

UNIT ELECTRONICS

UNIT STARTER KIT

BY UNIT ELECTRONICS

GUÍA DE APRENDIZAJE

Aprende a programar y crear diferentes proyectos

UNIT
ELECTRONICS



ÍNDICE

▶ INTRODUCCIÓN	1
▶ Lección 0	2
▶ ¿QUÉ ES ARDUINO IDE?	3
▶ ¿QUÉ ES UNO R3?	3
▶ DESCARGA E INSTALACIÓN DE ARDUINO IDE	7
▶ INSTALACIÓN DEL CONTROLADOR CH340	8
▶ ¿CÓMO CARGAR TU PRIMER CÓDIGO A LA TARJETA UNO R3 CON ARDUINO IDE?	9
▶ ¿CÓMO INSTALAR UNA BIBLIOTECA?	11
▶ Lección 1 LED Parpadeante	12
▶ Lección 2 Control de un LED Mediante un Pulsador	16
▶ Lección 3 Control de LED RGB	19
▶ Lección 4 Zumbador Activo	23
▶ Lección 5 Zumbador Pasivo	25
▶ Lección 6 Ocho Led Controlados con el 74HC595	27
▶ Lección 7 Fotorresistor	30
▶ Lección 8 74HC595 y Display de 7 segmentos	34
▶ Lección 9 74HC595 y Display 4 Dígitos 7 segmentos	37
▶ Lección 10 Sensor de Inclinación	40
▶ Lección 11 Módulo de Sensor Ultrasónico HC-SR04	42
▶ Lección 12 Módulo de Joystick Analógico	45

ÍNDICE

- ▶ Lección 13 Sensor de Sonido y Relevador 5V..... 49
- ▶ Lección 14 Sensor de Lluvia Pantalla LCD I2C..... 53
- ▶ Lección 15 Sensor de Temperatura LM35 Pantalla LCD I2C..... 57
- ▶ Lección 16 DHT11 Sensor de Temperatura y Humedad..... 60
- ▶ Lección 17 Módulo de Interruptor de Membrana y Led RGB..... 63
- ▶ Lección 18 Módulo de Tiempo Real DS1302 y Led..... 67
- ▶ Lección 19 Led Receptor IR con Mando a Distancia y Control de Motor a Pasos..... 70
- ▶ Lección 20 RFID- RC522 y Servo Motor SG90..... 75

LECCIÓN 0

¿Quiénes somos?

UNIT ELECTRONICS es un distribuidor de componentes electrónicos en línea. Ofrecemos una amplia variedad de módulos, tarjetas de desarrollo y semiconductores con presencia en México. Nuestro objetivo es proveer componentes electrónicos y productos de ingeniería a estudiantes, escuelas, empresas e instituciones de forma ágil y sencilla. Así como tener contenido de apoyo, como tutoriales, especificaciones e información adicional que facilite la creación de proyectos, prototipos y prácticas de electrónica, robótica, mecatrónica y más.

¿Qué es UNIT Starter Kit?

Es un kit de electrónica enfocado al aprendizaje y creación de proyectos con Arduino. El kit está basado en la tarjeta de desarrollo UNO R3 y en el lector de tarjetas RFID, incluye otros módulos, sensores y componentes electrónicos que te ayudarán a dar tus primeros pasos en el mundo de Arduino, así como en la tecnología RFID.

¿Para qué sirve UNIT Starter Kit?

Este Kit es ideal para iniciar en el mundo de la programación con Arduino, así como impulsar la creatividad y destreza para crear diferentes prototipos y proyectos de electrónica e ingeniería. Con este Kit podrás aprender conceptos básicos de electrónica y programación en Arduino, también te ayudará a realizar prototipos de control de acceso por RFID, control de motores a paso, despliegue de información mediante LCD o matriz de leds, monitoreo de temperatura, encendido y apagado de leds, control por IR entre otros proyectos interesantes que te pondrán a prueba.

¿Qué incluye el UNIT Starter Kit?

Este Kit incluye tutoriales y prácticas guiadas en formato digital, para iniciar y tener conocimiento de cómo utilizar, conectar y programar los diversos módulos, sensores, componentes y actuadores que incluye el kit.

El material que incluye es el siguiente: 

LECCIÓN 0

Lista de materiales:

Tarjeta de desarrollo: UNO R3 con Cable USB

Sensores	Componentes	Actuadores y Conexión	Optoelectrónica
Lector RFID RC522 con llavero y tarjeta	Buzzer Zumbador 5V Activo	Motor a pasos 28BYJ-48 con controlador	Leds Amarillo, Rojo y Verde de 5mm
Sensor de temperatura y humedad DHT11	Buzzer Zumbador 5V Pasivo	Servomotor SG90 RC 9g	Display 7 segmentos cátodo común
Tiempo Real DS1302	4 Push Button Grande 12mm x 12mm	Teclado Matricial 4x4	Display 4 Dígitos 7 Segmentos cátodo común
Detención de líquido	Potenciómetro de 10k 3 pines	Display 4 Dígitos 7 Segmentos cátodo común	Alimentación
Sensor JoyStick KY-023	Fotorresistencia LDR 5537 x3	Control remoto IR	CR2032 Pila de Litio Tipo Botón
SW-520D Sensor de inclinación x2	Receptor Infrarrojo VS1838B	Protoboard 830 Pts MB-102	Pila de 9V Cuadrada
Sensor Led RGB	Led IR fototransistor 5mm iNo incluye fotodiodo!	10 cables Dupont Largos Macho - Hembra	8 Pines Header Macho
Sensor de Sonido	SN74HC595N Registro de Desplazamiento	65 cables para Protoboard	Fuente para Protoboard
Sensor de Temperatura LM35	Resistencia de precisión 1K 1/2W x10	Conector de Batería 9V para UNO R3	
Relevador de 5V SRD-05VDC-SL-C	Resistencia de precisión 10K 1/2W x10		
Sensor Ultrasónico HC-SR04	Resistencia de precisión 220 Ohms 1/2W x10		

LECCIÓN 0

LECCIÓN 0

Para comenzar a utilizar el UNIT Starter Kit es necesario tener instalado el software de programación Arduino IDE, así como tener conocimientos y manejo de la tarjeta de desarrollo UNO R3, ya que este software y placa son la parte fundamental del kit.

En esta lección se explica qué es Arduino IDE, instalación, abrir ejemplo de pruebas, cargar código Blink, importación de librerías, así como mencionar que es la tarjeta de desarrollo UNO R3, especificaciones, pines y cómo emplear con Arduino IDE.

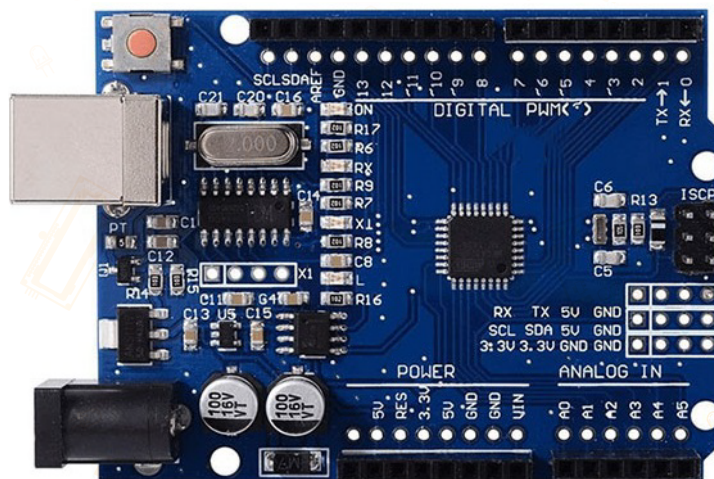
¿QUÉ ES ARDUINO IDE?

Es un entorno de programación diseñado para escribir, compilar, cargar y probar códigos en cualquier placa de Arduino compatible y de terceros. Es de código abierto y su interfaz es muy amigable para comenzar a programar microcontroladores, así como hacer funcionar a gran variedad de sensores, actuadores y prototipos de electrónica.

Este software de programación permite que el UNIT Starter Kit tenga un entorno dónde se pueda escribir código y probarlo en la tarjeta UNO R3 que es compatible con el software y con cada uno de los componentes, módulos y sensores que incorpora este kit.

¿QUÉ ES UNO R3?

UNO R3 es una tarjeta de desarrollo programable y compatible con el entorno de programación Arduino IDE. Es fácil de utilizar, también te ayudará a cargar y ejecutar todos los códigos que elabores en Arduino IDE. Esta tarjeta tiene varios periféricos de entrada y salida que facilitan la conexión y comunicación con diferentes sensores o actuadores, también incorpora pines reservados, así como pines y puertos de alimentación.



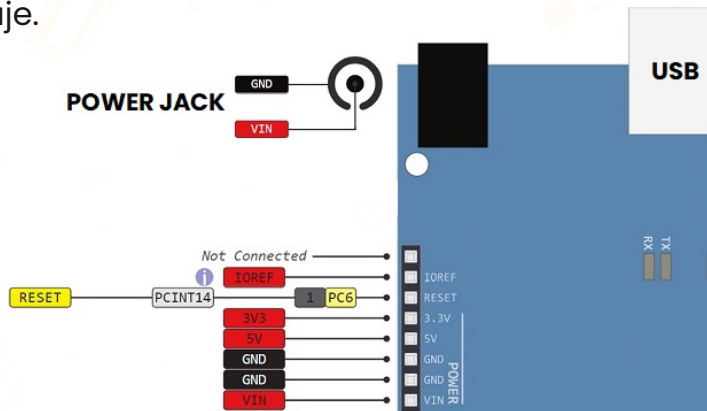
A continuación, se especifican los puertos, pines e interfaces de la tarjeta UNO R3.

LECCIÓN 0

PUERTOS DE ALIMENTACIÓN

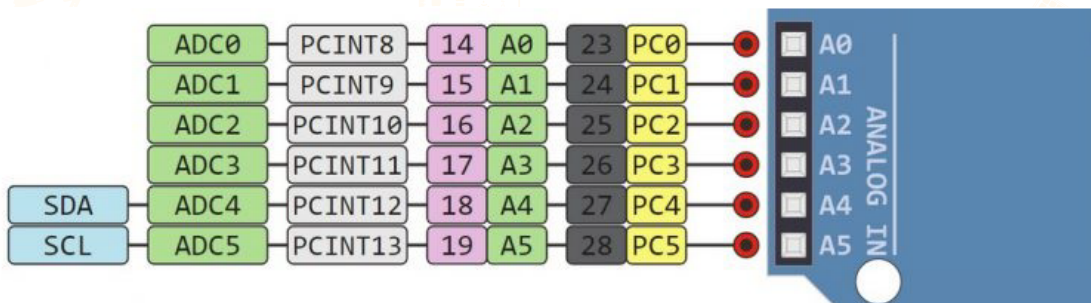
Hay 3 formas de alimentar el UNO R3:

- **Power Jack:** Con este puerto puede ser usado para alimentar la tarjeta UNO R3, con un adaptador DC de pared. La tarjeta puede ser alimentada por 5 a 12V. Por encima de 12 voltios, los reguladores podrían sobrecalentarse, y por debajo de 7 voltios, podrían no ser suficiente.
- **Pin VIN:** Este pin se utiliza para alimentar la placa UNO R3 utilizando una fuente de alimentación externa. El voltaje debe estar dentro del rango mencionado anteriormente.
- **Puerto USB:** Se utiliza para alimentar y programar la tarjeta UNO R3.
- **3V, 5V y GND:** Son salidas que sirven para alimentar sensores y otros dispositivos. Evitar conectar voltaje de alimentación a estos pines, ya que, son pines de salida que suministran voltaje.



PINES ANALÓGICOS

La tarjeta UNO R3 tiene 6 pines analógicos, que utilizan ADC (Convertidor de Analógico a Digital). Estos pines sirven como entradas analógicas, pero también pueden funcionar como entradas o salidas digitales. Los pines A4 y A5 sirven para realizar una comunicación I2C (SDA y SCL respectivamente) con diferentes dispositivos que tengan esta misma interfaz.



Los pines Arduino A0-A5 son capaces de leer tensiones analógicas. En UNO R3 el ADC tiene una resolución de 10 bits, lo que significa que puede representar una tensión analógica de 1,024 niveles digitales. El ADC convierte el voltaje en bits que el MCU puede entender.

PINES DIGITALES

¿QUÉ SIGNIFICA DIGITAL?

Digital es una forma de representar la tensión en 1 bit: 0 o 1. Los pines digitales del UNO R3 son pines diseñados para ser configurados como entradas o salidas según las necesidades del usuario. Los pines digitales están activados o desactivados. Cuando están en ON se encuentran en un estado de ALTO tensión de 5V y cuando están en OFF se encuentran en un estado de BAJO tensión de 0V. En el UNO R3, cuando los pines digitales están configurados como salida, se ajustan de 0 o 5 voltios.

Cuando los pines digitales se configuran como entrada, la tensión se suministra desde un dispositivo externo. Este voltaje puede variar entre 0-5 voltios, que se convierte en representación digital (0 o 1). Para determinar esto, hay dos umbrales:

- Por debajo de 0.8v – considerado como 0.
- Por encima de 2v – considerado como 1.

¿QUÉ ES UNA SEÑAL PWM?

En general, la modulación de ancho de pulso, PWM, es una técnica de modulación utilizada para codificar un mensaje en una señal pulsante. Un PWM se compone de dos componentes clave: frecuencia y ciclo de trabajo. La frecuencia PWM dicta el tiempo que se tarda en completar un solo ciclo (período) y la rapidez con la que la señal fluctúa de alta a baja. El ciclo de trabajo determina cuánto tiempo una señal permanece alta fuera del período total. El ciclo de trabajo se representa en porcentaje.

En la tarjeta UNO R3, los pines habilitados para PWM producen una frecuencia constante de ~500Hz, mientras que el ciclo de trabajo cambia de acuerdo con los parámetros establecidos por el usuario.

PROTOCOLOS DE COMUNICACIÓN

Serial (TTL) – Los pines digitales 0 y 1 son los pines seriales del UNO R3.

¿QUÉ ES LA COMUNICACIÓN SERIE?

La comunicación serie o secuencial, se utiliza para intercambiar datos entre la placa UNO R3 y otro dispositivo en serie como ordenadores, pantallas, sensores y más. Cada placa UNO R3 tiene al menos un puerto serie. La comunicación serie se produce en los pines digitales 0 (RX) y 1 (TX), así como a través de USB. UNO R3 también soporta la comunicación serie a través de pines digitales con la librería Software Serial Library. Esto permite al usuario conectar varios dispositivos habilitados para serie y dejar el puerto serie principal disponible para el USB.

LECCIÓN 0

¿QUÉ ES SPI?

La Interfaz Periférica Serial (SPI) es un protocolo de datos en serie utilizado por los microcontroladores para comunicarse con uno o más dispositivos externos en una conexión tipo bus. El SPI también se puede utilizar para conectar 2 microcontroladores. En el bus SPI, siempre hay un dispositivo que se denomina Maestro, Master, y todos los demás Esclavos, Slaves. En la mayoría de los casos, el microcontrolador es el dispositivo maestro. El pin SS (Slave Select) determina con qué dispositivo se está comunicando actualmente el Maestro.

Los dispositivos habilitados para SPI siempre tienen los siguientes pines:

- MISO (Master In Slave Out) –Una línea para enviar datos al dispositivo Maestro.
- MOSI (Master Out Slave In) –La línea Master para el envío de datos a dispositivos periféricos
- SCK (Serial Clock) –Una señal de reloj generada por el dispositivo Master para sincronizar la transmisión de datos.

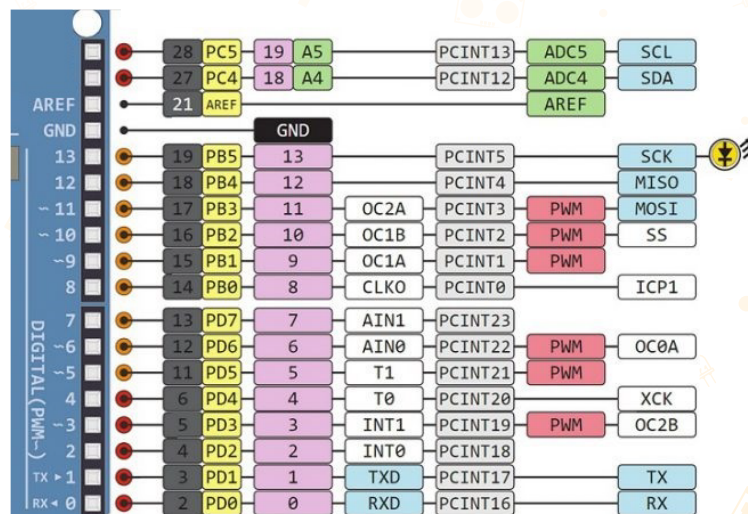
¿QUÉ ES I2C?

I2C (Inter Integrated Circuit) es un protocolo de comunicación comúnmente conocido como «bus I2C». El protocolo I2C fue diseñado para permitir la comunicación entre componentes en una sola tarjeta de circuito. Con I2C hay 2 cables denominados SCL y SDA.

- SCL es la línea de reloj que está diseñada para sincronizar las transferencias de datos.
- SDA es la línea utilizada para transmitir datos.

Los pines I2C – SCL/SDA son los pines dedicados para la comunicación I2C. En la tarjeta de desarrollo UNO R3 se encuentran en los pines analógicos A4 y A5.

En la siguiente imagen se especifican los pines y las interfaces de comunicación de la tarjeta UNO R3.



LECCIÓN 0

DESCARGA E INSTALACIÓN DE ARDUINO IDE

Para descargar el entorno de programación en su computadora abra el siguiente enlace:

<https://www.arduino.cc/en/software>

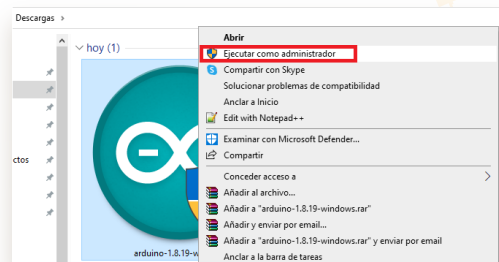
Si ya tienes instalado Arduino IDE puedes omitir estos pasos, de lo contrario descargarlo de acuerdo con tu sistema operativo, por ejemplo, si tienes Windows selecciona la siguiente opción:



Te saldrá la siguiente ventana, dar clic en ¡JUST DOWNLOAD!:



Se descarga un .EXE que tendrás que ejecutar como administrador y seguir todos los pasos de instalación, así como aceptar todos los permisos que requiera la instalación para que el entorno de programación se instale correctamente.



LECCIÓN 0

INSTALACIÓN DEL CONTROLADOR CH340

Para que tu computadora reconozca la tarjeta UNO R3 será necesario instalar el controlador CH340 ya que la tarjeta utiliza este chip que permite comunicarse con el MCU y programarlo. Abre el siguiente enlace y descarga el controlador de acuerdo con tu sistema operativo de tu PC.

http://www.wch.cn/download/CH341SER_EXE.html

Por ejemplo, si tu PC tiene Windows descarga la siguiente opción:

适用范围	版本	上传时间	资料大小
CH340G, CH340T, CH340C, CH340N, CH340K, CH340E, CH340B, CH341A, CH341F, CH341T, CH341B, CH341C, CH341U	3.6	2021-12-23	459KB

CH340/CH341 USB转串口Windows驱动程序, 支持32/64位 Windows 11/10/8.1/8/7/VISTA/XP, SERVER 2022/2019/2016/2012/2008/2003, 2000/ME/98, ARM64 WIN11,通过微软数字签名认证, 支持USB转3线和9线串口等, 用于随产品发行到最终用户。

下载

Se descargará un archivo .exe, ejecútalo como administrador y por último da clic en INSTALL, saldrá una ventana que tendrás que dar clic en Aceptar y cierra la ventana de instalación.

DriverSetup(X64)

Device Driver Install / UnInstall

Select INF: CH341SER.INF

INSTALL

UNINSTALL

HELP

WCH.CN
|_ USB-SERIAL CH340
|_ 12/01/2021, 3.6.2021.12

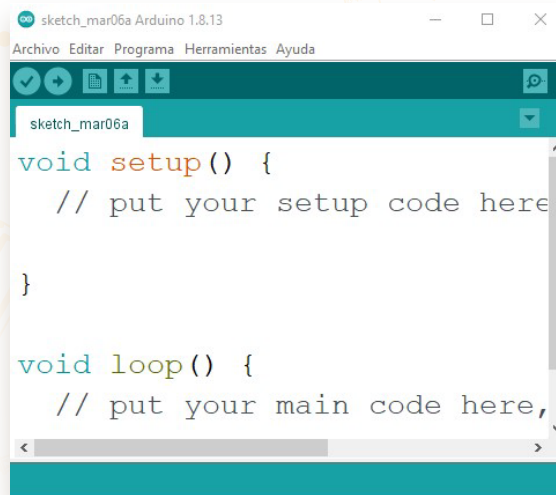
Y listo ya tendrás instalado el controlador para comenzar a utilizar la tarjeta UNO R3 con Arduino IDE.

LECCIÓN 0

¿CÓMO CARGAR TU PRIMER CÓDIGO A LA TARJETA UNO R3 CON ARDUINO IDE?

Ya instalado el controlador y el Arduino IDE cargaremos tu primer código, utilizaremos el ejemplo de blink que incluye Arduino IDE en su galería de códigos de ejemplo.

