led = 13;
//define el puerto de shock en int Shock = 3;
//define la variable digital val int val;
void setup()
{
//define el LED como una salida PinMode (Led, OUTPUT);
//define el sensor shock como un puerto de salida

pinMode(Shock, INPUT);
}
void loop()
{
/Lee el valor de la interfaz digital 3 asignada a val

val = digital Read(Shock);
//cuando el sensor shock tiene señal, el LED parpadea

if (val == HIGH)
{
digitalWrite(Led, LOW);
```

```
}

else
{
digitalWrite(Led, HIGH);
}
}
```

A partir del programa mostrado arriba, aprendimos a programar la estructura de control:

(1) if/else

if/else permite tener mayor control del flujo de código que con el enunciado if básico, al permitir que se agrupen múltiples pruebas. Por ejemplo, se podría programar una entrada analógica y tomar una acción si el valor fuese menor que 500, o tomar otra acción cuando sea mayor que 500. Este código serían algo como esto: if (pinFiveInput < 500)

```
{
    // acción A }
else
{
    // acción B }
else puede proceder con otra prueba if, así que es posible realizar múltiples pruebas mutuamente excluyentes al mismo tiempo: if (pinFiveInput < 500)
{
    // Ejecuta la acción A }
else if (pinFiveInput >= 1000)
{
    //Ejecuta la acción B } else
{
    // Ejecuta la acción C }
```

Puedes tener innumerables opciones para ejecutar en el programa. (Otra forma de expresar las pruebas mutuamente excluyentes es a través del enunciado del interruptor)

Nota de programación: si utilizas if/else y quieres estar seguro de que siempre se tome alguna acción por defecto, es una buena idea. Terminar tus pruebas con un enunciado else configurado para ejecutar la acción que deseas.

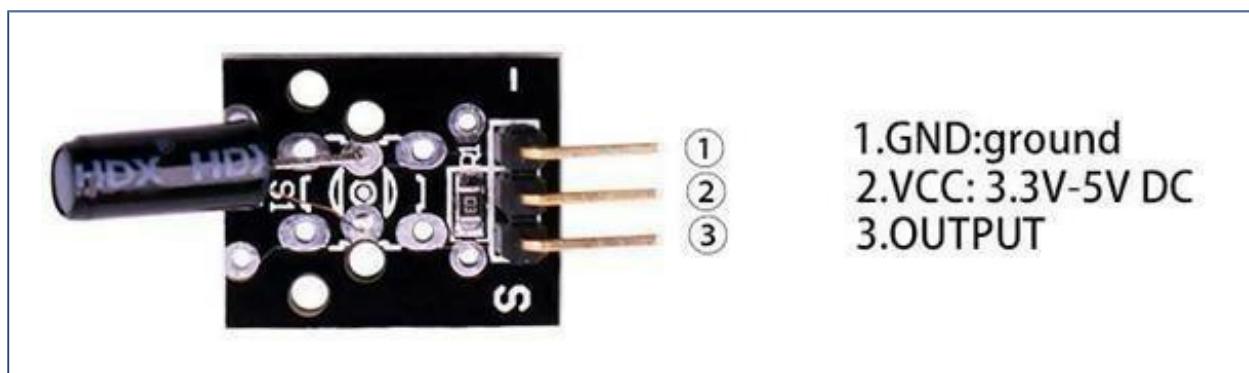
Lección 7 Módulos de Interruptor

Resumen

En esta lección, aprenderemos como utilizar módulos de interruptor. Incluyendo los módulos de interruptor de choque, inclinación y toque.

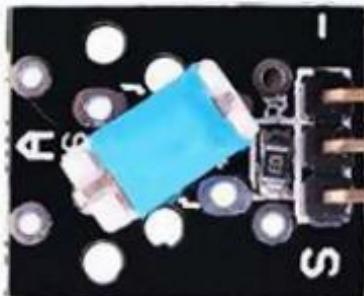
Módulo de interruptor de choque

Un resistor de 10 K ohm incorporado se conecta entre el pin central y el pin "S" para ser usado como resistor pull up o pull down según se requiera. Los contactos del interruptor conectan los dos pines externos.



Módulo de interruptor de inclinación

Un resistor de 10 K ohm incorporado se conecta entre el pin central y el pin "S" para ser usado como resistor pull up o pull down según se requiera. Los contactos del interruptor conectan los dos pines externos. Máxima carga del interruptor: 12VDC 50mA.



- 1.GND:ground
- 2.VCC:3.3V-5V DC
- 3.OUTPUT

Módulo de interruptor de toque

También llamado modulo sensor de vibración Los contactos de interruptor momentáneos van conectados entre los dos pines externos.



Componentes Requeridos:

- 1 x Elegoo Uno R3 1 x cable USB
- 1 x Módulo de interruptor de choque 1 x Módulo de interruptor de inclinación
- 1 x Módulo de interruptor de toque
- 3 x Cables F-M

Principio

El módulo de interruptor y el puerto digital 13 de la tarjeta UNO (o de la Mega 2560) tienen circuitos

LED incorporados. Por tanto, para hacer que el interruptor lo haga parpadear solo necesitas conectar el pin "S" del módulo interruptor al puerto digital 13 de la tarjeta Elegoo Uno (o de la Mega 2560). Después de conectar, el módulo envía una señal que cuando el interruptor está censando, hace que el LED parpadee.

Esquema de conexión

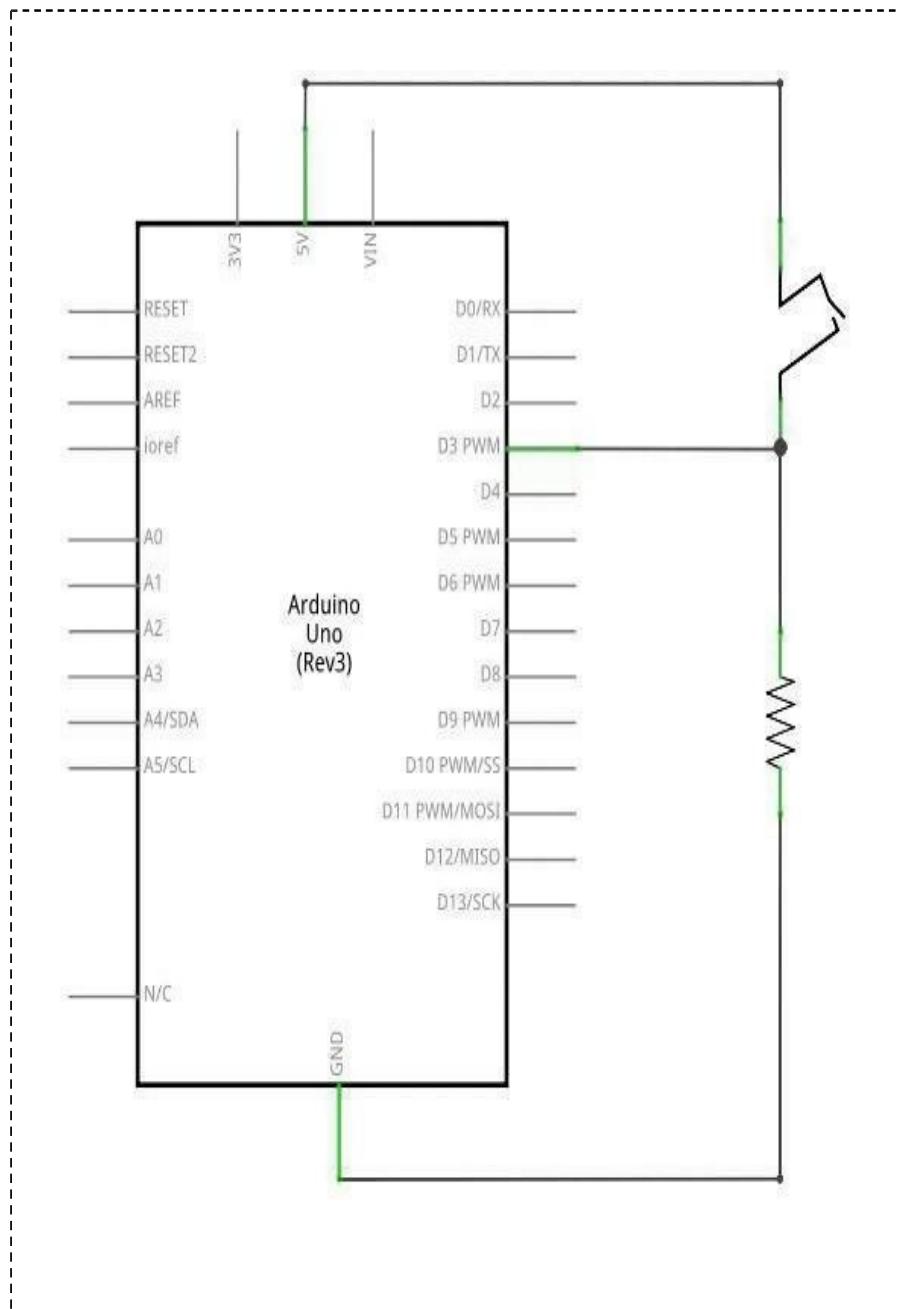
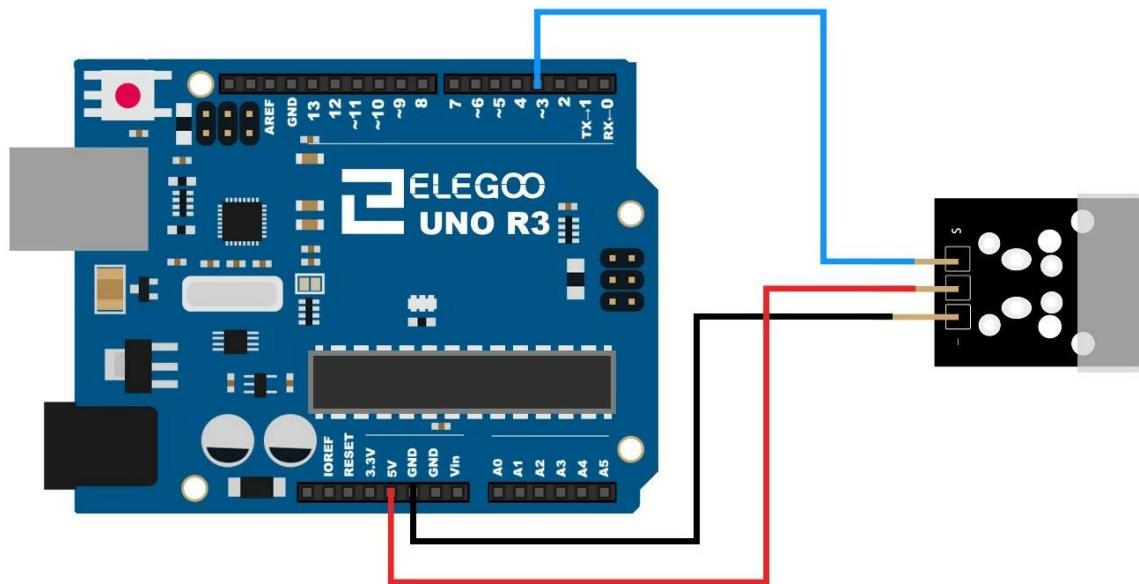
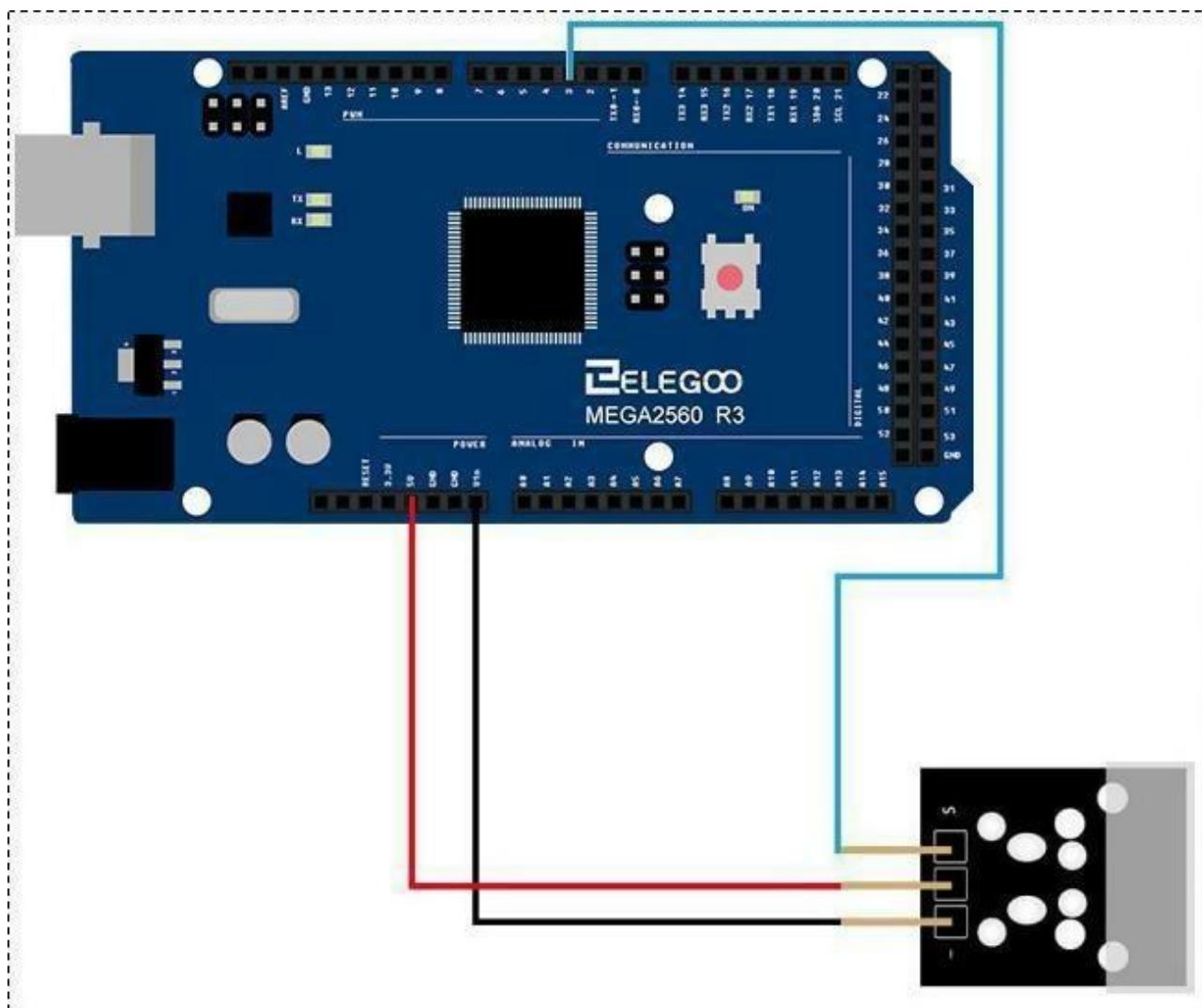
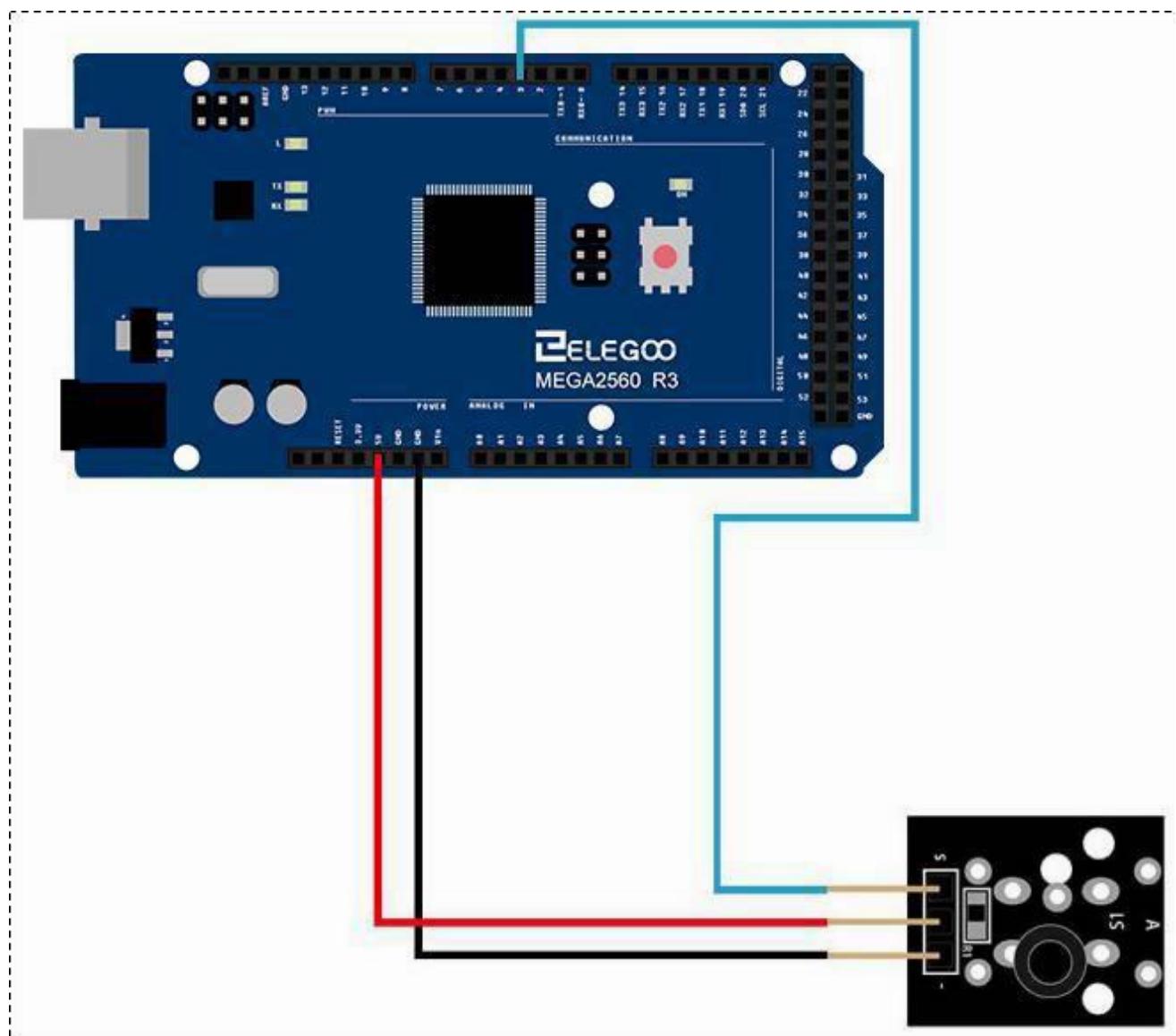
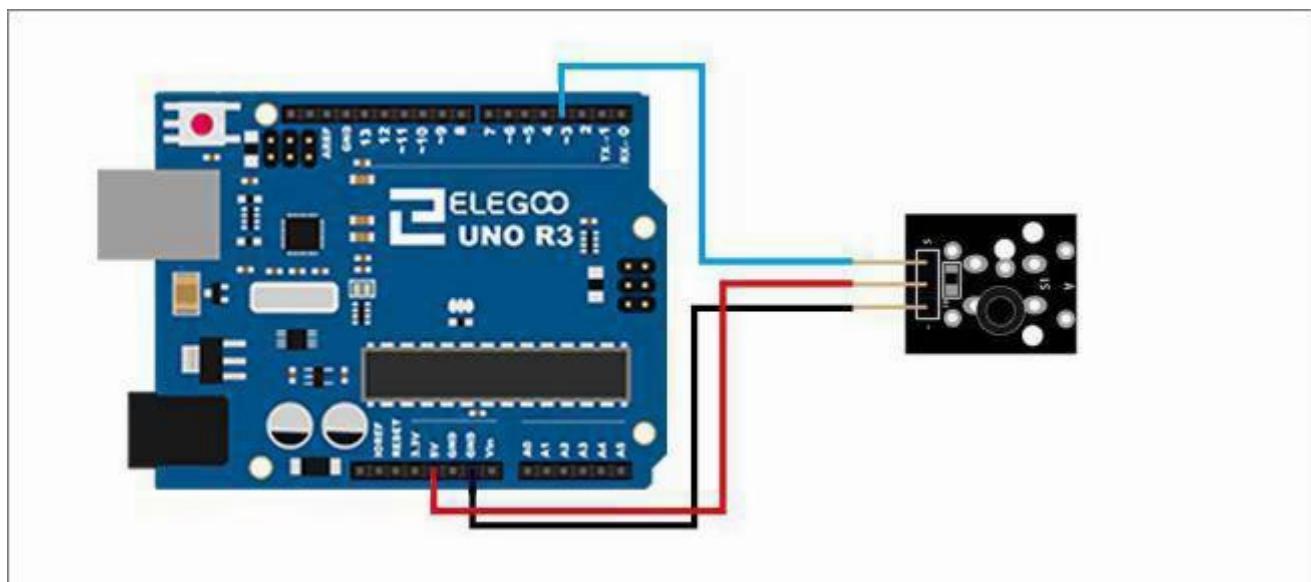
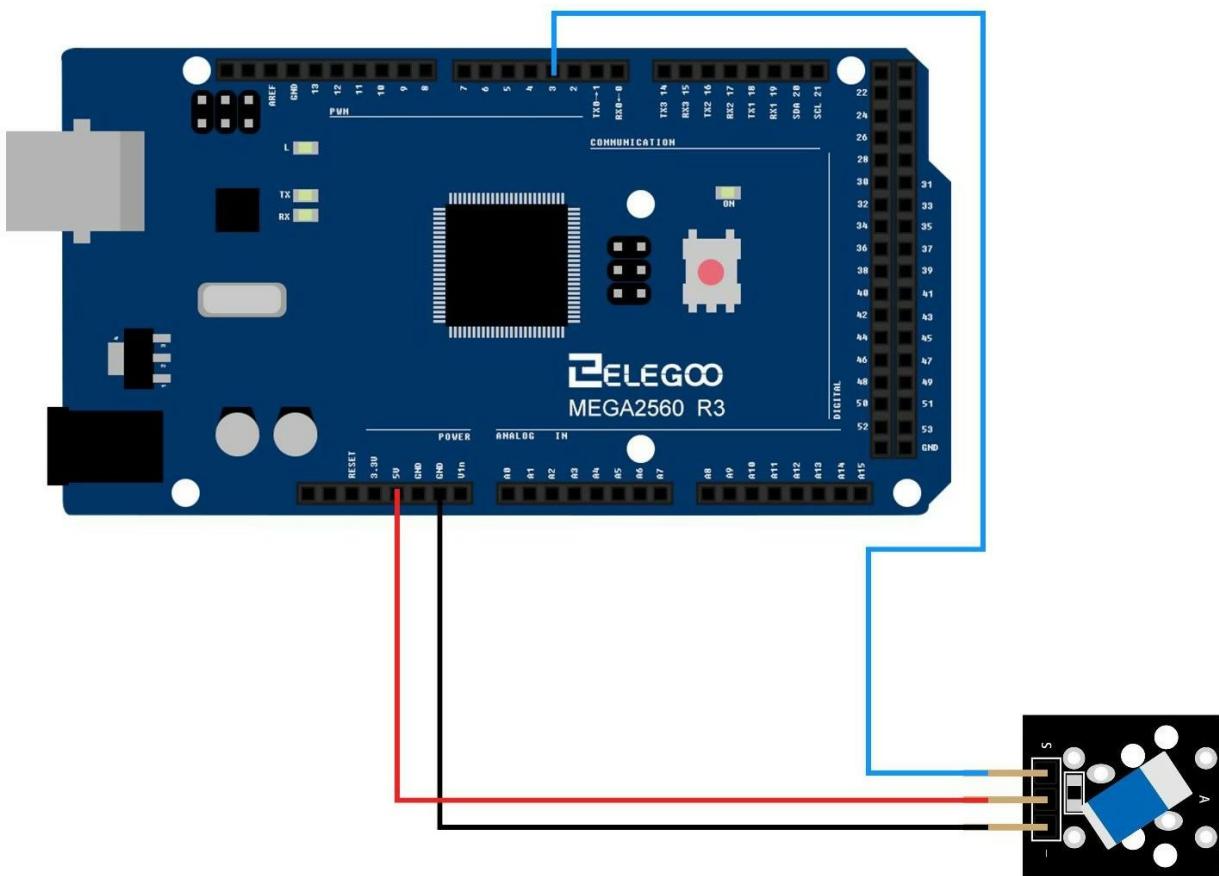
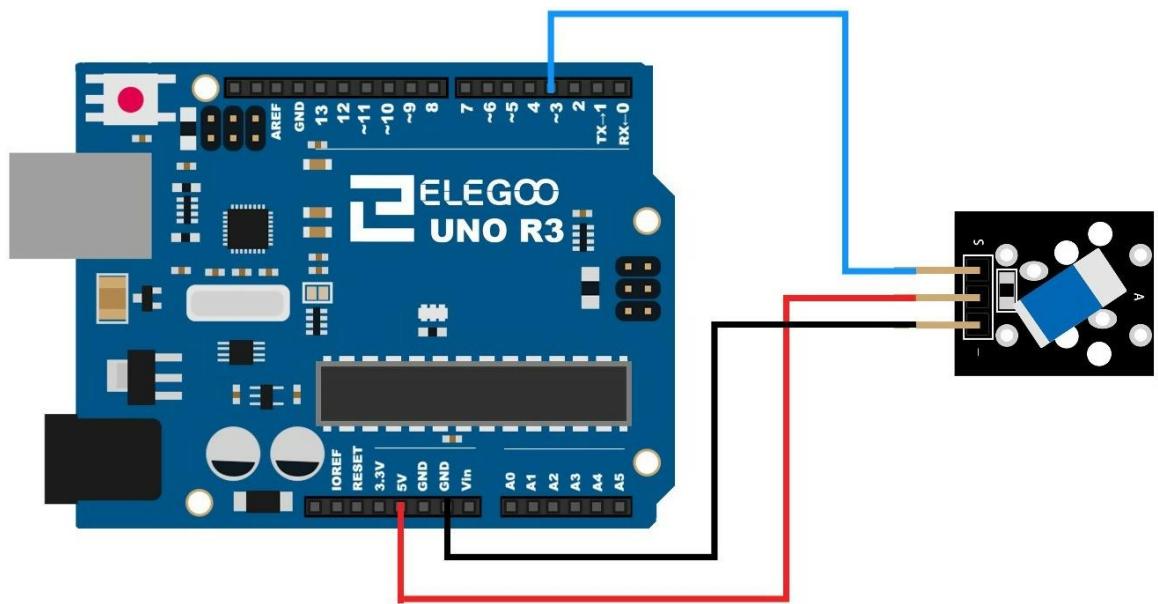


Diagrama de cableado





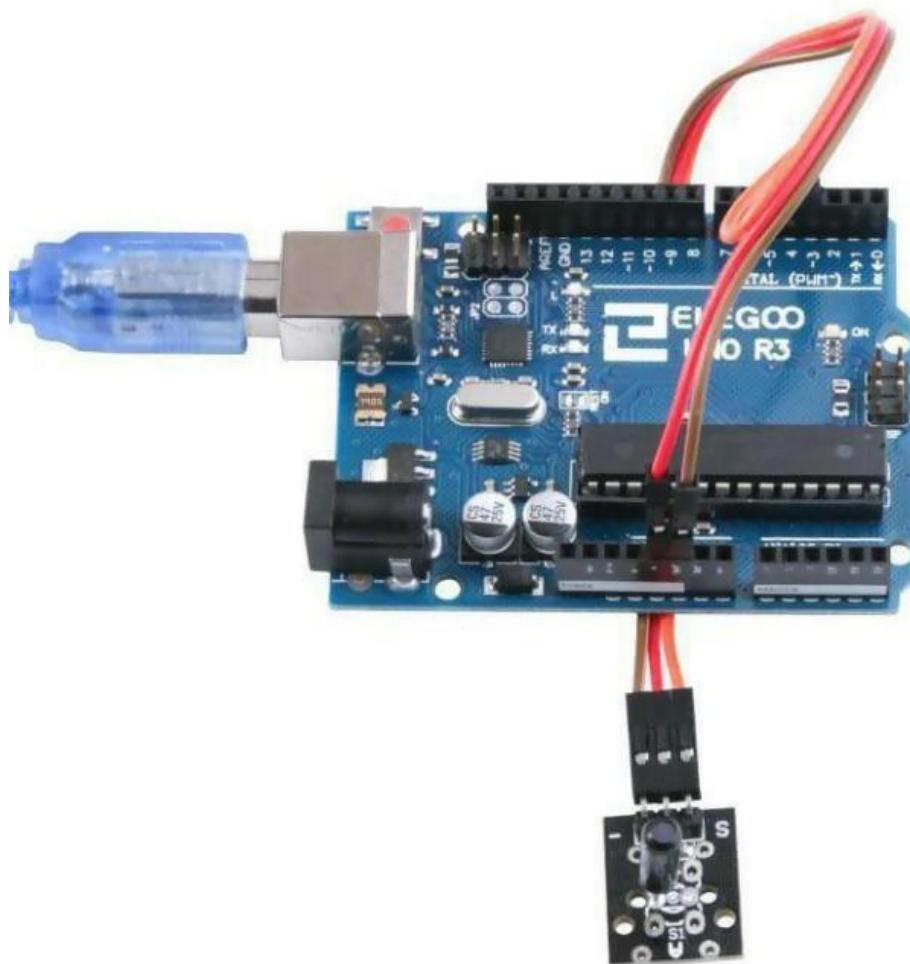




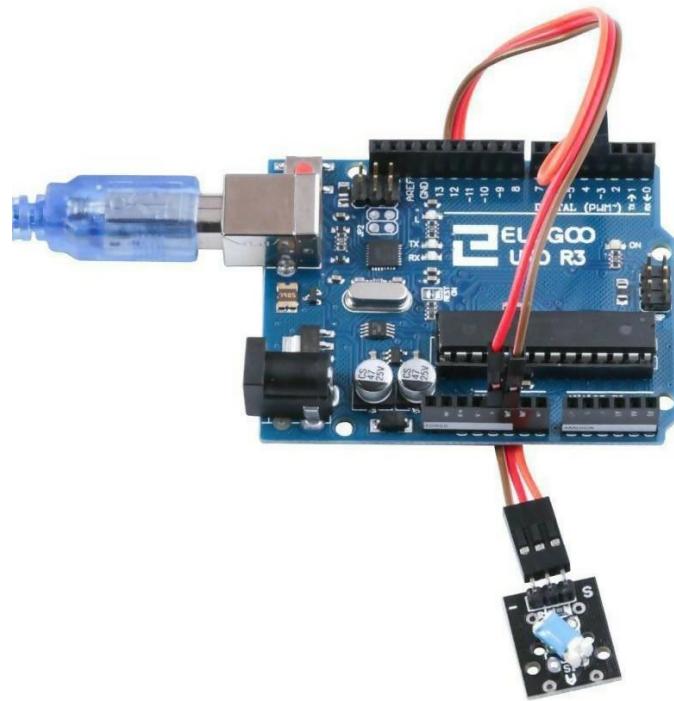
Código

Después de realizar el cableado, por favor abre el programa en la carpeta de código (Lección 7 todos los tipos de interruptores de vibración) y haz clic en UPLOAD para cargar el programa. Si hay algún error, consulta los detalles en la lección 2 para cargar el programa. Ya puedes inclinar o chocar el sensor para que veas como el LED se enciende y apaga

Imagen ejemplo







A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
//Define el puerto del LED
int Led=13;
//define el puerto del interruptor
int buttonpin=3;
//define la variable digital val
int    val;
void  setup()
{
//define el LED como un puerto de salida
pinMode(Led,OUTPUT);

//define el interruptor como un puerto de salida
pinMode(buttonpin,INPUT);} void loop()
{
/Lee el valor de la interfaz digital 3 asignada a val

val=digitalRead(buttonpin);
//cuando el sensor interruptor tenga señal, el LED destellará
if(val==HIGH)
{
/*Puedes cambiar el estado de la luz de control entre LOW y HIGH*/
digitalWrite (Led,HIGH);
}
else
{
digitalWrite (Led,LOW);
}
}
```

A partir del programa mostrado arriba, aprendimos a programar la estructura de control:

(1) if/else

if/else permite tener mayor control del flujo de código que con el enunciado if básico, al permitir que se agrupen múltiples pruebas. Por ejemplo, se podría programar una entrada analógica y tomar una acción si el valor fuese menor que 500, o tomar otra acción cuando sea mayor que 500.

Este código sería algo como esto:

```
if (pinFiveInput < 500)
{
    // acción A
}
else
{
    // acción B }
```

else puede proceder con otra prueba if, así que es posible realizar múltiples pruebas mutuamente excluyentes al mismo tiempo: if (pinFiveInput < 500)

```
{
    // Ejecuta la acción A }

else if (pinFiveInput >= 1000)
{
    //Ejecuta la acción B } else

{
    // Ejecuta la acción C }
```

Puedes tener innumerables opciones para ejecutar en el programa. (Otra forma de expresar las pruebas mutuamente excluyentes es a través del enunciado del interruptor)

Nota de programación: si utilizas if/else y quieres estar seguro de que siempre se tome alguna acción por defecto, es una buena idea terminar tus pruebas con un enunciado else configurado para ejecutar la acción que deseas.

Lección 8 Módulo Receptor y Transmisor IR

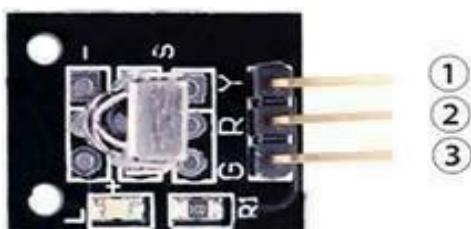
En este experimento, aprenderemos como utilizar el Receptor Infrarrojo y el módulo transmisor IR.

De hecho, en nuestra vida diaria esta clase de dispositivos juegan un rol en muchos aparatos eléctricos del hogar, tales como aires acondicionados, TVs, DVDs, etc. Se basan en tecnologías de sensores sin cables (wireless) y es muy conveniente usarlos.

Receptor IR

Sensor infrarrojo tipo 1838 para usar con señales IR de 38 KHz.

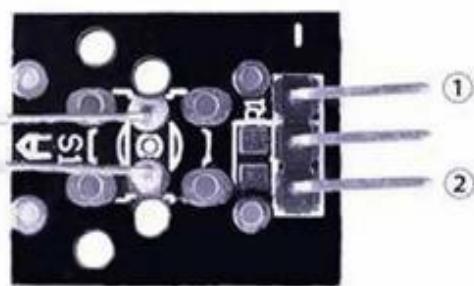
- Voltaje de alimentación: 2.7 a 5.5 V
- Frecuencia: 37.9 KHz
- Rango del receptor: 18m (típico)
- Ángulo de recepción: 90°



- 1.OUTPUT
- 2.VCC:3.3V-5V DC
- 3.GND:ground

Módulo transmisor IR

El LED-IR puede ser usado para construir una barrera de luz o un transmisor de señal para control remoto



- 1.GND:ground
- 2.INPUT

Componentes Requeridos:

2x Elegoo Uno R3 2x Cable USB

1x Módulo receptor IR 1x Módulo emisor IR 5 x Cables F-M

Introducción de componente Sensor Receptor IR:

Los detectores IR son pequeños microchips con una fotocelda calibrada para activarse al recibir luz infrarroja. Casi siempre se utilizan para detección de control remoto - Todos los TVs y DVDs tienen uno al frente para recibir la señal del mando manual. Dentro del control remoto se genera una señal de IR que activa el detector. LED, que emite pulsos IR para decir al televisor que se encienda, apague o cambie los canales. La luz IR no es visible para el ojo humano, lo que hace que sea un poco más trabajoso probar una configuración.

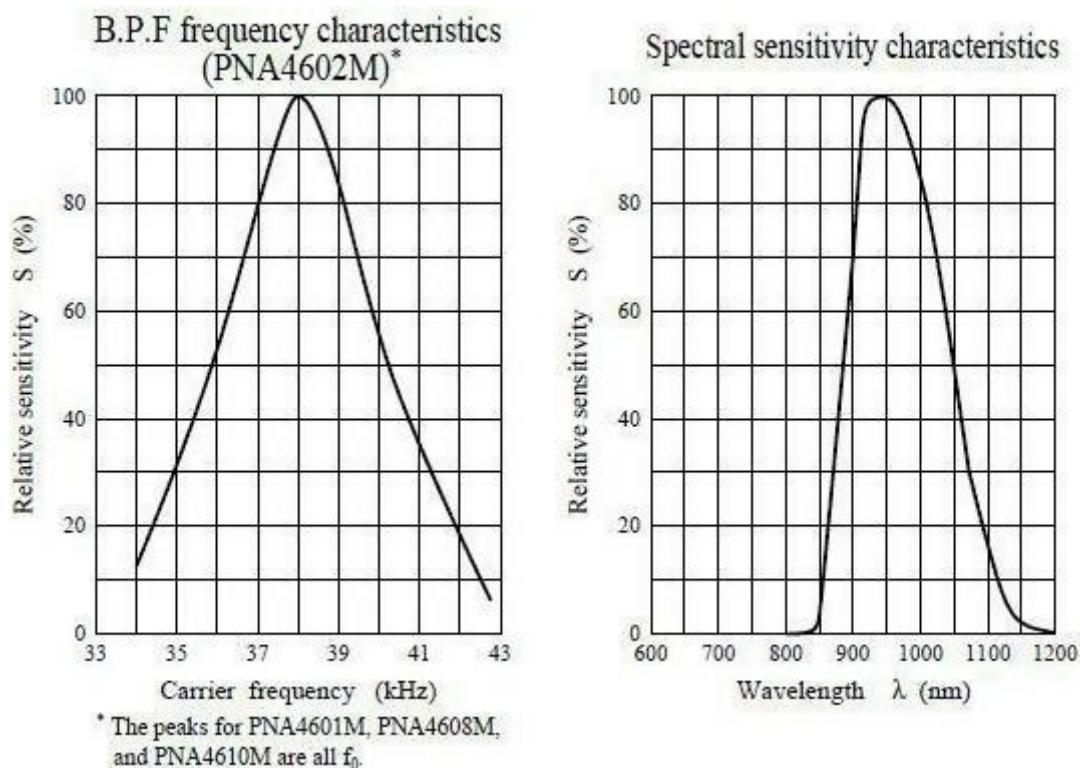
Hay algunas diferencias entre estas y las fotoceldas CdS, por ejemplo.

Los detectores IR tienen filtros especialmente diseñados para luz infrarroja, por lo que no son buenos para detectar la luz visible. Por otro lado, las fotoceldas son buenas detectando la luz visible amarilla / verde, pero no para detectar luz IR.

Los detectores IR tienen un demodulador interno que les permite captar la luz IR modulada a 38 KHz. Si los tratas de alumbrar con un LED-IR no funcionarán; necesitan una luz modulada bajo PWM a 38KHz. Las fotoceldas no poseen demodulador y son capaces de detectar cualquier frecuencia (incluyendo DC) dentro de su tiempo de respuesta de aproximadamente 1Khz

Los detectores IR tienen salidas digitales - Pueden detectar la señal IR de 38Khz y entregar una salida de nivel bajo (Low - 0V) o de nivel alto (High - 5V). Las fotoceldas actúan como resistores, su resistencia cambia dependiendo de la cantidad de luz a la que estén expuestas.

Lo que puedes medir

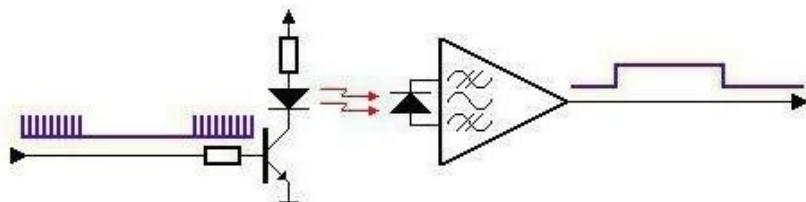


Como puedes ver en estos gráficos de la hoja técnica, la frecuencia de detección pico es de 38 KHz y el peak LED color es 940 nm.. . Puedes usar un rango de entre 35 KHz a 41 KHz pero la sensibilidad caerá y podrás detectar muy bien a la distancia. Del mismo modo, puedes usar LEDs de 850 a 1100 nm LEDs pero no serán tan eficientes como los de 900 a 1000 nm así que trata de utilizar los que recomendamos. Verifica la hoja técnica de tu LED IR para constatar la longitud de onda Trata de usar 940 nm - recuerda que este valor de 940 nm no es luz visible, sino infrarroja!

Principio

Primeramente, conozcamos la estructura del cabezal receptor de infrarrojos: este tiene dos elementos importantes que lo componen, el IC y el PD. El IC tiene los componentes de procesamiento del cabezal receptor, principalmente compuestos de silicio y circuitería. Es un dispositivo de alta integración. Su función principal es filtrar, decodificar, amplificar, etc. El PD es un diodo foto sensitivo. Su principal función es recibir la señal de luz.

Abajo podrás encontrar un diagrama breve del principio de funcionamiento.



El diodo emisor de luz infrarroja genera una señal modulada que será recibida por el cabezal receptor de infrarrojos, donde se filtrará, decodificará y procesará.

Diodo emisor de luz infrarroja: mantener limpio y en buenas condiciones. Durante la operación del dispositivo, todos los parámetros deben mantenerse dentro de los valores límite (positivo para la corriente 30~60mA, corriente de pulso positiva 0.3~1A, voltaje reverso 5V, potencia de disipación 90mW, rango de temperatura de trabajo -25~+80 °C, y rango de temperatura de almacenamiento entre 40~100 °C, temperatura de soldadura

260°C) el infrarrojo del cabezal debe corresponder a la aplicación, pues de otro modo influenciará la sensibilidad.

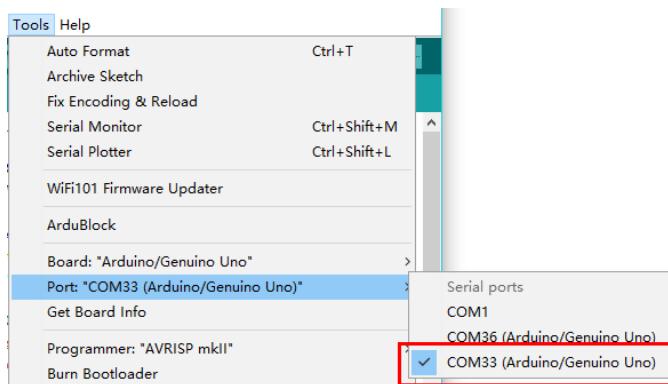
Código

Después de realizar el cableado, por favor abre el programa en la carpeta de código (Lección 8 Receptor y Emisor IR) y haz clic en UPLOAD para cargar el programa. Si hay algún error, consulta los detalles en la lección 2 para cargar el programa.

Antes de poder correrlo, asegúrate de tener la librería <IRremote> instalada, o reinstálala de ser necesario. De otra forma, tu código no funcionará. Para detalles del tutorial acerca de cómo cargar los archivos de librerías, consulta la lección 1.

Para empezar, toma la tarjeta UNO y conéctale el módulo emisor de infrarrojos de acuerdo con la imagen que se muestra. Luego sigue el procedimiento previo para escribir el programa de emisión de infrarrojos. Luego, toma otra tarjeta UNO, conéctale el módulo receptor de infrarrojos como se muestra abajo y escribe el programa del receptor IR.

Nota: antes de descargar, asegúrate de seleccionar de nuevo el puerto de descarga. Habrá dos puertos seriales cuando conectas dos tarjetas UNO, debemos seleccionar uno nuevo como se muestra en la figura.



Después de escribir el programa, alinea los módulos de emisión y recepción de infrarrojos. Presta atención al UNO del receptor infrarrojo. Si la luz L destella, significa que el experimento fue exitoso. El principio del experimento es que el módulo de emisión de infrarrojos transmita la señal codificada específica. Una vez que el módulo receptor capta la señal, el UNO la decodifica y opera la lámpara L, haciéndola destellar.

Este experimento es la base de muchas aplicaciones: puedes usar la señal IR para que el UNO controle otros módulos tales como robots、juguetes、electrodomésticos, etc.

Esquema de conexión

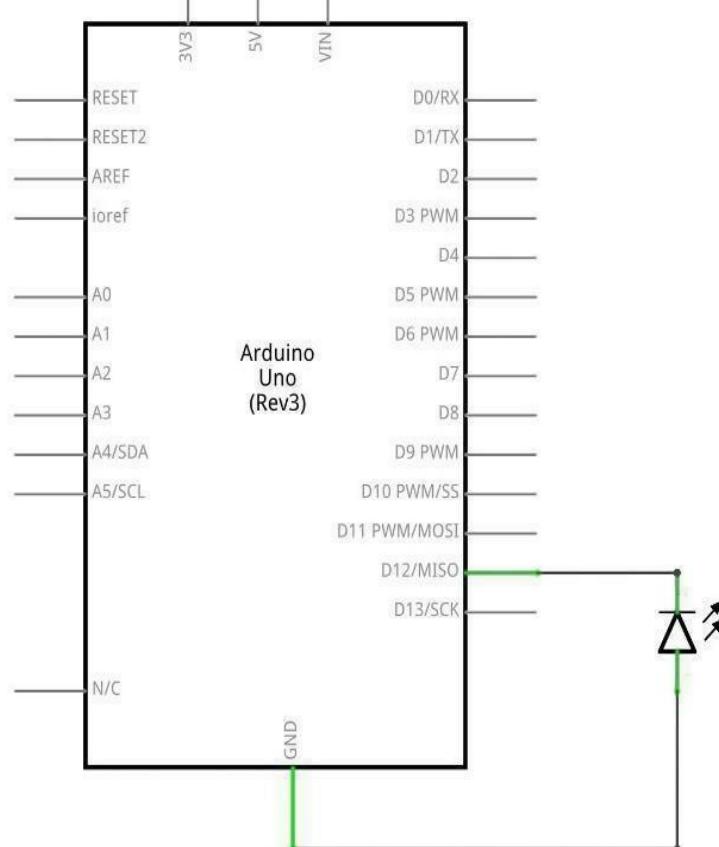
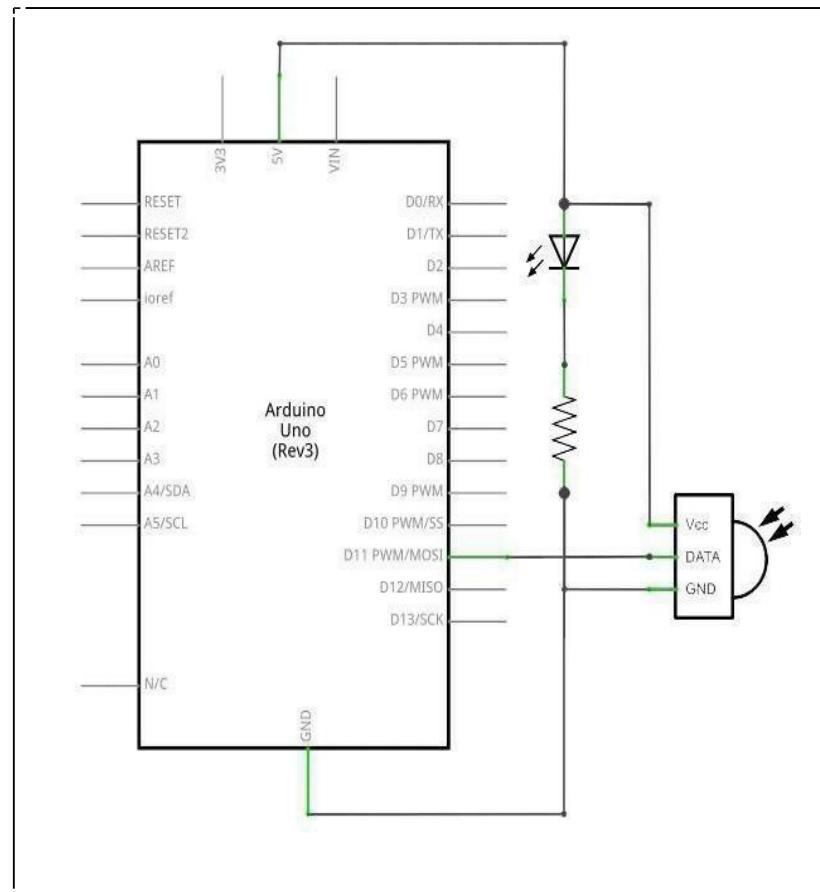
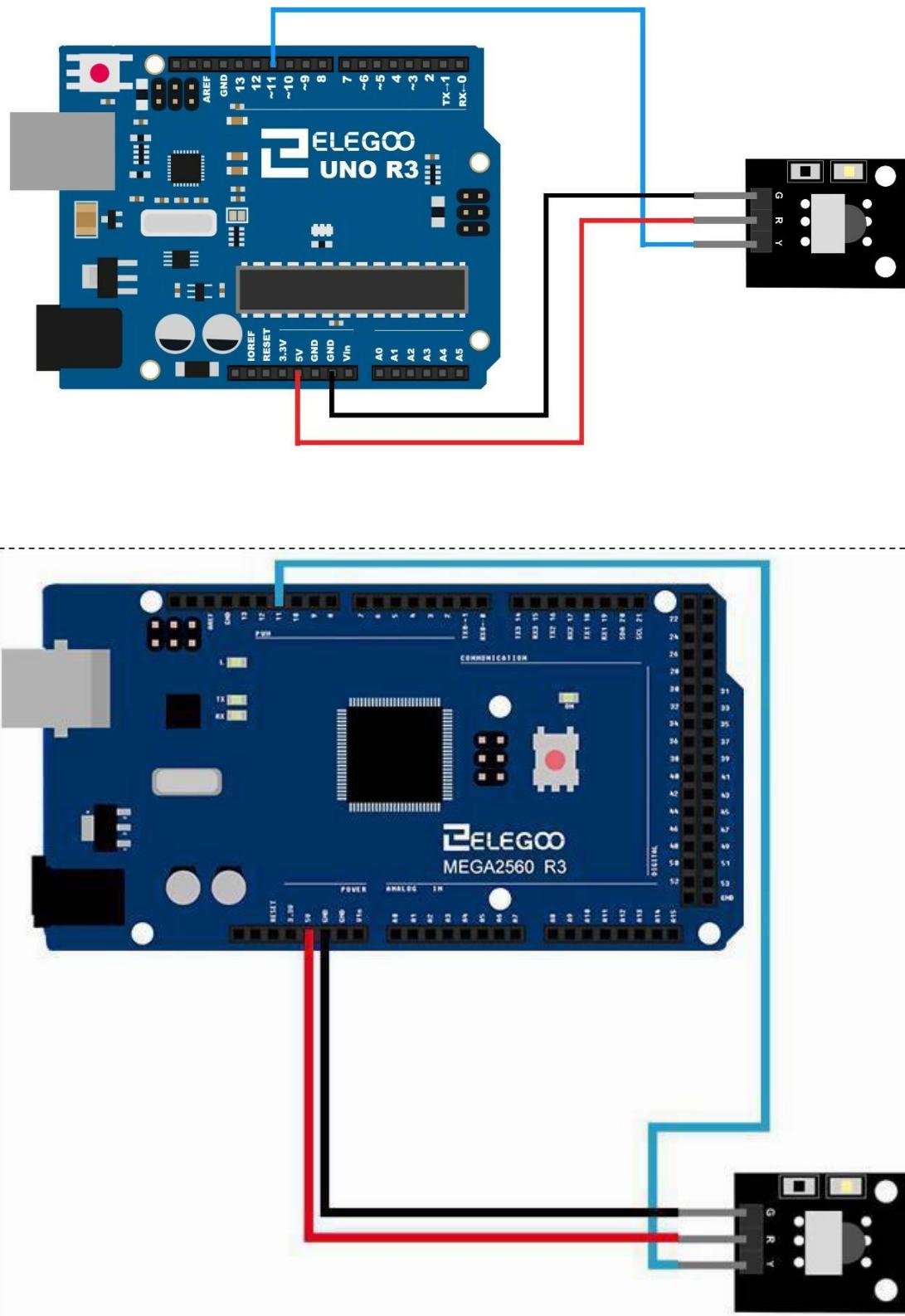
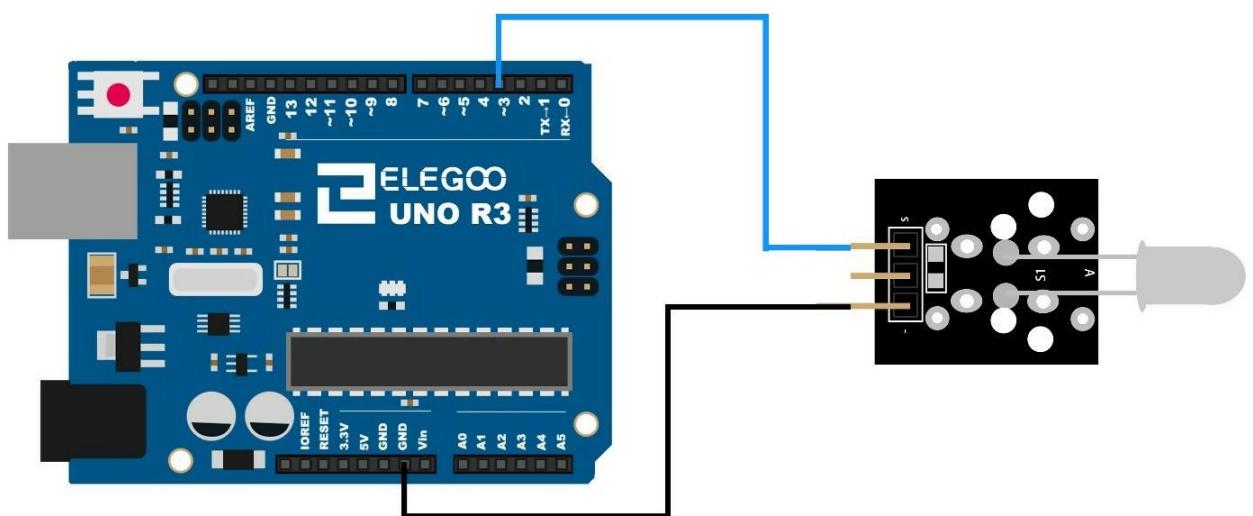




Diagrama de cableado





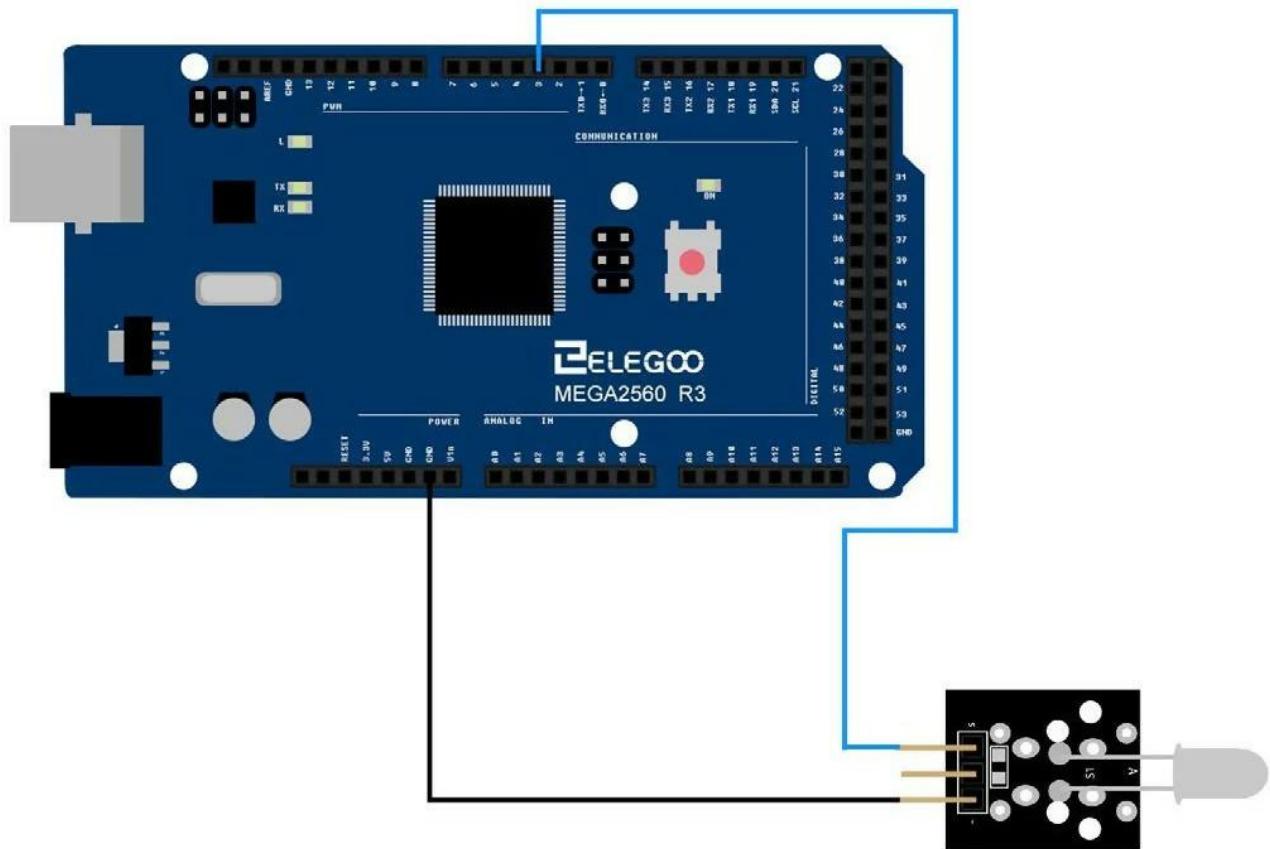
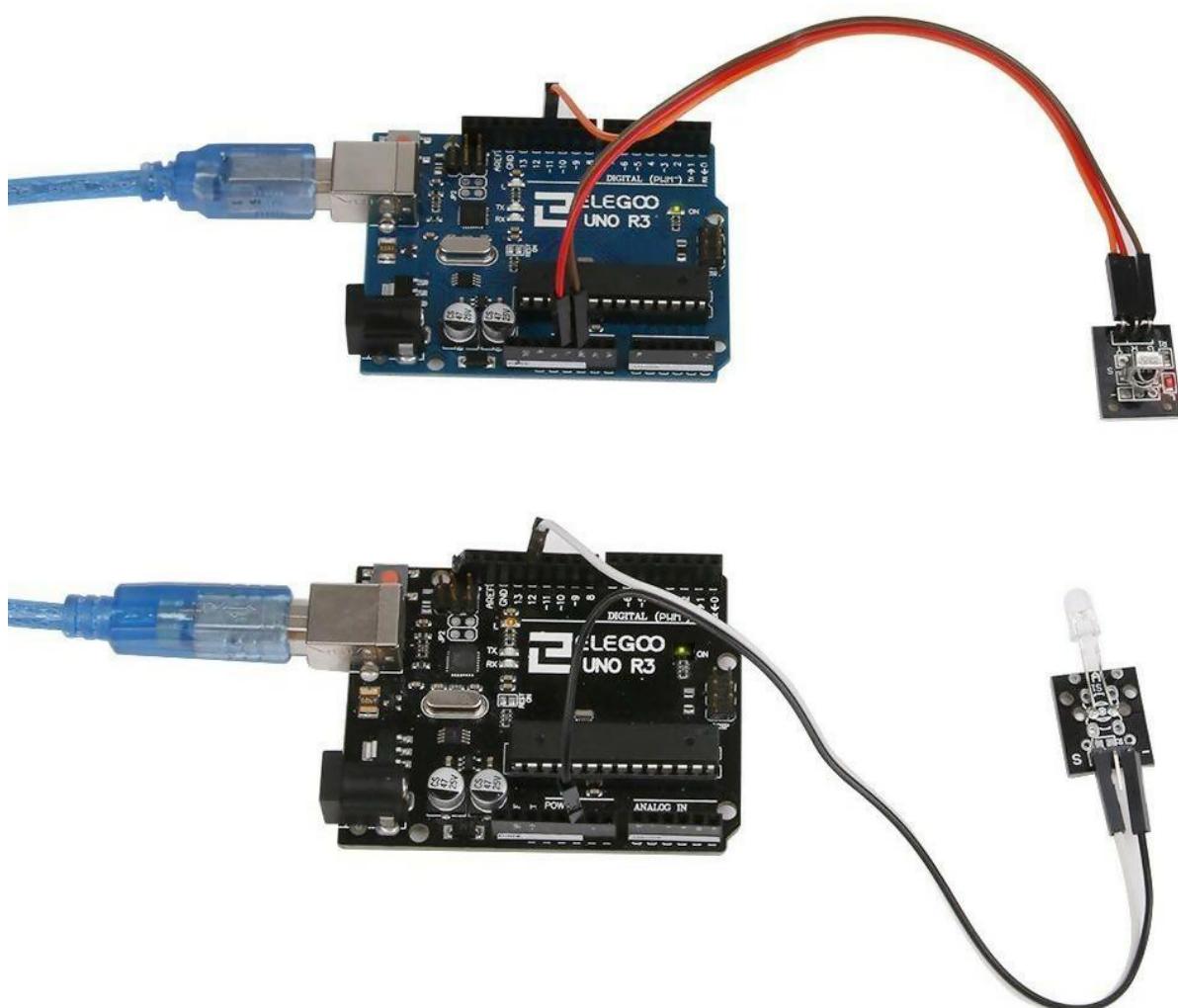


Imagen ejemplo



Resolución de fallas

Si la lámpara L no destella, puede ser un error de conexión o un problema en la escritura del programa.

Por favor verifica los pasos anteriores y corrige los errores.

Pero si estás seguro que tanto la conexión como el programa están correctos. Hay dos maneras para determinar si el módulo receptor IR o el emisor IR están dañados.

1. El módulo receptor

Por favor utiliza un control remoto de TV para controlar el módulo. Si el receptor IR destella, significa que el mismo está en buen estado. Si no lo hace, está averiado.

2. El módulo emisor.

Abre la cámara de tu teléfono Android y ponla frente al LED del emisor. Si puedes ver una luz púrpura, el módulo está bien. Si no la puedes ver, está averiado.

Si alguno de los módulos te llego dañado, por favor contáctanos y te enviaremos el recambio.

A continuación se muestra el código usado en este experimento y su correspondiente explicación:

Programa de emisión de infrarrojos:

```
/* Un LED-IR debe estar conectado al pin 3 PWM del Arduino.*/
```

```
/*Se hace referencia a la librería <IRremote.h> */
```

```
#include <IRremote.h>
IRsend irsend; void setup()
{
void loop()
{
/* send 0x0 code (8 bits)*/ irsend.sendRC5(0x0, 8);
Delay (200); irsend.sendRC5 (0x1, 8);
Delay (200);
}
```

(2) programa del receptor de infrarrojos:

/*Se hace referencia al archivo de librería <IRremote.h> */

```
#include <IRremote.h>
/* Infrared signal receiving pin */
```

```
#define RECV_PIN
```

11

```
/* define LED pin */ #define LED13
IRrecv irrecv(RECV_PIN); decode_results results; void setup() {
/* Inicializa el LED como una salida */ pinMode(LED, OUTPUT); Serial.begin(9600);
/* Arranca el receptor */ irrecv.enableIRIn();
}
void loop() {
if (irrecv.decode(&results))      {
int state;
if ( results.value == 1 )      {
state = HIGH;
}
else{
State = LOW;
}
digitalWrite( LED, state ); Serial.println(results.value);
/* Preparando para recibir el siguiente valor */
irrecv.resume();
}
}
```

Lección 9 Módulo de Timbre Activo

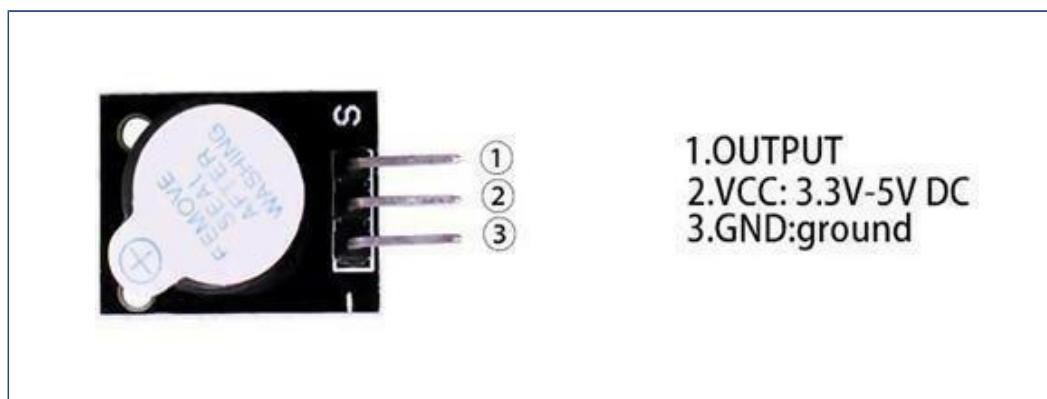
Resumen

En este experimento, aprenderemos a utilizar el módulo de timbre.

Utilizando Arduino podemos hacer muchas aplicaciones interactivas. Hasta ahora solo hemos utilizado luces haciendo destellar LEDs. Así como hemos utilizado luces, también podemos implementar otros circuitos como el de este experimento, donde estaremos produciendo sonidos. Los componentes más comunes para este tipo de aplicaciones son los timbres y los altavoces. Si se los compara con los altavoces, los timbres son más sencillos y fáciles de usar, por lo que para este experimento implementaremos el timbre.

Introducción de Componente Módulo de Timbre

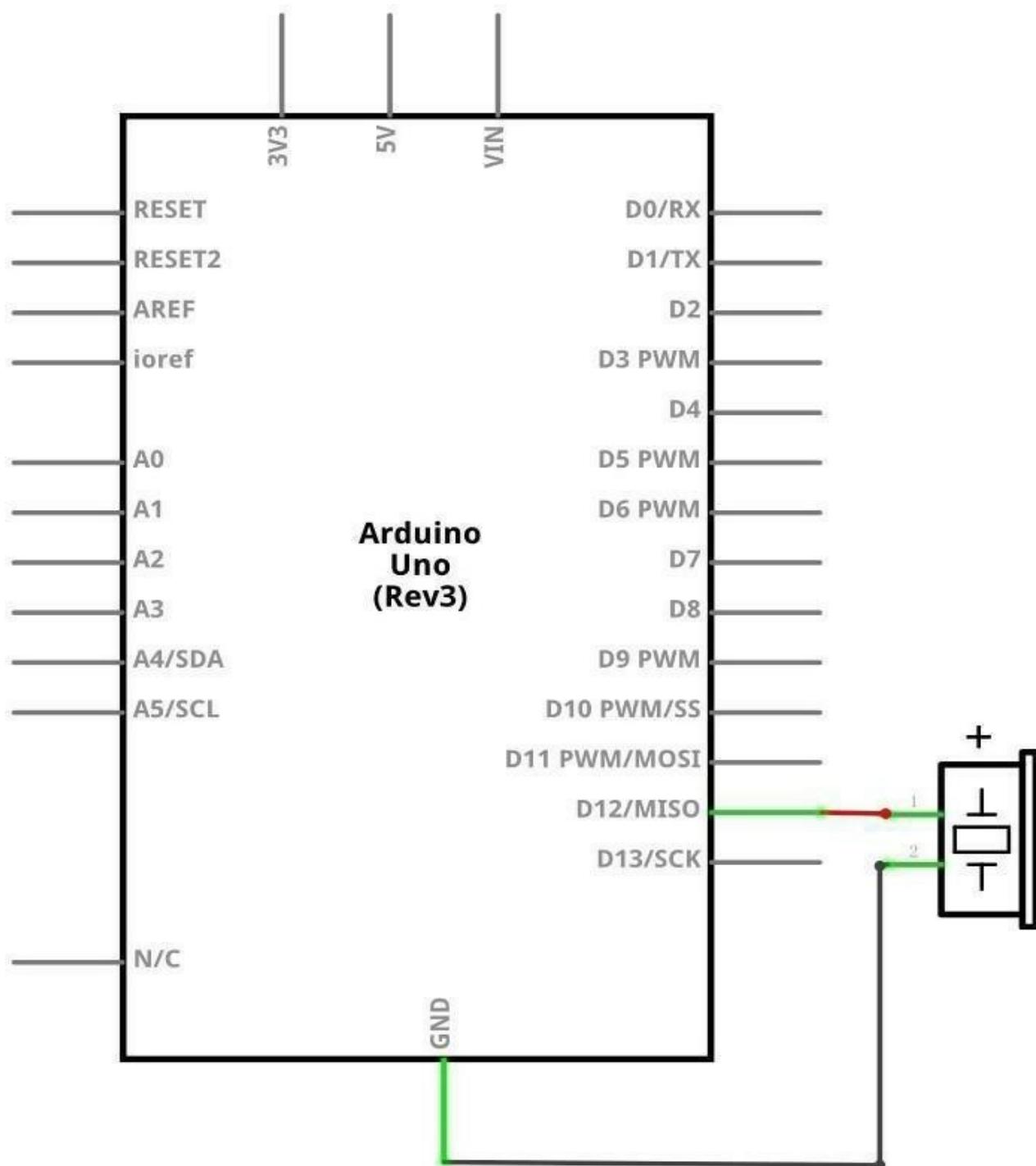
Los timbres electrónicos se alimentan con corriente directa y vienen equipados con un circuito integrado. Son ampliamente utilizados en computadoras, impresoras, alarmas, juguetes electrónicos, dispositivos electrónicos automotrices, teléfonos, temporizadores y otros productos electrónicos para dispositivos de voz. Hay dos tipos de timbres: los activos y los pasivos. Si volteas los timbres te podrás fijar en que el que tiene una tarjeta de circuito verde es un timbre pasivo. El que está cerrado y tiene cinta negra es uno activo.



Componentes requeridos:

- 1 x Elegoo Uno R3 1x Timbre activo
- 3 x cables F-M (Cables DuPont Hembra a Macho)

Esquema de conexión



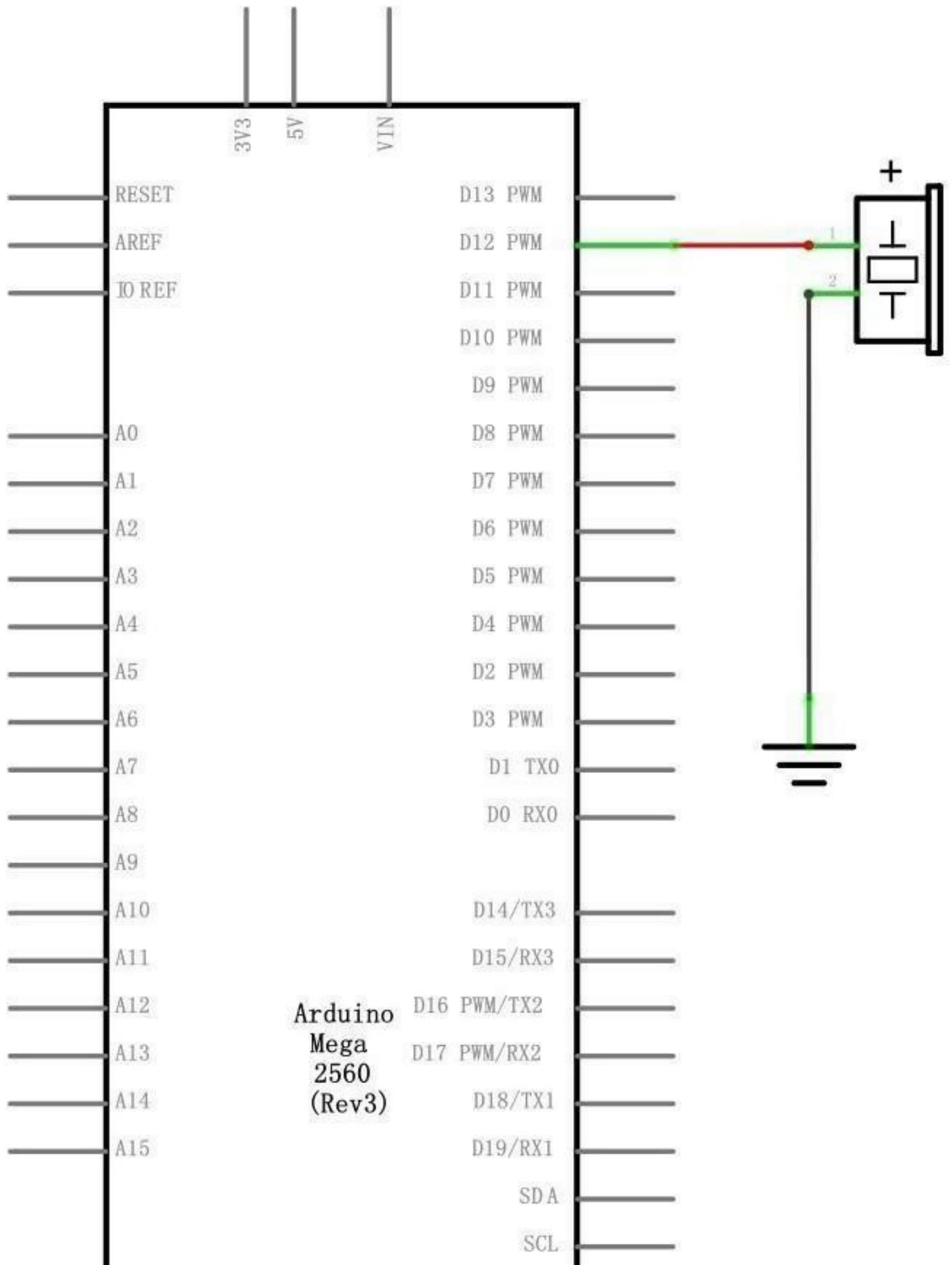
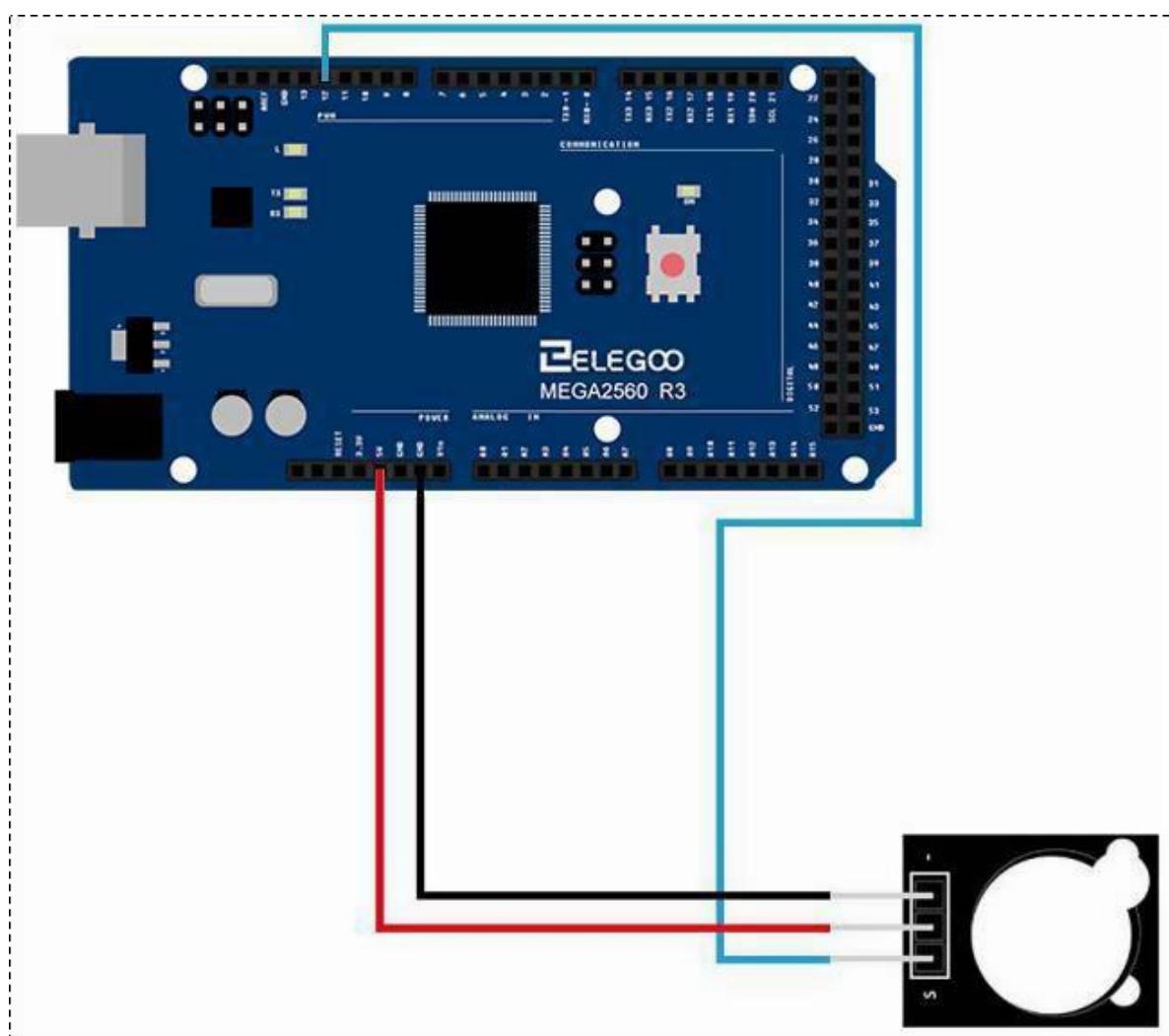
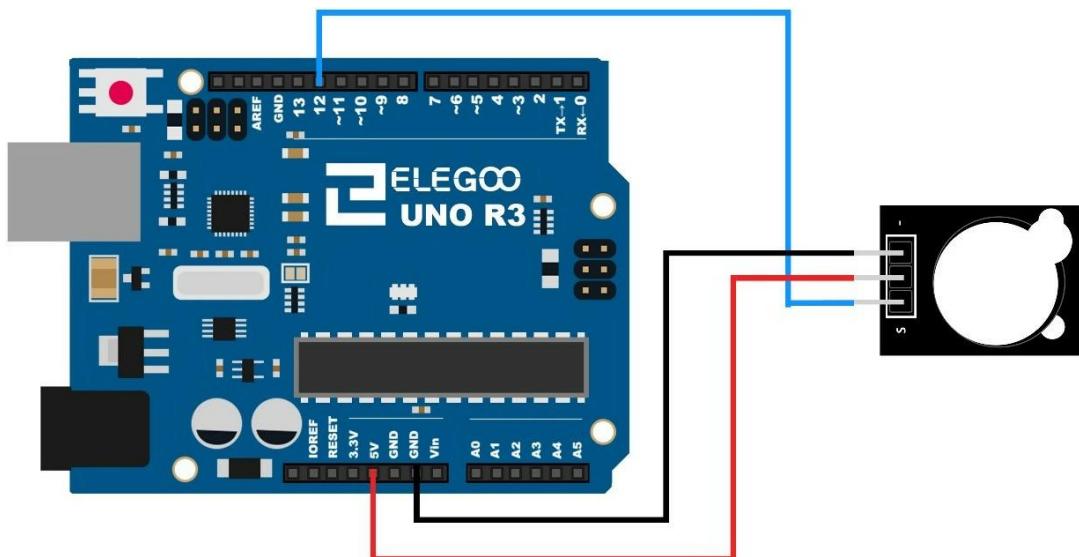


Diagrama de cableado

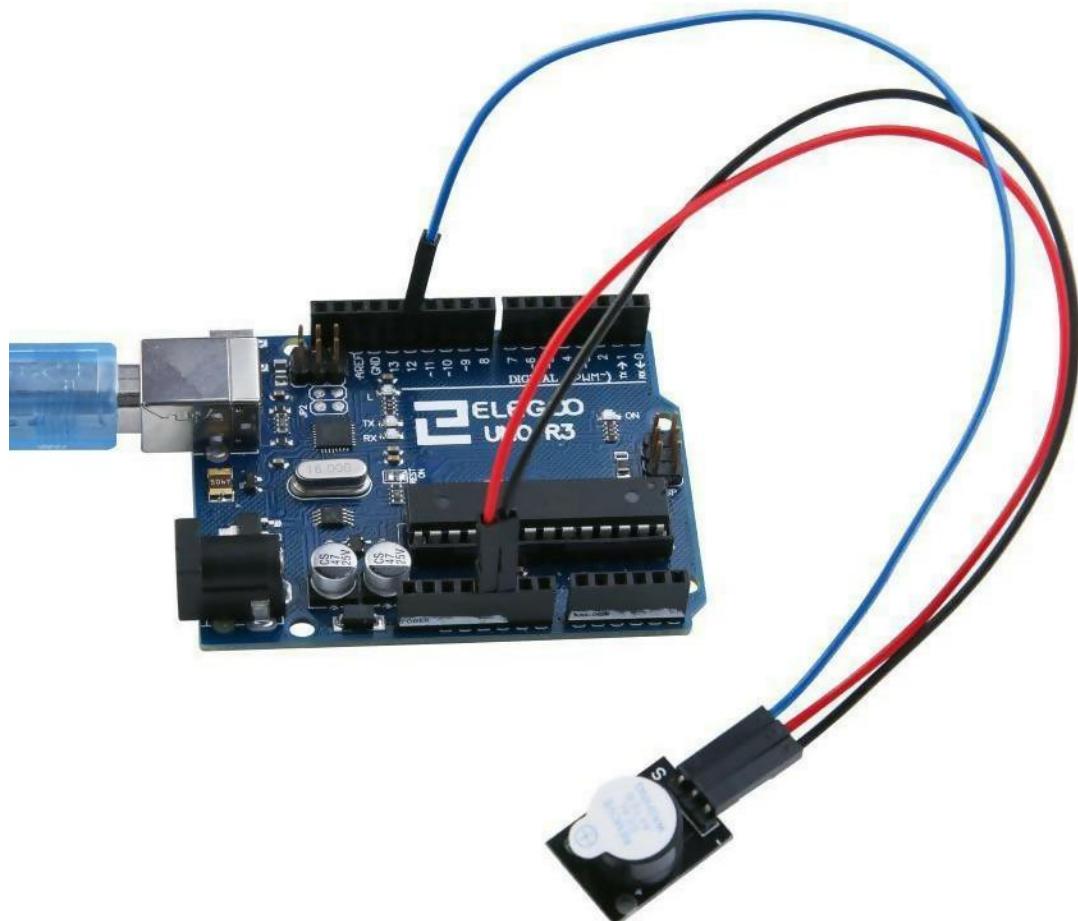


Código

Después de realizar el cableado, por favor abre el programa en la carpeta de código (Lección 9 Módulo de Timbre Activo) y haz clic en UPLOAD para cargar el programa. Si hay algún error, consulta los detalles en la lección 2 para cargar el programa.

Imagen ejemplo

A continuación se muestra el código usado en este



experimento y su correspondiente explicación:

```
/* El pin del timbre activo */  
int buzzer = 12; void setup()  
{  
    /* Inicializa el pin del timbre como una salida*/ pinMode(buzzer, OUTPUT);  
}  
void loop()  
{  
}
```

/*define la variable char i */

```
unsigned char i; while(1)
{
/* frecuencia de salida */ for(i=0;i<80;i++)
{
/*Cuando el timbre recibe un nivel de voltaje alto, suena */ digitalWrite(buzzer,HIGH);

delay(1);

// espera 1ms

digitalWrite(buzzer,LOW);

delay(1);

}

// espera 1ms

/* Otra frecuencia de salida */ for(i=0;i<100;i++)
{
digitalWrite(buzzer,HIGH);

delay(2);

}

//espera 2ms

digitalWrite(buzzer,LOW);

delay(2);

}

//espera 2ms

//espera 2ms

}
```

A partir del programa mostrado arriba, es posible aprender la sintaxis de la estructura de control de la programación.

(1) Lazos while

Descripción

Los lazos while se mantendrán funcionando infinitamente hasta que la expresión entre los paréntesis () se haga falsa. Debe haber algún cambio en la variable a probar, o el lazo jamás se detendrá. Este cambio puede darse en tu código, como por ejemplo al incrementar una variable, o mediante una condición externa, como la señal de un sensor.

Sintaxis

```
while(expresión){    // enunciado(s) } parámetros
```

expresión - Un enunciado (booleano) de lenguaje C que realiza una evaluación de verdadero o falso

Ejemplo

```
var = 0; while(var < 200){    // Ejecuta algo repetitivo 200 veces  var++; }
```

Enunciado "for"

El enunciado "for" se utiliza para repetir un bloque de enunciados encerrados entre llaves. Un contador de incremento se usa por lo general para monitorear los incrementos y terminar el lazo. El enunciado "for" es útil para cualquier operación repetitiva, y se usa frecuentemente en combinación con arreglos para operar recolección de datos desde los pines..