Para comenzar abre el Arduino IDE. Ya abierto saldrá la siguiente ventana



```

sketch_mar06a Arduino 1.8.13
Archivo Editar Programa Herramientas Ayuda

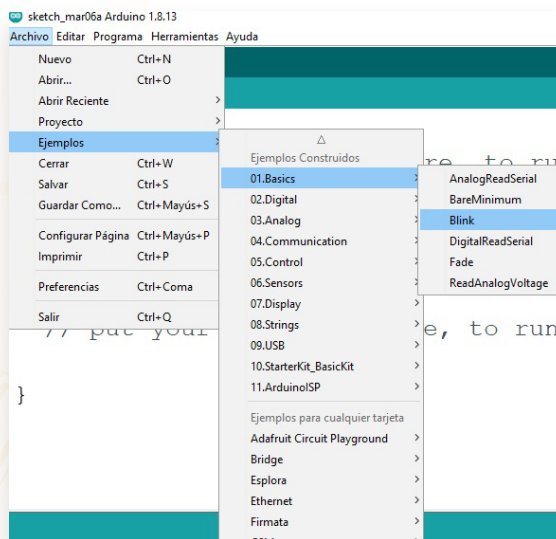
sketch_mar06a

void setup() {
  // put your setup code here
}

void loop() {
  // put your main code here,

```

Después abre el ejemplo de blink sigue las siguientes ventanas:



```

sketch_mar06a Arduino 1.8.13
Archivo Editar Programa Herramientas Ayuda

Nuevo Ctrl+N
Abrir... Ctrl+O
Abrir Reciente >
Proyecto >
Ejemplos
Cerrar Ctrl+W
Salvar Ctrl+S
Guardar Como... Ctrl+Mayús+S
Configurar Página Ctrl+Mayús+P
Imprimir Ctrl+P
Preferencias Ctrl+Coma
Salir Ctrl+Q

Ejemplos Construidos
01.Basics
02.Digital
03.Analog
04.Communication
05.Control
06.Sensors
07.Display
08.Strings
09.USB
10.StarterKit_BasicKit
11.ArduinoISP

Ejemplos para cualquier tarjeta
Adafruit_Circuit_Playground
Bridge
Esplora
Ethernet
Firmata
CCM

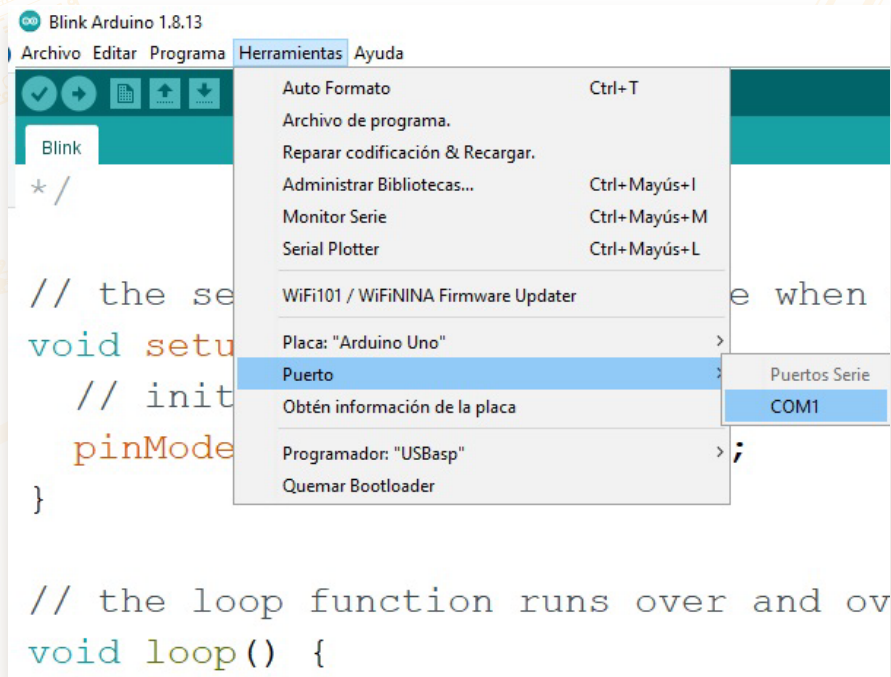
AnalogReadSerial
BareMinimum
Blink
DigitalReadSerial
Fade
ReadAnalogVoltage

```

Se abrirá una ventana nueva con el código. Después conecta la tarjeta UNO R3 a tu PC con la ayuda del cable USB. Dirígete a la pestaña de herramientas y asegúrate que tengas seleccionada la tarjeta "Arduino Uno" y tener seleccionado el puerto COM que le asigno tu PC a la tarjeta UNO R3.

LECCIÓN 0

La siguiente imagen te servirá de guía:



Por último, carga el código a la tarjeta UNO R3, para hacer esto da clic en el siguiente símbolo y comenzará a cargar el código a la tarjeta.



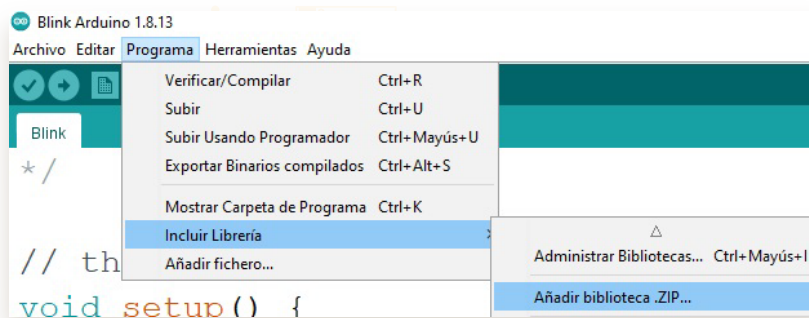
Al término de la carga la placa comenzará a encender y apagar el led que se incorpora en el pin 13 de la tarjeta UNO R3.

LECCIÓN 0

¿CÓMO INSTALAR UNA BIBLIOTECA?

Para realizar las lecciones de esta guía se necesitará la utilización de librerías para que funcionen correctamente algunos sensores que incorpora el Kit, así que será de suma importancia conocer cómo importar librerías en Arduino IDE, sigue los siguientes pasos:

- 1 Tener la librería descargada en formato .ZIP
- 2 Después dar clic en la pestaña de Programa – Incluir librería y seleccionar Añadir librería .ZIP



- 3 Abrir la librería descargada y esperar a que se importe. Cuando se importe en la parte inferior mostrará un mensaje que la librería importó correctamente.



Lección 1

LED Parpadeante

Introducción

En esta lección, aprenderemos a utilizar la tarjeta UNO R3 mediante la activación de un LED que parpadea una vez por segundo.

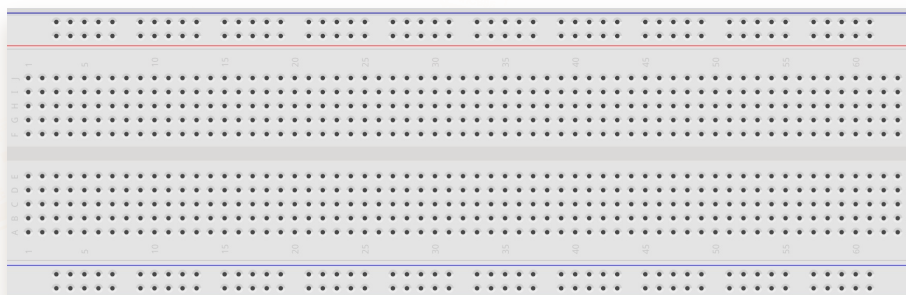
Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Led de 5mm rojo x 1
- 4 Resistencia de 220 Ohms x 1
- 5 Cables Macho – Macho x 2

Conocimientos previos

PROTOBOARD (PLACA DE PRUEBAS)

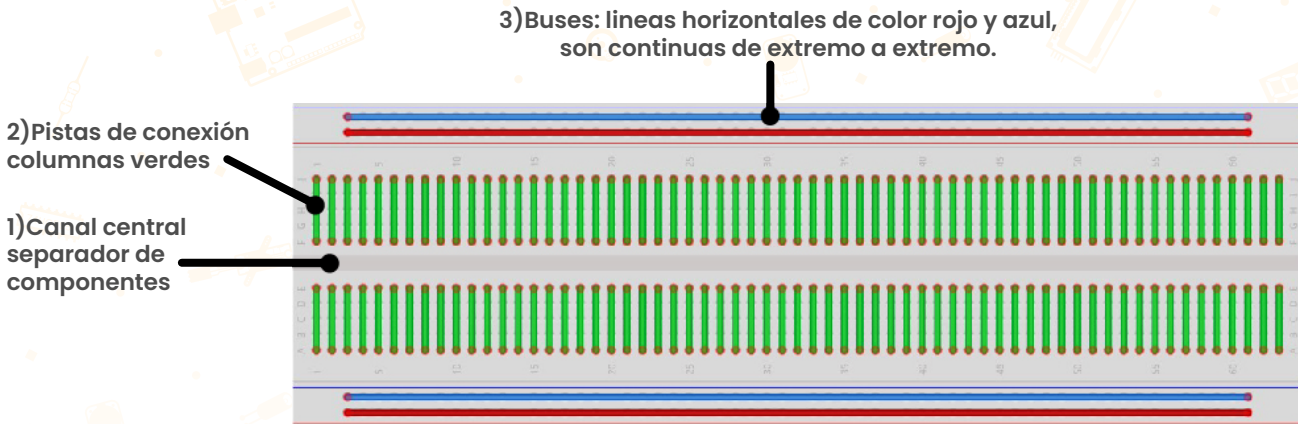
Una placa de pruebas o también llamada protoboard es un pequeño tablero lleno de agujeros unidos electrónicamente y sirve para interconectar diferentes componentes, así como realizar diferentes circuitos de forma práctica, rápida y sencilla.



Un protoboard se divide en 3 regiones:

- 1 **Canal central:** Es la región localizada en medio del Protoboard, se utiliza para colocar de manera adecuada circuitos integrados.
- 2 **Buses:** Los buses se localizan en ambos extremos del Protoboard, se representan por las líneas rojas (buses positivos o de voltaje) y azules (buses negativos o de tierra) y conducen de acuerdo con estas, no existe conexión física entre ellas. La fuente de alimentación generalmente se conecta aquí.
- 3 **Pistas de conexión:** Las pistas se localizan en la parte central del protoboard aquí puedes conectar tu circuito considerando que son columnas verticales de manera lineal.

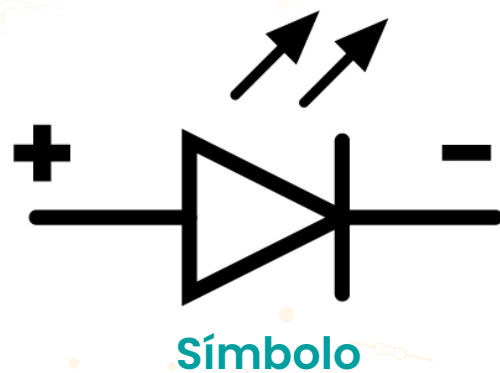
Lección 1 LED Parpadeante



LED

LED es un diodo emisor de luz, es un dispositivo óptico, convierte energía eléctrica en energía luminosa, además tiene 2 terminales. La terminal más larga es positiva (+) y la terminal más corta es negativa (-). Otra forma de diferenciar la terminal negativa, el led tiene un lado plano para indicar que esa es la terminal negativa.

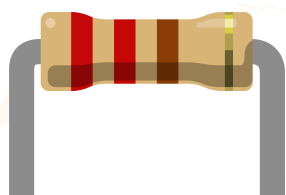
Para utilizar el led debe polarizarse de manera correcta, no invertir su polaridad y su voltaje de funcionamiento ya que se dañará.



Lección 1 LED Parpadeante

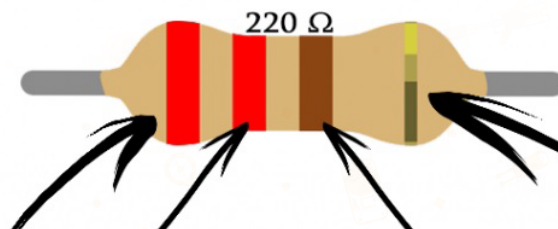
RESISTENCIA

Una resistencia es un pequeño componente electrónico pasivo que sirve para limitar la corriente en un circuito eléctrico o ayudar al correcto funcionamiento de otros componentes electrónicos. La unidad de medida es Ohmio que se representa con la letra griega omega (Ω).



Símbolo

A diferencia de los LEDs, las resistencias no tienen polaridad. Pueden ser conectadas de cualquier manera. Las resistencias incorporan bandas de colores que indican cuál es su valor. Para saber el valor de una resistencia se utiliza el código de colores, que facilita la identificación y el valor de las resistencias de acuerdo con el color que incorporan en sus bandas. En la siguiente imagen se muestra el código de colores:



Color	Banda 1	Banda 2	Banda 3 (multiplicadora)	Tolerancia
Negro	0	0	0 x1	
Café	1	1	1 x10	1%
Rojo	2	2	2 x100	2%
Naranja	3	3	3 x1000	
Amarillo	4	4	4 x10000	
Verde	5	5	5 x100000	0.5%
Azul	6	6	6 x1000000	0.25%
Morado	7	7	7 x10000000	0.10%
Gris	8	8	8 x100000000	0.05%
Blanco	9	9	9 x1000000000	

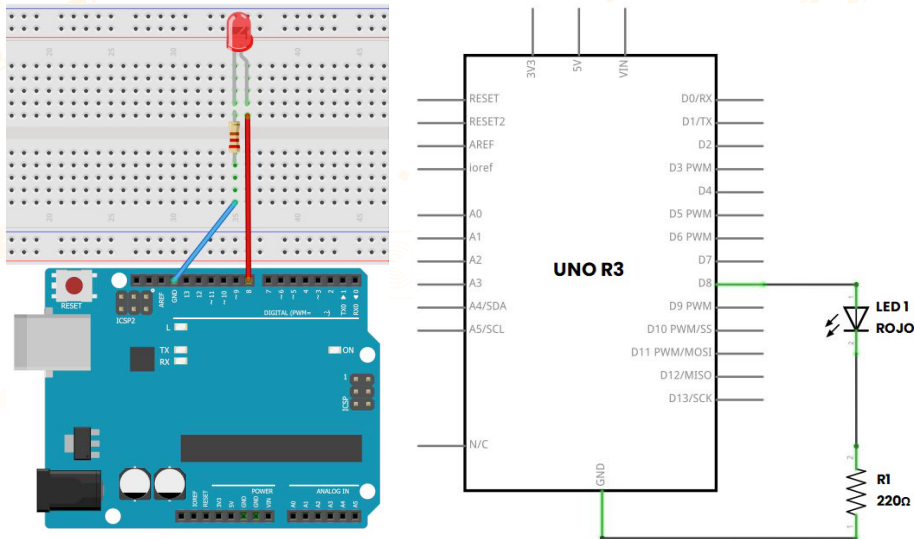
Dorado 5%

Plata 10%

Lección 1 LED Parpadeante

Diagrama de Conexión

A continuación, se muestra cuál será la conexión del led al protoboard y a la tarjeta UNO R3.



Código de Funcionamiento

Realizaremos un código para encender y apagar un led cada segundo ubicado en el pin 8 de la tarjeta UNO R3.

```
//Parpadeo de un led cada segundo

int ledPin=8;           //Control del Led por medio del pin 8

void setup()
{
  pinMode(ledPin, OUTPUT); //Se declara al "ledPin" como pin de salida
}

void loop()
{
  digitalWrite(ledPin, HIGH); // Encendido del Led (HIGH marca el nivel de voltaje)
  delay(1000);                //Permanecerá en este estado 1000 ms (1s)
  digitalWrite(ledPin, LOW); //Apagado del Led (LOW marca el nivel de voltaje bajo)
  delay(1000);                //Permanecerá en este estado 1000 ms (1s)
}
```

Lección 2

Control de un LED Mediante un Pulsador

Introducción

En esta lección, aprenderás cómo utilizar pulsadores con entradas digitales para tener control de encendido y apagado de un led.

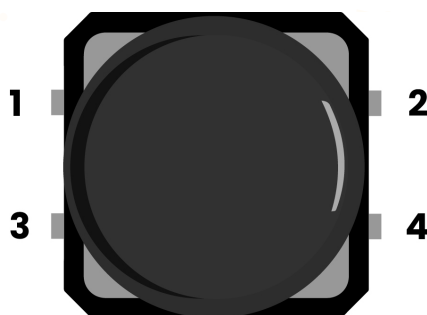
Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Led de 5mm rojo x 1
- 4 Resistencia de 220 Ohms
- 5 Push Button 4 pines Grande 12x12mm x 2

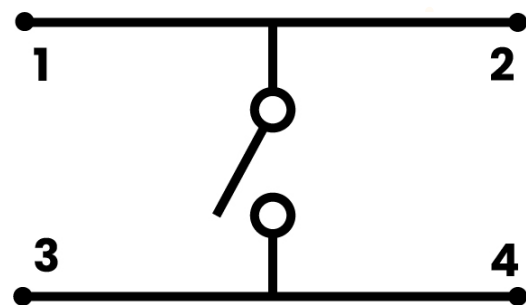
Conocimientos previos

Push Botton 4 pines o también conocido como MicroSwitch, botón o pulsador es un dispositivo táctil que sirve como interruptor ya que puede ser activado, al ser pulsado con el dedo y permiten el flujo de corriente mientras es accionado. Los pulsadores son de diversas formas y tamaños y se encuentran en todo tipo de dispositivos, aunque principalmente en aparatos eléctricos y electrónicos.

Para esta lección se utilizará un botón de 4 pines. A continuación se muestra los pines del botón de 4 pines, así como su símbolo del botón, esto ayudará a identificar correctamente los pines y como está conformado internamente el botón para después realizar las conexiones al protoboard.



Pinout



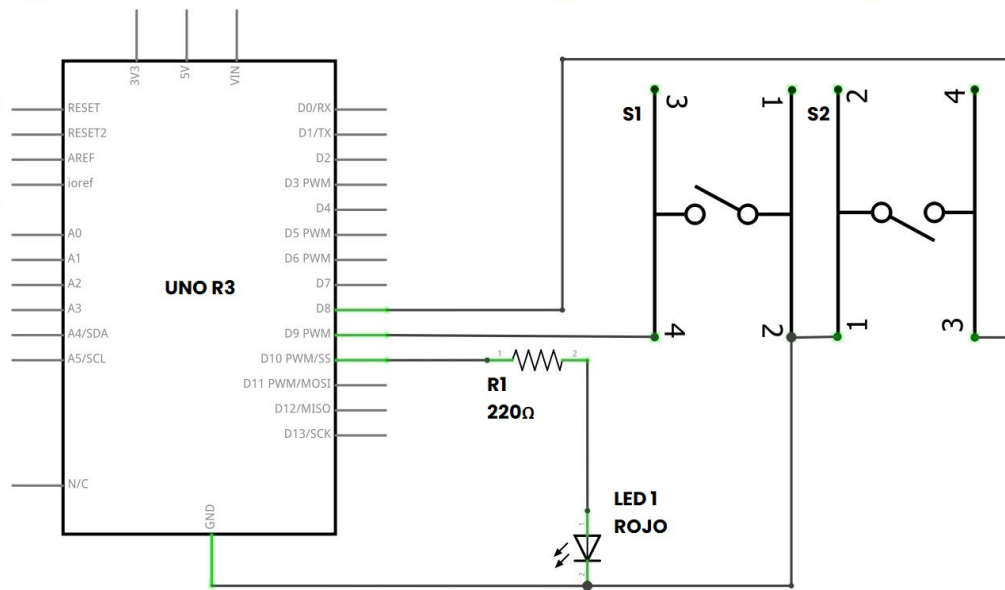
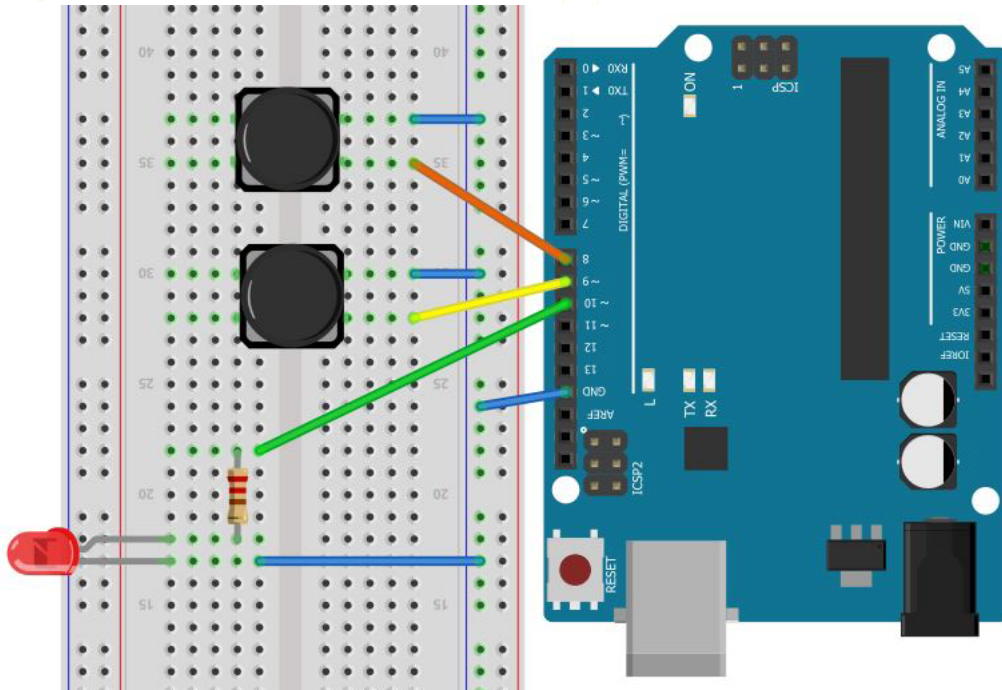
Símbolo

Para realizar el circuito en el protoboard se utilizará 2 botones, 1 para encender el led y otro para apagarlo, también se utilizará una resistencia de 220 Ohms para proteger al led.