El encabezado del lazo "for" consta de tres partes: for (inicialización; condición; incremento) { //enunciado(s); } La inicialización sucede al principio y solo una vez.. Cada vez que se cumple una vuelta del lazo se evalúa la condición; si es verdadera, se ejecuta el bloque de enunciado y el incremento, para volver a probar la condición al continuar. Cuando la condición se hace falsa, se termina el lazo.

Ejemplo

```
// Baja la intensidad de un LED utilizando un pin PWM pin int PWMPin = 10; // LED en serie con resistor de 1k en pin 10 void setup()
{
    // no es necesario configurar } void loop()
{
    for (int i=0; i <= 255; i++)
{
    analog Write(PWMPin, i);
delay(10);
}
```

Tip de programación

El loop for de lenguaje C es mucho más flexible que los de otros lenguajes de programación, incluyendo a BASIC. Cualquiera de los tres elementos del encabezado puede ser omitido, aunque se requiere utilizar

<http://www.elegoo.com>

punto y coma. De igual manera, los enunciados para inicialización, condición e incremento pueden ser cualquier enunciado válido para C con

variables no relacionadas. Estos raros tipos de enunciado "for" inusuales pueden proporcionar soluciones a extraños problemas de programación.

Lección 10 Módulo de Timbre Pasivo

Resumen

En este experimento, aprenderemos a utilizar el módulo de timbre pasivo.

El propósito de este experimento es generar ocho sonidos diferentes, cada uno con duración de 0.5 segundos: empezando con Do mayor (523 Hz), Re (587 Hz), Mi (659 Hz), Fa (698 Hz), So (784 Hz), La (880 Hz), Si (988 Hz) y terminando con Do menor (1047 Hz).

Componentes requeridos:

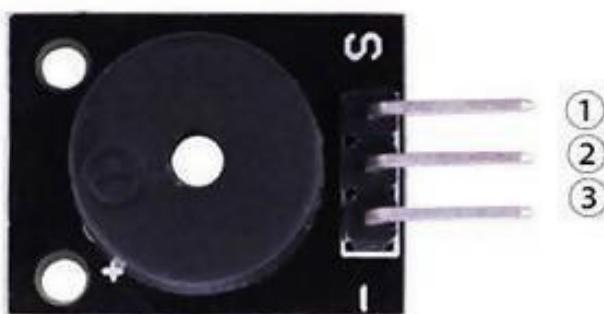
1x Elegoo Uno R3 1x Timbre pasivo

3x Cables F-M (Cables DuPont Hembra a Macho)

Introducción de Componente

Timbre Pasivo:

El principio de funcionamiento del timbre pasivo es utilizar un audio generador de PWM para hacer que el aire vibre. por tanto tiempo como la frecuencia de vibración se mantenga de forma apropiada. Por ejemplo, si se envía un pulso de 523 Hz, se genera un Do mayor; un pulso de 587 Hz genera un Re; un pulso de 659 Hz.



- 1.OUTPUT
- 2.VCC: 3.3V-5V DC
- 3.GND:ground

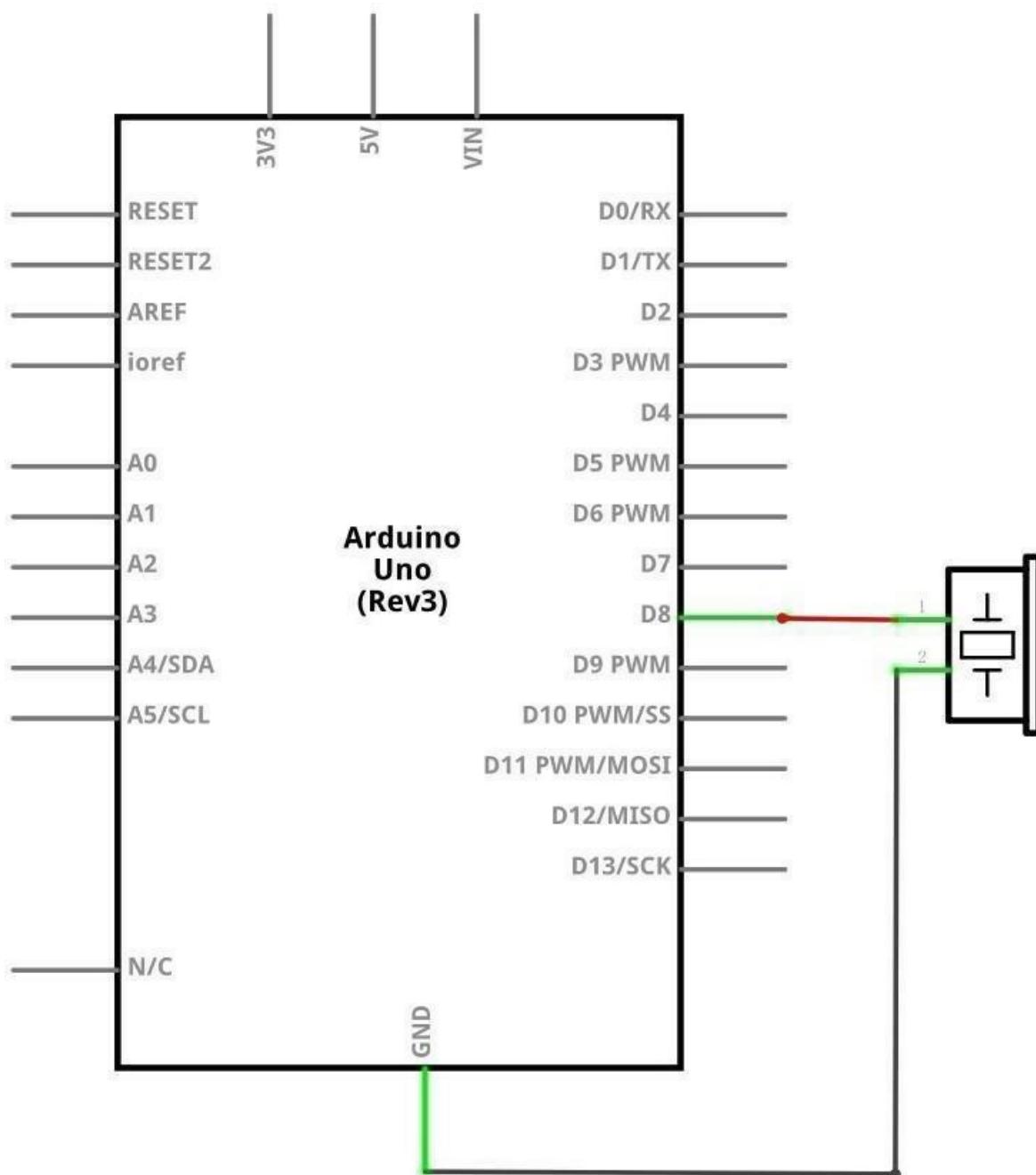
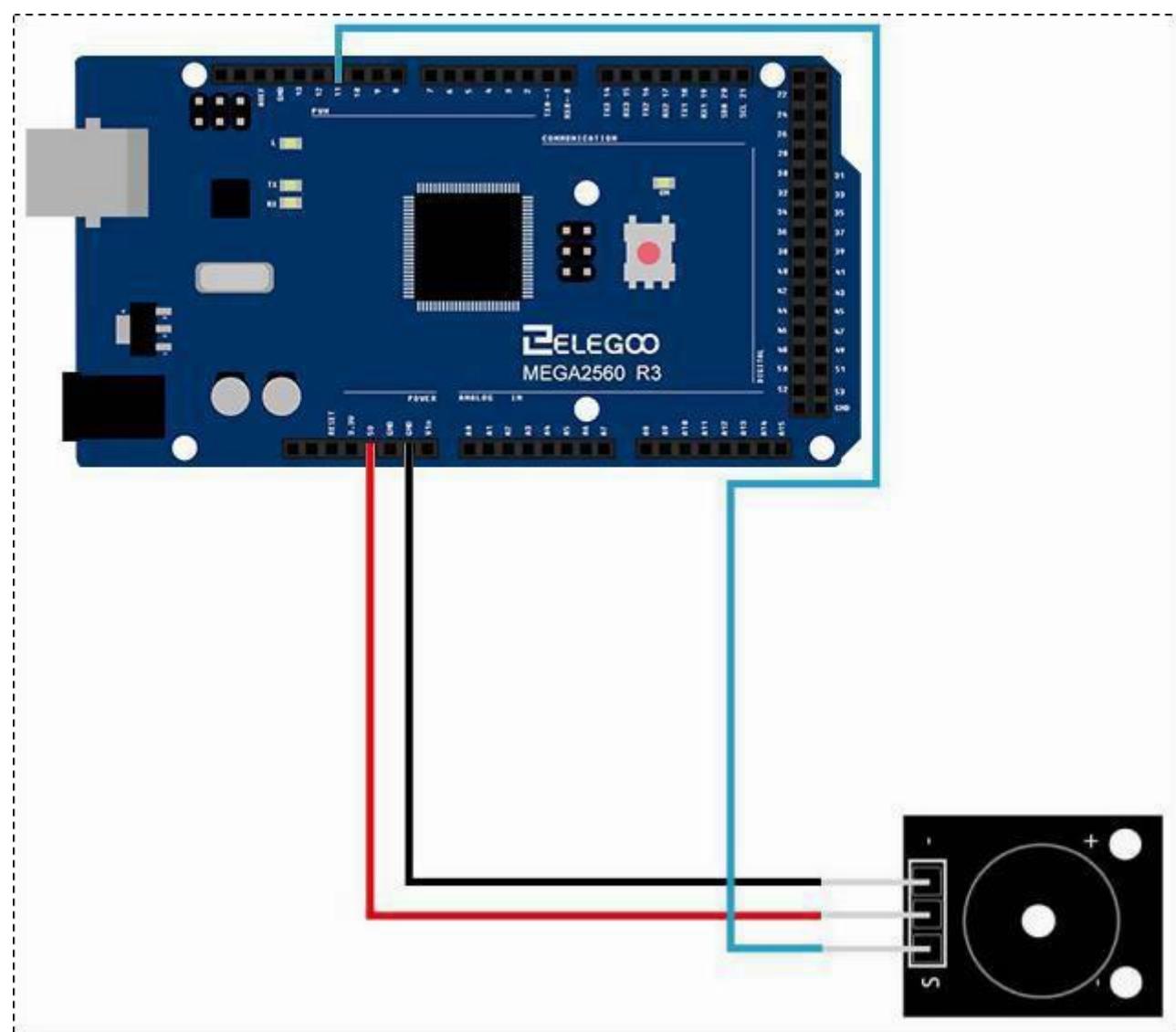
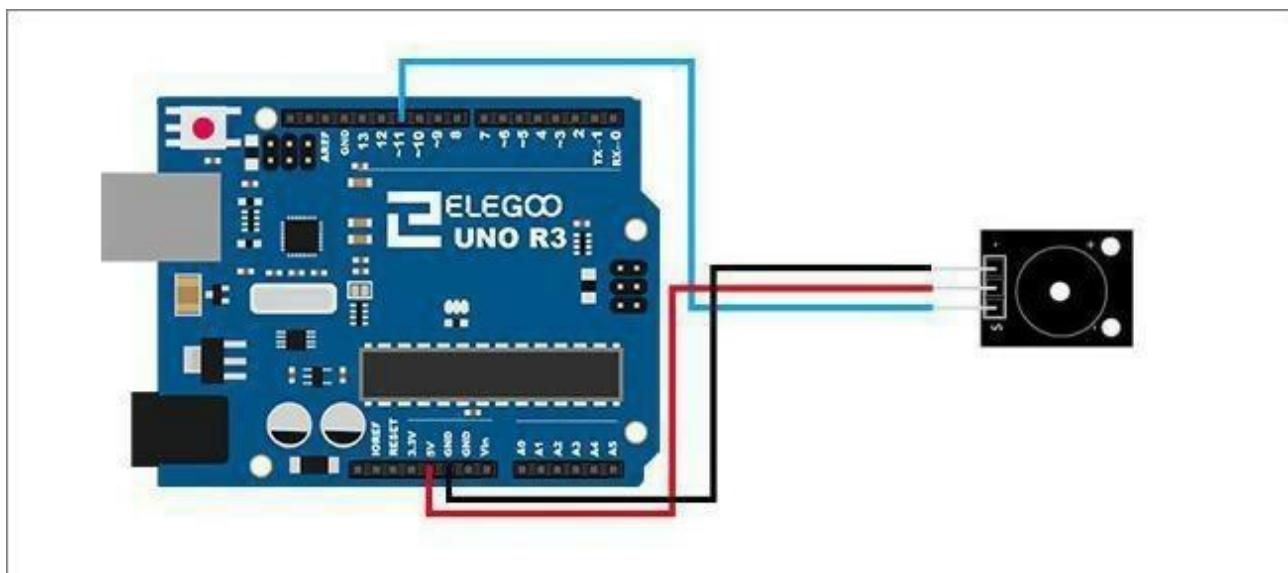


Diagrama de cableado

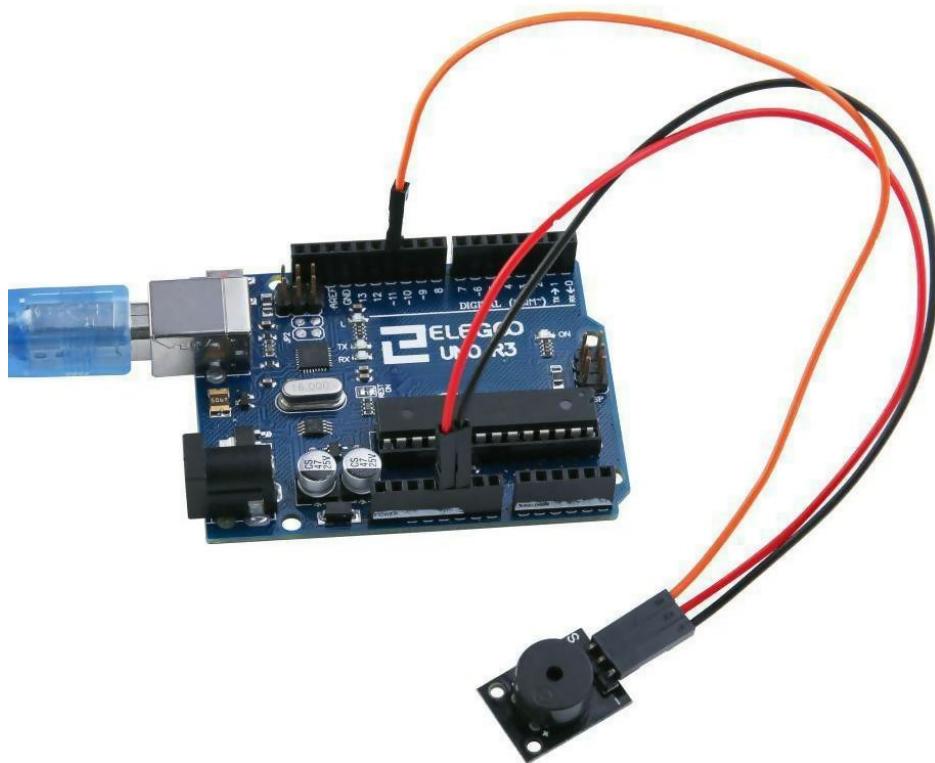


Código

Después de realizar el cableado, por favor abre el programa en la carpeta de código (Lección 10 Timbre Pasivo)

y haz clic en UPLOAD para cargar el programa. Si hay algún error, consulta los detalles en la lección 2 para cargar el programa y verificar donde está el error.

Antes de poder correrlo, asegúrate de tener la librería < pitches> instalada, o reinstálala de ser necesario. Instalada, o reinstálalas de ser necesario. De otra forma, tu código no funcionará. Para detalles del tutorial acerca de cómo cargar los archivos de librerías, consulta la lección 1. Imagen ejemplo



A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
#include "pitches.h"

// Notas en la melodía: int melody[] = {
NOTE_C5, NOTE_D5, NOTE_E5, NOTE_F5, NOTE_G5, NOTE_A5, NOTE_B5, NOTE_C6};
int duration = 500; // 500 milisegundos void setup() {
}

void loop() {
for (int thisNote = 0; thisNote < 8; thisNote++) {
// la voz sale por el pin 8, el tono de cada escala es de 0.5 segundos de(11, melody[thisNote],
duración);

// Envía la voz a la salida después de varios minutos de retraso(1000);
}

// Reinicia después de un retardo de dos segundos (2000);
}
```

A partir del programa mostrado arriba, es posible aprender la sintaxis de la estructura de control de la programación.

(1) Lazos while

Descripción

Los lazos while se mantendrán funcionando infinitamente hasta que la expresión entre los paréntesis () se haga falsa. Debe haber algún cambio en la variable a probar, o el lazo jamás se detendrá. Este cambio puede darse en tu código, como por ejemplo al incrementar una variable, o mediante una condición externa, como la señal de un sensor.

Sintaxis

```
while(expresión){    // enunciado(s) } parámetros
```

Expresión - Un enunciado (booleano) de lenguaje C que realiza una evaluación de verdadero o falso

Ejemplo

```
var = 0; while(var < 200){    // Ejecuta algo repetitivo 200 veces  var++; }
```

(2) Enunciado "for"

El enunciado "for" se utiliza para repetir un bloque de enunciados encerrados entre llaves. Un contador de incremento se usa por lo general para monitorear los incrementos y terminar el lazo. El enunciado "for" es útil para cualquier

operación repetitiva, y se usa frecuentemente en combinación con arreglos para operar recolección de datos desde los pines. El encabezado del lazo "for" consta de tres partes: for (inicialización; condición; incremento) { //enunciado(s); } La inicialización sucede al principio y solo una vez.. Cada vez que se cumple una vuelta del lazo se evalúa la condición; si es verdadera, se ejecuta el bloque de enunciado y el incremento, para volver a probar la condición al continuar. Cuando la condición se hace falsa, se termina el lazo.

Ejemplo

```
// Baja la intensidad de un LED utilizando un pin PWM pin int PWMPin = 10; //
LED en serie con resistor de 1k en pin 10 void setup()

{ // no es necesario configurar } void loop()

{ for (int i=0; i <= 255; i++)

{

analog Write(PWMPin, i); delay(10);
```

}

}

Tip de programación

El loop for de lenguaje C es mucho más flexible que los de otros lenguajes de programación, incluyendo a BASIC. Cualquiera de los tres elementos del encabezado puede ser omitido, aunque se requiere utilizar punto y coma. De igual manera, los enunciados para inicialización, condición e incremento pueden ser cualquier enunciado válido para C con variables no relacionadas. Estos raros tipos de enunciado "for" inusuales pueden proporcionar soluciones a extraños problemas de programación.

Lección 11 Módulo Laser

Resumen

En este experimento, aprenderemos a utilizar el módulo laser.

Emisor de láser

Conecta el primer pin ("") del módulo al pin GND de la tarjeta y el segundo pin al pin de 5V en la tarjeta. Por último, conecta el tercer pin ("S") al puerto digital 9.

Longitud de onda de transmisión: 650 nm



- 1.GND:ground
- 2.VCC:3.3V-5V DC
- 3.OUTPUT

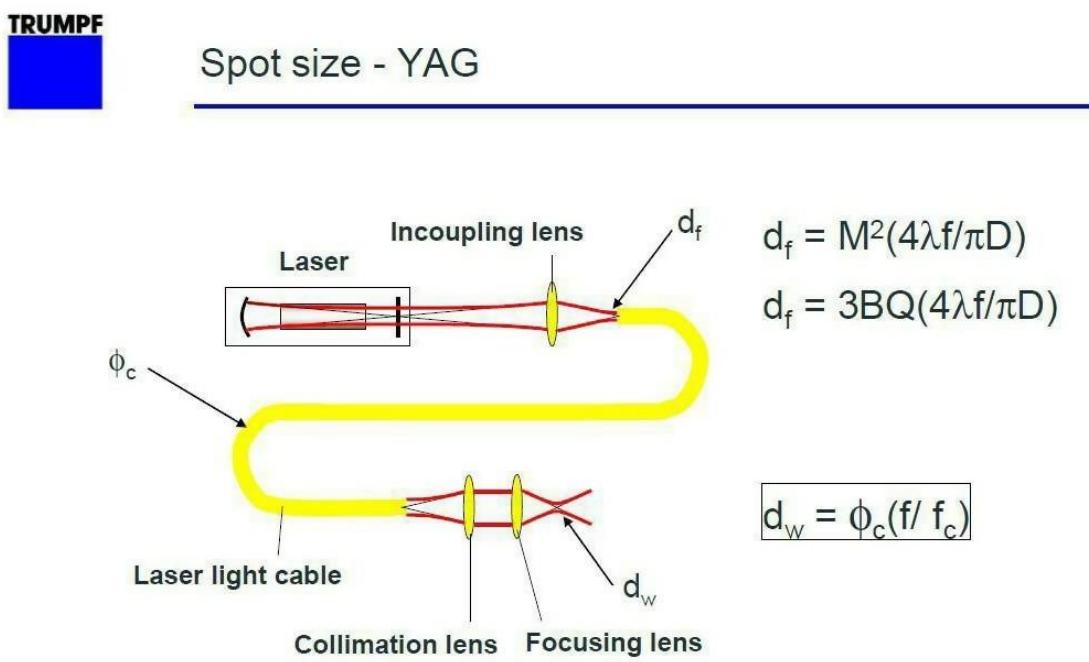
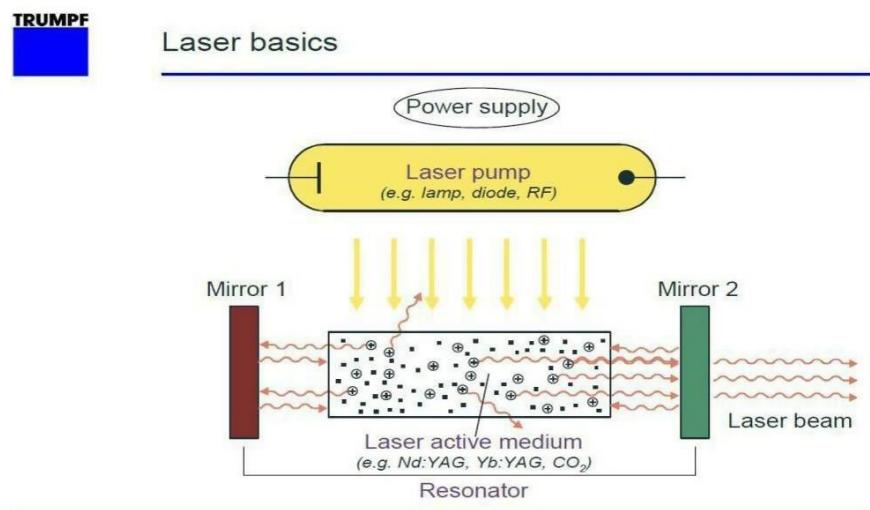
Componentes Requeridos:

1x Elegoo Uno R3 1x cable USB

1x Módulo Láser 3x Cables F-M

Introducción de Componente

Sensor Láser:



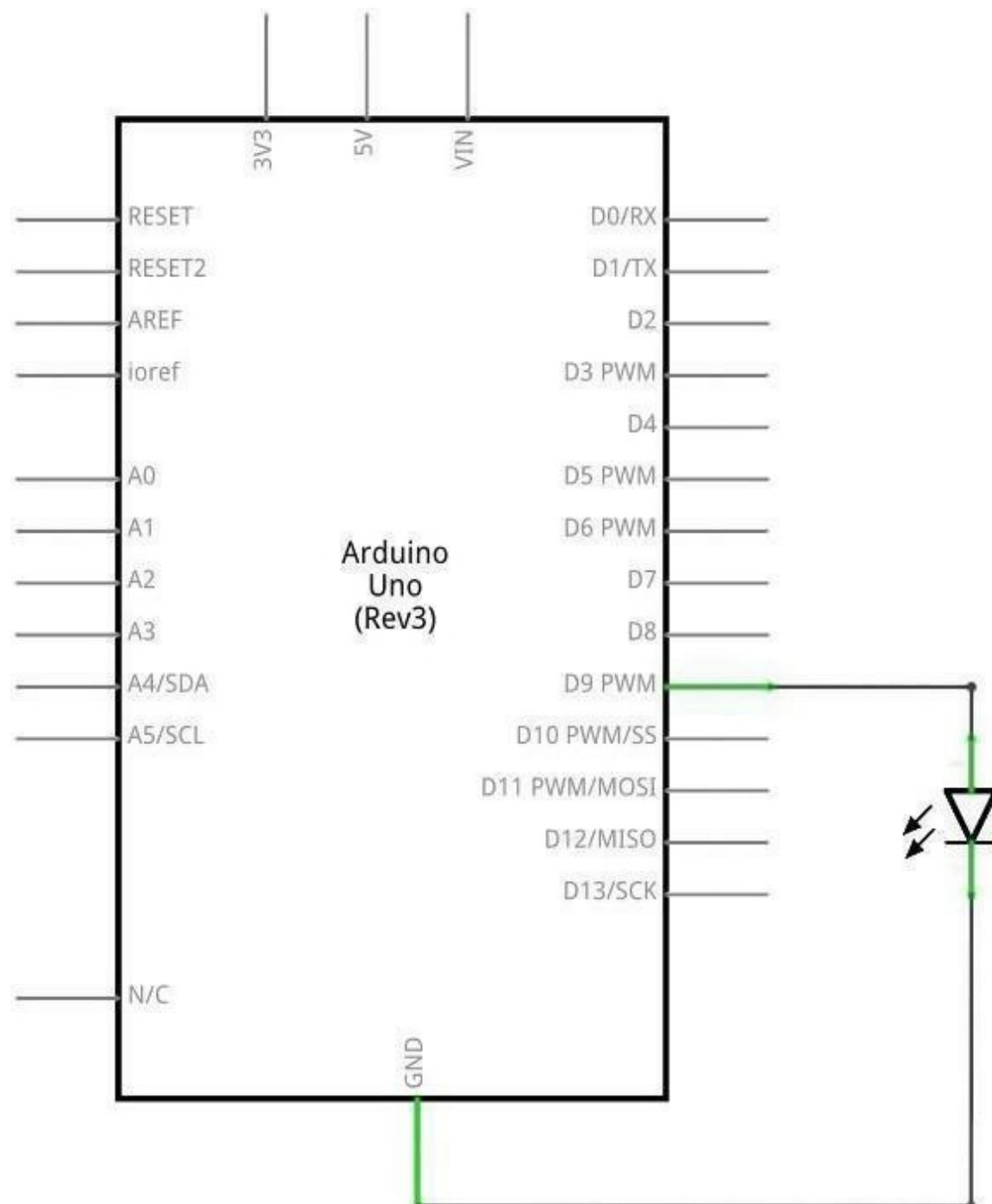
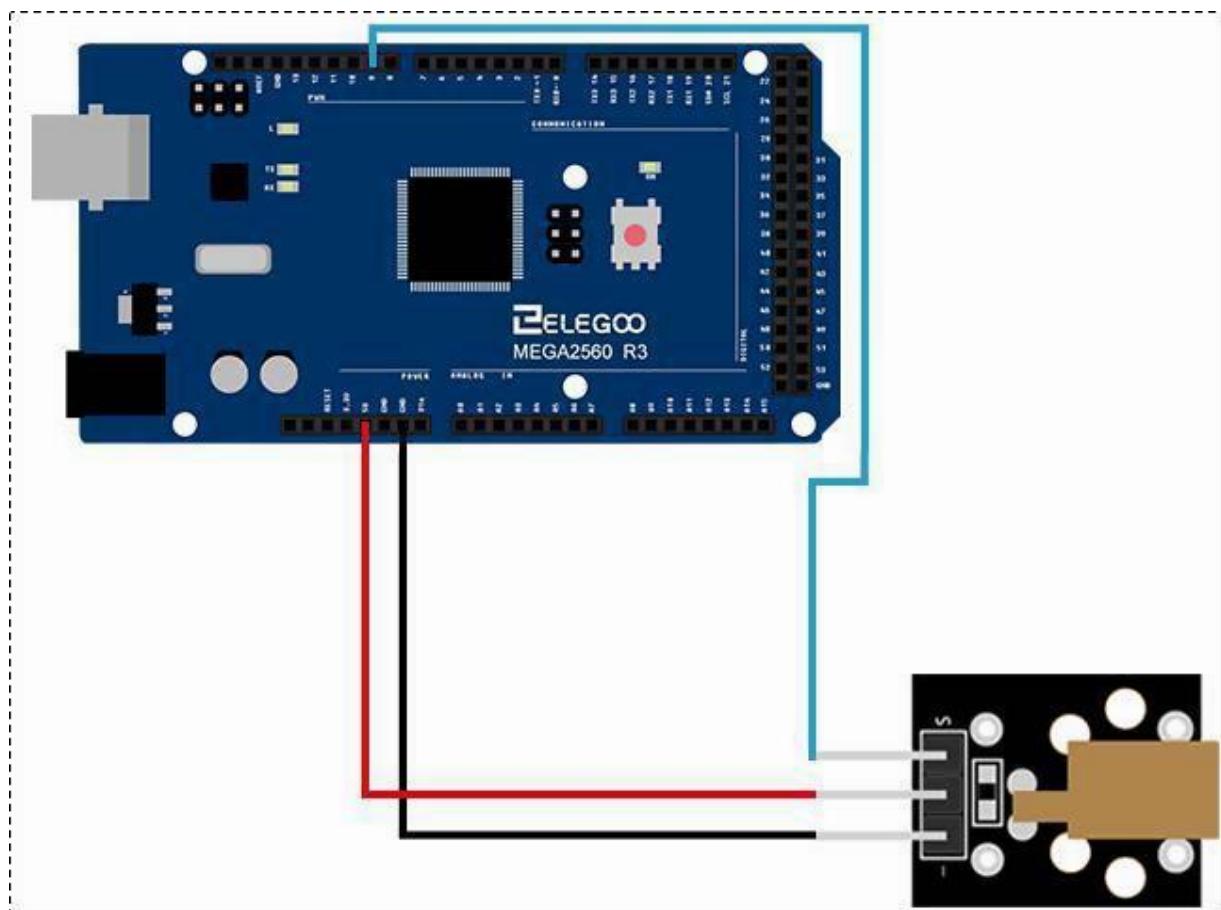
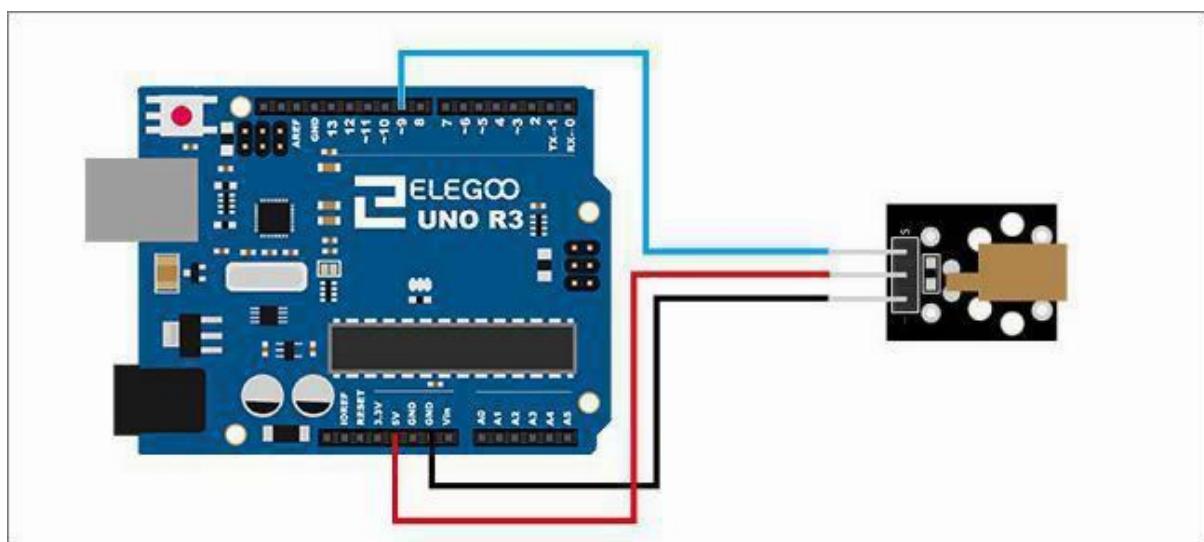


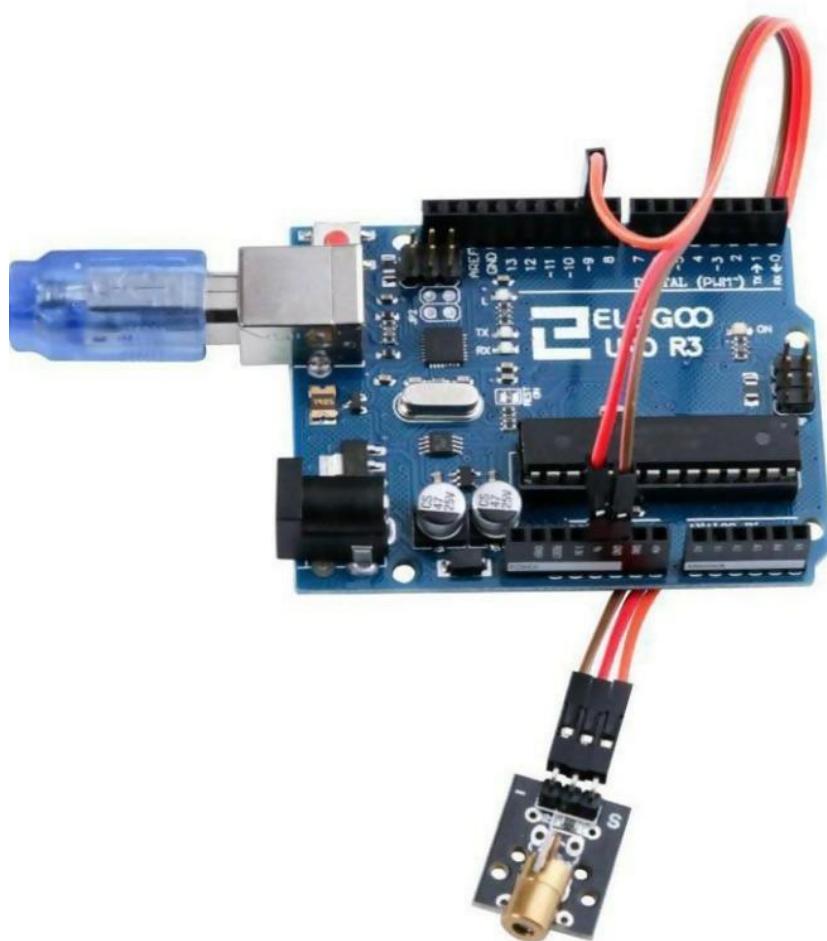
Diagrama de cableado



Código

Mientras conectamos el circuito, busquemos la "Lección 11 Módulo Láser" en la carpeta de códigos del tutorial y carguemosla. Verás como el módulo laser comienza a emitir luz

Imagen ejemplo



A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
/*Define una variable integral pos y le asigna el valor cero*/ int pos = 0;  
/* void setup () La función de configuración () se llama al arrancar la tarjeta Arduino. Se usa para  
 inicializar variables, patrones de pines, comenzar a usar librerías y mucho más. Esta función  
 corre solo una vez al encender o reiniciar la tarjeta Arduino */  
void setup()  
{  
/*Define los pines del módulo y configura el tipo de salida*/  
pinMode (9,OUTPUT);  
  
}  
  
void loop()  
{  
  
for (pos = 0; pos <= 255; pos += 1)  
{  
/*Lee el valor de la variable del módulo*/ analog Write(9,pos);  
delay(25);  
  
}  
for (pos = 255; pos >= 0; pos -= 1)  
{  
  
analog Write(9,pos); delay(25);  
}  
}
```

Lección 12 Módulo SMD RGB y Módulo RGB

En este experimento, aprenderemos a utilizar el módulo LED SMD RGB y el módulo LED RGB

En realidad, la función de los módulos LED SMD RGB y RGB es casi la misma. Pero podemos escoger la forma que nos guste o necesitemos

Los módulos SMD RGB LED y RGB LED están hechos de LEDs a todo color. Al ajustar el voltaje de entrada a los pines R, G, B, podemos ajustar la fuerza de los tres colores primarios (rojo/azul/verde) para así poder implementar un resultado a todo color.

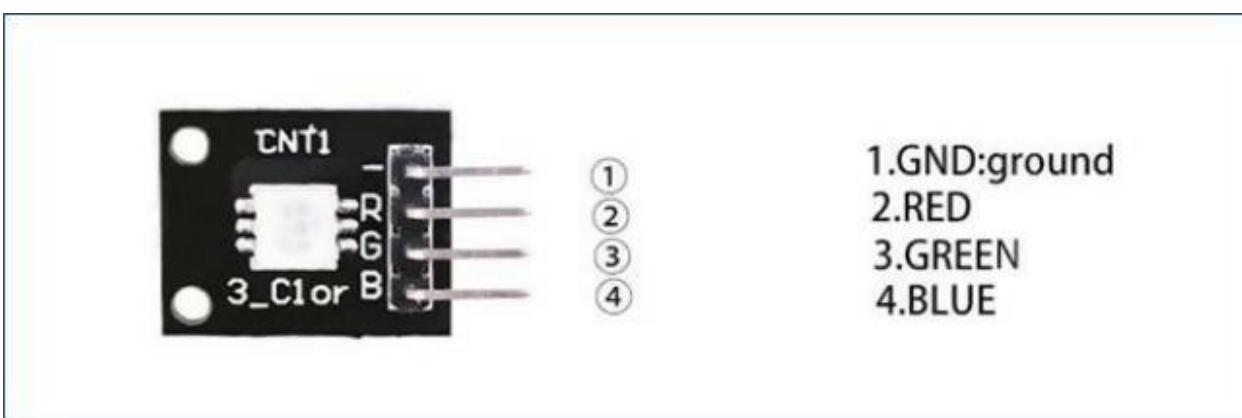
Módulo RGB LED

RGB-LED con lente clara y resistor en serie de 150 ohm incorporado para operar a 5V.



Módulo SMD RGB LED

RGB-LED con encapsulado SMD y sin resistores en serie.



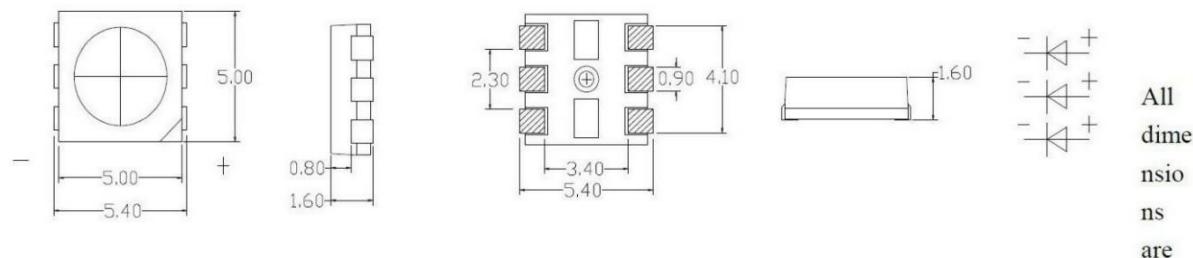
Componentes Requeridos:

1x Elegoo Uno R3 1x cable USB

1x Módulo SMD RGB 1x Módulo RGB

4x Cables F-M

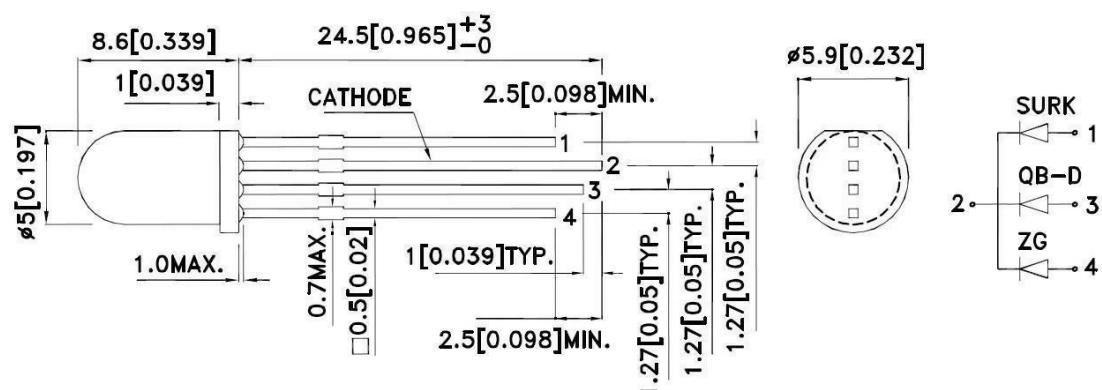
Introducción de ComponenteSMD RGB:



in millimeter

Tolerance is $\pm 0.25\text{mm}$ ($0.10''$) unless otherwise noted

| Parameter | Symbol | Value | Unit |
|-----------------------|-----------------|--------------------|------|
| Forward Current | If | 20 | mA |
| Reverse Voltage | Vr | 5 | V |
| Operating Temperature | Topr | -25~+85 | °C |
| Storage Temperature | Tstg | -35~+85 | °C |
| Soldering temperature | Tsol | 260±5°C (for 4sec) | °C |
| Power Dissipation | Pd | R=40 C/D=60 | mW |
| Pulse Current | I _{FP} | 100 | mA |

RGB LED:**Electrical / Optical Characteristics at TA=25°C**

| Symbol | Parameter | Device | Typ. | Max. | Units | Test Conditions |
|-----------------------|--------------------------|----------------------------|--------------------|-----------------|-------|------------------|
| λ_{peak} | Peak Wavelength | Hyper Red Blue Green | 650 468 515 | | nm | $I_F=20mA$ |
| λ_D [1] | Dominant Wavelength | Hyper Red Blue Green | 630 470 525 | | nm | $I_F=20mA$ |
| $\Delta\lambda_{1/2}$ | Spectral Line Half-width | Hyper Red Blue Green | 28 25 30 | | nm | $I_F=20mA$ |
| C | Capacitance | Hyper Red Blue Green | 35 100 45 | | pF | $V_F=0V; f=1MHz$ |
| V_F [2] | Forward Voltage | Hyper Red Blue Green | 1.95 3.3 3.3 | 2.5 4 4.1 | V | $I_F=20mA$ |
| I_R | Reverse Current | Hyper Red Blue Green | | 10 50 50 | uA | $V_R=5V$ |

Notes:

1. Wavelength: +/-1nm.
2. Forward Voltage: +/-0.1V.

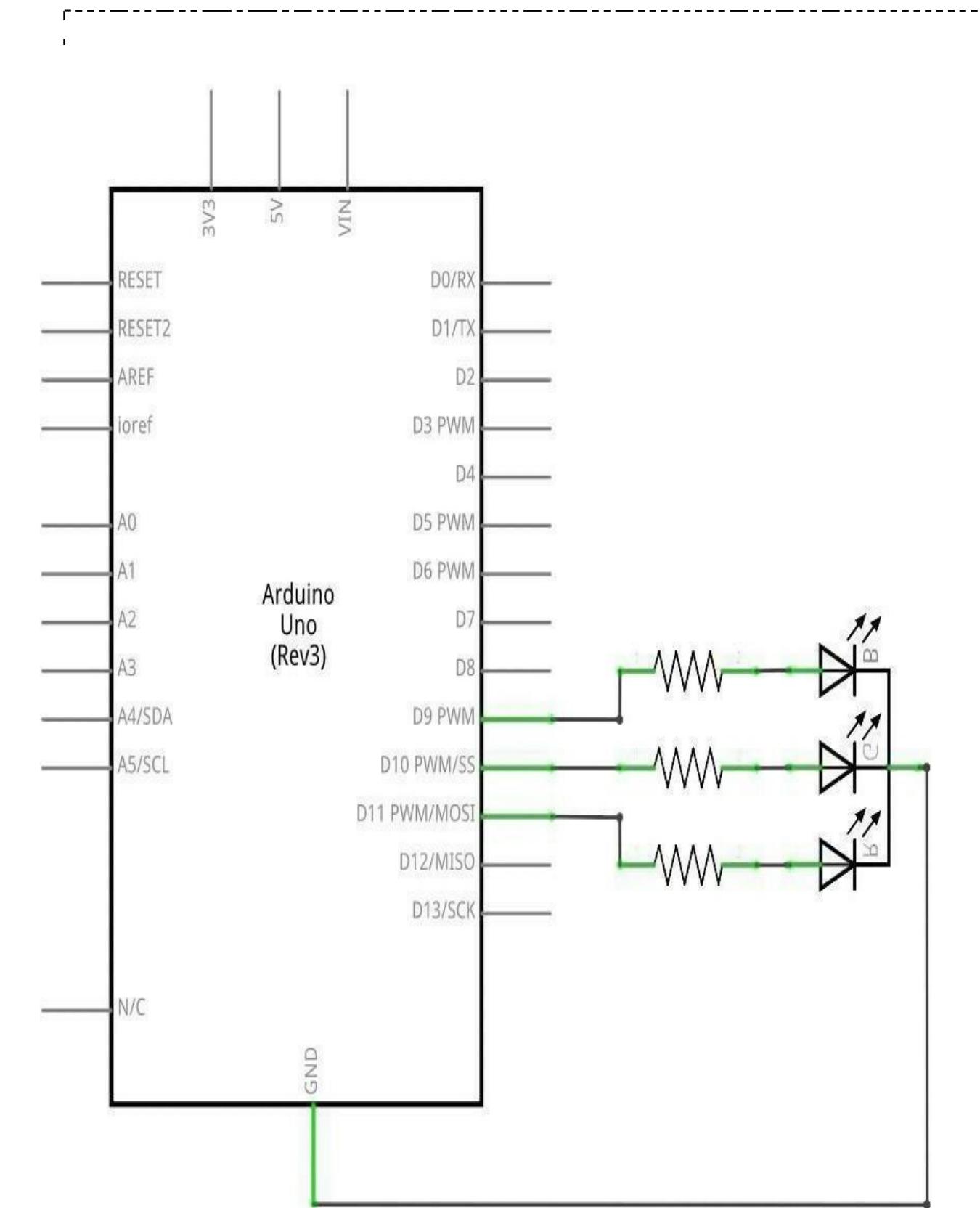
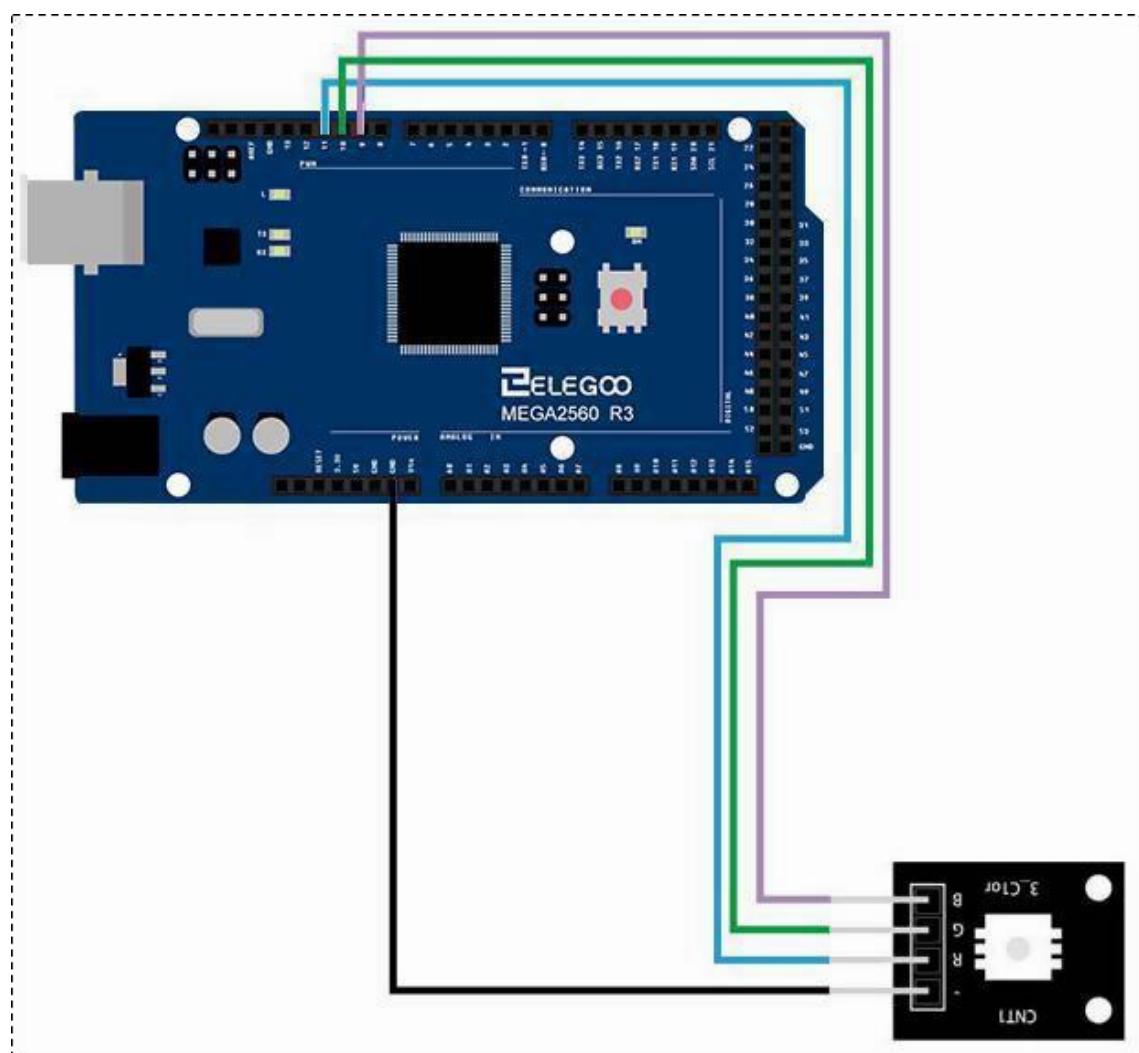
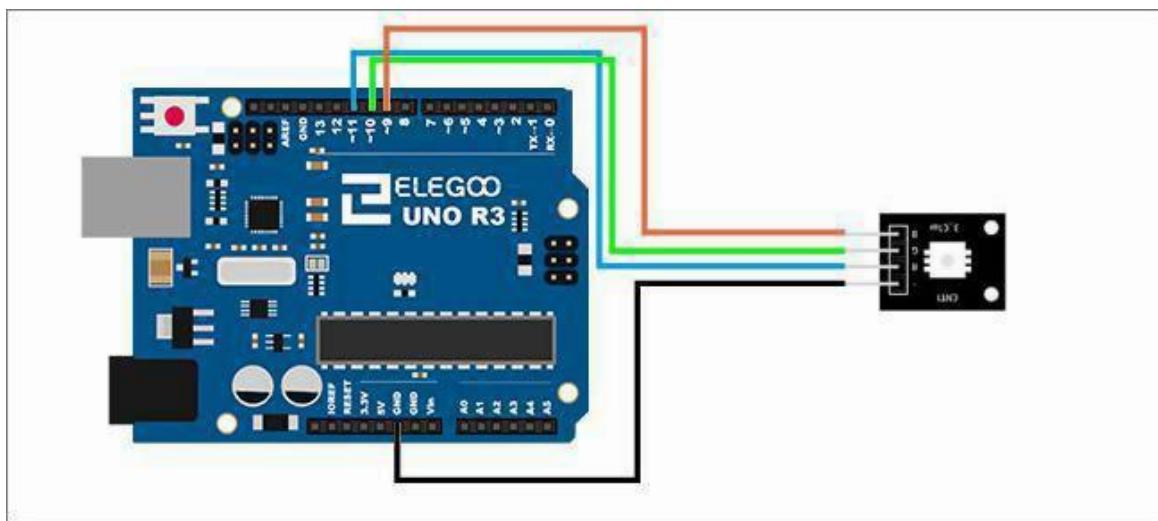
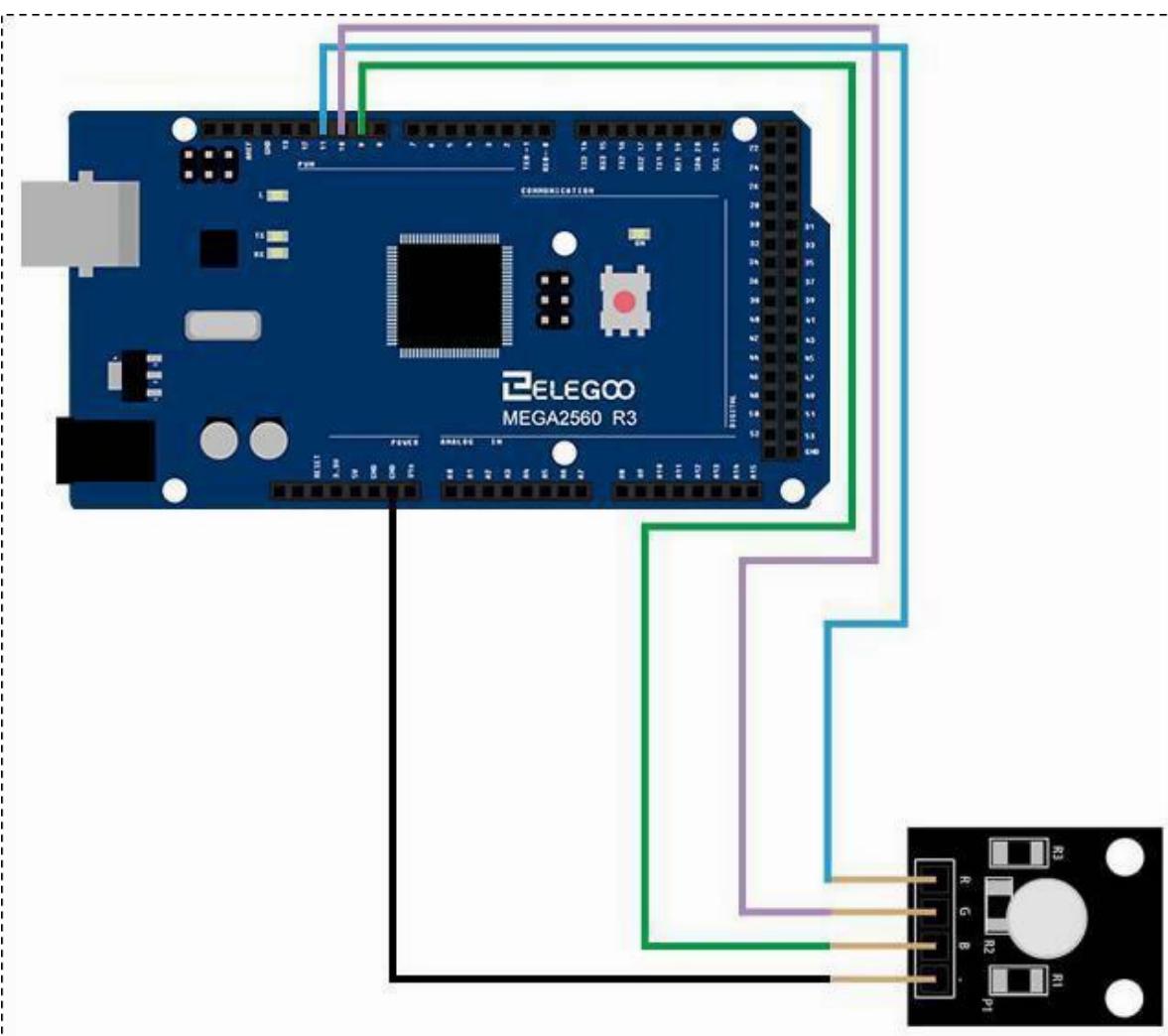
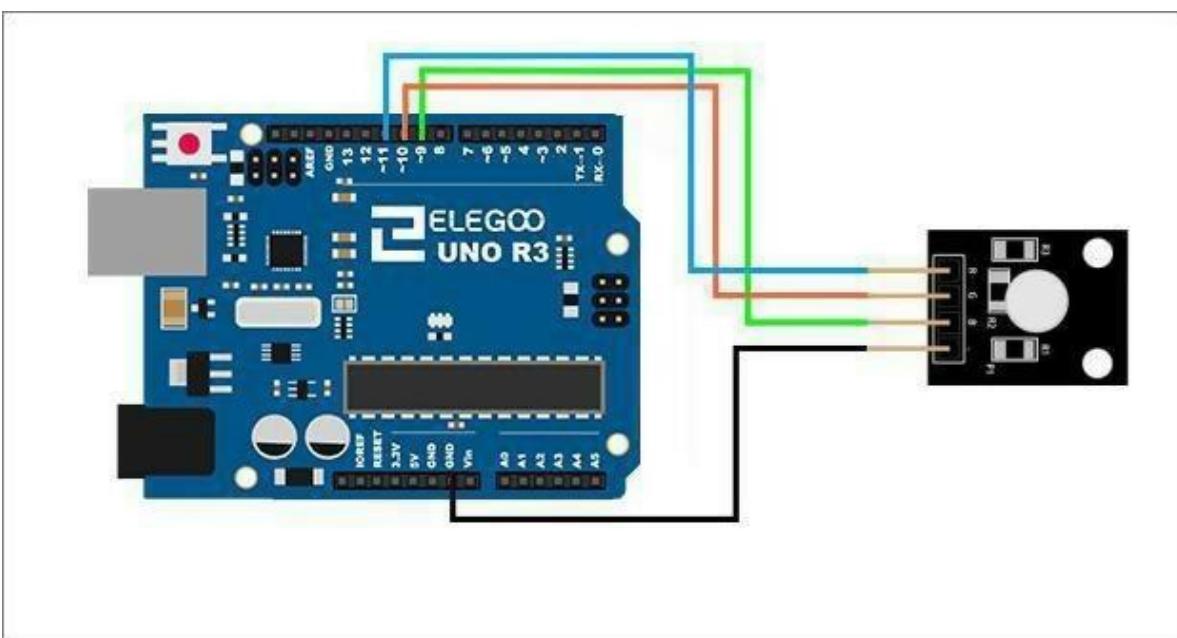


Diagrama de cableado

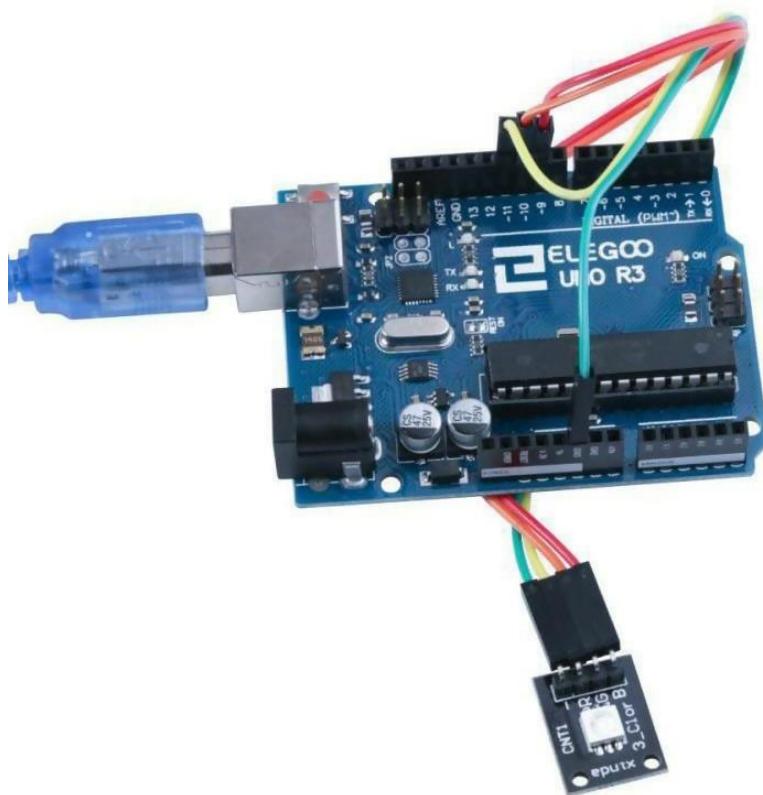


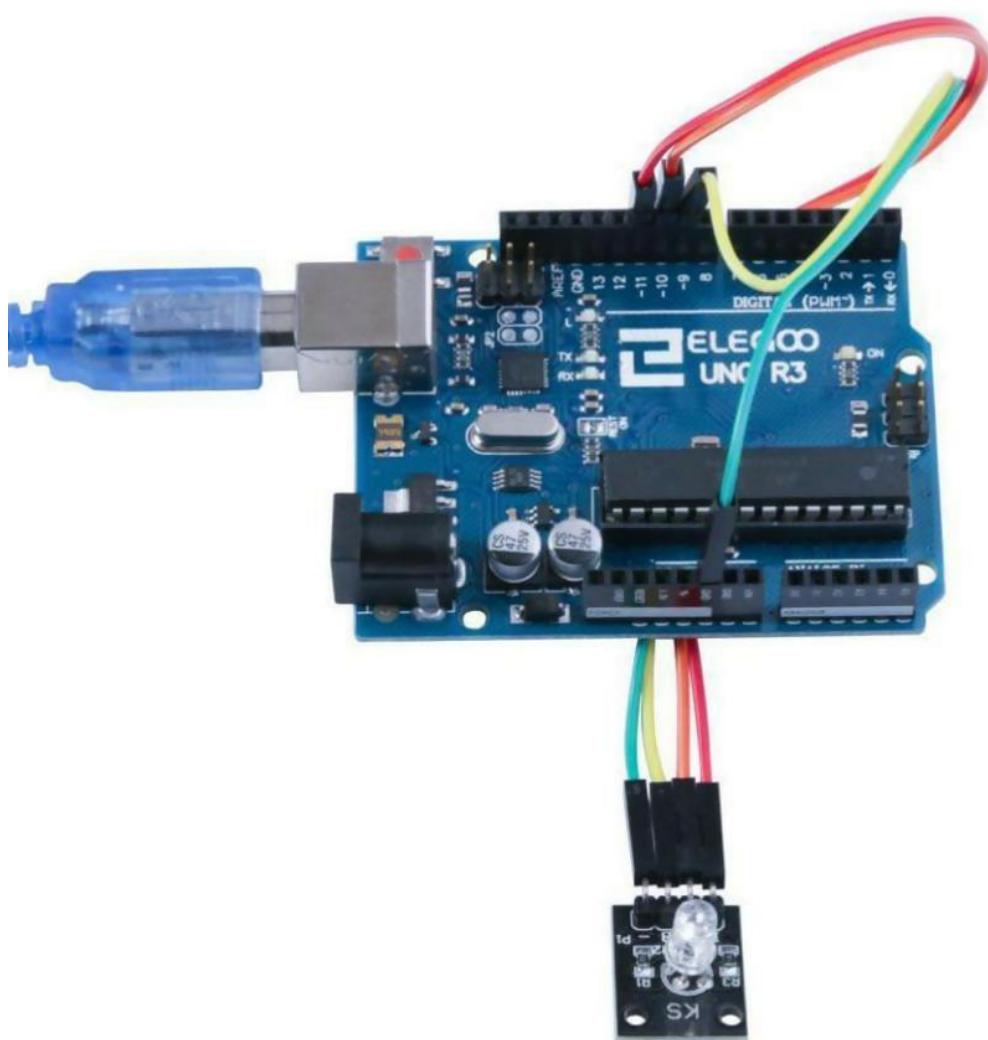


Resultados

Después de conectar los circuitos, abrimos la carpeta de códigos de nuestra documentación para encontrar la carpeta "Lección 12 Módulo SMD RGB y Módulo RGB". Abre el programa, cárgalo y córrelo. Podrás ver que el módulo cambia de color de acuerdo a lo establecido en el código. Si quieres que el color se comporte de otra forma, puedes modificar el código.

Imagen ejemplo





A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
/* Selecciona el pin para el LED rojo */ int redpin = 11;  
/* Selecciona el pin para el LED verde */ int greenpin =10;  
/* Selecciona el pin para el LED azul */ int bluepin =9;  
/*Define una variable integral val*/ int val;  
void setup(){  
/*Set redpin, bluepin, greenpin output type*/ pinMode(redpin, OUTPUT); pinMode(bluepin, OUTPUT);  
pinMode(greenpin, OUTPUT); Serial.begin(9600);  
}  
void loop(){ for(val=255; val>0; val--)  
{analogWrite(11, val); analogWrite(10, 255-val); analogWrite(9, 128-val); delay(50);  
}  
for(val=0; val<255; val++)  
{  
analogWrite(11, val); analogWrite(10, 255-val); analogWrite(9, 128-val); delay(50);  
}  
Serial.println(val, DEC);  
}
```

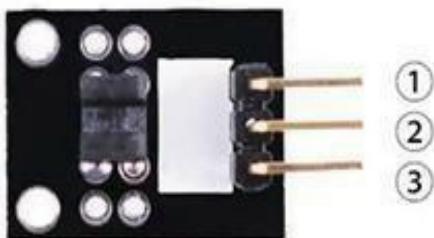
Lección 13 Módulo de Foto-interruptor

Resumen

En este experimento, aprenderemos a utilizar el módulo de foto-interruptor.

Bloqueo de luz

Barrera de luz con ranuras El pin del medio se conecta a los +5V de suministro y el marcado con el símbolo '-' se conecta a tierra. The output signal (with a 10 K ohm pullup to +5 V) is available on the pin on the right.



- 1.OUTPUT
- 2.VCC:3.3V-5V DC
- 3.GND:ground

Componentes Requeridos:

1 x Elegoo Uno R3 1 x cable USB

1 x Módulo Foto-interruptor 3 x Cables F-M

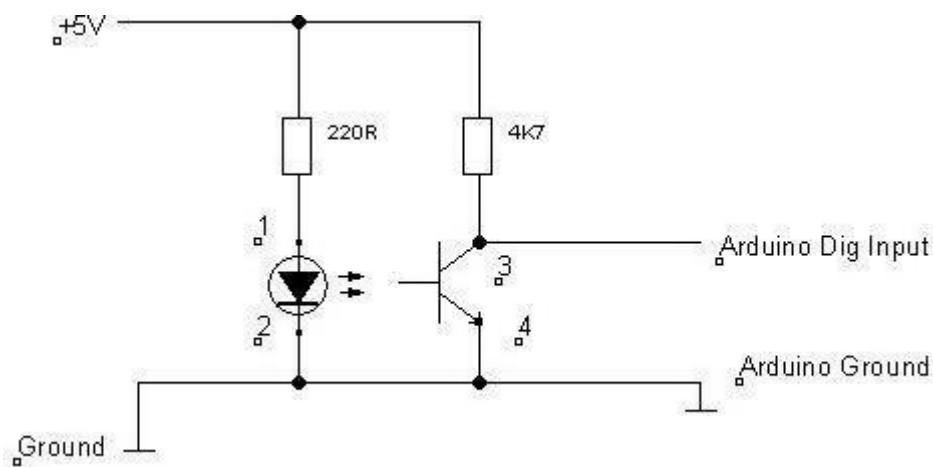
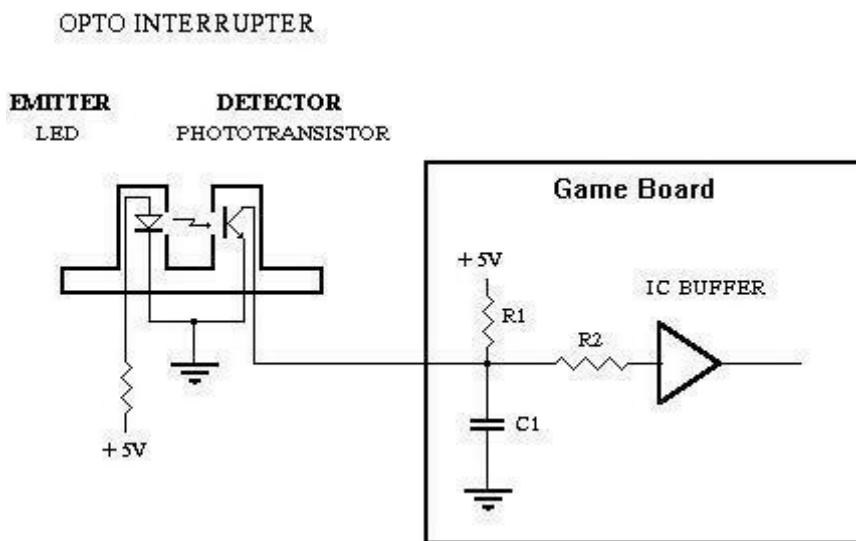
Introducción de componente

Sensor Opto interruptor:



Los opto interruptores se usan comúnmente en muchos juegos de arcade como por ejemplo en la columna de dirección de los antiguos juegos de conducción de vehículos, o en los interruptores para anotar puntos en Whack A Crock, etc. Un rayo de luz ininterrumpido encenderá el fototransistor al conectar la tierra hacia la entrada de la tarjeta de juegos. Si el rayo de luz se interrumpe, el fototransistor se apaga, por lo cual se desconecta la tierra de la entrada y el resistor pull-up R1 forzará la misma a "HIGH" (nivel de tensión de 5V).

Principio



El módulo foto-interruptor y el puerto 13 forman parte de un simple circuito con el LED integrado. Para hacer una interrupción más rápida, podemos conectar el puerto digital número 13 al LED integrado y el puerto S de la foto-interruptor al puerto 3 de la tarjeta Elegoo Uno. Cuando el interruptor sense, el LED parpadeará de acuerdo a su señal.

Esquema de conexión

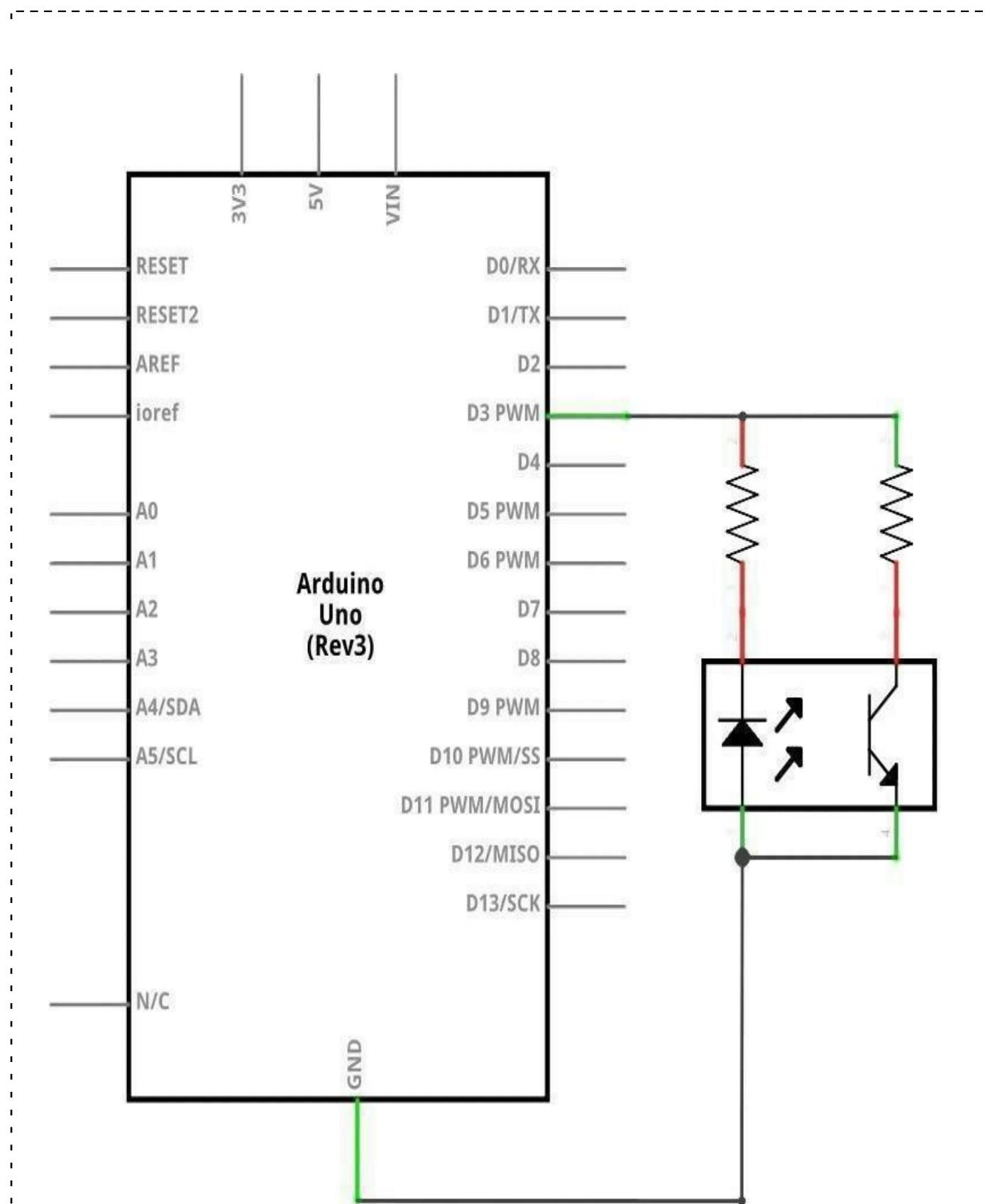
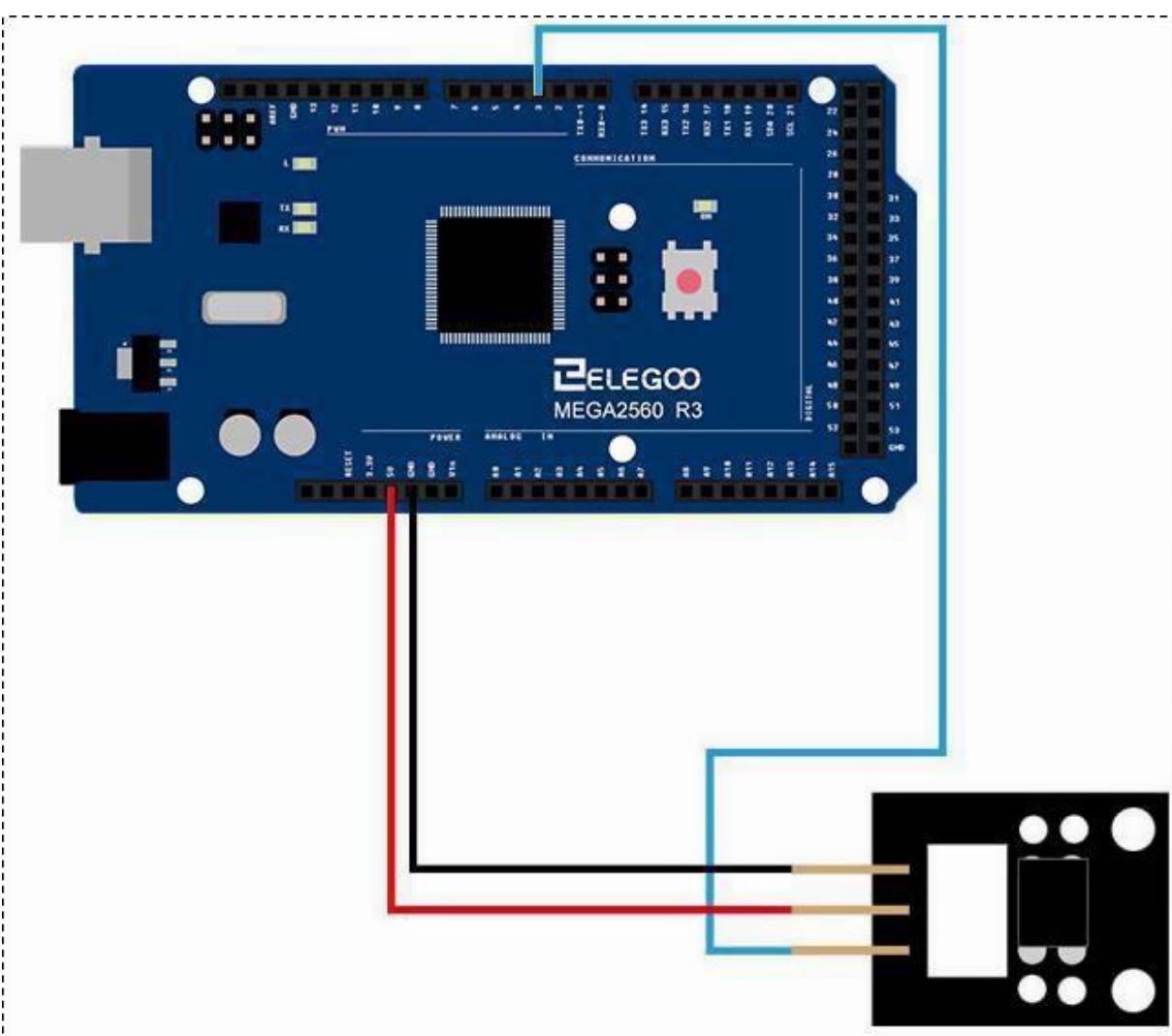
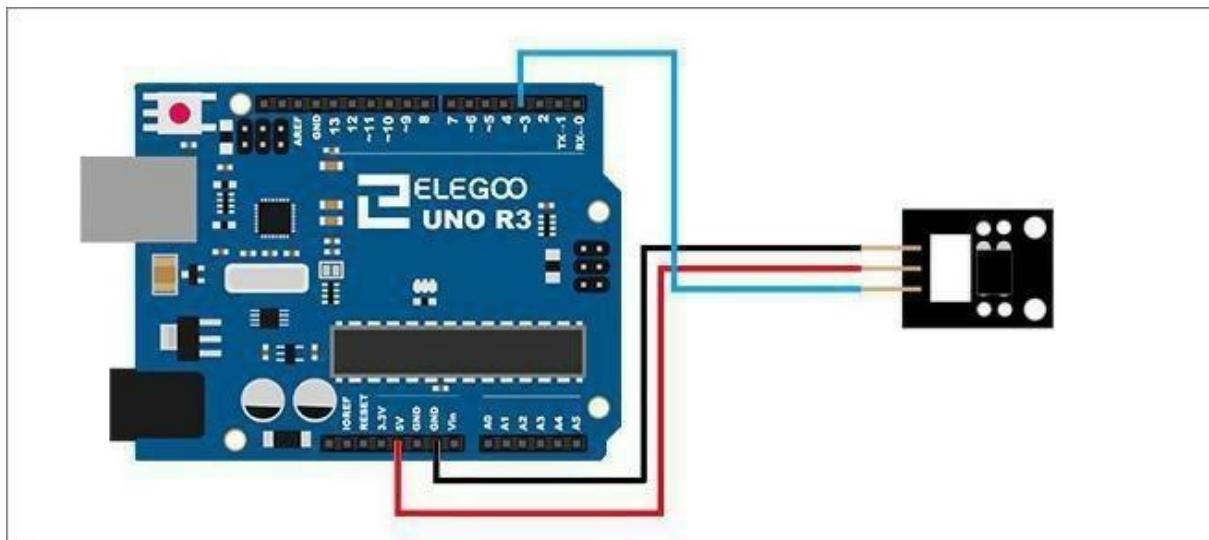


Diagrama de cableado

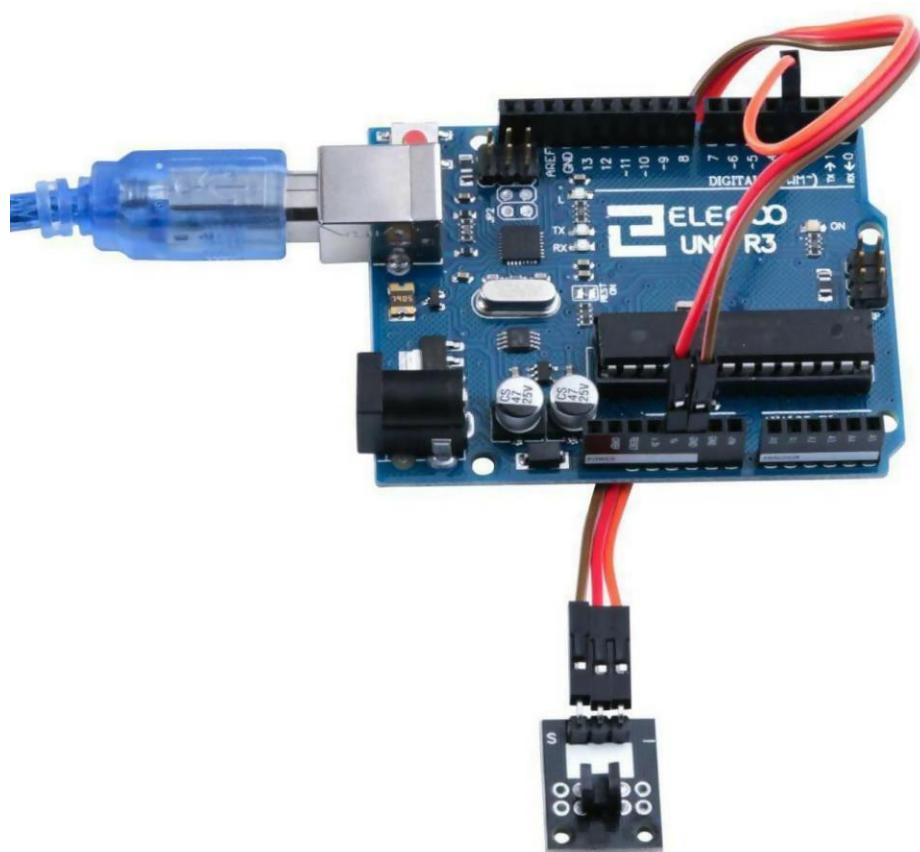


Resultados

Cuando termines de hacer el cableado, abre el tutorial y ve a code > Lección 13 Módulo foto-interruptor y corre el programa (el archivo .ino).

Coloca un pedazo de papel en la ranura del módulo y luego retíralo; verás como el LED 13 se enciende y apaga mientras lo haces

Imagen ejemplo



A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
/* define el puerto del LED */ int Led=13;  
/* define el puerto del módulo bloqueador de luz */ int buttonpin=3;  
/* Define la variable digital val */ int val;  
void setup ()  
{  
/* Define digital variable val */ pinMode(Led,OUTPUT);  
/* Define el módulo bloqueador de luz como un puerto de salida */ pinMode(buttonpin,INPUT);  
}  
void loop()  
{  
/* Lee el valor de la interfaz digital 3 asignada a val */ val=digitalRead(buttonpin);  
/* Cuando el sensor de bloqueo de luz tiene señal, el LED destella */ if(val==HIGH)  
{ digitalWrite(Led,LOW);  
} else  
{ digitalWrite(Led,HIGH);  
}  
}
```

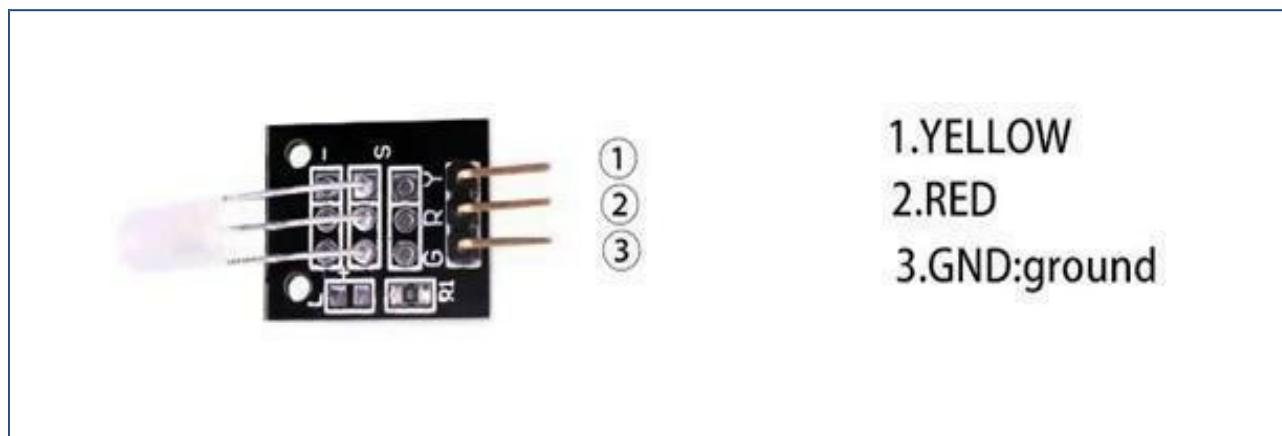
Lección 14 Módulo LED de Dos Colores (5 mm)

Resumen

En este experimento, aprenderemos a utilizar un LED de cátodo común de dos colores.