Lección 2 Control de un LED Mediante un Pulsador

Diagrama de Conexión

A continuación, se muestra cuál será la conexión de los pulsadores y el led a la tarjeta UNO R3.



Lección 2 Control de un LED Mediante un Pulsador

Código de Funcionamiento

Realizaremos un código para encender y apagar un led mediante 2 pulsadores.

```
//Control del encendido y apagado de un led por medio de push button

int ledPin = 10; // Asignación del Pin 10 para el Led

int onApin = 9; //Asignación del Pin 9 para el push button que nos ayudará a encender el Led

int offBpin = 8; //Asignación del Pin 8 para el push button que nos ayudará a apagar el Led

byte leds = 0; //Se declara una variable leds para controlar en estado, actual del led (on/off)
                //Tipo byte (números que representan 8 bits)

void setup()
{
  pinMode(ledPin, OUTPUT); //Declaración del pin de salida a ledPin / pin 10

  //Declaración de los push button en modo PULL UP ()

  //Podamos conectar directamente al Arduino sin apoyo de una resistencia física

  pinMode(onApin, INPUT_PULLUP);

  pinMode(offBpin, INPUT_PULLUP);
}

void loop()
{
  //Realizaremos lectura de la entrada digital del pin 9 / pin 8 y realizando acciones dependiendo
  //el estado en que se encuentre

  if (digitalRead(onApin) == LOW) //Si se presiona el botón del pin 9 que se encuentra en nivel
                                  //Alto pasará a nivel Bajo y por consiguiente...

  {
    digitalWrite(ledPin, HIGH); //El led se encenderá
  }

  if (digitalRead(offBpin) == LOW) //Si se presiona el botón del pin 8

  {
    digitalWrite(ledPin, LOW); //El led se apagará
  }
}
```

Lección 3

Control de LED RGB

En esta lección aprenderemos a utilizar un led RGB para emitir diferentes colores mediante PWM y la combinación de sus 3 colores básicos: Rojo (Red), Verde (Green) y Azul (Blue). Para facilitar las conexiones se utilizará el Sensor LED RGB, también la tarjeta UNO R3.

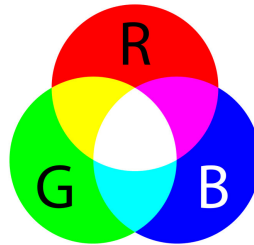
Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Sensor Led RGB x 1
- 4 Cables Hembra - Macho x 4

Conocimientos previos

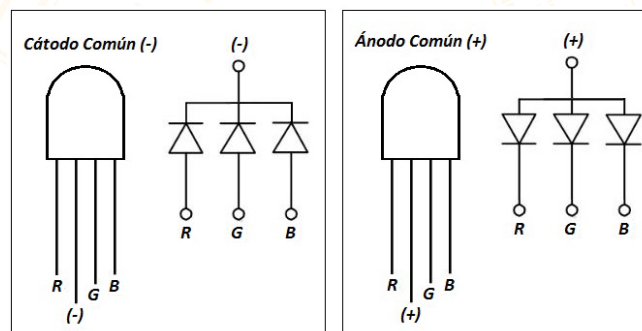
LED RGB

RGB es un sistema de color basado en tres colores primarios Rojo, Verde y Azul (por sus siglas en inglés (R) Red, (G) Green, (B) Blue), el cual en base a la combinación de estos 3 colores podemos representar otros colores como se muestra a continuación.



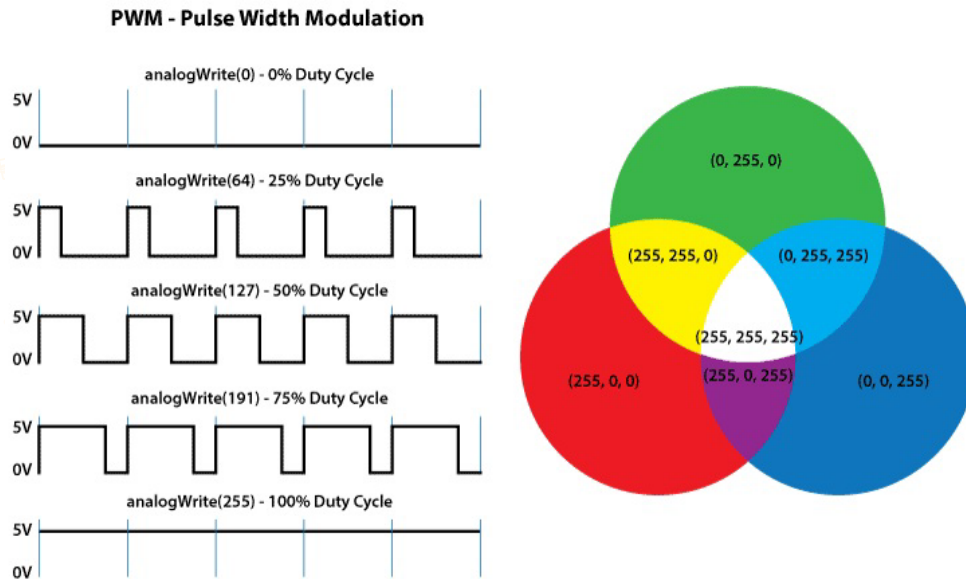
Por lo tanto, un Led RGB está formado internamente por tres diodos emisores de luz, uno rojo, uno verde y por último uno azul, con el propósito de poder crear, gran variedad de colores mediante la combinación de cada color con intensidades distintas.

Los Led's RGB tiene 4 terminales y es común utilizar el encapsulado de 5mm y hay 2 tipos de cátodo o ánodo común, que define cómo está conformado internamente el led RGB, (como se muestra a continuación), aunque en esta lección vamos a trabajar con un LED RGB de cátodo común.



Lección 3 Control de LED RGB

El control se recomienda que se haga por medio de PWM, de tal forma que, la resolución del PWM en cada color es de 8 bits, tendremos la posibilidad de generar más de 16 millones de colores distintos. De esta manera el led RGB brillará con una intensidad de color determinada en base al voltaje que se le suministre a cada diodo led interno, pudiéndose representar cualquier color derivado de los primarios.

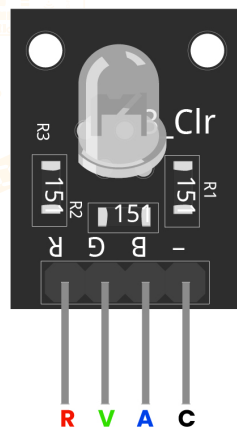


Sensor Led RGB

Para simplificar el circuito y las conexiones en general, vamos a utilizar el Sensor Led RGB Módulo KY-016 ya que integra el LED RGB de cátodo común y resistencias limitadoras de 150 Ω para evitar que se dañe el led RGB.

El módulo tiene 4 terminales, que hacen referencia a:

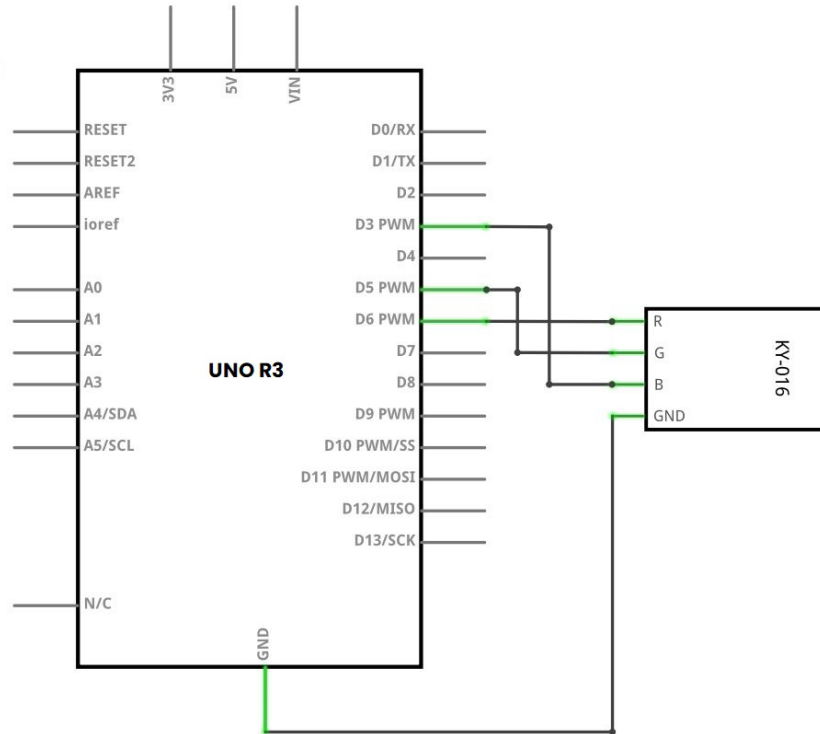
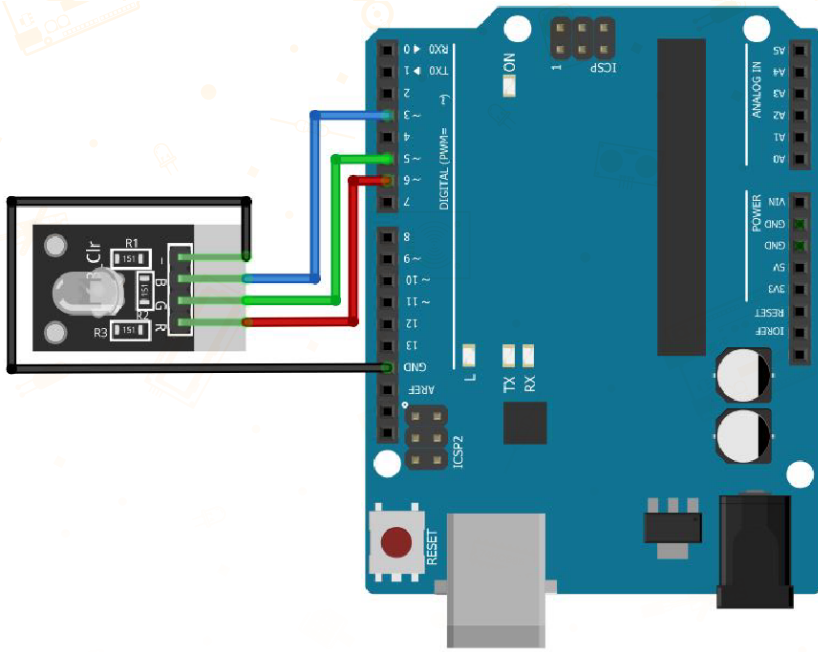
- 1 R: Pin de luz roja
- 2 V: pin de luz verde
- 3 A: pin de luz azul
- 4 C: Cátodo común (0V)



Lección 3 Control de LED RGB

Diagrama de Conexión

A continuación, se muestra cuál será la conexión del sensor led RGB con la tarjeta UNO R3.



Lección 3 Control de LED RGB

Código de Funcionamiento

Realizaremos un código para emitir los colores Rojo, Verde, Azul, Blanco, Morado y cambiarlos cada segundo. Se creará una función llamada setColor en el programa para que sea más limpia la forma en que se estará realizando el cambio de color en el led RGB.

```
//Uso de un Led RGB y cambio de algunos colores cada segundo
int rojoPin= 3;    // El control del color Rojo por el pin 3
int verdePin = 5; //El control del color Verde por el pin 5
int azulPin = 6;   //El control del color Azul por el pin 6

// Se definen como salidas a los pines 3, 5, 6; correspondientes al color del RGB
void setup() {
  pinMode(rojoPin, OUTPUT);
  pinMode(verdePin, OUTPUT);
  pinMode(azulPin, OUTPUT);
}
void loop() {

  // Valor del ciclo de trabajo PWM (0 a 255), que nos ayudará a tener control de la intensidad de
  //brillo por cada uno de los colores (rojo, verde, azul), con ayuda de la función "setColor"

  setColor(255, 0, 0); //Brillará en color Rojo
  delay(1000);        //Instrucción que durará 1s (1000ms)
  setColor(0, 255, 0); //Brillará en color Verde
  delay(1000);        //Instrucción que durará 1s (1000ms)
  setColor(0, 0, 255); //Brillará en color Azul
  delay(1000);        //Instrucción que durará 1s (1000ms)

  //Si colocamos el PWM de todos los colores en máximo
  setColor(255, 255, 255); //Brillara en Blanco
  delay(1000);            //Instrucción que durará 1s (1000ms)

  //Si realizamos combinación de los valores podremos obtener más gama de colores
  setColor(180, 0, 255); // Brillará en color Morado
  delay(1000);          //Instrucción que durará 1s (1000ms)
}
```

Lección 4

Zumbador Activo

Introducción

En esta lección aprenderemos a utilizar un buzzer activo para generar un tono mediante un pin digital de la tarjeta desarrollo UNO R3.

Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Zumbador activo x 1
- 4 Cables Macho – Macho x 2

Conocimientos previos

Buzzer activo es un pequeño dispositivo que puede producir un sonido cuando se aplica una corriente directa DC. Por ejemplo, si se conecta directamente a los 5V de la tarjeta UNO R3 este genera un tono fijo con una frecuencia ya predeterminada debido a que incorpora un oscilador interno. El buzzer activo tiene 2 terminales, negativo y positivo, comúnmente incorporan una etiqueta que nos indica cual es la terminal positiva mediante el símbolo (+), la terminal que se encuentre más cerca a esta referencia será la terminal positiva.



Hay 2 tipos de buzzer el activo y el pasivo, en la siguiente tabla se detallan las diferencias:

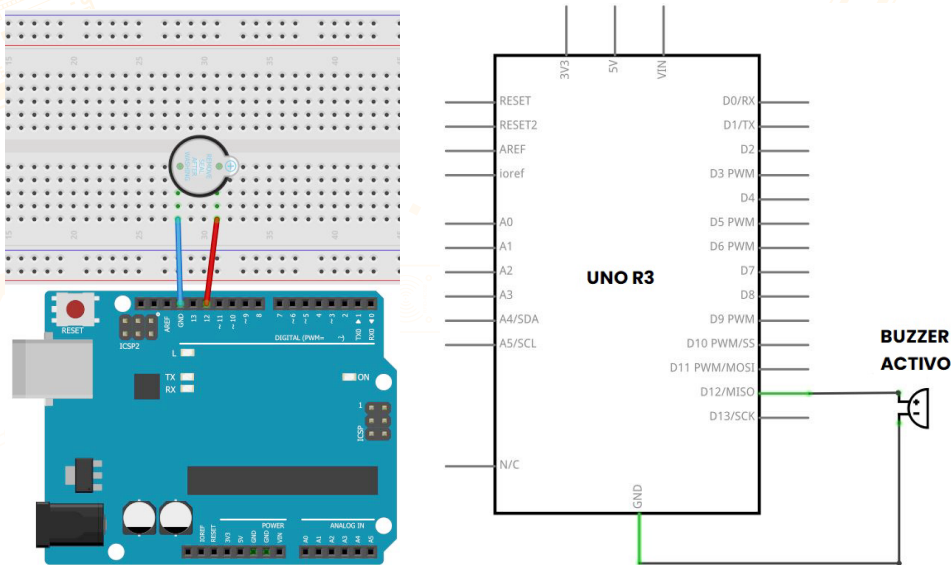
BUZZER ACTIVO	BUZZER PASIVO
Genera un tono fijo cuando se aplica una corriente directa DC.	Genera tonos a diferentes frecuencias con una señal o corriente alterna AC.
Frecuencia fija	Frecuencia de trabajo es desde los 100Hz a 10KHz
Oscilador interno	Sin oscilador interno

Para esta lección se utilizará un buzzer activo para demostrar su funcionamiento y a través de programación se variará el tiempo de encendido y pagado para generar tonos básicos.

Lección 4 Zumbador Activo

Diagrama de Conexión

A continuación, se muestra cuál será la conexión entre el buzzer y la tarjeta UNO R3.



Código de Funcionamiento

Realizaremos un código para controlar el buzzer activo y generar diferentes pitidos por 50ms.

```
//Encendido y control de tono por medio de la tarjeta Uno R3
int buzzer = 12;           // Pin 12 para el buzzer activo
void setup()
{
  pinMode(buzzer,OUTPUT); //Declaración de pin de salida asignado al buzzer
}
void loop(){
  unsigned char i;        //Variable tipo carácter sin signo llamada "i" para simular una frecuencia
  for(i=0;i<80;i++){      //Generación de frecuencia de pitidos
    digitalWrite(buzzer,HIGH); //El buzzer emitirá un pitido...
    delay(50);             //...por 50ms
    digitalWrite(buzzer,LOW); //El buzzer se apagará...
    delay(50);            //...por 50ms
  }
  digitalWrite(buzzer,LOW); //pausa de sonido...
  delay(5000);            //por 5 segundos
}
```

Lección 5

Zumbador Pasivo

En esta lección aprenderemos a utilizar el buzzer pasivo y la tarjeta UNO R3 para generar ocho sonidos diferentes, cada sonido durará 1 segundos desde Do (262Hz), Re (294Hz), Mi (330Hz), Fa (349Hz), Sol (392Hz), La (440Hz), Si (494Hz) hasta Do Agudo (524Hz).

Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Zumbador pasivo x 1
- 4 Cables Macho – Macho x 2

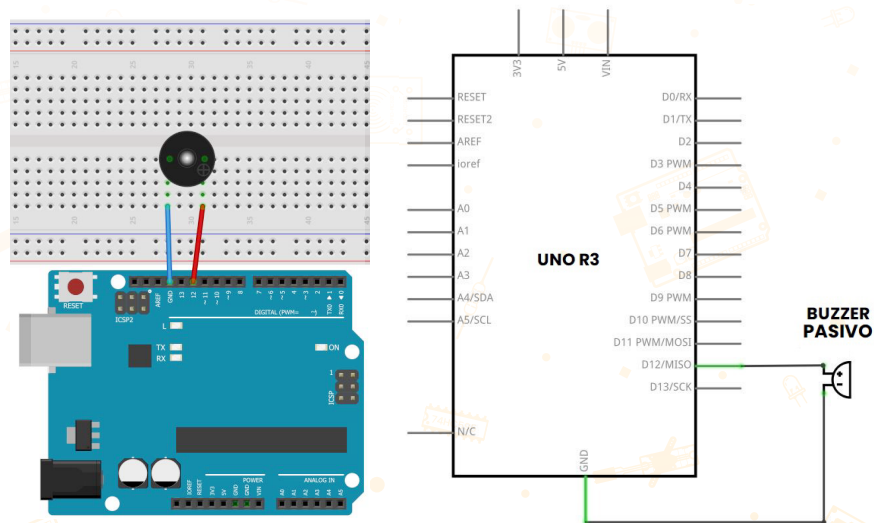
Conocimientos previos

Buzzer pasivo es un pequeño dispositivo capaz de generar tonos a diferente frecuencia cuando se aplica una señal o corriente alterna AC. Por ejemplo, si se aplica una señal analógica PWM desde la tarjeta UNO R3 el buzzer producirá un tono con diferente volumen que varía dependiendo del ciclo de trabajo del PWM.

Para cambiar la frecuencia del tono y generar diferentes melodías se recomienda aplicar una señal de onda cuadrada con una frecuencia específica, para este caso se recomienda utilizar en programación la función `Tone()` la cual permitirá definir 3 parámetros, el pin del buzzer la frecuencia y la duración de la nota. El buzzer pasivo tiene 2 terminales, negativo y positivo, para identificar la terminal positiva el buzzer comúnmente incorpora en la parte superior una referencia con el símbolo (+), la terminal que se encuentre más cerca a esta referencia será la terminal positiva.

Diagrama de Conexión

A continuación, se muestra cuál será la conexión entre el buzzer pasivo y la tarjeta UNO R3.



Lección 5 Zumbador Pasivo

Código de Funcionamiento

Realizaremos un código para demostrar el funcionamiento del buzzer pasivo el cual permite generar 8 notas básicas Do, Re, Mi, Fa, So, La y Si.

```
//Se usará el buzzer pasivo para generar 8 notas básicas: Do (262Hz), Re (294Hz),
//Mi (330Hz), Fa (349Hz), So (392Hz), La (440Hz), Si (494Hz) cada segundo
//Se declaran las variables de las notas, con los valores de frecuencia
int Do = 262, Re = 294, Mi = 330, Fa = 349, Sol = 392, La = 440, Si = 494, Do2 = 524;
int buzz = 12; //Pin 12 para control del buzzer pasivo
int wait = 0; //Variable para control de tiempo inicializada con valor 0
void setup()
{
  pinMode(buzz, OUTPUT); //Inicialización del pin 12 como salida
}
void loop()
{
  wait = 500; //Duración de cada una de las notas será de 500ms / .5 segundos
  tone(buzz, Do, wait); //Generación de onda cuadrada en el pin 12, con valor de DO

  delay(1000); //Espera 1s
  tone(buzz, Re, wait); //Generación de onda cuadrada en el pin 12, con valor de RE

  delay(1000); //Espera 1s
  tone(buzz, Mi, wait); //Generación de onda cuadrada en el pin 12, con valor de MI

  delay(1000); //Espera 1s
  tone(buzz, Fa, wait); //Generación de onda cuadrada en el pin 12, con valor de FA

  delay(1000); //Espera 1s
  tone(buzz, Sol, wait); //Generación de onda cuadrada en el pin 12, con valor de SOL

  delay(1000); //Espera 1s
  tone(buzz, La, wait); //Generación de onda cuadrada en el pin 12, con valor de LA
  delay(1000); //Espera 1s
  tone(buzz, Si, wait); //Generación de onda cuadrada en el pin 12, con valor de SI y una
  delay(1000); //Espera 1s
  tone(buzz, Do2, wait); //Generación de onda cuadrada en el pin 12, con valor de DO sostenido
  delay(1000); //Espera 1s
  noTone(buzz); //Terminara la reproducción de sonido
}
```

Lección 6

Ocho Led Controlados con el 74HC595

Introducción

En esta lección aprenderemos a utilizar el circuito 74HC595 para encender 8 led's mediante la utilización de 3 pines de la tarjeta UNO R3. Ya que el CI 74HC595 es un registro de desplazamiento que permite ampliar el número de pines de E/S de la tarjeta UNO R3 o cualquier otro microcontrolador MCU.

Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Led de 5mm x 8
- 4 Resistencia de 220 Ohms x 8
- 5 IC 74HC595 x 1
- 6 Cables Macho – Macho x 18

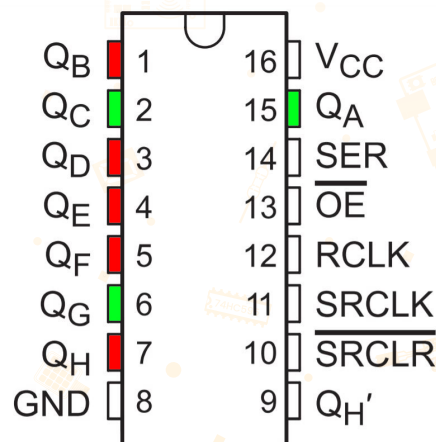
Conocimientos previos

El circuito integrado CI 74HC59 o SN74HC595N consta de un registro de desplazamiento de 8 bits y de un registro de almacenamiento con salidas paralelas en tres estados.



Convierte la entrada serie en salida paralela para que usted pueda guardar los puertos IO de un MCU. El chip contiene ocho pines que podemos utilizar como salida, cada uno de ellos está asociado con un bit en el registro. En el caso del CI 74HC595, nos referimos a ellos desde QA hasta QH. Para escribir estas salidas con la tarjeta UNO R3, tenemos que enviar un valor binario al registro de desplazamiento y con ese número el registro de desplazamiento puede comprender qué salidas utilizar.

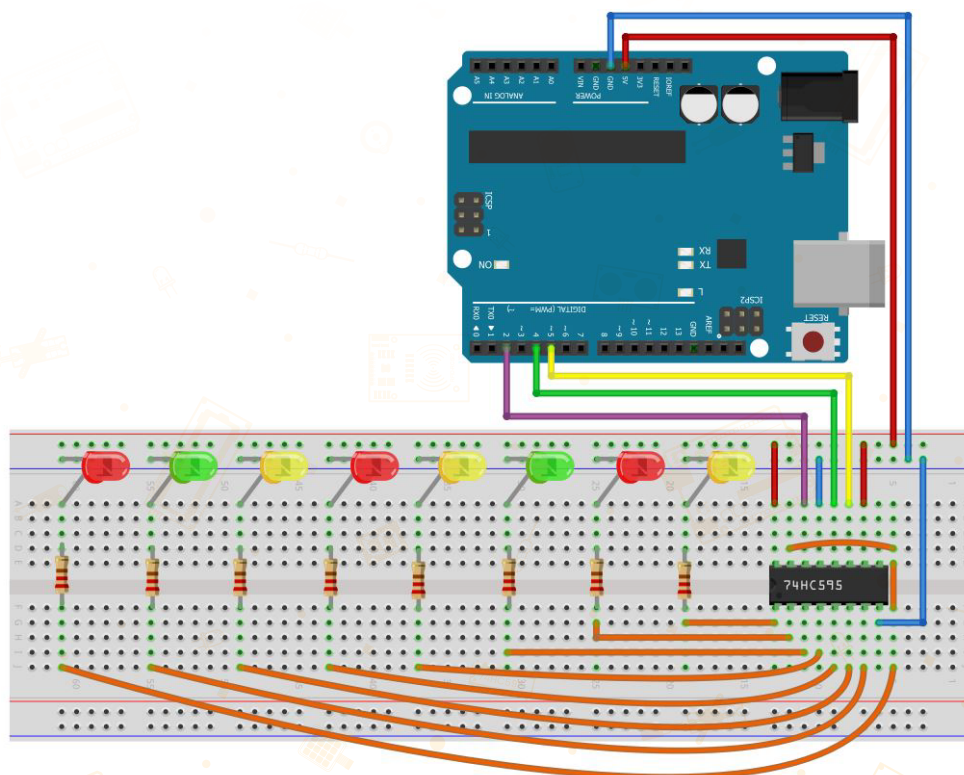
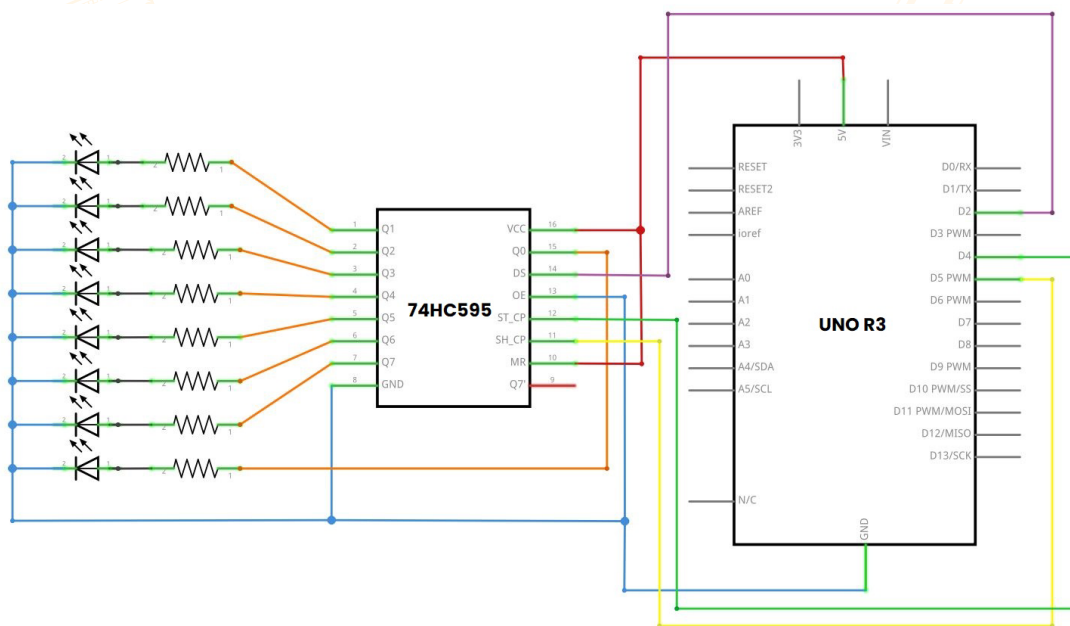
Por ejemplo, si enviamos el valor binario 10100010, los pines destacados en verde en la imagen siguiente estarán activos y los destacados en rojo permanecerán inactivos.



Lección 6 Ocho Led Controlados con el 74HC595

Diagrama de Conexión

A continuación, se muestra cuál será la conexión de 8 led's conectados al IC 74HC595 y a la tarjeta UNO R3.



Lección 6 Ocho Led Controlados con el 74HC595

Código de Funcionamiento

Realizaremos un código para demostrar el funcionamiento del IC 74HC595 para encender y apagar 8 led's utilizando 3 pines de la tarjeta UNO R3.

```

int RCLKPin = 4; //Usaremos el Pin 4 de la tarjeta UNO R3 para el manejo del Reloj de
                //Registro (Pin 12 RCLK del 74HCC595).

int SRCLKPin = 5; //Usaremos el Pin 5 de la tarjeta UNO R3 para el manejo del Reloj
                //de Registro de desplazamiento (Pin 11 SRCLK del 74HCC595).

int datoPin = 2; //Usaremos el Pin 2 de la tarjeta UNO R3 para entrada Serial del Dato
                //(Pin 14 SER del 74HCC595).

byte leds = 0; //Variable llamada leds con espacio de 8 bits, donde se
                //guardará el patrón de encendido/apagado de leds

int nextLED = 0; //Variable para detectar el byte en la variable leds
void setup()
{
//Declaración de variables, como salidas digitales
  pinMode(RCLKPin, OUTPUT);
  pinMode(datoPin, OUTPUT);
  pinMode(SRCLKPin, OUTPUT);
  leds = 0; //Inicialización del programa y de la variable tipo byte en 0
}
void loop()
{
  leds = 0; //Limpieza del bucle de la variable byte en 0, para que el led encienda una vez
  if (nextLED == 7) //Se comienza el corrimiento de los bits 0, si se llega a 7
  {
    nextLED = 0; //Reiniciamos el avance en 0
  }
  else { nextLED++; //De lo contrario seguirá avanzando hasta llegar a 8 bits
  }
  bitSet(leds, nextLED); //Usará la función bitSet para el control de c/led dependiendo el avance del bit
  digitalWrite(RCLKPin, LOW); //Pone en nivel bajo a latchPin durante la transmisión
  shiftOut(datoPin, SRCLKPin, MSBFIRST, leds);

//Función shiftOut para desplazar indicando por que pin sale el bit (dataPin), cambio del pin una vez que //dataPin
//tenga el valor adecuado(clockPin), dirección de desplazamiento (LSBFIRST o MSBFIRST) los //datos que se
//desplazaran
  digitalWrite(RCLKPin, HIGH); //Devuelve el latchpin a nivel alto
  delay(1000); //Un segundo de tiempo para esta rutina
}

```


Lección 7

Fotorresistor

Introducción

En esta lección aprenderás a medir la intensidad de la luz mediante una fotorresistencia, se utilizará el circuito de la lección 6 para que cuando haya oscuridad se prendan los leds y si hay iluminación se apaguen uno a uno los leds con la ayuda del IC 74HC595 y la utilizando de una entrada analógica de la tarjeta UNO R3.

Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Led's de 5mm x 8
- 4 Resistencia de 220 Ohms x 8
- 5 IC 74HC595 x 1
- 6 Fotorresistencia LDR x 1
- 7 Resistencia de 1k x 1
- 8 Cables Macho – Macho x 20

Conocimientos previos

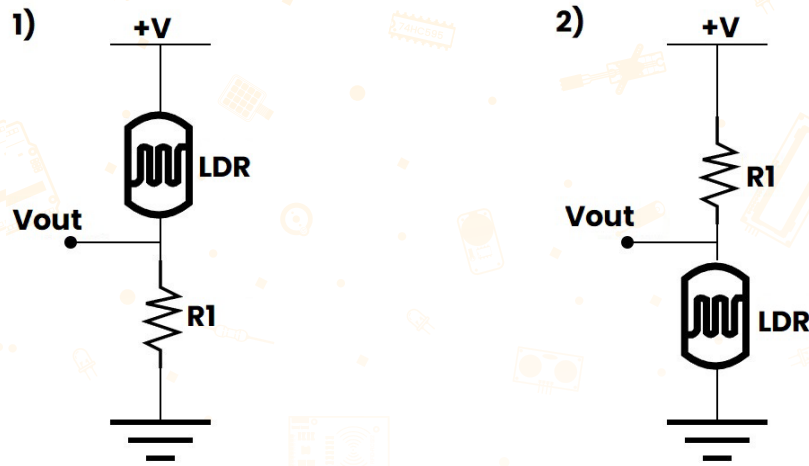
Una fotorresistencia o LDR (por sus siglas en inglés "light-dependent resistor") es un componente electrónico cuya resistencia varía en función de la luz. Se trata de un sensor que actúa como una resistencia variable en función de la luz que capta. A mayor intensidad de luz, menor resistencia.



Para conocer la cantidad de luz que el sensor capta en cierto ambiente, necesitamos medir la tensión de salida de este. Para lograr esto se debe realizar un divisor de tensión colocando una resistencia en serie. Hay dos formas básicas para conectar una LDR:

- 1 Mayor luz, mayor voltaje: Al conectar la fotorresistencia al nodo positivo de nuestra fuente de voltaje tendremos que, al incidir una mayor cantidad de luz provocará una menor caída de voltaje o diferencia de potencial entre la fuente y el pin de referencia (V_{out}), por lo tanto, se tendrá una lectura mayor.
- 2 Mayor luz, menor voltaje: En pocas palabras la fotorresistencia se conecta al nodo de GND y provocará un comportamiento opuesto al punto 1.

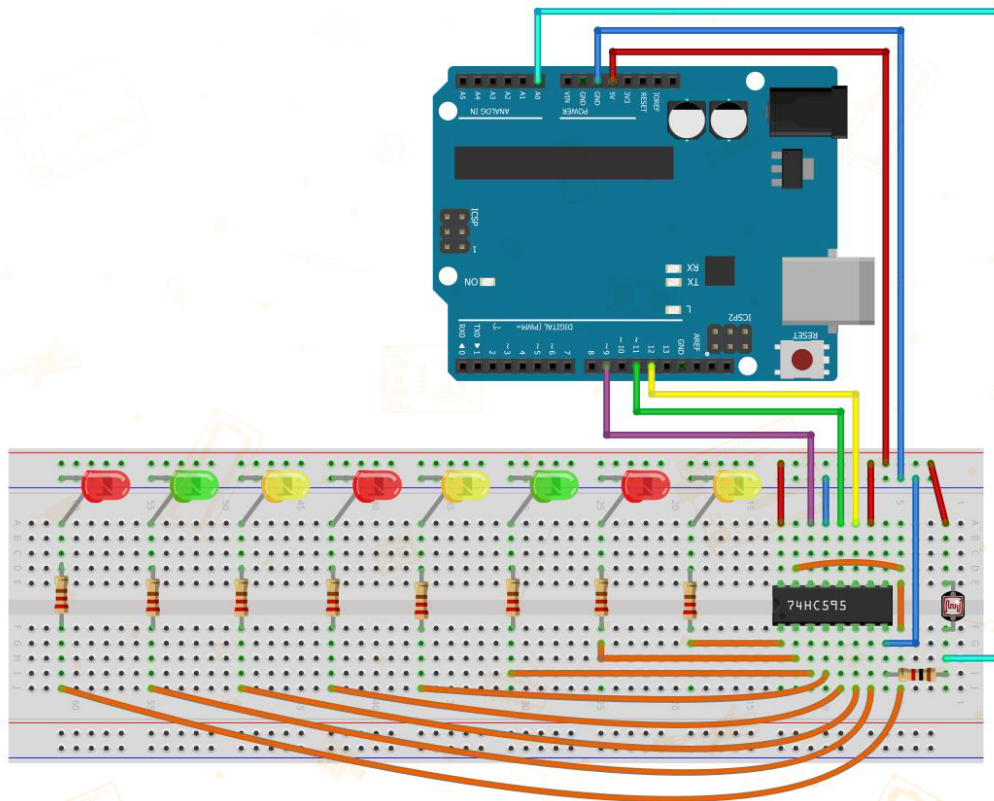
Lección 7 Fotoresistor



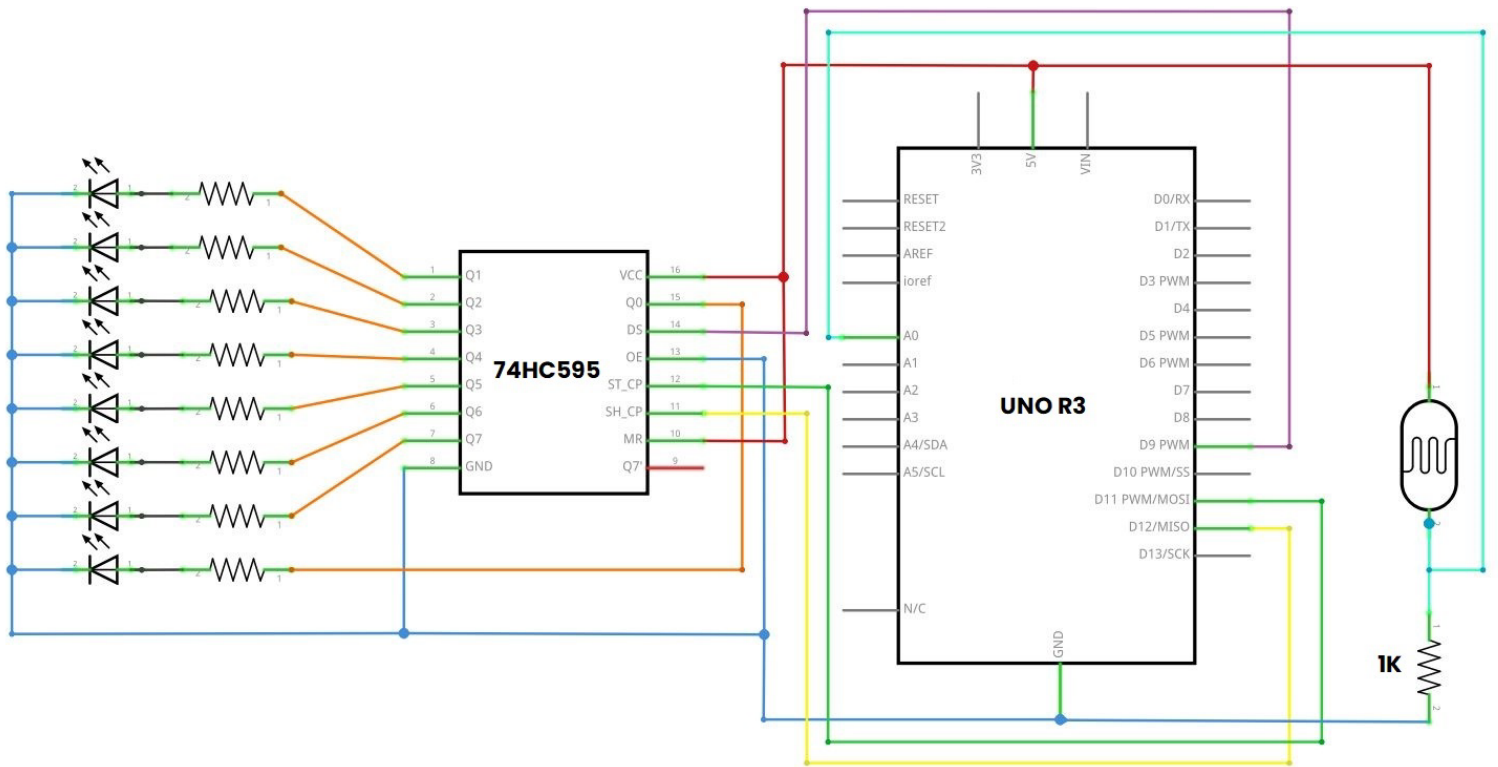
Para esta lección se utilizará la conexión 1 “Mayor luz, mayor voltaje”, donde R1 conectaremos una resistencia de 1K, +V lo conectaremos a 5V de la tarjeta UNO R3 y por último Vout a un pin analógico para esta práctica utilizaremos el pin A0.

Diagrama de Conexión

A continuación, se muestra cuál será la conexión de la LDR con la tarjeta UNO R3 y con el circuito de la práctica 6.



Lección 7 Fotoresistor



Lección 7 Fotoresistor

Código de Funcionamiento

Realizaremos un código para que cuando haya oscuridad se prendan los leds y si hay iluminación se apagaran uno a uno los leds con la ayuda del IC 74HC595 y la utilizando de una entrada analógica de la tarjeta UNO R3.

```

int lightPin = 0;    // Pin analógico A0 de la tarjeta UNO R3 para conectar la LDR

int latchPin = 11;  // Usaremos el Pin 11 para el manejo del Reloj de Registro (Pin 12 RCLK del 74HCC595).

int clockPin = 12;  // Usaremos el Pin 12 para el manejo del Reloj de Registro de Desplazamiento
                    //(Pin 11 SRCLK del 74HCC595)

int dataPin = 9;    // Usaremos el Pin 9 para entrada Serial del Dato (Pin 14 SER del //74HCC595)

int leds = 0;       // Variable entera para el registro de leds

void setup()
{
  // Declaración de variables, como de salida digital
  pinMode(latchPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
}

// El valor que se obtiene la fotorresistencia es analógico entre 0 a 1023
// 0 (oscuro) y 1023(luz)

void loop()
{
  int reading = analogRead(lightPin); // Lectura del valor analógico del fotorresistor

  int numLEDSLit = reading / 63;      // 1023 entre 8 (8 bites) / 2 = ~63

  numLEDSLit = 9 - numLEDSLit;        // Para cambiar el sentido de cambio de encendido en los leds

  if (numLEDSLit > 8) numLEDSLit = 8; // Si el valor del led es mayor que 8, el valor será igual a 8

  leds = 0;                           // ...no habrá iluminación en los leds

  for (int i = 0; i < numLEDSLit; i++) // tendremos un incremento de la variable i, hasta que
                                        // sea menor a numLEDSLit
  {
    leds = leds + (1 << i);           // Establece el i-décimo bit
  }
  actualizaRegistro();                // Se ejecuta la función
}

void actualizaRegistro()              // Función para realizar el encendido de los leds a partir del Reloj de Registro
{
  digitalWrite(latchPin, LOW);        // Pone en nivel bajo a latchPin, led apagado
  shiftOut(dataPin, clockPin, LSBFIRST, leds); // función shiftOut para desplazar el bit,
  // Indicando porque pin sale el bit (dataPin), cambio del pin una vez que dataPin tenga el valor //
  // adecuado(clockPin),
  // dirección de desplazamiento (LSBFIRST o MSBFIRST) los datos que se //desplazaran(leds)byte
  digitalWrite(latchPin, HIGH);       // pone en nivel bajo a latchPin, led encendido
}

```

Lección 8

74HC595 y Display de 7 segmentos

Introducción

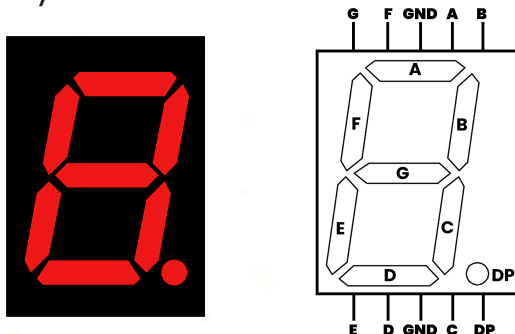
En esta lección aprenderás a utilizar el display de 7 segmentos cátodo común con el registro de desplazamiento 74HC595 para mostrar números del 0 al 9 y las letras A, B y C, con ayuda de la tarjeta UNO R3.