Módulo de LED de dos colores 5 mm

El LED de 5 mm tiene un cátodo común conectado a el pin ‘-‘ del PCB. El pin central se conecta al ánodo rojo y el pin ‘S’ al ánodo verde. No se incluyen resistores en serie en el circuito. Un valor adecuado de resistencia para operación a bajo voltaje sería de 220 ohm



Componentes requeridos:

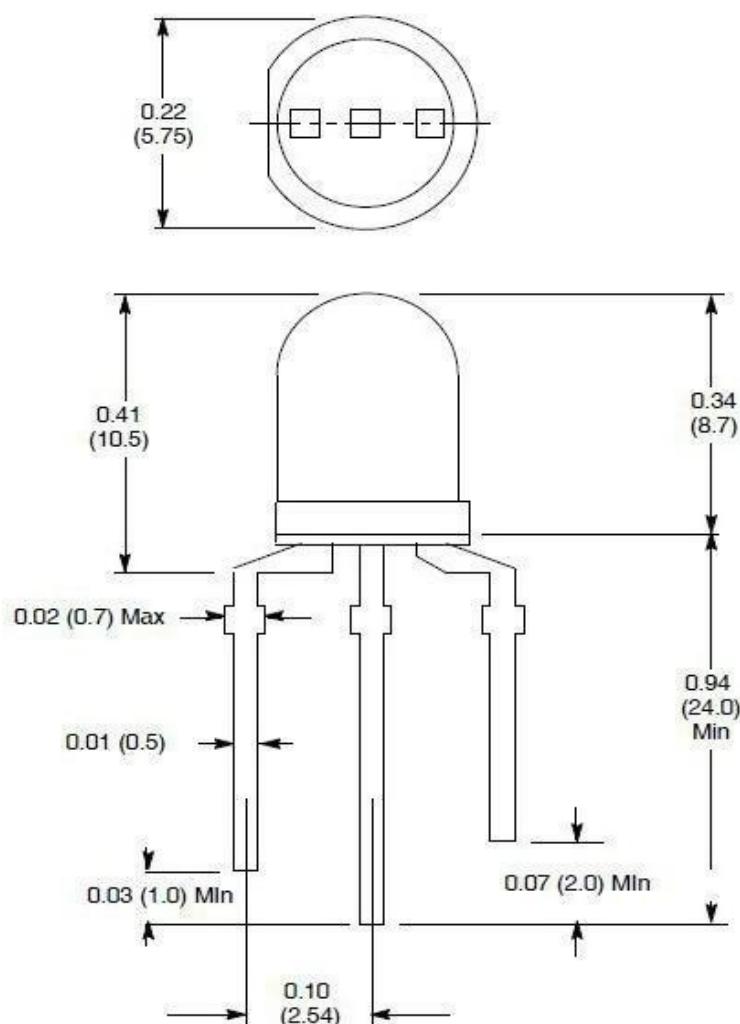
- 1x Elegoo Uno R3
- 1x cable USB
- 1 x LED de cátodo común de dos colores
- 2x Cables F-M

Introducción de componente

LED de cátodo común de dos colores

Electro–Optical Characteristics: ($T_A = +25^\circ\text{C}$ unless otherwise specified)

| Parameter | Symbol | Test Conditions | Min | Typ | Max | Unit |
|--|-------------------------|-----------------|-----|------|------|------|
| View Angle of Half Power | $2\theta_{1/2}$ | IF = 20mA | – | 40 | – | deg |
| Forward Voltage High Efficiency Red | VF | IF = 20mA | – | 2.05 | 2.80 | V |
| Yellow–Green | | | – | 2.15 | 2.80 | V |
| Luminous Intensity (Note 1) | IV | IF = 20mA | 35 | 60 | – | mcd |
| Peak Emission Wavelength High Efficiency Red | λ_p | IF = 20mA | – | 625 | – | nm |
| Yellow–Green | | | – | 570 | – | nm |
| Dominant Wave Length (Note 2) High Efficiency Red | $\lambda_d(\text{HUE})$ | IF = 20mA | – | 618 | – | nm |
| Yellow–Green | | | – | 567 | – | nm |



1. Red +
2. Common Lead -
3. Green +

Esquema de Conexión

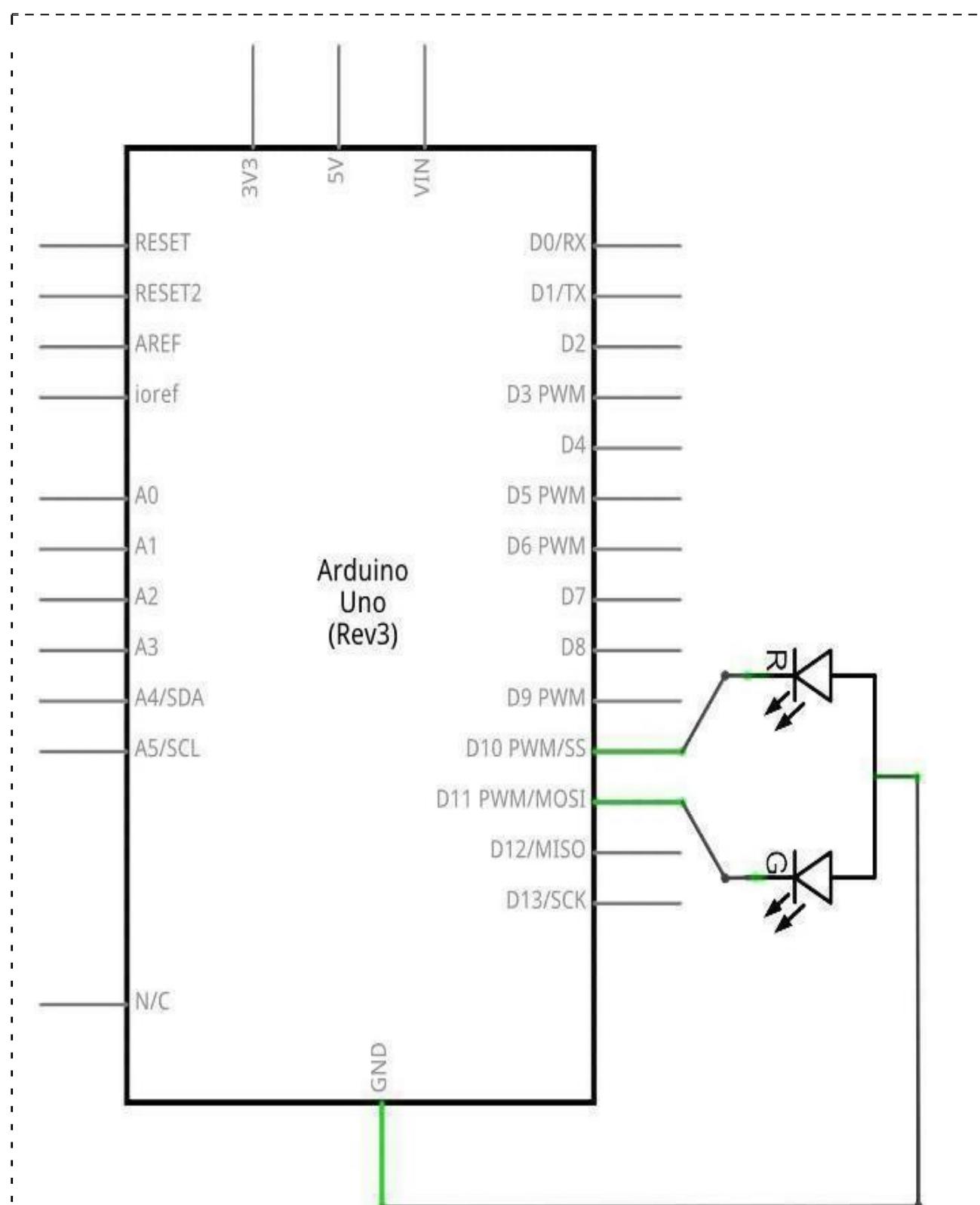
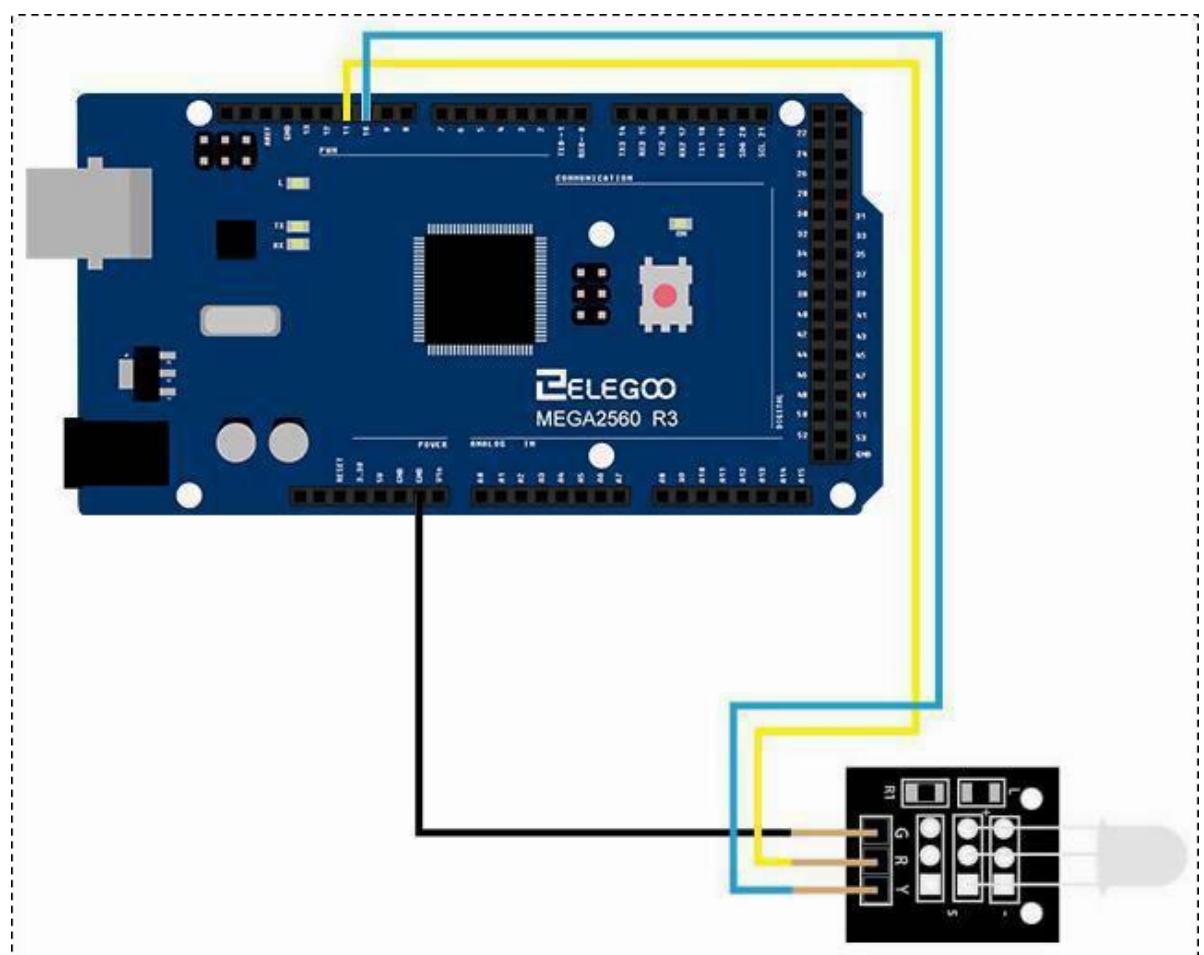
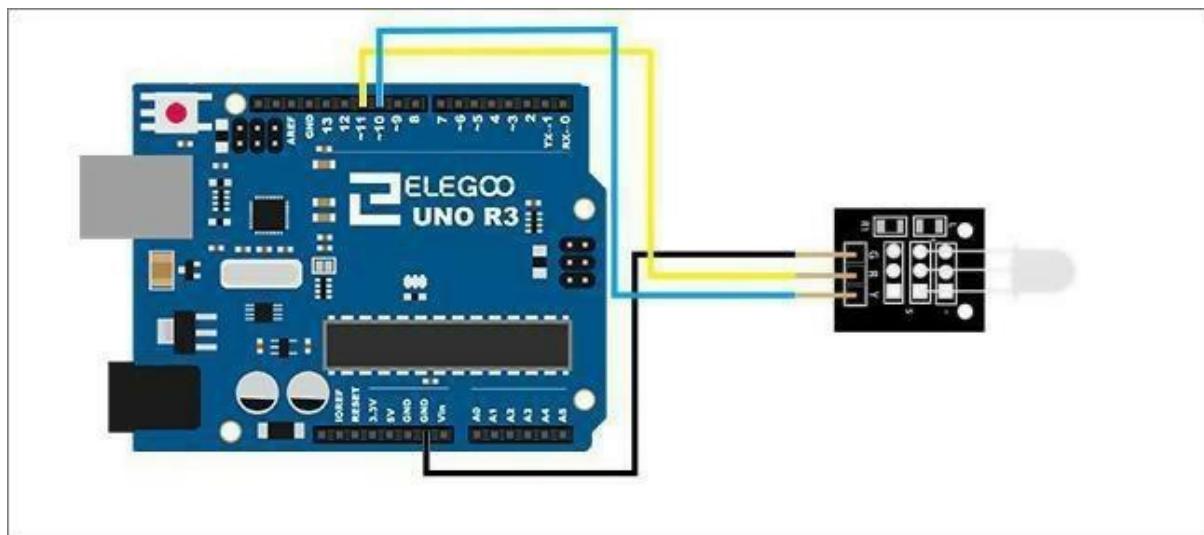


Diagrama de cableado

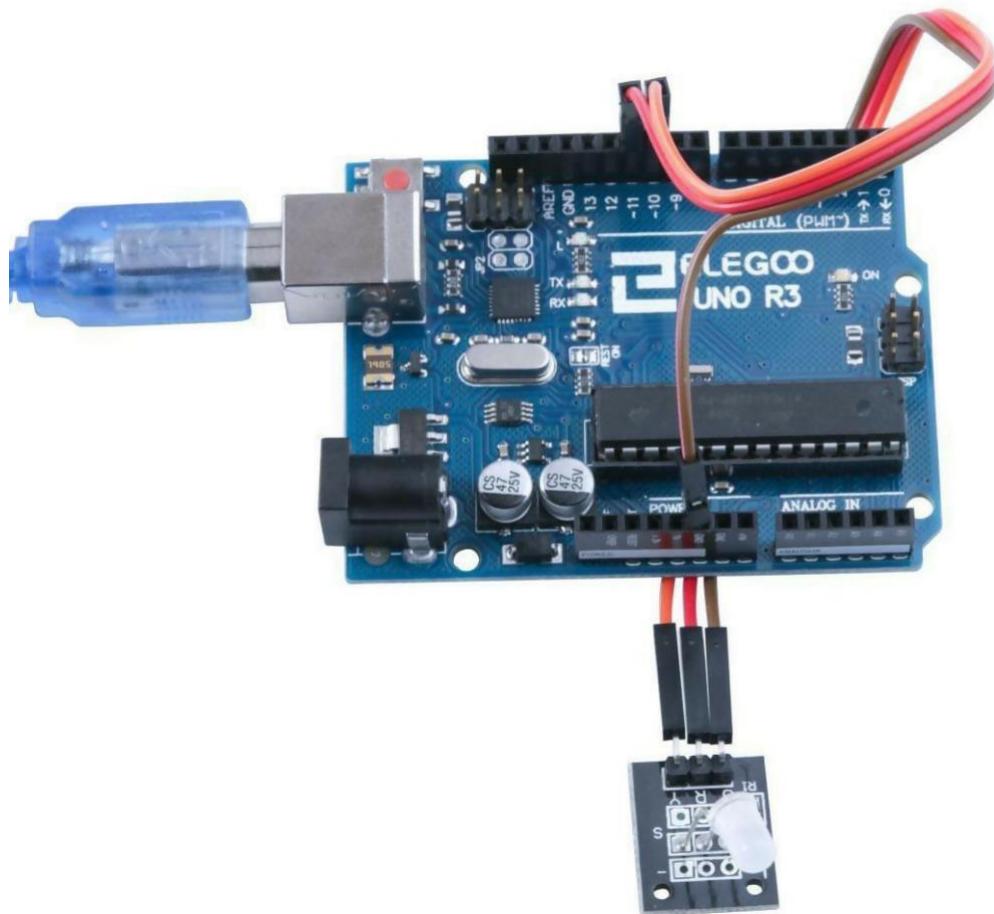


Código

Después de conectar el circuito, abrimos la carpeta "code" en el tutorial y buscamos la carpeta "Lección 14 Módulo LED de Dos Colores" para abrir el programa.

Podrás ver que el módulo cambia de color de acuerdo a lo establecido en el código. Si quieres que el color se comporte de otra forma, puedes modificar el código.

Imagen ejemplo



A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
/* Selecciona el pin para el LED rojo */
int redpin = 11;
/* Selecciona el pin para el LED azul */

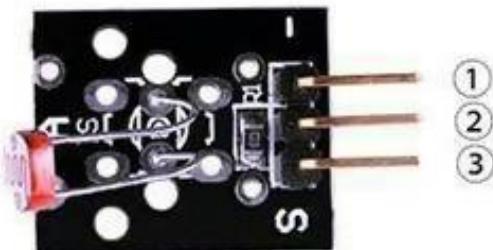
int yellowpin =10;
/*Define una variable integral val*/ int val;
void setup()
{
/*Define el pin rojo como una salida*/ pinMode(redpin, OUTPUT);
/*Se define el pin amarillo como una salida*/ pinMode(yellowpin, OUTPUT);
/*Configura la velocidad de transmisión en 9600*/ Serial.begin(9600);
}
void loop()
{
for (val = 255;val > 0;      val--)
{ analogWrite(11, val);
/*Los colores del LED cambiarán de acuerdo a valores numéricos de la variable val*/
analogWrite(10, 255 - val);
delay(15);
}
for (val = 0; val < 255;      val++)
{ analogWrite(11, val);
/*Los colores del LED cambiarán de acuerdo a valores numéricos de la variable val*/
analogWrite(10, 255 - val);
delay(15);
}
Serial.println(val, DEC);
}
```

Lección 15 Módulo de Resistencia Variable por Luz

En este experimento, aprenderemos a utilizar la resistencia variable por luz, también conocida como módulo foto resistor.

Los resistores variables por luz son muy comunes en nuestra vida diaria. Son usados como interruptores inteligentes para traer conveniencia a nuestra vida. Del mismo modo, en nuestra vida diaria, los usamos en diseño electrónico. Así que para usarlos de la mejor forma, proveemos los módulos correspondientes para ayudarle a aplicarlos de forma conveniente y eficiente.

LDR (Light Dependent Resistor). Resistencia en la oscuridad >20M Ohm, Luz <80 Ohm. Los dos pines exteriores se conectan al LDR. Un resistor fijo de 10 K ohm se conecta entre el pin del medio y el 'S'; viene incluido con el módulo. Esto simplifica la construcción de un circuito puente de medición.



- 1.GND:ground
- 2.VCC:3.3V-5V DC
- 3.OUTPUT

Componentes Requeridos:

- 1 x Elegoo Uno R3 1 x cable USB
- 1 x Módulo foto resistor 3 x Cables F-M

Introducción de componentes

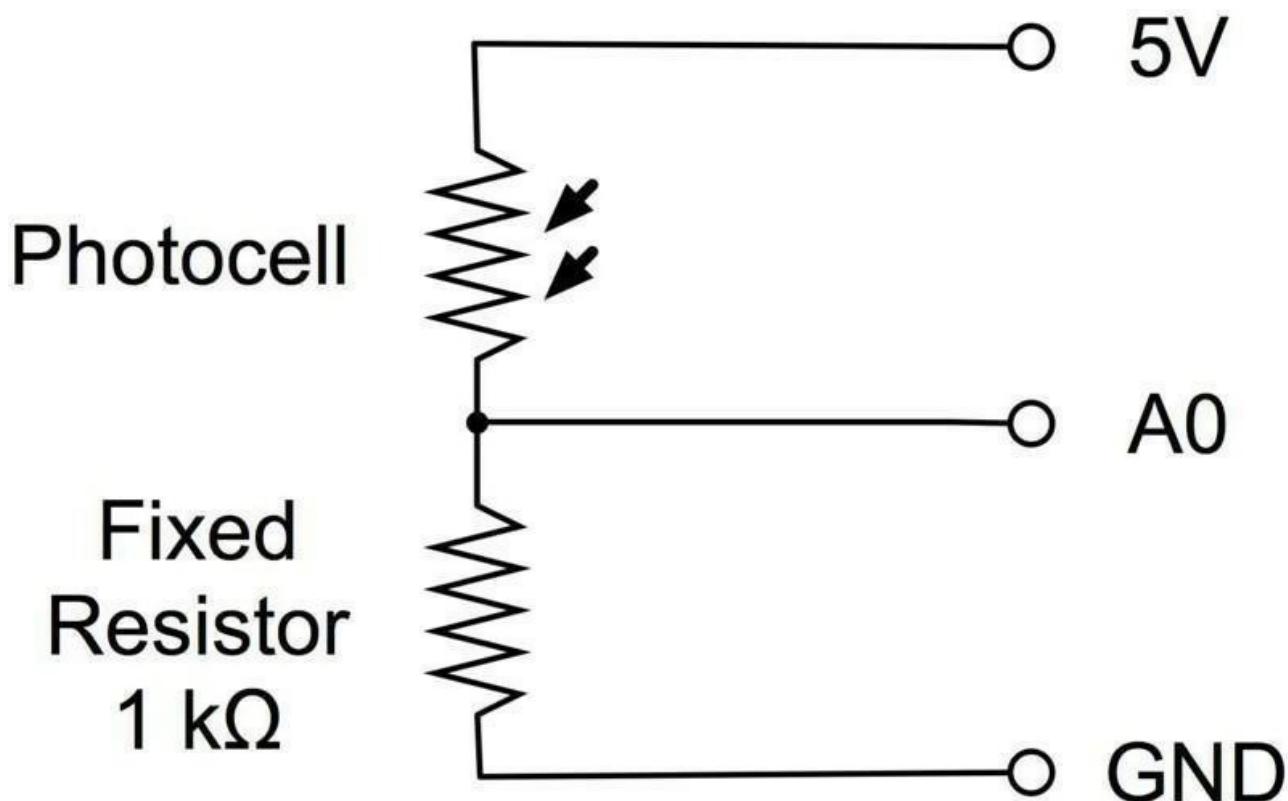
Resistencia variable por luz:

La photocelda utilizada es de un tipo denominado resistencia variable por luz, o LDR. Como sugiere el nombre, estos componentes actúan como un resistor, excepto por el hecho que su resistencia varía en respuesta a la cantidad de luz que incide sobre ellos.

Este tiene una resistencia de aproximadamente $50\text{ k}\Omega$ en la oscuridad y $500\ \Omega$ en la luz brillante. Esta variación del valor de resistencia puede convertirse en algo que usamos para medir a través de la entrada analógica de un Arduino.

Es necesario convertir este valor a voltaje.

La manera más simple de hacerlo es combinándolo con un resistor de valor fijo.



El resistor y la fotocelda funcionan como un medidor de nivel al estar juntos. Cuando la luz es muy brillante, la resistencia de la fotocelda es muy pequeña comparada con la del resistor de valor fijo, por lo que actúa como si el medidor estuviese marcando cerca del máximo.

Cuando la fotocelda está en la oscuridad, su resistencia se hace mayor que la del resistor fijo de 1 K ohm, por lo que el sistema actúa como si el medidor está en un valor bajo, cercano a GND.

Carga el sketch que se muestra en la siguiente sección y cubre la fotocelda con un dedo, manteniéndola cerca de una fuente de luz.

Esquema de conexión

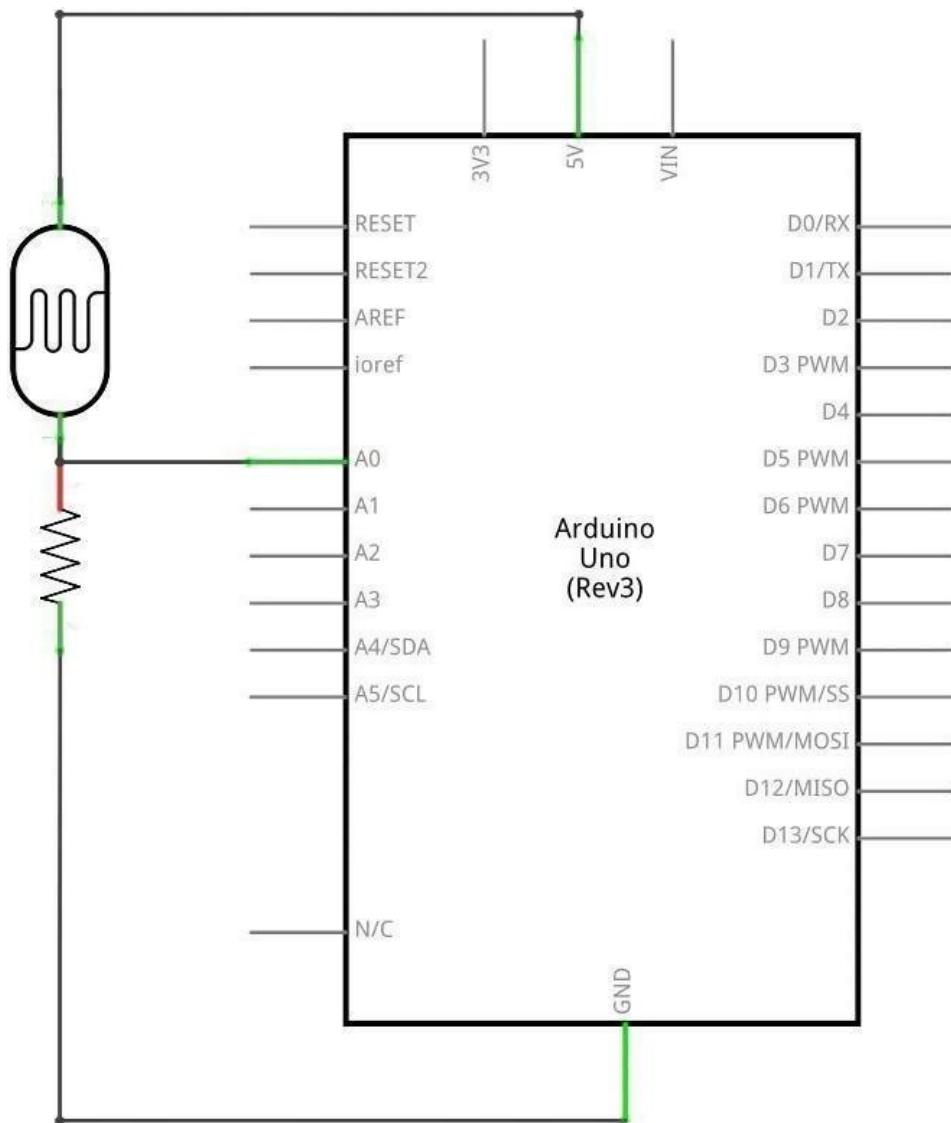
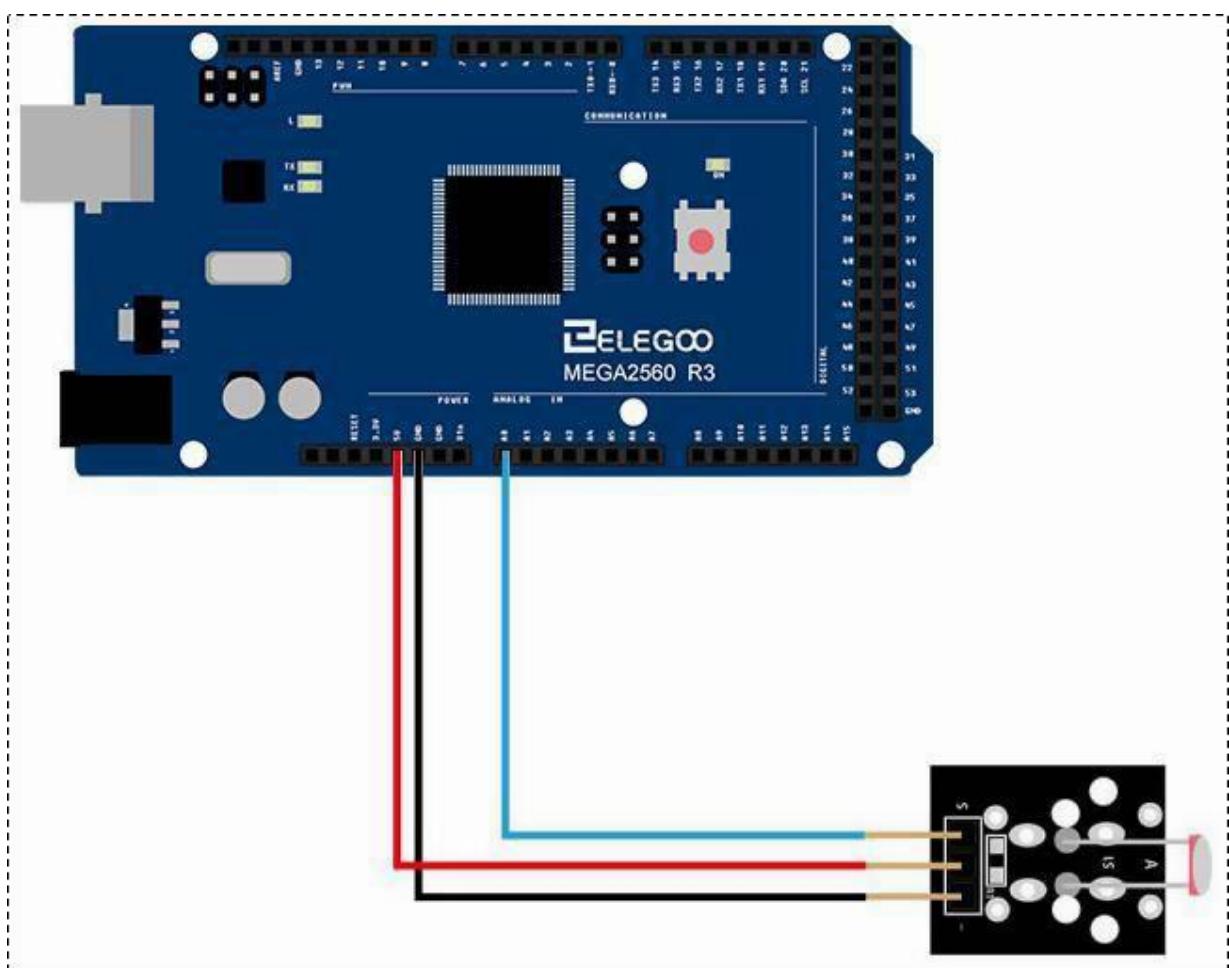
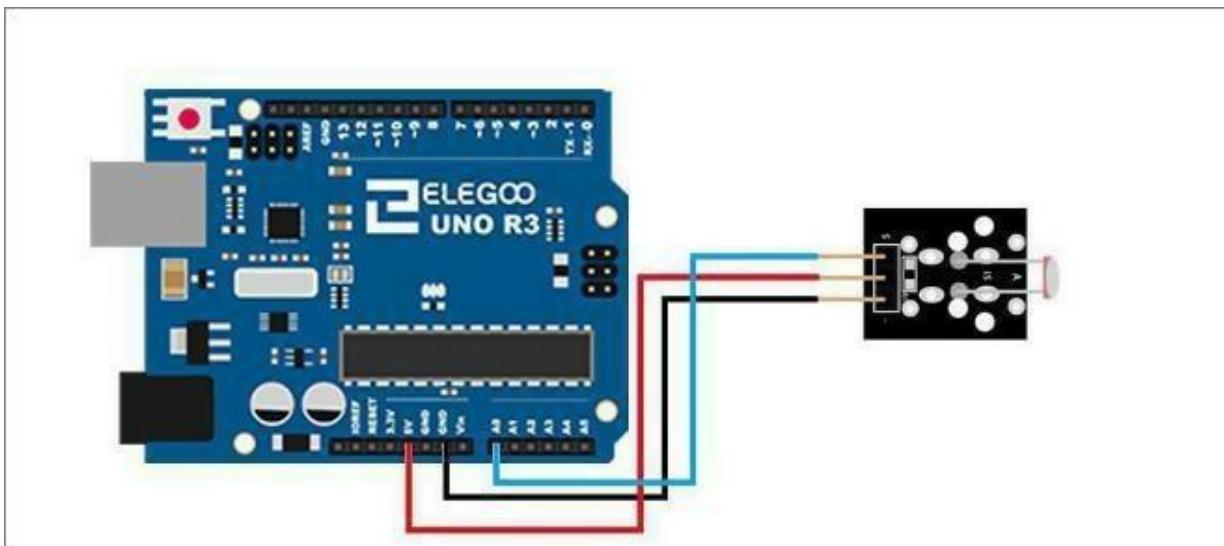
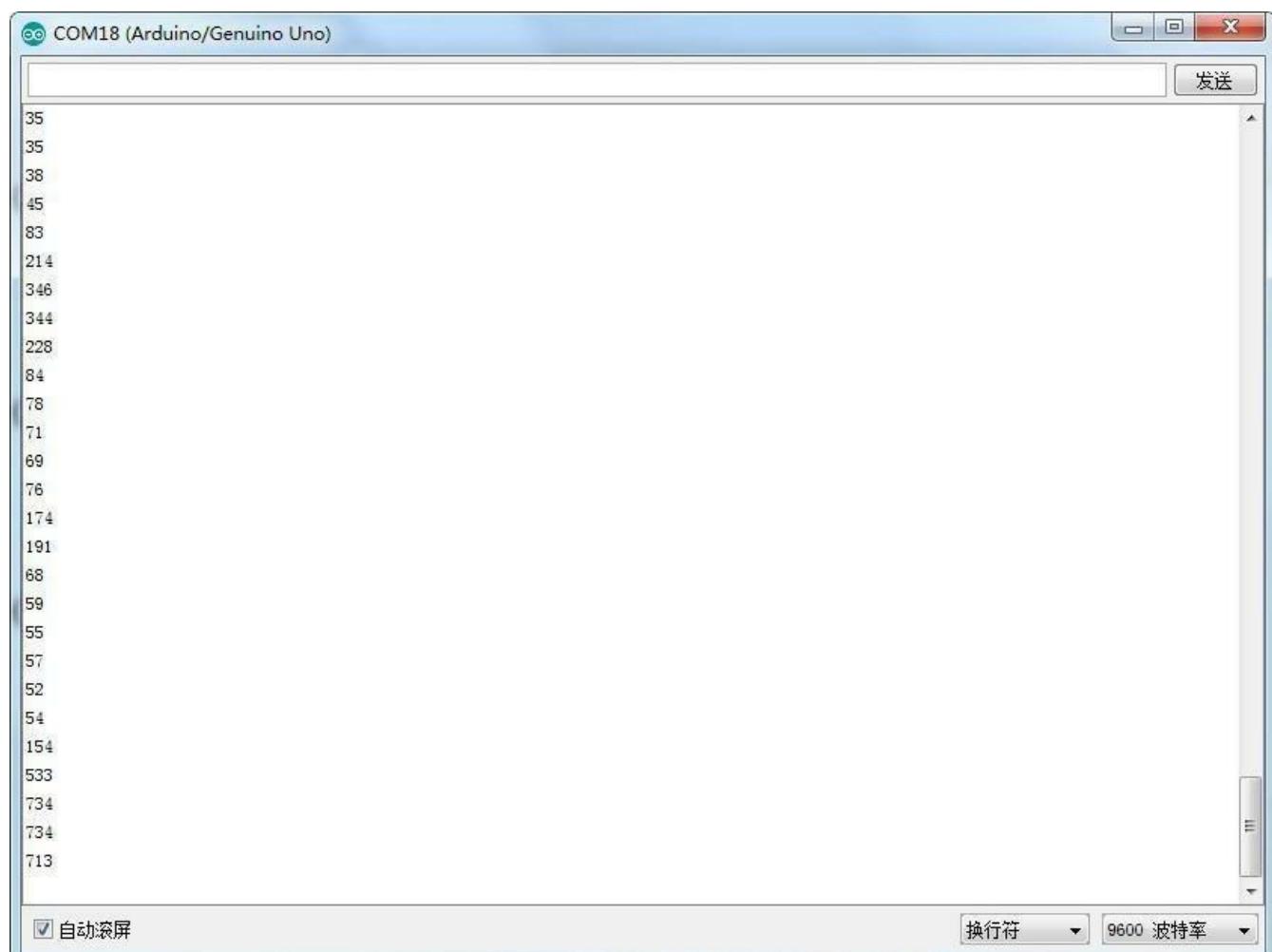


Diagrama de cableado



Código

Cuando termines de hacer el cableado, abre el tutorial y ve a code > Lección 15 Módulo foto resistor y corre el programa (el archivo .ino). Recuerda que en la lección 2 se explica como abrir el monitor serial y encontrar los siguientes datos en él.

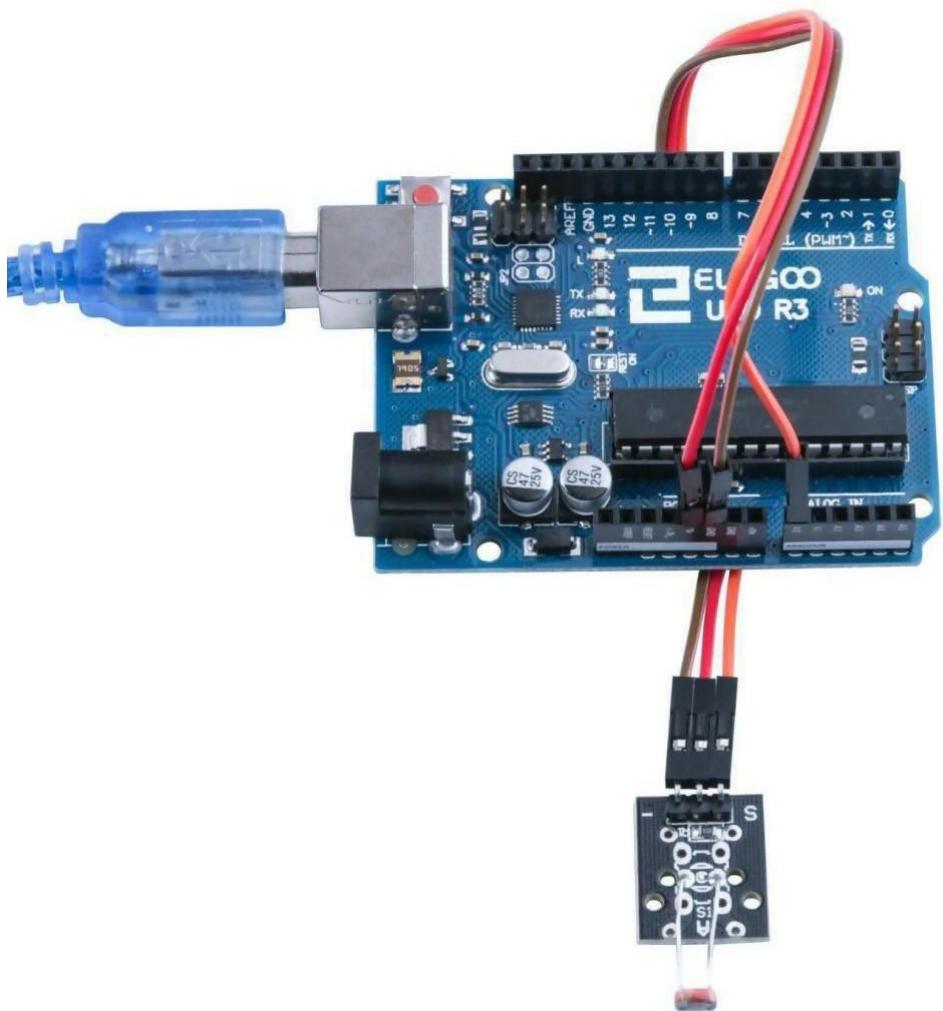


The screenshot shows the Arduino Serial Monitor window titled "COM18 (Arduino/Genuino Uno)". The window displays a list of numerical values, likely sensor readings, in the following sequence:
35
35
38
45
83
214
346
344
228
84
78
71
69
76
174
191
68
59
55
57
52
54
154
533
734
734
713

At the bottom of the window, there are two status indicators: a checked checkbox labeled "自动滚屏" (Auto Scroll) and a dropdown menu showing "换行符" (Line Break) and "9600 波特率" (9600 Baud Rate).

En la prueba, solo leemos el valor de voltaje de la salida analógica del módulo foto resistor. En la prueba Resulta, encontraremos que si hay iluminación se emitirá un voltaje alto equivalente al estado de encendido. En la ausencia de luz, el voltaje será bajo, equivalente al estado de apagado. Eso puede ser usado en la práctica

Imagen ejemplo



A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
/* Selecciona el pin de entrada para el potenciómetro */ int sensorPin = A0;
/* Selecciona el pin para el LED */ int ledPin = 13;
/* Variable para almacenar el valor que indica el sensor */ int sensorValue = 0;

void setup()
{
  /*Define ledPin como una salida*/ pinMode(ledPin,OUTPUT);
  /*Configura la velocidad de transmisión en 9600*/ Serial.begin(9600);
}

void loop()
{
  /*Lee el valor sensorPin asignado a sensorValue*/ sensorValue = analogRead(sensorPin);

  digitalWrite(ledPin, HIGH);
  /*Sensor de retraso almacena el valor*/ delay(sensorValue); digitalWrite(ledPin, LOW);
  delay(sensorValue);

  Serial.println(sensorValue,
}
Serial.println(sensorValue,
}

DEC);
```

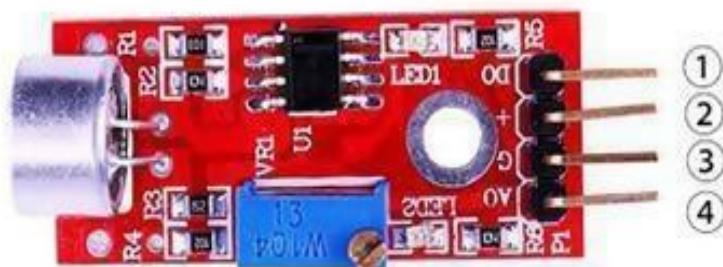
Lección 16 Módulos de Micrófono Grande y Pequeño

Resumen

En este experimento, aprenderemos a utilizar sensores de voz de alta sensibilidad.

Módulo de Micrófono Grande

Este es un módulo de micrófono que posee una cápsula electret de alta sensibilidad y formato grande. La salida 'DO' (alto activo) es conmutada cuando el nivel de sonido excede un valor predeterminado. Dicho nivel se ajusta a través de un potenciómetro.



- 1.D0:digital output
- 2.VCC: 3.3V-5V DC
- 3.GND:ground
- 4.AO:analog output

Módulo de Micrófono Pequeño

Es un módulo de micrófono con una cápsula electret pequeña. La salida 'DO' (alto activo) es conmutada cuando el nivel de sonido excede un valor predeterminado. Dicho nivel se ajusta a través de un potenciómetro. A excepción de su cápsula de tamaño más pequeño y su menor sensibilidad, este módulo es idéntico al grande.



- 1.D0:digital output
- 2.VCC: 3.3V-5V DC
- 3.GND:ground
- 4.AO:analog output

Componentes requeridos:

1 x Elegoo Uno R3 1 x cable USB

1 x Módulo de micrófono grande 1 x Módulo de micrófono pequeño 4 x cables F-M

Introducción de componentes

Sensor de sonido:

El módulo de sensor de sonido proporciona una manera sencilla de detectar sonido y se usa generalmente para detectar la intensidad del sonido. Este módulo puede utilizarse para aplicaciones de seguridad, conmutación y monitoreo. Puede ajustarse su precisión de forma sencilla para usarlo como convenga. Utiliza un micrófono que entrega una señal de entrada para el amplificador, detector de picos y memoria intermedia. Cuando el sensor detecta un sonido, produce una señal de voltaje de salida la cual es enviada a un microcontrolador que realiza cierto procesamiento de la misma.

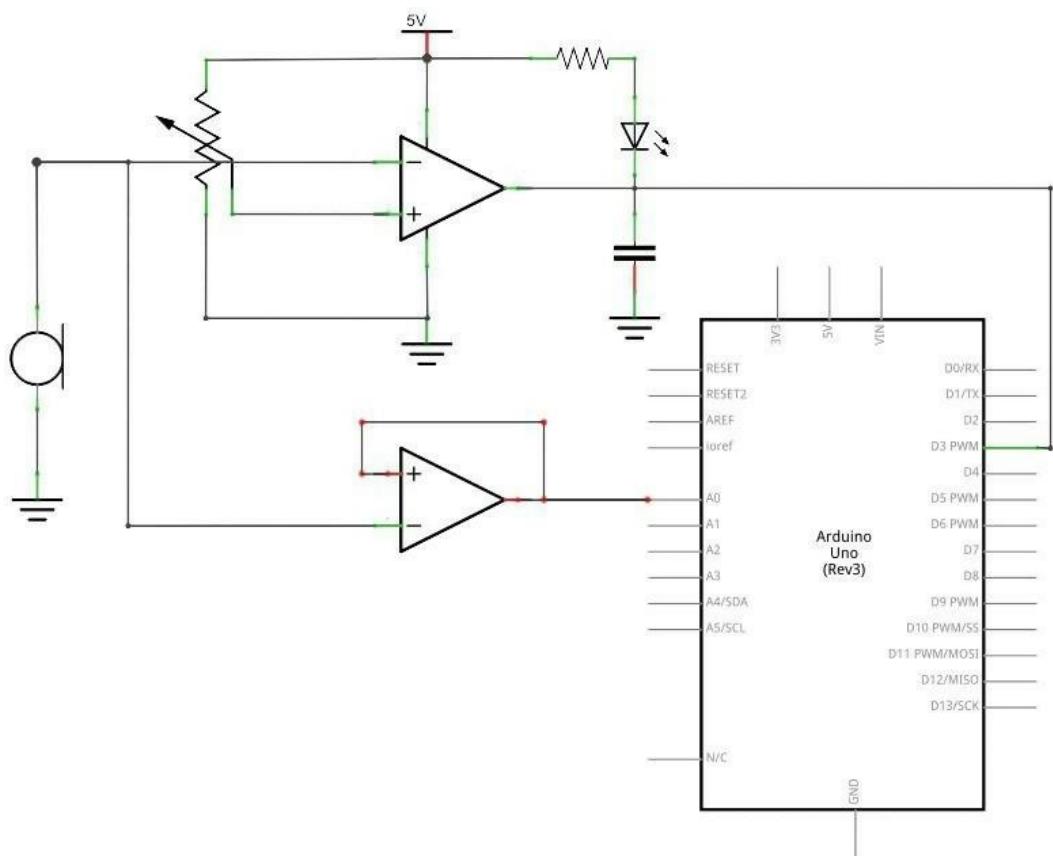


Estos micrófonos se usan ampliamente en circuitos electrónicos para detectar sonidos menores o vibraciones de aire para convertirlas en señales eléctricas a fin de usar dichas señales posteriormente. Las dos patas que se muestran en la imagen de arriba se utilizan para hacer las conexiones eléctricas con el circuito.



Un cuerpo sólido de metal conductor encapsula las varias partes del micrófono. La parte de arriba está cubierta de un material poroso adherido con pegamento. Actúa como un filtro para las partículas de polvo. Las señales de sonido o vibraciones del aire pasan a través del material poroso y caen en el diafragma a través del agujero que se muestra en la imagen de arriba

Esquema de Conexión



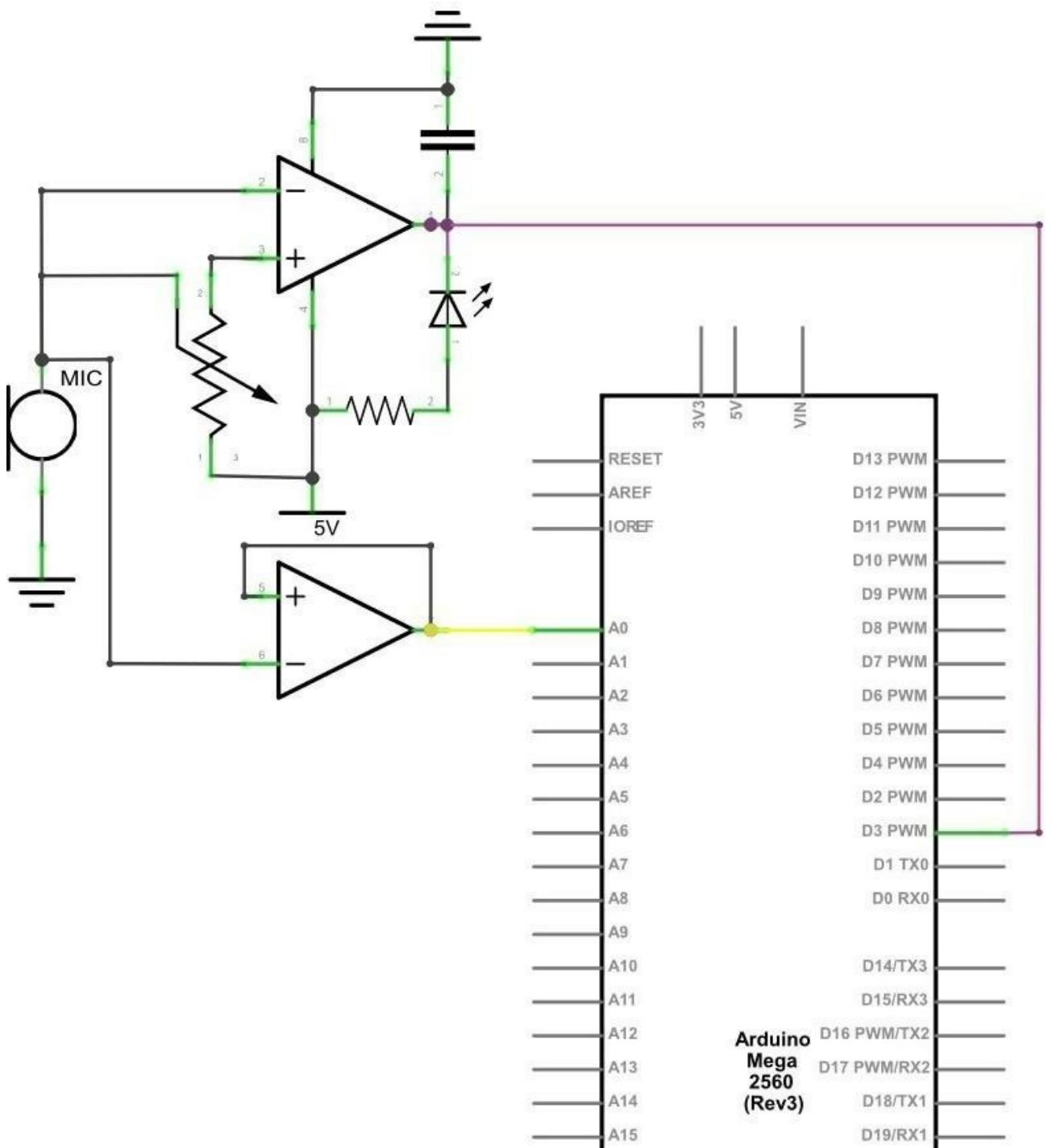
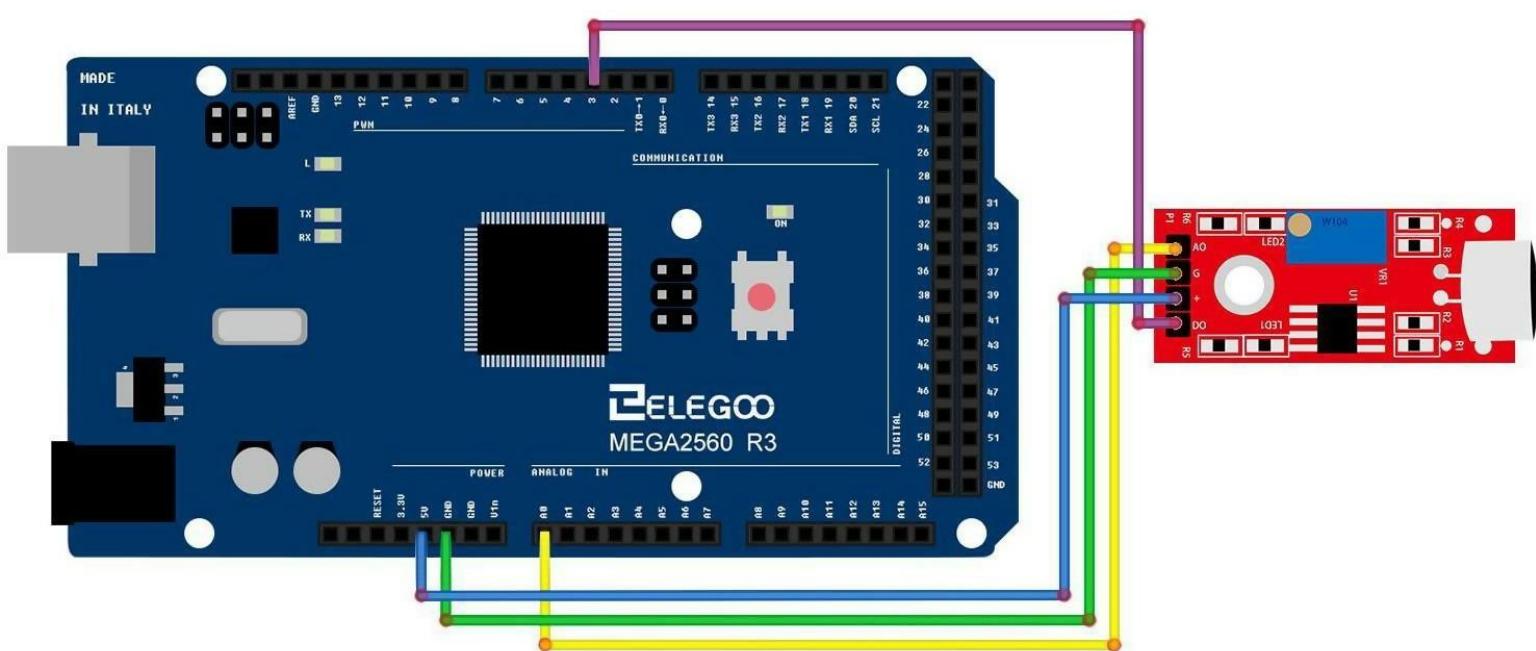
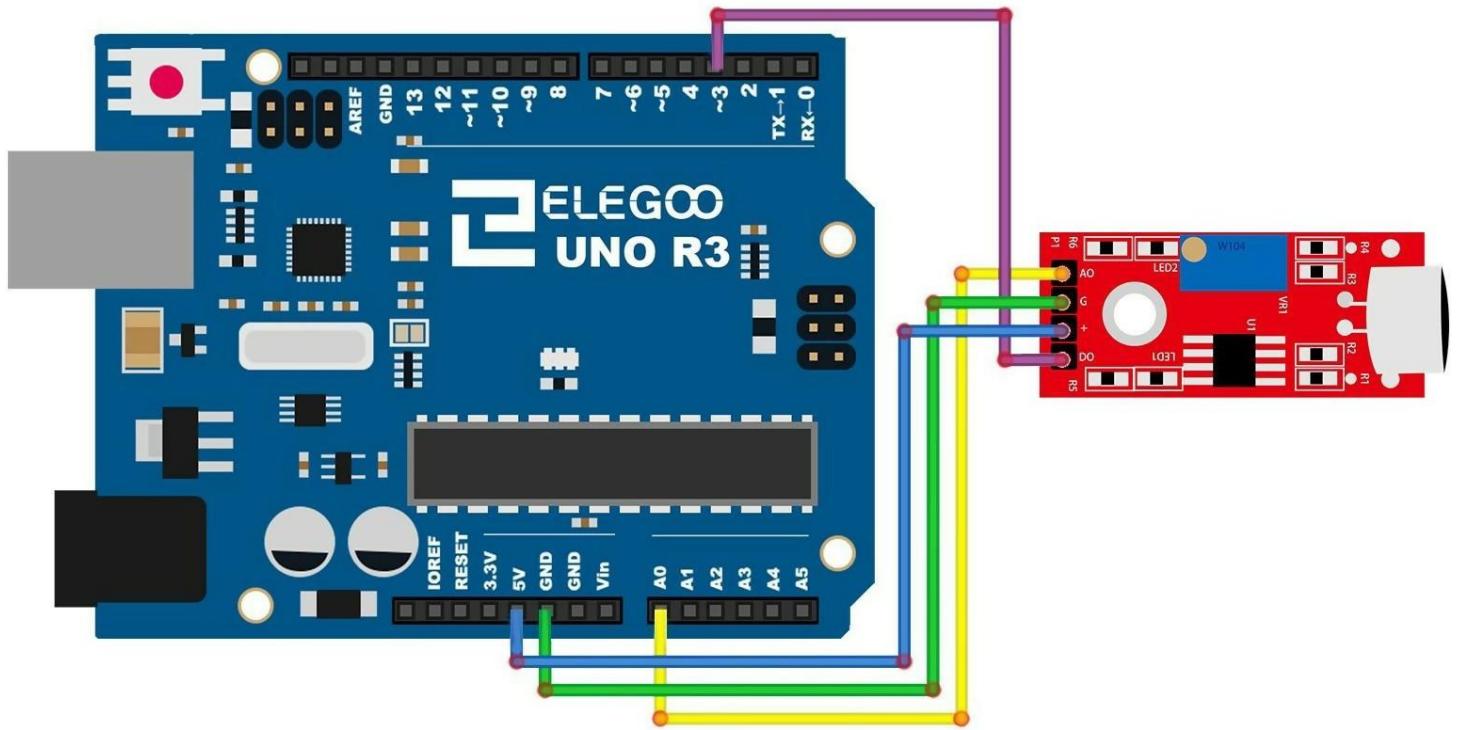


Diagrama de cableado



Código

Después de realizar el cableado, por favor abre el programa en la carpeta de código (Lección 17 Módulos de sensor de sonido grande y pequeño) y haz clic en UPLOAD para cargar el programa.

Si hay algún error, consulta los detalles en la lección 2 para cargar el programa. El sensor de voz de alta sensibilidad tiene dos salidas:

AO: salida analógica, es la salida en voltaje en tiempo real del micrófono.

DO: salida digital cuando la intensidad del sonido alcanza cierto umbral. Puede ser ajustada utilizando el potenciómetro.

Por favor fíjate en que necesitas mover el tornillo con un destornillador en sentido contrario a las manecillas del reloj hasta que el LED 2 se apague, para así ajustar el potenciómetro de 10K usando el mismo destornillador.

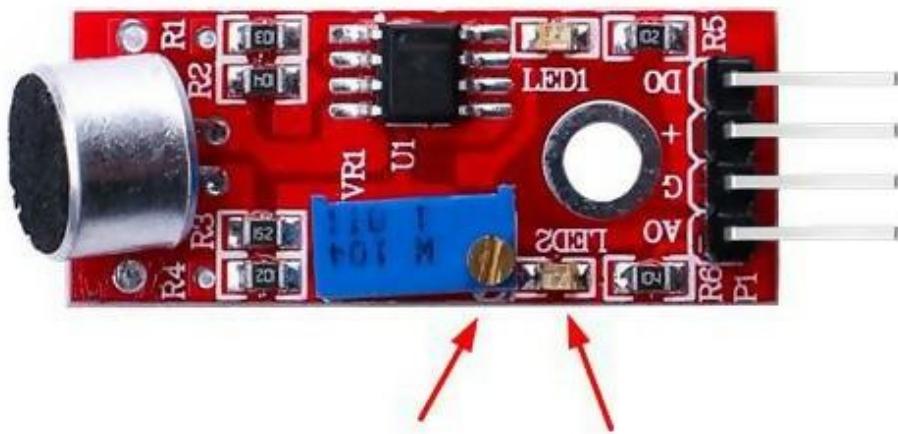
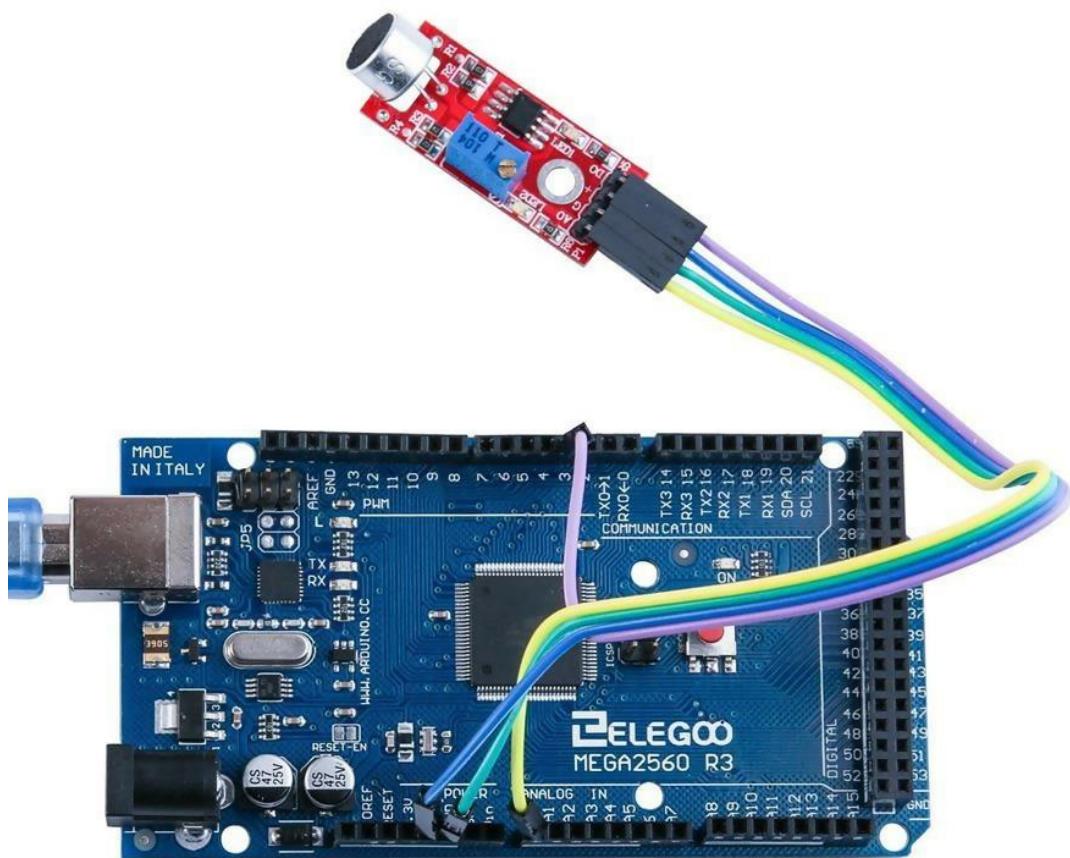
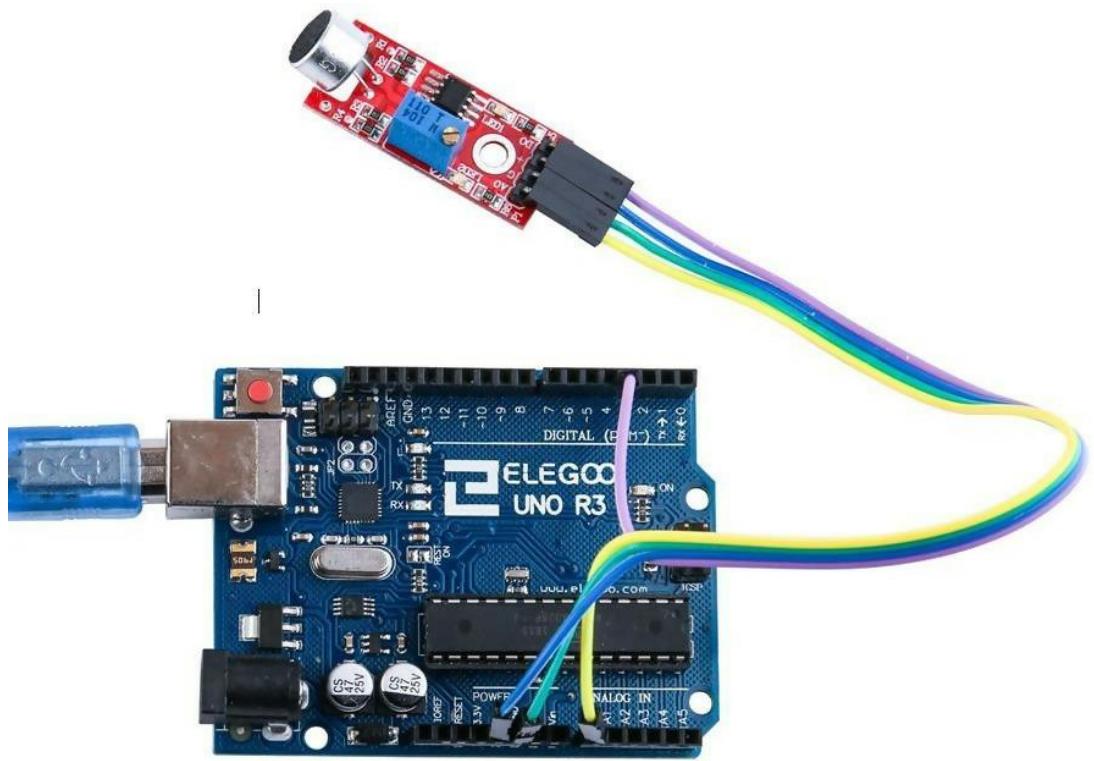
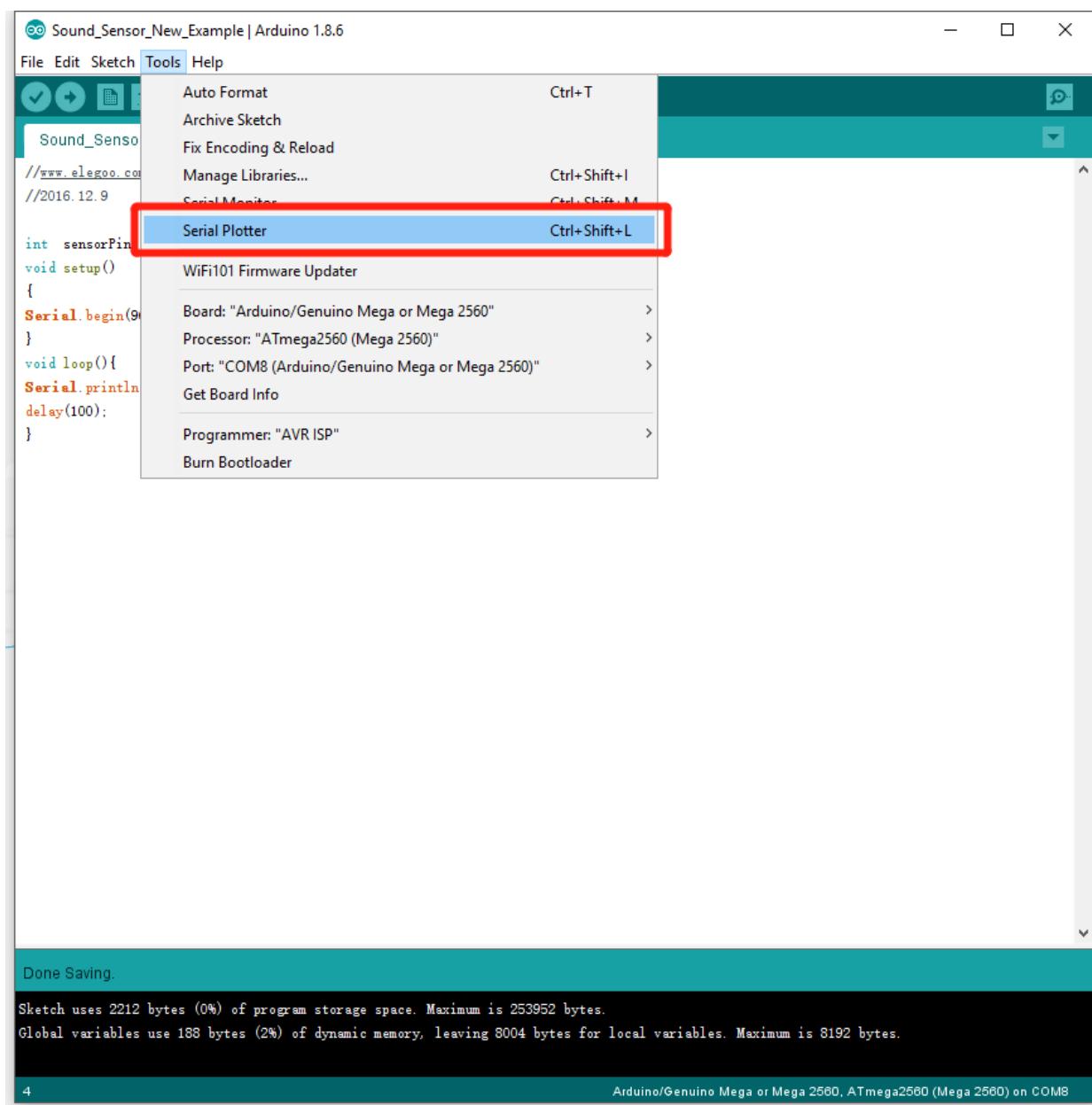


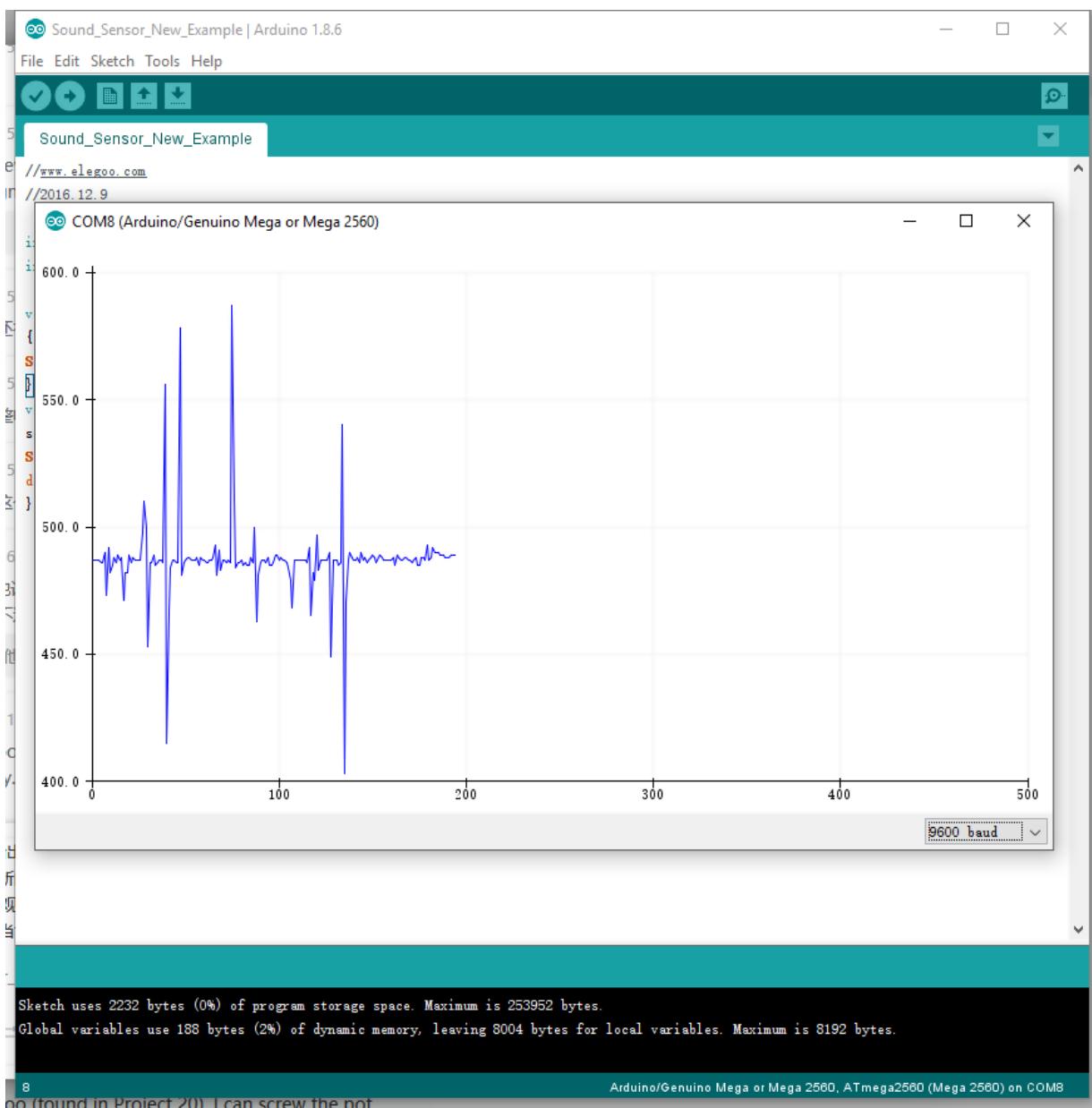
Imagen ejemplo



Graficador serial abierto:



Ejemplo del graficador serial:



Cuando hablas por el micrófono o respiras cerca del mismo, puedes observar como las formas de onda cambian.