Materiales necesarios

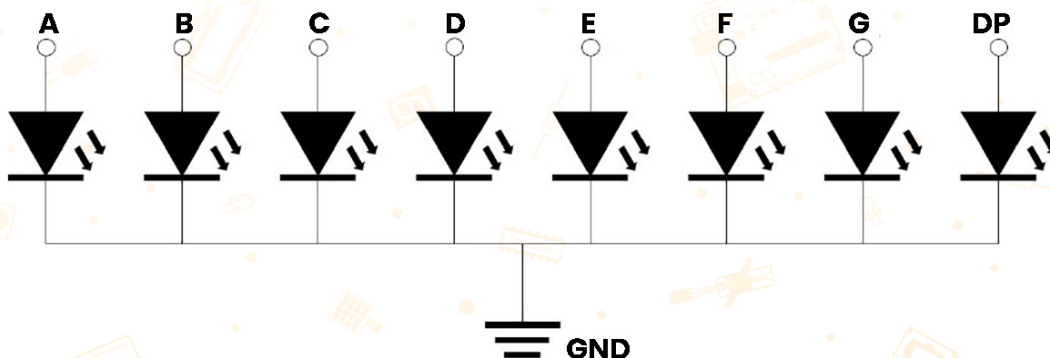
- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Resistencia de 220 Ohms x 1
- 4 IC 74HC595 x 1
- 5 Display 7 segmentos cátodo común x 1
- 6 Cables Macho – Macho x 18

Conocimientos previos

El display 7 Segmentos es un dispositivo optoelectrónico que permite visualizar números del 0 al 9. Está compuesto por 8 led's, 7 son para segmentos de un dígito (mostrados como A a G) y el otro LED es para el punto decimal (mostrado cómo DP). En la siguiente imagen se muestra los pines del display:



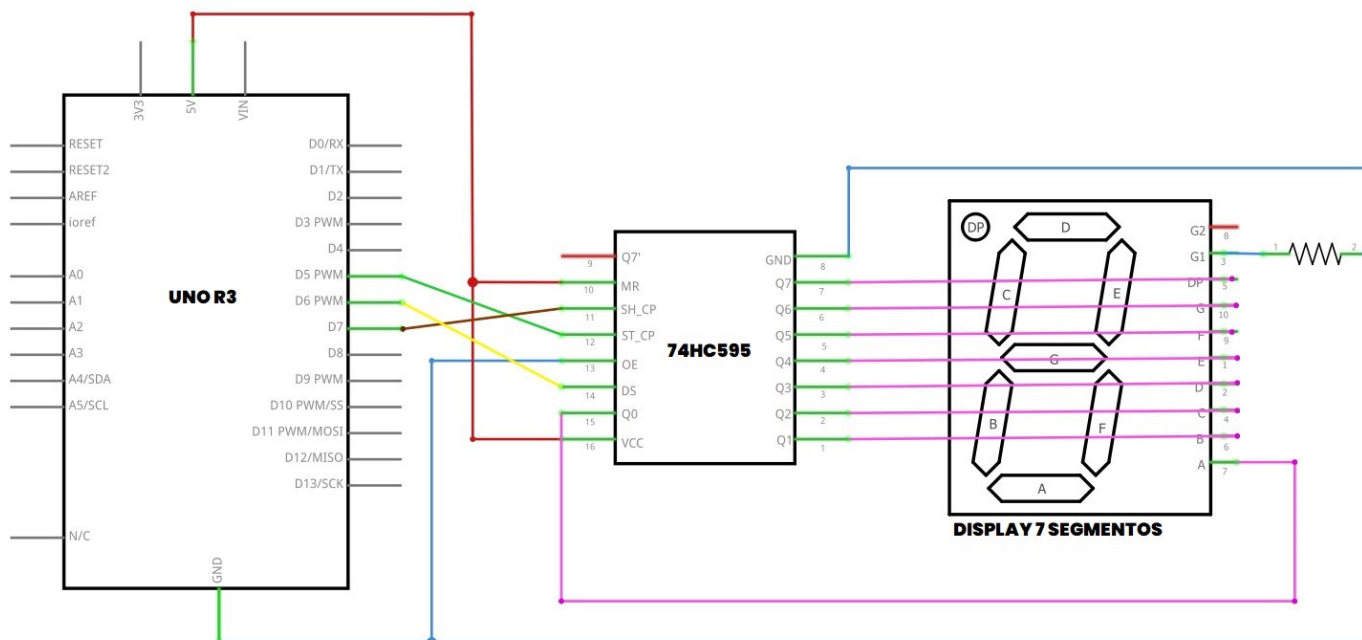
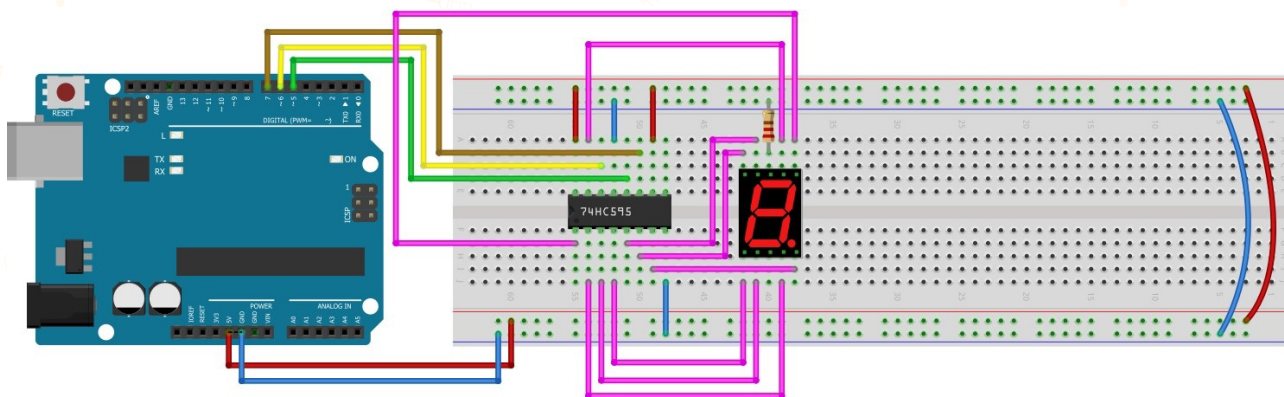
Hay dos tipos de display, de cátodo común y de ánodo común. Para esta lección utilizaremos display de cátodo común el cual internamente está conectado de la siguiente manera:



Lección 8 74HC595 y Display de 7 segmentos

Diagrama de Conexión

A continuación, se muestra cuál será la conexión del display de 7 segmentos con el IC 74HC595 y la tarjeta UNO R3. Se utilizará una resistencia de 220 ohms la cual estará conectada al pin común del display y al GND de la tarjeta UNO R3.



Lección 8 74HC595 y Display de 7 segmentos

Código de Funcionamiento

Realizaremos un código para controlar un display de 7 segmentos de forma que podamos contar del 0 al 9 mostrar letras A, B y C, con ayuda del IC 74HC595 y la tarjeta UNO R3.

```
int latchPin = 5;           // Pin conectado al Pin 12 del 74HC595 (Latch)
int clockPin = 7;          // Pin conectado al Pin 11 del 74HC595 (Clock)
int dataPin = 6;           // Pin conectado al Pin 14 del 74HC595 (Data)
int i = 0;                 //Variable para realizar conteo

//Combinación de bit por led del display de 7 segmentos realizando una secuencia
//desde 0 a la letra C: A,B,C,D,E,F,G,PUNTO DECIMAL
const byte numeros[14] = {
  B1111101,    // = 0.
  B01100000,   // = 1
  B11011010,   // = 2
  B11110010,   // = 3
  B01100110,   // = 4
  B10110110,   // = 5
  B10111110,   // = 6
  B11100000,   // = 7
  B11111110,   // = 8
  B11100110,   // = 9
  B11101110,   // = A
  B00111110,   // = B
  B10011101,   // = C
};

void setup() {
  Serial.begin(9600);
  //Declaración de variables, como de salida digital
  pinMode(latchPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
}

void loop() {
  for (i = 0; i < 14; i++) { //repetición de la secuencia de encendido y apagado de byte del 1 al 14
                              //anteriormente declarada en la función números
    digitalWrite(latchPin, LOW); //desactiva a Reloj de Registro de Desplazamiento

    //función shifOut que nos ayudará a controlar al 74HC para desplazar el bi
    //,indicando por que pin sale el bit (dataPin),cambio del pin una vez
    // que dataPin tenga el valor adecuado(clockPin), dirección de desplazamiento
    //(LSBFIRST o MSBFIRST) los datos que se desplazarán(leds)byte
    shiftOut(dataPin, clockPin, LSBFIRST, numeros[i]);
    digitalWrite(latchPin, HIGH); //activado el Reloj de Registro de Desplazamiento
    delay(1000);                  //espera de 1 segundo entre cada secuencia
  }
}
```

Lección 9

74HC595 y Display 4 Dígitos 7 Segmentos

Introducción

En esta lección aprenderás a utilizar el display 4 dígitos 7 segmentos de cátodo común con el registro de desplazamiento 74HC595 para mostrar y cambiar HALO a 0000 o EEEE, por ejemplo.

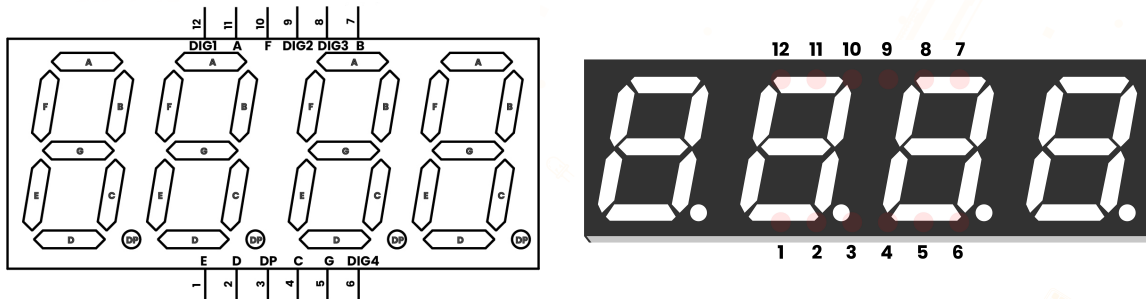
Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Resistencia de 220 Ohms x 4
- 4 IC 74HC595 x 1
- 5 Display 4 Dígitos 7 Segmentos cátodo común x 1
- 6 Cables Macho – Macho x 30

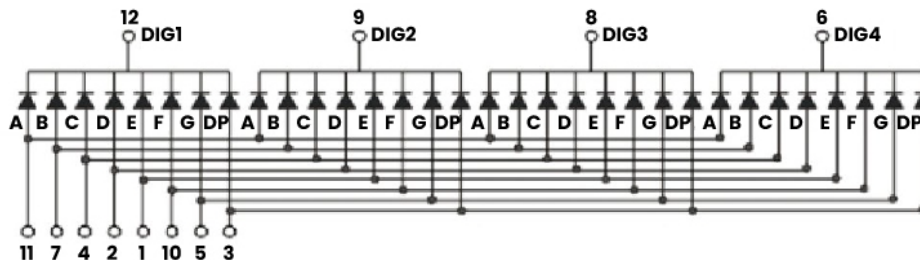
Conocimientos previos

El display 4 dígitos 7 Segmentos es un dispositivo optoelectrónico que permite visualizar números del 0 al 9. Se utiliza para representar visualmente números y algunos caracteres. El display está compuesto por 4 dígitos, cada dígito tiene 7 segmentos y un punto decimal que pueden encender o apagar individualmente.

En la siguiente imagen se muestra los pines del display 4 dígitos 7 Segmentos:



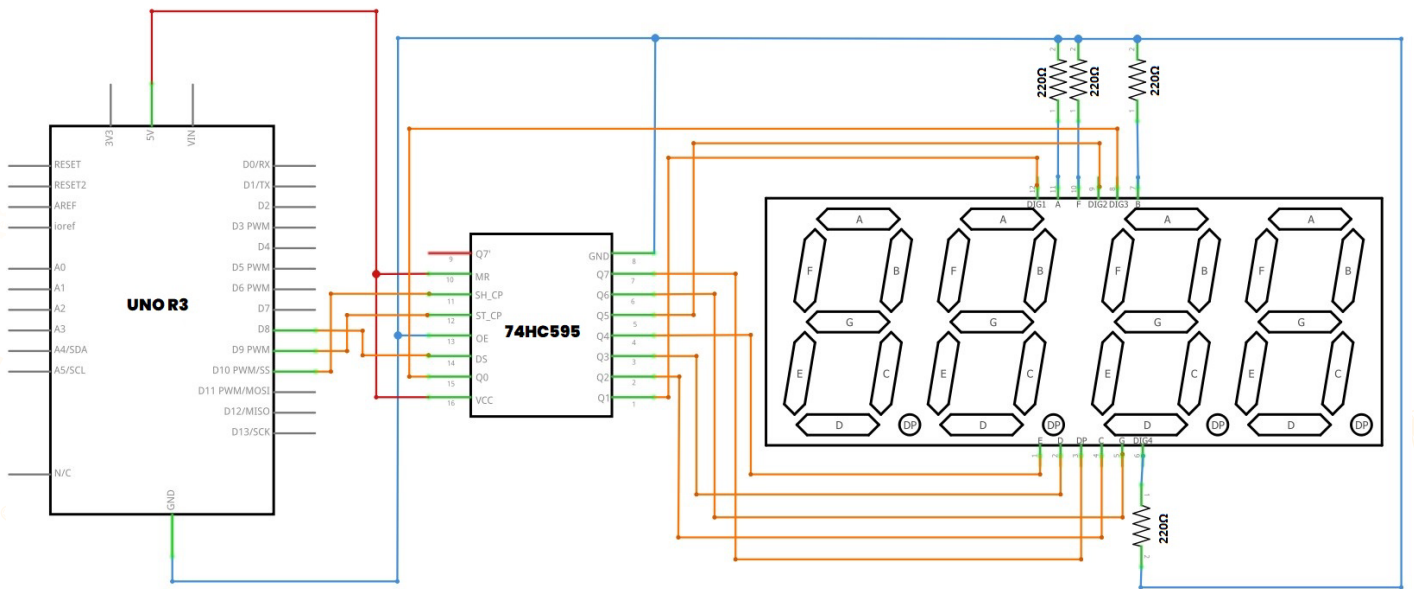
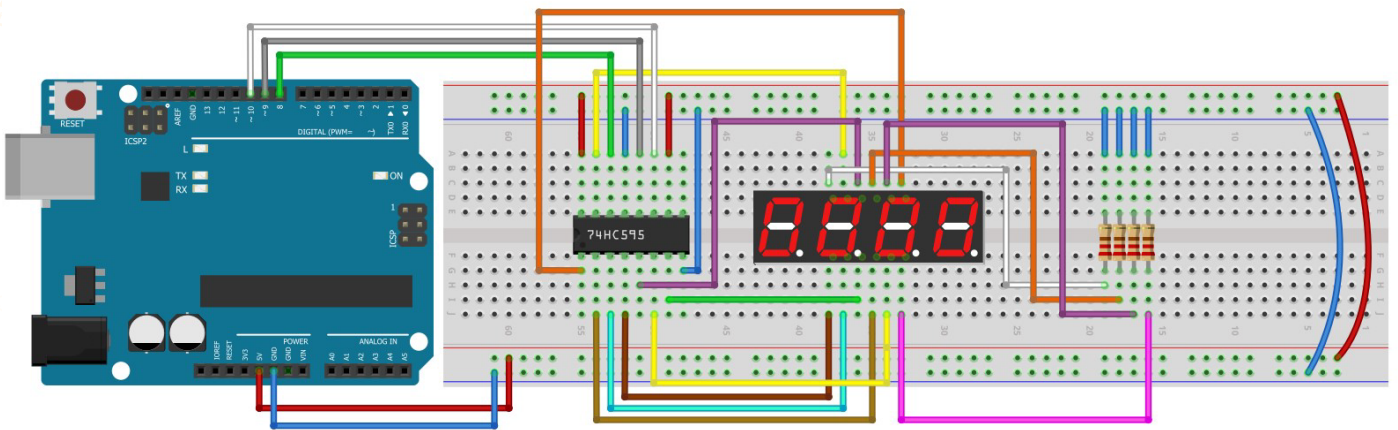
Hay dos tipos de display de 4 dígitos, de cátodo y ánodo común. Para esta lección utilizaremos display de cátodo común el cual internamente está conectado de la siguiente manera:



Lección 9 74HC595 y Display 4 Dígitos 7 segmentos

Diagrama de Conexión

A continuación, se muestra cuál será la conexión del display de 4 dígitos con el IC 74HC595 y la tarjeta UNO R3. Se utilizarán 4 resistencias de 220 ohms, que se conectarán a los pines de cátodo común del display y al GND de la tarjeta UNO R3.



Lección 9 74HC595 y Display 4 Dígitos 7 segmentos

Código de Funcionamiento

Realizaremos un código para controlar el display 4 dígitos 7 segmentos de forma que pueda mostrar y cambiar caracteres como HALO a 0000 o EEEE, por ejemplo. Utilizando el IC 74HC595 y la tarjeta UNO R3.

```
int latch=9;           //Pin 9 al 12 (STCP) del 74HC595
int clockPin=10;      //Pin 10 al 11 (SHCP) del 74HC595
int data=8;           //Pin 8 al 14 (DS) del 74HC595

//Matriz de codificación del display de 7 segmentos - hexadecimal
unsigned char table[]={0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f,0x77,0x7c
,0x39,0x5e,0x79,0x71,0x00};

void setup() {        //inicializa pines de salida
  pinMode(latch, OUTPUT);
  pinMode(clockPin, OUTPUT);
  pinMode(data, OUTPUT);}

//Función Display en donde controlaremos el Latch del dato
void Display(unsigned char num){
  digitalWrite(latch,LOW);
  shiftOut(data, clockPin, MSBFIRST,table[num]);
  digitalWrite(latch, HIGH);
}

//impresión de los valores de la matriz dependiendo del orden que fueron agregados los datos
void loop() {
  Display(1);        //Se llama a la Función Display para mostrar el primer elemento de la matriz
  delay(2000);      //2 segundos de espera para comenzar cada instrucción
  Display(2);
  delay(2000);
  Display(3);
  delay(2000);
  Display(4);
  delay(2000);
  Display(5);
  delay(2000);
  Display(6);
  delay(2000);
  Display(7);
  delay(2000);
  Display(8);
  delay(2000);
  Display(9);
  delay(2000);
  Display(10);
  delay(2000);
  Display(11);
  delay(2000);
  Display(12);
  delay(2000);
  Display(13);
  delay(2000);
  Display(14);
  delay(2000);
  Display(15);
  delay(2000);
}
```

Lección 10

Sensor de Inclinación

Introducción

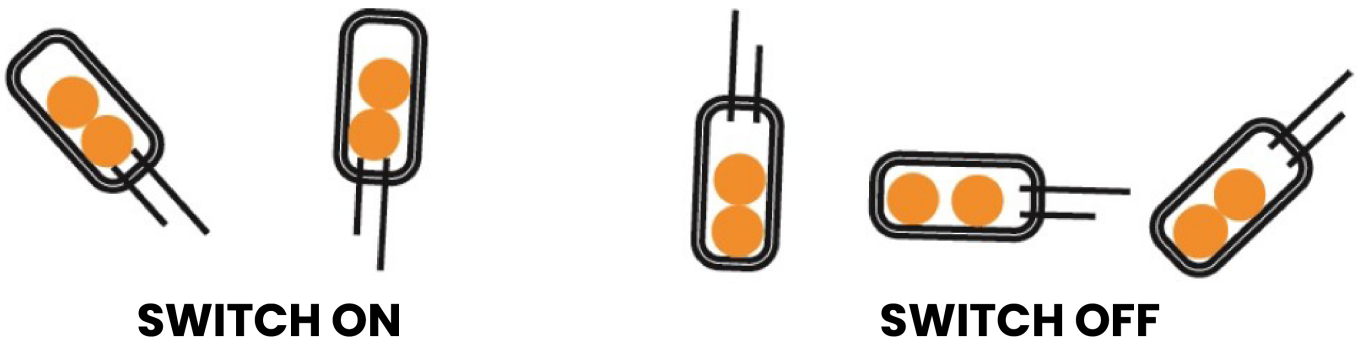
En esta lección aprenderás cómo usar un sensor de inclinación para detectar pequeños ángulos de inclinación.

Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Led de 5mm rojo x 1
- 4 Resistencia de 220 Ohms x 2
- 5 SW-520D Sensor de inclinación x 1
- 6 Cables Macho – Macho x 5

Conocimientos previos

El SW-520D sensor de inclinación es un dispositivo que proporciona una señal en caso de que su inclinación supere un umbral. Este tipo de sensor no permite saber el grado de inclinación, simplemente actúa como un sensor que se cierra a partir de una cierta inclinación. Está constituido por un cilindro que en su interior tiene un par de esferas metálicas que al inclinarse cierran un circuito el cual será detectado en el interior, por lo que es usado para detectar vibraciones y/o inclinaciones.

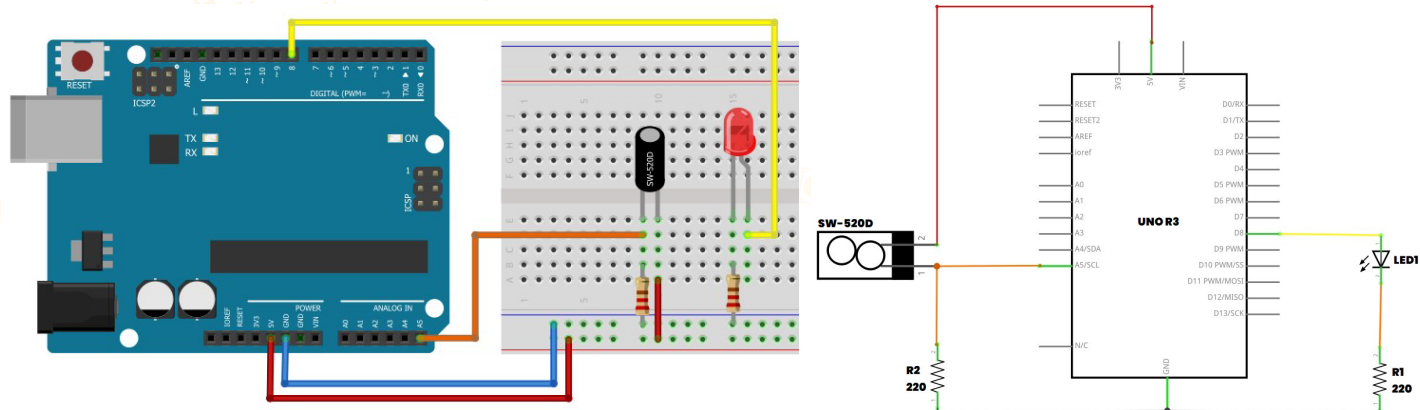


Como se muestra en la imagen, cuando el sensor se inclina hacia el extremo de la tapa y el ángulo de inclinación es superior a 10 grados, estará abierto (OFF), de lo contrario estará el estado cerrado (ON). Estos estados los podrás interpretar con la tarjeta UNO R3 para encender y apagar un led dependiendo de la inclinación del sensor.

Lección 10 Sensor de Inclinación

Diagrama de Conexión

A continuación, se muestra cuál será la conexión con el sensor de inclinación con la tarjeta UNO R3 y el led que encenderá o se apagará dependiendo a la inclinación del sensor.



Código de Funcionamiento

Realizaremos un código para leer el estado del sensor de inclinación para encender o apagar un led dependiendo de la inclinación del sensor.

```

const int LedPin=8;           //Variable para el Pin 8 que se conectará con el Led
void setup()
{
  pinMode(LedPin,OUTPUT);
}
void loop()                  //Función Loop
{
  int i;
  while(1)                   //While para tener una instrucción bucle sin fin
  {
    i=analogRead(5);         //La variable i tendrá lectura de los valores provenientes del pin 5 analógico
    if(i > 512)              //rango ajustable de 0 a 1023, 512 tendremos una sensibilidad media en el sensor
    {
      digitalWrite(LedPin, LOW); //El led apagará, en posición vertical del sensor
    }
    else
    {
      digitalWrite(LedPin, HIGH); //El led se encenderá, en posición horizontal del sensor
    }
  }
}

```

Lección 11

Módulo de Sensor Ultrasónico HC-SR04

Introducción

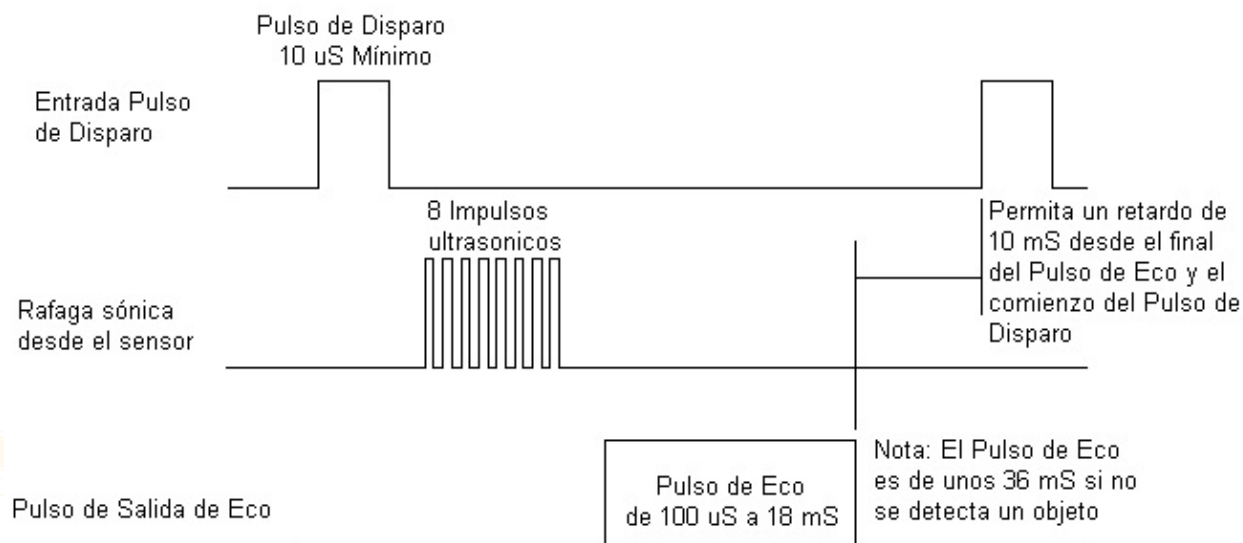
En esta práctica aprenderemos cómo usar el sensor ultrasónico HC-SR04, el cual puede ser útil en la medición de distancia, detección de objetos y evadir obstáculos. En nuestro caso lo usaremos para detección de presencias y que se pueda activar una alarma con ayuda de un buzzer activo.

Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Cables Macho – Macho
- 4 Sensor HC-SR04
- 5 Buzzer Activo

Conocimientos previos

El sensor HC-SR04 cuenta con un rango de medición de 2cm a 400 cm, mide el tiempo entre el envío y recepción del pulso. Conformado por un Trigger y Echo, el primero efectuará el disparo del ultrasonico y el segundo la recepción. Usando el Trigger para al menos 10 μ s de señal de alto nivel, envía automáticamente 8 pulsos a una frecuencia de 40kHz y detecta si hay una señal de pulso de regreso.



Si la señal regresa, a través de un nivel alto, el tiempo de salida del Trigger de salida alta es el tiempo desde el envío de ultrasonidos hasta el retorno y captado en Echo.

Lección 11

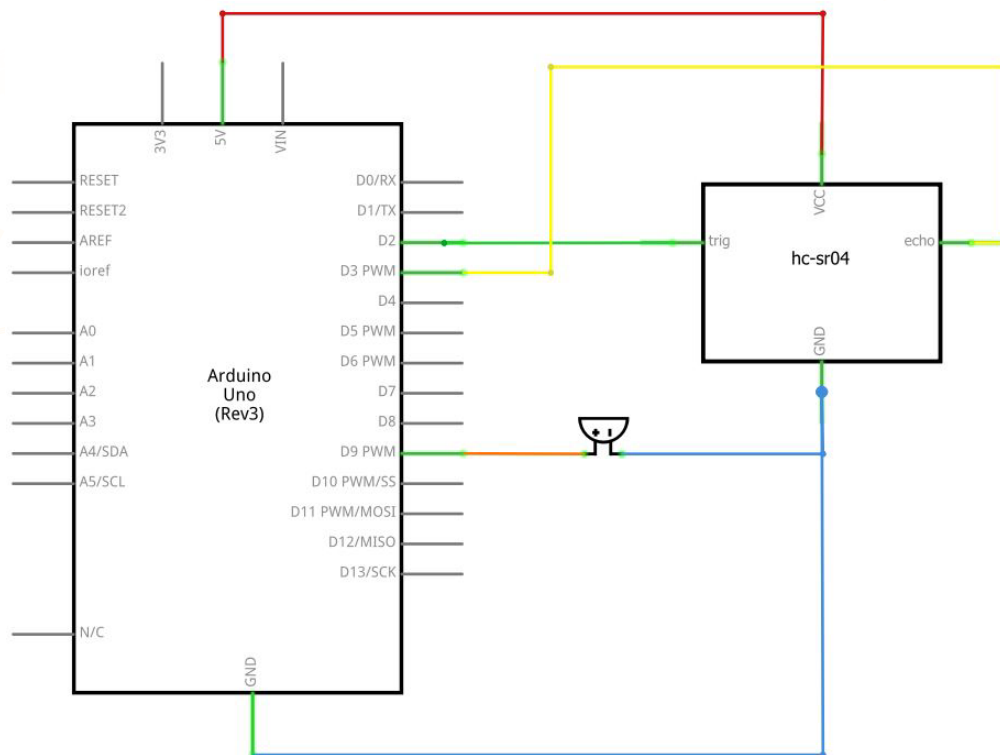
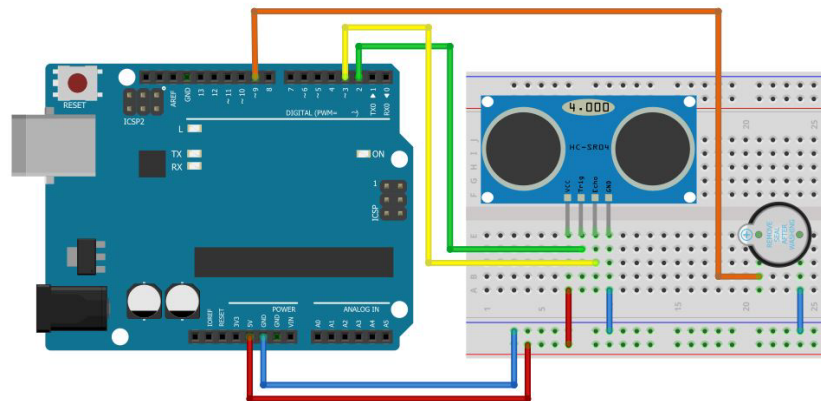
Módulo de Sensor Ultrasónico HC-SR04

Para el cálculo de la distancia de envío y recepción de señal, es:

$$\text{Distancia (cm)} = \mu s / 58$$

Diagrama de Conexión

A continuación, se muestra cuál será la conexión del módulo HC-SR04 en la tarjeta UNO R3 y el buzzer activo, para que, a cierta distancia, se active una alarma de detección de presencia.



Lección 11

Módulo de Sensor Ultrasónico HC-SR04

Código de Funcionamiento

Realizaremos un código para detectar objetos a 50 cm, cuando los objetos y/o personas se encuentren en un rango menor a esta distancia se activará el buzzer, emitiendo un pitido. Para este programa no se requiere alguna librería en especial.

```

int trig = 2;           // pin 2 a Trigger del sensor
int echo = 3;          // pin 3 a echo del sensor
int buzz = 9;          // pin 9 al buzzer
void setup() {
  Serial.begin(9600);   //Se inicia el puerto Serial
                       // Configuración de pines de salida

  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  pinMode(buzz, OUTPUT);
}
void loop() {
  //Declaramos dos variables auxiliares para el cálculo de la distancia respecto al tiempo
  long tiempo;
  long distancia;

  digitalWrite(trig, LOW); //El pin Trigger apagado empieza en el ciclo apagado
  delayMicroseconds(10);   //Pasan 10 microsegundos después...
  digitalWrite(trig, HIGH); //El pin Trigger se enciende
  delayMicroseconds(10);   //Pasan 10 microsegundos después...
  digitalWrite(trig, LOW); //El pin Trigger se apagará
  tiempo = pulseIn(echo, HIGH); //La variable tiempo guardará el valor que tarda en llegar
                                //al receptor de eco, ancho del pulso
  distancia = tiempo/58;     //Hacemos la relación de la distancia en cm
                             //Programación del buzzer
  if (distancia <= 50) {    //Si la distancia es menor a 50 cm el buzzer se encenderá
    digitalWrite(buzz, HIGH);
    delay(500);
  } else {
    digitalWrite(buzz, LOW);
  }

  //Mostrará la distancia en el monitor serial (9600)
  Serial.print(distancia);
  Serial.println("cm");
  delay(50);                //se actualizará la información cada 50 ms
}

```

Lección 12

Módulo de Joystick Analógico

Introducción

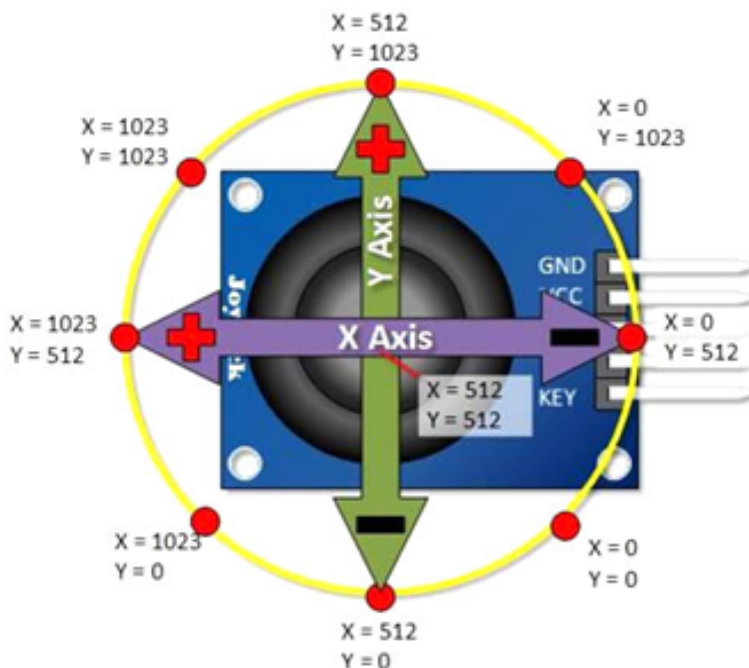
En esta lección aprenderemos a usar el módulo Joystick Analógico que con ayuda de leds y un buzzer activo, comprenderemos el funcionamiento para poderlo incluir en proyectos más robustos.

Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 4 Led de diferentes colores 5mm
- 4 Resistencia de 220 Ohms x 4
- 5 Cables Macho – Macho
- 6 Módulo Joystick
- 7 Buzzer Activo

Conocimientos previos

El módulo Joystick cuenta con un posicionamiento de eje dado por valores entre 0 a 1023 y conforme se mueve, varía su valor dando las coordenadas de cada eje.



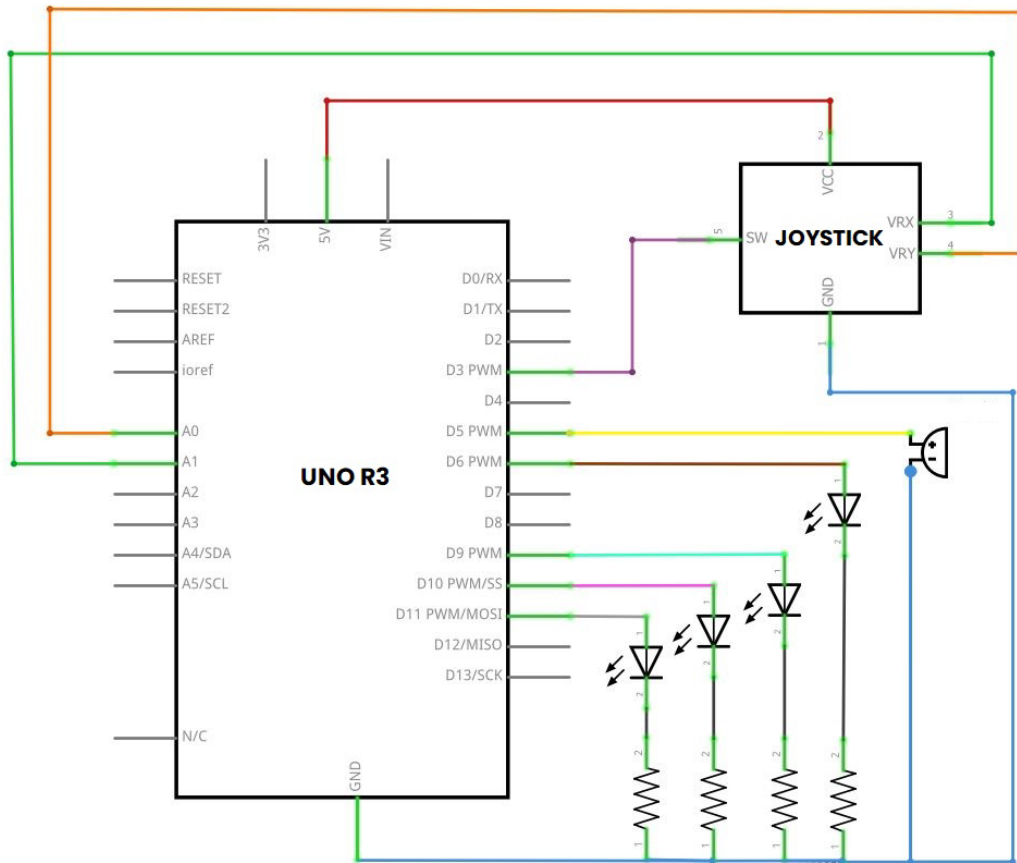
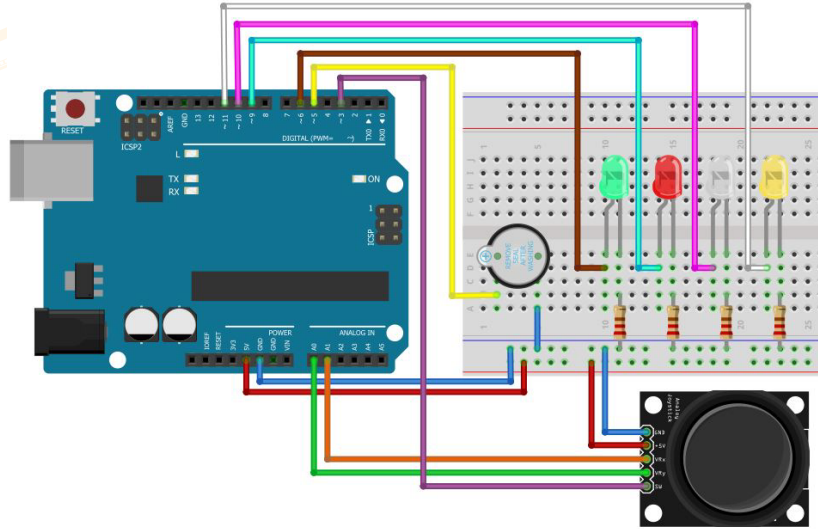
Entonces los valores de las posiciones a tomar en cuenta para realizar el código de programación serán:

- Reposo (central): $X = 512$, $Y = 512$
- Arriba: $X = 1023$, $Y = 512$
- Abajo: $X = 0$, $Y = 512$
- Derecha: $X = 512$, $Y = 1023$
- Izquierda: $X = 512$, $Y = 0$

Lección 12 Módulo de Joystick Analógico

Diagrama de Conexión

La conexión entre los componentes y la tarjeta de desarrollo UNO R3 es el siguiente:



Lección 12 Módulo de Joystick Analógico

Código de Funcionamiento

Para cada dirección que tome el mando del joystick, encenderá un led correspondiente. En caso de presionar el Joystick el buzzer activo emitirá un sonido.

```
int SW_pin = 3;           // Pin 3 a SW(pushbutton) del joystick
int X_pin = A0;          // Pin Analogo A0 al pin VRX del joystick
int Y_pin = A1;          // Pin Análogo A 1 al pin VRY del joystick
int buzz = 5;           // PinPWM 5 al Buzzer +
int ledG = 6;           // PinPWM 6 al Led verde +
int ledR = 9;           // PinPWM 9 al Led rojo +
int ledW = 10;          // PinPWM 10 al Led ROJO+
int ledY = 11;          // PinPWM 11 al Led amarillo +

void setup() {
  //Declaración de variables, como de salida digital
  pinMode(SW_pin, INPUT);
  pinMode(buzz, OUTPUT);
  pinMode(ledG, OUTPUT);
  pinMode(ledR, OUTPUT);
  pinMode(ledW, OUTPUT);
  pinMode(ledY, OUTPUT);

  digitalWrite(SW_pin, HIGH); //lectura de botón del joystick por default estado:1=no
  Serial.begin(9600);         //presionado,0=presionado
                              //Velocidad de Puerto Serial,útil para ver la lectura de valores en
                              //el Monitor Serial
}

void loop() {
  //impresión de valores del joystick dependiendo el eje que sea manipulado
  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));
}
```

Lección 12 Módulo de Joystick Analógico

```

Serial.print("\n\n");
delay(50);
//Dependiendo los valores que el joystick, habrá condiciones para controlar los leds y buzzer
if (digitalRead(SW_pin) == LOW) { //si es presionado el botón del joystick dará un valor 0
  digitalWrite(buzz, HIGH); //por lo tanto el buzz emitirá un pitido
} else { digitalWrite(buzz, LOW); //de lo contrario estará en silencio
}
if (analogRead(X_pin) == 0) { //inicialmente el eje x está en un valor de 523(aprox), si se
  //va a la extrema derecha..
  digitalWrite(ledG, HIGH); //dará un valor 0 encendiendo el led verde
} else { digitalWrite(ledG, LOW);
  } if (analogRead(X_pin) >= 600) { //cuando sea mayor a 600 , indica que se está moviendo
  //a la izquierda..
  digitalWrite(ledR, HIGH); //el led rojo encenderá
} else {
  digitalWrite(ledR, LOW);
}
if (analogRead(Y_pin) == 0) { //inicialmente el eje y está en un valor de 523(aprox), si se
  //mueve hacia arriba..
  digitalWrite(ledW, HIGH); //enciende el led ROJO
} else {
  digitalWrite(ledW, LOW);
}
if (analogRead(Y_pin) >= 600) { //cuando sea mayor a 600 , indica que se está moviendo
  //hacia arriba...
  digitalWrite(ledY, HIGH); //se encendera el led amarillo
} else {
  digitalWrite(ledY, LOW);
}
}
}

```

Lección 13

Sensor de Sonido y Relevador 5V

Introducción

Aprenderemos a usar el Sensor de Sonido en conjunto con el Relevador de 5V, para poder obtener una señal activada por sonido y poder encender un led.