A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
void setup()
{
    Serial.begin(9600); speed

    // Los ajustes del IDE para el Monitor Serial / Graficador (el preferido) deben ser parecidos a
    estos

    pinMode(sensorDigitalPin,INPUT); // Define el pin 7 como un puerto de entrada, para aceptar
    entradas digitales pinMode(Led13,OUTPUT); // Define el LED13 como un puerto de salida, para
    indicar que se alcanzó el gatillo digital
}

void loop(){
    analogValue = analogRead(sensorAnalogPin);
    // Lee el valor de la interfaz analógica A0 asignada a digitalValue
    digitalValue=digitalRead(sensorDigitalPin);
    //Lee el valor de la interfaz digital 7 asignada a digitalValue
    Serial.println(analogValue);
    // Envía el valor analógico a la interfaz de transmisión serial

    if(digitalValue==HIGH)
    {
        // Cuando el sensor de sonido envía la señal, ilumina el LED13 (L)

        digitalWrite(Led13,HIGH);
    }
    else
    {

```

```
digitalWrite(Led13,LOW);
}

delay(50); // Pequeña pausa para que no se cuelgue la interfaz serial
}
```

Lección 17 Módulo de Interruptor tipo Reed

Resumen

En este experimento, aprenderemos a utilizar el módulo interruptor tipo Reed y mini interruptor tipo Reed

Módulo interruptor tipo Reed

Este interruptor tipo Reed ofrece tanto una interfaz analógica como una digital. El pin 'G' conectado a GND, el pin '+' a 5V DC, el pin 'AO' ofrece una señal analógica y el 'DO' una salida digital. Se usa un potenciómetro como resistor pull up.



- 1.DO:digital output
- 2.VCC: 3.3V-5V DC
- 3.GND:ground
- 4.AO:analog output

Componentes Requeridos:

1x Elegoo Uno R3 1x cable USB

1x Módulo de interruptor tipo Reed 4 x cables F-M

Component Introduction

Reed Switch and Reed Sensor Activation:

Aunque un interruptor tipo Reed puede ser activado al colocarlo dentro de una bobina eléctrica, muchos interruptores tipo Reed y sensores Reed se usan para medir proximidad y son activados por un magneto. En lo que el magneto se acerca a las proximidades del sensor / interruptor tipo Reed, se activa el dispositivo. Cuando el magneto se aleja del interruptor tipo Reed, este se desactiva. Sin embargo, la interacción magnética involucrada en activar los contactos del interruptor tipo Reed no es obvia necesariamente. Una manera de pensar acerca de dicha interacción es que el magneto induce polos magnéticos en las partes metálicas del interruptor tipo Reed, lo que resulta en una atracción entre los contactos eléctricos y cause que se active el interruptor.

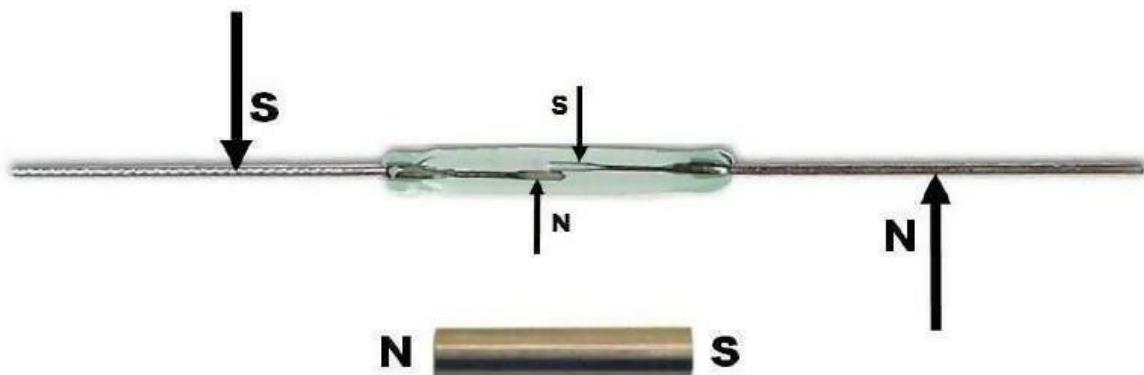


Figure 1 – Magnetic Induction

Otra forma igualmente válida de pensar acerca de la interacción entre un magneto y un interruptor tipo Reed es que el magneto induce un flujo magnético a través de los contactos eléctricos. Cuando el flujo magnético es lo suficientemente grande, la atracción magnética entre los contactos causa que se cierre el interruptor tipo Reed.



Figure 2 – Magnetic Flux

Los siguientes son ejemplos de las distancias típicas de activación de los sensores e interruptores tipo Reed.

Diferencias entre el módulo de interruptor tipo Reed y el módulo de mini interruptor tipo Reed

Lo primero que puede notarse es que uno es más grande que el otro. El más grande puede albergar más funciones que el mini. La salida del interruptor tipo Reed puede ser digital y analógica. El mini Reed solo posee salida digital.

En el kit de 37 sensores, hay 7 módulos con pcb rojos. La diferencia entre el pcb rojo y el pequeño es la misma mencionada arriba.

Principio

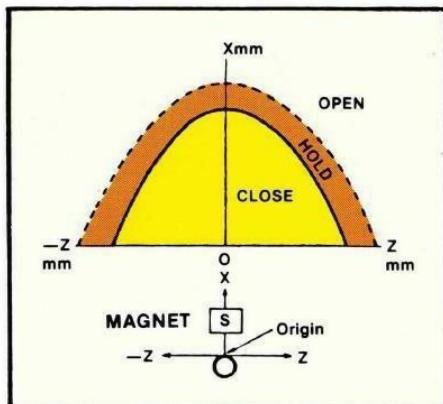


Figure 3 – Magnet Parallel to Reed Sw.

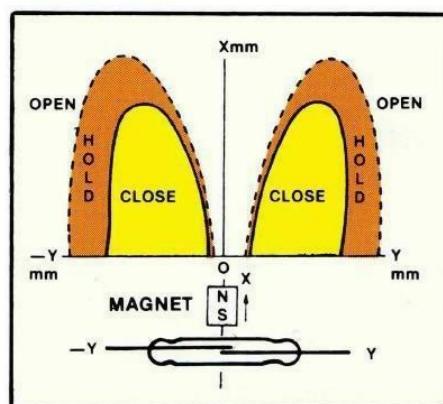


Figure 4 – Magnet Perpendicular to Reed Sw.

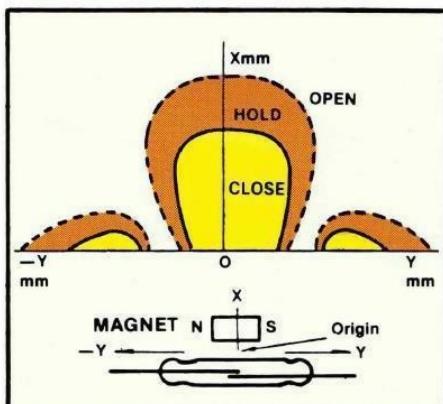


Figure 5 – Magnet Parallel to Reed Sw.

Como puede observarse, la orientación magnética y la ubicación relativa con respecto al interruptor tipo Reed juegan un papel importante en la distancia de activación. Adicionalmente, el tamaño de las regiones de activación (lóbulos) variará dependiendo de la fuerza del magneto y la sensibilidad del interruptor tipo Reed. La orientación apropiada del magneto respecto al sensor Reed es una consideración importante a tener en cuenta para cumplir los requerimientos de la aplicación en cuanto a los rangos de tolerancia para los sistemas mecánicos, la fuerza magnética y sensibilidad del sensor Reed o interruptor tipo Reed.

Esquema de conexión del módulo de interruptor tipo Reed

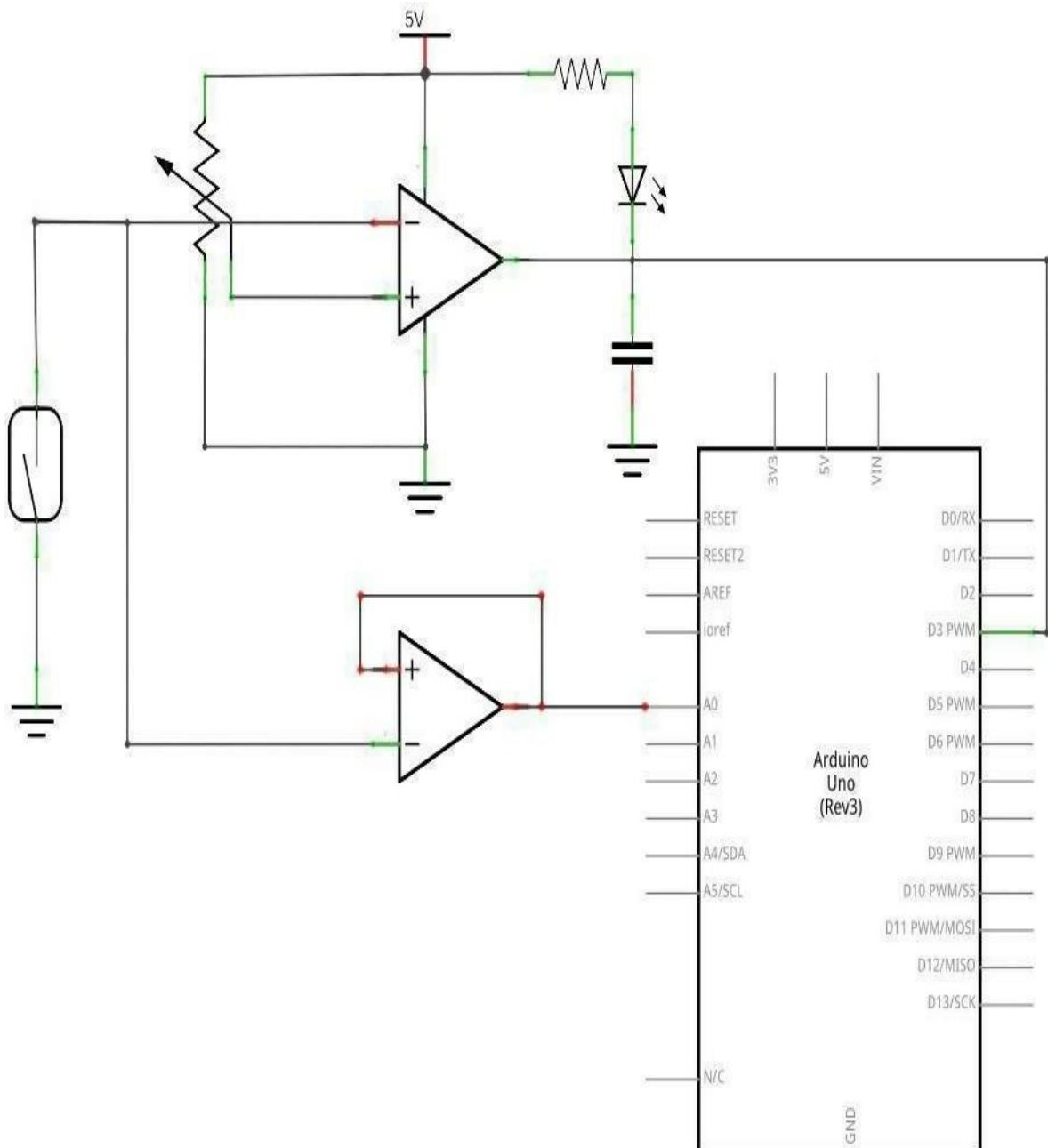
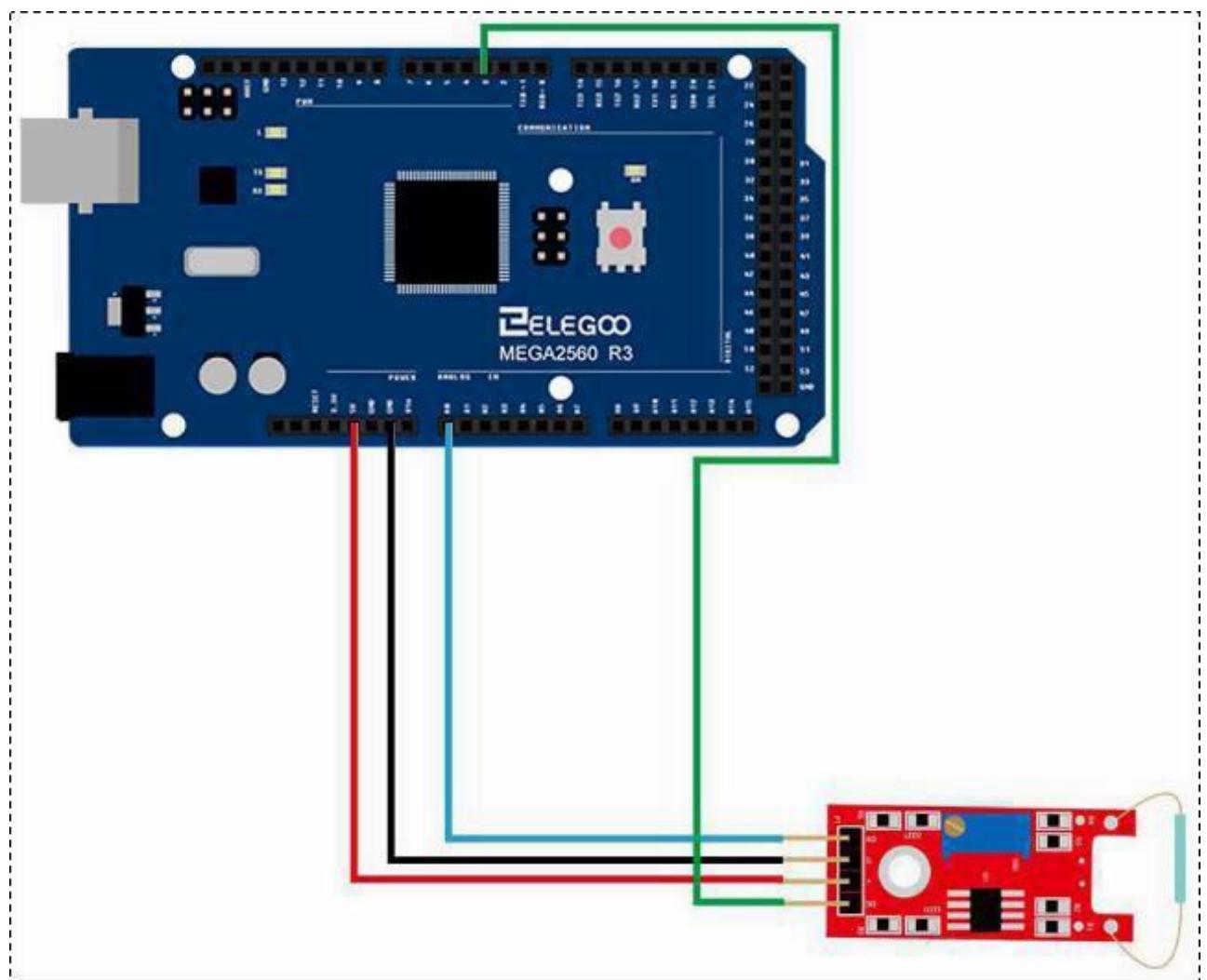
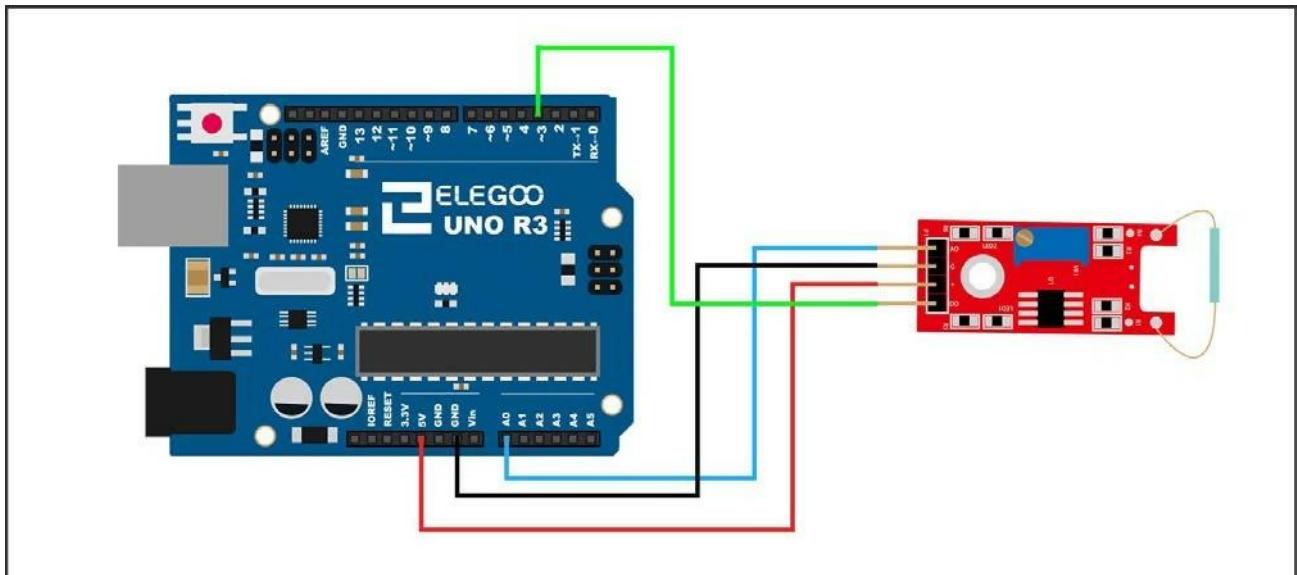


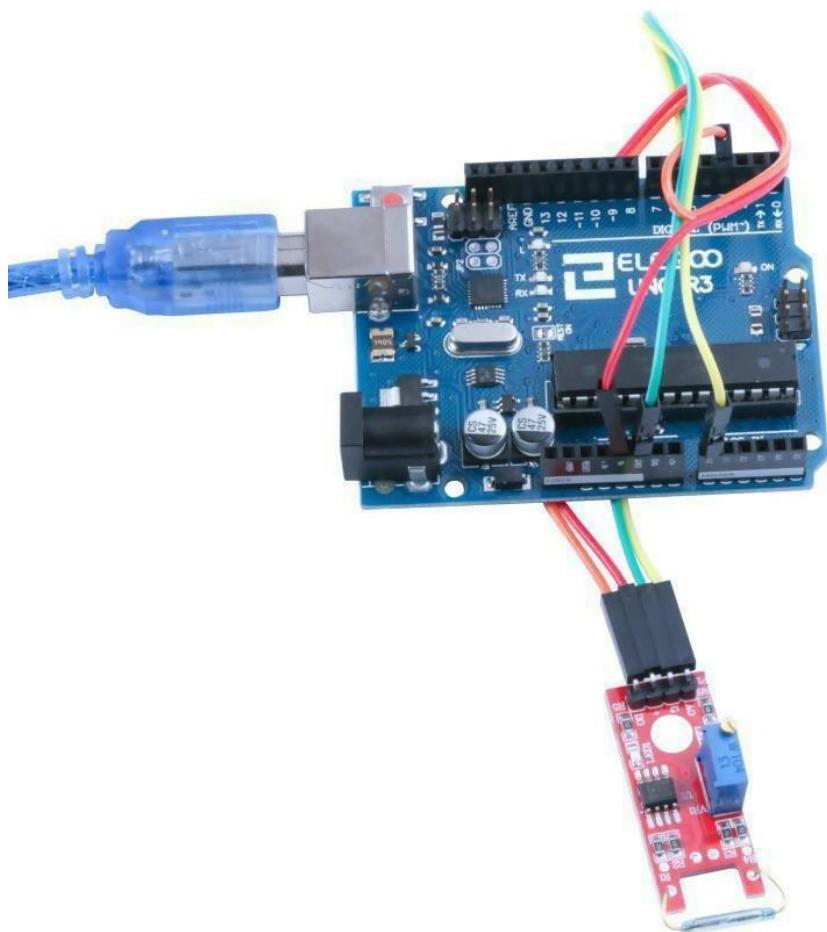
Diagrama de cableado

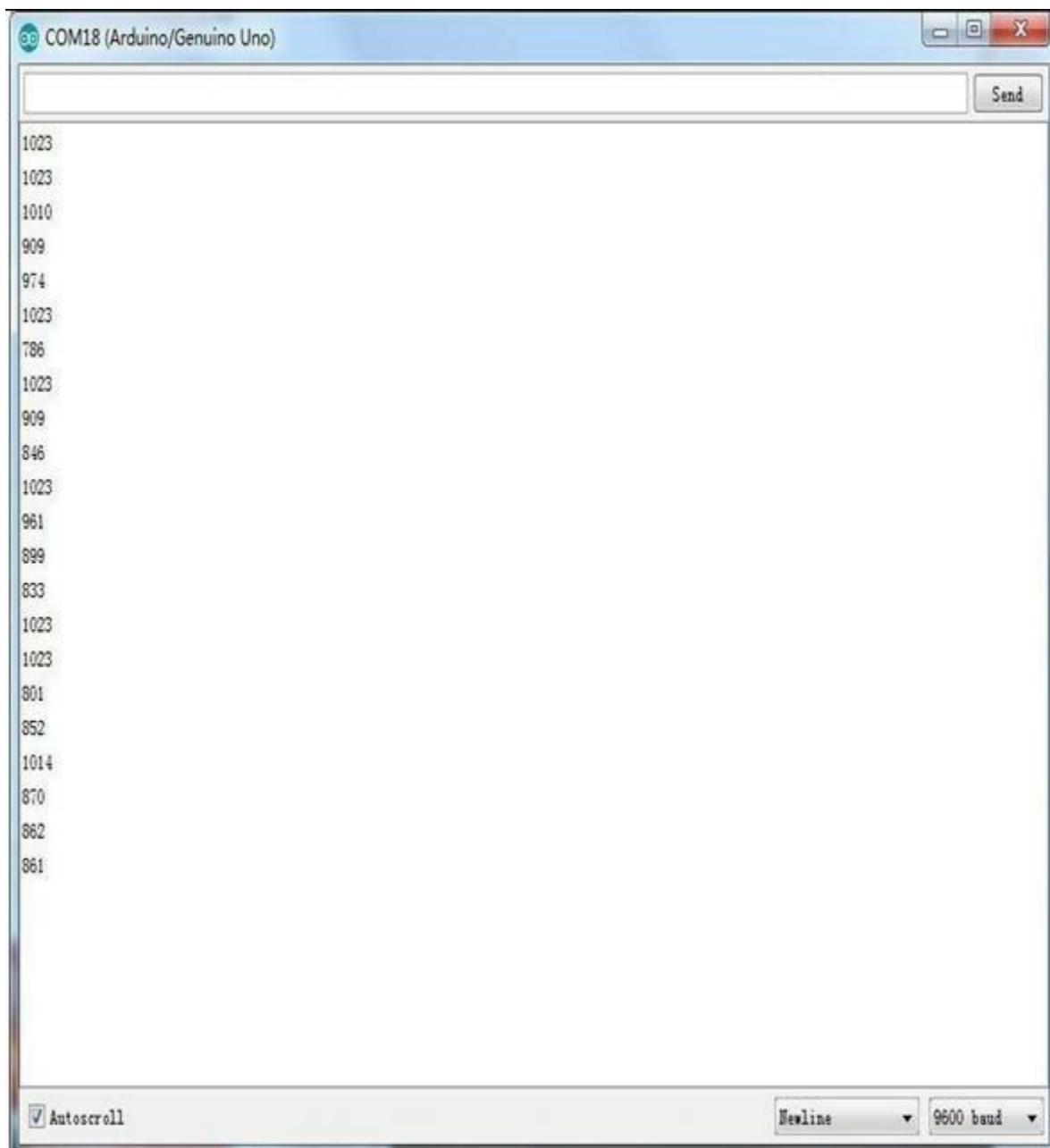


Código

Después de realizar el cableado, por favor abre el programa en la carpeta de código "Lección 17 Módulo de Interruptor tipo Reed y Mini Reed" y corre el compilador. Cuando el sensor detecte magnetismo, el módulo enviará en su salida la data que refleja la fuerza de dicho magnetismo. Este valor estará entre 0 y 1023. Verás que el LED comenzará a destellar. Abre el monitor y te mostrará datos parecidos a los siguientes:

Imagen ejemplo





A continuación se muestra el código usado en este experimento y su correspondiente explicación:

(1) Simulación del programa de control de puertos

```
// Selecciona el pin de entrada para el potenciómetro int sensorPin = A0;  
// Selecciona el pin para el LED int ledPin = 13;  
// Variable para almacenar el valor que viene del sensor int sensorValue = 0;  
  
void setup()  
{ pinMode(ledPin,OUTPUT); Serial.begin(9600);  
}  
void loop()  
{  
sensorValue = analogRead(sensorPin); digitalWrite(ledPin, HIGH); delay(sensorValue);  
digitalWrite(ledPin, LOW); delay(sensorValue); Serial.println(sensorValue, DEC);  
}
```

(2) Programa de control del puerto digital

```
/* define el puerto del LED */ int Led=13;  
/* define el puerto del interruptor */  
  
int buttonpin=3;  
/* define la variable digital val */ int val;  
void setup()  
{  
/* define el LED como una salida */ pinMode(Led,OUTPUT);  
/* define el interruptor como una salida */ pinMode(buttonpin,INPUT);  
}  
void loop()
```

```
{  
/* Lee el valor de la interfaz digital 3 asignada a val */ val=digitalRead(buttonpin);  
/* cuando el sensor o interruptor tenga señal, el LED destellará */ if(val==HIGH)  
{  
digitalWrite(Led,HIGH);  
}  
else  
{  
digitalWrite(Led,LOW);  
}  
}
```

Lección 18 Módulo de Temperatura Digital

Resumen

En este experimento, aprenderemos como utilizar el módulo digital de temperatura y el módulo analógico de temperatura.

Módulo Digital de Temperatura

Módulo sensor de temperatura a base de un termistor NTC. La señal de salida en 'DO' pasa a nivel alto cuando se alcanza el valor predeterminado de temperatura (ajustable). En el pin 'AO' se encuentra disponible una señal analógica de salida.



- 1.DO:digital output
- 2.VCC: 3.3V-5V DC
- 3.GND:ground
- 4.AO:analog output

Componentes Requeridos:

1x Elegoo Uno R3 1x cable USB

1x Módulo Digital de Temperatura 4x Cables F-M

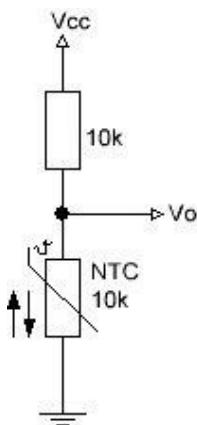
Introducción de Componente Termistor :



Qué es un termistor NTC?

Los termistores son elementos sensores de temperatura fabricados a partir de materiales semiconductores que han sido sinterizados para mostrar grandes cambios de resistencia en proporción a pequeños cambios de temperatura.

Esta resistencia se mide utilizando una corriente directa pequeña que se pasa a través del termistor para producir una caída de tensión, la cual es medida de forma apropiada.



Los termistores NTC son resistores no lineales, que alteran su resistencia de acuerdo a la temperatura. La resistencia del NTC disminuirá si la temperatura se incrementa. La manera en la cual la resistencia se disminuye tiene relación con una constante conocida en la industria electrónica como beta o β . Esta Beta se mide en $^{\circ}\text{K}$.

El Arduino posee varios puertos ADC que podemos usar para medir un voltaje, o mejor dicho, un 'ADC value'. Si se conecta el puerto analógico a Vcc, el valor máximo que se puede leer es 1023 y cuando se conecta a tierra es 0

Ahora, si hacemos un divisor de voltaje, constituido normalmente por dos resistores en serie entre Vcc y Ground, con el puerto analógico en el medio, la lectura dependerá de la relación entre los dos resistores: si son iguales, la lectura será de 512, es decir, la mitad de 1023. Si uno de los resistores, digamos el de abajo, es un NTC, las lecturas del puerto analógico variarán de acuerdo a la temperatura.

Si la temperatura baja, el valor de la resistencia se incrementa y del mismo modo la lectura del puerto analógico.

Supongamos que tenemos un resistor de 10K en serie con un NTC al cual por ahora llamamos 'R'. El voltaje a medir en el medio será

$$V_o = R / (R + 10K) * V_{cc}$$

La lectura en analogPort no dan un voltaje, sino un valor ADC; pero en base a ese valor podemos aplicar una simple regla de tres ADC value= $V_o * 1023 / V_{cc}$ // Si por ejemplo el $V_o = 4$ Volt el ADC = 818

o

$$\text{ADC value} = 1023 * (V_o / V_{cc})$$

Si ahora combinamos las dos fórmulas o aplicamos el método de substituir V_o en la fórmula para ADC obtendremos lo siguiente:

$$\text{ADC value} = (R / (R + 10K)) * V_{cc} * 1023 / V_{cc}$$

Como dividimos y multiplicamos por V_{cc} , podemos despejar ese valor de la ecuación y quedamos con

$$\text{ADC value} = (R / (R + 10k)) * 1023$$

$$\text{ADC value} = 1023 * R / (10 + R)$$

Si queremos el valor de resistencia para esa ecuación, queda en $R = 10k / (1023 / \text{ADC} - 1)$

Si eso fue muy rápido, aquí se muestra la ecuación paso a paso Me gustan más las formulas 'in line'

$$ADC = 1023 * \frac{R}{10 + R}$$

Esto se convierte en

$$\frac{1023 * R}{ADC} = 10 + R$$

Substracción de R

$$\frac{1023 * R}{ADC} - R = 10$$

Trabaja-R en la multiplicación

$$\left(\frac{1023}{ADC} - 1 \right) * R = 10$$

Como nos interesa R, divide ambas partes entre la fracción encerrada

$$R = \frac{10}{\left(\frac{1023}{ADC} - 1 \right)}$$

El '10' se usó en lugar de '10k' y como no siempre usamos 10k lo hacemos más general:

$$R = \frac{10k}{\left(\frac{1023}{ADC} - 1 \right)}$$

como no siempre usamos 10k lo hacemos más general:

$$R_{ntc} = \frac{R_{series}}{\left(\frac{1023}{ADC} - 1 \right)}$$

Así que, si sabemos el valor del resistor en serie, podremos calcular el valor del NTC a partir del valor ADC medido. Pero recuerda que esto es válido para la configuración pull up. Si es pull down, el cálculo valor ADC a resistencia es el inverso.

```
Rntc = Rseries*(1023/ADC-1); // para pull down  
Rntc = Rseries/(1023/ADC - 1)); // para configuración pull-up
```

Como se vería eso en un programa?

```
//Mide el valor del byte NTC NTCPin = A0;  
  
const int SERIESRESISTOR = 10000;  
  
void setup()  
{  
  
Serial.begin(9600);  
}  
  
void loop()  
{  
  
float ADCvalue; float Resistance;  
  
ADCvalue = analogRead(NTCPin); Serial.print("Analoge "); Serial.print(ADCvalue);  
Serial.print(" = ");  
  
//Convierte el valor a resistencia  
  
Resistance = (1023 / ADCvalue) - 1; Resistance = SERIESRESISTOR / Resistance;  
Serial.print(Resistance);  
  
Serial.println(" Ohm"); delay(1000);
```

```
}
```

```
//end program
```

Conocer la resistencia del NTC es interesante, pero no nos dice mucho acerca de la temperatura... o sí?

Bueno, la mayoría de los NTC tienen un valor nominal medido a 25 grados centígrados, por tanto si tienes un NTC de 10K y mides precisamente ese valor, significa que la temperatura es de 25 grados en ese momento. Pero eso no es mucha ayuda cuando la medida es diferente.

Como guía, podrías armar una tabla en la cual se reflejen los valores de temperatura correspondientes a cada valor de resistencia. Esas tablas son muy precisas pero requieren mucho trabajo y espacio de memoria.

Sin embargo, existe una fórmula llamada la ecuación de Steinhart-Hart, que sirve para obtener muy buenos resultados aproximados al convertir valores de una NTC a temperatura. No es tan exacta la tabla de valores de termistores (después de todo es una aproximación), pero si es bastante precisa.

La ecuación de Steinhart-Hart es algo así:

$$\frac{1}{T} = A + B \ln(R) + C (\ln(R))^3$$

Como puedes notar, es una ecuación algo complicada que requiere de varios parámetros (A, B, C) que no tenemos normalmente al comprar un NTC. Pero hay dos cosas que podemos hacer. Podríamos tomar 3 lecturas con patrón de temperatura calibrado y deducir los parámetros A, B y C a partir de allí.

$$\begin{bmatrix} 1 & \ln(R_1) & \ln(R_1)^3 \\ 1 & \ln(R_2) & \ln(R_2)^3 \\ 1 & \ln(R_3) & \ln(R_3)^3 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} \frac{1}{T_1} \\ \frac{1}{T_2} \\ \frac{1}{T_3} \end{bmatrix}$$

Afortunadamente existe una simplificación de esta fórmula, llamada la ecuación del parámetro B. La misma luce como sigue:

$$\frac{1}{T} = \frac{1}{T_0} + \frac{1}{B} \ln \left(\frac{R}{R_0} \right)$$

To es la temperatura nominal, el equivalente de 25 °C en Kelvin (= 298.15 K). B es el coeficiente del termistor (3950 es un valor común). Ro es la resistencia nominal del NTC (estimada a 25 grados). Digamos que tenemos un NTC de 10Kohm. Solo necesitaremos saber el valor de R (la resistencia medida) para obtener T (la temperatura en Kelvin) para después convertir ese valor a °C.

El programa para lograrlo sería algo como esto:

```
//-----"
```

```
Byte NTCPin = A0;
#define SERIESRESISTOR 10000
#define NOMINAL_RESISTANCE 10000
#define NOMINAL_TEMPERATURE 25
#define BCOEFFICIENT 3950
void setup(){ Serial.begin(9600);
}
void loop(){ float ADCvalue; float Resistance;
ADCvalue = analogRead(NTCPin); Serial.print("Analoge "); Serial.print(ADCvalue); Serial.print(" =
");
//Convierte el valor a resistencia Resistance = (1023 / ADCvalue) - 1;
Resistance = SERIESRESISTOR / Resistance; Serial.print(Resistance);
Serial.println(" Ohm"); float steinhart;
steinhart = Resistance / NOMINAL_RESISTANCE; // (R/Ro) steinhart = log(steinhart); // ln(R/Ro)
steinhart /= BCOEFFICIENT; // 1/B * ln(R/Ro)
steinhart += 1.0 / (NOMINAL_TEMPERATURE + 273.15); // + (1/To)
steinhart = 1.0 / steinhart; // Invert steinhart -= 273.15; // convert to C
Serial.print ("TemperatureSerial.print(steinhart)"); Serial.println(" oC");
delay(1000);
```

<http://www.elegoo.com>

}

//-----

Esquema de conexión del módulo de temperatura

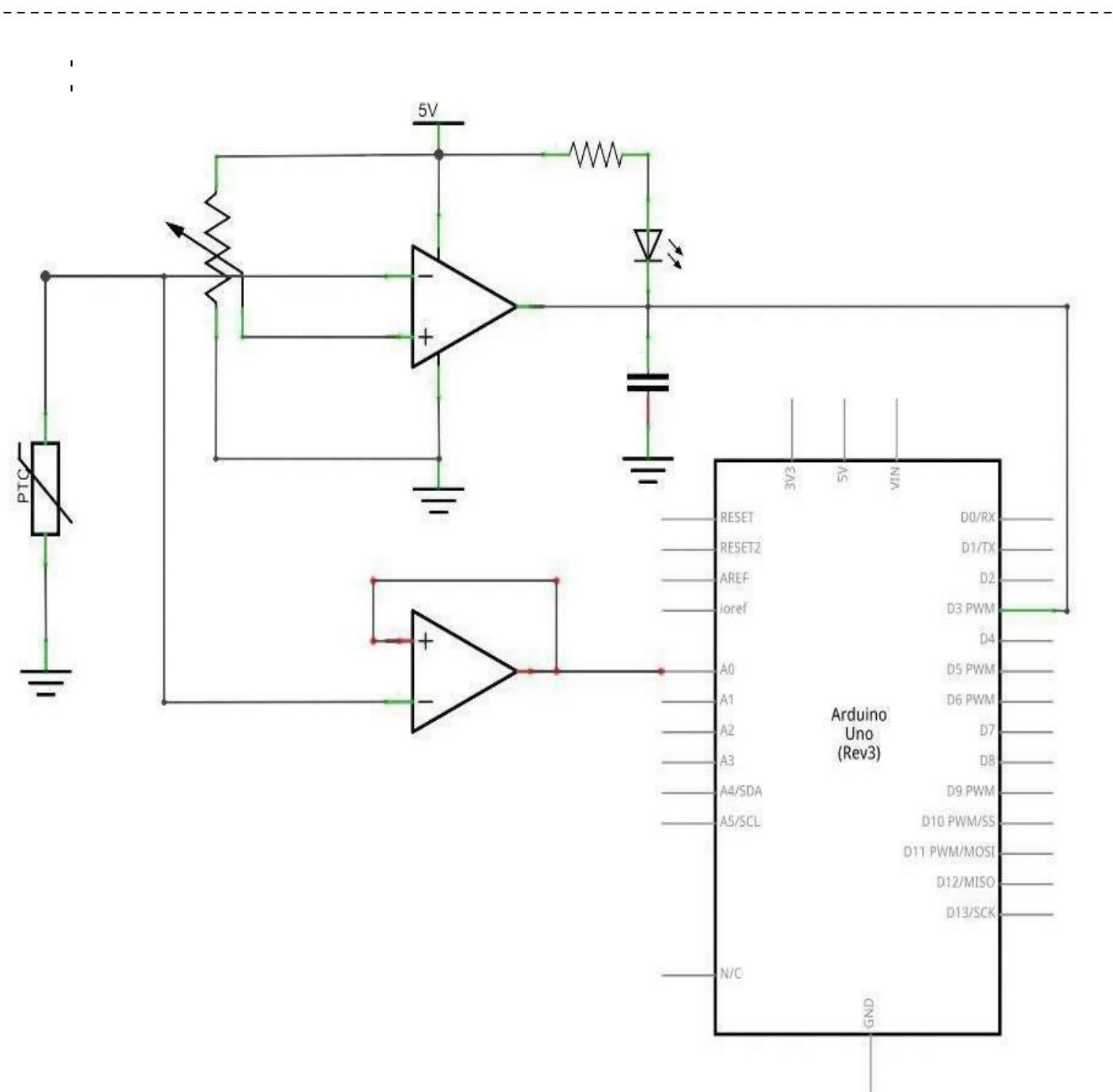
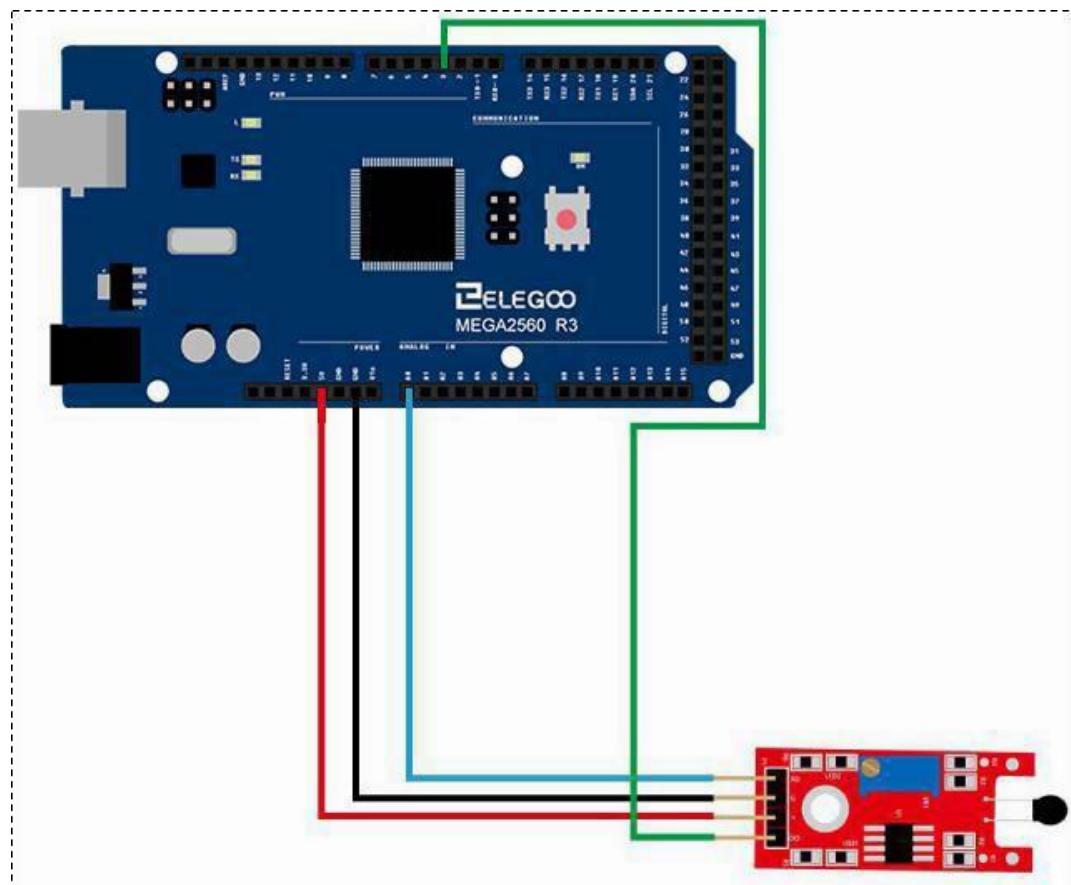
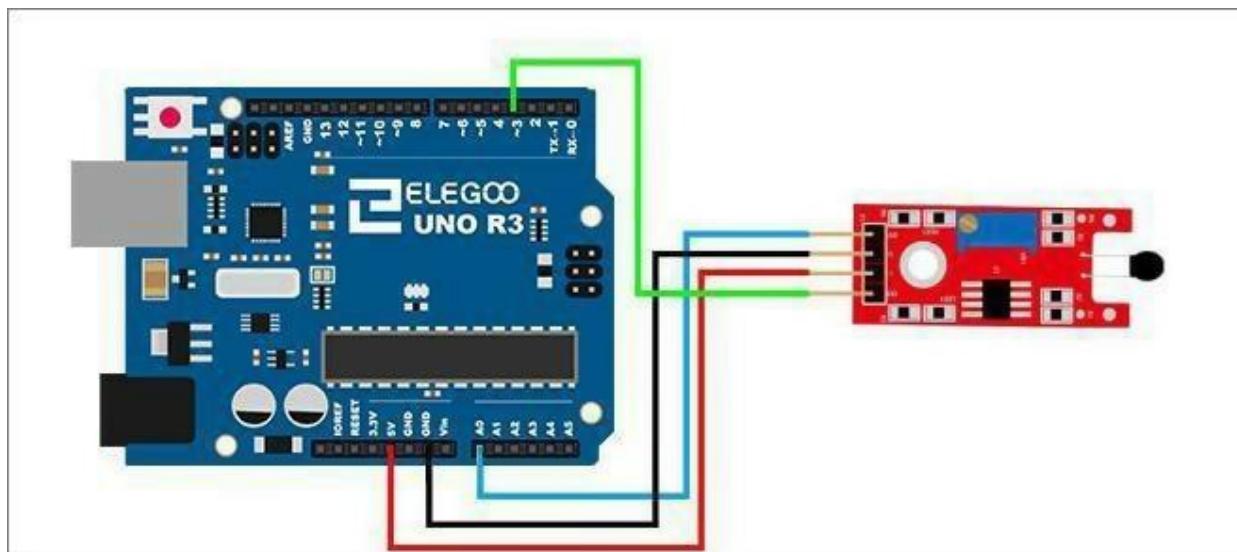


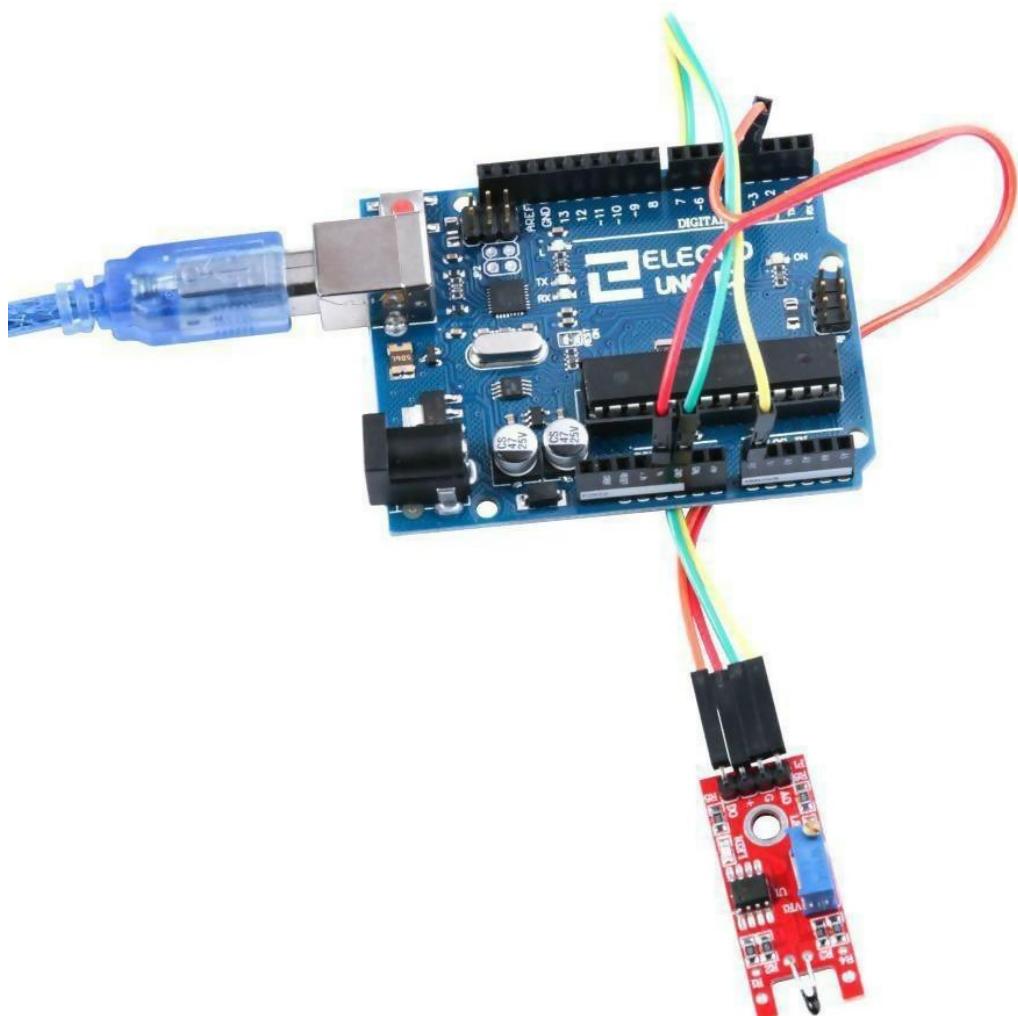
Diagrama de cableado

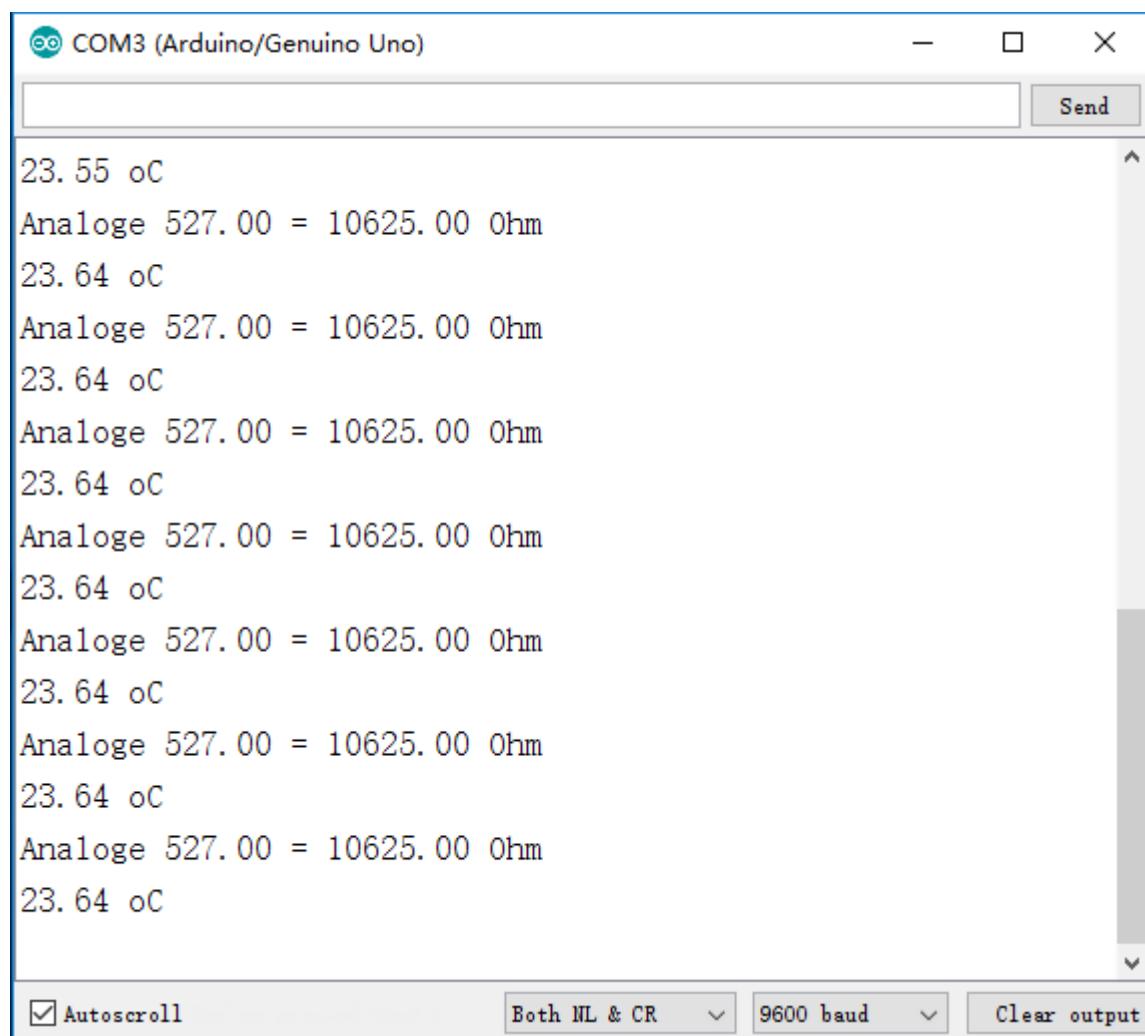


Código

Después de realizar el cableado, por favor abre el programa en la carpeta de código "Lección 18 Módulo Digital de Temperatura " y corre el compilador. Cuando el sensor mide la temperatura, el módulo emite datos que reflejan la misma. Verás que el LED comenzará a destellar. Abre el monitor y te mostrará datos parecidos a los siguientes:

Imagen ejemplo





The screenshot shows a terminal window titled "COM3 (Arduino/Genuino Uno)". The window displays a series of sensor readings. The text in the terminal is as follows:

```
23.55 °C
Analoge 527.00 = 10625.00 Ohm
23.64 °C
```

At the bottom of the window, there are several configuration options: "Autoscroll" (checked), "Both NL & CR", "9600 baud", and "Clear output".

A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
byte NTCPin = A0;  
  
#define SERIESRESISTOR 10000  
  
#define NOMINAL_RESISTANCE 10000  
  
#define NOMINAL_TEMPERATURE 25  
  
#define BCOEFFICIENT 3950  
  
void setup()  
  
{Serial.begin(9600);  
  
}  
  
void loop()  
  
{float ADCvalue; float Resistance;  
ADCvalue      =  
analogRead(NTCPi  
n);  
Serial.print("Anal  
oge          ");  
Serial.print(ADCv  
alue);  
Serial.print(" = ");  
//convert value to  
resistance  
Resistance     =  
(1023 / ADCvalue)  
- 1;  
Resistance      =  
SERIESRESISTOR /  
Resistance;  
Serial.print(Resistance);  
Serial.
```

```
http://www.elegoo.com
println(
"
Ohm");
float
steinh
art;
steinhart = Resistance /
NOMINAL_RESISTANCE; // (R/Ro)
steinhart = log(steinhart); // ln(R/Ro)
steinhart /= BCOEFFICIENT; // 1/B * ln(R/Ro)

steinhart += 1.0 / (NOMINAL_TEMPERATURE + 273.15); // + (1/To)

steinhart = 1.0 /
steinhart; // Invert
steinhart -=
273.15; // convert
to C
Serial.print(steinh
art);
Serial.println("C");
delay(1000);

}
```

Lección 19 Módulo de Sensor Hall Magnético Lineal

Resumen

En este experimento, aprenderemos a utilizar el módulo de sensor hall.

Módulo de Sensor Hall Magnético Lineal

El Módulo de Sensor Hall Magnético Lineal sirve para detectar la presencia de un campo magnético cerca del sensor. Variables tales como la fuerza del campo, polaridad y posición de la magneto en relación al sensor afectarán el punto en el cual la salida 'DO' hace el cambio al nivel alto (High). La sensibilidad del circuito se puede ajustar con un potenciómetro.

En el pin 'AO' se encuentra disponible una señal analógica de salida.



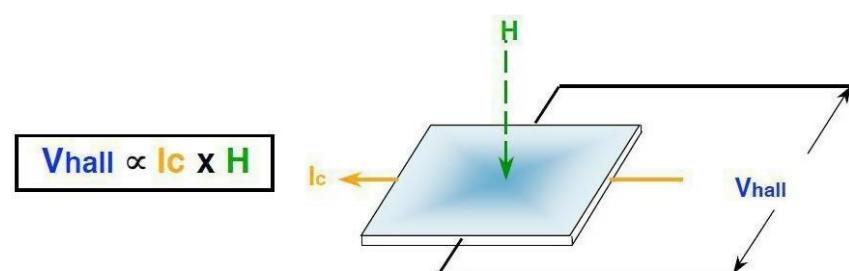
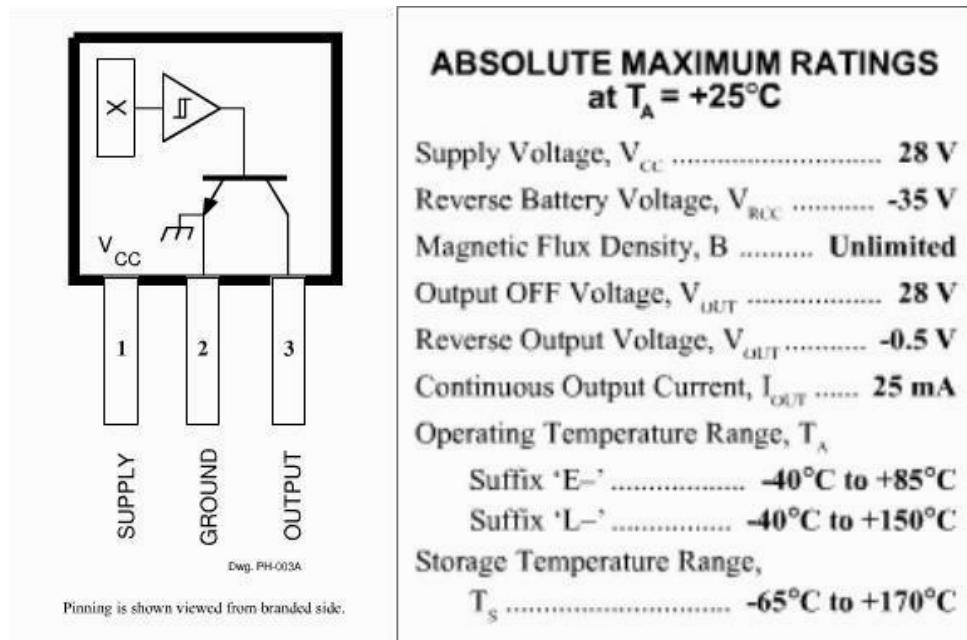
- 1
- 2
- 3
- 4

- 1.D0:digital output
- 2.VCC: 3.3V-5V DC
- 3.GND:ground
- 4.AO:analog output

Componentes Requeridos:

- 1x Elegoo Uno R3
- 1x Módulo de Sensor Hall Magnético Lineal 1x Cable USB
- 4x Cables F-M

Introducción de Componente Sensor Hall



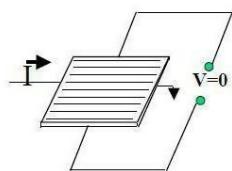
V_{Hall} = Output Hall-effect voltage

H = Magnetic Flux created by magnet or current-carrying conductor

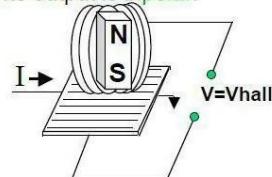
I_c = Constant supply current

Hall-effect Sensing Mechanism

- The current source is applied through a thin sheet of semiconductor material.

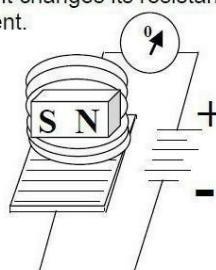


- A magnetic field applied **perpendicular** to the element creates a voltage change = V_{Hall} . Its output is **bipolar**.

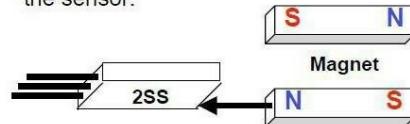


Magnetoresistive Sensing Mechanism

- A magnetic field applied **parallel** to the element changes its resistance and creates a current.



- MR is **omnipolar**—either pole will operate the sensor.



Factores de diseño – Tipos magnéticos

Unipolar: Solo un polo sur operará el sensor. El sensor enciende cuando censa un polo sur (+) y se apaga cuando el mismo ya no está.

Bipolar: La salida del sensor depende del polo. Un polo sur (+) activa el sensor; un polo norte (-) lo desactiva. Es posible que el sensor encienda y apague permaneciendo en un nivel Gauss positivo.

Enclavamiento: Las especificaciones son estrictas en cuanto al enclavamiento. Algunas veces se les diseña para asegurarse que cuando un polo sur (+) se retire del sensor, el mismo permanezca enclavado hasta ver el polo opuesto (-).

Omni polar: El sensor se diseña para operar de forma lineal Radiométrica: La salida es proporcional a la fuerza del campo magnético. El rango de sensibilidad se la salida es de 2.5 – 3.75 mV por unidad de Gauss

Esquema de conexión

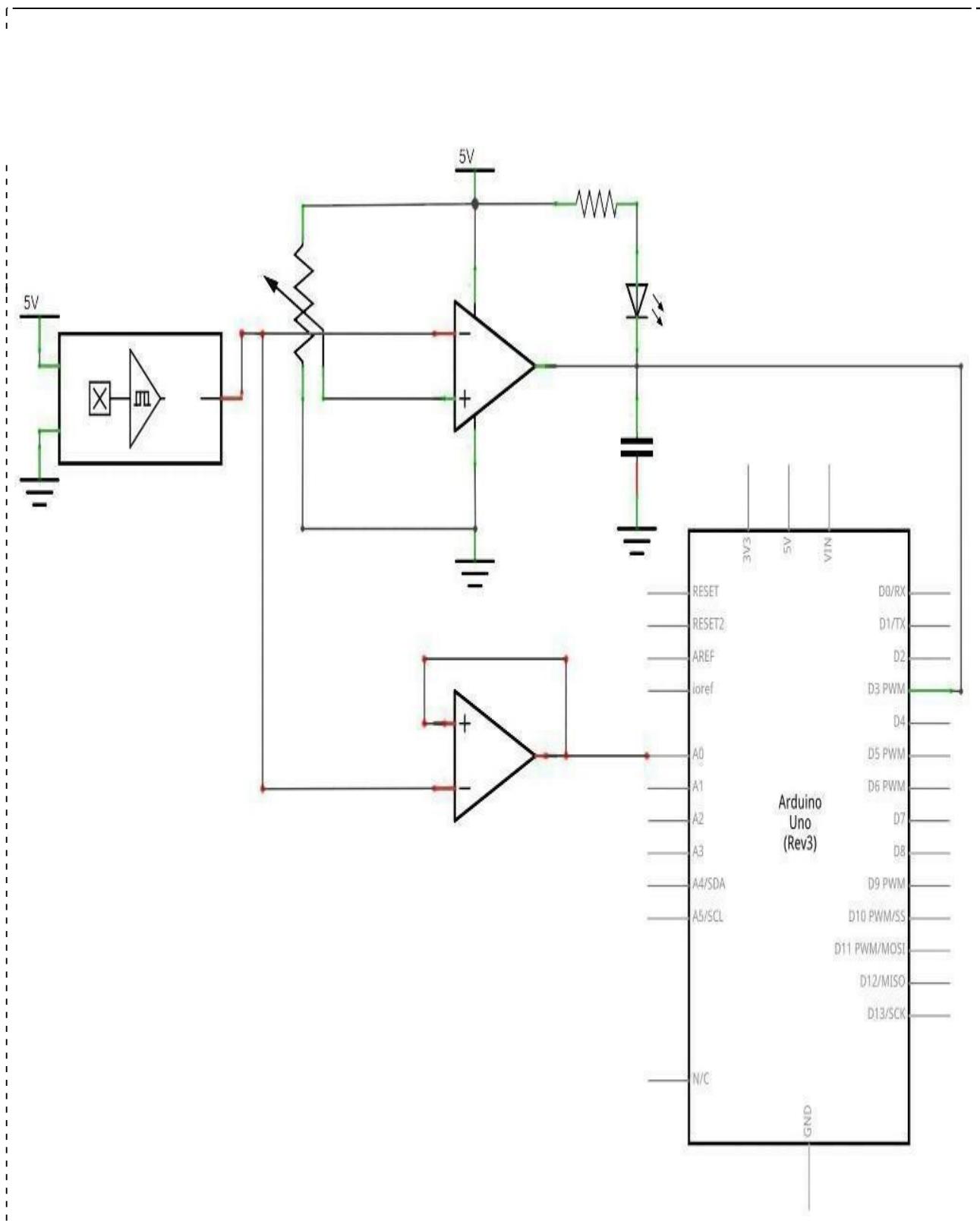
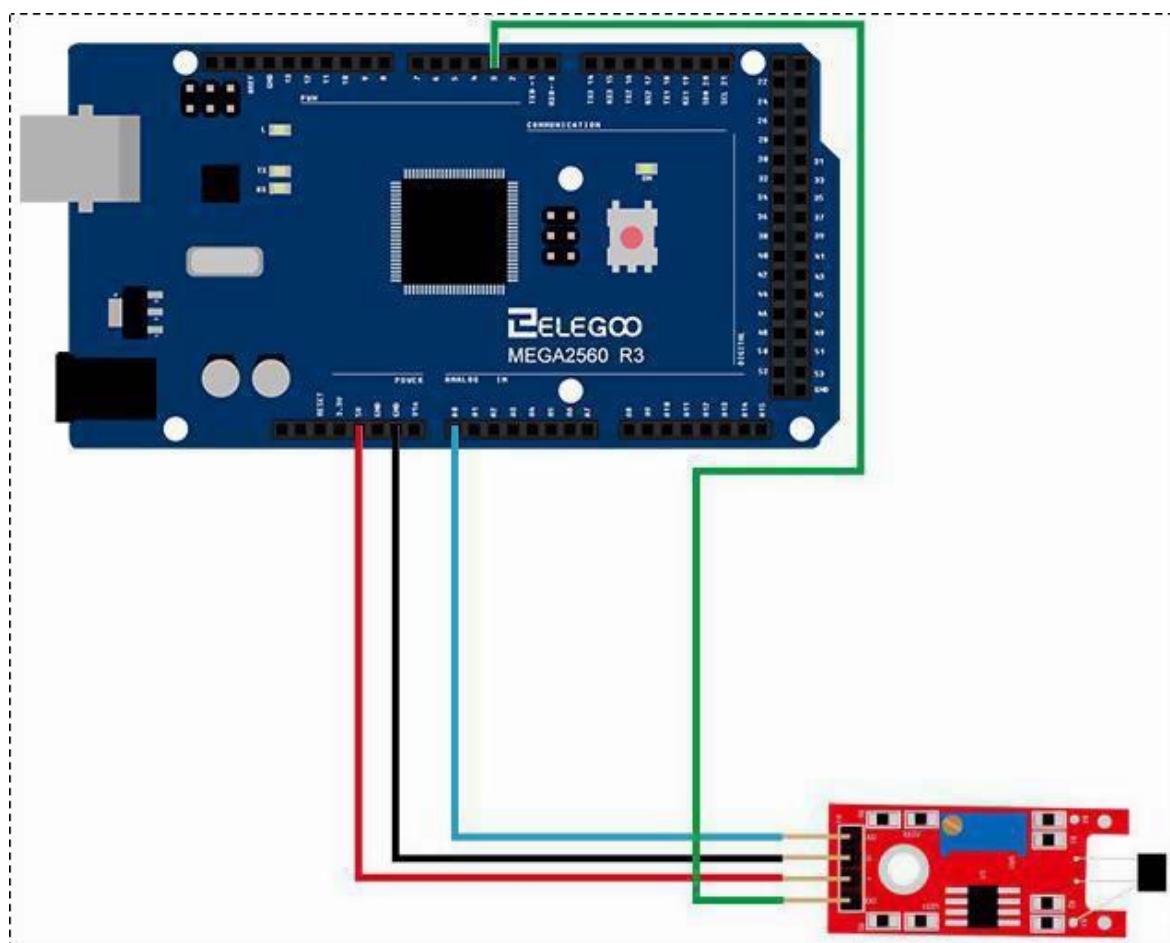
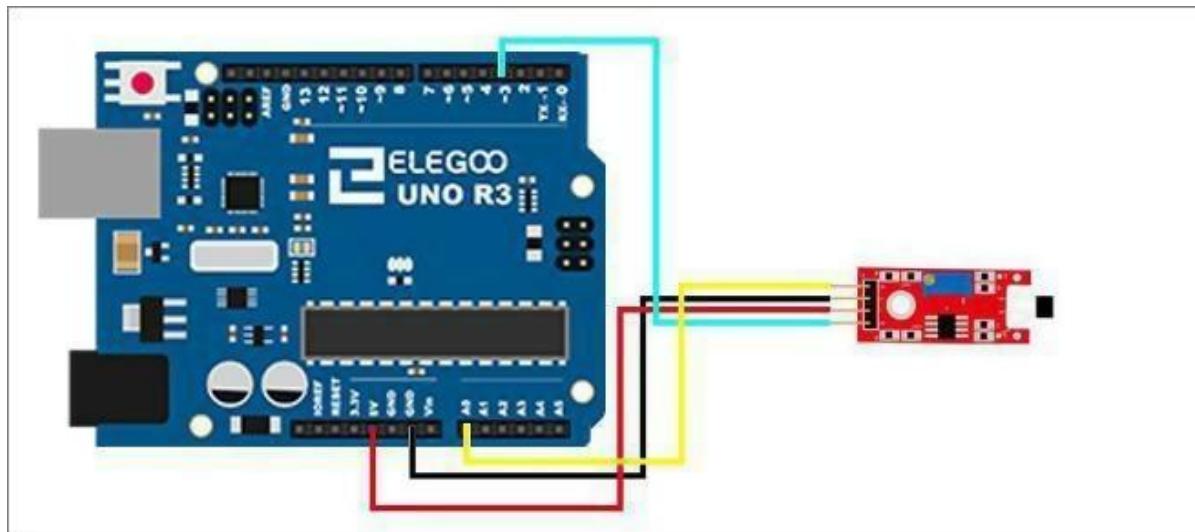


Diagrama de cableado

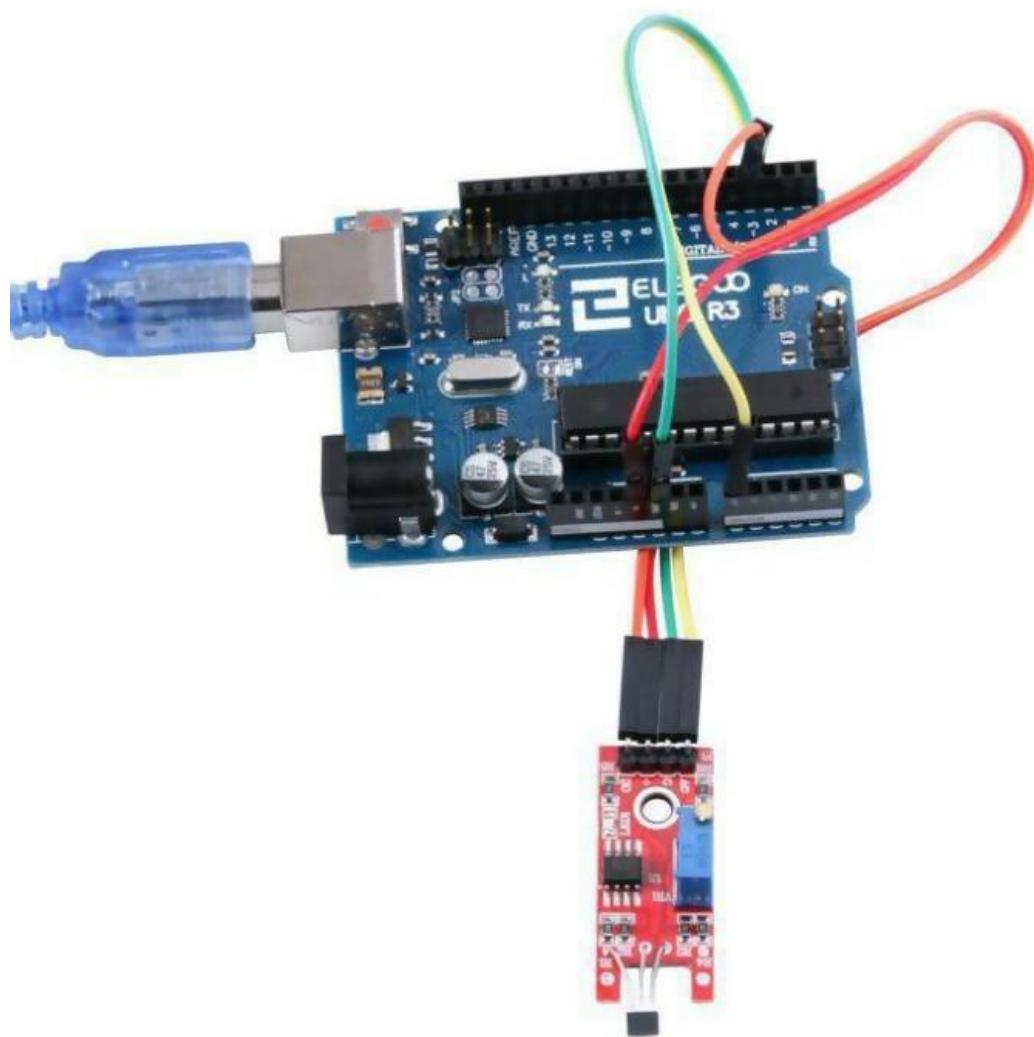


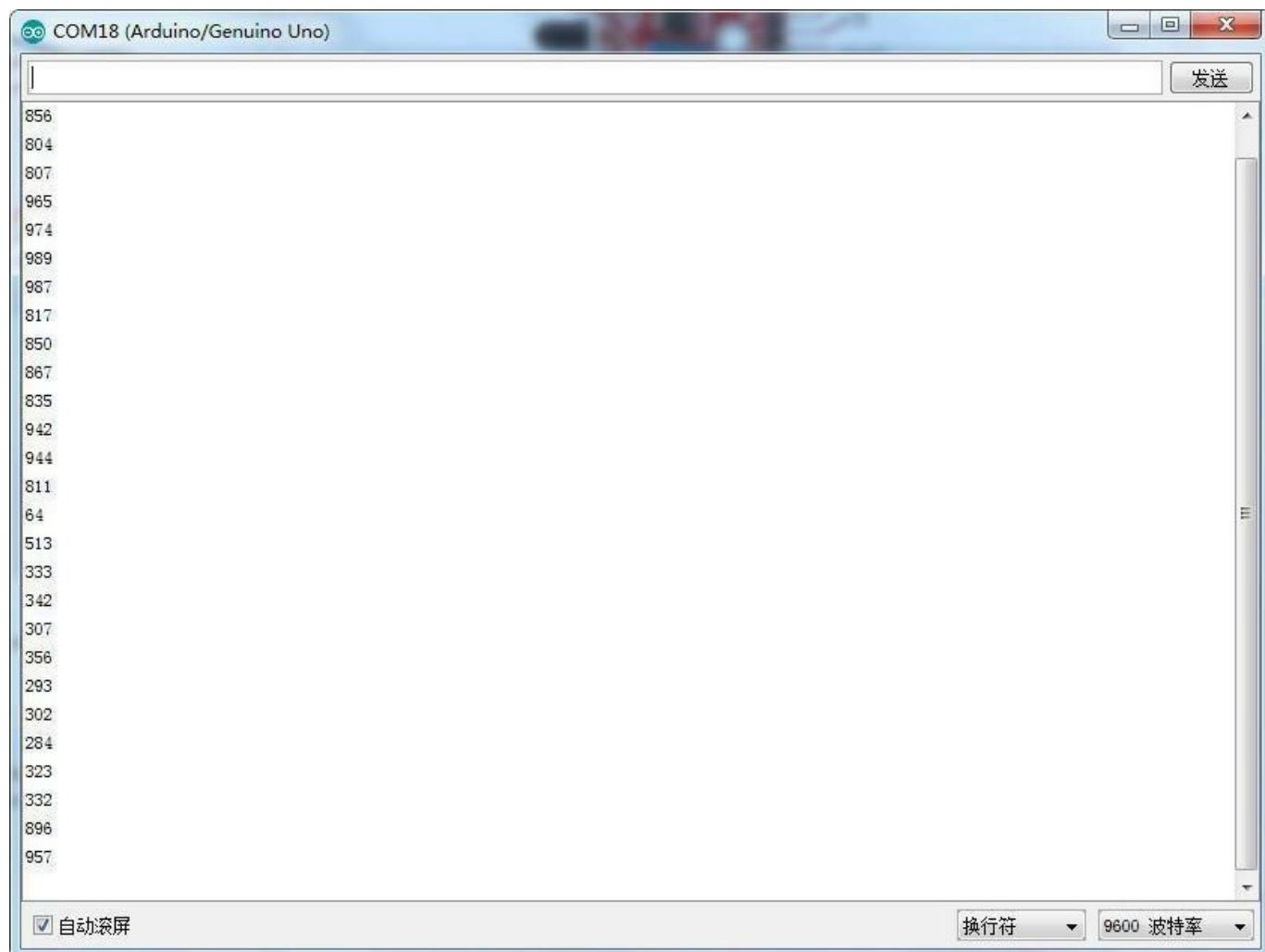
Código

Después de conectar el circuito, abrimos la carpeta "code" en el tutorial y buscamos la carpeta "Lección 19 Módulo Hall Lineal y Hall Analógico" para abrir el programa.

Para el módulo de sensor hall, podemos seleccionar el tipo de salida: digital o analógica. En la siguiente figura, usamos el puerto de salida DO. Para saber si el sensor hall está captando la fuerza magnética, después de que encienda la luz, entra al monitor para ver los siguientes datos:

Imagen ejemplo





A continuación se muestra el código usado en este experimento y su correspondiente explicación:

(1) Programa de control del puerto digital

```
//define el puerto del LED int Led = 13;  
//define el puerto del interruptor int buttonpin = 13;  
//define la variable digital val int val;  
void setup()  
{  
//define el LED como una salida PinMode (Led, OUTPUT);  
//define el interruptor como un puerto de salida  
  
pinMode(buttonpin,INPUT);  
}
```

```
{  
//Lee el valor de la interfaz digital 3 asignada a val val=digitalRead(buttonpin);  
// cuando el sensor o interruptor tenga señal, el LED destellará if(val==HIGH)  
{  
digitalWrite(Led,HIGH);  
}  
else  
{  
digitalWrite(Led,LOW);  
}  
}
```

(2) Simulación de los procedimientos de control de boca

```
// Selecciona el pin de entrada para el potenciómetro int sensorPin = A0;  
// Selecciona el pin para el LED int ledPin = 13;  
// variable para almacenar el valor que indica el sensor int sensorValue = 0;
```

```
void setup()  
{  
pinMode(ledPin,OUTPUT); Serial.begin(9600);  
}  
  
void loop(){  
sensorValue =analogRead(sensorPin); digitalWrite(ledPin, HIGH); delay(sensorValue);  
digitalWrite(ledPin, LOW); delay(sensorValue); Serial.println(sensorValue, DEC);  
}
```

Lección 20 Módulo de Sensor de Llama

Resumen

En este experimento, aprenderemos a utilizar el módulo de sensor de llama.

Este módulo puede censar la presencia de llamas y radiación. También puede detectar fuentes de luz ordinarias en el rango de longitud de onda de entre 760nm-1100 nm. La distancia de detección es de hasta 100 cm. El sensor de llama puede emitir salidas digitales o analógicas. Puede utilizarse como para alarmas de llama o en robots bomberos.

Módulo de sensor de llama

Es un módulo de sensor que se utiliza para detectar llamas. La sensibilidad espectral del sensor está optimizada para detectar las emisiones de las llamas. La señal de salida en 'DO' pasa a nivel alto cuando se detecta una llama. El umbral de conmutación se puede ajustar por medio de un potenciómetro pre-ajustado. En el pin 'AO' se encuentra disponible una señal analógica de salida.

- Sensibilidad espectral típica: 720-1100 nm
- Ángulo típico de detección: 60°



- 1.DO:digital output
- 2.VCC: 3.3V-5V DC
- 3.GND:ground
- 4.AO:analog output

Componentes Requeridos:

1 x Elegoo Uno R3 1 x cable USB

1 x Modulo sensor de llama 4 x Cables F-M

Introducción de Componentes Módulo de sensor de llama:

Detecta llamas o una fuente de luz con una longitud de onda en el rango de 760nm-1100 nm

Distancia de detección: 20cm (4.8V) ~ 100cm (1V)

Angulo de detección de aproximadamente 60 grados, sensible al espectro de la llama. Chip comparador LM393 hace que las lecturas del módulo sean estables.

Rango de detección ajustable. Voltaje de funcionamiento 3.3V-5V Salidas digitales y analógicas

La salida DO comuta entre 0 y Salida analógica de voltaje AO Indicador de encendido e indicador de conmutación salida digital

Esquema de Conexión

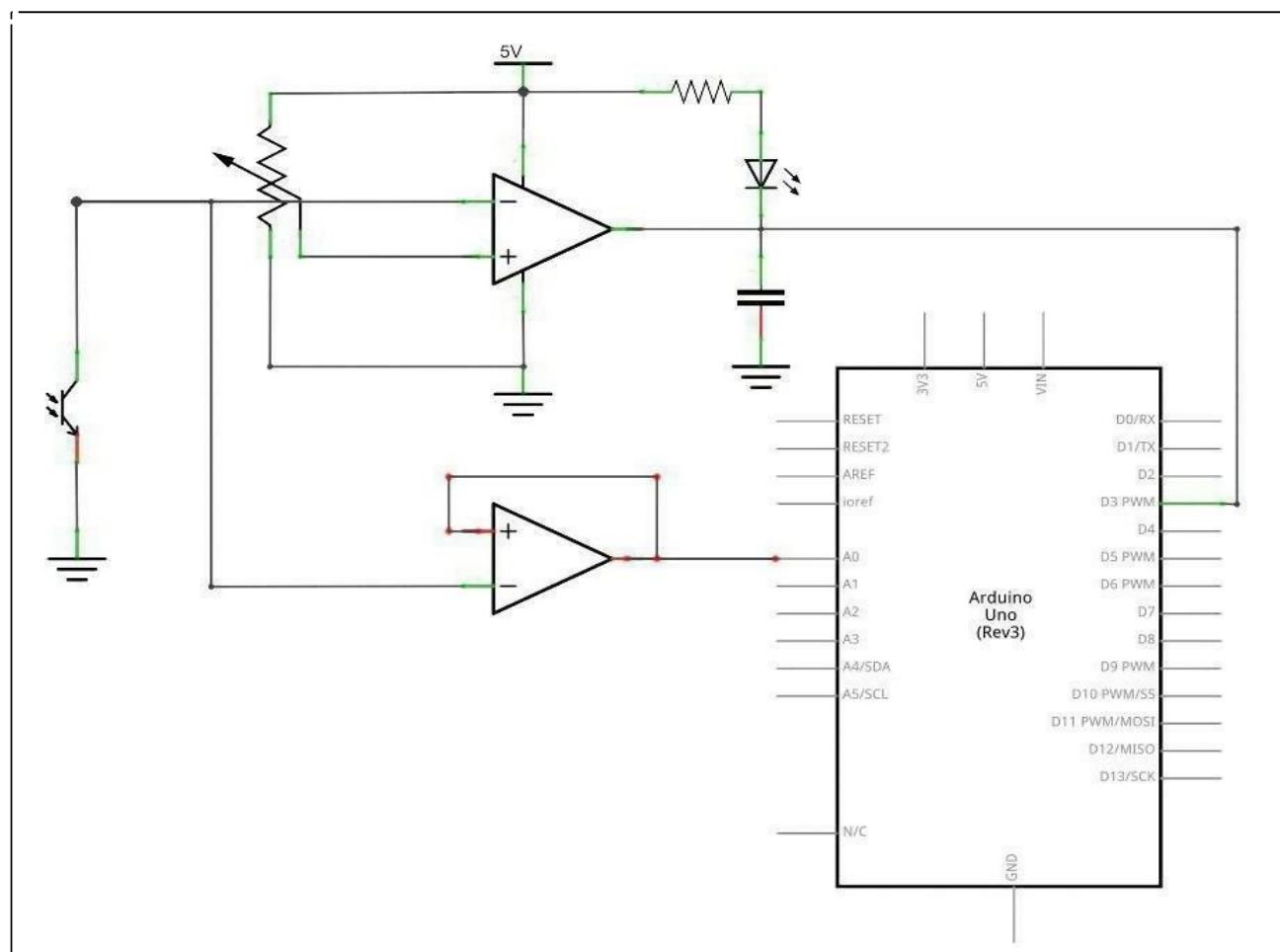
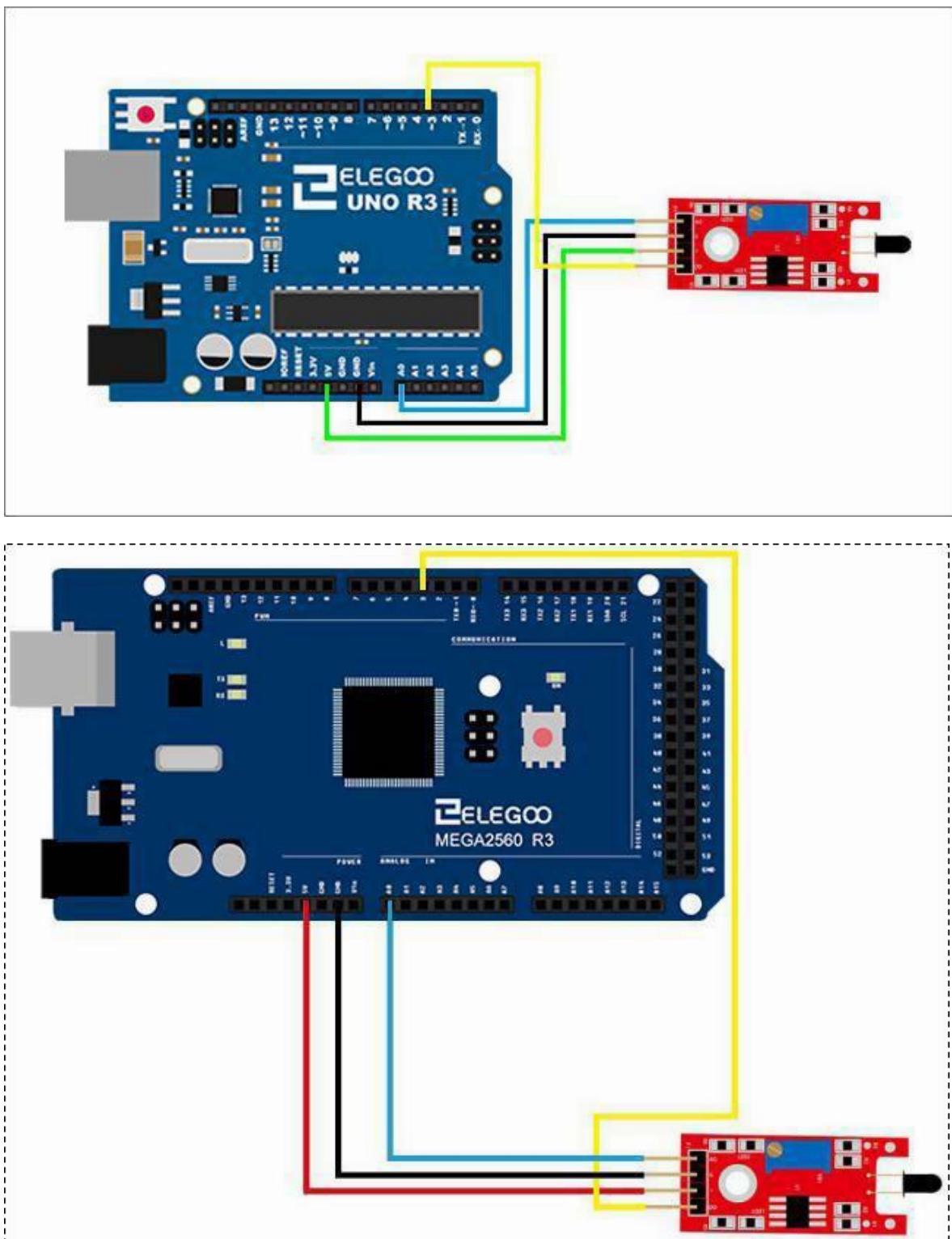


Diagrama de cableado

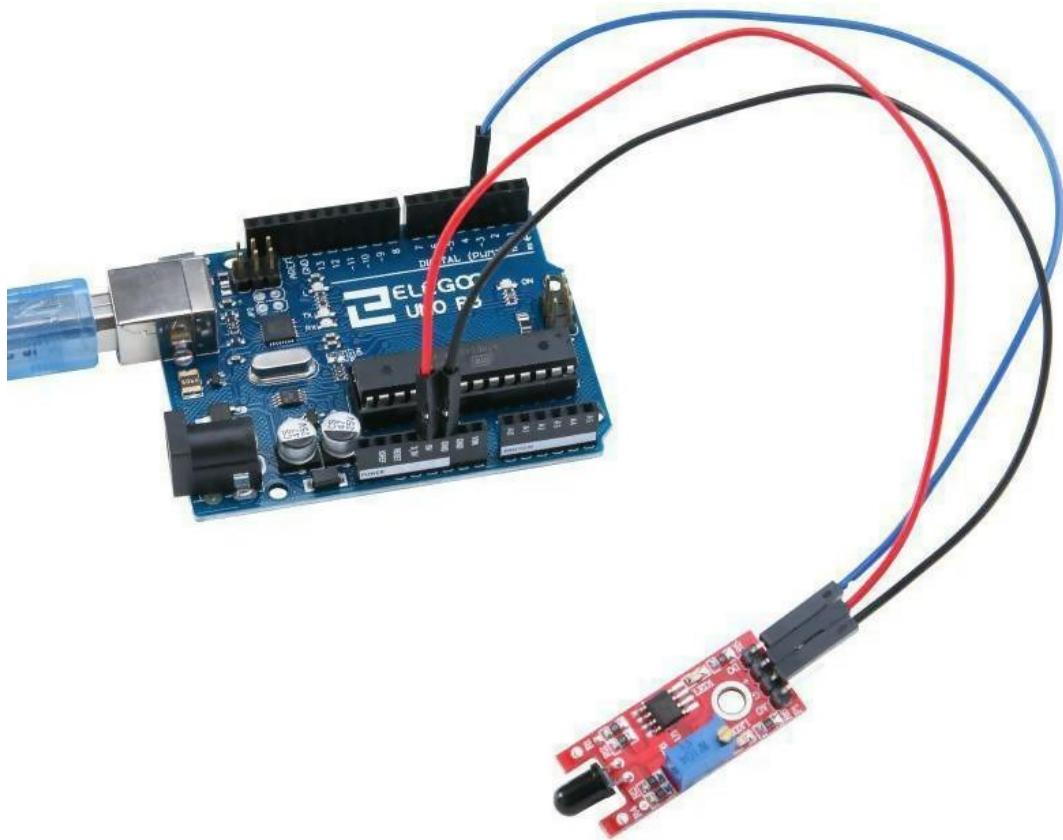


Código

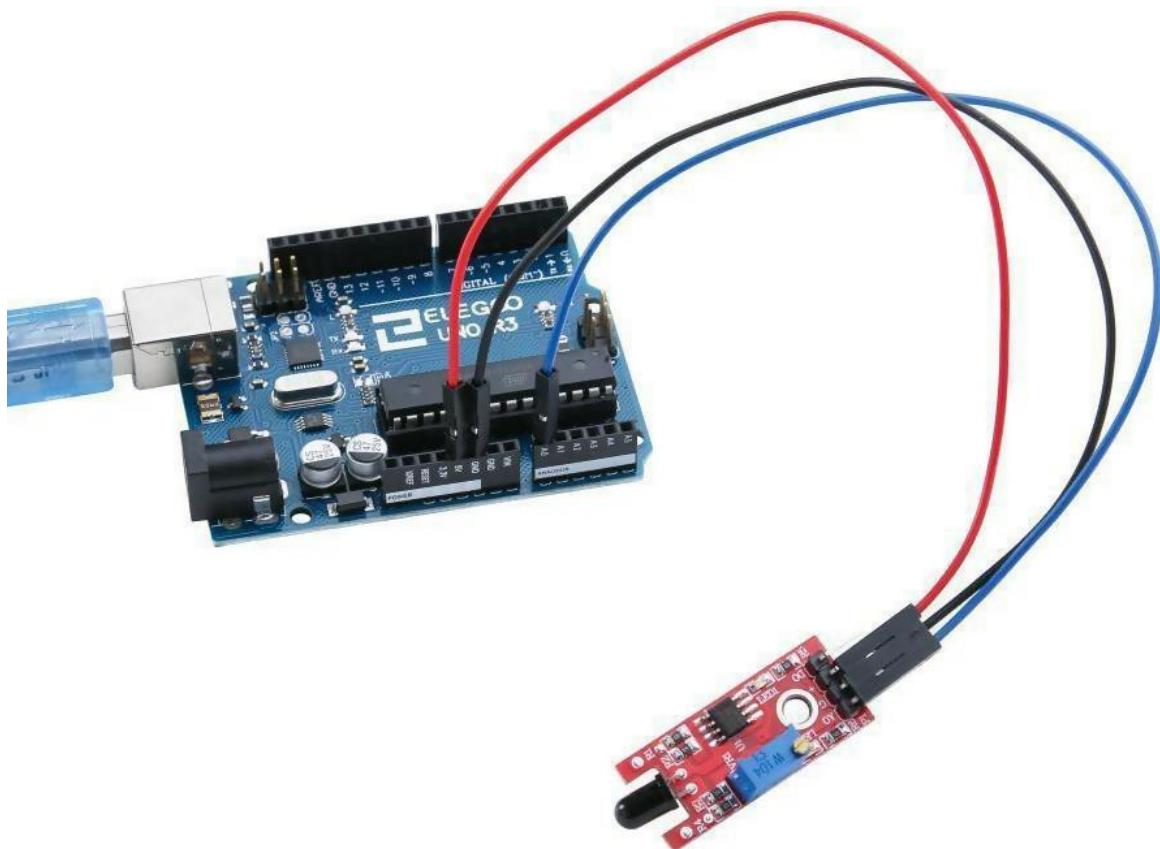
Después de conectar el circuito, abrimos la carpeta "code" en el tutorial y buscamos la carpeta "Lección 20 Módulo de sensor de llama" para abrir el programa,

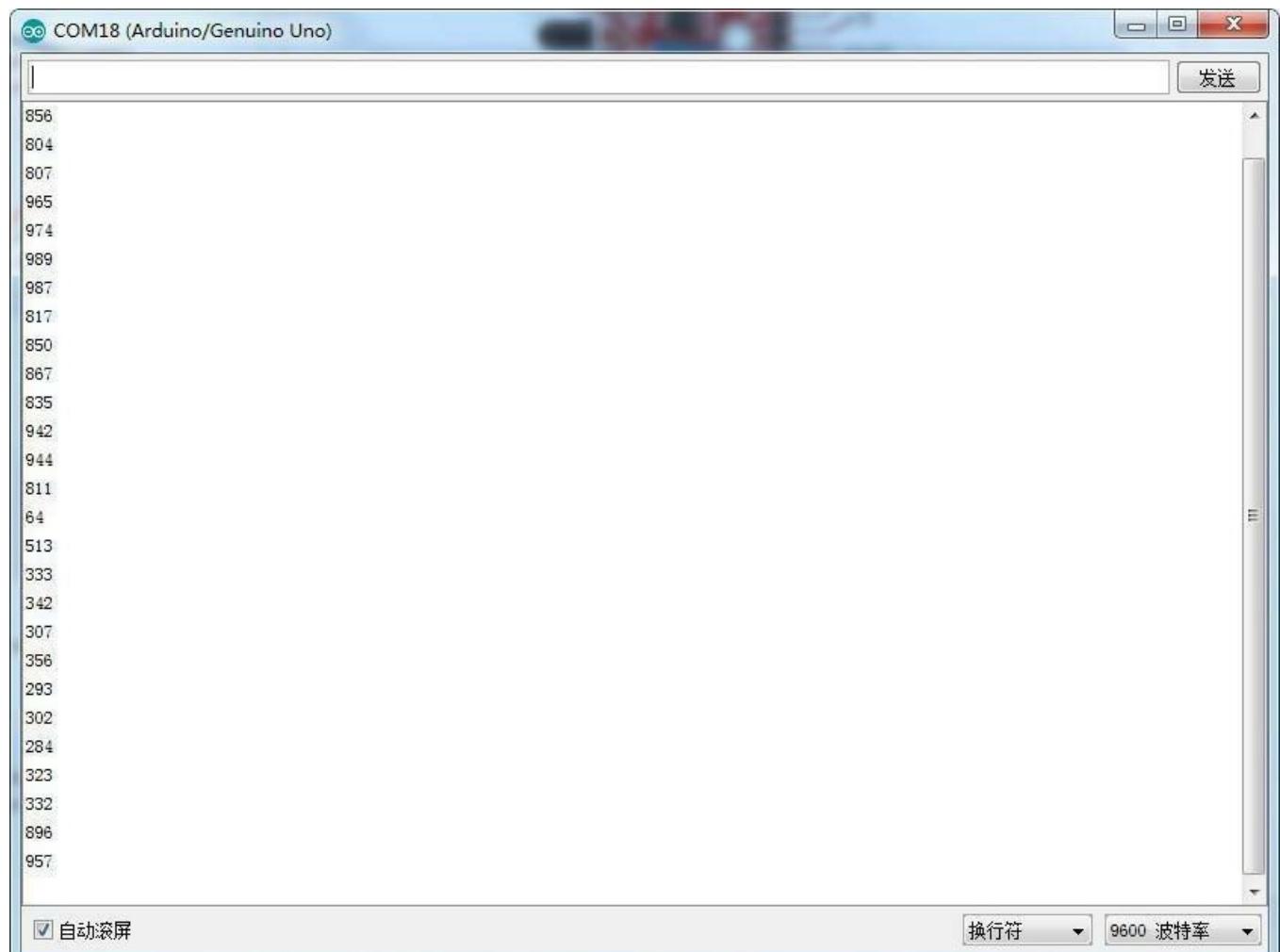
Para el módulo de sensor de llama, podemos seleccionar el tipo de salida: digital o analógica. En la siguiente figura, usamos el puerto de salida DO. Para saber si el detector de llama está censando, después de que encienda la luz, entra al monitor para ver los siguientes datos:

Imagen ejemplo



En la siguiente figura, usamos el puerto de salida AO. Cuando el sensor detecte llama, el módulo enviará en su salida la data que refleja la fuerza de la misma. Este valor estará entre 0 y 1023.