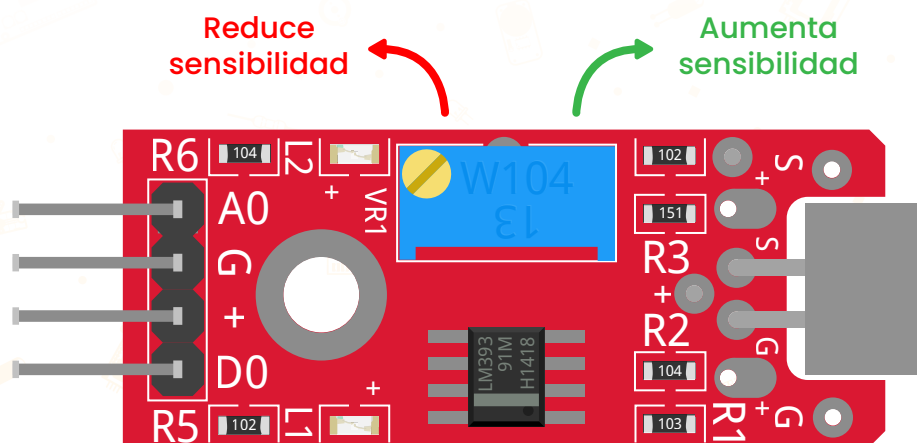
Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Led de 5mm x 1
- 4 Módulo Sensor de Sonido
- 5 Resistencia de 220 Ohms x 1
- 6 Cables Macho – Macho y Hembra-Macho
- 7 Módulo Relevador 5V

Conocimientos previos

Sensor de Sonido

El sensor de sonido, también conocido como KY-037 nos ayuda en la detección de sonidos por medio del micrófono. Este módulo tiene la ventaja de poder ajustar la sensibilidad por medio del potenciómetro además de poder trabajar con señal digital o analógica.



Además, el dispositivo cuenta con dos LEDs:

L1: Confirma que el dispositivo está alimentado (3 a 5 V).

L2: Detección de sonido, si enciende; entonces se ha detectado el sonido en el rango

Lección 13 Sensor de Sonido y Relevador 5V

Relevador 5V

Adicional el relevador de 5V es un módulo de un canal que ayuda a la conmutación de estados, pudiendo controlar cargas en DC y AC.

El módulo provee la electrónica necesaria para manejar la bobina del relevador y la señal de control puede provenir de cualquier circuito de control (tarjeta de desarrollo, CMOS). Para esta lección lo usaremos para una carga en DC (led de 5mm).



El dispositivo cuenta con los siguientes pines:

- IN: Pin de entrada para señal de control
- GND: Pin negativo fuente de alimentación
- VCC: Pin positivo para fuente de alimentación
- 常开 (NO o NA): Contacto normalmente abierto
- 公共端 (COM): Contacto común del relé
- 常闭 NC: Contacto normalmente cerrado
- Led indicador de encendido (Verde)
- Led indicador de estado del led (Rojo)

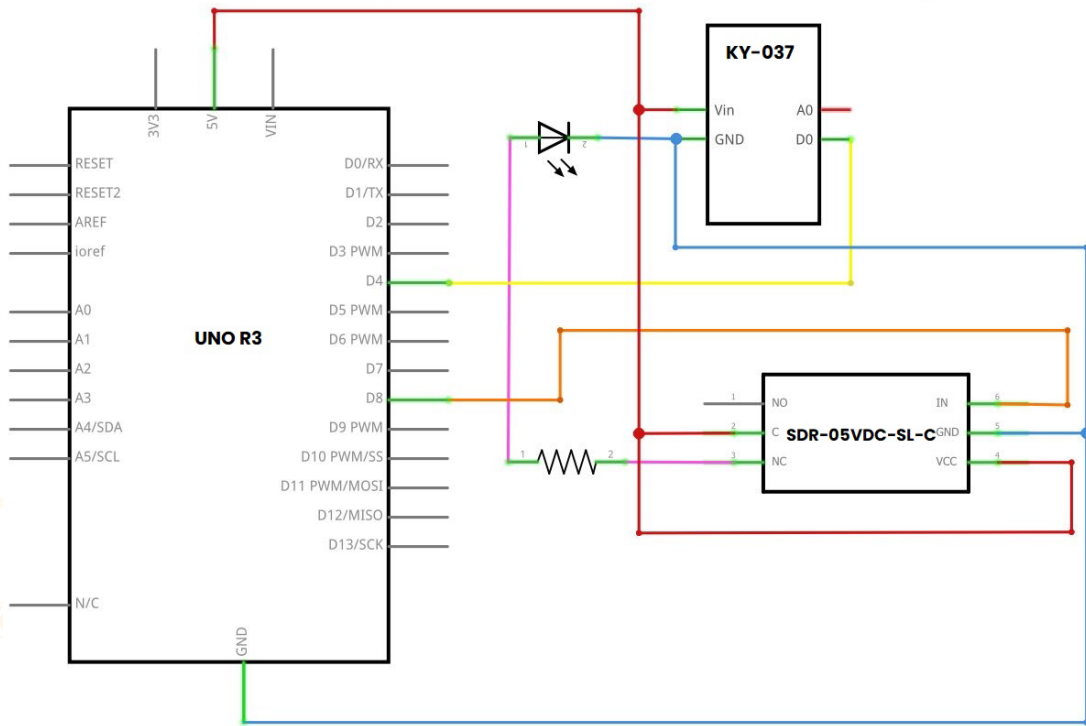
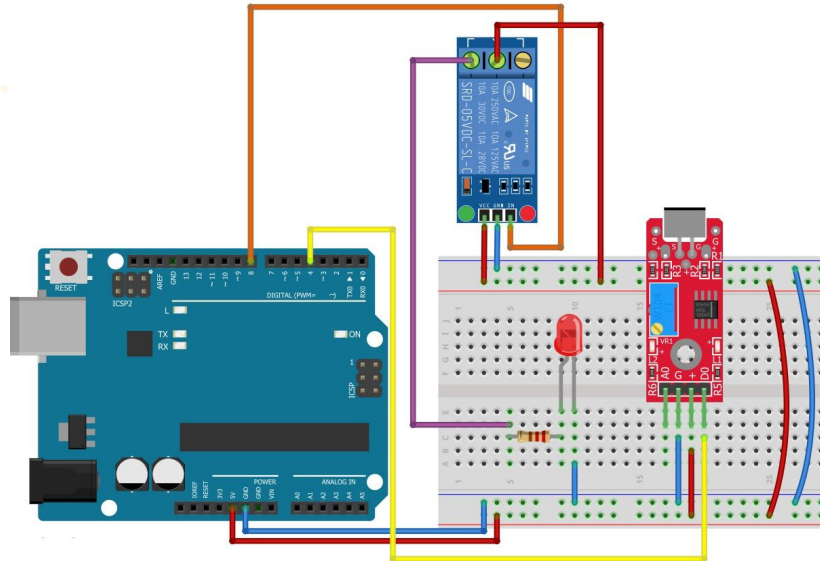
La activación de este dispositivo depende que salida es la que se ocupe: 常开 (NO o NA), contacto normalmente abierto o 常闭 NC, contacto normalmente cerrado.

Será un disparador de bajo nivel cuando el puente se conecta al pin NA-Normalmente Abierto y será un disparador de alto nivel cuando el puente se conecta al pin NC-Normalmente Cerrado.

Lección 13 Sensor de Sonido y Relevador 5V

Diagrama de Conexión

La conexión que se utilizará en el relé de 5V será de Normalmente Cerrado, este nos dará un valor BAJO, sin tener ninguna activación y el relevador cambiará cuando detecte una señal en ALTO.



Lección 13 Sensor de Sonido y Relevador 5V

Código de Funcionamiento

La detección del sonido se da por flancos de subida en la señal digital de salida del módulo de sonido. El siguiente programa tiene como finalidad detectar un sonido para poder activar la conmutación del relevador.

Si los flancos de subida son igual a 4 activará al relevador para poder hacer la conmutación de nivel BAJO a ALTO y poder mandar una señal al led y encenderlo, cuando la detección de sonido sea igual a 8 daremos la orden para apagar el led.

```
int AUDIO= 4; //Pin 4 para la salida digital del KY
int flanco = 0; //Variable de apoyo para saber la lectura del sensor
int rele = 8; //Salida al relé en el Pin 8 del Arduino que activará a su vez el foco
void setup()
{
  Serial.begin(9600); //Inicialización del puerto serial a 9600 baudios
  pinMode( AUDIO, INPUT); //pin del KY como entrada de datos
  pinMode( rele, OUTPUT); //pin 13/reley como pin de salida
}
void loop()
{ if (digitalRead(AUDIO) == HIGH) //Si se detecta un sonido el KY(dependiente a la sensibilidad //determinada por el
    usuario)
  {
    flanco = flanco + 1; // Se realiza un contador +1 por cada vez que el sensor detecte //un flanco ALTO
    Serial.print("Número de Flancos: "); // Escribe el número de flancos detectados dependiendo //el sonido censado...
    Serial.println(flanco, DEC); // escribiendo el valor en forma decimal
    if (flanco == 4) { //Si el contador llega a 4, que es el valor anteriormente calibrado //a nuestro sistema
      Serial.println(flanco); //Escribe el valor del flanco=4 y...
      digitalWrite(rele, HIGH); //...se mantendrá encendido el led hasta que...
    } else if (flanco == 8) { //El contador esperara un nuevo sonido para desactivar el led y //eso es cuando se
      llega al valor de flanco=8
      Serial.println(flanco); //Escribe el valor del flanco=8 y...
      digitalWrite(rele, LOW); //...se apagará el led y ..
      flanco = 0; //Se reiniciará el valor del flanco para que no se quede ningún //valor guardado en flanco
    }
  }
}
```

Lección 14

Sensor de Lluvia Pantalla LCD I2C

Introducción

En la siguiente lección trabajaremos con el sensor de lluvia, también conocido como sensor de nivel de líquido, el cual nos ayudará para detectar si un área en específico se encuentra húmeda o que porcentaje de líquido contiene.

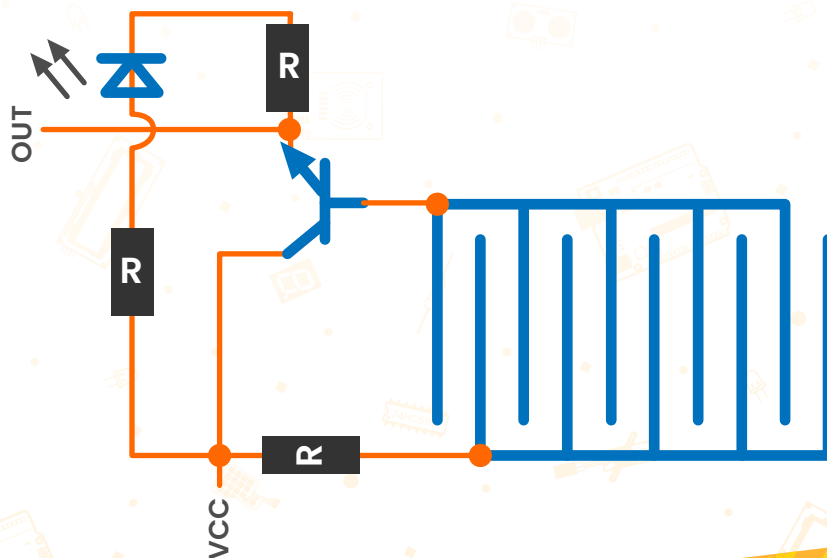
Para poder visualizar esta medición nos apoyaremos de la Pantalla LCD que ya cuenta con comunicación I2C, la cual será conveniente para reducir las conexiones entre la tarjeta de desarrollo y este módulo.

Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Cables Macho – Macho y Hembra-Macho
- 4 Sensor de Lluvia
- 5 Pantalla LCD 16X2 I2C

Conocimientos previos

El sensor de Nivel de Lluvia es un dispositivo analógico, para poder obtener los valores se trabaja con una tarjeta de desarrollo, que por medio de la entrada de pines analógicos podremos obtener valores entre 0 a 1023, ya que la resolución de este módulo es de 10 bit. En su circuito incluye un arreglo de resistencias, una en especial de $1M\Omega$; que aumenta en caso de que el agua entre en contacto con sus bandas conductoras. Cuanto mayor sea la cantidad de agua, mayor resistencia se genera.



Lección 14 Sensor de Lluvia Pantalla LCD I2C

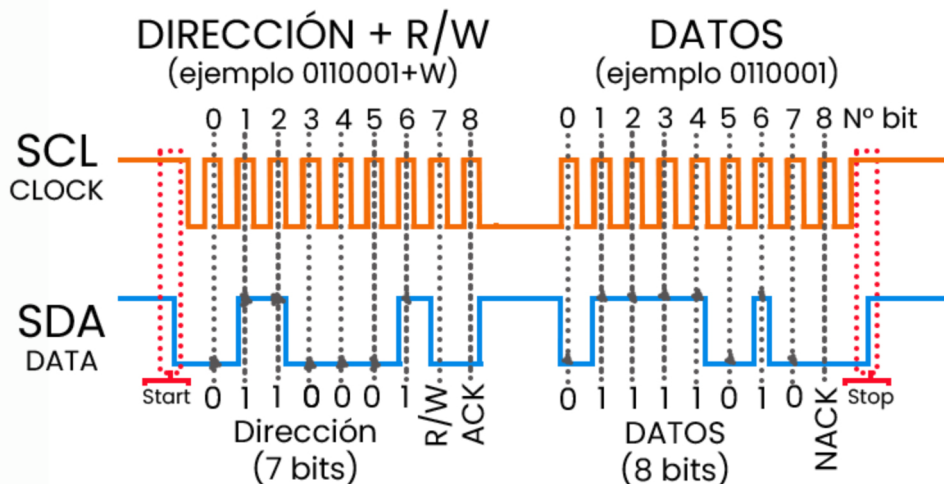
Es útil para automatizar cultivos caseros apalancados de un sistema de riego, para detección de fugas o filtraciones.

Por otro lado, el display 16x2 LCD incluye un módulo de comunicación I2C. Siendo un protocolo de comunicación serial, este tipo de arquitectura permite tener una confirmación de datos recibidos.

Comunicación I2C

Para lograr este tipo de comunicación se requiere de un Maestro y un Esclavo, además de un bus que incluye dos líneas llamadas SDA (Serial Data) y SCL (Serial Clock), estas líneas son bidireccionales conectadas por resistencias físicas pull-up. Si ambas líneas están libres se encontrarán en un nivel alto.

El Maestro se encarga de controlar a SCL, además de iniciar y parar la comunicación. Mientras tanto el envío de información serial enviada por SDA. El Esclavo normalmente es el sensor y dará al Maestro información en paquetes de 8 bits y enviar confirmaciones de recepción, llamadas ACK.



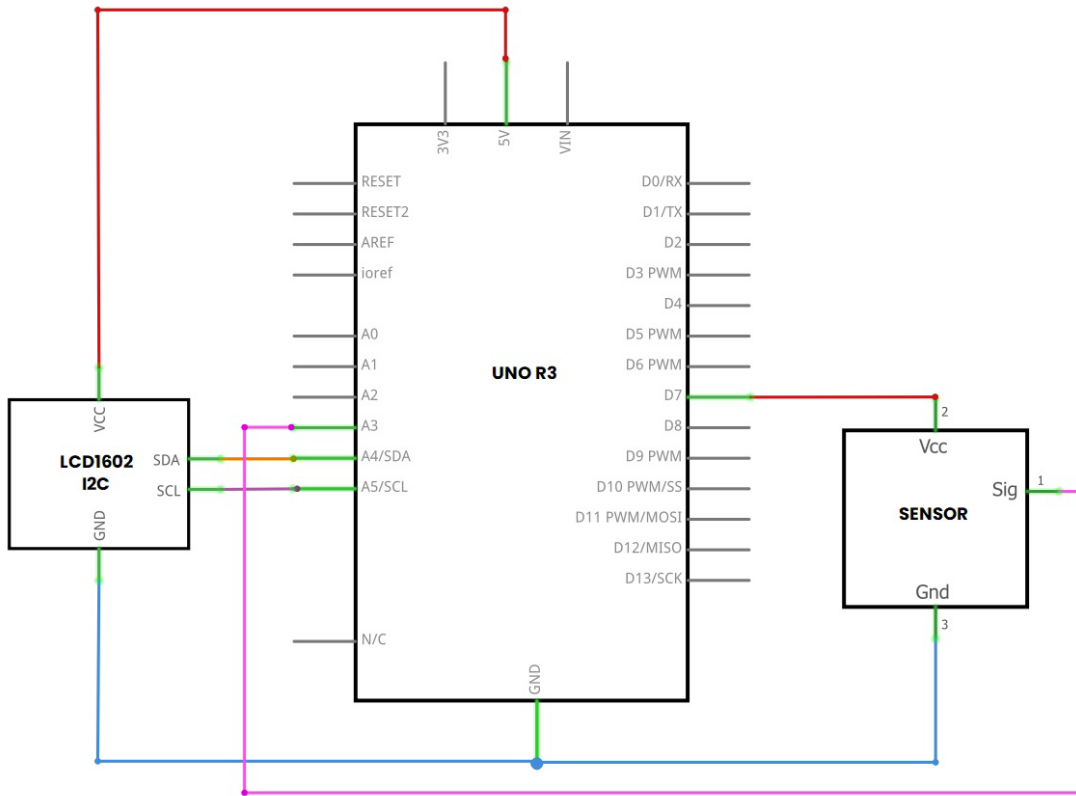
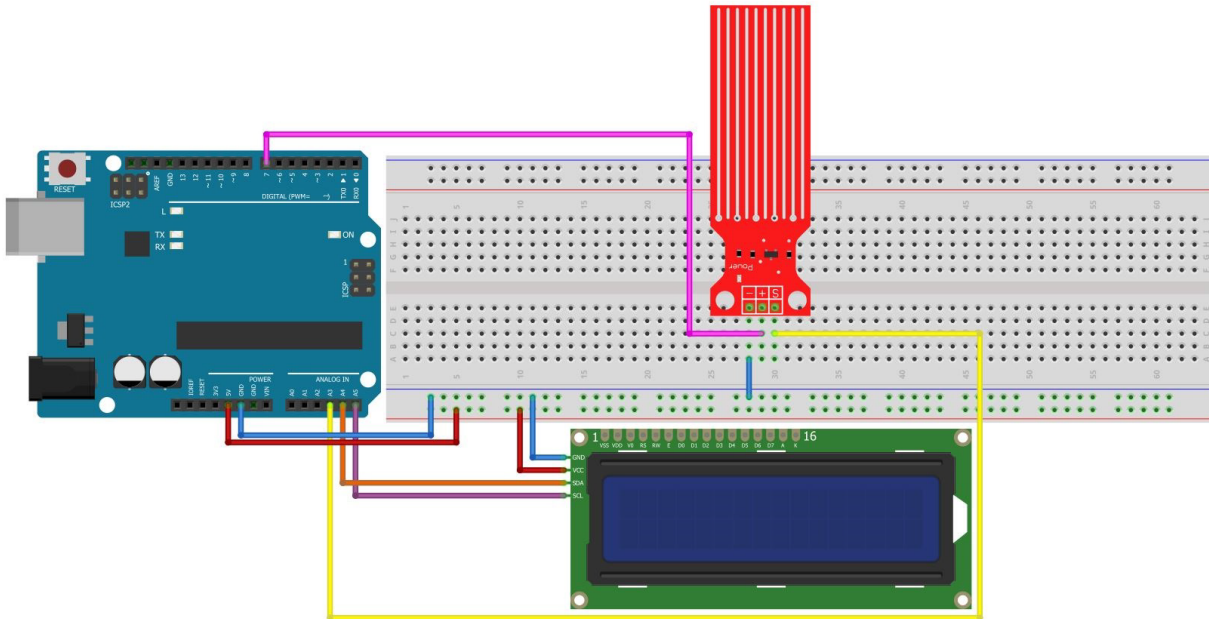
Como se puede visualizar hay toda una trama de bit que son necesarios para enviar y recibir información, entre ellos están:

- Star, Inicio
- Stop, Parada
- ACK, Confirmación
- NACK, No Confirmación
- R/W, Read/Write, Lectura/Escritura
- 7 bits para la dirección del dispositivo Esclavo/Maestro
- 8 bits de dirección y datos

Pareciera complejo, pero este protocolo nos facilita la conexión y transferencia de datos entre la tarjeta de desarrollo UNO R3 y la pantalla LCD 16X2.