A continuación se muestra el código usado en este experimento y su correspondiente explicación:

(1) Programa de control del puerto digital

```
//define el puerto del LED int Led = 13;  
//define el puerto del interruptor int buttonpin = 13;  
//define digital variable val int val;  
void setup()  
{  
//define el LED como una salida PinMode (Led, OUTPUT);  
//define el interruptor como una salida pinMode(buttonpin,INPUT);  
}  
void loop()  
{  
//Lee el valor de la interfaz digital 3 asignada a val val=digitalRead(buttonpin);  
// cuando el sensor del interruptor tenga señal, el LED destellará if(val==HIGH)  
{  
digitalWrite(Led,HIGH);  
}  
else  
{  
digitalWrite(Led,LOW);  
}  
}
```

(2) Simulación de los procedimientos de control de boca

```
// Selecciona el pin de entrada para el potenciómetro int sensorPin = A0;  
// Selecciona el pin para el LED int ledPin = 13;  
// Variable para almacenar el valor que viene del sensor int sensorValue = 0;  
  
void setup()  
{  
pinMode(ledPin,OUTPUT); Serial.begin(9600);  
}  
void loop()
```

```
{  
sensorValue = analogRead(sensorPin); digitalWrite(ledPin, HIGH); delay(sensorValue); digitalWrite(ledPin,  
LOW); delay(sensorValue); Serial.println(sensorValue, DEC);  
}
```

Lección 21 Módulo de Sensor Táctil

Resumen

En este experimento, aprenderemos a utilizar el módulo metálico de sensor táctil.

Módulo de Sensor Táctil

Interruptor sensible al tacto. Tocar el sensor producirá una salida en el pin 'DO' La salida no es una señal limpia pero incluye señales inducidas de 50 Hz ('mains hum'). La señal de salida es "Alto activo" y la sensibilidad del circuito se pueden ajustar con un potenciómetro. En el pin 'AO' se encuentra disponible una señal analógica de salida.



- 1.DO:digital output
- 2.VCC: 3.3V-5V DC
- 3.GND:ground
- 4.AO:analog output

Componentes requeridos:

1 x Elegoo Uno R3 1 x cable USB

1 x Módulo de Sensor Táctil 4 x Cable F-M

Esquema de Conexión

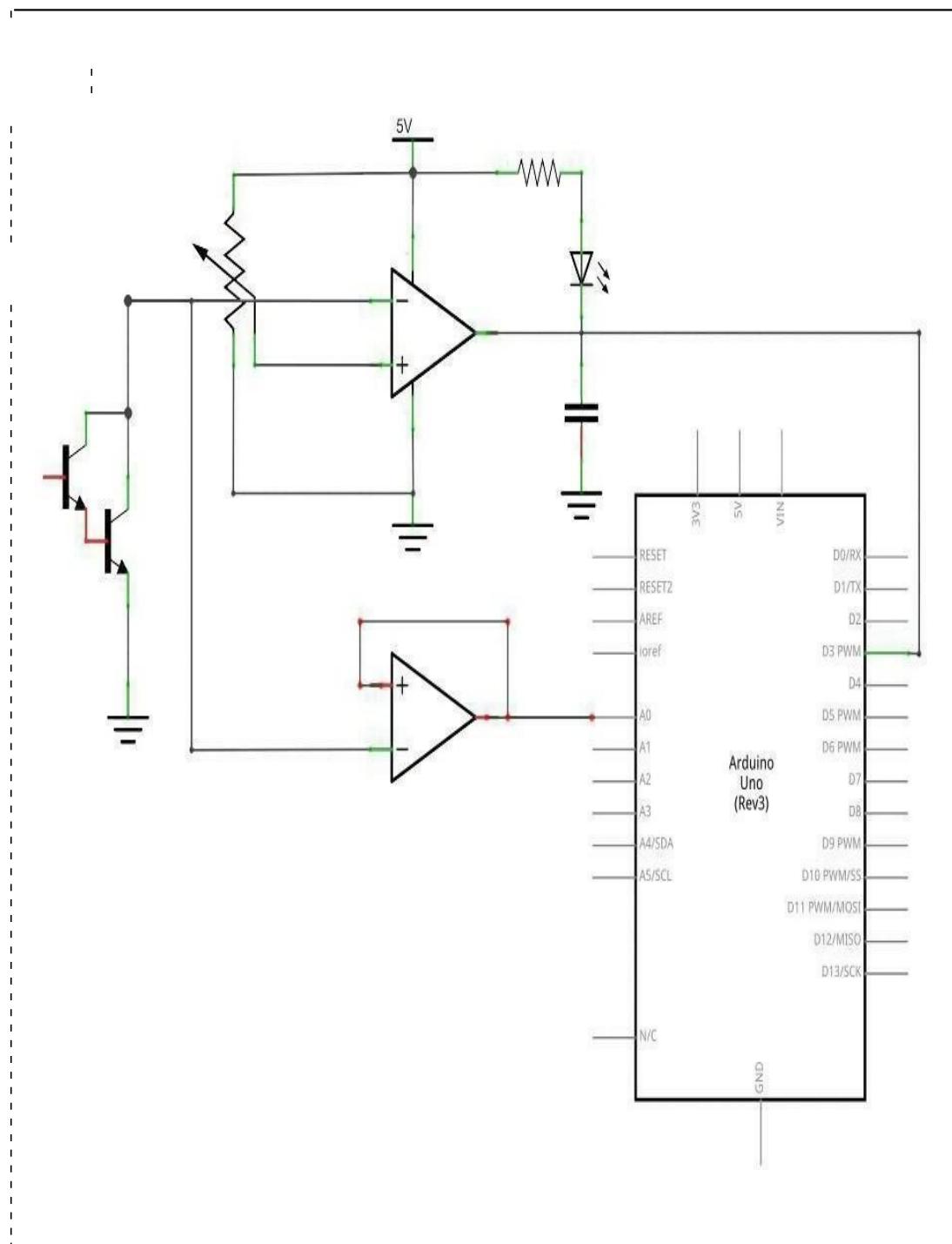
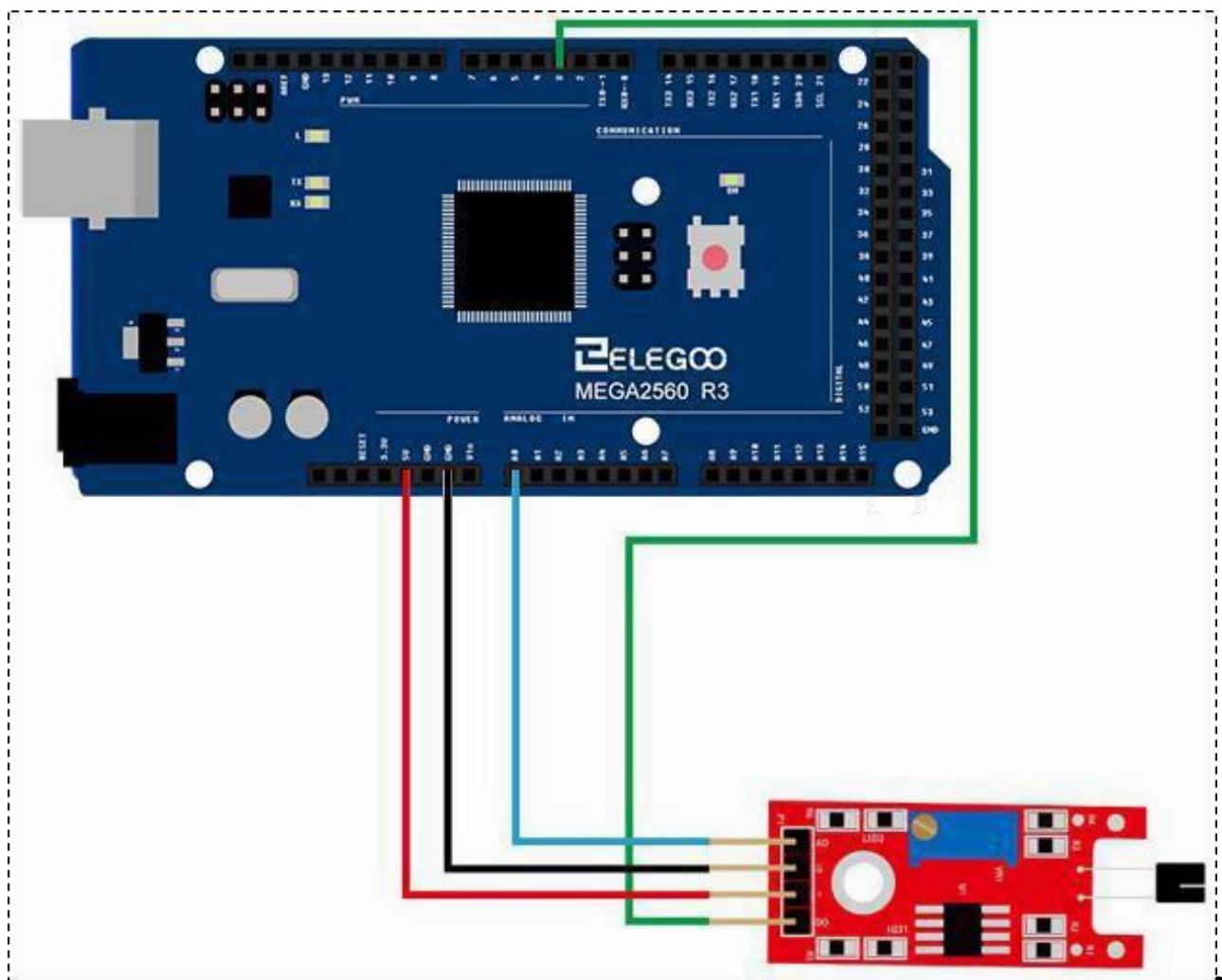
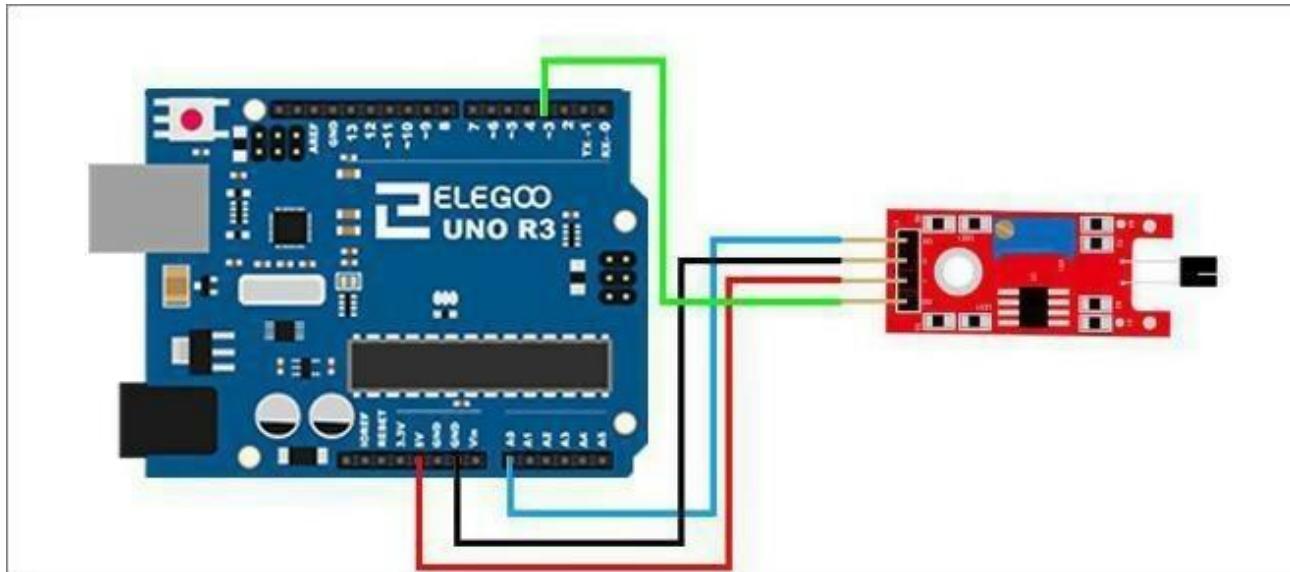


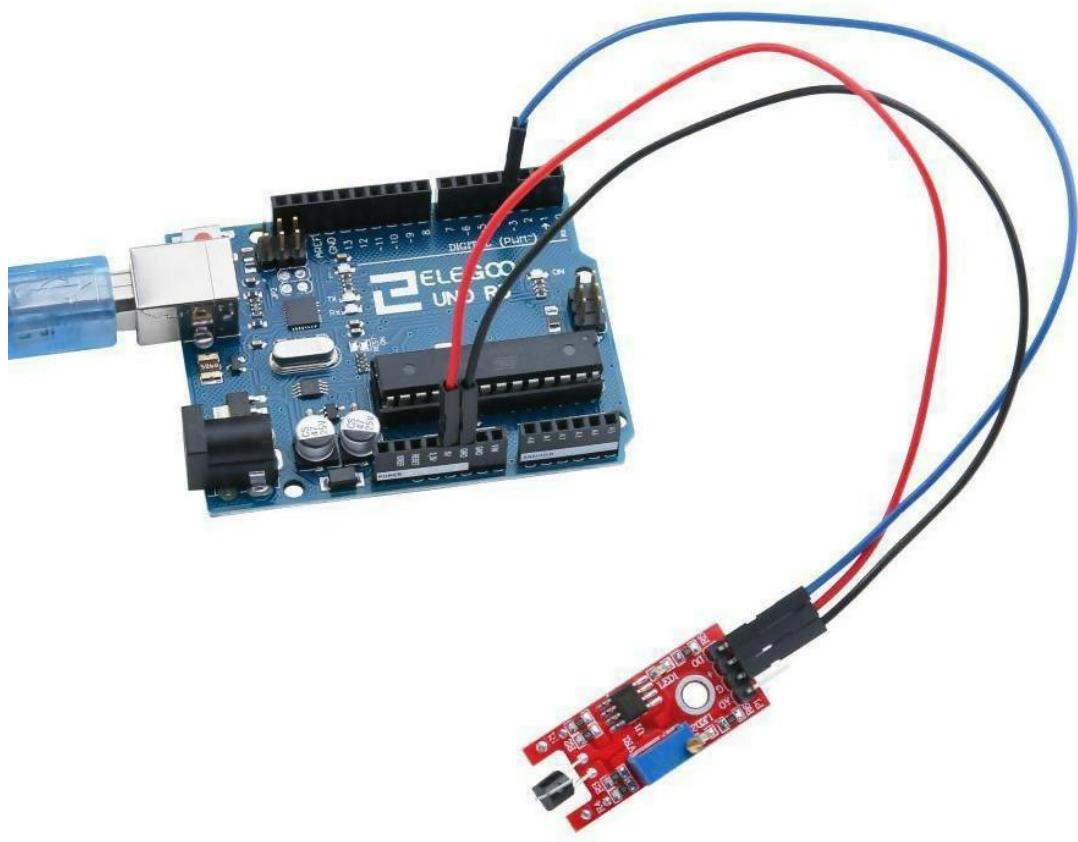
Diagrama de cableado



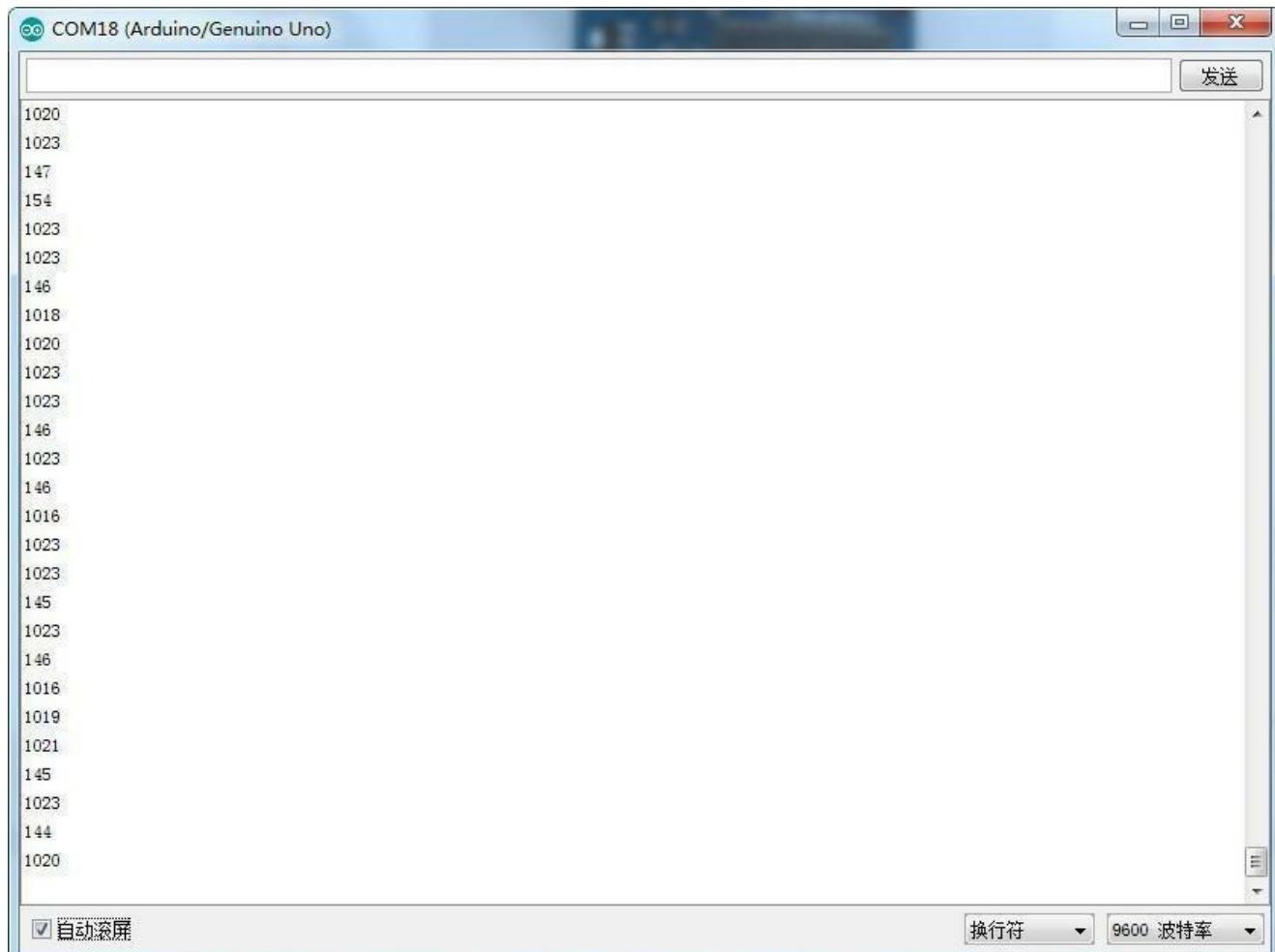
Código

Después de conectar los circuitos, abrimos la carpeta de códigos de nuestra documentación para encontrar la carpeta "Lección 21 Módulo de sensor táctil metálico". Abre y corre el compilador cargador. El módulo táctil metálico puede tener salidas digitales o analógicas. En la siguiente figura, usamos el puerto de salida DO. Para que veas como enciende la luz cuando el sensor se activa. Luego abre el monitor, para ver los siguientes datos:

Imagen ejemplo



En la siguiente imagen, utilizamos el puerto AO para salida. Cuando el sensor se active, el módulo generara una salida de datos. Este valor estará entre 0 y 1023.



A continuación se muestra el código usado en este experimento y su correspondiente explicación:

(1) Simulación del programa de control de puertos

```
//define el puerto del LED int Led = 13;  
//define el puerto del interruptor int buttonpin = 13;  
//define la variable digital val
```

```
int void
```

```
{
```

```
val;
```

```
setup()
```

```
//define el LED como una salida PinMode (Led, OUTPUT);
```

```
//define el interruptor como una salida pinMode(buttonpin,INPUT);
```

```
}
```

```
void
```

```
{
```

```
loop()
```

```
//Lee el valor de la interfaz digital 3 asignada a val val=digitalRead(buttonpin);
```

```
// cuando el sensor del interruptor tenga señal, el LED destellará if(val==HIGH)
```

```
{
```

```
digitalWrite(Led,HIGH);
```

```
}
```

```
else
```

```
{
```

```
digitalWrite(Led,LOW);
```

}

}

(2) Programa de control del puerto digital

```
// Selecciona el pin de entrada para el potenciómetro int sensorPin      = A0;  
// Selecciona el pin para el LED int ledPin    = 13;  
// variable para almacenar el valor que indica el sensor int sensorValue =0;  
  
void setup()  
{  
pinMode(ledPin,OUTPUT); Serial.begin(9600);  
}  
void loop()  
{  
sensorValue = analogRead(sensorPin); digitalWrite(ledPin, HIGH); delay(sensorValue); digitalWrite(ledPin,  
LOW); delay(sensorValue); Serial.println(sensorValue, DEC);  
}
```

Lección 22 Módulo LED Flash de Siete Colores

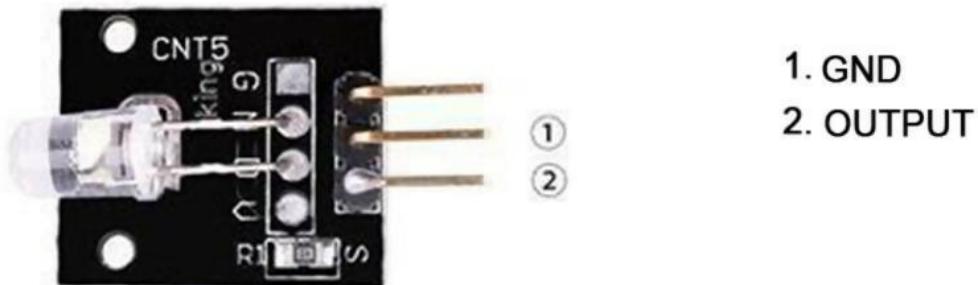
Resumen

En este experimento, aprenderemos a utilizar el Módulo LED Flash de Siete Colores.

Módulo LED Flash de Siete Colores

LED claro de 5mm para operación directa a 5V. El color del LED cambia en un ciclo automático.

con una secuencia de siete colores Los 5V de la fuente de poder se conectan con el pin 'S' y la tierra con el pin del centro.



Componentes Requeridos:

1 x Elegoo Uno R3 1 x cable USB

1 x Modulo LED Flash de Siete Colores 2 x Cables F-M

Introducción de Componente LED Flash de Siete Colores

Este módulo utiliza LEDs redondos de 5mm y alto brillo con las siguientes características:

Tipo de Producto: LED

Modelo de Producto: YB-3120B4 Pn YG-PM

Forma: LED redondo de 5mm tipo DIP

Color: Rosado Amarillo Verde (Alto brillo)

Tipo de lente: whitemist

Voltaje estándar: 3.0-4.5V

Esquema de conexión

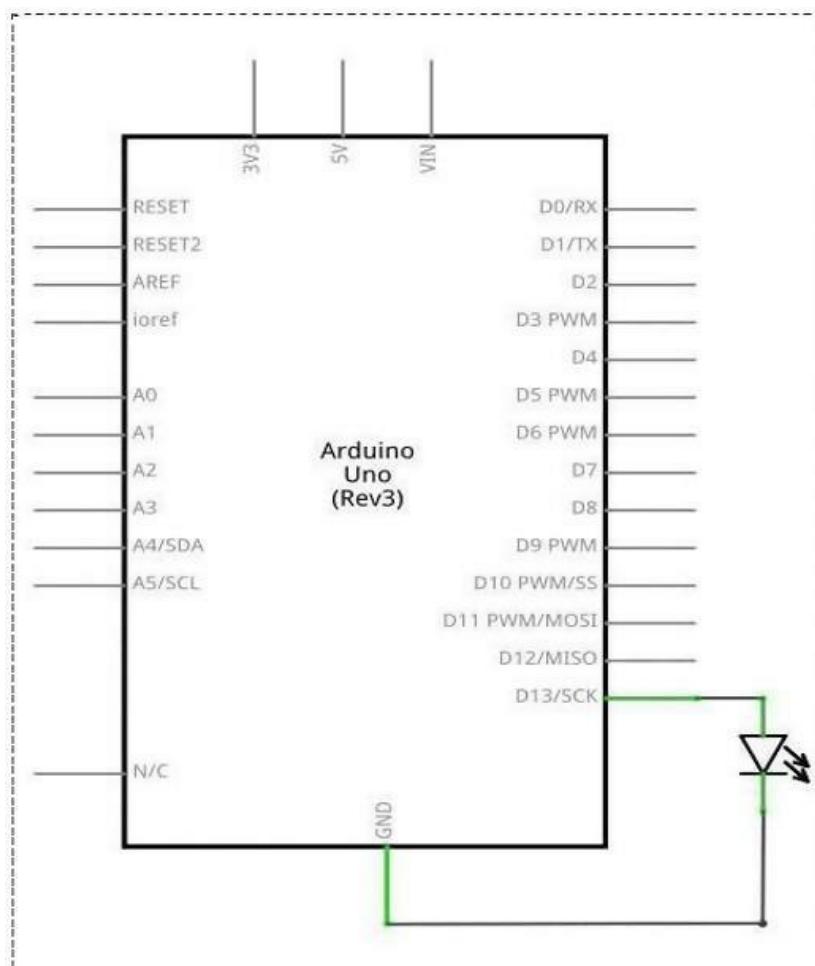
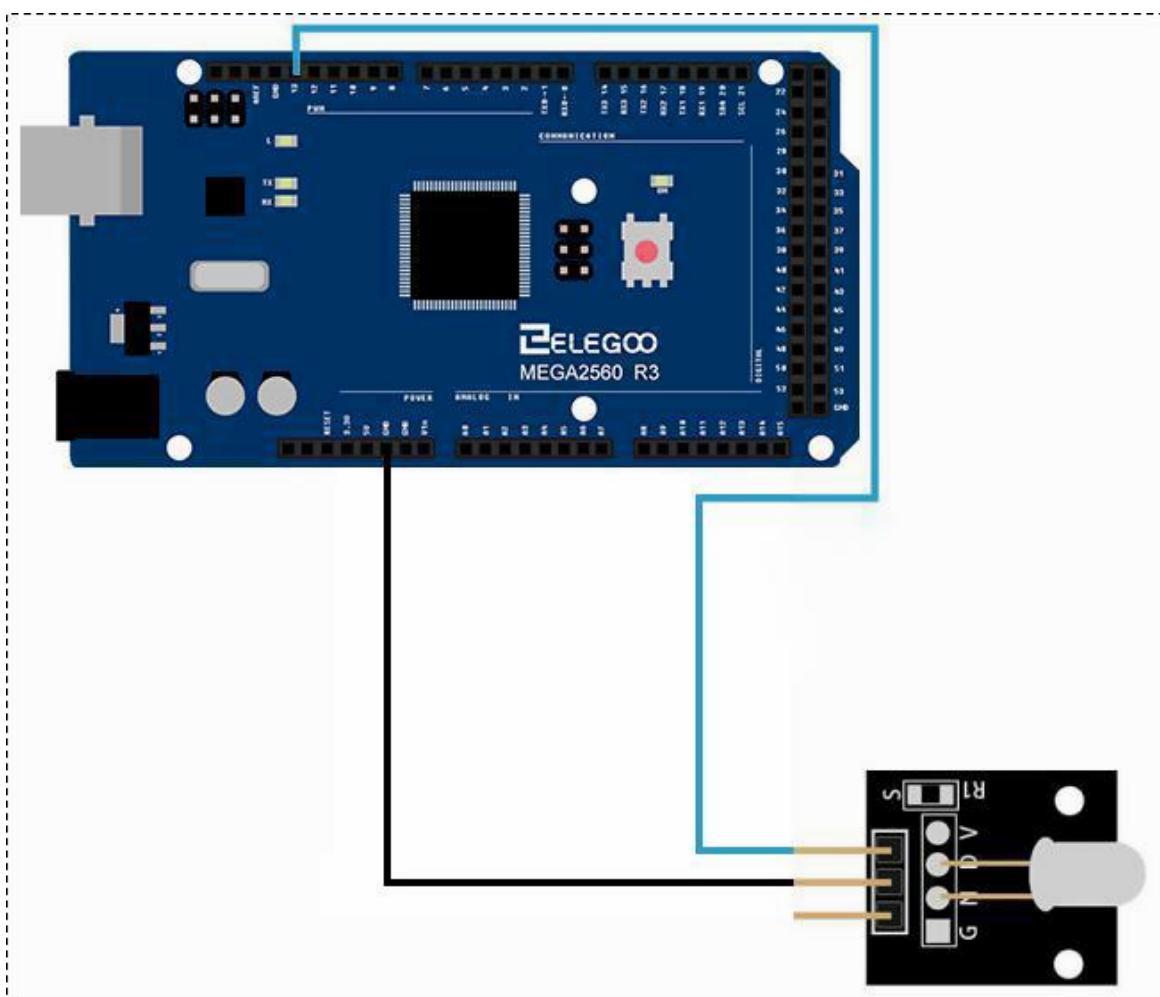
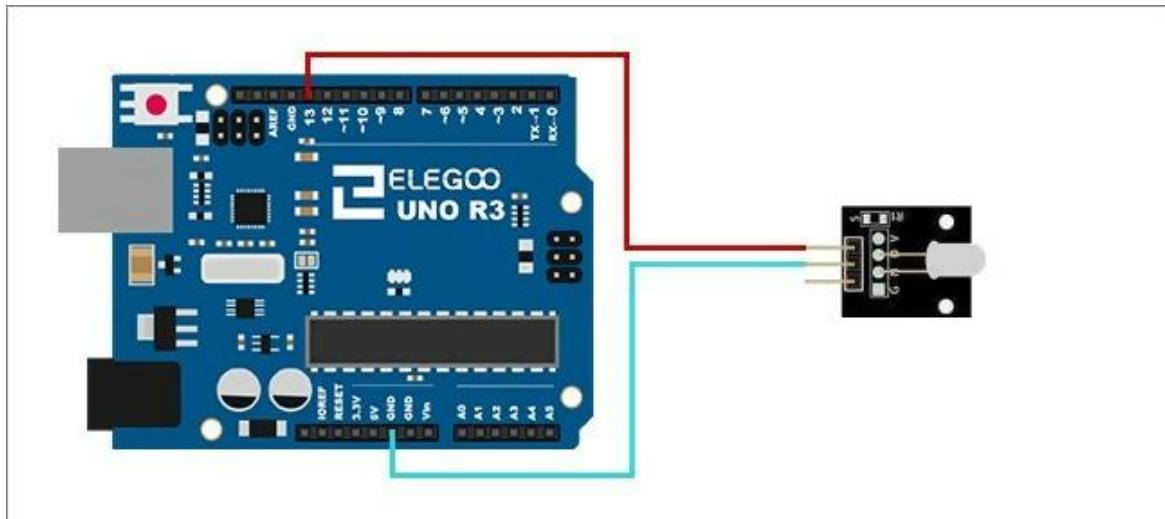


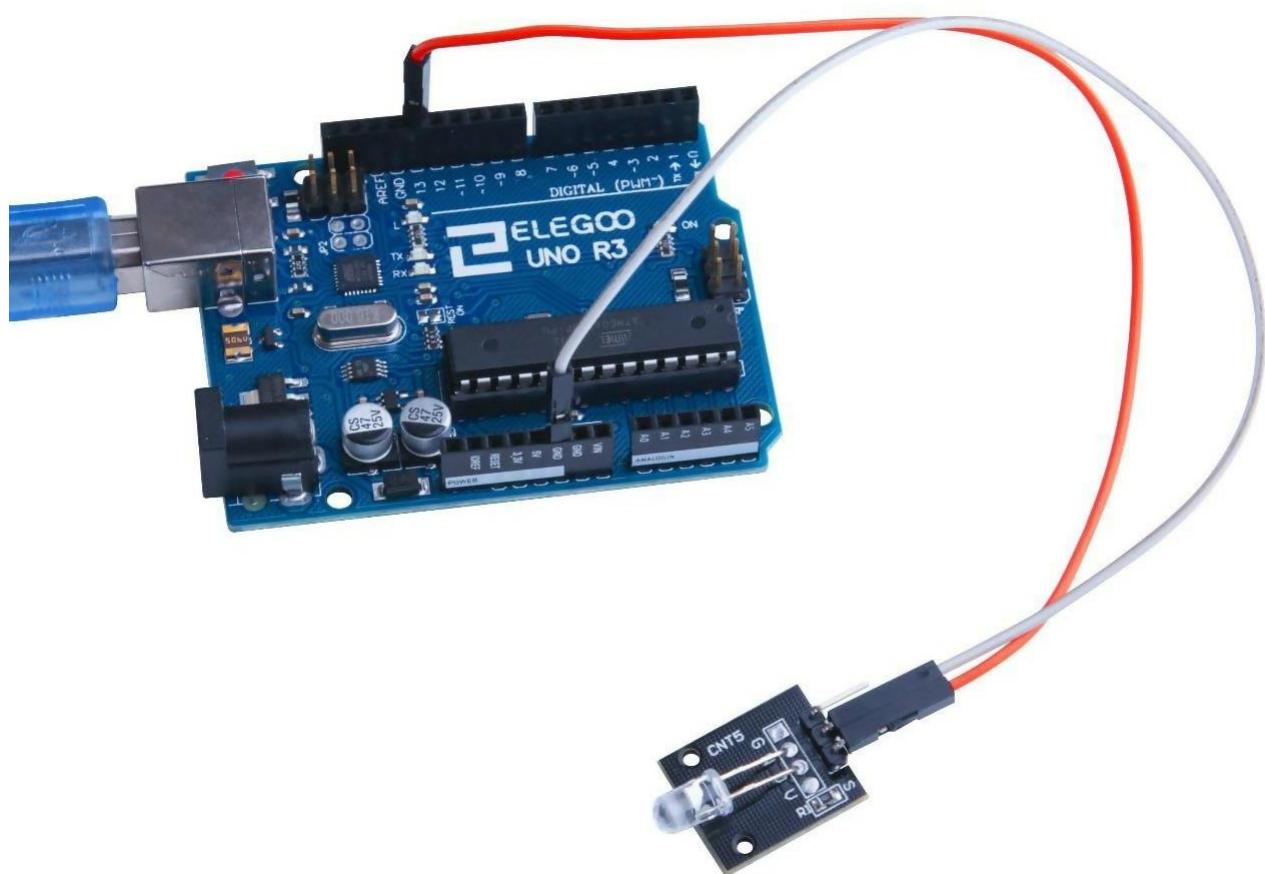
Diagrama de cableado



Código

Después de conectar el circuito, abrimos la carpeta "code" en el tutorial y buscamos la carpeta "Lección 22 Módulo de LED Flash de Siete Colores" para abrir el programa. Podrás ver nuestro fenomenal y colorido LED

Imagen ejemplo



A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
void setup()
{
// inicializar el pin digital como una salida.
// El pin 13 se conecta a un LED en la mayoría de las tarjetas Arduino: pinMode(13, OUTPUT);
}

void loop()
{

digitalWrite(13, HIGH);

}
```

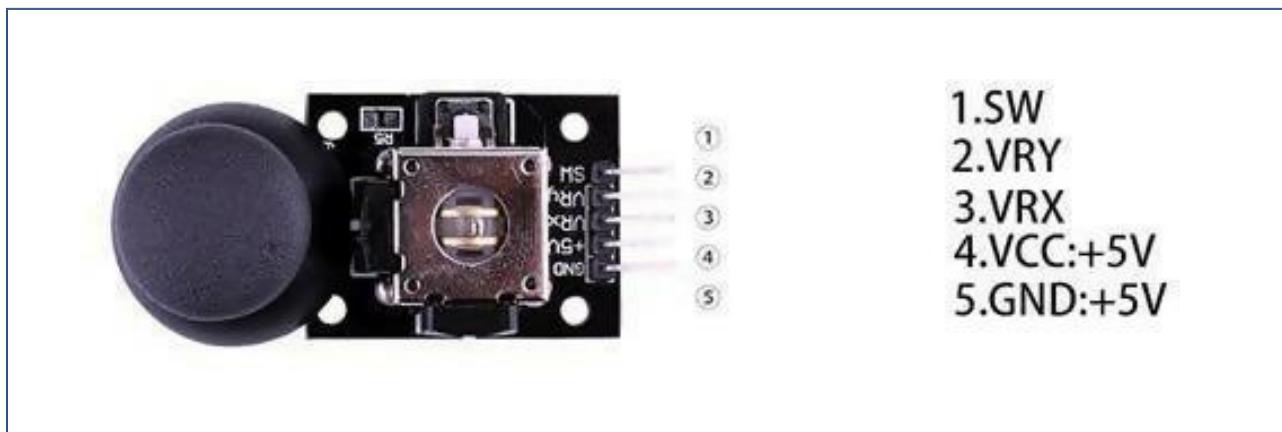
Lección 23 Módulo de Palanca de Mando

Resumen

Funciona tal como una palanca de mando en un juego de consola. Puedes controlar las dimensiones x, y, z con este módulo. Pueden ser considerado una combinación de potenciómetros con un botón. Las dimensiones x, y son señales analógicas y la dimensión z es una entrada digital. Por tanto los puertos x, y van conectados con los pines analógicos del Shield de sensores y el puerto z con el pin digital.

Módulo de palanca de mando

Una palanca de mando de 2 ejes con 2x potenciómetros de 10K ohm y un botón pulsador. Las descripciones de los pines están impresas en el PCB. Con el módulo viene una leva de operación incluida.



Componentes Requeridos:

- 1 x Elegoo Uno R3 1 x cable USB
- 1 x Módulo de palanca de mando 5 x Cables F-M

Introducción de Componente sensor de palanca de mando

Muchos robots necesitan utilizar palancas de mando. Este módulo constituye una solución económica para esos casos. Con solo conectarte a dos entradas analógicas, el robot está listo para que lo operes en los ejes x, y. También posee un interruptor que se conecta a un pin digital. Este módulo de palanca de mando puede conectarse de manera sencilla al Shields IO de Arduino. Se usa para un Arduino con sus cables respectivos.

Especificaciones

Voltaje de alimentación: 3.3V a 5V

Interfaz: Analógica x2, Digital x1 Tamaño: 40*28mm Peso: 12g

El módulo tiene 5 pines: Vcc, Ground, X, Y, Key. Es probable que tu etiqueta diga algo distinto, dependiendo de dónde la compraste. La palanca de mando es analógica y provee lecturas más precisas que otras simples palancas direccionales que utilizan otro tipo de botones o interruptores mecánicos. Adicionalmente, puedes presionar la palanca de mando hacia abajo (quizás algo fuerte) para activar un pulsador 'press to select'.

Para leer los datos de los pines X/Y tenemos que usar entradas analógicas del Arduino y para los botones necesitamos entradas digitales. El pin Key se conecta a tierra cuando la palanca de mando se presiona y queda flotando si no. Para obtener lecturas estables de el pin Key / Select, se necesita conectarlo a Vcc por medio de un resistor pull up. Se pueden utilizar los resistores incorporados en los pines digitales del Arduino para tal fin. Para un tutorial de como activar los resistores pull up desde los pines de Arduino, configurados como entradas

Esquema de Conexión

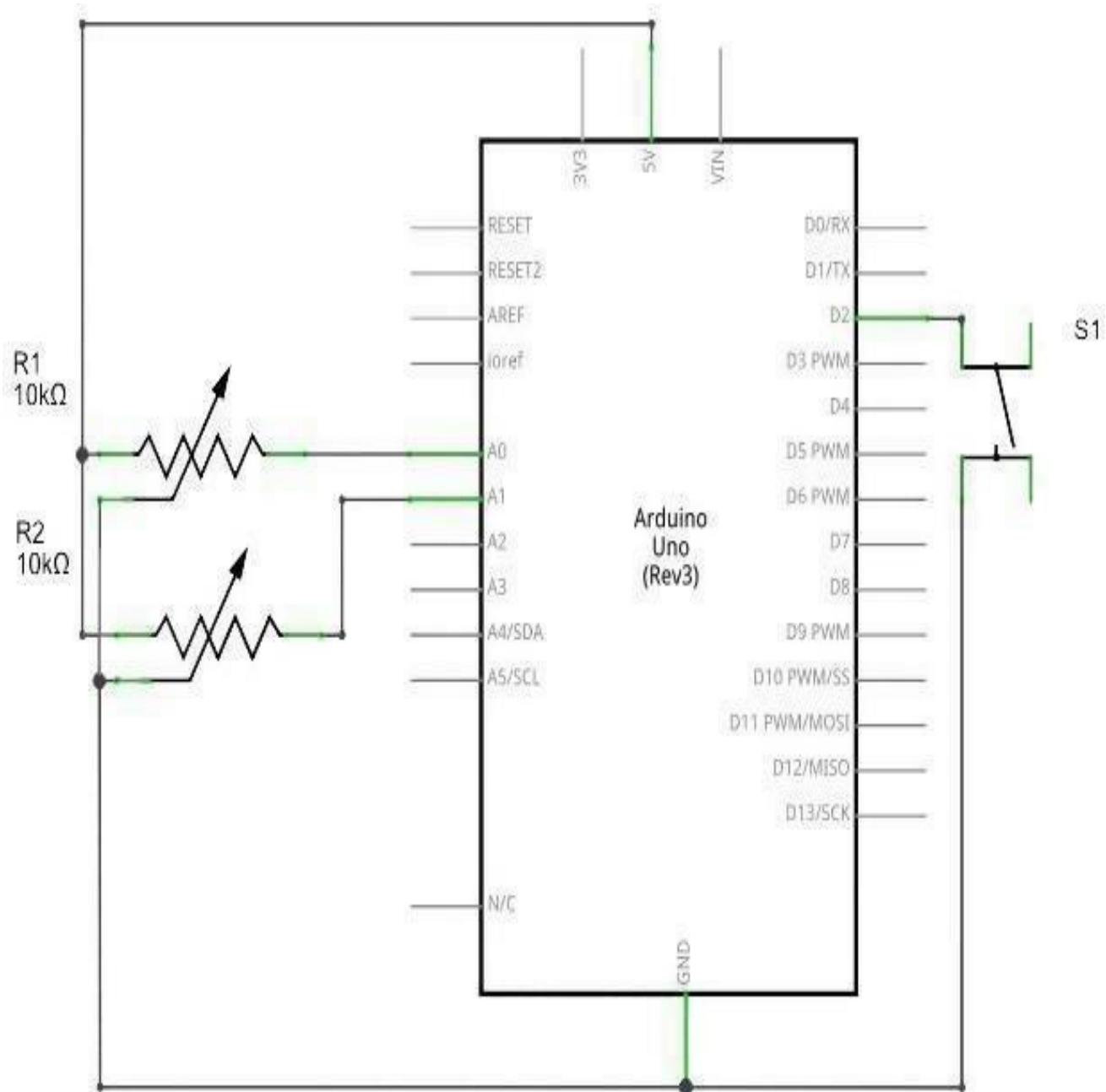
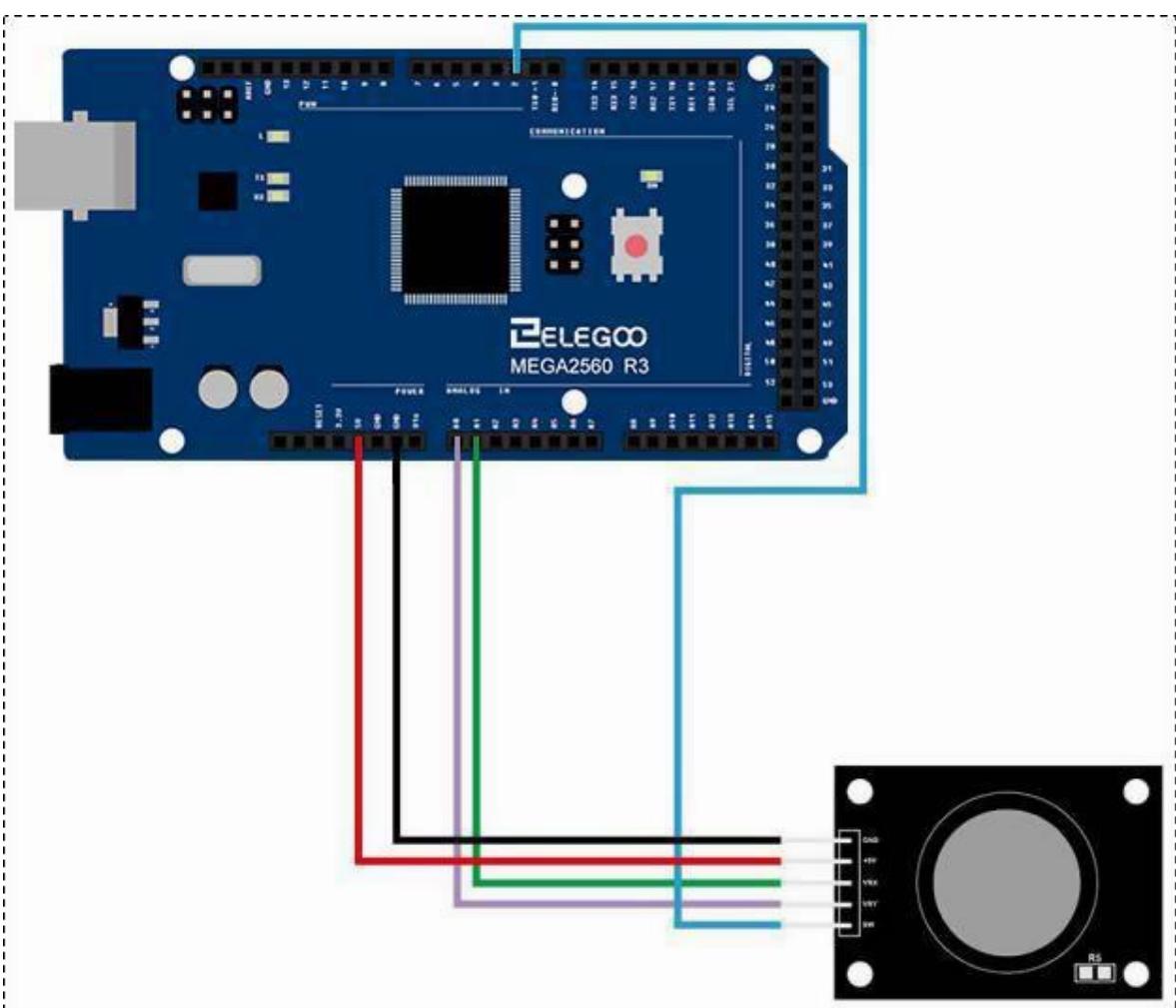
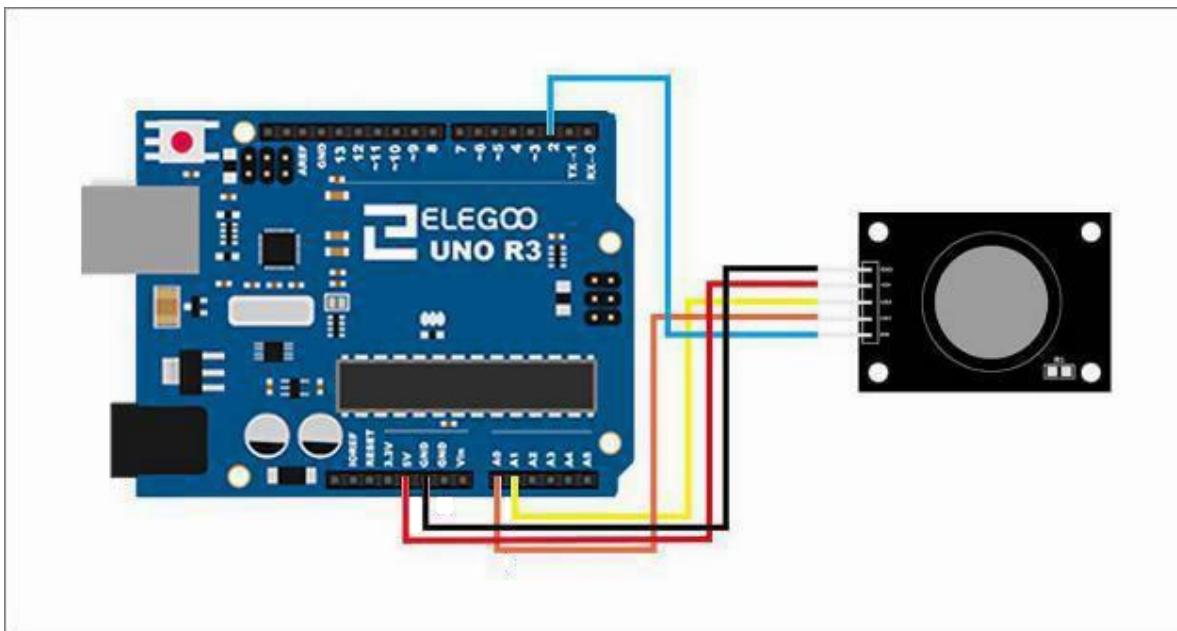


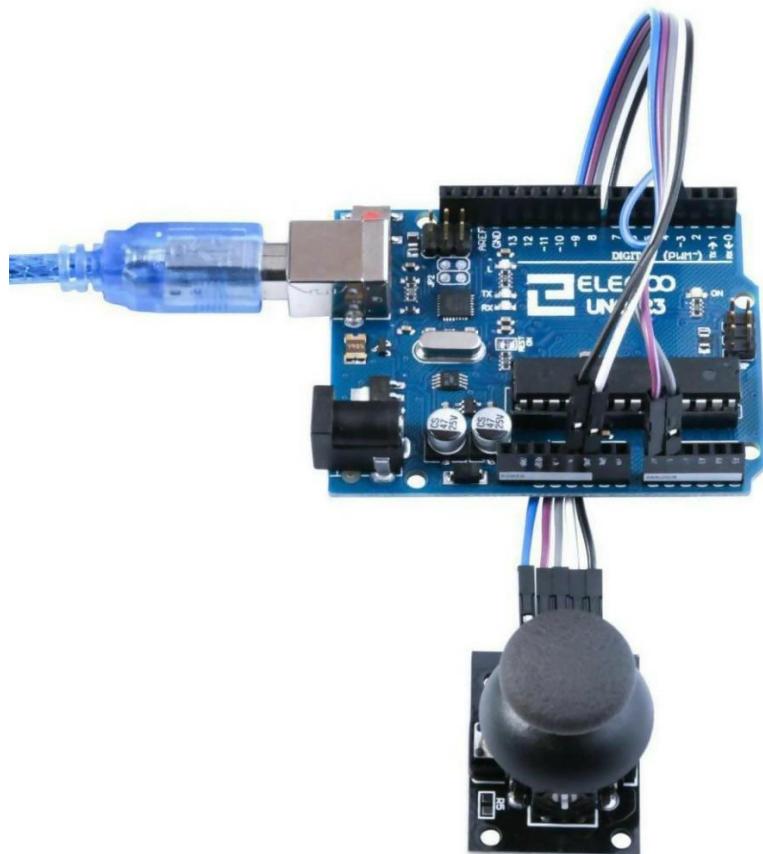
Diagrama de cableado

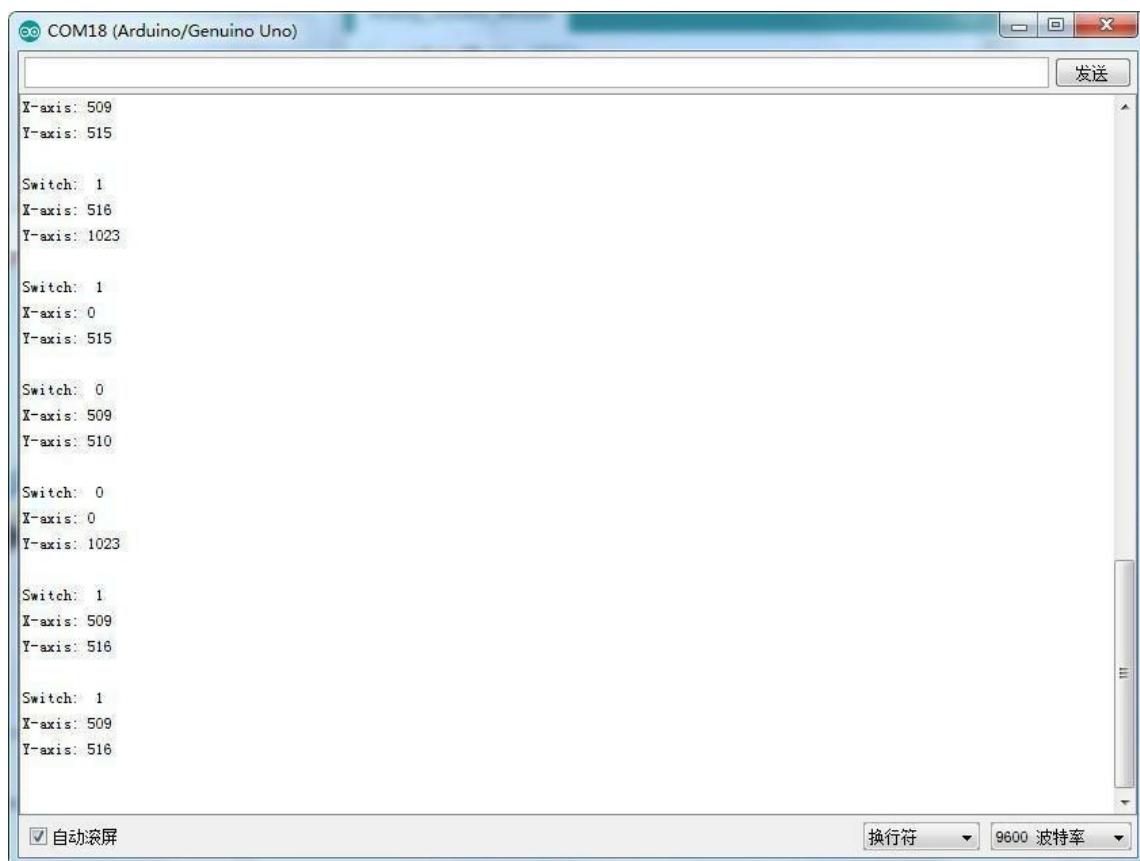


Código

Las palancas de mando son básicamente potenciómetros, por lo que retornan valores analógicos. Cuando la palanca de mando está en reposo o en una posición neutral, debería retornar un valor de 512, con valores en el rango entre 0 a 1023. Además; SW (eje Z) es una entrada digital, conectada a un puerto digital, y habilita el resistor pull up. Valores del SW: 1 significa no presionado, 0 significa presionado
Después de conectar el circuito, abrimos la carpeta "code" en el tutorial y buscamos la carpeta "Lección 23 Módulo de palanca de mando" para abrir el programa y ver los siguientes datos.

Imagen ejemplo





A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
// Números de pin Arduino  
// Pin digital conectado a la salida del interruptor const int SW_pin = 2;  
// Pin analógico conectado a la salida X const int X_pin = A1;  
// Pin analógico conectado a la salida Y const int Y_pin = A0;  
  
void setup() { pinMode(SW_pin, INPUT); digitalWrite(SW_pin, HIGH); Serial.begin(9600);  
}  
  
void loop() { Serial.print("Switch: ");  
Serial.print(digitalRead(SW_pin)); Serial.print("\n");  
Serial.print("X-axis: "); Serial.print(analogRead(X_pin)); Serial.print("\n");  
Serial.print("Y-axis: "); Serial.println(analogRead(Y_pin));  
Serial.print("\n"); delay(500);  
}
```

Lección 24 Módulo Seguidor de Líneas

En este experimento, aprenderemos a utilizar el módulo de rastreo y evasión.

El sensor infrarrojo de evasión de obstáculos está diseñado para ser utilizado en un robot con ruedas con distancia de evasión ajustable. Este sensor de luz ambiental adaptable y de alta precisión cuenta con un elemento transmisor y un receptor. Durante su operación normal, el elemento transmisor emite una luz infrarroja de cierta frecuencia a su alrededor que rebota en los obstáculos y es captada por el elemento receptor. Cuando eso sucede, se enciende un indicador luminoso. La distancia de detección puede ser ajustada a través de un potenciómetro, en un rango de entre 2 a 40 cm y con un voltaje de operación de 3.3-5V. El módulo es capaz de trabajar con una amplia variedad de microcontroladores, tales como Arduino, controlador BS2, etc. para aplicarse en robots con la finalidad de censar los cambios en sus alrededores.

Módulo de seguidor de líneas

Interruptor de reflexión de luz IR, útil para evasión de obstáculos y seguimiento de líneas en modelos que se mueven en el suelo. Un obstáculo en frente de los diodos emisores / receptores causará que el pin de salida se active en negativo. La sensibilidad del circuito se ajusta mediante un potenciómetro. La distancia de detección puede ser de hasta aproximadamente 1 cm.



1.OUTPUT
2.VCC: 3.3V-5V DC
3.GND:ground

Componentes Requeridos:

1x Elegoo Uno R3 1x cable USB

1x Módulo seguidor de líneas 3x Cables F-M

Introducción de Componente

Esquema de conexión

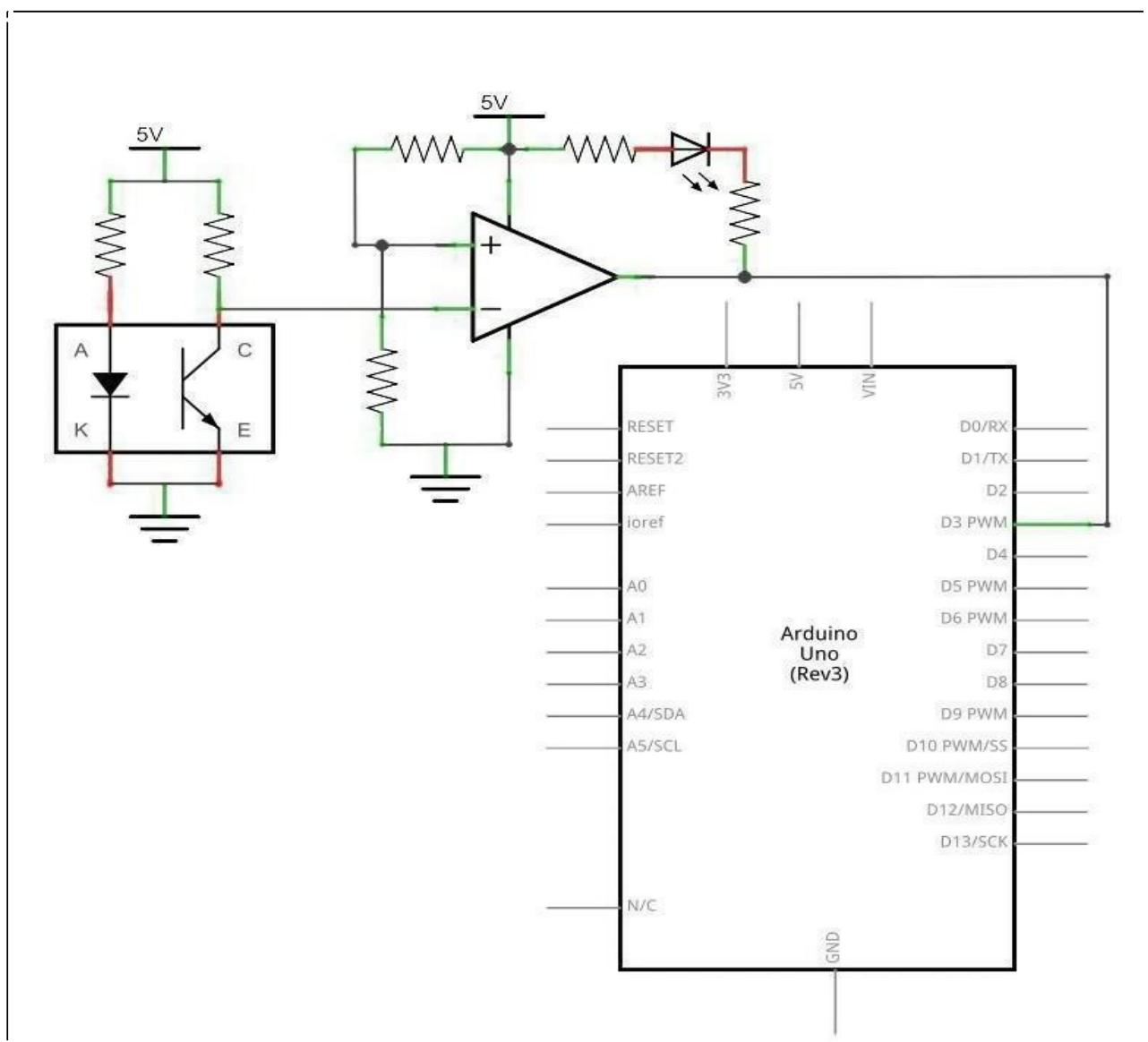
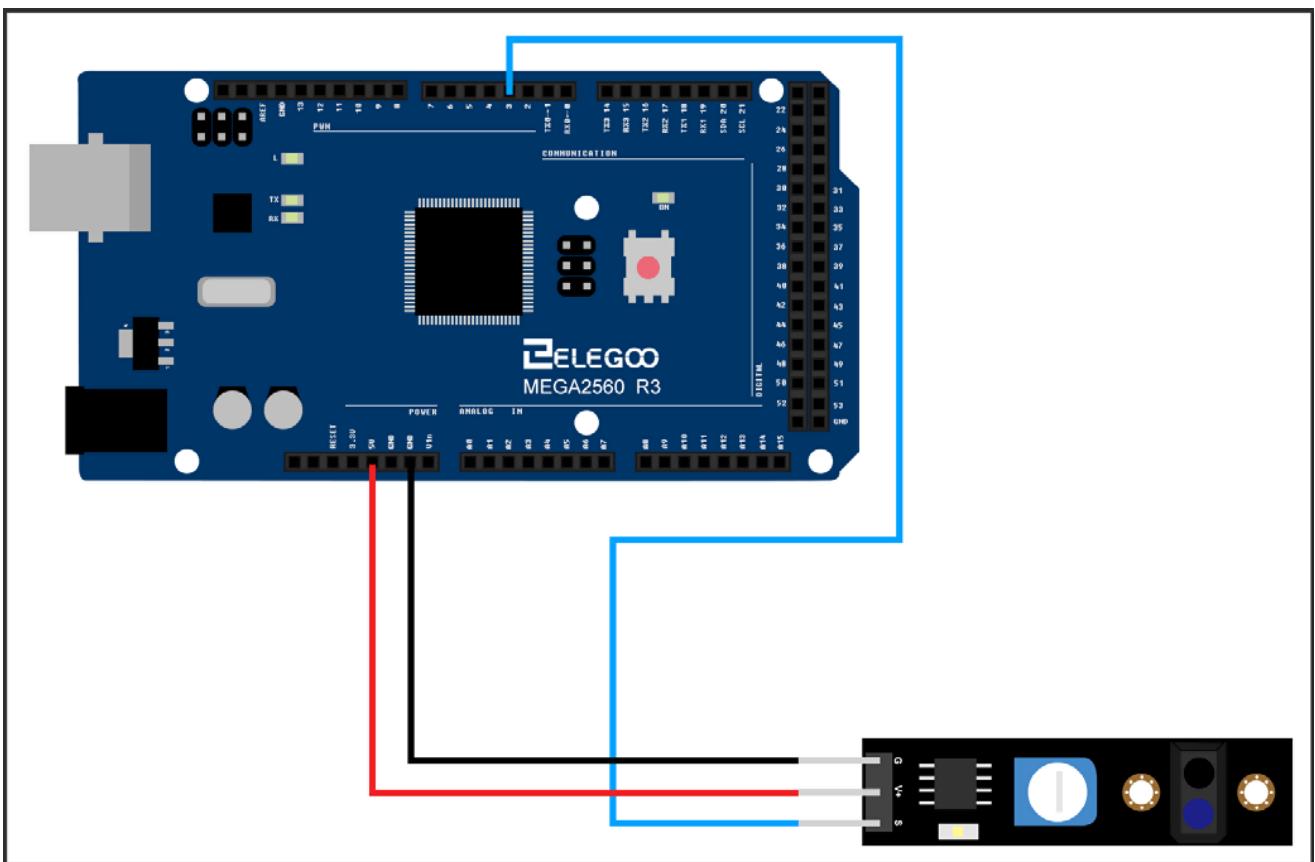
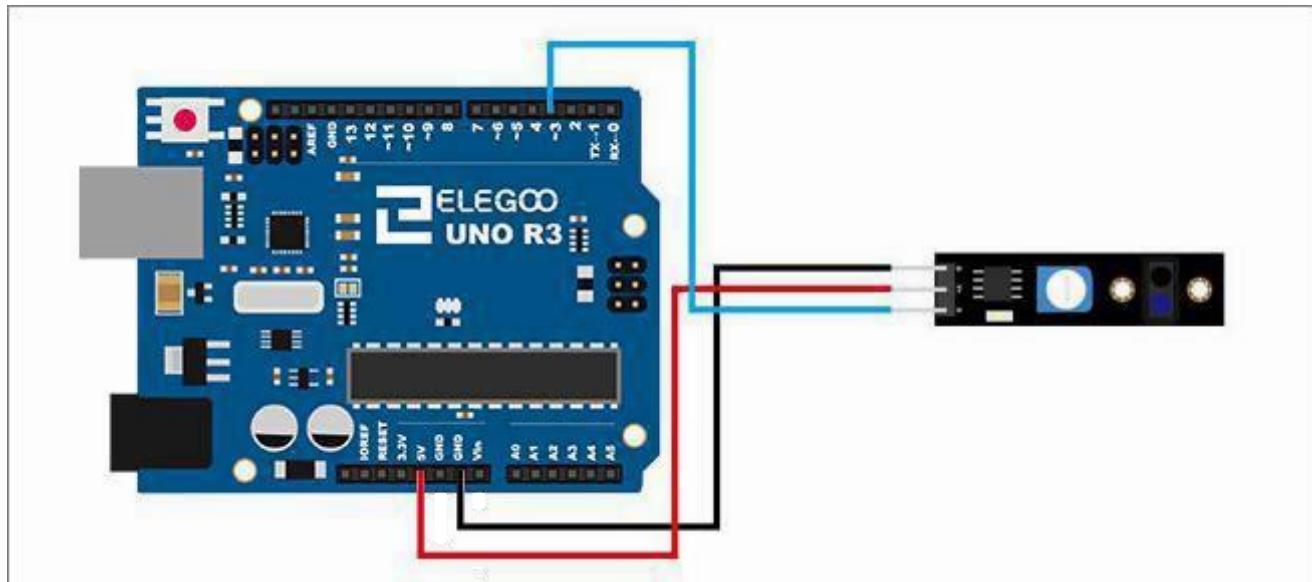


Diagrama de cableado

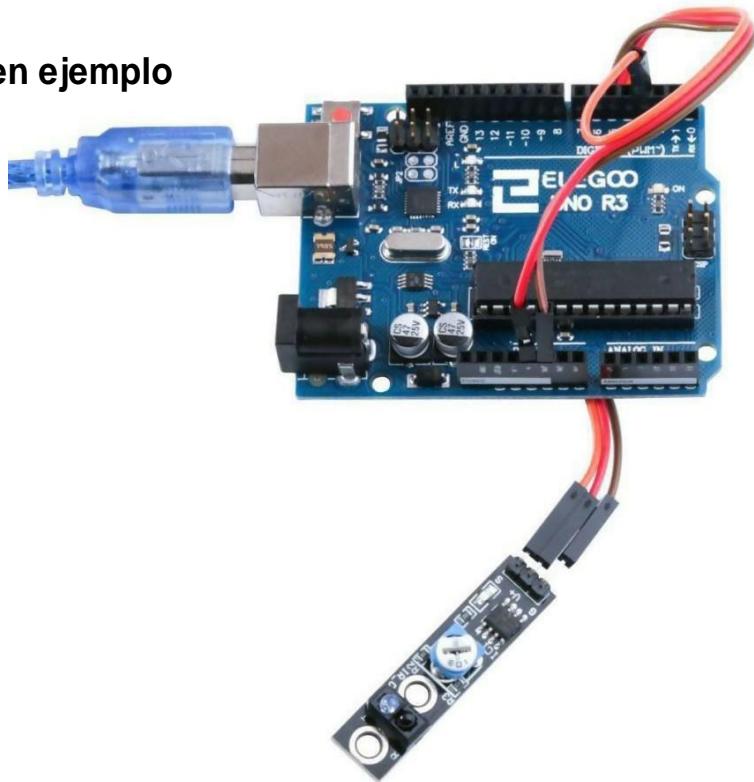


Código

Después de conectar el circuito, abrimos la carpeta "code" en el tutorial y buscamos la carpeta "Lección 24 Módulo de seguidor de línea" abrimos el programa, corremos el compilador y esperamos que se encienda el indicador L. Es normal que dicho LED este iluminado cuando no hay nada obstruyéndolo. El módulo se comporta de acuerdo con los principios de la óptica infrarroja. Cuando el LED ilumina un objeto negro, el receptor recibe una señal de rebote de baja intensidad, comparable a la que recibe cuando no tiene obstáculos en frente. Por el contrario, al tener un objeto blanco en frente, sucede el fenómeno contrario, ya que se recibe una señal de alta intensidad que la electrónica interpreta como un obstáculo y la luz del módulo se apaga.

Como se puede apreciar la distancia de detección se ajusta con un potenciómetro; si este se gira en el sentido de las agujas del reloj la distancia se hace más pequeña y si se gira al contrario, se hace más grande.

Imagen ejemplo



A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
//define el puerto del LED int Led = 13;  
//define el puerto del interruptor int buttonpin = 13;  
//define la variable digital val int val;  
void setup()  
{  
//define el LED como una salida PinMode (Led, OUTPUT);  
//define switch as a output port pinMode(buttonpin,INPUT); Serial.begin(9600);  
}  
void loop()  
{  
//Lee el valor de la interfaz digital 3 asignada a val val=digitalRead(buttonpin);  
// cuando el sensor del interruptor tenga señal, el LED destellará if(val==HIGH)  
{  
digitalWrite(Led,HIGH); Serial.println("HIGH!");  
}  
else  
{  
digitalWrite(Led,LOW); Serial.println("LOW!");  
}  
}
```

Lección 25 Módulo Sensor para Evasión de Obstáculos

Resumen

Non logic chip oscillation frequency regulation 38KHz detection circuit. Este módulo para evasión de obstáculos infrarrojo de 38KHZ es mucho mejor que la tradicional detección de obstáculos fotoeléctrica y posee una salida sencilla de contactos normalmente abiertos o cerrados.

Módulo Sensor para Evasión de Obstáculos

Sensor de reflexión IR, útil para aplicaciones de evasión de obstáculos. Un obstáculo en frente de los emisores / receptores IR causará que el pin de salida se active en negativo. La sensibilidad del circuito se puede ajustar con un potenciómetro. La distancia de detección puede ajustarse hasta aproximadamente 1 cm. Un Jumper de habilitación (EN) se coloca para operación continua. Si se quita ese jumper (en el pin EN) es posible controlar el detector con una señal externa (low = activa, high = off).



- 1.EN
- 2.OUTPUT
- 3.VCC: 3.3V-5V DC
- 4.GND:ground

Componentes Requeridos:

1x Elegoo Uno R3 1x cable USB

1x Módulo Sensor para Evasión de Obstáculos 4x Cables F-M

Introducción de Componente

Módulo de Evasión de obstáculos

Especificaciones

Voltaje de alimentación: 3.3V a 5V

Corriente de trabajo: $\geq 20\text{mA}$

Temperatura de operación: - 10 °C - +50 °C

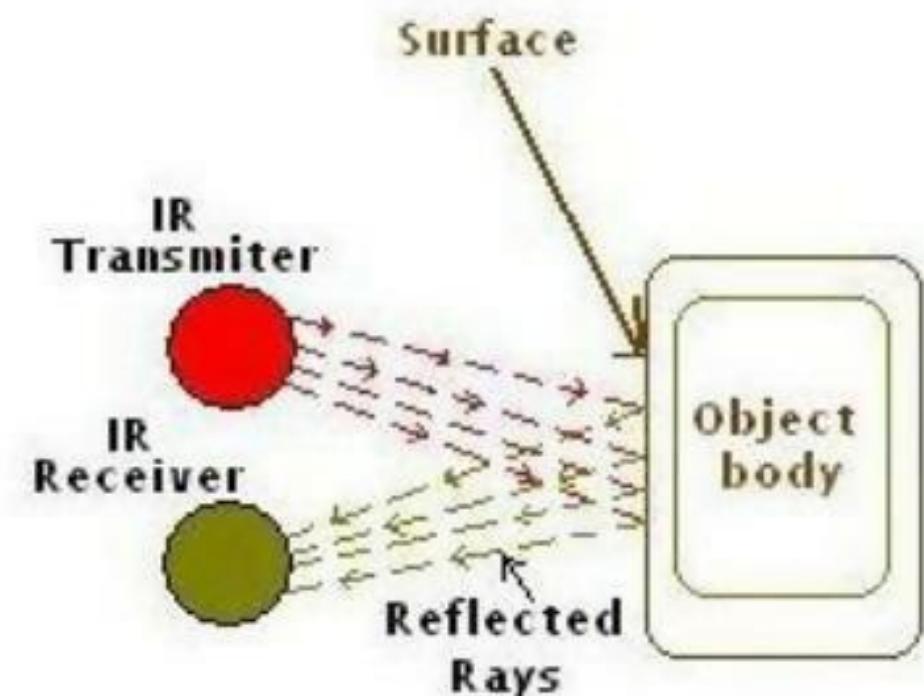
Distancia de detección: 2-40cm

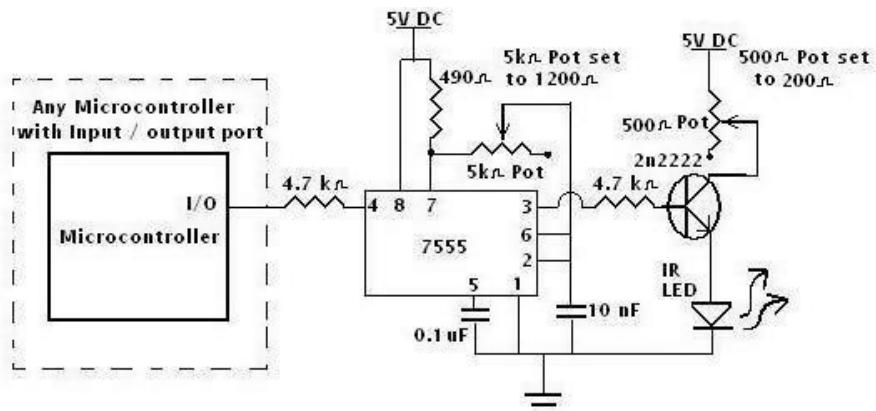
Interfaz IO: interfaces tipo 4-wire (- / + / S /EN)

Señal de salida: Nivel TTL (si hay obstáculos nivel bajo, si no los hay nivel alto)

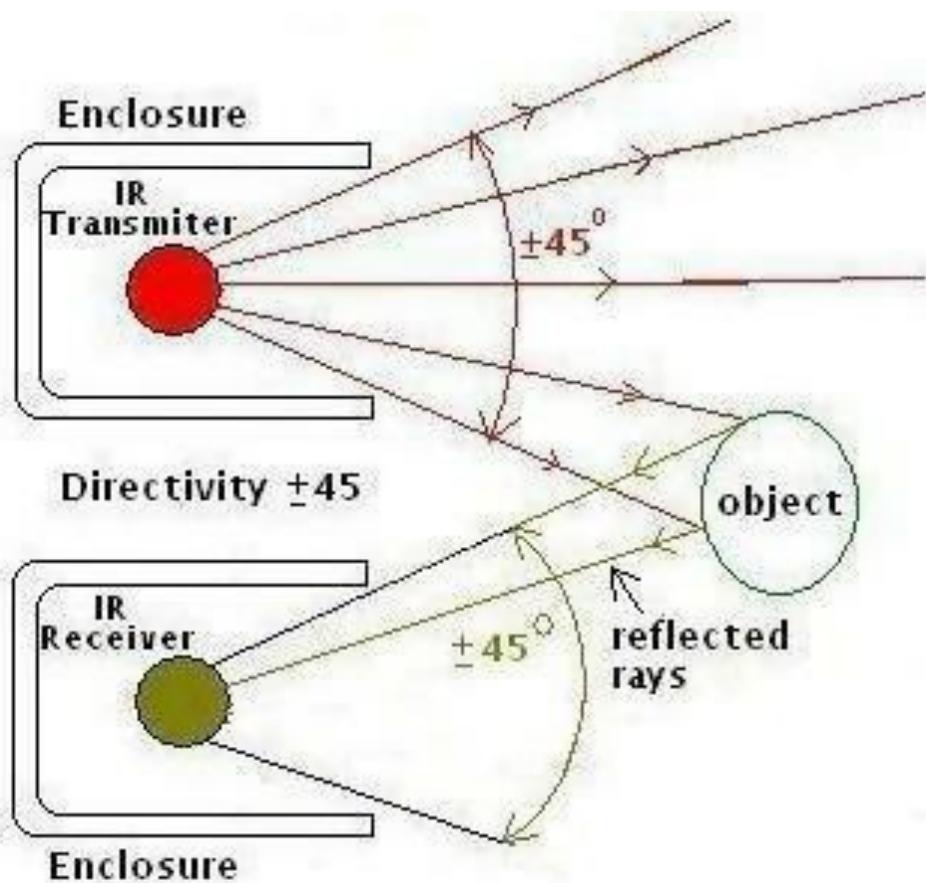
Ajuste: Resistencia de ajuste multi vueltas Ángulo efectivo: 35 ° Tamaño: 28mm × 23mm Peso: 9g

El concepto básico de la detección de obstáculos IR es transmitir una señal infrarroja (de radiación) en una dirección determinada y esperar a recibir el rebote de la misma en su receptor de IR al detectar un objeto



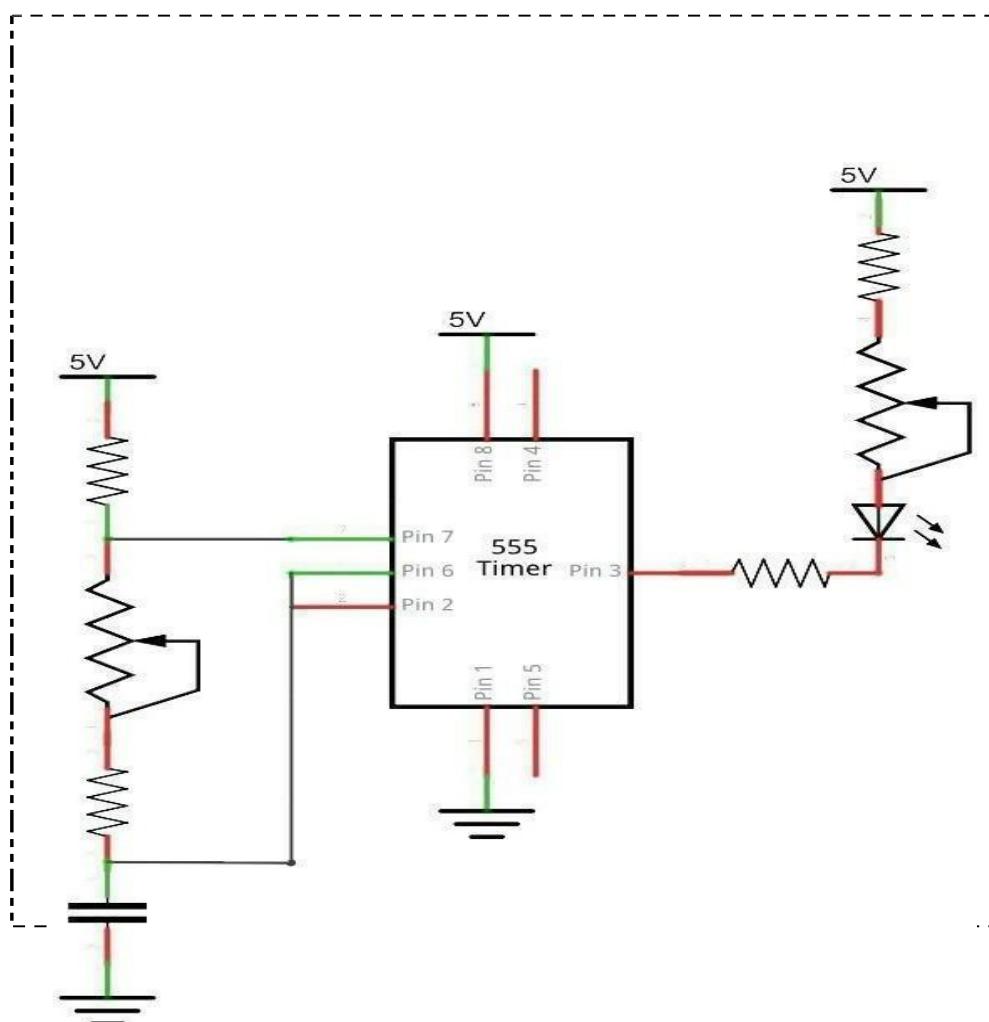


Hay dos potenciómetros en el módulo, uno para controlar la frecuencia de operación (centrado en 38 kHz) y el otro para controlar la intensidad. El detector se diseñó para 38 kHz y el oscilador que forma parte del circuito se basa en un temporizador 555. Se puede jugar con los ajustes para mejor rango, pero la sugerencia es dejarlo como viene de fábrica ya que el rango es bastante pequeño.



El sensor infrarrojo de evasión de obstáculos está diseñado para ser utilizado en un robot con ruedas con distancia de evasión ajustable. Este sensor de luz ambiental adaptable y de alta precisión cuenta con un elemento transmisor y un receptor. Durante su operación normal, el elemento transmisor emite una luz infrarroja de cierta frecuencia a su alrededor infrared receiver tube que rebota en los obstáculos y es captada por el elemento receptor. Cuando eso sucede, se enciende un indicador luminoso. La distancia de detección puede ser ajustada a través de un potenciómetro, en un rango de entre 2 a 40 cm y con un voltaje de operación de 3.3-5V.

Esquema de conexión



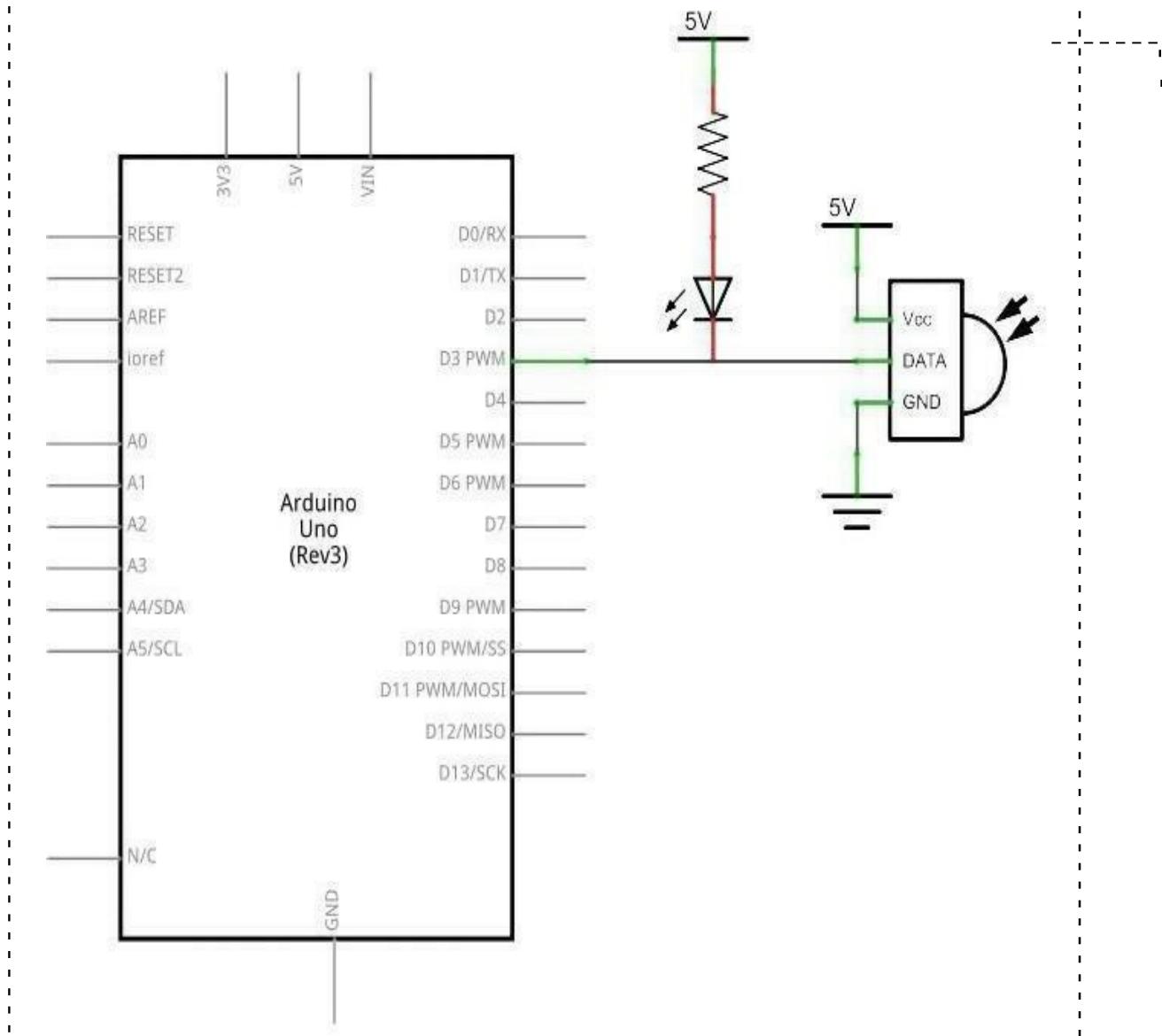
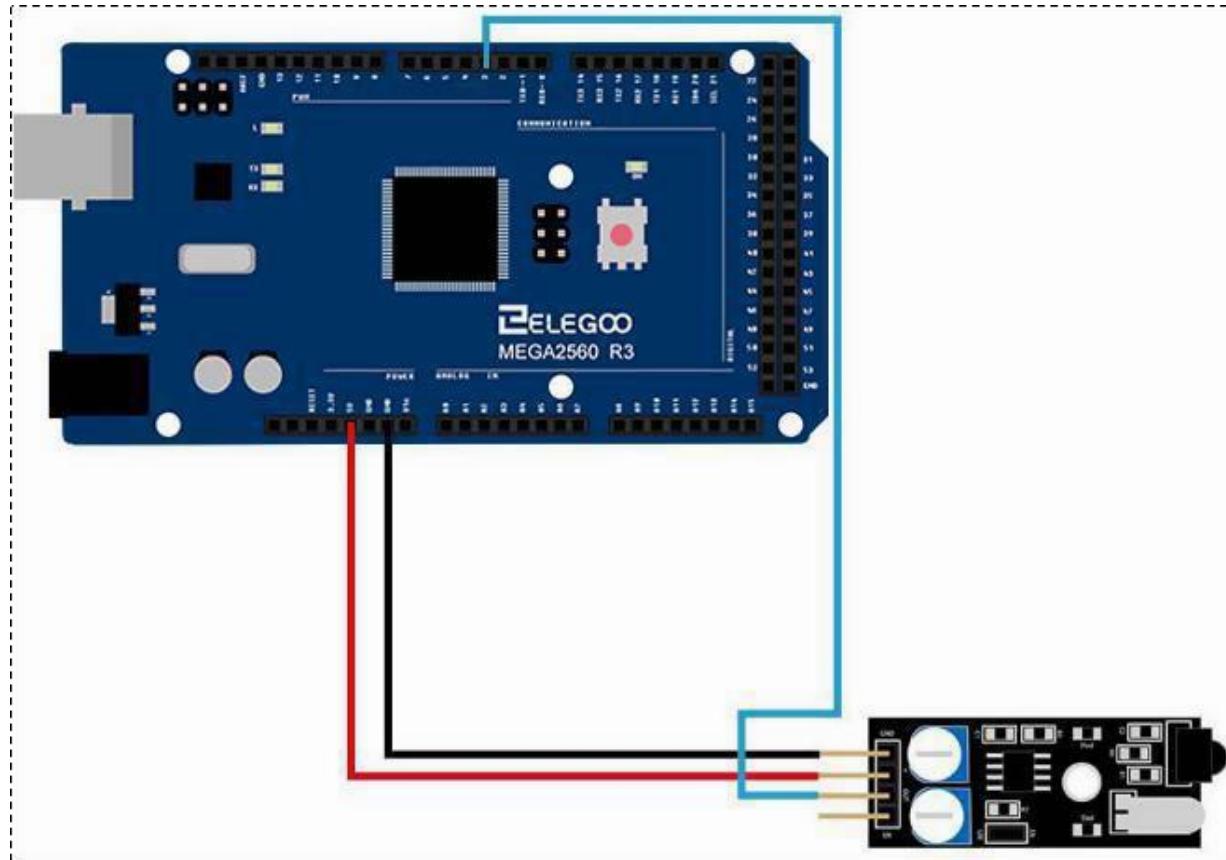
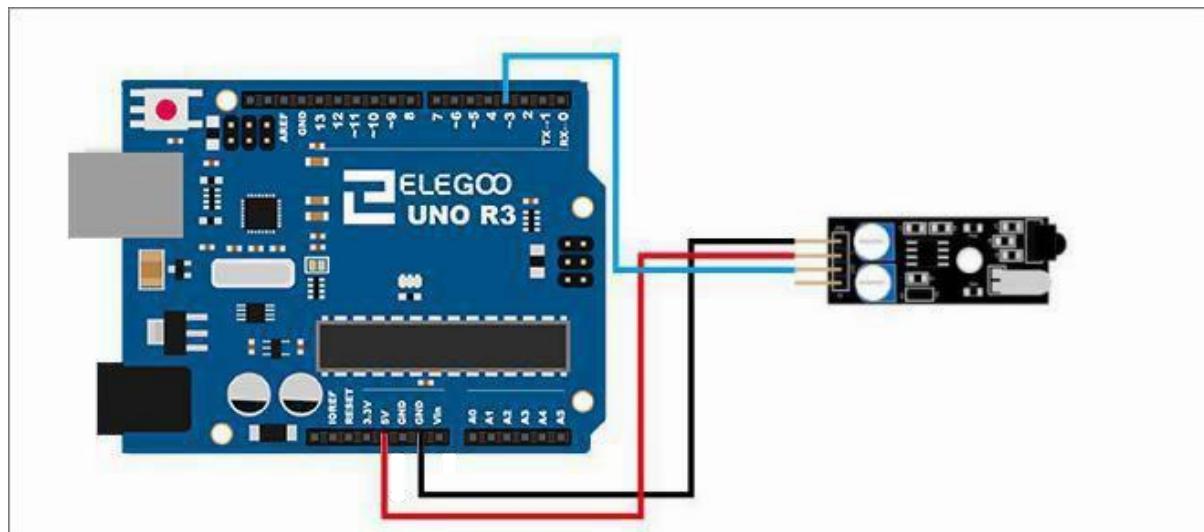


Diagrama de cableado



Código

Después de conectar el circuito, abrimos la carpeta "code" en el tutorial y buscamos la carpeta "Lección 25 Módulo de Sensor Evasión de Obstáculos" para abrir el programa. Intenta tapar el frente del módulo con la mano y verás lo que sucede.

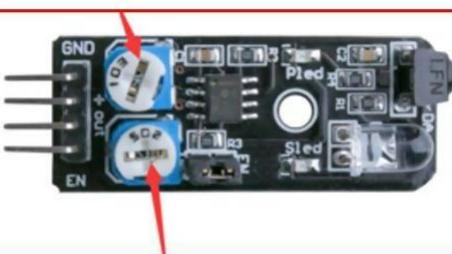
Imagen ejemplo



Aquí utilizamos el módulo sensor para evasión de obstáculos y una interfaz digital, junto al LED integrado número 13 y un pequeño circuito para construir una lámpara de advertencia para evasión. Cuando el sensor detecta una señal el LED se ilumina y de lo contrario se apaga.

Es necesario ajustar los potenciómetros de 5K y 10K antes de usar el módulo de evasión de obstáculos. Por lo general, el mejor funcionamiento se obtiene girando el potenciómetro de 5K 90 grados en el sentido de las agujas del reloj y el de 10K entre 10 y 30 grados. Te recomendamos armarte de paciencia para hacerlo funcionar.

10K potentiometers: Ajust it clockwise at a angle between 10 to 30 degrees.



5K potentiometers: Ajust it clockwise at a angle about 90 degrees.

A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
//define el puerto del LED int Led = 13;  
//define el puerto del interruptor int buttonpin = 13;  
//define la variable digital val int val;  
void setup()  
{  
    //define el LED como una salida PinMode (Led, OUTPUT);  
    //define el interruptor como una salida pinMode(buttonpin,INPUT);  
}  
void loop()  
{  
    //Lee el valor de la interfaz digital 3 asignada a val val=digitalRead(buttonpin);  
    // cuando el sensor del interruptor tenga señal, el LED destellará if(val==HIGH)  
    {  
        digitalWrite(Led,HIGH);  
    }  
    else  
    {  
        digitalWrite(Led,LOW);  
    }  
}
```

Lección 26 Módulo de Encoder Rotativo

Resumen

En este experimento, aprenderemos a utilizar el Módulo de Encoder Rotativo.

Encoder rotativo

Este encoder rotativo es ideal para hacer un potenciómetro electrónico, etc. Las descripciones de los pines están impresas en el PCB



- 1.CLK
- 2.DT
- 3.SW
- 4.VCC:3.3V-5V DC
- 5.GND:ground

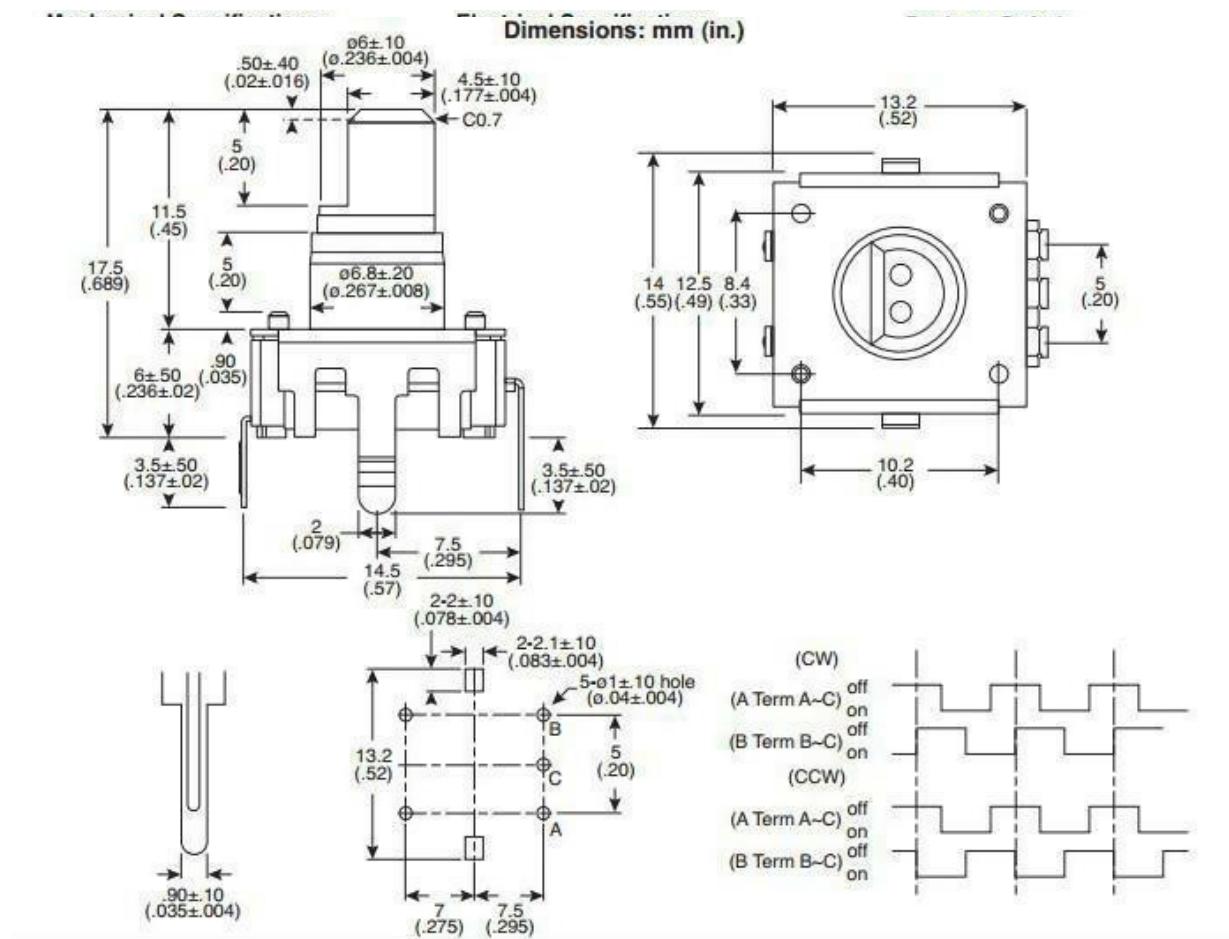
Componentes Requeridos:

1x Elegoo Uno R3 1x cable USB

1x Módulo de Encoder Rotativo 5x Cables F-M

Introducción de Componente

Encoder Rotativo



Principio

Encoder Incremental

Los encoder incrementales entregan una señal de onda cuadrada de dos fases, con una diferencia de 90 grados en sí, llamadas comúnmente canales A y B. Uno de los dos canales tiene información concerniente a la velocidad y al mismo tiempo, se comparan ambos canales de forma de obtener la información de la dirección de rotación del elemento. También poseen una señal especial denominada Z o canal Z, que da la referencia de la posición cero del encoder. Generalmente es otra onda cuadrada que coincide con la línea central del canal A.

En sentido de las agujas del reloj en contra de las agujas del reloj

| A | B |
|---|---|
| 1 | 1 |
| 0 | 1 |
| 0 | 0 |
| 1 | 0 |
| 1 | 1 |
| 1 | 0 |
| 0 | 0 |
| 0 | 1 |

La precisión de los encoders incrementales depende de dos factores, los cuales son: errores de indexado de tramas, excentricidad de disco, excentricidad de rodamientos, más otras fuentes diversas de errores e imprecisiones ópticas. La resolución del encoder se mide en grados y su precisión depende del pulso del mismo. Los grados eléctricos de seguimiento comprenden 360 grados de rotación del eje de una máquina determinada. Para saber el equivalente eléctrico del ángulo mecánico de 360 grados, se puede utilizar la siguiente fórmula: Electrical 360 = Machine 360 ° / n ° pulses /revolution

El error de indexado es el ángulo eléctrico de la unidad representada por la diferencia de dos pulsos máximos. Cualquier encoder tiene errores causados por los factores arriba mencionados. El encoder Eltra tiene un error máximo de ± 25 grados eléctricos (en cualquier condición), equivalente al valor de Offset nominal $\pm 7\%$, con una diferencia de fase de 90 ° (eléctricos) de los dos canales de desviación máxima \pm

35 grados eléctricos que son igual a $\pm 10\%$ de desviación

Esquema de Conexión

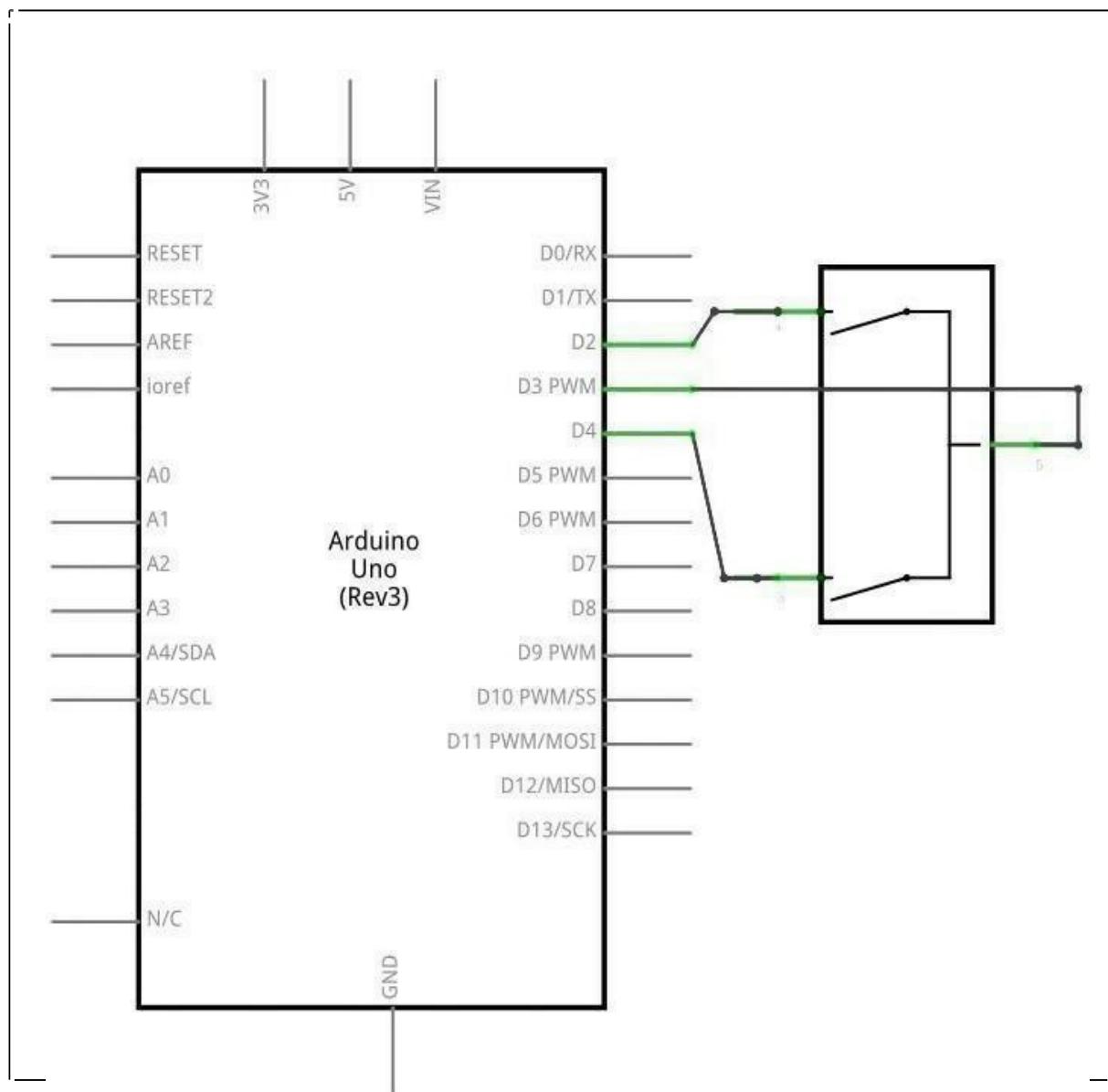
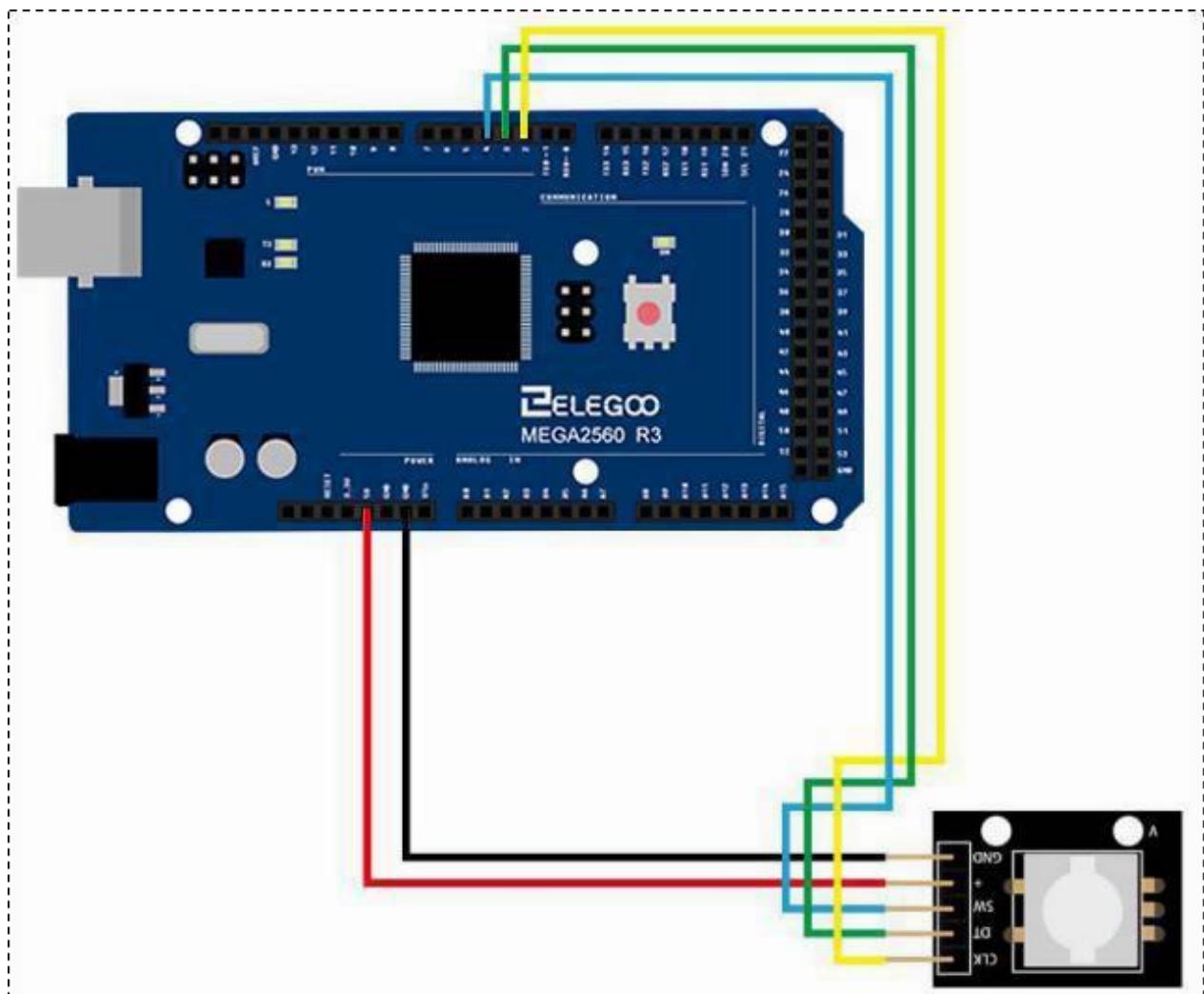
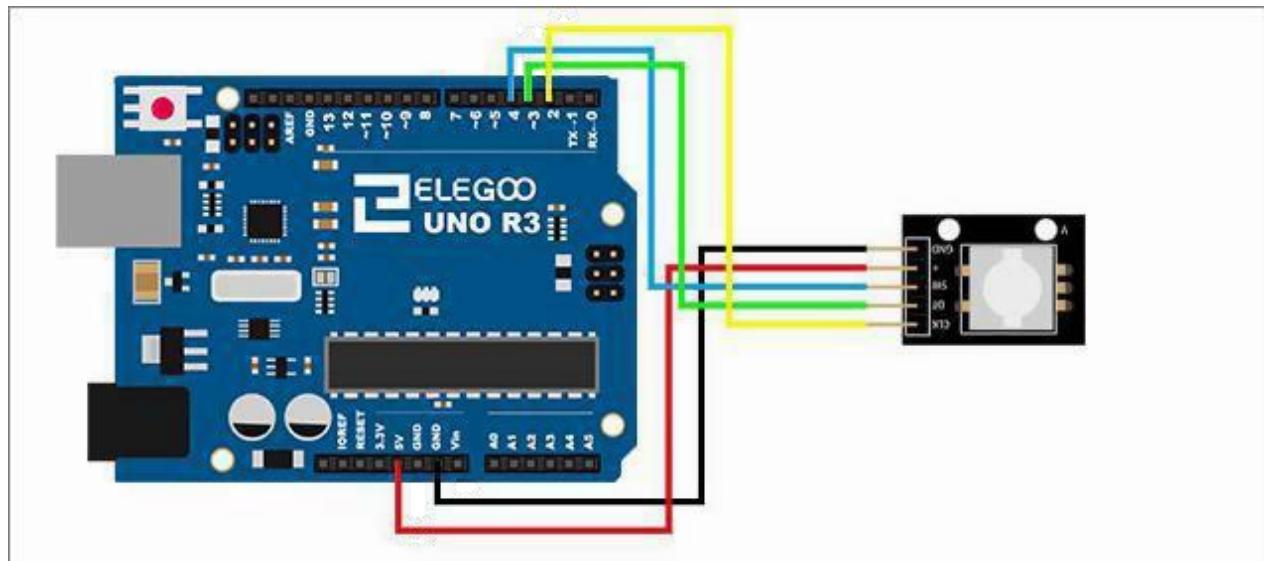


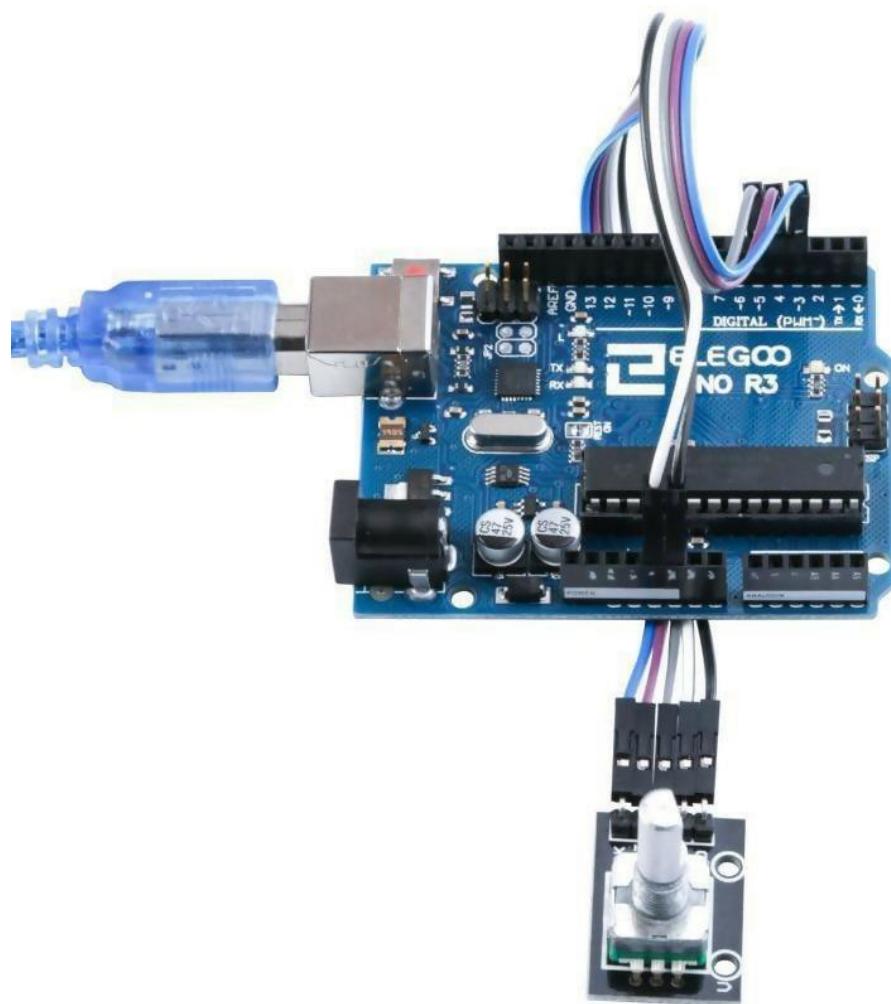
Diagrama de cableado

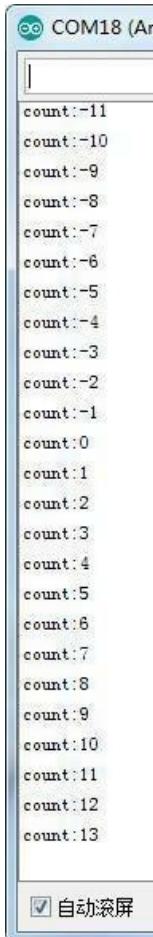


Código

Cuando termines de hacer el cableado, abre el tutorial y ve a code > Lección 26 Módulo de Encoder Rotativos y corre el programa (el archivo .ino). Luego ve al monitor serial y mueve el encoder; verás los siguientes datos:

Imagen ejemplo





```
ee COM18 (Ar)
|
count:-11
count:-10
count:-9
count:-8
count:-7
count:-6
count:-5
count:-4
count:-3
count:-2
count:-1
count:0
count:1
count:2
count:3
count:4
count:5
count:6
count:7
count:8
count:9
count:10
count:11
count:12
count:13
```

自动滚屏

A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
//Define la conexión del pin int CLK = 2;//CLK->D2
int DT = 3;//DT->D3 int SW = 4;//SW->D4
// Interrupt 0 en pin 2 const int interrupt0 = 0;
//Define el conteo int count = 0;
//CLK initial value int lastCLK = 0; void setup()
{
    pinMode(SW, INPUT); digitalWrite(SW, HIGH); pinMode(CLK, INPUT); pinMode(DT, INPUT);
    //Configura el manejador del interruptor 0, nivel de cambio del disparador attachInterrupt(interrupt0,
    ClockChanged, CHANGE);

    Serial.begin(9600);
}

void loop()
{//Lee cuando se presiona el botón y coloca el valor del contador en 0 cuando se reinicia if
(!digitalRead(SW) && count != 0)
{
    count = 0; Serial.print("count:"); Serial.println(count);
}
}

//The interrupt handlers void ClockChanged()
{
    //Lee el nivel del pin CLK
    int clkValue = digitalRead(CLK);
    //Lee el nivel del pin DT
    int dtValue = digitalRead(DT); if (lastCLK != clkValue)
```

```
{  
  
lastCLK = clkValue;  
//CLK y DT inconsistente + 1, de otro modo - 1 conteo += (clkValue != dtValue ? 1 : -1);  
  
Serial.print("count:"); Serial.println(count);  
}  
}
```

Lección 27 Módulo de Relé

En este experimento, aprenderemos a utilizar el módulo de relé

Un relé es un tipo de componente que responde a un cambio de una variable de entrada basándose en parámetros específicos del circuito de entrada. El circuito de salida conmuta tan pronto como se cumplen las condiciones. Nuestra empresa produce módulos de relé en un rango que va desde los 28V hasta los 240V, tanto en AC como en DC. Se pueden usar en combinación con un MCU para crear un interruptor de control de tiempo. Los relés se usan en todas partes, ya sea para sistemas de alarmas, juguetes o en equipos de construcción. Constituyen dispositivos de control eléctrico con dos subsistemas: el circuito de entrada y el de salida, que interactúan para lograr la salida deseada.

Módulo de relé

Un módulo de relé adecuado para la conexión directa a una tarjeta Arduino. Este módulo requiere una fuente de poder de 5V. La señal de entrada de control recibe el nombre de 'S'. El relé posee un contacto que cambia su estado. Puede conmutar cargas resistivas de hasta 10 A 250 VAC y hasta 10 A 30 V máximo.



- 1.GND:ground
- 2.VCC:3.3V-5V DC
- 3.OUTPUT

Don't forget to provide interference suppression for the switched load!

Componentes Requeridos:

1x Elegoo Uno R3 1x cable USB

1x 1 Módulo de relé de 1 canales

3x Cables F-M

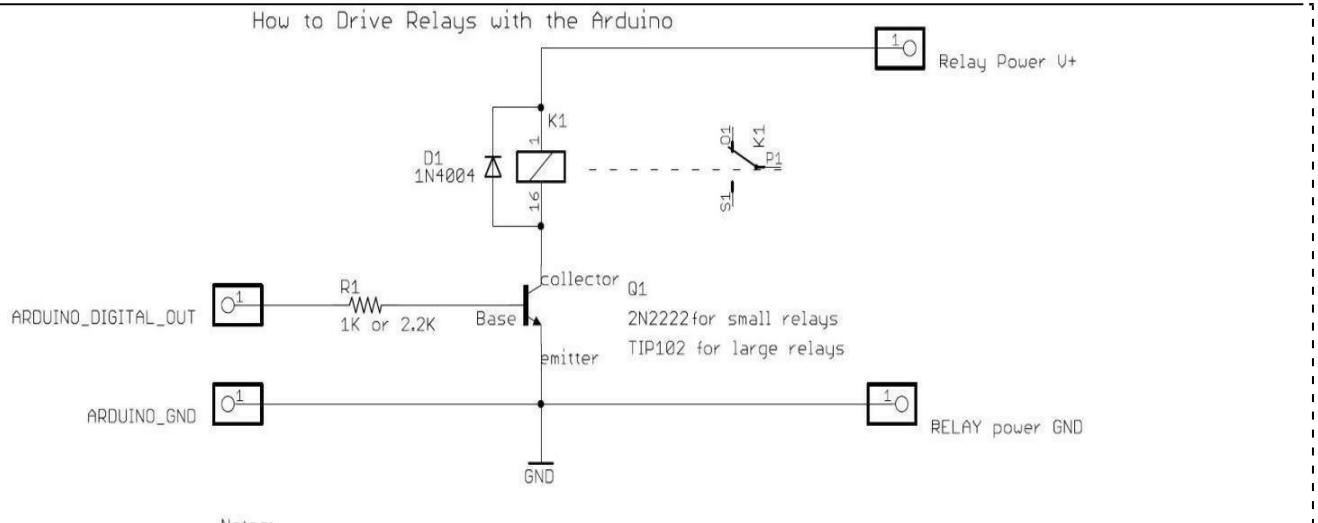
Introducción de componente

Relé:

Un relé es un interruptor operado eléctricamente. Muchos relés utilizan electro magnetos para operar mecánicamente un interruptor, pero también se usan otros principios de operación, como los relés de estado sólido. Los relés se usan cuando es necesario controlar un circuito mediante una señal de baja potencia (siempre conservando un aislamiento eléctrico completo entre los circuitos de control y potencia), o cuando se desean controlar varios circuitos con una sola señal. Los primeros relés se usaron en los circuitos de los telégrafos de larga distancia como amplificadores: Ellos repetían la señal proveniente de un circuito y la re-transmitían en el otro. Se usaban ampliamente en los Intercambios telefónicos y las computadoras de primera generación para realizar operaciones lógicas.

Un tipo de relé con la capacidad de manejar grandes potencias destinadas a controlar motores eléctricos y otras cargas es el contactor. Los relés de estado sólido controlan circuitos de potencia sin tener partes móviles, a través de dispositivos semiconductores que realizan la commutación. Relés con características de operación calibradas y con múltiples bobinas se utilizan para protección de circuitos eléctricos de sobrecargas o fallas; en los sistemas de potencia modernos se les denomina relés de protección.

Abajo se muestra el esquema de como operar un relé con Arduino (descarga de arduino.cc)



Notes:

- For a small 5V relay, use the 5v Arduino supply, and a 2N2222 transistor
- For larger relays, use an appropriate external power supply to drive the relay coil, and a TIP102 transistor
- This diagram is for DC relays

Esquema de Conexión

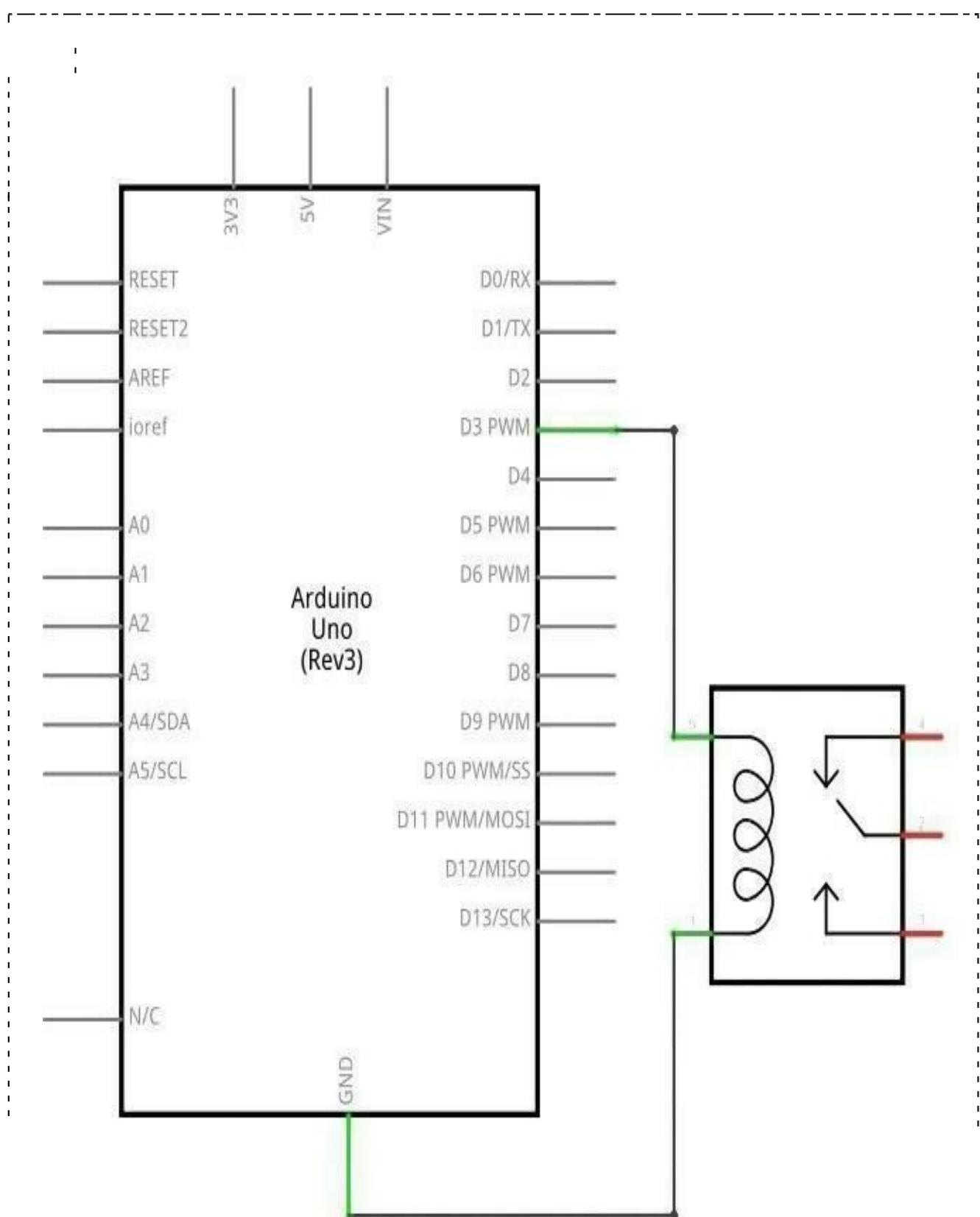
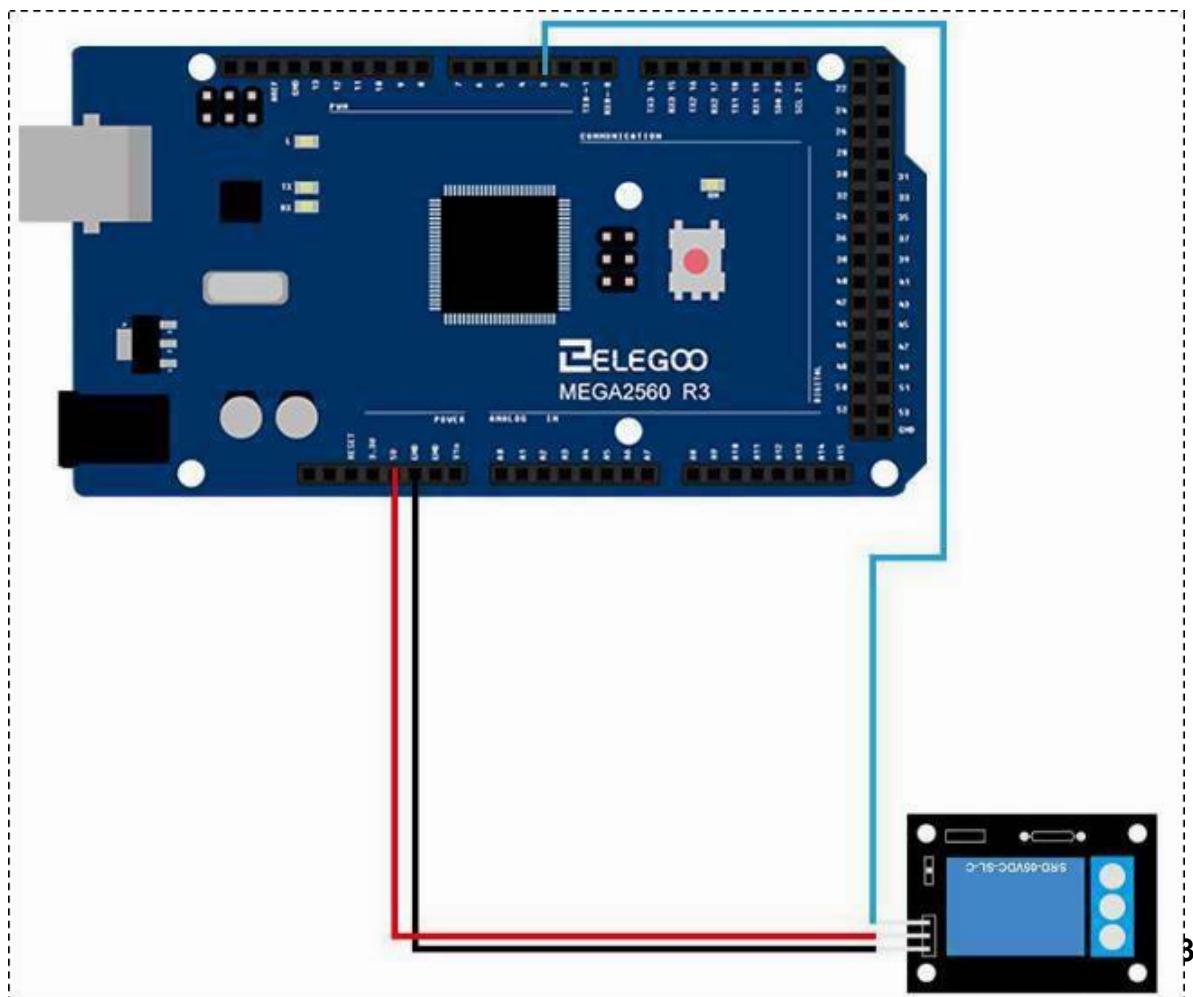
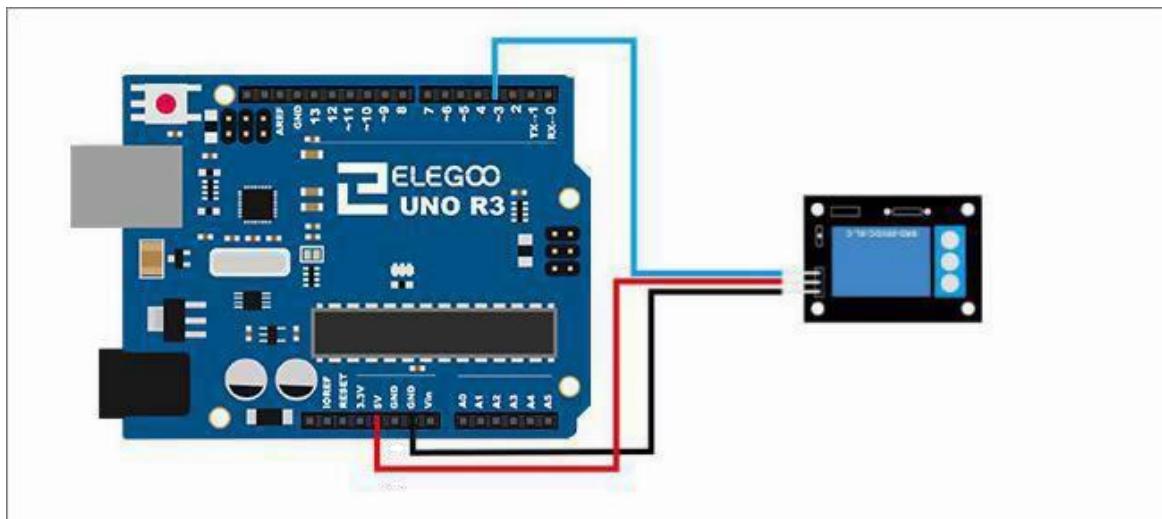


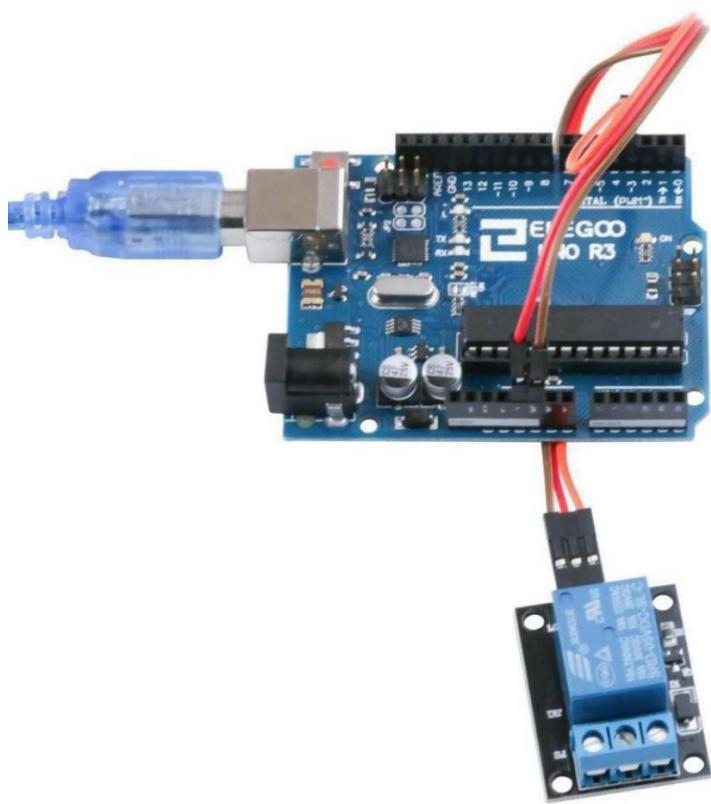
Diagrama de cableado



Código

Después de conectar los circuitos, abrimos la carpeta de códigos de nuestra documentación para encontrar la carpeta "Lección 27 1 Módulo de Relé de 1 Canal" . Abre y corre el compilador cargador y podrás escuchar el sonido del relé al mover sus contactos.

Imagen ejemplo



A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
int relayPin = 3; void setup()
{
    pinMode(relayPin, OUTPUT);
}

void loop()
{
    // Enciende el relé (con el nivel de voltaje HIGH) digitalWrite(relayPin, HIGH);
    // espera por un segundo retardo (2000);
    // Apaga el relé bajando el voltaje a LOW digitalWrite(relayPin, LOW);
    // espera por un segundo retardo (2000);
}
```

Lección 28 Display LCD

En este experimento, aprenderemos a cablear y utilizar un display LCD alfanumérico. El display posee iluminación posterior mediante LEDs y puede mostrar dos filas de hasta 16 caracteres cada una. Puedes ver los rectángulos para cada carácter en el display y los pixeles que componen cada carácter. El display muestra texto de color blanco sobre un fondo azul.

En esta lección, correremos el programa ejemplo de Arduino para la librería LCD, pero en la siguiente, haremos que nuestro display muestre la temperatura, utilizando sensores.

Componentes Requeridos: 1x Elegoo Uno R3
1x LCD1602 module 1 x Resistor(10k)
1 x Protoboard de 830 puntos
16 x Cables M-M (Jumpers Macho-Macho)



Introducción de Componente LCD1602

Introducción a los pines del LCD1602:

VSS: Un pin que se conecta a tierra

VDD: Un pin que se conecta al terminal +5V de la fuente de poder

VO: Un pin para ajustar el contraste del LCD1602

RS: Un pin de selección de registro que controla en qué parte de la memoria del LCD está escribiendo data. Puedes seleccionar o bien sea el registro de datos, que guarda lo que se muestra en pantalla, o el registro de instrucciones, que es donde el LCD buscará las instrucciones acerca de lo que debe hacer.

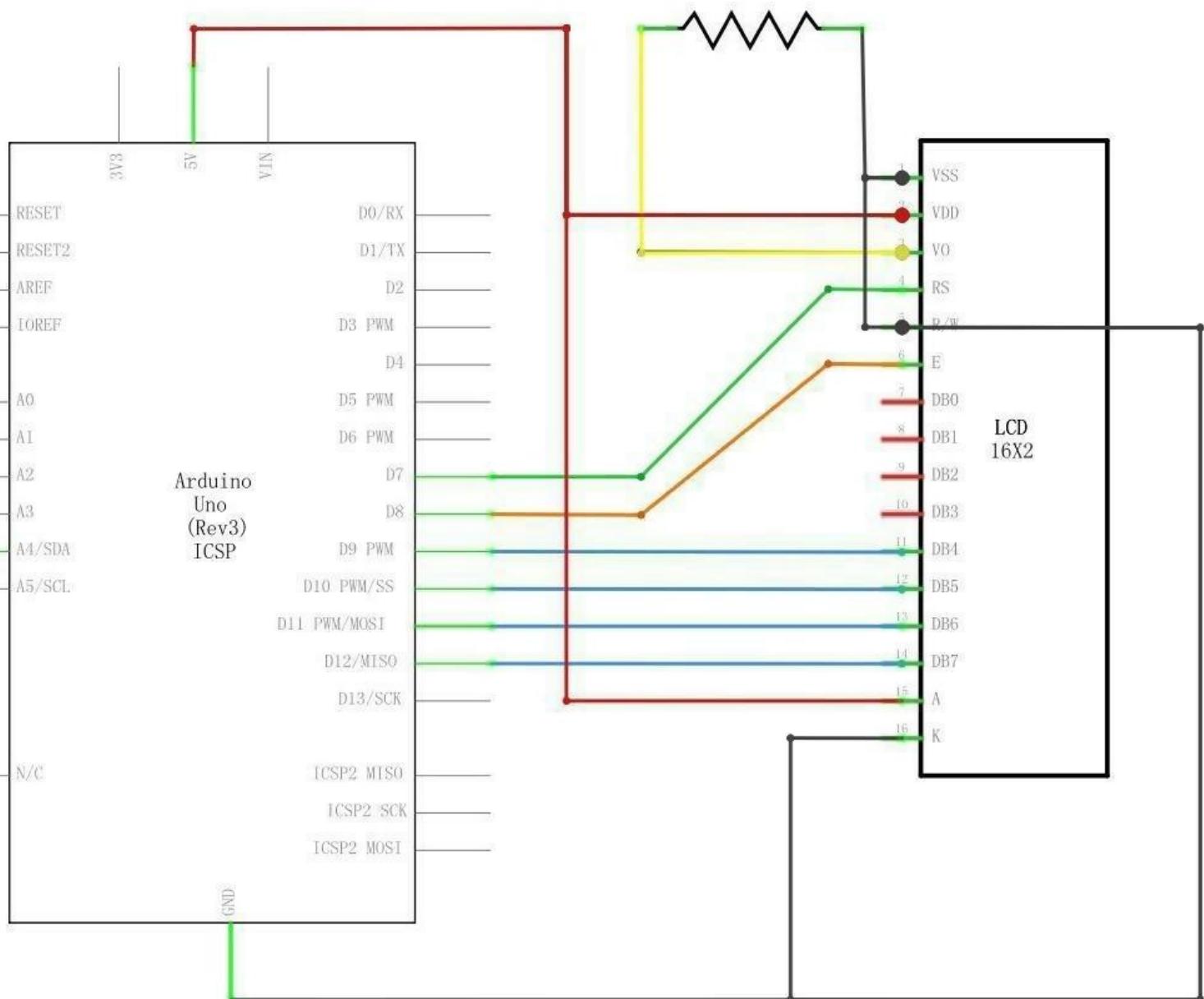
R/W: Un pin Read/Write (de lecturas/ escritura) que selecciona el modo en que deseas trabajar.

E: Un pin de habilitación que, cuando se le entrega un nivel de voltaje bajo, hace que el módulo LCD ejecute instrucciones relevantes.

D0-D7 : Pines para leer y escribir datos

A y K: Pines que controlan la iluminación LED posterior

Esquema de conexión



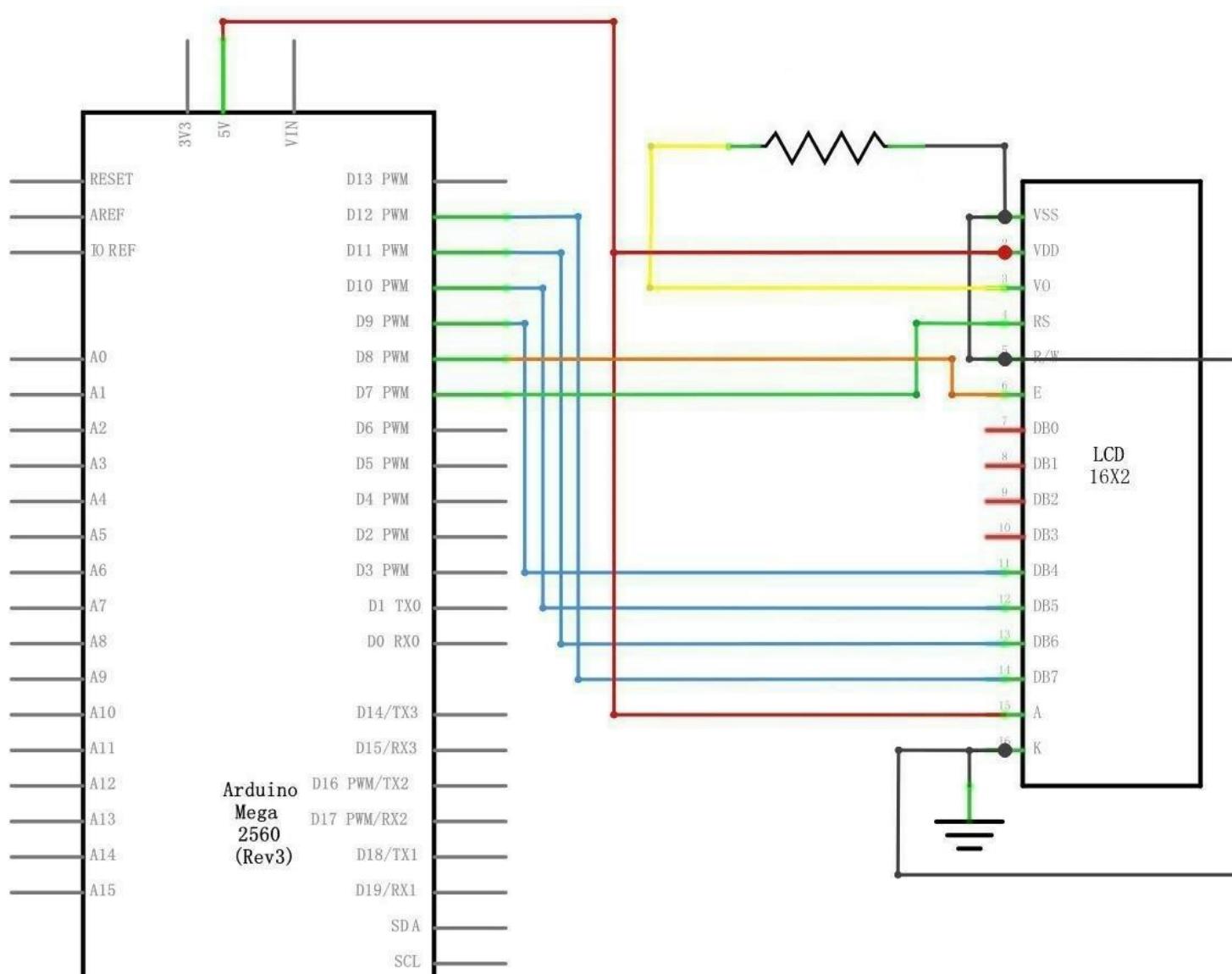
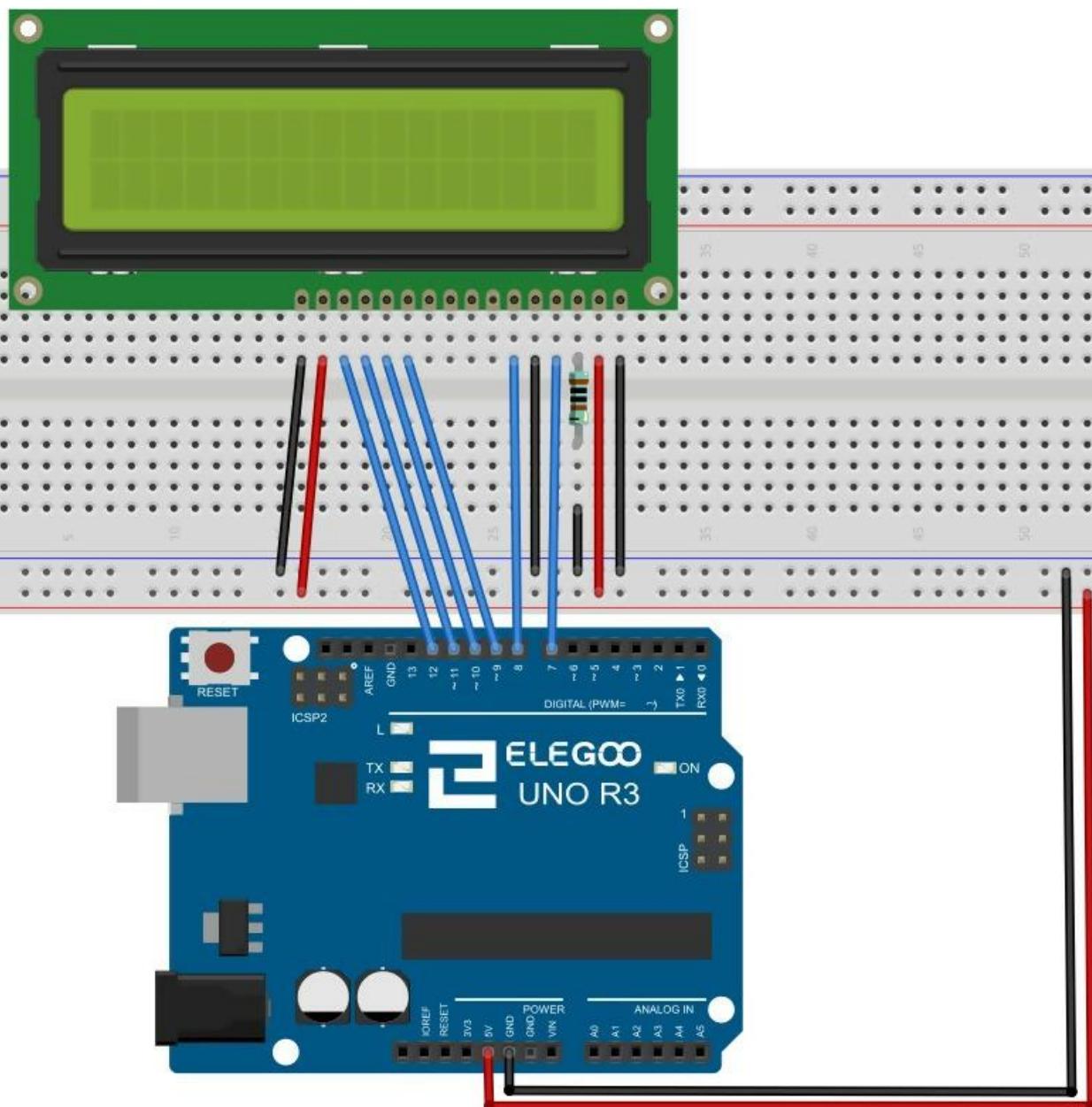
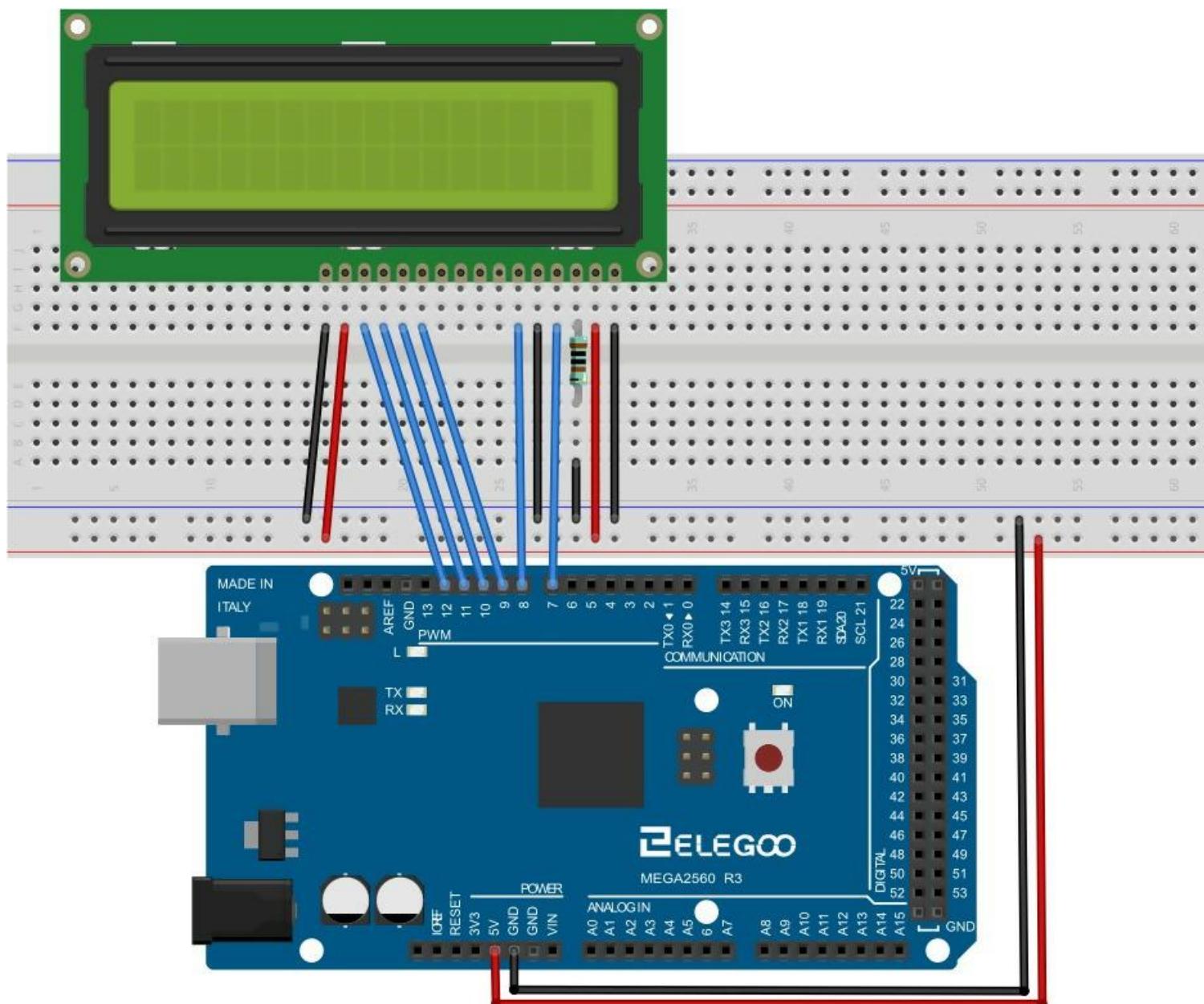


Diagrama de cableado





El Display LCD necesita seis pines Arduino, todos configurados como salidas digitales También necesita conexiones de 5V y GND.

Como ves, es necesario hacer algunas conexiones. Si alineas el displays desde el tope del protoboard, podrás identificar fácilmente los pines sin estar contando tanto, especialmente si el protoboard es de los que vienen numerados con la línea 1 en la parte superior.

El display te podría llegar también separado del cabezal de conexión. Si ese es tu caso, sigue las instrucciones en la siguiente sección

Código:

Después de realizar el cableado, por favor abre el programa en la carpeta de código - Lección 28 Display LCD y haz clic en UPLOAD para cargar el programa. Si hay algún error, consulta los detalles en la lección 2 para cargar el programa.

Antes de poder correrlo, asegúrate de tener la librería < Liquid Crystal > instalada, o reinstálala de ser necesario. De otra forma, tu código no funcionará.

Para detalles del tutorial acerca de cómo cargar los archivos de librerías, consulta la lección 1.

Cuando cargues el código en tu tarjeta Arduino deberías ver el mensaje 'hello, world' seguido de un número que lleva una cuenta a partir de cero.

Lo primero en lo que debes fijarte en el sketch es la línea: #include<LiquidCrystal.h>

Esta le dice al Arduino que queremos usar la librería Liquid Crystal

Después encontraremos la línea que tenemos que modificar. En ella se define que pines del Arduino se deben conectar a que partes del display.

LiquidCrystal LCD(7, 8, 9, 10, 11,12);

Después de cargar este código, asegúrate que la iluminación posterior esté encendida y ajusta el potenciómetro hasta que puedas ver claramente en mensaje de texto en el display.

En la función 'setup' tenemos dos comandos: lcd.begin(16, 2);

lcd.print("Hello,World!");

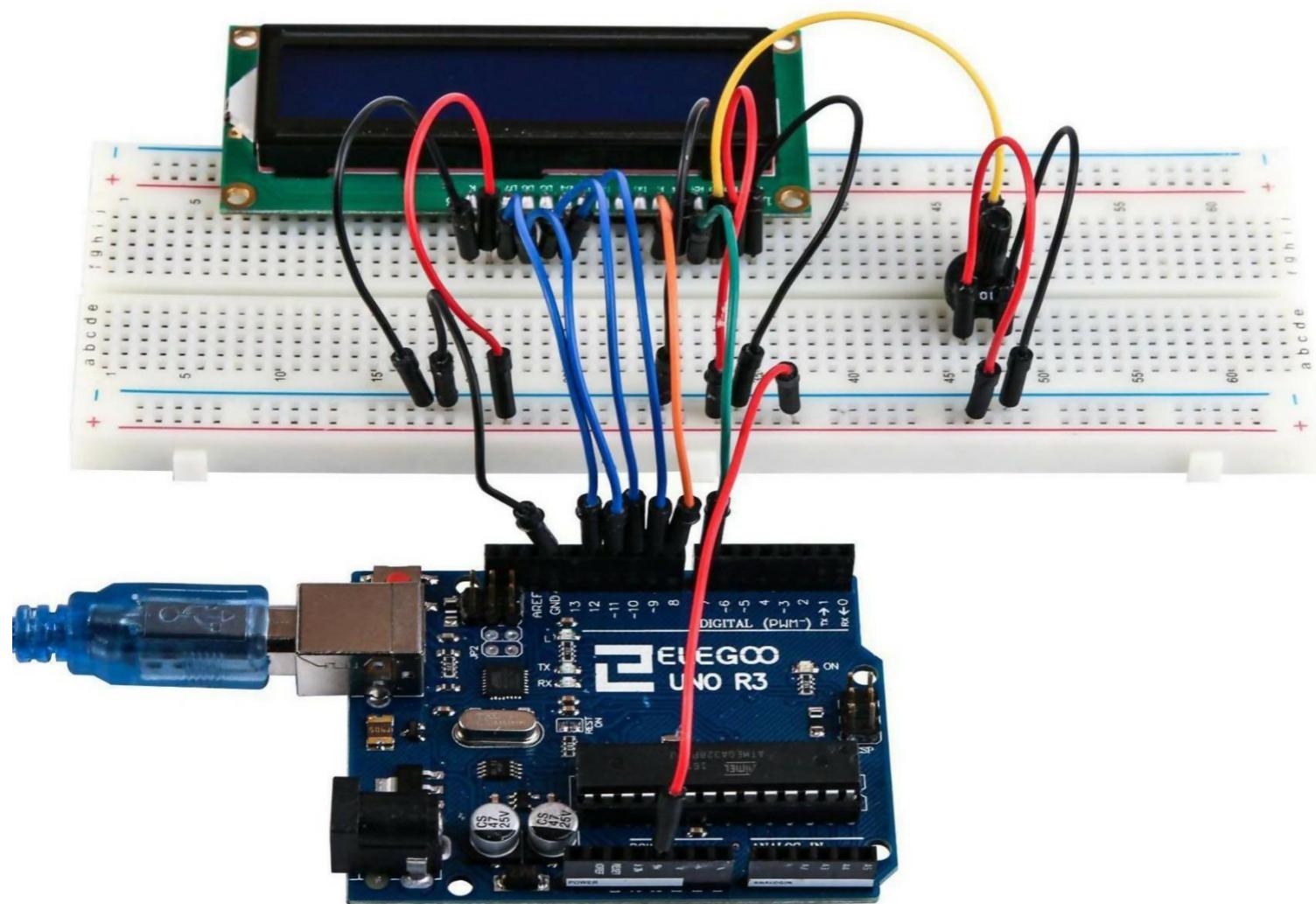
El primero le dice a la librería Liquid Crystal cuantas columnas y filas tiene el display. La segunda línea muestra el mensaje que vemos en la primera línea de la pantalla.

En la función 'loop' tenemos también dos comandos: lcd.setCursor(0, 1);

lcd.print(millis()/1000);

<http://www.elegoo.com>

El primero configura la posición del cursor (donde aparecerá el próximo texto) a columna



A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
// Incluye la librería de código: #include <LiquidCrystal.h>

// Inicializa la librería con los números de pin de la interfaz LiquidCrystal LCD(7, 8, 9, 10, 11, 12);

void setup() {
    // Configura el número de columnas y filas del LCD: lcd.begin(16, 2);
    // Imprime un mensaje al LCD. lcd.print("hello, world!");
}

void loop() {
    // Coloca el cursor en la columna 0, línea 1
    // (nota: línea 1 es la segunda fila, debido a que el conteo empieza con 0): lcd.setCursor(0, 1);
    // Imprime el número de segundos desde el reinicio: lcd.print(millis() / 1000);
}
```

Lección 29 Módulo de Sensor Ultrasónico

Los sensores ultrasónicos son excelentes para todas clases de proyectos que necesiten tomar mediciones de distancia, como por ejemplo los de evasión de obstáculos.

El sensor HC-SR04 es económico y fácil de usar, ya que estaremos utilizando una librería específicamente diseñada para este sensor.

Componentes Requeridos:

1 x Elegoo Uno R3

1 x Módulo de Sensor Ultrasónico

4 x Cables F-M (Cables DuPont Hembra a Macho)



Introducción de Componente

Sensor Ultrasónico

El módulo de sensor ultrasónico HC-SR04 provee una función de medición sin contacto en un rango entre 2cm-400cm , con una exactitud de 3mm. Incluye transmisores ultrasónicos, receptor y circuito de control.

El principio de funcionamiento básico:

Usando un disparador IO para una señal mínima de alto nivel de 10us,

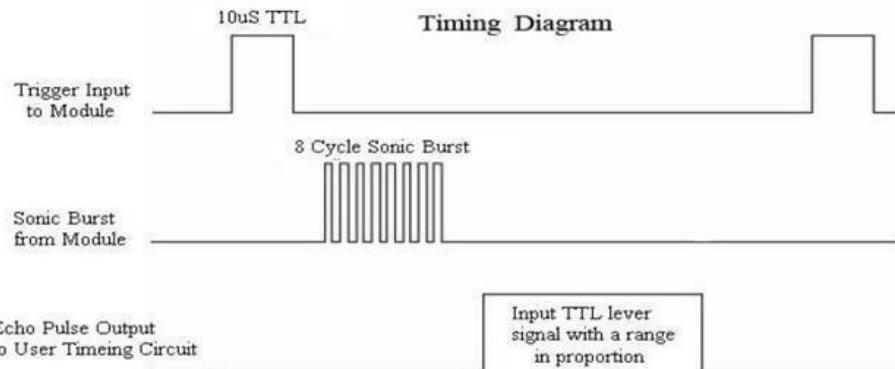
El módulo envía automáticamente ocho pulsos de 40 kHz y detecta si hay algún pulso de rebote. Si se recibe una señal de vuelta, el tiempo de duración de la señal de nivel alto puede servir como indicativo de la distancia

Distancia de prueba = $(\text{Tiempo de nivel alto} \times \text{velocidad del sonido (340m/s)}) / 2$

El diagrama de tiempo se muestra abajo. Solo se necesita un pequeño pulso de 10us para que la señal del disparador comience a medir. Luego el módulo enviará un ciclo de ocho pulsos de 40 kHz y escuchará el eco. El eco le indica la distancia respecto al obstáculo. Puedes calcular un rango a través de la señal Formula: us

el intervalo de tiempo que transcurre entre el envío de la señal y la recepción del eco.

/ 58 = centímetros / 148 =inch; o: el rango = tiempo de nivel alto * velocidad (340M/S) / 2; sugerimos utilizar ciclos de medición mayores a 60ms, para prevenir que la señal del disparador se confunda con el eco.



Esquema de conexión

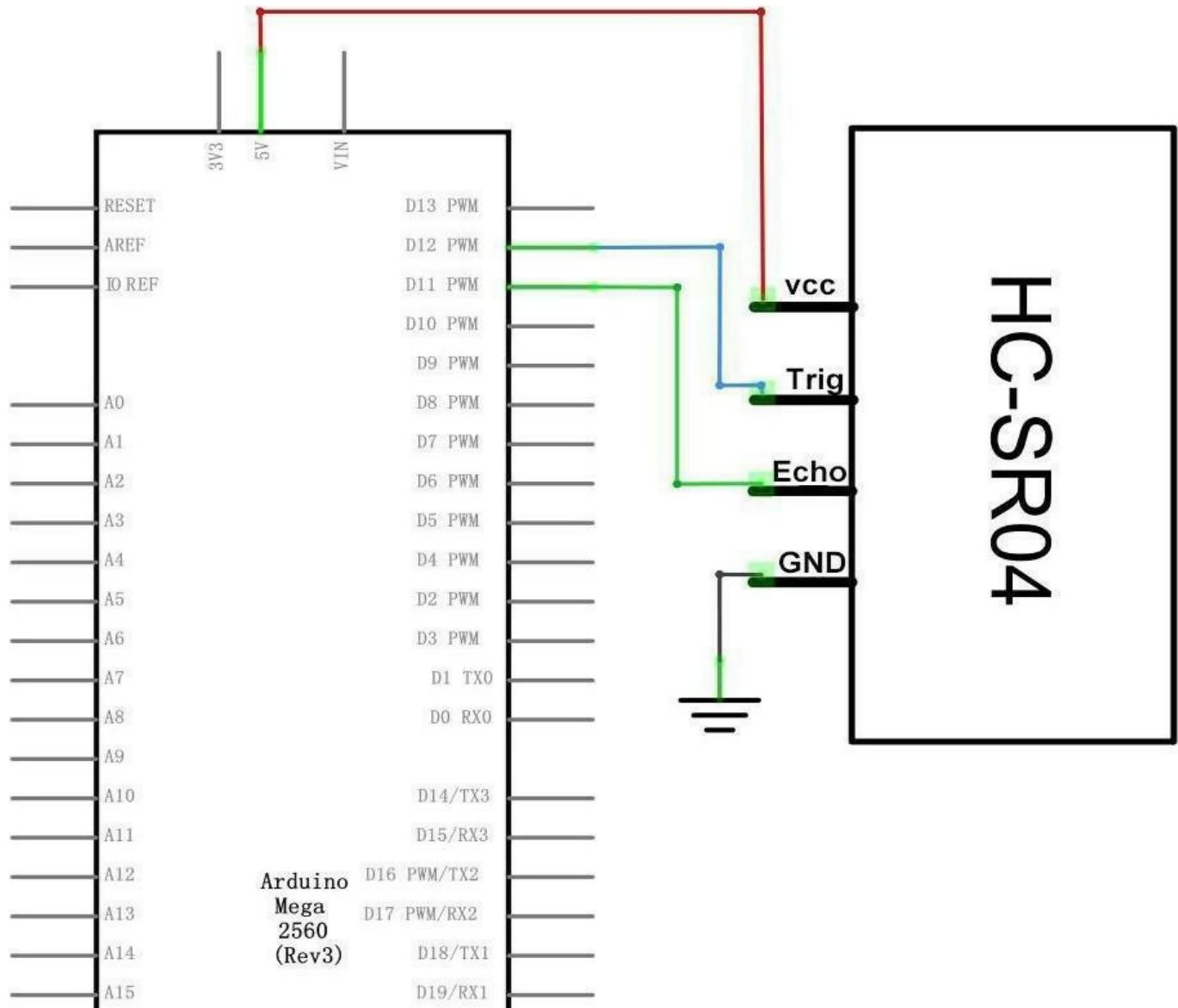
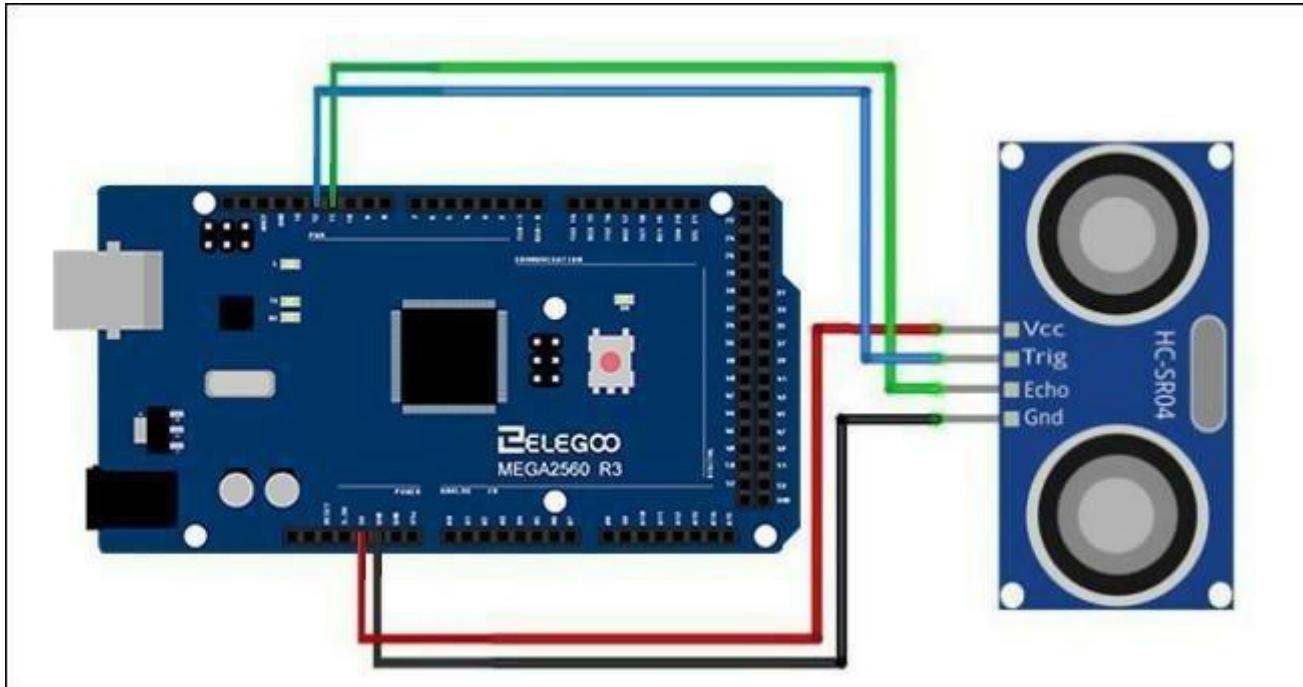
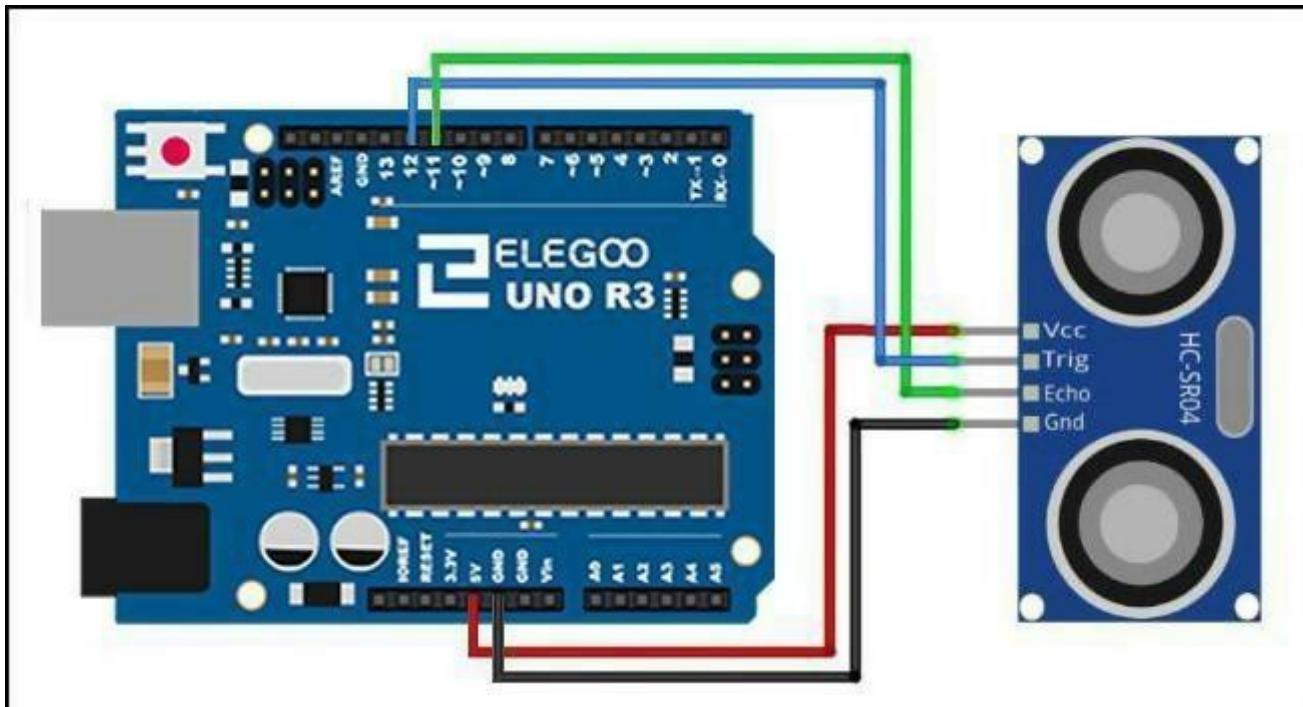


Diagrama de cableado



Código

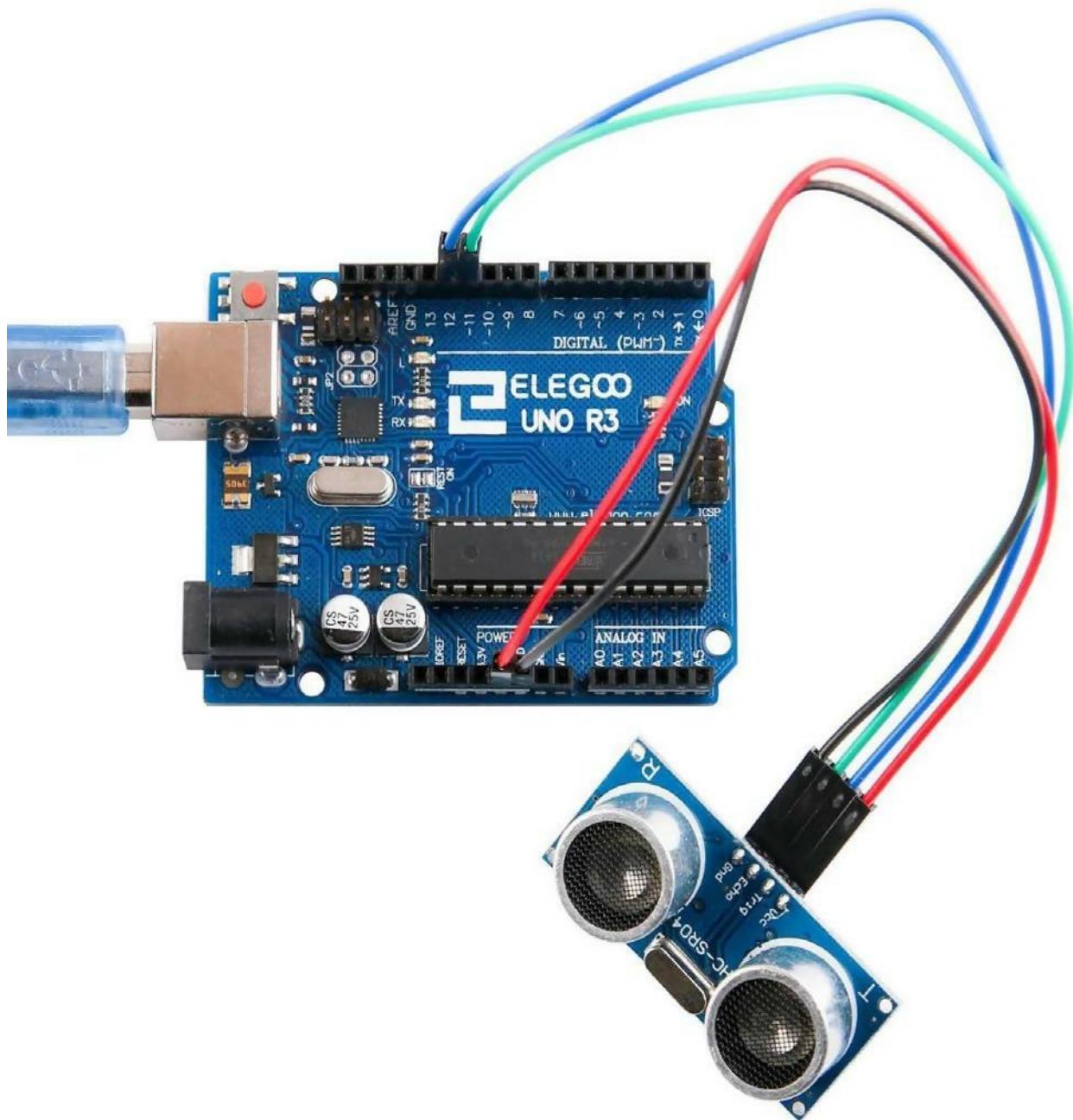
Si utilizamos una librería diseñada para estos sensores haremos que nuestro código sea corto y simple. Incluimos la librería al principio de nuestro código y luego usamos comandos sencillos para controlar los sensores.

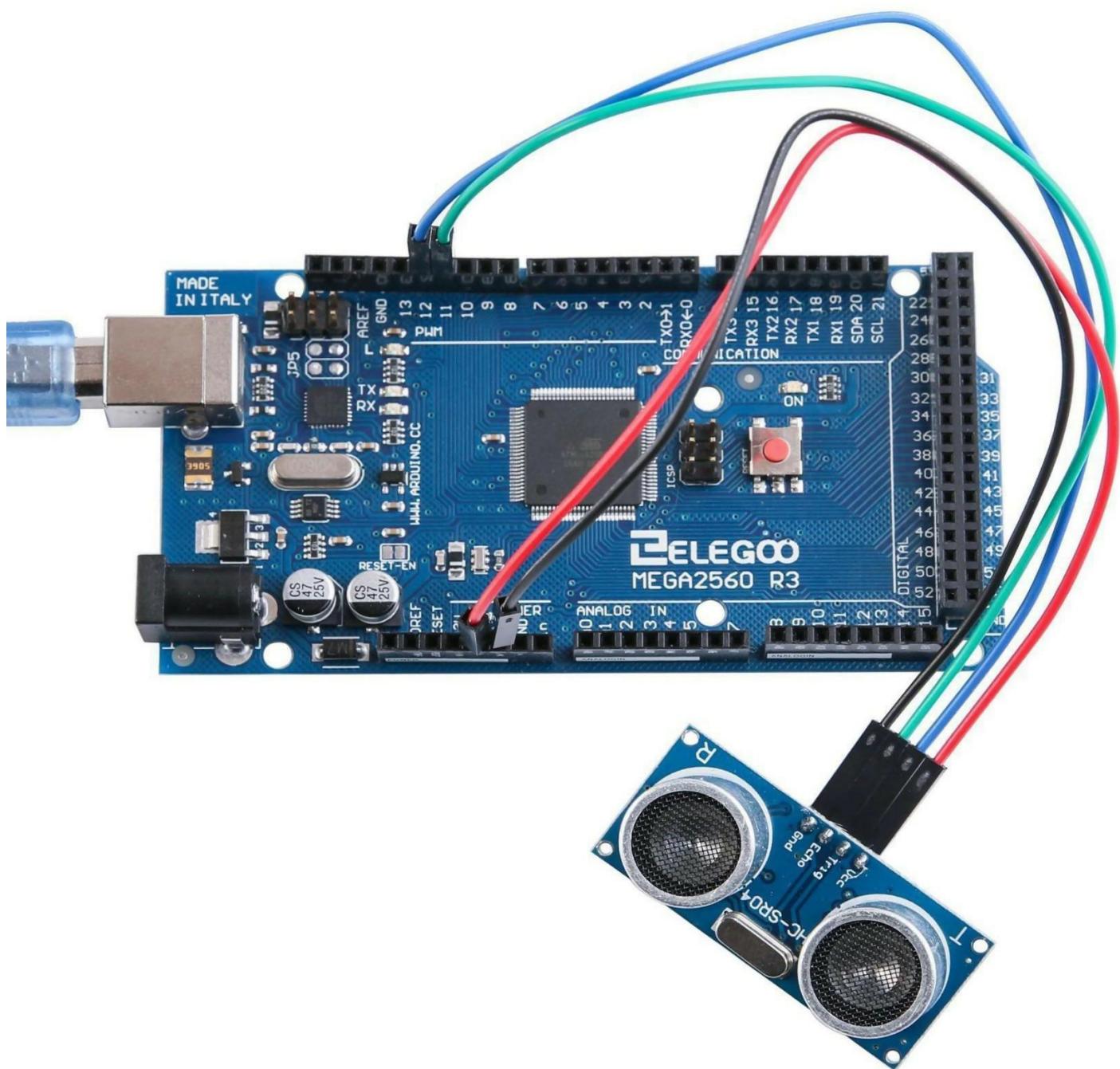
Después de realizar el cableado, por favor abre el programa en la carpeta de código (Lección 29 Módulo de Sensor Ultrasónico) y haz clic en UPLOAD para cargar el programa. Si hay algún error, consulta los detalles en la lección 2 para cargar el programa.

Antes de poder correrlo, asegúrate de tener la librería < NewPing> instalada, o reinstálala de ser necesario. De otra forma, tu código no funcionará.

Para detalles del tutorial acerca de cómo cargar los archivos de librerías, consulta la lección 1.

Imagen ejemplo





Luego abre el monitor, para ver los siguientes datos:

Haz clic en el Monitor Serial para encenderlo

Lo básico a saber del monitor serial se enseña en la lección 1

```
4cm
124cm
125cm
126cm
125cm
125cm
126cm
127cm
125cm
125cm
125cm
126cm
125cm
126cm
12cm
125cm
125cm
8cm
126cm
8cm
10cm
11cm
0cm
5cm
0cm
5cm
178cm
126
```

Autoscroll Newline 9600 baud

A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
#include <NewPing.h>

// El pin del Arduino a conectar con el pin del disparador del sensor ultrasónico. #define TRIGGER_PIN 12
// El pin del Arduino a conectar con el pin del eco del sensor ultrasónico. #define ECHO_PIN    11
/*Distancia máxima (en centímetros) a la que queremos probar. La distancia máxima nominal del sensor
es de 400-500cm.*/ #define MAX_DISTANCE 200
// NewPing configuración de pines y distancia máxima.
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

void setup() {
// Abre el monitor serial a 115200 baudios para ver los resultados de las pruebas. Serial.begin(9600);
}

void loop() {
// Espera 500ms entre pruebas (aproximadamente 2 pings/sec). el retraso más corto entre pruebas
debería ser de 29ms delay(500);
// envía un ping, obtener el tiempo de prueba en microsegundos (uS). unsigned int uS = sonar.ping();
Serial.print("Ping: ");
// Convierte el tiempo de prueba en distancia e imprime el resultado (0 = fuera del rango de prueba, no se
recibe eco) Serial.print(uS / US_ROUNDTRIP_CM);
Serial.println("cm");
}
```

Lección 30 Módulo GY-521

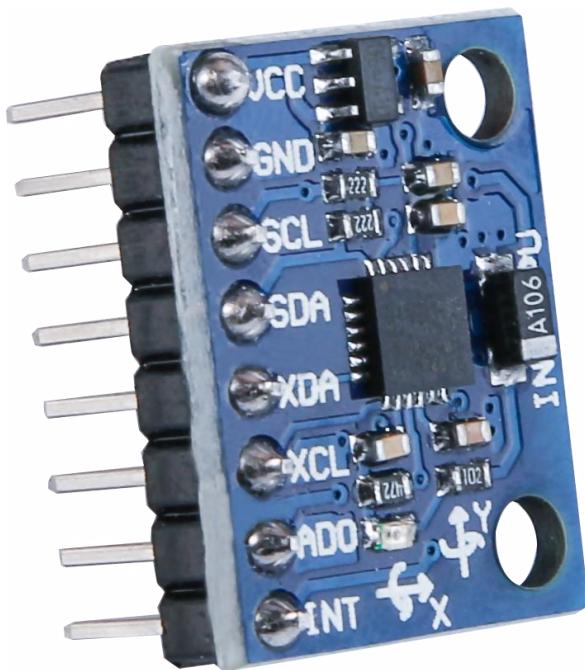
En esta lección, aprenderemos a usar el Módulo GY-521 (MPU-6050), uno de los mejores IMU (Inertia Measurement Unit, Unidad de Medición de Inercia) compatibles con Arduino.

Sensores IMU como el GY-521(MPU-6050) se utilizan en robot auto equilibrado, UAVs, teléfonos inteligentes, etc.

Componentes requeridos:

1 x Elegoo Uno R3

1 x Módulo GY-521 4 x Cables F-M



Introducción de Componente

Sensor GY-521

El sensor InvenSense GY-521 contiene tanto un acelerómetro como un giroscopio MEMS en el mismo chip. Es muy preciso, ya que contiene hardware de conversión analógica - digital de 16-bits para cada canal. Por tanto, captura los canales x, y, z al mismo tiempo. El sensor utiliza la interfaz I2C-bus para comunicarse con el Arduino.

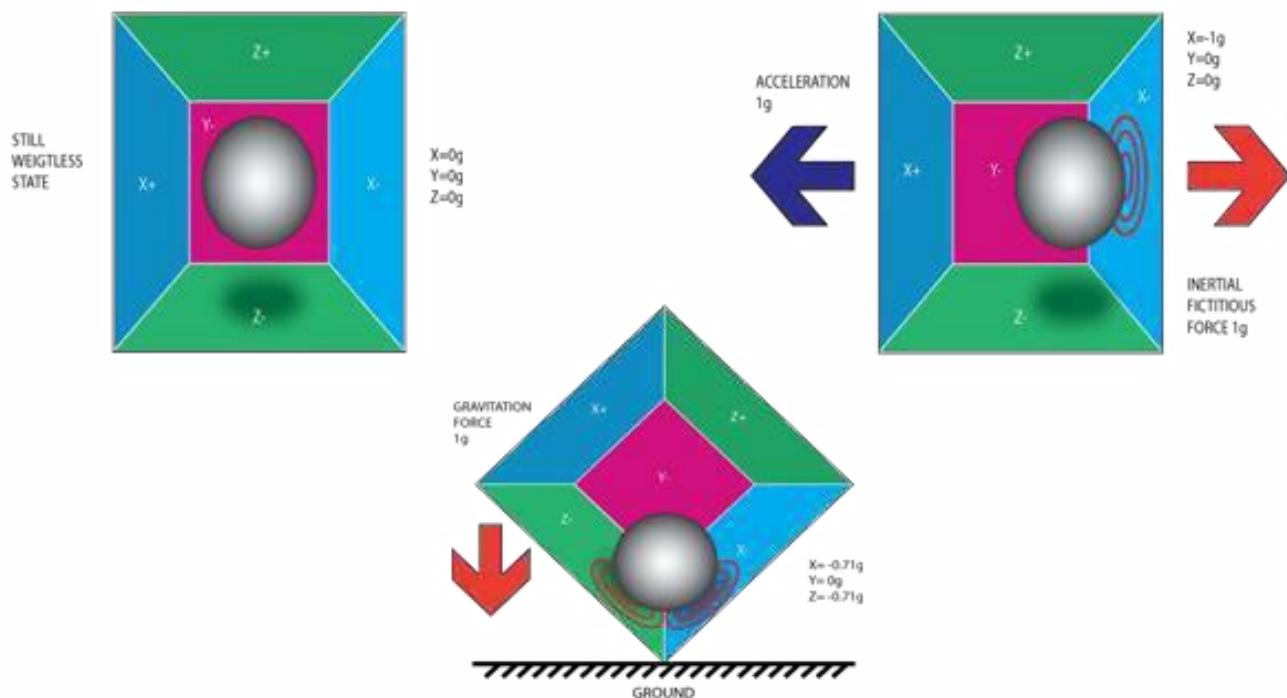
Este módulo GY-521 no es caro, especialmente si tomas en cuenta el hecho de que combina un acelerómetro y un giroscopio.

Los sensores IMU constituyen se utilizan en la actualidad en prácticamente toda clase de dispositivos electrónicos. Vienen en los teléfonos inteligentes, equipos vestibles, controladores de juegos, etc. Nos ayudan a entender cómo se siente un objeto unido al sensor en el espacio tridimensional. Tiene valores usualmente en ángulos, que nos ayudan a determinar la posición de los objetos. Se usan en los teléfonos inteligentes por ejemplo, para determinar su orientación. Dispositivos tales como el Nike fuel band o el fit bit, utilizan sensores IMU para registrar movimiento.

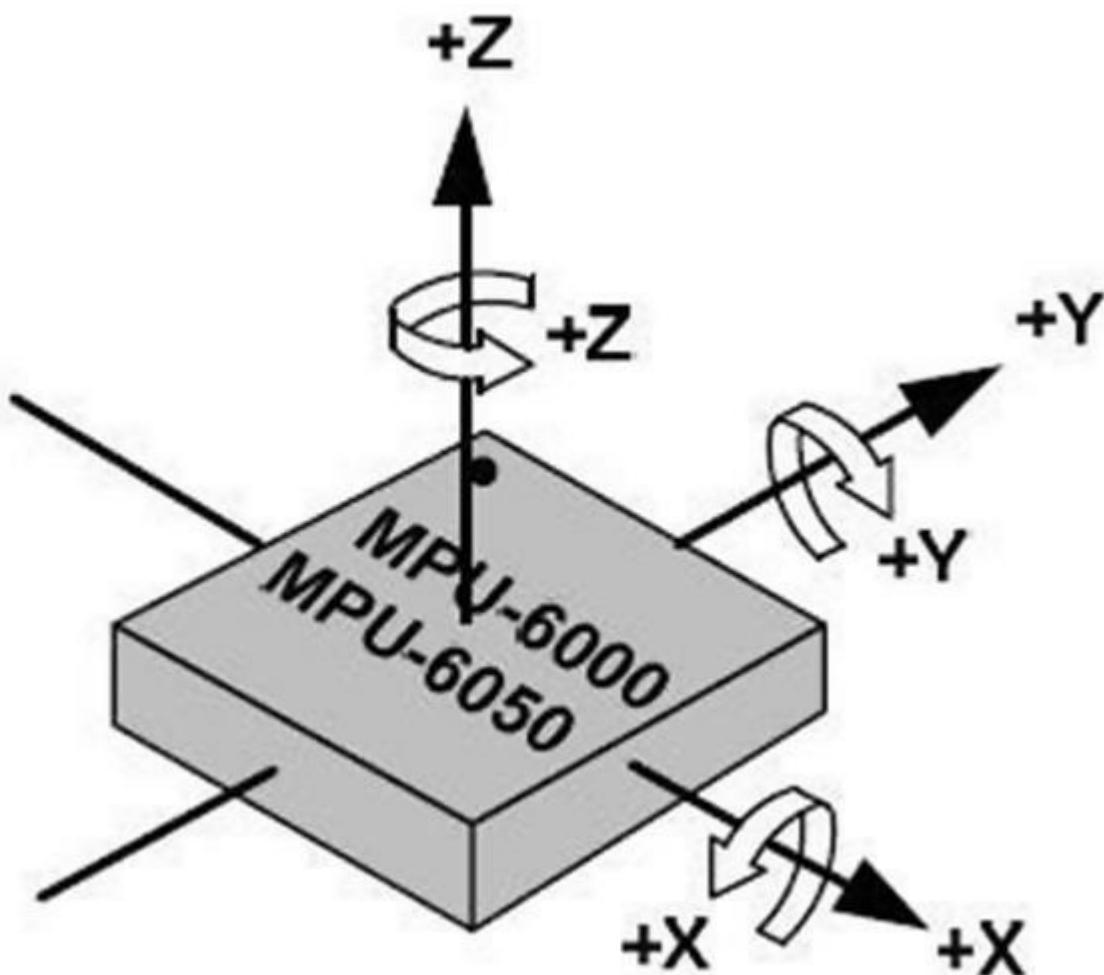
Cómo trabajan?

Un sensor IMU está formado de una o más partes. En orden de prioridad, son: acelerómetro, giroscopio, magnetómetro y altímetro. El sensor GY-521 consta de 6 DOF (Degrees of Freedom / grados de libertad) es decir, un sensor IMU de seis ejes, lo que significa que entrega seis valores de salida. Tres valores corresponden al acelerómetro y tres al giroscopio. El GY-521 es un sensor basado en la tecnología MEMS (Micro Electro Mechanical Systems / Sistemas Micro Electromecánicos). Tanto el acelerómetro como el giroscopio están incluidos en el mismo chip. Utiliza el protocolo I2C (Inter Integrated Circuit) para comunicarse.

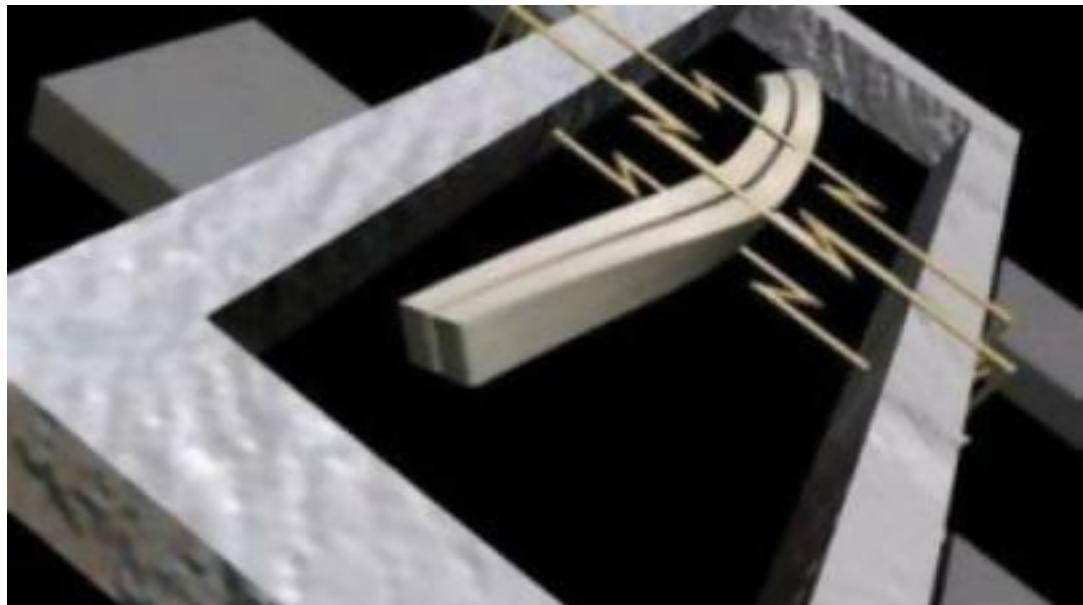
Cómo trabaja un acelerómetro?



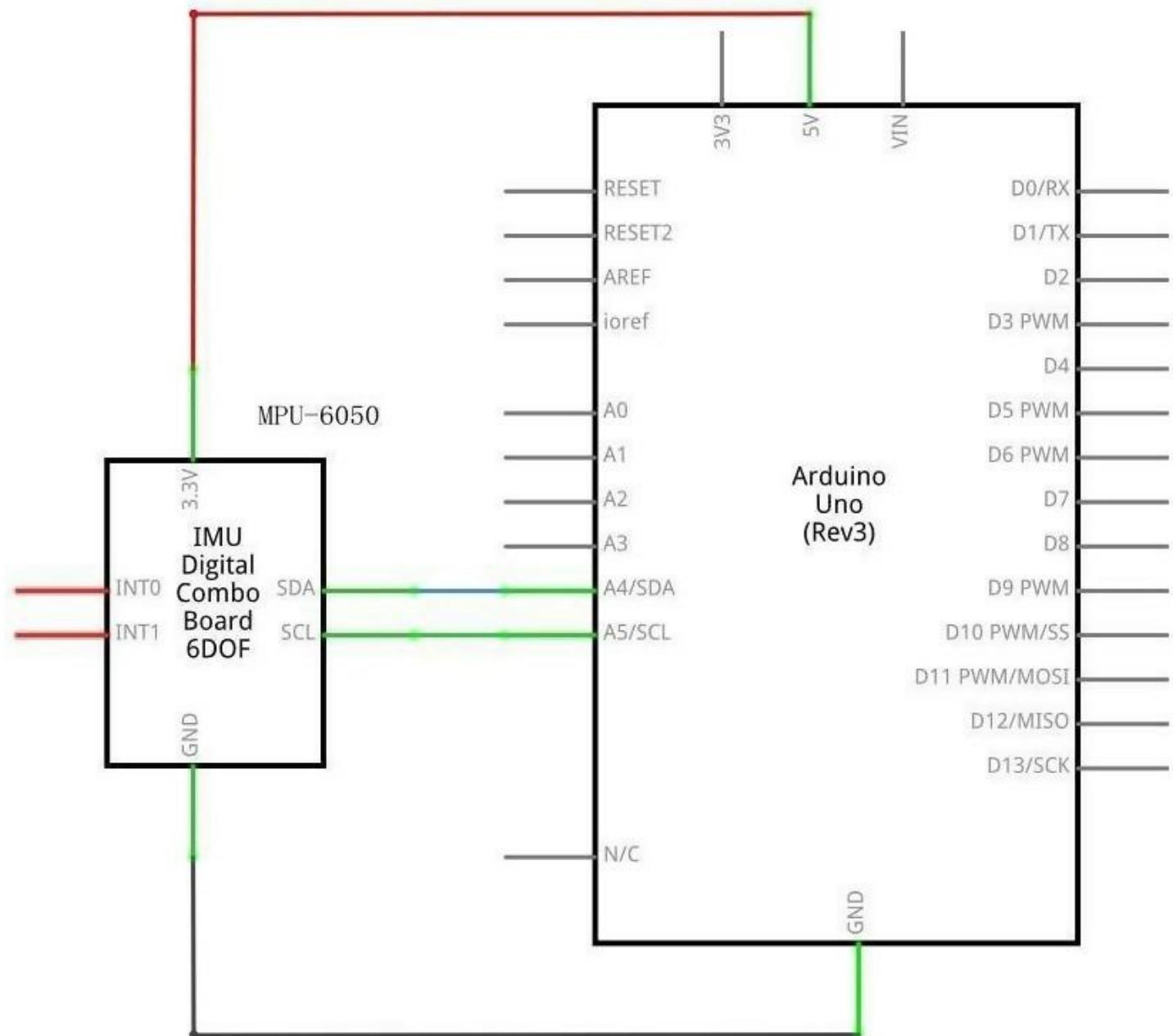
Un acelerómetro trabaja de acuerdo con el principio del efecto piezo eléctrico. Imagina una caja en forma de cubo, con una pequeña pelota dentro de ella, como se ve en la imagen de arriba. Las paredes de esta caja están hechas de cristales piezo eléctricos. Debido a la gravedad, cuando la caja se mueve, la pelota se ve forzada a moverse de acuerdo a la inclinación de la misma. Las paredes con las que choca la bola crean pequeñas corrientes piezo eléctricas. Hay en total tres pares opuestos de paredes en un cubo. Cada par corresponde a un eje en el espacio 3D: X, Y, Z. Dependiendo de la corriente producida por las paredes piezo eléctricas, podemos determinar la dirección de inclinación y la magnitud de la misma. Para más información, lee aquí.



Cómo trabaja un giroscopio?



Los giroscopios trabajan de acuerdo con el principio de la aceleración Coriolis. Imagina que hay una estructura en forma de horquilla, que se mueve constantemente hacia adelante y hacia atrás. Se mantiene en su lugar utilizando cristales piezo eléctricos. Cuando tratas de inclinar la estructura, los cristales reciben una fuerza en la misma dirección de la inclinación. Esto sucede debido a la inercia de la horquilla en movimiento. Los cristales producirán una corriente en consenso con el efecto piezo eléctrico, la cual será amplificada. Los valores obtenidos son luego refinados en el microcontrolador.



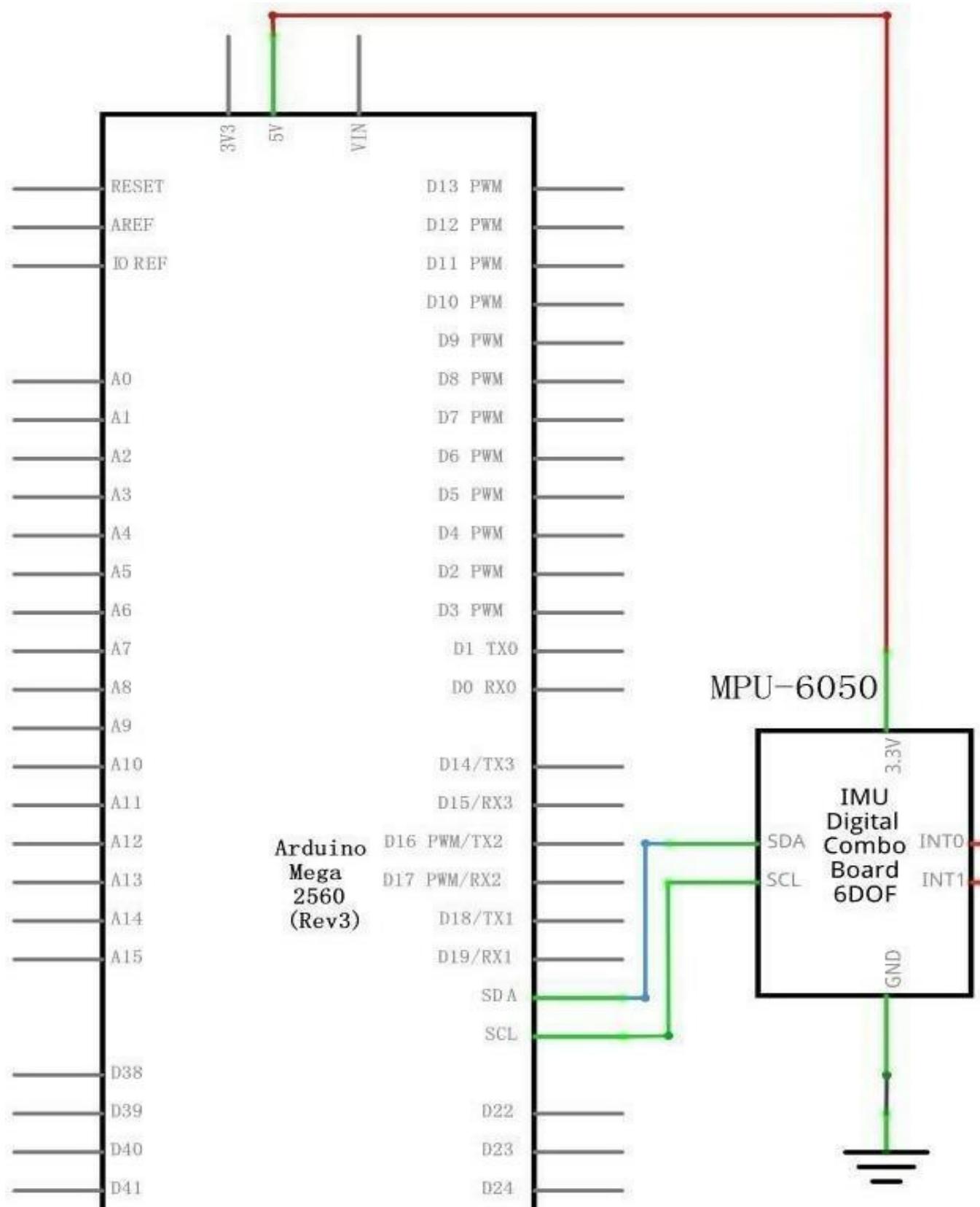
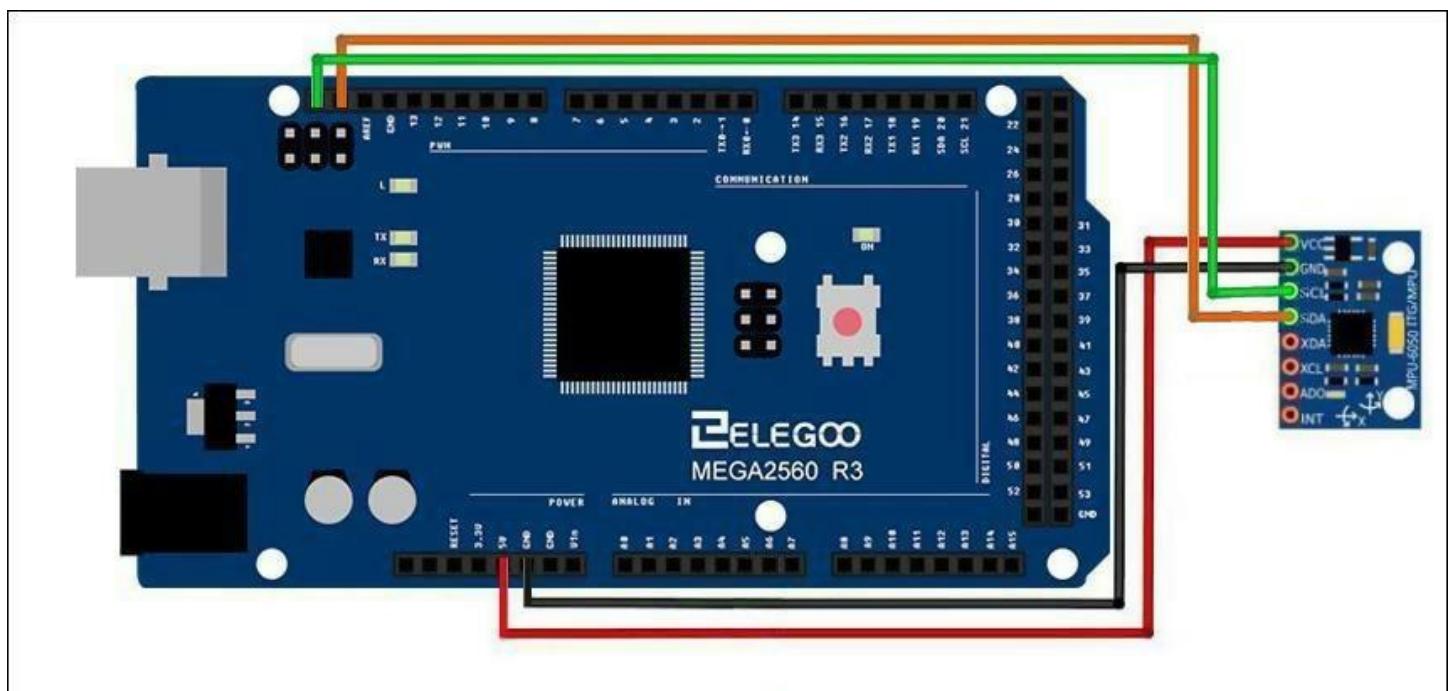
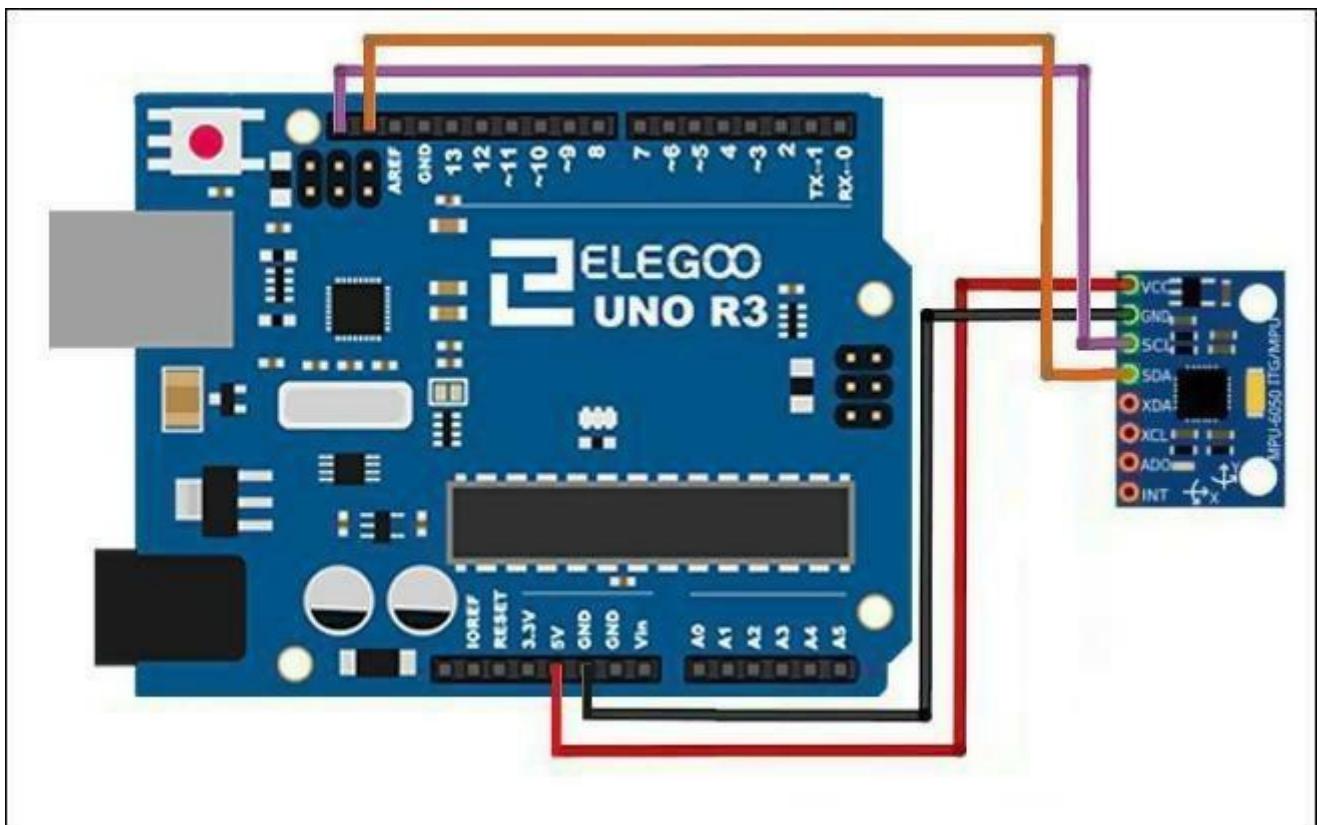


Diagrama de cableado



Ahora, necesitamos configurar las líneas I2C. Para tal fin, conecta el pin etiquetado SDA en el GY-521 al pin analógico 4 del Arduino (SDA). y el pin etiquetado SCL en el GY-521 al pin analógico 5 del Arduino (SCL). De esta forma habrás terminado de cablear el Arduino GY-521(GY-521).

Librerías necesarias GY-521

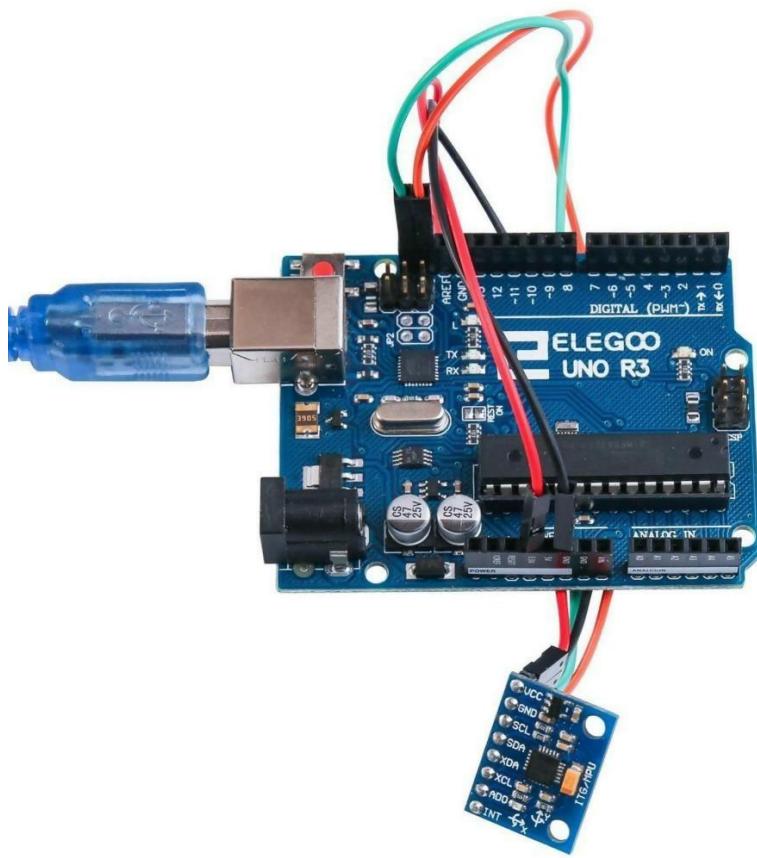
El código

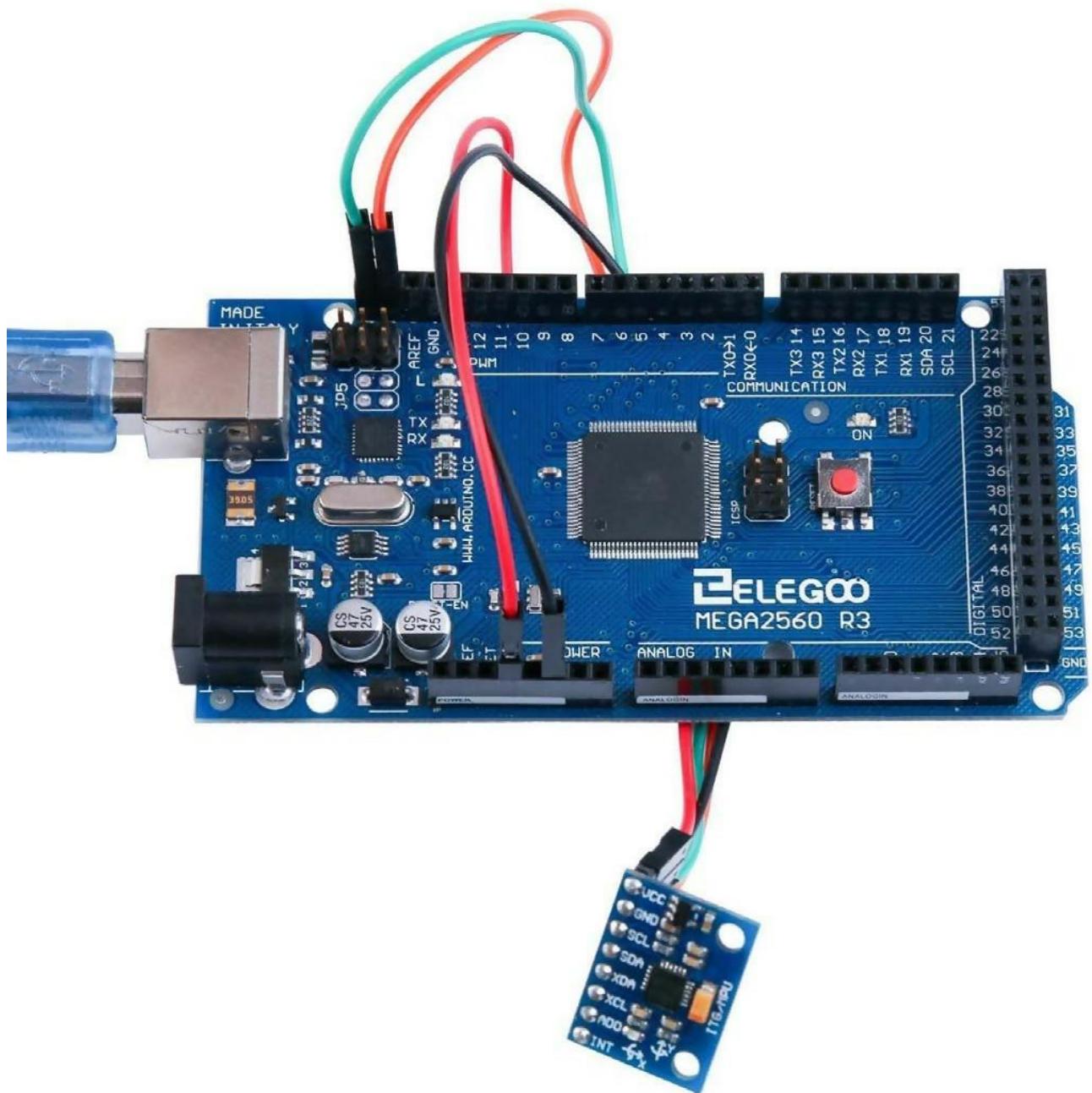
En el sketch de ejemplos se muestran todos los valores crudos (acelerómetro, giroscopio y temperatura).

Debería trabajar con el Arduino Uno, Nano, Leonardo y Due.

Después de realizar el cableado, por favor abre el programa en la carpeta de código Lección 30 GY-521 y haz clic en UPLOAD para cargar el programa. Si hay algún error, consulta los detalles en la lección 2 para cargar el programa.

Imagen ejemplo





Luego abre el monitor, para ver los siguientes datos:

Haz clic en el Monitor Serial para encenderlo Los básicos a saber del monitor serial se enseña en la lección

1

The screenshot shows the Arduino Serial Monitor window titled "COM21 (Arduino/Genuino Uno)". The window displays a continuous stream of sensor data in a text-based format. Each line of data represents a measurement with five components: Acceleration (AcX, AcY, AcZ) and Gravity (GyX, GyY, GyZ). The values are floating-point numbers ranging from approximately -5960 to 3600. The data is displayed in a scrollable text area. At the bottom of the window, there are several configuration options: "Autoscroll" (checked), "Newline" (dropdown menu), and "9600 baud" (dropdown menu).

```
AcX = 15976 | AcY = -4280 | AcZ = -596 | Imp = 24.62 | GyX = 230 | GyY = -26 | GyZ = -1231
AcX = 15940 | AcY = -4408 | AcZ = -648 | Imp = 24.53 | GyX = -357 | GyY = -590 | GyZ = -519
AcX = 15852 | AcY = -4328 | AcZ = -668 | Imp = 24.62 | GyX = -189 | GyY = -79 | GyZ = -198
AcX = 15844 | AcY = -3972 | AcZ = -708 | Imp = 24.62 | GyX = -254 | GyY = -216 | GyZ = -44
AcX = 15740 | AcY = -4232 | AcZ = -968 | Imp = 24.53 | GyX = -319 | GyY = -141 | GyZ = -227
AcX = 15900 | AcY = -4936 | AcZ = -1008 | Imp = 24.62 | GyX = 126 | GyY = 111 | GyZ = 3870
AcX = 15356 | AcY = -5080 | AcZ = -1192 | Imp = 24.58 | GyX = -1670 | GyY = -1741 | GyZ = -2571
AcX = 14592 | AcY = -6504 | AcZ = -5700 | Imp = 24.53 | GyX = 662 | GyY = 264 | GyZ = 3219
AcX = 13740 | AcY = -7020 | AcZ = -2744 | Imp = 24.58 | GyX = 8265 | GyY = 4962 | GyZ = 8163
AcX = 3600 | AcY = -16556 | AcZ = 4244 | Imp = 24.58 | GyX = -17048 | GyY = -12197 | GyZ = 3845
AcX = 12248 | AcY = -12292 | AcZ = 7256 | Imp = 24.62 | GyX = 12046 | GyY = 24428 | GyZ = -5483
AcX = 588 | AcY = -3832 | AcZ = 19208 | Imp = 24.53 | GyX = 9258 | GyY = -4420 | GyZ = -4557
AcX = 1896 | AcY = -3784 | AcZ = 6320 | Imp = 24.62 | GyX = -7486 | GyY = -32768 | GyZ = 2677
AcX = 32767 | AcY = -19068 | AcZ = -1920 | Imp = 24.58 | GyX = -9262 | GyY = -19403 | GyZ = 25320
AcX = -19160 | AcY = 12004 | AcZ = -2452 | Imp = 24.58 | GyX = -32768 | GyY = -32768 | GyZ = -4809
AcX = -25124 | AcY = 1616 | AcZ = 32767 | Imp = 24.62 | GyX = 7628 | GyY = 7064 | GyZ = 6299
AcX = 11976 | AcY = -8432 | AcZ = -32600 | Imp = 24.53 | GyX = 29381 | GyY = 32767 | GyZ = -19841
AcX = 972 | AcY = -22992 | AcZ = -12480 | Imp = 24.62 | GyX = -31051 | GyY = -32768 | GyZ = 32767
AcX = -27260 | AcY = 16868 | AcZ = 10704 | Imp = 24.62 | GyX = 32767 | GyY = 28603 | GyZ = -20636
AcX = 32268 | AcY = -32468 | AcZ = -21952 | Imp = 24.58 | GyX = -27684 | GyY = -32768 | GyZ = 32767
AcX = -22476 | AcY = -8436 | AcZ = -3976 | Imp = 24.58 | GyX = 32156 | GyY = 32767 | GyZ = 25696
AcX = -3836 | AcY = -13428 | AcZ = -9628 | Imp = 24.58 | GyX = -30925 | GyY = -32768 | GyZ = 32767
AcX = 3164 | AcY = -5392 | AcZ = -19464 | Imp = 24.48 | GyX = -30769 | GyY = -17986 | GyZ = 17236
AcX = 3408 | AcY = -3584 | AcZ = -13752 | Imp = 24.58 | GyX = 1820 | GyY = -2660 | GyZ = -186
AcX = 4404 | AcY = -5552 | AcZ = -15216 | Imp = 24.58 | GyX = -578 | GyY = -234 | GyZ = -425
AcX = 4160 | AcY = -5456 | AcZ = -15304 | Imp = 24.53 | GyX = -445 | GyY = -154 | GyZ = -277
AcX = 4152 | AcY = -5192 | AcZ = -15300 | Imp = 24.53 | GyX = -404 | GyY = -114 | GyZ = -262
```

A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
// Sketch corto de ejemplo GY-521
// Por el usuario Arduino John Chi
// August 17, 2014
// De dominio público #include<Wire.h>
// Dirección I2C del GY-521 const int MPU_addr=0x68;
int16_t AcX,AcY,AcZ,Tmp,GyX,GyY,GyZ; void setup(){
Wire.begin(); Wire.beginTransmission(MPU_addr);
// PWR_MGMT_1 register Wire.write(0x6B);
// Ajustar a cero (despierta el GY-521) Wire.write(0); Wire.endTransmission(true); Serial.begin(9600);
}
void loop(){ Wire.beginTransmission(MPU_addr);
// Comenzando con el registro 0x3B (ACCEL_XOUT_H) Wire.write(0x3B); Wire.endTransmission(false);
// solicita un total de 14 registros Wire.requestFrom(MPU_addr,14,true);
// 0x3B (ACCEL_XOUT_H) & 0x3C (ACCEL_XOUT_L)
AcX=Wire.read()<<8|Wire.read();
// 0x3D (ACCEL_YOUT_H) & 0x3E (ACCEL_YOUT_L)
AcY=Wire.read()<<8|Wire.read();
// 0x3F (ACCEL_ZOUT_H) & 0x40 (ACCEL_ZOUT_L)

AcZ=Wire.read()<<8|Wire.read();
// 0x41 (TEMP_OUT_H) & 0x42 (TEMP_OUT_L)
Tmp=Wire.read()<<8|Wire.read();
// 0x43 (GYRO_XOUT_H) & 0x44 (GYRO_XOUT_L)
GyX=Wire.read()<<8|Wire.read();
// 0x45 (GYRO_YOUT_H) & 0x46 (GYRO_YOUT_L)
GyY=Wire.read()<<8|Wire.read();
// 0x47 (GYRO_ZOUT_H) & 0x48 (GYRO_ZOUT_L)
GyZ=Wire.read()<<8|Wire.read(); Serial.print("AcX = "); Serial.print(AcX); Serial.print(" | AcY = ");
Serial.print(AcY); Serial.print(" | AcZ = "); Serial.print(AcZ);
//equation for temperature in degrees C from datasheet Serial.print(" | Tmp = ");
Serial.print(Tmp/340.00+36.53); Serial.print(" | GyX = "); Serial.print(GyX);
```

```
Serial.print(" | GyY = "); Serial.print(GyY); Serial.print(" | GyZ = "); Serial.println(GyZ); delay(333);
}
```

Lesson 31 Sensor PIR HC-SR501

Resumen

En esta lección aprenderás a utilizar el detector de movimiento PIR con el UNO.

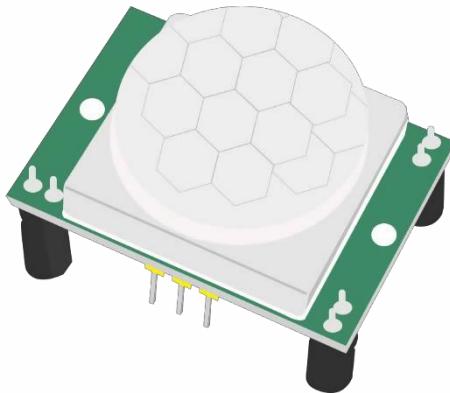
El UNO es el corazón de este proyecto. Escucha al sensor PIR y cuando se detecta movimiento, le indica al LED que se ilumine o se apague.

Componentes Requeridos:

1 x Elegoo Uno R3

1 x Sensor de movimiento PIR HC-SR501

3 x cables F-M (Cables DuPont Hembra a Macho)

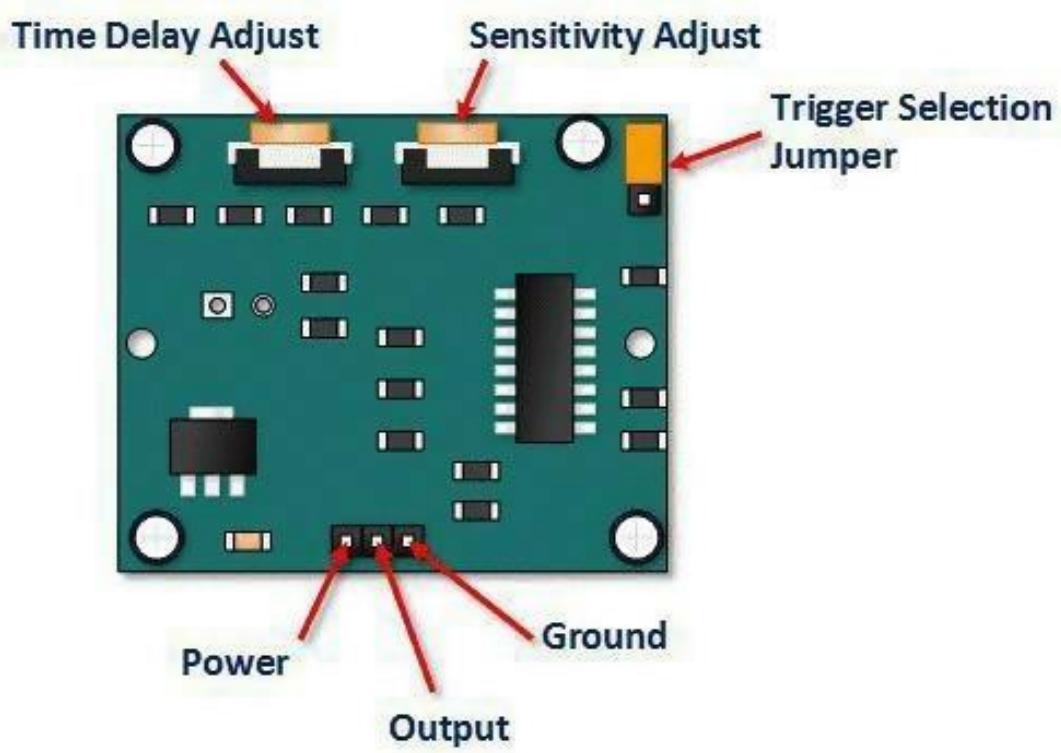


Introducción de Componente

Sensor PIR:

Los sensores PIR son más complicados de explicar que los demás sensores de este tutorial (tales como fotoceldas, FSRs e interruptores de inclinación) porque hay muchas variables que afectan las salidas y entradas de los sensores.

Los sensores PIR tienen dos ranuras. Cada una de ellas está fabricada de un material especial que es sensible a la IR. Los sensores pueden ver hasta cierta distancia (lo que constituye la sensibilidad del sensor). Cuando el sensor está inactivo, ambas ranuras detectan la misma cantidad de IR, que sería la propia del ambiente donde está (emitida por las paredes o captada en espacios abiertos). Cuando un cuerpo caliente como el de una persona o un animal pasa cerca del sensor, será captado por la primera mitad del sensor PIR, lo que causa un diferencial positivo entre las dos mitades. Cuando el cuerpo caliente se aleja del área, pasa lo contrario y se genera un diferencial negativo. Estos cambios de pulso son detectados por el módulo.



| Pin o Control | Función |
|-----------------------------------|--|
| Ajuste del tiempo de retardo | Configura por cuanto tiempo permanece alta la salida después de detectar movimiento.... Ese tiempo puede estar entre 5 segundos a 5 minutos. |
| Ajuste de sensibilidad | Configura el rango de detección.... de 3 a 7 metros |
| Jumper de selección de disparador | Configura si el disparador es sencillo o repetible. |
| Pin de Tierra | Entrada de Tierra |
| Pin de salida | Low cuando no se detecta movimiento. High cuando se detecta. High es 3.3V |

Descripción Funcional del HC SR501 PIR

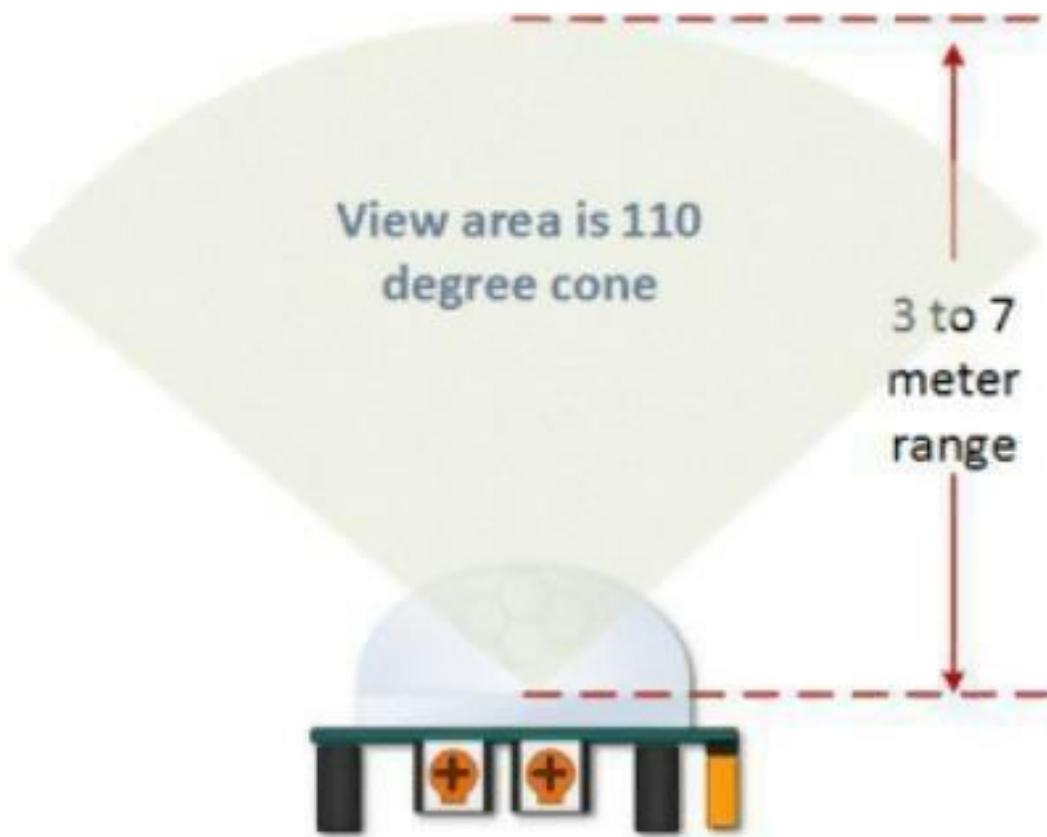
El SR501 detectará cambios infrarrojos y si los interpreta como movimiento, entregará una salida baja. Lo que se interprete o no como movimiento depende en buena medida a los ajustes y configuraciones que realice el usuario.

Inicialización del dispositivo

El dispositivo requiere casi un minuto para inicializar. Durante este periodo puede generar falsas medidas. Se debe tomar esto en cuenta para el circuito y la lógica del controlador.

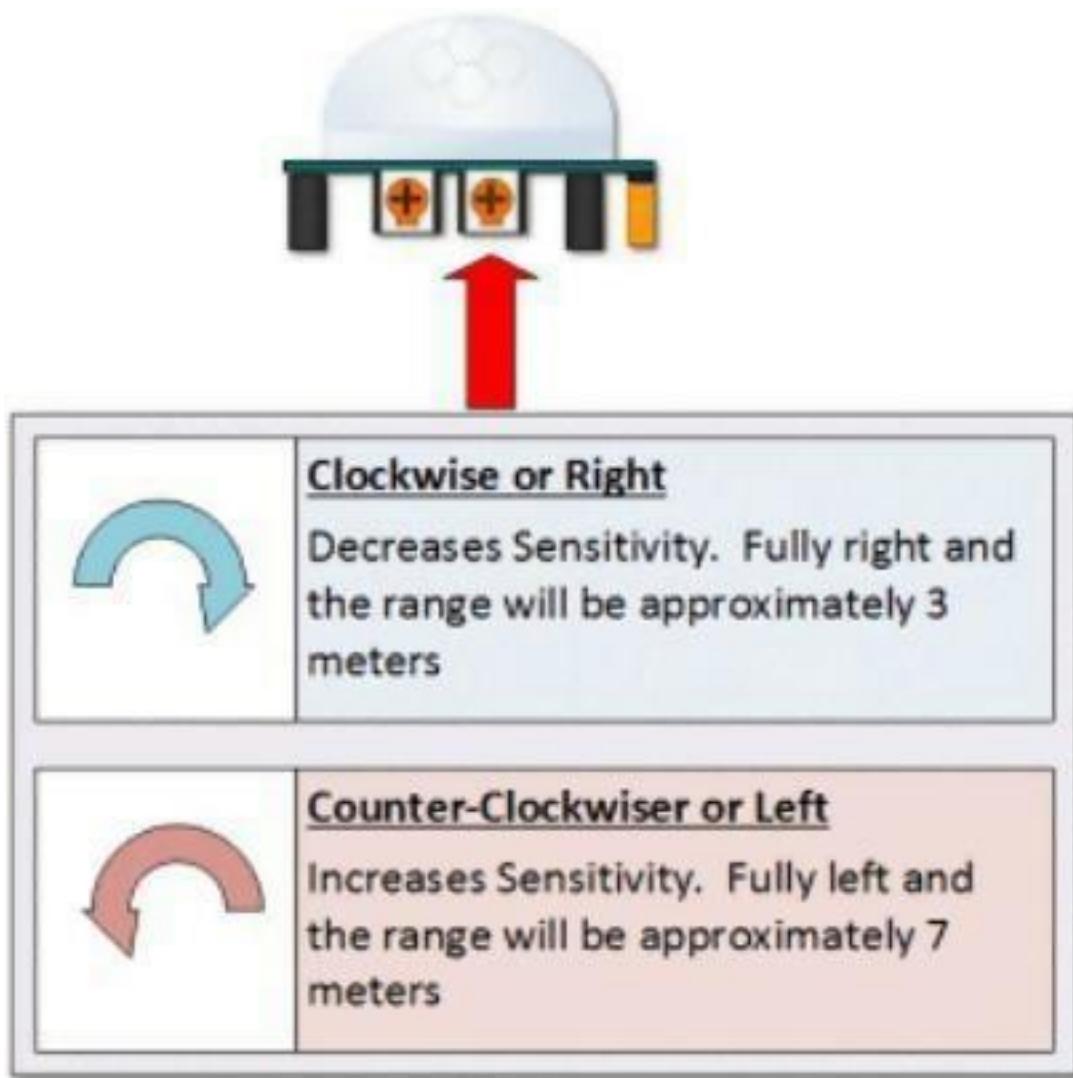
Área de detección del dispositivo

El dispositivo detectará movimiento dentro de un cono de 110 grados con un rango de 3 a 7 metros.



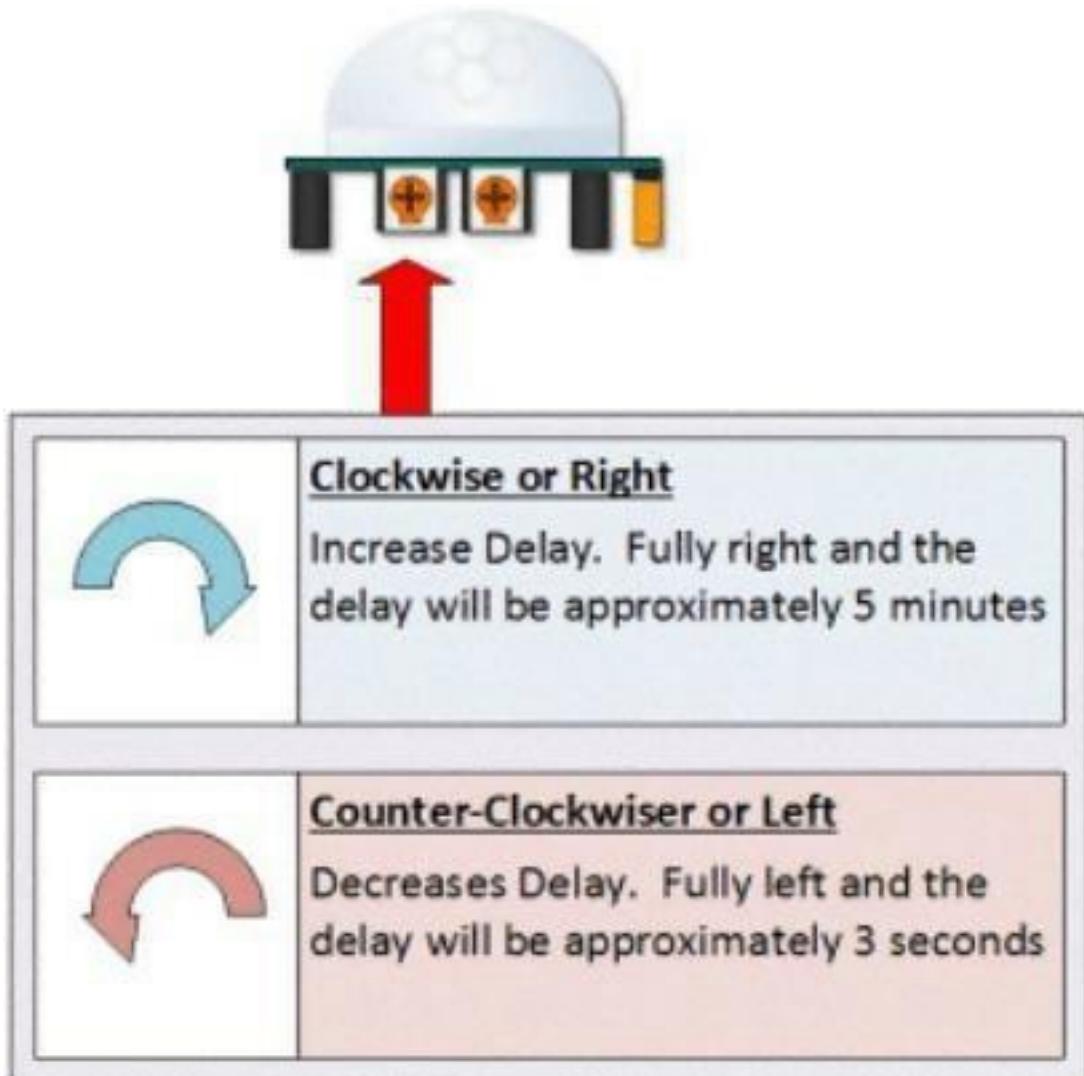
Área de visión del HC SR501

Rango PIR (sensibilidad ajuste de acuerdo a lo mencionado, en un rango de aproximadamente 3 a 7 metros. La ilustración de abajo muestra este ajuste.



Ajuste del tiempo de retraso y sensibilidad del HC SR50

El ajuste de tiempo de rechazo determina por cuanto tiempo la salida del sensor PIR permanecerá alta después de haber detectado movimiento. Ese tiempo puede estar entre 5 segundos a 5 minutos.



Ajuste del tiempo de retraso del HC SR501

3 segundos apagado después que se completa el tiempo de retraso – IMPORTANTE

La salida de este dispositivo se apagará por 3 segundos aproximadamente después que se complete el tiempo de retraso. En otras palabras, se bloquea toda detección de movimiento durante ese periodo de tiempo.

Por ejemplo:

Imagina que estás operando en modo de disparador sencillo y tu tiempo de retraso es de 5 segundos.

Cuando el PIR detecte movimiento, mantendrá la salida activada por 5 segundos.

Después de ese tiempo, el PIR mantendrá la salida apagada por 3 segundos.

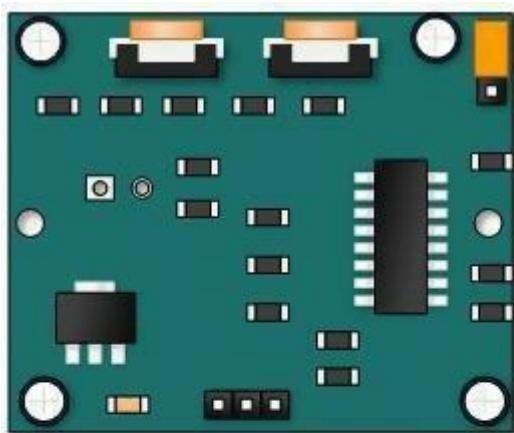
En ese periodo de tres segundos, el PIR no detectará movimiento. Después de ese tiempo el PIR volverá a censar y si detecta movimiento se activará una vez más por 5 segundos.

Jumper de selección de modo de disparador

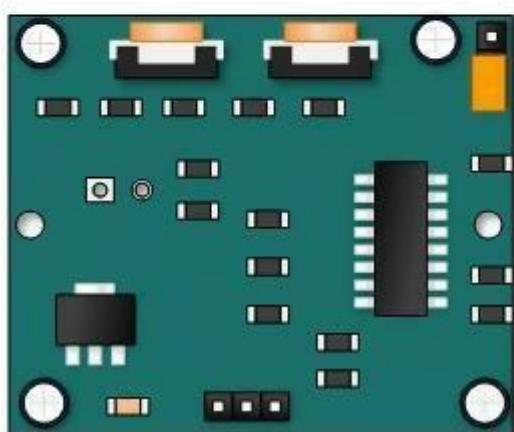
Este jumper te permite seleccionar entre un disparador sencillo y uno con repetición. Dependiendo de este jumper, variará el tiempo de retraso configurado.

Disparador sencillo – El tiempo de retraso comienza inmediatamente cuando se detecta el movimiento.

Disparador repetible – Cada movimiento detectado reinicia el tiempo de retraso. Eso significa que el tiempo de retraso comienza con el último movimiento detectado.



Single Trigger Mode – Time
Delay is started immediately upon detecting motion.
Continued detection is blocked



Repeatable Trigger Mode –
Time Delay is re-started every time motion is detected.

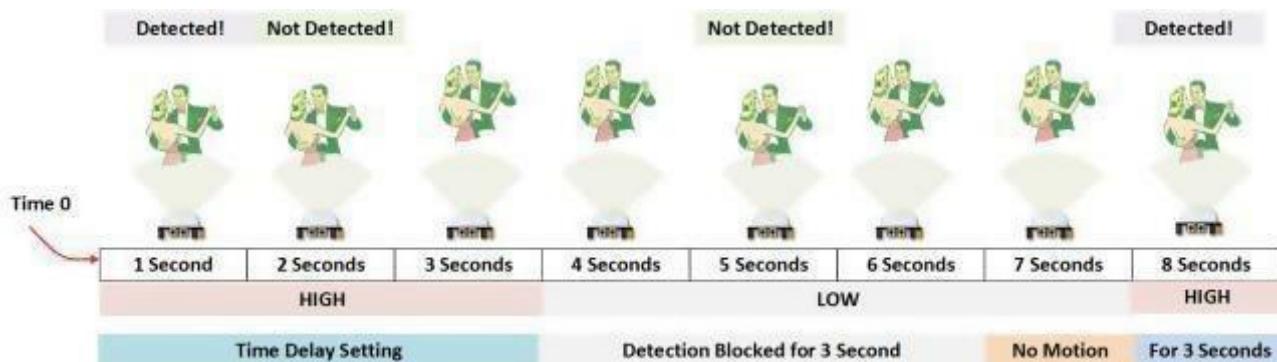
Ejemplos de aplicación del HC-SR501 Sala de Baile

Imagina que deseas controlar la iluminación de una sala de baile basándote en donde están bailando las personas. Entender la manera en que trabajan los disparadores y el tiempo de retraso es importante para controlar la iluminación de la manera que quieras.

Ejemplo Uno

En este primer ejemplo, el tiempo de retraso se configura en tres segundos y el modo de disparador en sencillo. Como puedes verlo en la ilustración de abajo, no siempre se detecta movimiento. De hecho, existe un periodo de cerca de seis segundos en los que no se puede detectar ningún movimiento.

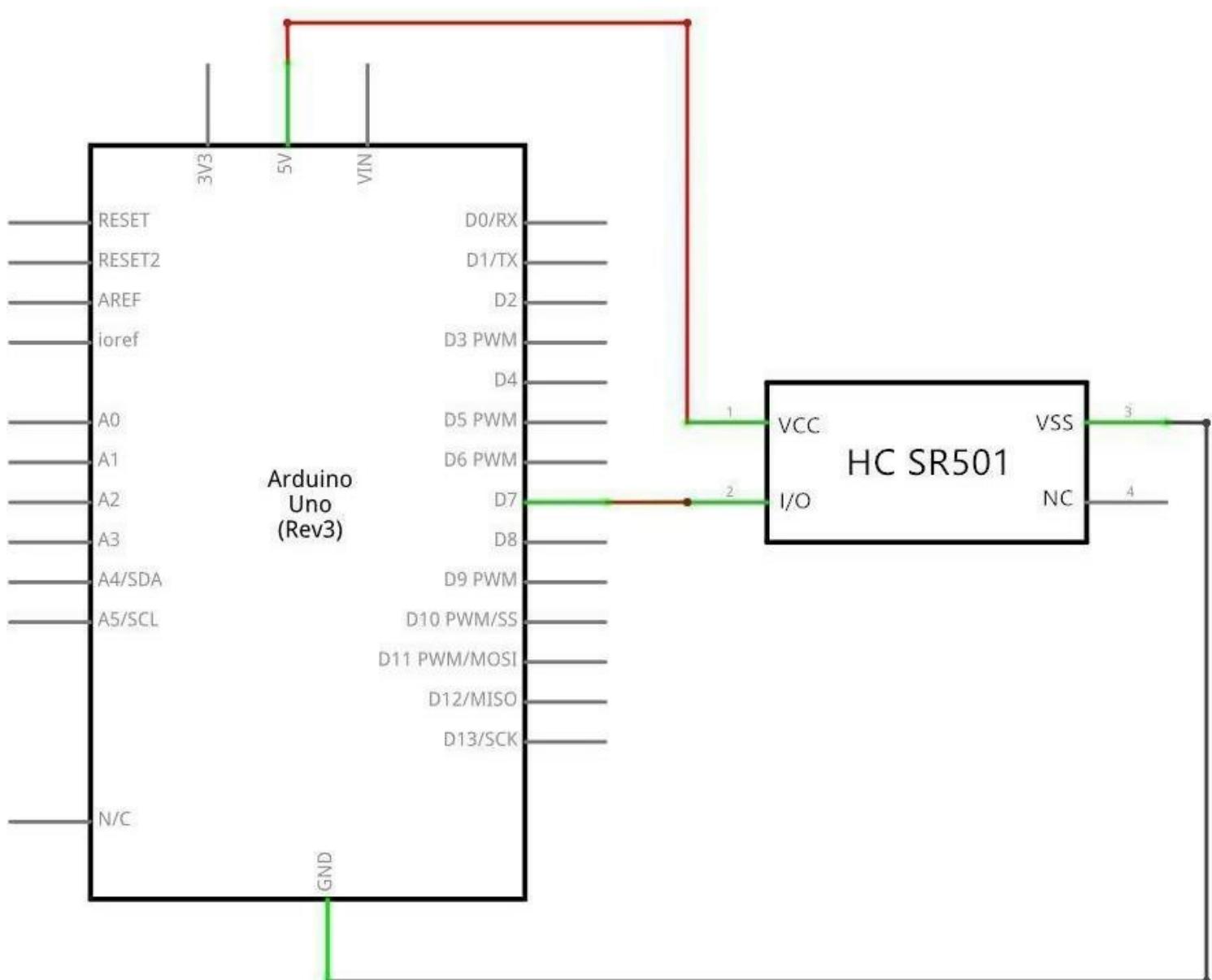
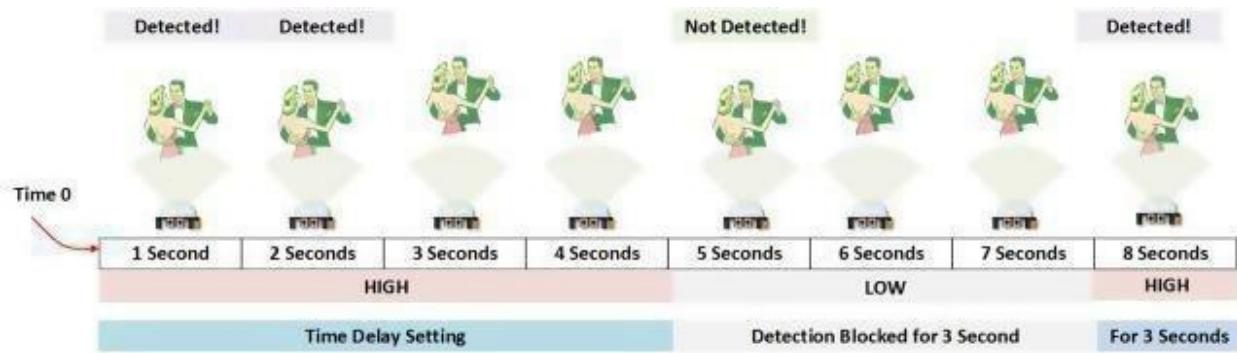
Puedes hacer clic en la imagen para agrandarla sin problemas

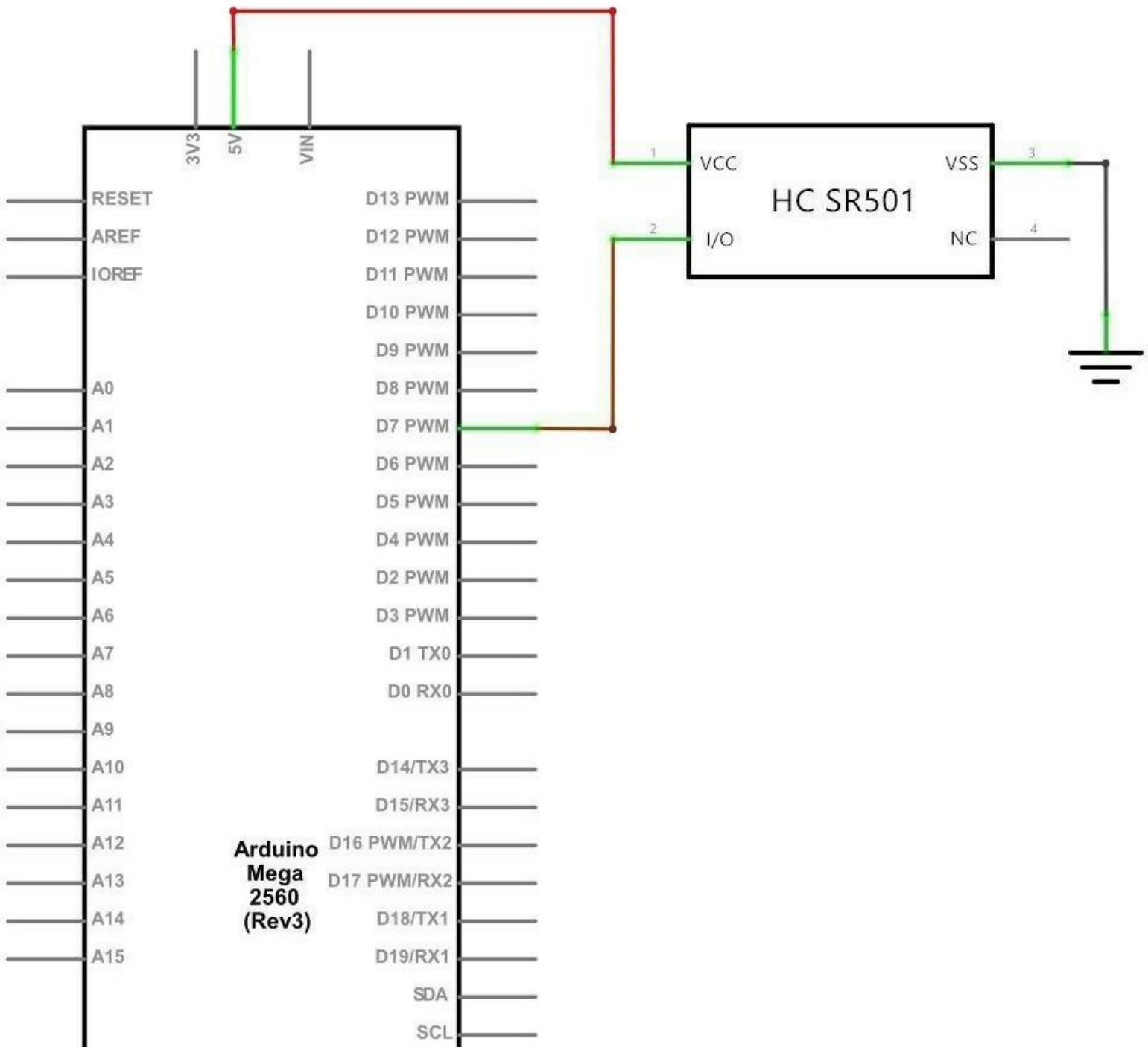


Ejemplo dos

En el siguiente ejemplo, el tiempo de retraso está en tres segundos y el disparador en repetible. En la ilustración de abajo, puedes ver que se reinicia el tiempo de retraso. La detección aún se bloquea por tres segundos.

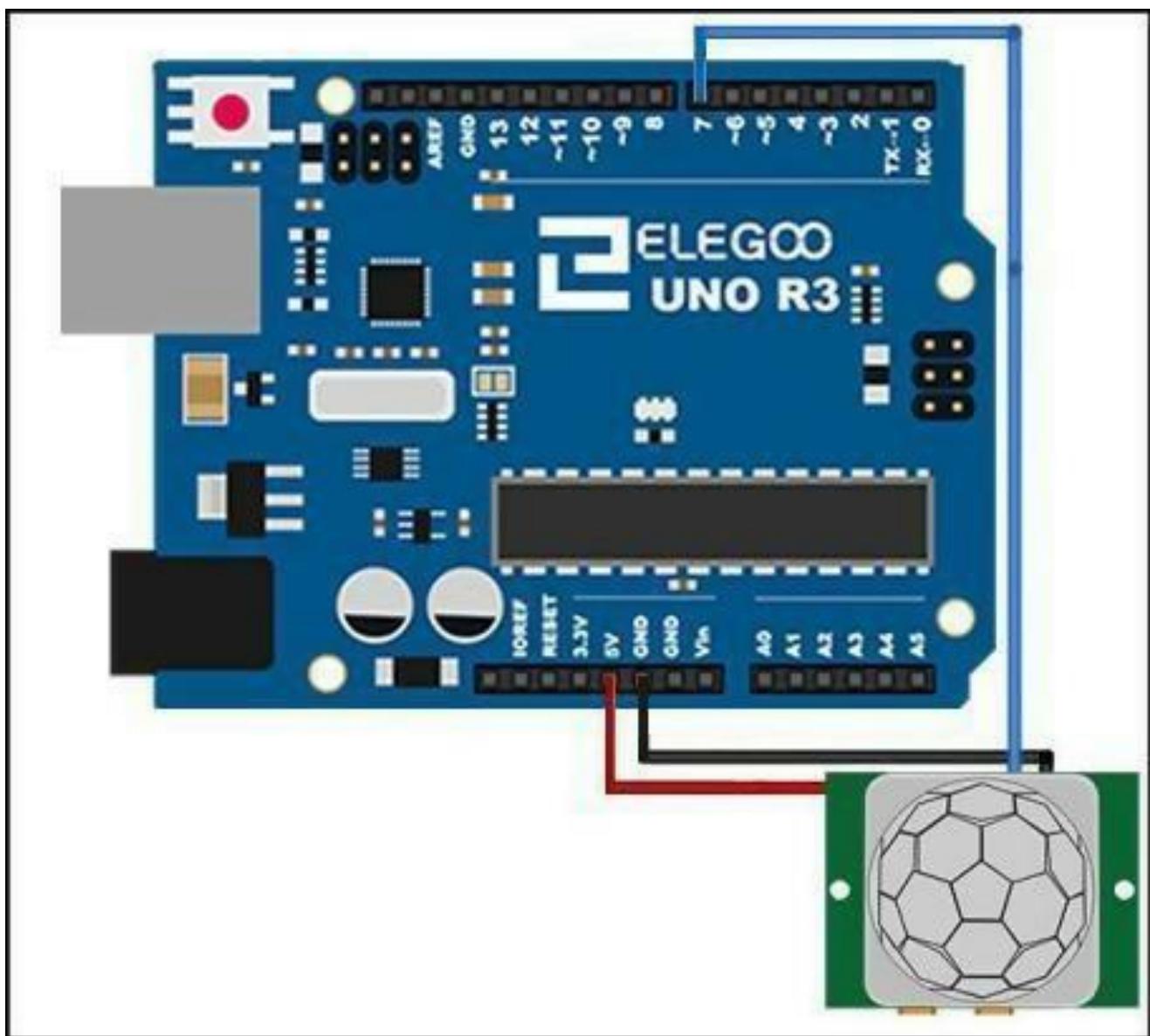
Como lo mencione anteriormente, podrías anular ese periodo de bloqueo de 3 segundos con un código creativo. Pero ten en cuenta que algunos equipos electrónicos no trabajan bien si se les enciende y apaga rápidamente. Los tres segundos sirven para darles descanso a esta clase de equipos.

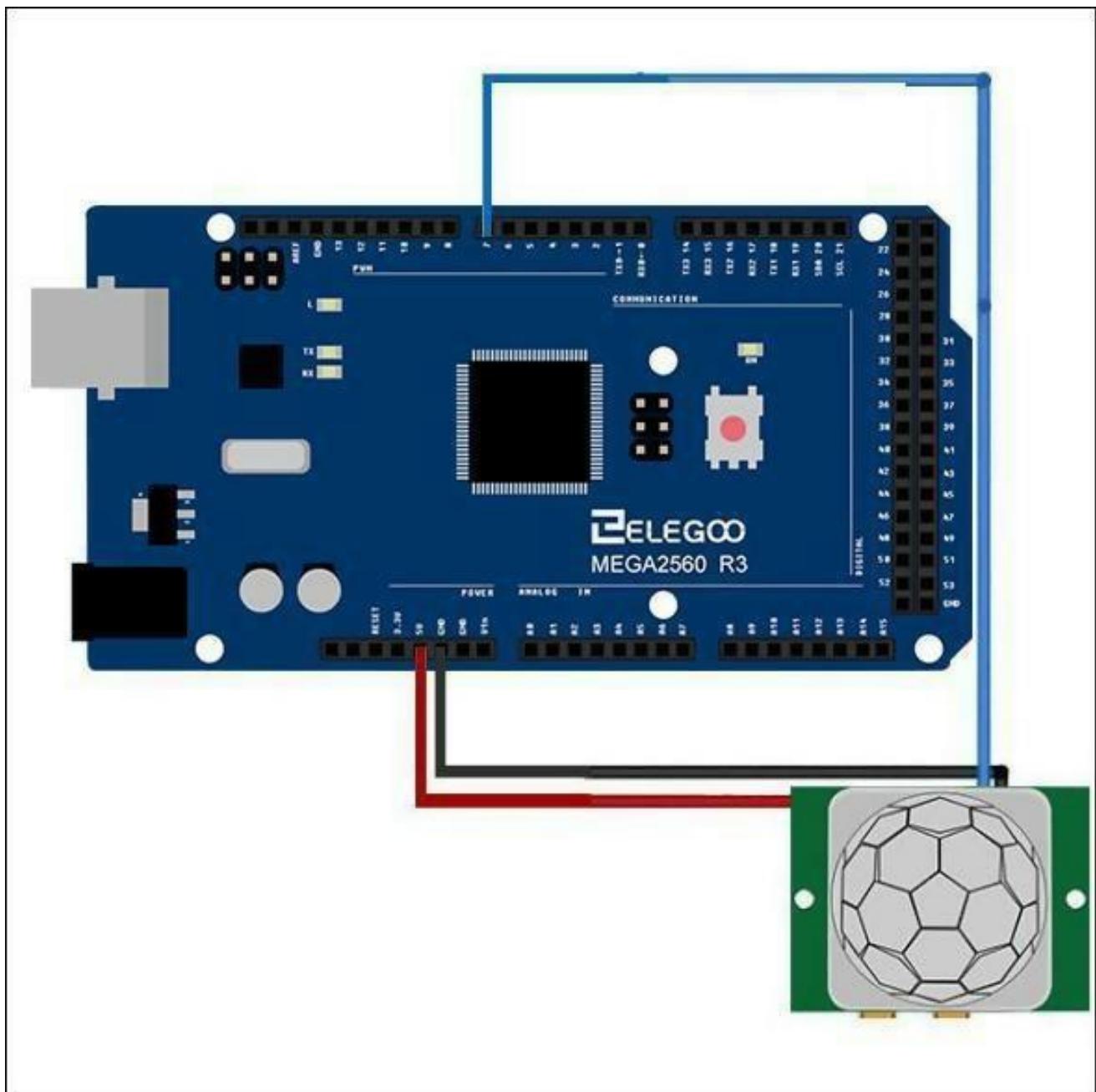




<http://www.elegoo.com>

Diagrama de cableado





Conectar sensores PIR a microcontroladores es bastante sencillo. El PIR actúa como una salida digital

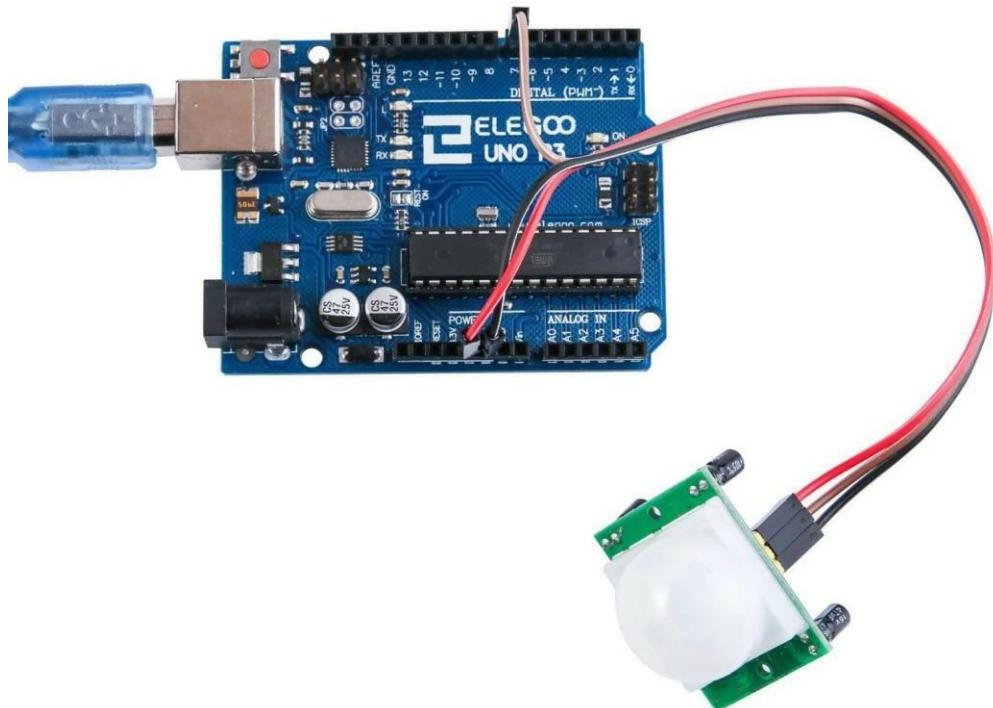
Conectar sensores PIR a microcontroladores es bastante sencillo. El PIR actúa como una salida digital así que solo necesitarás escuchar al pin alto (significa detectado) y al bajo (significa no detectado). Si quieres que el disparador se repita asegúrate de colocar el jumper en la posición H! Energiza el PIR con 5V y tierra Luego conecta la salida a un pin digital, en este caso, pin 2

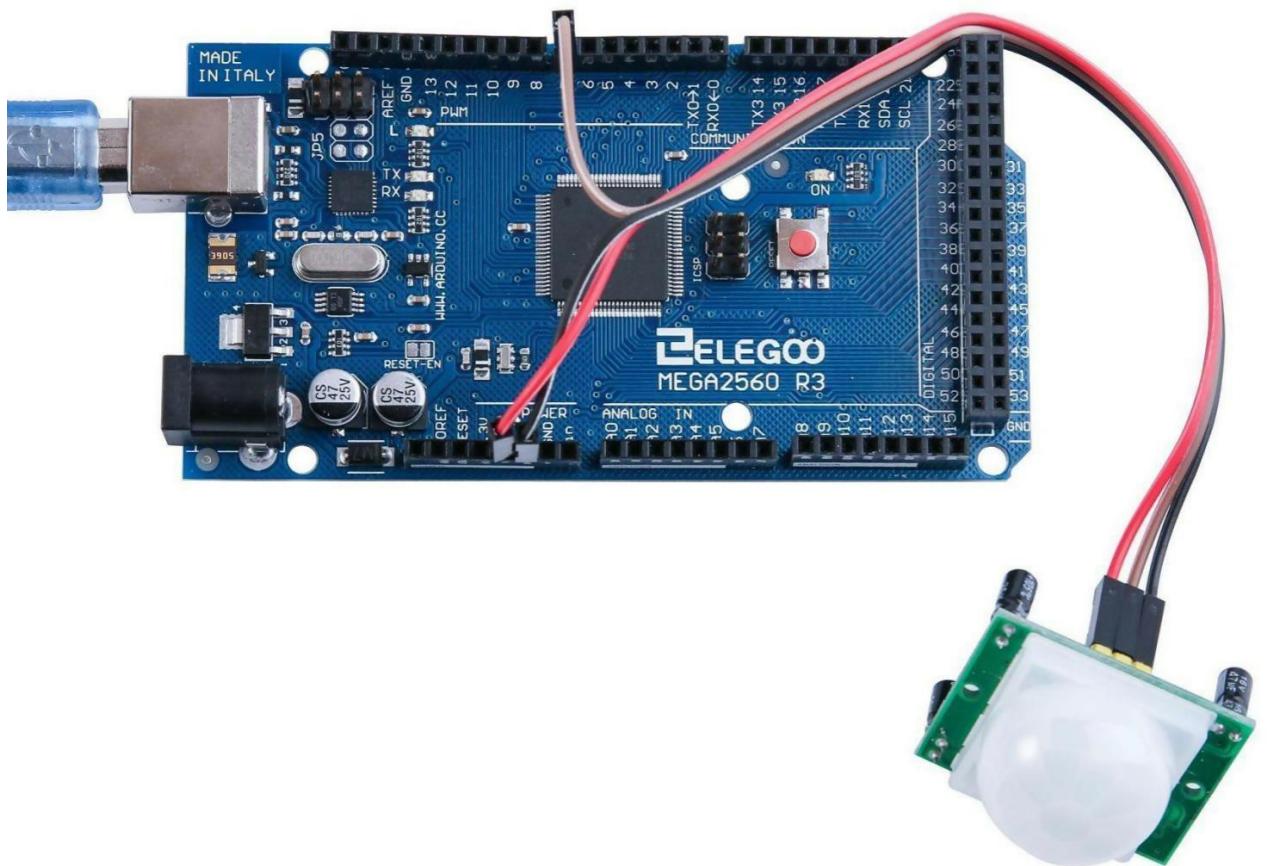
Código

Después de realizar el cableado, por favor abre el programa en la carpeta de código Lección 31 Sensor PIR HC-SR501 y haz clic en UPLOAD para cargar el programa. Si hay algún error, consulta los detalles en la lección 2 para cargar el programa.

Este sketch solo enciende el LED de tu Arduino conectado al pin 13 cuando se detecta movimiento. Recuerda tomar precauciones en cuanto al minuto de inicialización que la aplicación que desarrolles.

Imagen ejemplo





A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
// Selecciona el pin para el LED int ledPin  
// Selecciona el pin para el Sensor PIR int ledPin = 7;  
//Comienza asumiendo que no se detecta movimiento int pirState = LOW;  
//Variable para leer el estatus del pin int val = 0;  
  
void setup() {  
    // Define al LED como una salida pinMode(ledPin, OUTPUT);  
    // Declara al sensor como una entrada pinMode(inputPin, INPUT); Serial.begin(9600);  
}  
  
void loop(){  
    // read input value  
    val = digitalRead(inputPin); if (val == HIGH) {  
        // Verifica si la entrada está HIGH  
        // Enciende el LED digitalWrite(ledPin, HIGH); if (pirState == LOW) {  
        // Lo hemos encendido Serial.println("Motion detected!");  
        // Solo imprimir el cambio de estado, no el estado como tal pirState = HIGH;  
    }  
    }  
    else  
    {  
  
        // Apaga el LED digitalWrite(ledPin, LOW); if (pirState == HIGH){  
        // Lo hemos apagado Serial.println("Motion ended!");  
        // Solo imprimir el cambio de estado, no el estado como tal pirState = LOW;  
    }  
}
```

Lección 32 Módulo de Sensor de Nivel de Agua

Resumen

En este experimento, aprenderemos a utilizar el Módulo de Sensor de Nivel de Agua. Este módulo puede percibir la profundidad del agua y como componente central tiene un circuito amplificador hecho en base a transistores. Cuando entra en contacto con el agua, el circuito presenta un cambio en su resistencia, directamente proporcional a la profundidad del agua. En base a esto, se convierte la lectura de profundidad a una señal eléctrica que sirve para indicarle al UNO R3 el nivel a través de la función ADC.

Componentes requeridos:

- 1x Elegoo Uno R3
- 1x Módulo Sensor de Nivel de Agua
- 3 x cables F-M (Cables DuPont Hembra a Macho)



Introducción de Componente

Sensor de Agua

Un bloque sensor de agua diseñado para detectar agua de lluvia, nivel de un recipiente o tanque, fugas de agua, etc. Está compuesto principalmente de tres partes: Un conector electrónico, un resistor de $1M\Omega$ y varias líneas alambres conductores desnudos.

Este sensor funciona por medio de una serie de trazos expuestos conectados a tierra. Entrelazados con los trazos aterrados, se encuentran otros trazos que sirven para hacer la medición. Se encuentran otros trazos que sirven para hacer la medición.

Los trazos de medición tienen un resistor pull-up de $1 M\Omega$. Ese resistor mantiene el valor del trazo en High hasta que una gota de agua pone en corto el trazo de medición con uno de tierra. Aunque no lo creas, este circuito utiliza los pines digitales I/O de tu UNO R3.

si lo quieres como interruptor o entre los analógicos si quieras conocer el nivel propiamente dicho.

El equipo deduce el nivel a través de una serie de cables paralelos expuestos en diferentes puntos para dar una idea de la altura en la cual censan el agua, siendo capaces de estimar el nivel de la misma en un recipiente.

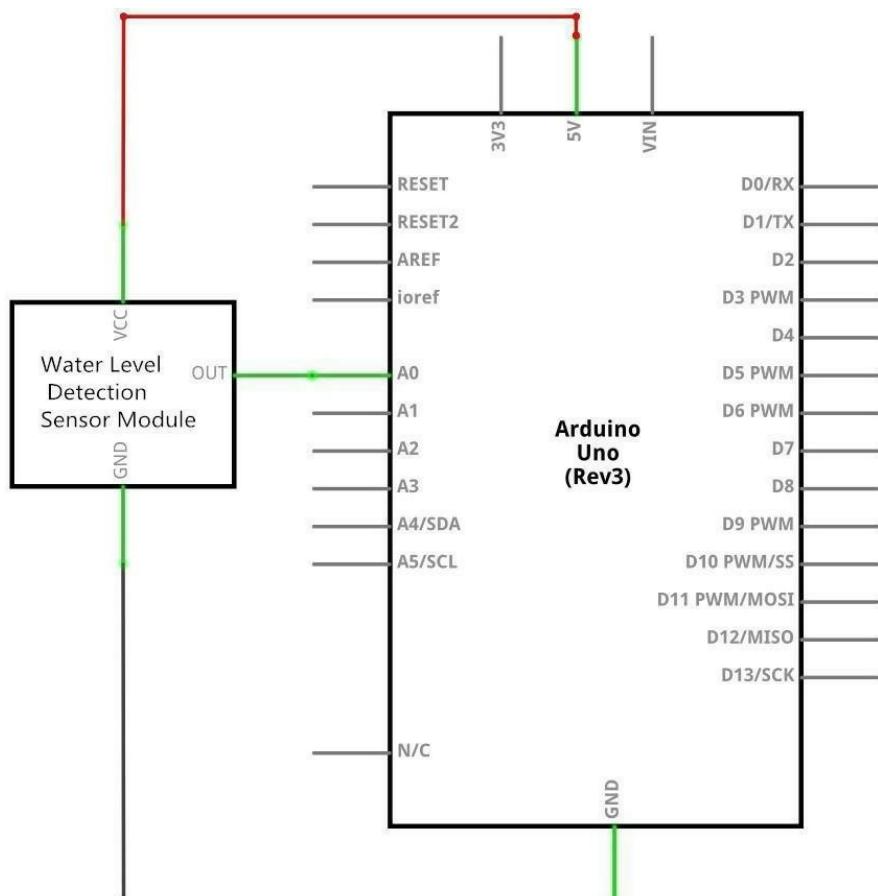
analog value can directly of water level alarm.

be used in the program function, then to achieve the function

Tiene un bajo consumo de energía y alta sensibilidad. Características

1. Voltaje de trabajo: 5V
2. Corriente de trabajo: <20ma
3. Interfaz: analógica
4. Ancho de detección: 40mm×16mm
5. Temperatura de trabajo: 10°C~30°C 6、 Señal de salida de voltaje: 0~4.2V

Esquema de Conexión



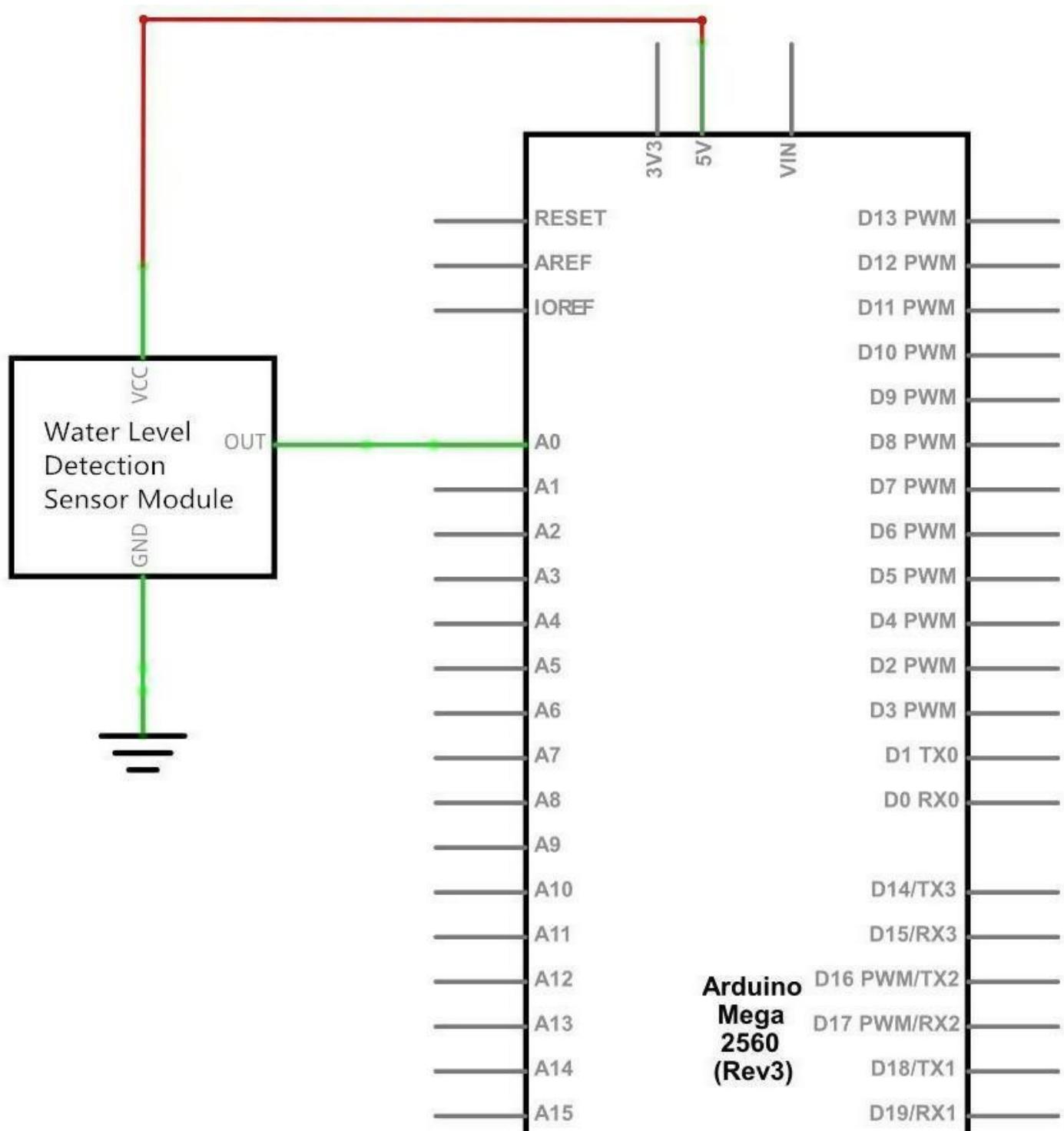
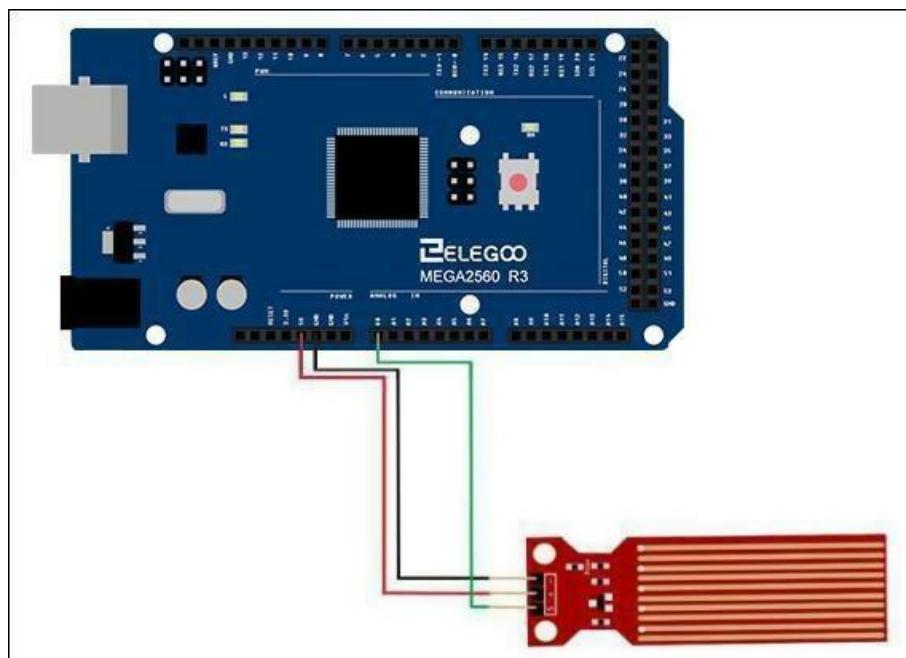
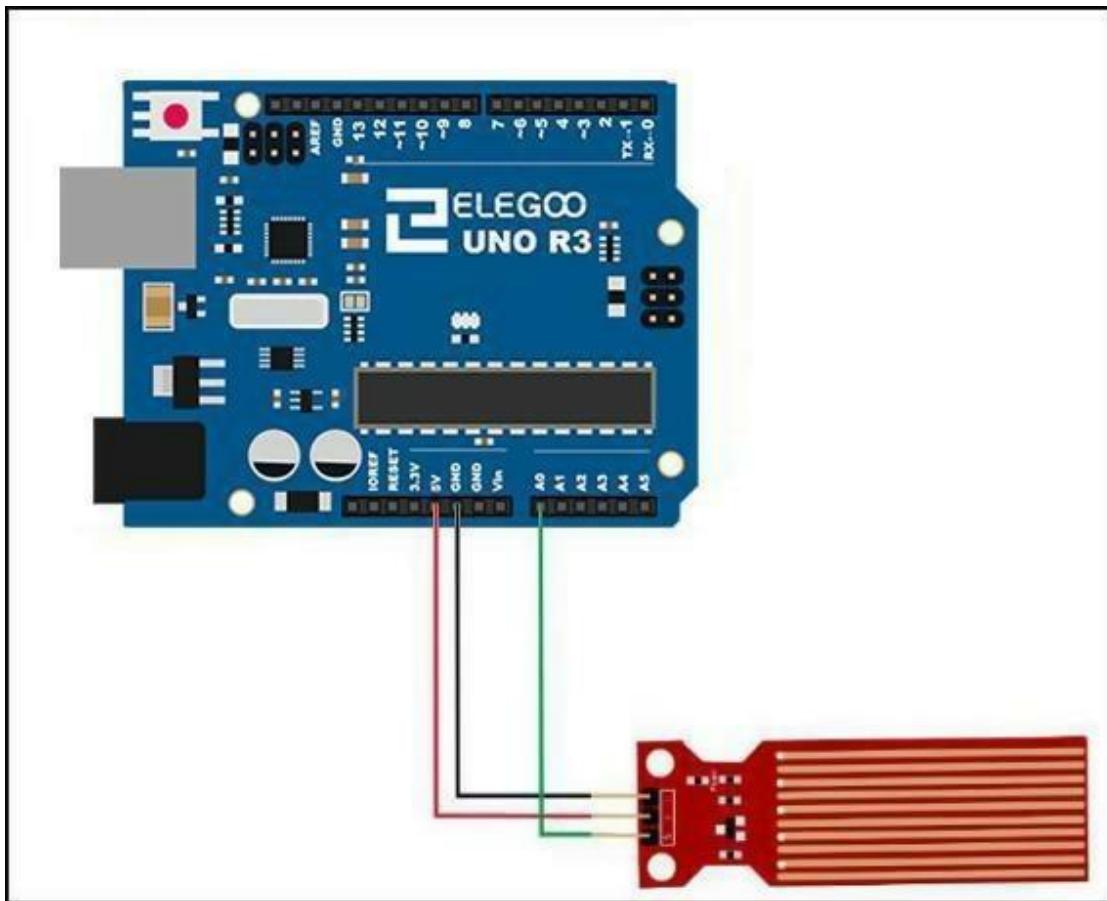


Diagrama de cableado

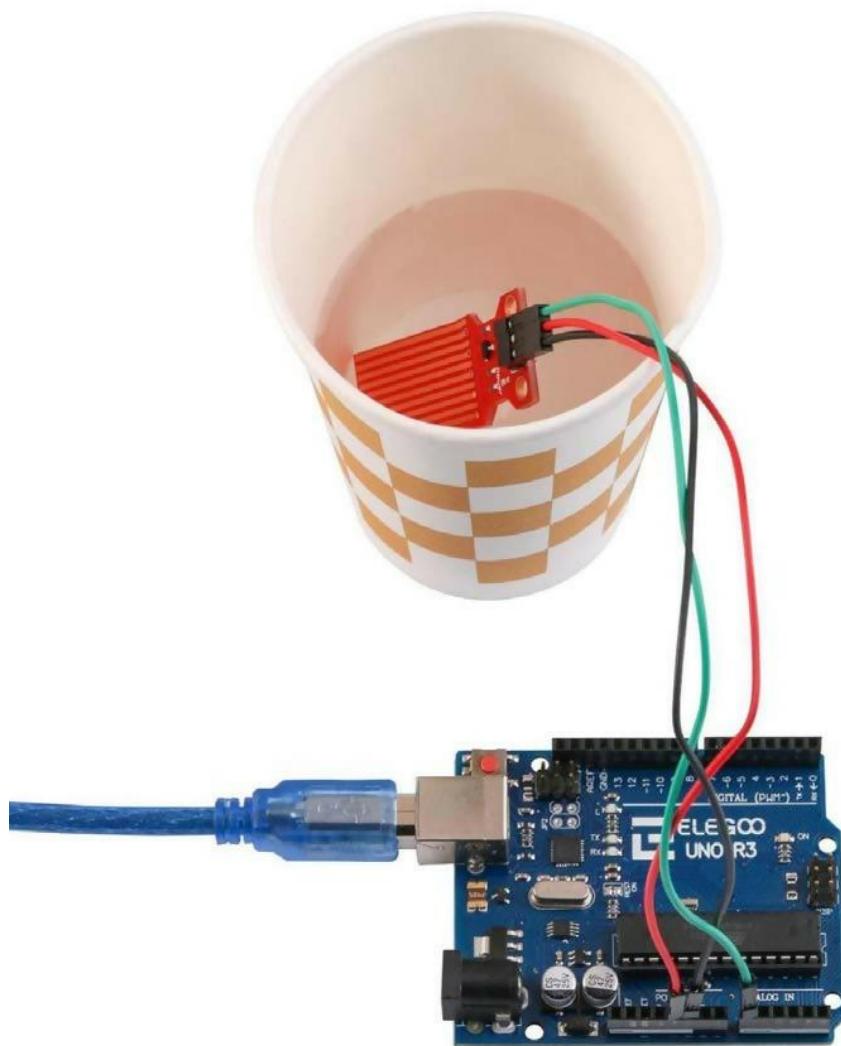


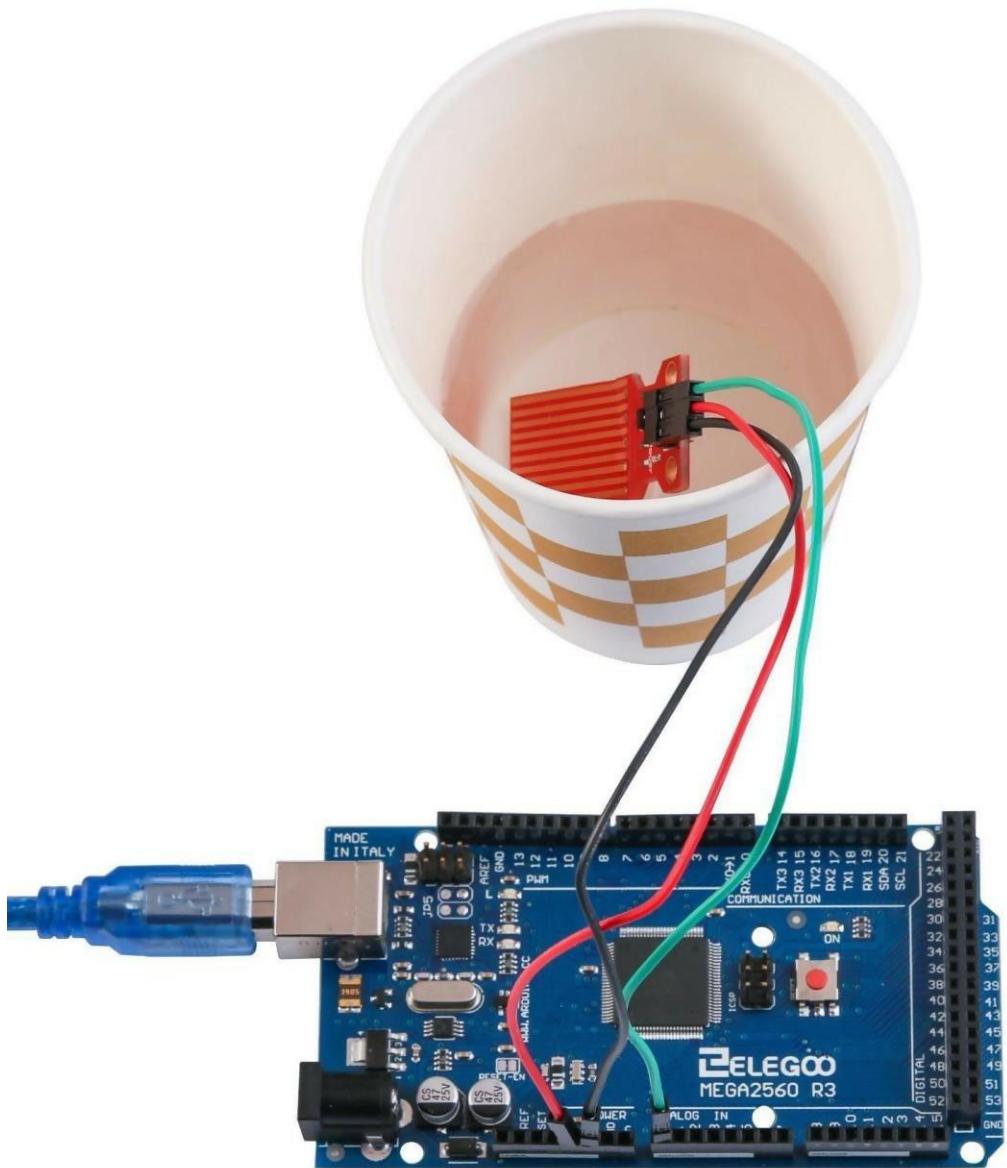
Tips de cableado: La fuente de poder (+) va conectada con el 5V del UNO R3 y el electrodo de tierra (-) va conectado a GND. La señal de salida (S) se conecta a los puertos (A0-A5) de entrada analógicas al UNO R3. Se debe definir el uso de los mismos en el código.

Código

Después de cablear, por favor abre el programa en la carpeta de código Lección 32 Módulo Sensor de Nivel de Agua y haz clic en UPLOAD para cargar el programa. Si hay algún error, consulta los detalles en la lección 2 para cargar el programa.

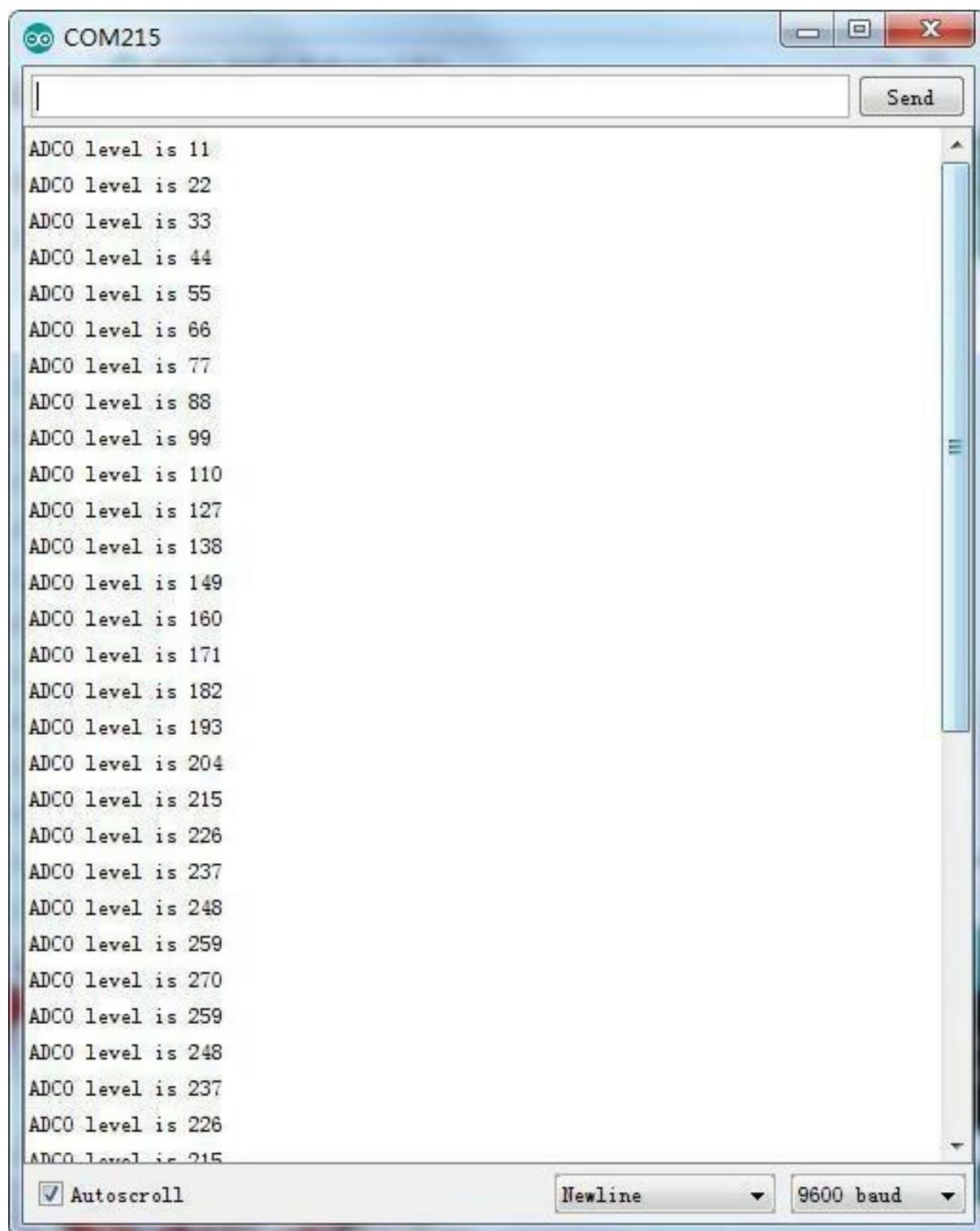
Imagen ejemplo





Luego abre el monitor, para ver los siguientes datos:

Haz clic en el Monitor Serial para encenderlo Los básico a saber del monitor serial se enseña en la lección 1



A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
int adc_id = 0;
int HistoryValue = 0; char printBuffer[128];

void setup()
{
Serial.begin(9600);
}

void loop()
{
int value = analogRead(adc_id); // get adc value

if(((HistoryValue>=value) && ((HistoryValue - value) > 10)) || ((HistoryValue<value) && ((value - HistoryValue) > 10)))
{
sprintf(printBuffer,"ADC%d level is %d\n",adc_id, value);

Serial.print(printBuffer); HistoryValue = value;
}
}
```

Lección 33 Módulo de Reloj en Tiempo Real

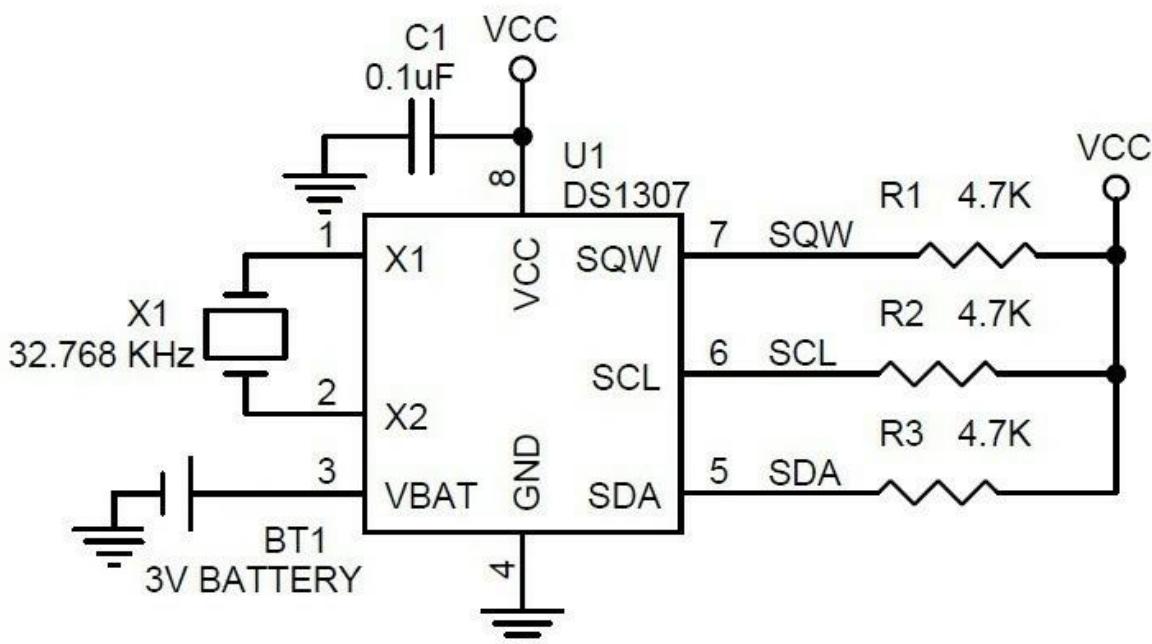
Resumen

En esta lección, aprenderás a utilizar el módulo RTC, el DS1307 es un chip de reloj de tiempo real de bajo consumo. La dirección y datos se transfieren de manera serial a través de I2C o a través de una conexión directa con tres cables de datos. El DS1307 provee información del tiempo en segundos, minutos, horas, días, fecha, mes y año. La operación de llevar el tiempo continúa mientras se cuente con un voltaje de alimentación de respaldo.

Componentes requeridos:

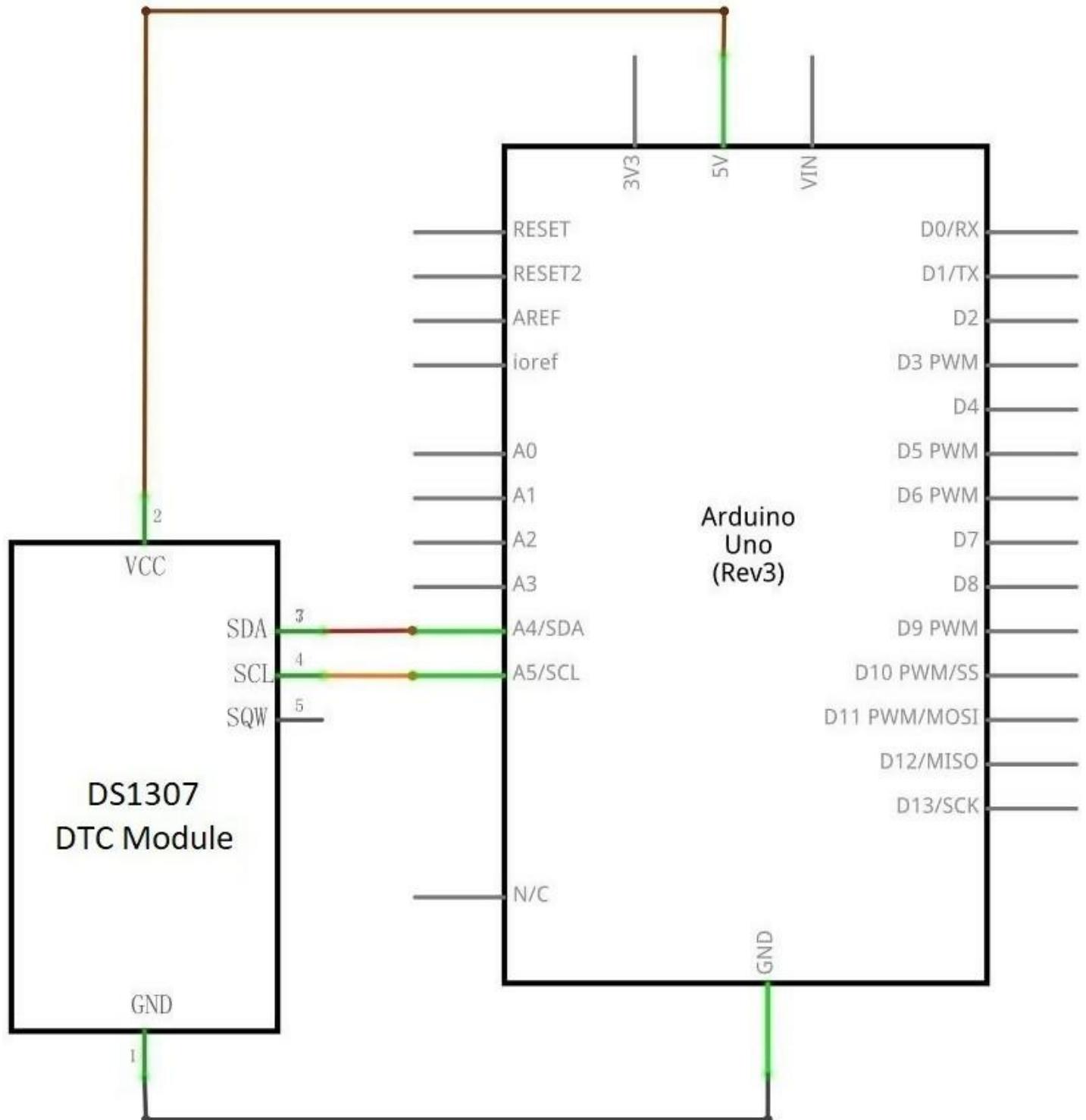
- 1x Elegoo Uno R3
- 1x Módulo RTC DS1307
- 4x Cables F-M (Cables DuPont Hembra a Macho)

Introducción de componente



NOTICE: If there is the left version of module in the Kit, don't worry, its function and pin names are the same as the new version. Just follow the wiring diagrams and sketch in the tutorial below to get it working.

Esquema de Conexión



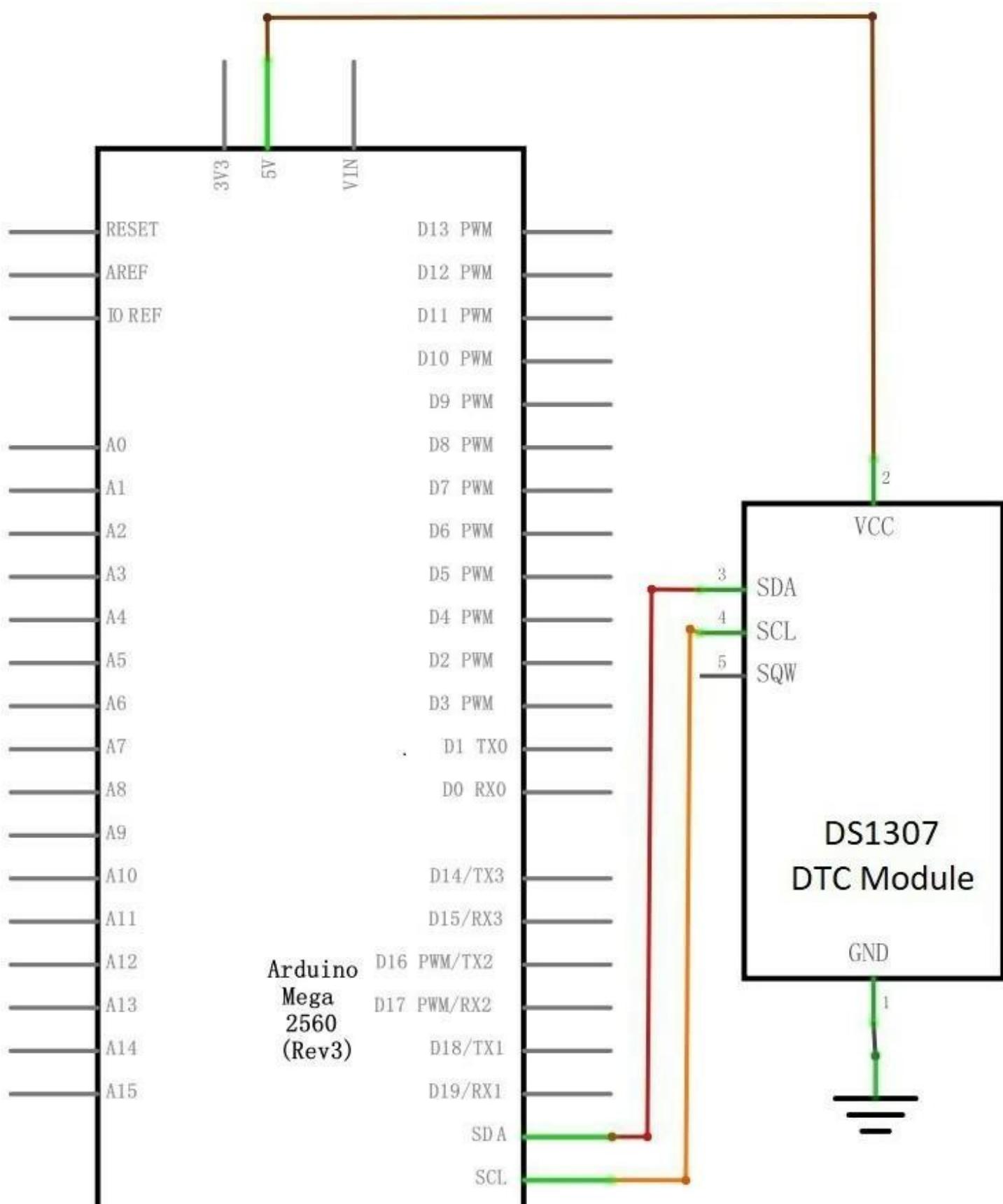
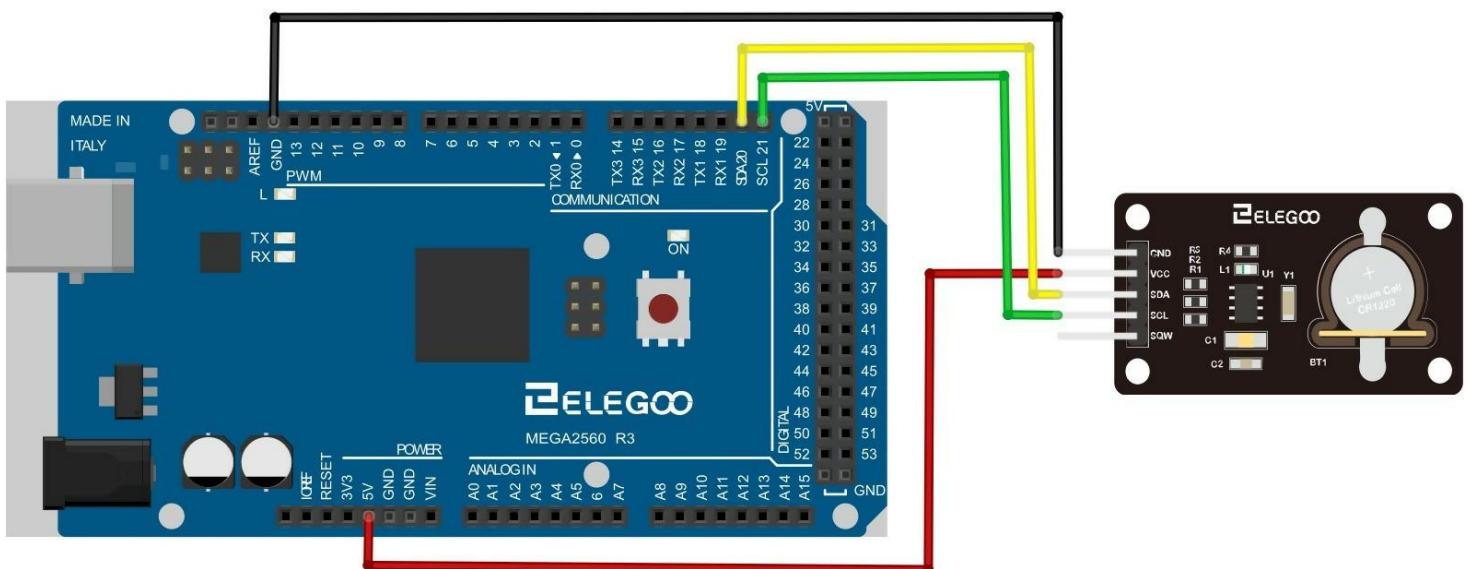
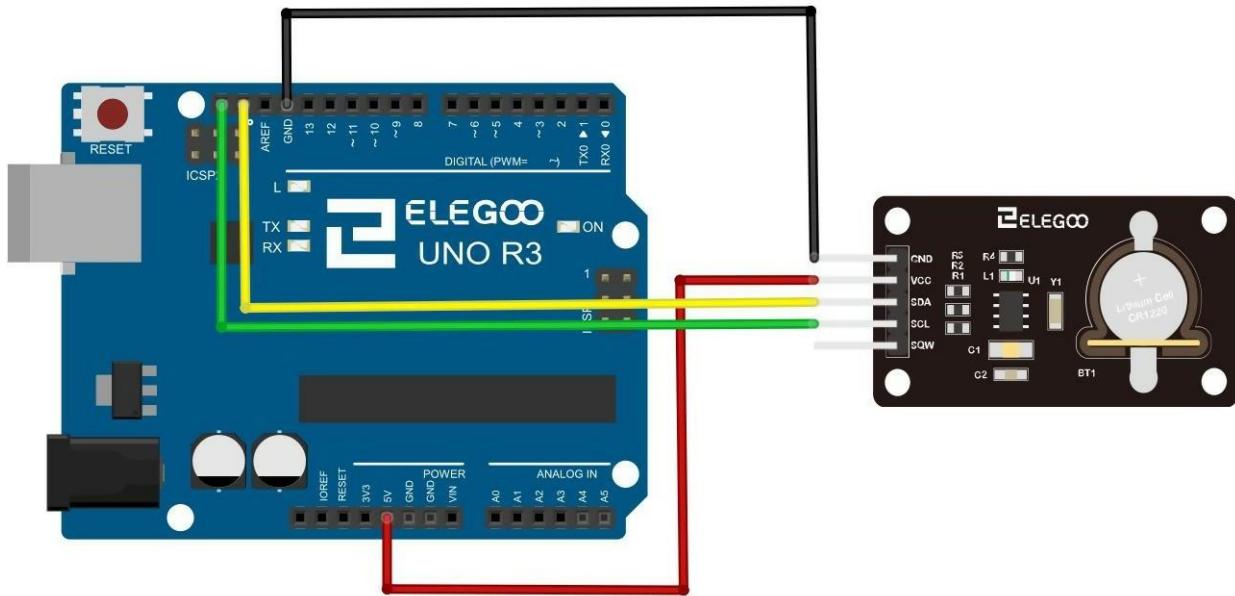


Diagrama de cableado



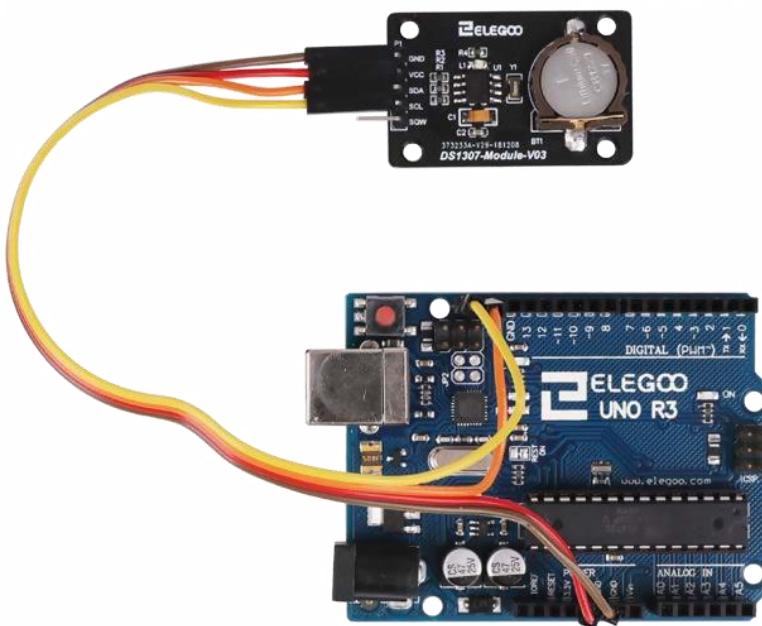
Configúralo de acuerdo a la siguiente Imagen.

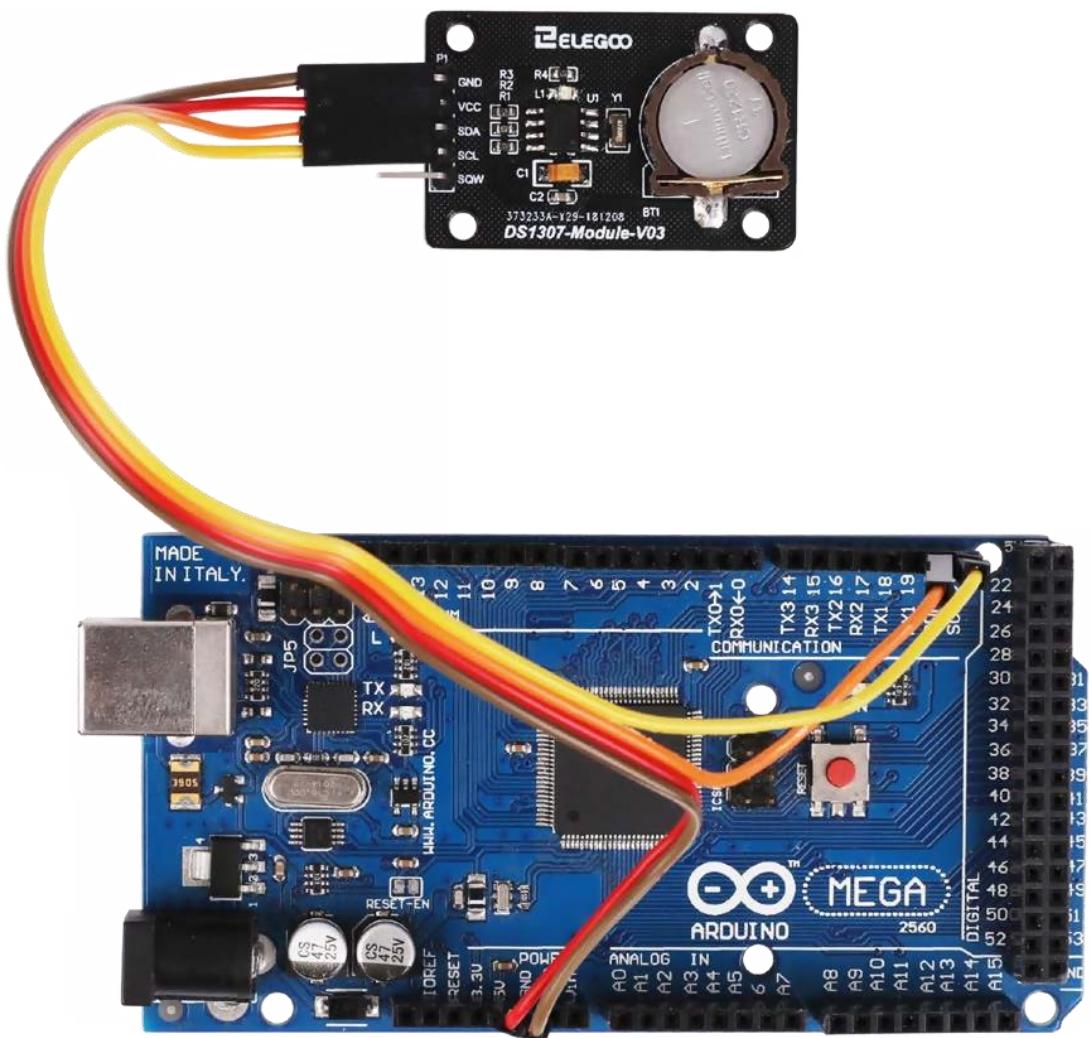
Nota: Ignora el pin SQW; no se necesita para esta lección. Conecta el pin SCL a tu UNO R3 en el puerto SCL y el pin SDA en el puerto SDA. El pin Vcc va a 5V y el GND a tierra.

Código

Después de realizar el cableado, por favor abre el programa en la carpeta de código Lección 33 Módulo de Reloj en Tiempo Real y haz clic en UPLOAD para cargar el programa. Si hay algún error, consulta los detalles en la lección 2 para cargar el programa. Antes de poder correrlo, asegúrate de tener la librería <DS1307> instalada, o reinstálala de ser necesario. De otra forma, tu código no funcionará. Para detalles del tutorial acerca de cómo cargar los archivos de librerías, consulta la lección 1.

Imagen ejemplo





Abre el monitor y verás como el módulo lee el tiempo como se ve abajo:

Haz clic en el Monitor Serial para encenderlo Los básicos a saber del monitor serial se enseña en la lección 1

The screenshot shows a Windows-style application window titled "COM8 (Arduino/Genuino Mega or Mega 2560)". The main text area displays a series of timestamp entries, each starting with "Raw data:". The timestamps are as follows:

```
Initialize DS3231
Raw data: 2018-12-11 18:22:3
Raw data: 2018-12-11 18:22:4
Raw data: 2018-12-11 18:22:5
Raw data: 2018-12-11 18:22:6
Raw data: 2018-12-11 18:22:7
Raw data: 2018-12-11 18:22:8
Raw data: 2018-12-11 18:22:9
Raw data: 2018-12-11 18:22:10
Raw data: 2018-12-11 18:22:11
Raw data: 2018-12-11 18:22:12
Raw data: 2018-12-11 18:22:13
Raw data: 2018-12-11 18:22:14
Raw data: 2018-12-11 18:22:15
Raw data: 2018-12-11 18:22:16
Raw data: 2018-12-11 18:22:17
```

At the bottom of the window, there are several control buttons: "Autoscroll" (checked), "Show timestamp" (unchecked), "Newline" (dropdown menu), "9600 baud" (dropdown menu), and "Clear output".

A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
#include <Wire.h> #include <DS3231.h>

DS3231 clock; RTCDateTime dt;

void setup()
{
    Serial.begin(9600);

    Serial.println("Initialize DS3231");
    // Inicialización del DS3231 clock.begin();

    // Se coloca el formato manualmente YYYY, MM, DD, HH, II, SS
    // clock.setDateTime(2016, 12, 9, 11, 46, 00);

    // Envía el tiempo de compilación al Arduino clock.setDateTime(  Fecha ,      Hora );
    /*ç
    Tips: este comando se ejecutará cada vez que el Arduino reinicie.
    Comentarios en esta línea para guardar en la memoria del módulo DS3231
    */
}

void loop()
{
    dt = clock.getDateTime();
    // For leading zero look to DS3231_dateformat example Serial.print("Raw data: ");
    Serial.print(dt.year);    Serial.print("-");    Serial.print(dt.month);    Serial.print("-");    Serial.print(dt.day);
        Serial.print(" ");    Serial.print(dt.hour);    Serial.print(":");    Serial.print(dt.minute);    Serial.print(":");
    Serial.print(dt.second); Serial.println("");
    delay(1000);
}
```

Lección 34 Módulo de Teclado

Resumen

En este proyecto, veremos como integrar un teclado con en UNO R3 para que pueda leer las teclas presionadas por un usuario.

Los teclados están en todo tipo de dispositivos tales como teléfonos celulares, máquinas de fax, hornos microondas, cerraduras de puertas, etc. Prácticamente están en todas partes. Muchísimos equipos electrónicos los utilizan para recibir las entradas de parte de los usuarios.

Así que saber conectar un teclado a un microcontrolador como UNO R3 es muy valioso a fin de construir muchos diferentes tipos de productos comerciales.

Cuando ya tengas listo el proyecto y todo esté conectado y programado, podrás presionar una tecla y ver el resultado en el monitor serial de tu computadora. Cada vez que presiones una tecla, podrá verse en el Monitor Serial. Por razones de simplicidad, comenzamos solo mostrando la tecla presionada en la computadora.

Para este proyecto, utilizamos un teclado estilo matriz. Este es un teclado que sigue un esquema de codificación que permite tener muchos menos pines de salida que teclas. Por ejemplo, el teclado matriz que usamos tiene 16 teclas (0-9, A-D, *, #), y solo 8 pines de salida. Con un teclado lineal, se necesitarían 17 pines de salida (uno por tecla más la tierra) para que funcionara. Esto es bueno porque al tener menos pines debemos hacer menos conexiones. Por tanto, el teclado de matriz es más eficiente que el lineal ya que tiene menos cableado.

Componentes requeridos:

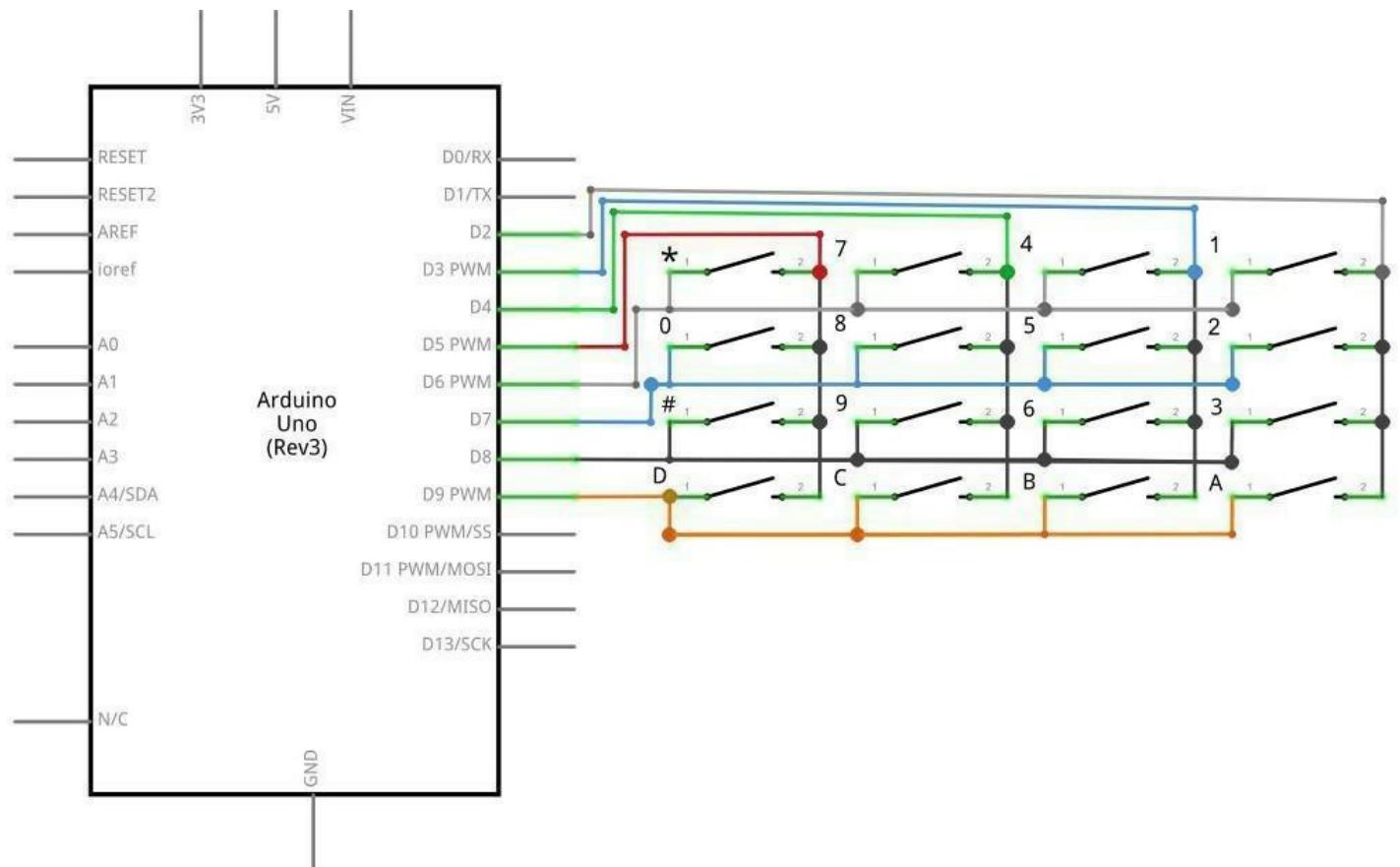
1 x Elegoo Uno R3

1 x Módulo de interruptor de membrana

8 x Cables M-M (Jumpers Macho - Macho)



Esquema de Conexión



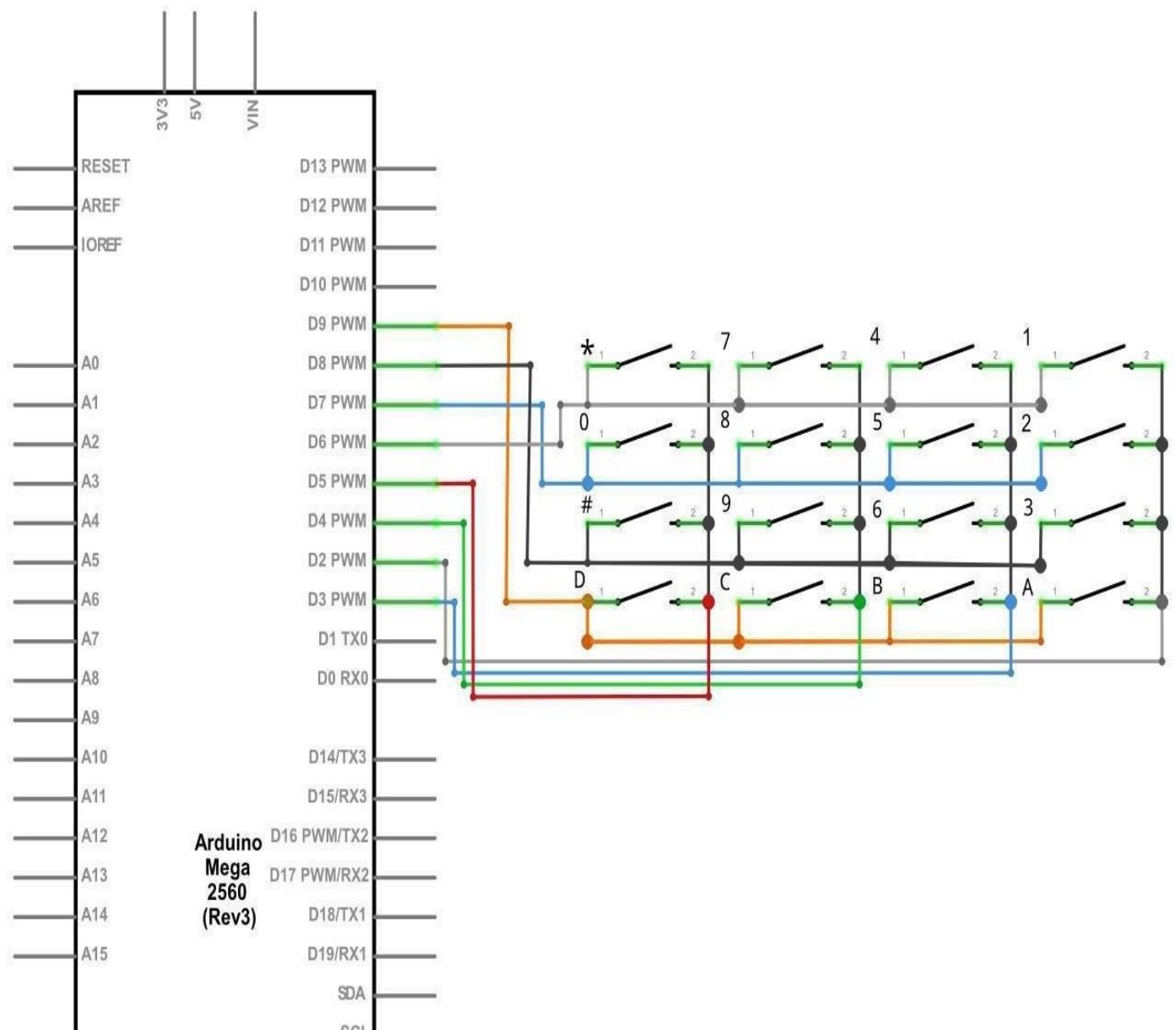
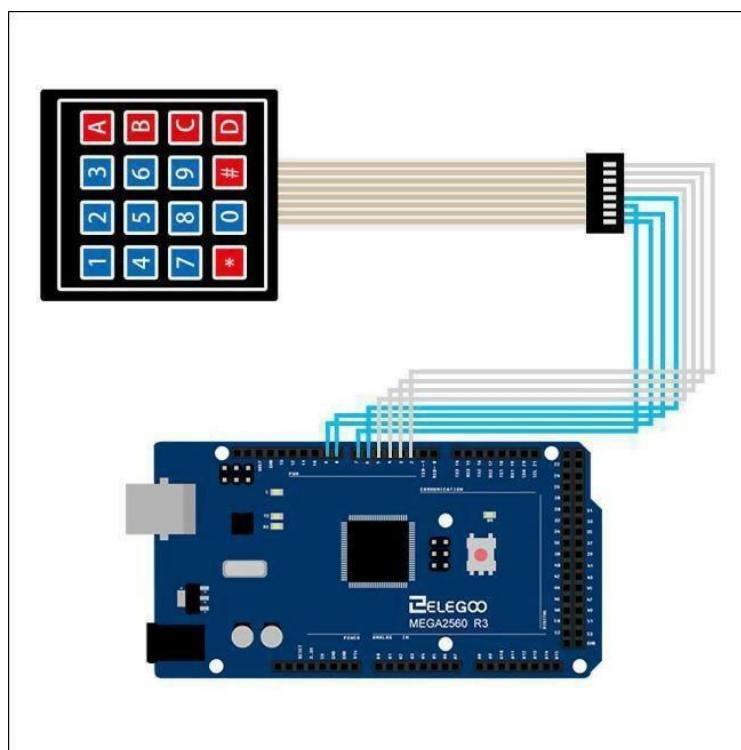
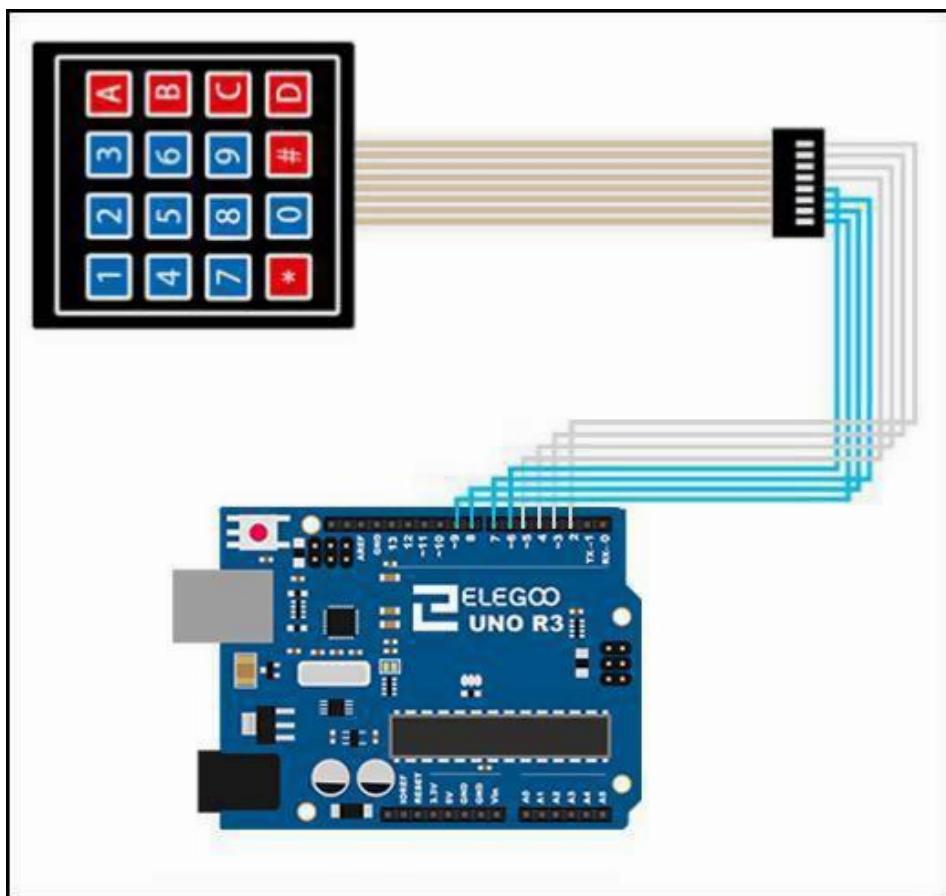


Diagrama de cableado



Conecta los pines a la entradas digitales del UNO R3

D9-D2. Conectamos el primer pin a D9, el segundo a D8, el tercero a D7, el cuarto a D6, el quinto a D5, el sexto a D4, el séptimo a D3, y el octavo a D2.

A continuación se muestran las conexiones en una tabla:

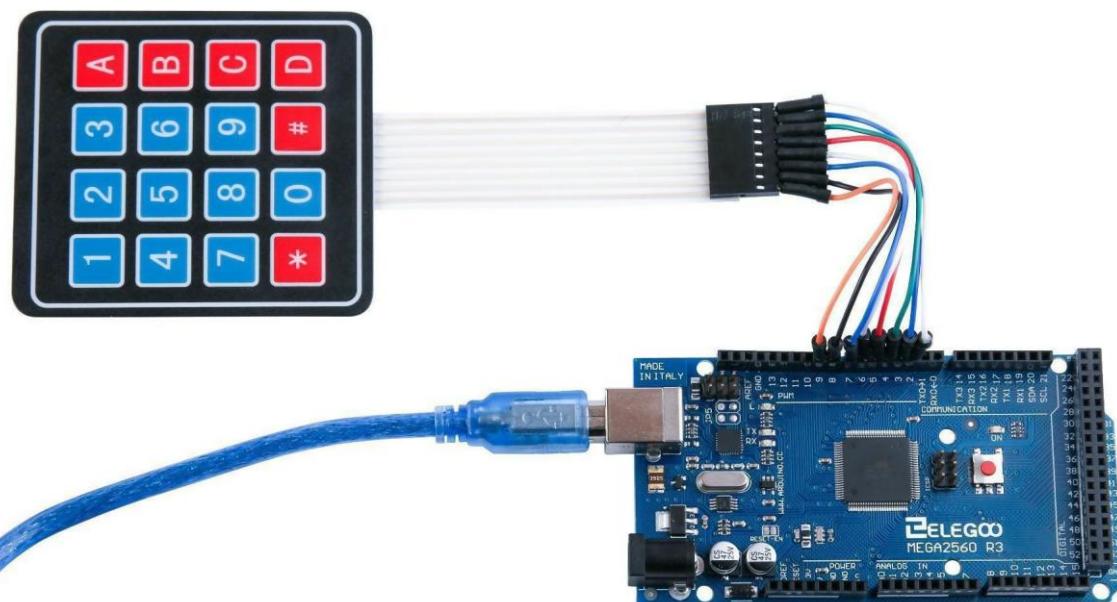
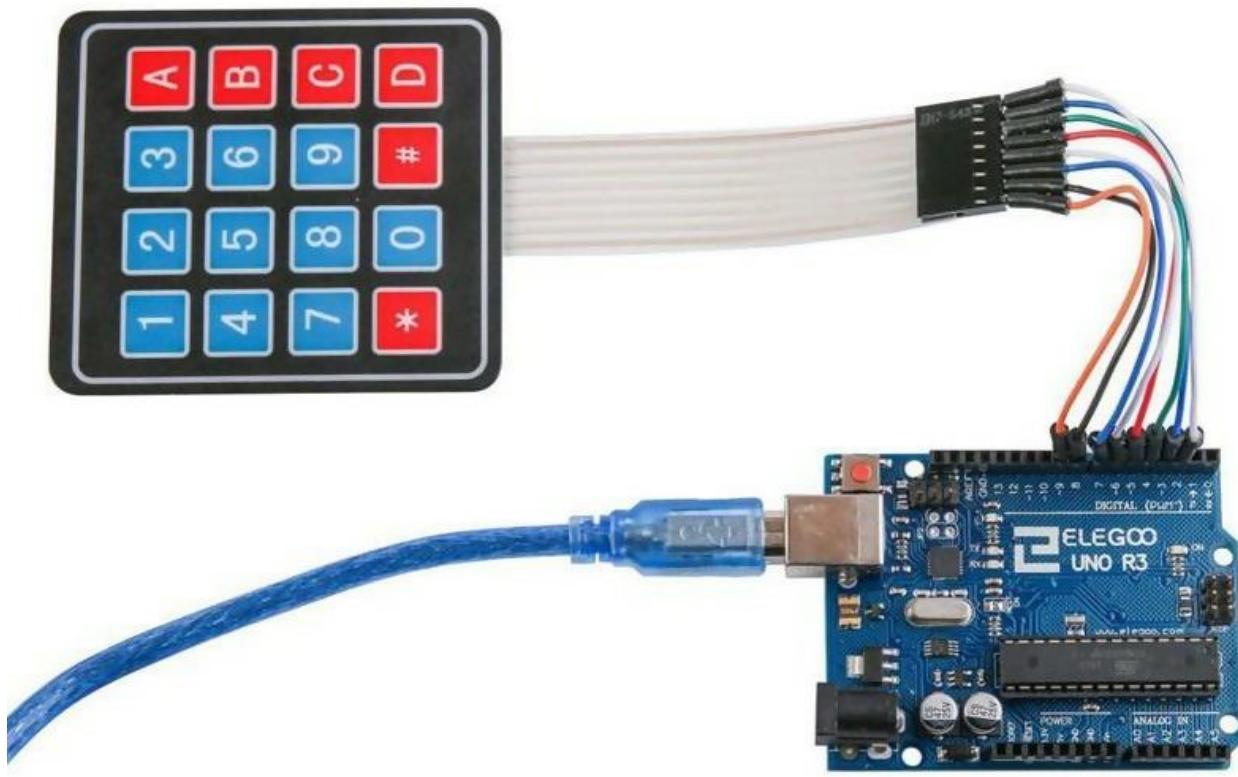
| Keypad Pin | Connects to Arduino Pin... |
|------------|----------------------------|
| 1 | D9 |
| 2 | D8 |
| 3 | D7 |
| 4 | D6 |
| 5 | D5 |
| 6 | D4 |
| 7 | D3 |
| 8 | D2 |

Código

Después de realizar el cableado, por favor abre el programa en la carpeta de código Lección 34 Módulo de Teclado y haz clic en UPLOAD para cargar el programa. Si hay algún error, consulta los detalles en la Lección 2 para cargar el programa. Antes de poder correrlo, asegúrate de tener la librería < Keypad> instalada, o reinstálala de ser necesario. De otra forma, tu código no funcionará.

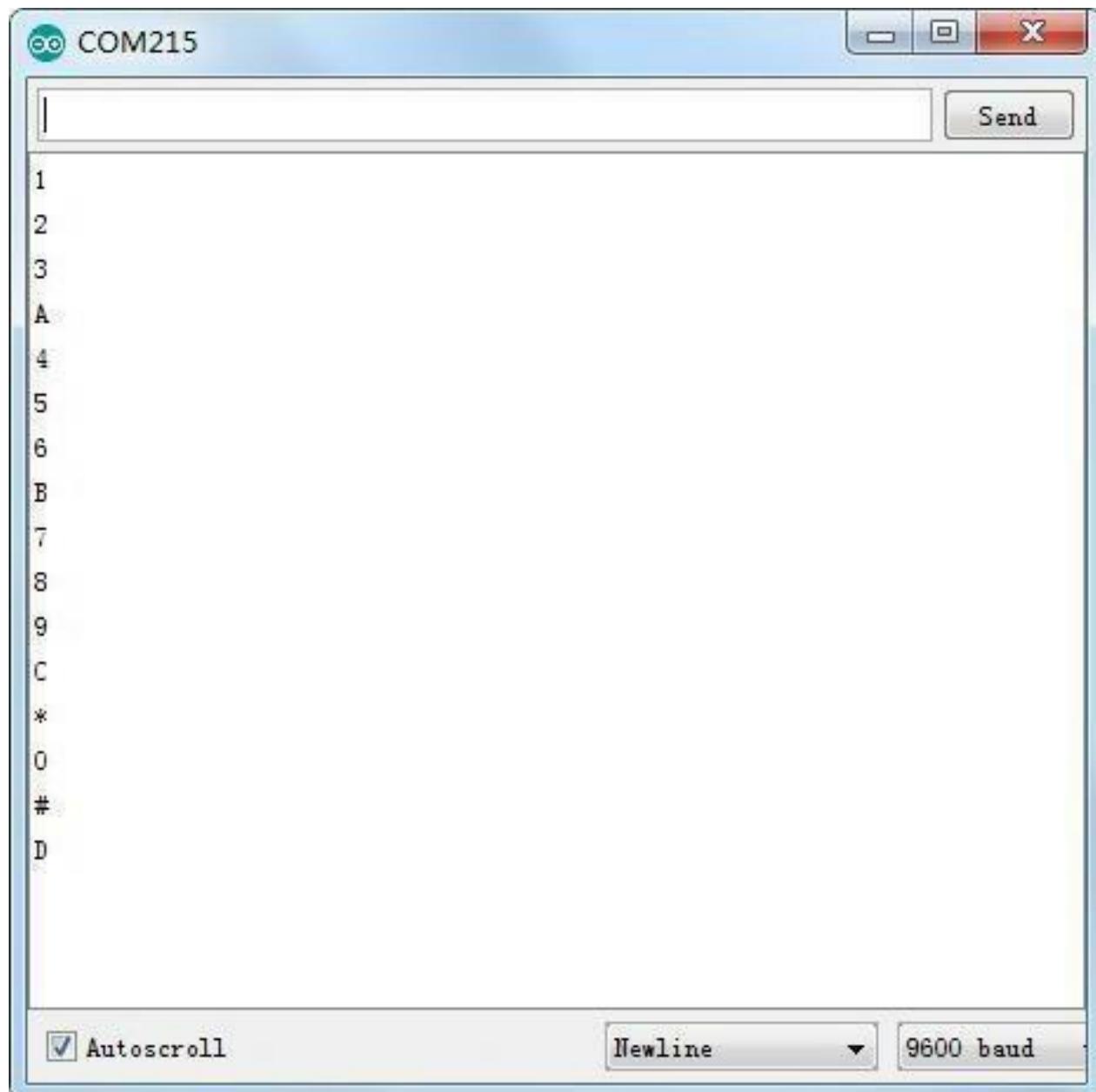
Para detalles del tutorial acerca de cómo cargar los archivos de librerías, consulta la lección 1.

Imagen ejemplo



Con este código, una vez que presiones una tecla en el teclado, se mostrar{a en el monitor serial de Arduino, después que el código sea compilado y cargado en la tarjeta UNO R3.

Haz clic en el botón del Monitor Serial para encenderlo. En la lección 2 se explica en detalle cómo utilizarlo.



A continuación se muestra el código usado en este experimento y su correspondiente explicación:

```
#include <Keypad.h>
//four rows
const byte ROWS = 4;
//four columns const byte COLS = 4;
//Define los símbolos de los botones del teclado char hexaKeys[ROWS][COLS] = {
{'1','2','3','A'},
{'4','5','6','B'},
{'7','8','9','C'},
{'*','0','#','D'}
```

};

```
//Conecta de acuerdo con el pinout del teclado byte rowPins[ROWS] = {9, 8, 7, 6};
//Conecta de acuerdo con el pinout del teclado byte colPins[COLS] = {5, 4, 3, 2};
//Inicialización de una instancia de clase NewKeypad
Keypad customKeypad = Keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS); void
setup(){
Serial.begin(9600);
```

}

```
void loop(){
char customKey = customKeypad.getKey();
```

if (customKey){ Serial.println(customKey);

Conclusión

Como creadores y editores de este tutorial, hemos intentado de la mejor forma posible enseñarte a resolver problemas para que poco a poco puedas manejar diferentes temas por tu propia cuenta. Si quieres modificar el kit o cualquiera de los proyectos, no tengas dudas y libera tu imaginación e innovación.

Si tienes dudas o consultas al momento de construir tus proyectos, puedes contactarnos pero por favor danos un poco de tiempo para responder tus correos ;) Mientras esperas por nuestra respuesta, puedes investigar en la web; existen muchas comunidades donde podrías encontrar soluciones. Será una experiencia enriquecedora para ti si logras resolver problemas por ti mismo y constituye del mismo modo una forma genial para aprender. Si igualmente deseas hablar con servicio al cliente, envíanos un mensaje a service@elegoo.com o euservice@elegoo.com si eres de Europa o Japón. ;) Gracias por tu apoyo y esperamos que te diviertas mucho con el kit. ;)