Lección 14 Sensor de Lluvia Pantalla LCD I2C

Diagrama de Conexión



Lección 14 Sensor de Lluvia Pantalla LCD I2C

Código de Funcionamiento

Para trabajar con el siguiente código, será necesario que en el IDE de Arduino se descargue la librería para poder trabajar con la pantalla LCD , la cual se encuentra en el siguiente link:

https://github.com/ELECTROALL/Codigos-arduino/blob/master/LiquidCrystal_I2C.zip

El sensor detectará la humedad, en el área o superficie correspondiente. La entrada de datos será por pin analógico obteniendo de esto, valores entre 0-1023, los cuales tendremos que manipular para registrar un porcentaje de humedad, que posteriormente se visualizarán en la pantalla LCD.

```
// Librería de comunicación Wire y para unos del display LCD
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
//Definición de pines para uso del sensor de lluvia
#define pin_on 7 //pin 7 a pin del Sensor (+)
#define pin_signal A3 //pin analógico A3 a pin del Sensor(S)
float valor = 0; //Variable con valor 0
LiquidCrystal_I2C lcd(0x27, 16, 2); //Crear el objeto lcd dirección 0x27 y 16 columnas x 2 filas
void setup() {
  pinMode(pin_on, OUTPUT); // configuración del pin 7 como salida
  digitalWrite(pin_on, LOW); // El estado del sensor comienza en bajo -apagado
  lcd.init(); // Inicializar el LCD
  lcd.backlight(); //Encender la luz de fondo.
  lcd.print("Display I2C"); // Escribimos el Mensaje en el LCD
  delay(1000);
  lcd.clear(); //limpiamos la pantalla para agregar nuevo mensaje
}
void loop() {
  digitalWrite(pin_on, HIGH); // El estado del sensor es activado-ON
  delay(10); // espera 10 milisegundos
  valor = analogRead(pin_signal); // la variable valor ahora tiene el dato registrado por el sensor
  float vneto = ((valor / 570) * 100); //realizamos cálculo para obtener el porcentaje de los valor
  //analógico 0 a 570 aprox
  digitalWrite(pin_on, LOW); // el sensor se apagará y actualiza la inf
  lcd.setCursor(0, 0); //colocamos el cursor en la posición columna, fila (0,0)
  lcd.print("ADC:"); //escribimos mensaje donde daremos a conocer el valor crudo de entrada
  //del pin analógico
  lcd.print(valor); //impresión del valor
  lcd.setCursor(0, 1); //colocamos el cursor en la posición columna, fila (0,1)
  lcd.print("Humedad:"); //escribimos mensaje
  lcd.print(vneto); //impresión del valor en porcentaje
  lcd.print("%"); //escribimos mensaje
  delay(1000); // esperamos 1 min para nueva medición
}
```

Lección 15

Sensor de Temperatura LM35 Pantalla LCD I2C

Introducción

En esta sección utilizaremos el Sensor de Temperatura LM35, el cual es un circuito integrado (CI) de precisión de temperatura y observaremos los datos obtenidos nuevamente por medio de la pantalla LCD 16X2.

Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Cables Macho – Macho y Hembra-Macho
- 4 Sensor de temperatura LM35
- 5 Pantalla LCD 16X2 I2C

Conocimientos previos

El LM35 entrega la lectura de datos por medio de una salida analógica directamente en grados centígrados, pero por cada grado centígrado medido, entregará en su salida un valor de voltaje de 10 mV.

Por ejemplo: una lectura de 150°C equivale a 1500mV y -55°C equivale a -550mV.
El pinout del IC es el siguiente:



Lección 15

Sensor de Temperatura LM35 Pantalla LCD I2C

Código de funcionamiento

Para trabajar con el siguiente código, será necesario que en el IDE de Arduino se descargue la librería para poder trabajar con la pantalla LCD , la cual se encuentra en el siguiente link:

https://github.com/ELECTROALL/Codigos-arduino/blob/master/LiquidCrystal_I2C.zip

Sabiendo que los valores que obtendremos del sensor en la temperatura en orden de mV, tendremos que preparar los datos para tener valores en grados Celsius. Y posteriormente se mostrará esta información en la pantalla LCD 16X2.

```
// Librería de comunicación Wire y para unos del display LCD
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
const int LM35 = A0; //Pin de entrada Analógico de datos del LM35
//Crear el objeto lcd dirección 0x27 y 16 columnas x 2 filas
LiquidCrystal_I2C lcd(0x27, 16, 2); //
void setup() {
  lcd.init(); // Inicializar el LCD
  lcd.backlight(); //Encender la luz de fondo.
  lcd.print("LM35 Y DISPLAY "); // Escribimos el Mensaje en el LCD.
  delay(1000);
  lcd.clear(); //limpiamos la pantalla para agregar nuevo mensaje
}
void loop() {
  // la variable temperatura guardará el valor ADC del sensor
  int temperatura = analogRead(LM35);
  //cálculo para convertir el valor ADC a mV, 10mV/°C
  float millivolts = (temperatura / 1023.0) * 5000;
  float celsius = millivolts / 10; //Conversión de voltaje a grados celsius
  lcd.setCursor(0, 0); //colocamos el cursor en la posición columna, fila (0,0)
  lcd.print("ADC:"); //escribimos mensaje donde daremos a conocer el valor crudo de
  //entrada del pin //analógico
  lcd.print(temperatura); //impresión del valor del sensor sin conversión
  lcd.setCursor(0, 1); //colocamos el cursor en la posición columna, fila (0,1)
  lcd.print("Celsius: "); //escribimos mensaje
  lcd.print(celsius); //impresión del valor de temperatura en celsius
  delay(1000); // esperamos 1 min para nueva medición
}
```

Lección 16

DHT11 Sensor de Temperatura y Humedad

Introducción

Para la realización de esta práctica usaremos otro sensor popular debido a que se puede realizar 2 mediciones en uno: temperatura y humedad; este dispositivo es el DHT11, también conocido como KY-015 y al igual que los otros dos sensores veremos los datos arrojados en la pantalla de 16x2 segmentos LCD con I2C.

Materiales necesarios

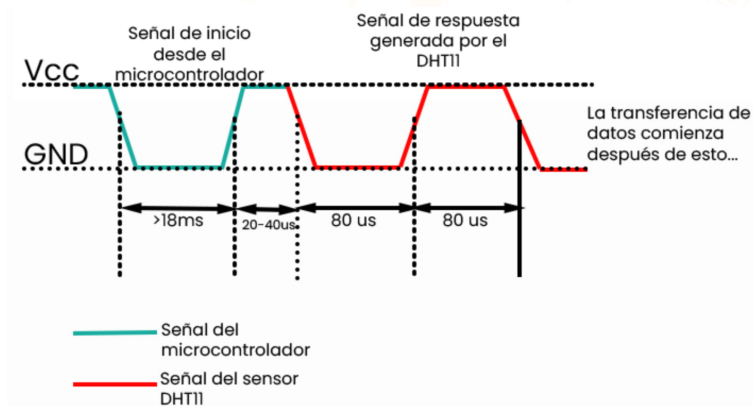
- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Cables Macho – Macho y Hembra-Macho
- 4 Sensor de temperatura y humedad DHT11
- 5 Pantalla LCD 16X2 I2C

Conocimientos previos

El DHT11 no utiliza una interfaz serial estándar como I2C, SPI o 1Wire. En cambio, requiere su propio protocolo para comunicarse a través de un solo hilo: Single bus.

El microcontrolador debe iniciar la comunicación con el DHT11 manteniendo la línea de datos en estado bajo durante al menos 18 ms. Luego el DHT11 envía una respuesta con un pulso a nivel bajo de 80 μ s y luego deja "flotar" la línea de datos por otros 80 μ s.

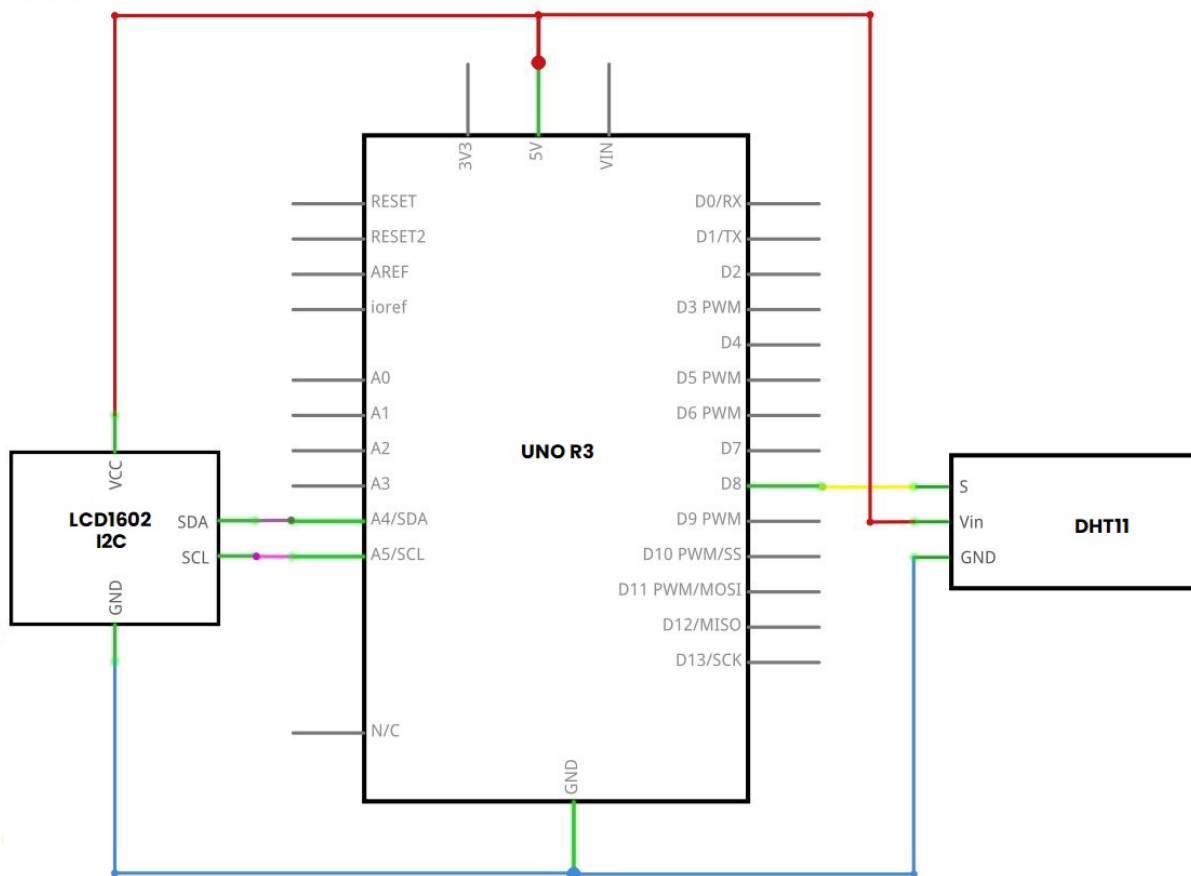
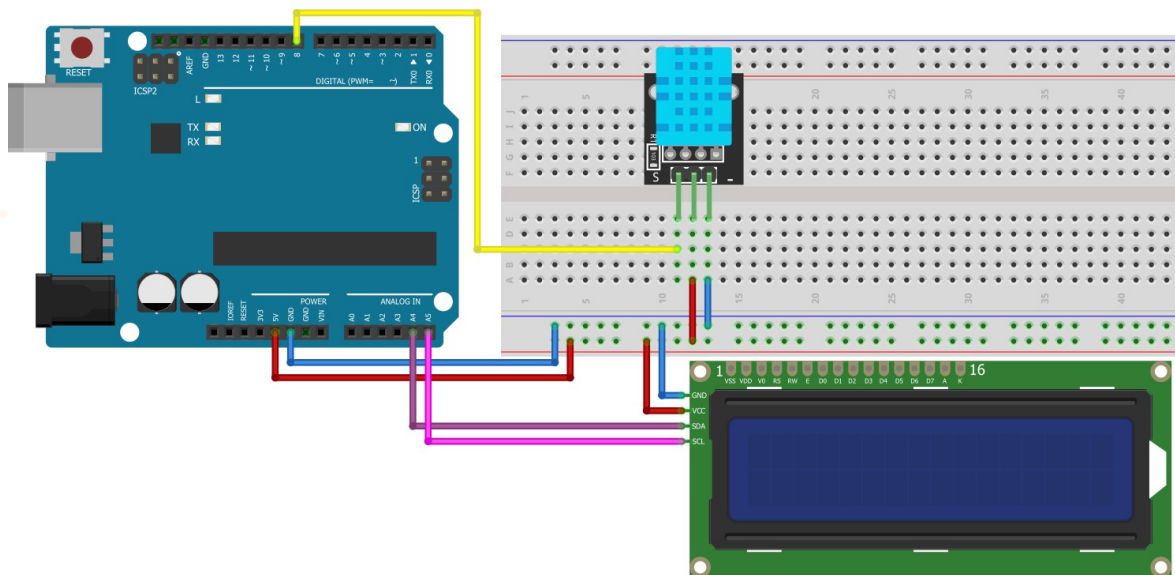
En la siguiente figura se puede visualizar, el pulso de inicio enviado por el microcontrolador está coloreado en azul, mientras que la respuesta desde el sensor está coloreada en rojo.



Los datos binarios se codifican según la longitud del pulso alto. Todos los bits comienzan con un pulso bajo de 50 μ s. En nuestra librería aprovechamos el pulso bajo en cada bit para sincronizar con la señal del DHT11. Luego viene un pulso alto que varía según el estado lógico o el valor del bit que el DHT11 desea transmitir. Se utilizan pulsos de 26-28 microsegundos para un "0" y pulsos de 70 microsegundos para un "1". Los pulsos se repiten hasta un total de 40 bits (5 Bytes).

Lección 16 DHT11 Sensor de Temperatura y Humedad

Diagrama de Conexión



Lección 16 DHT11 Sensor de Temperatura y Humedad

Código de Funcionamiento

Para trabajar con el siguiente código, será necesario que en el IDE de Arduino se descargue la librería para poder trabajar con la pantalla LCD, la cual se encuentra en el siguiente link:

https://github.com/ELECTROALL/Codigos-arduino/blob/master/LiquidCrystal_I2C.zip

Adicional, se usará la librería del sensor DHT11

DHT sensor library
by Adafruit Versión 1.4.3 **INSTALLED**
Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors
[More info](#)

A diferencia de los dos dispositivos anteriores, en donde se tenía que realizar un cálculo previo para poder desplegar los valores en la pantalla LCD, aquí el sensor DHT11 nos arroja los valores ya procesados en grados celsius y un rango de humedad.

```
// Librería del sensor DHT y para el display LCD
#include <DHT.h>
#include <LiquidCrystal_I2C.h>
#define DHTTYPE 11 //Definimos el modelo DHT11
#define DHTPIN 8 //Se define el pin para conectar el sensor
float h, t; //variables para guardar el dato de humedad y temperatura del sensor
DHT dht(DHTPIN, DHTTYPE);
LiquidCrystal_I2C lcd(0x27, 16, 2); //Crear el objeto lcd dirección 0x27 y 16 columnas x 2 filas

void setup() {
  dht.begin(); //Se inicia la dht, con la definición de modelo y pin
  lcd.init(); // Inicializar el LCD
  lcd.backlight(); //Encender la luz de fondo.
  lcd.print("DHT11 Y DISPLAY"); // Escribimos el Mensaje en el LCD.
  delay(1000); //limpiamos la pantalla para agregar nuevo mensaje
  lcd.clear();
}

void loop() {
  h = dht.readHumidity(); //se guarda en la variable h el valor de humedad
  t = dht.readTemperature(); //se guarda en la variable t el valor de temperatura
  lcd.setCursor(0, 0); //colocamos el cursor en la posición columna, fila (0,0)
  lcd.print("Humedad:"); //se visualiza el texto
  lcd.print(h); //se visualiza el dato de humedad

  lcd.setCursor(0, 1); //colocamos el cursor en la posición columna, fila (0,1)
  lcd.print("Temp:"); //se visualiza el texto
  lcd.print(t); //impresión del valor de temperatura
  lcd.print("C"); //escribimos mensaje
  delay(1000); // esperamos 1 min para nueva medición
}
```

Lección 17

Módulo de Interruptor de Membrana y Led RGB

Introducción

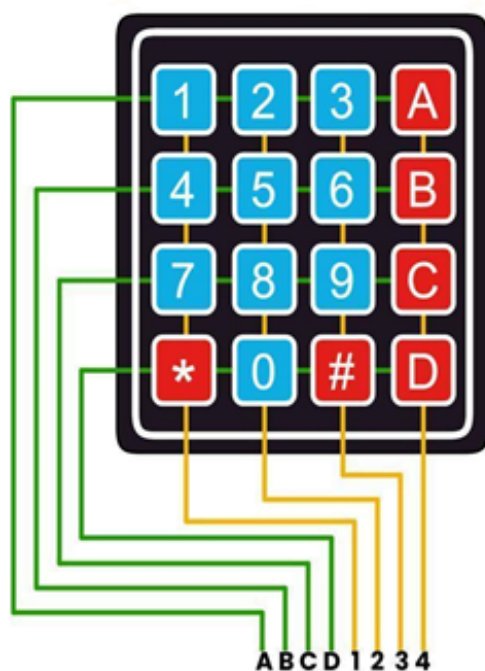
Realizaremos un sistema de control de accesos con módulos anteriormente utilizados; agregando un teclado de membrana para que el usuario pueda introducir una clave de acceso. Visualizando exitoso o denegado la entrada al usuario por medio de un led RGB.

Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Led RGB
- 4 Cables Macho – Macho
- 5 Teclado de Membrana 4x4

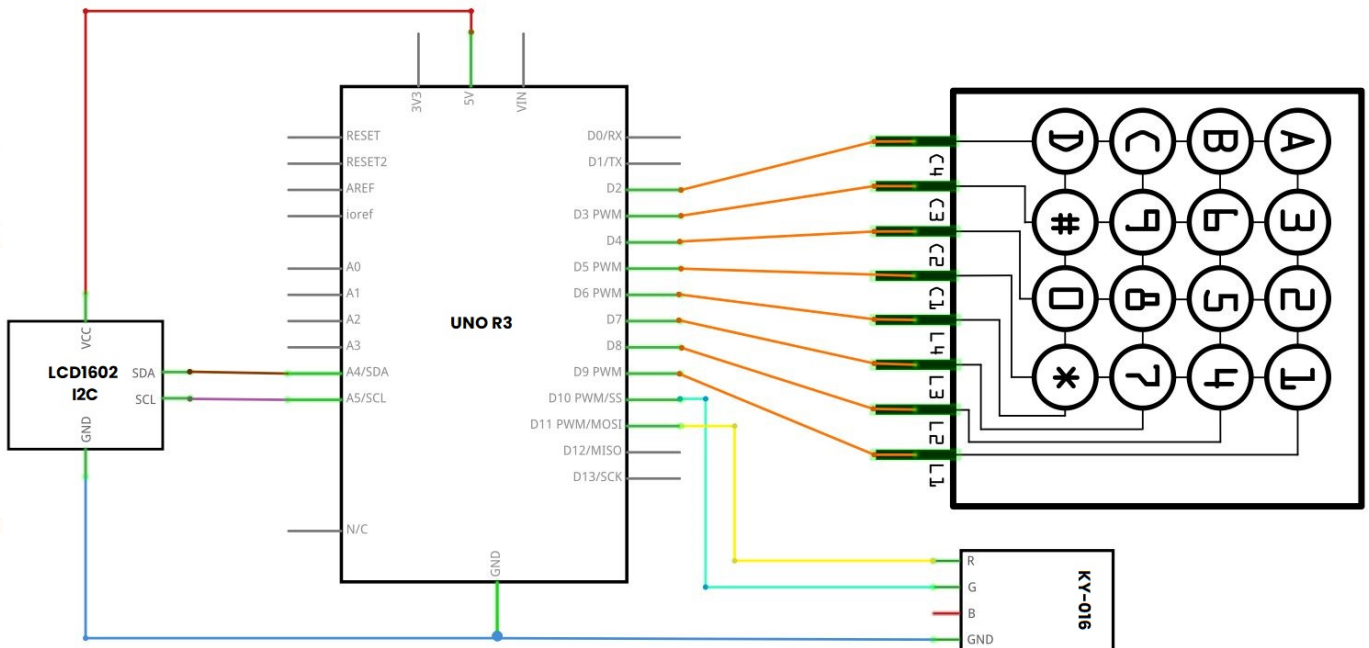
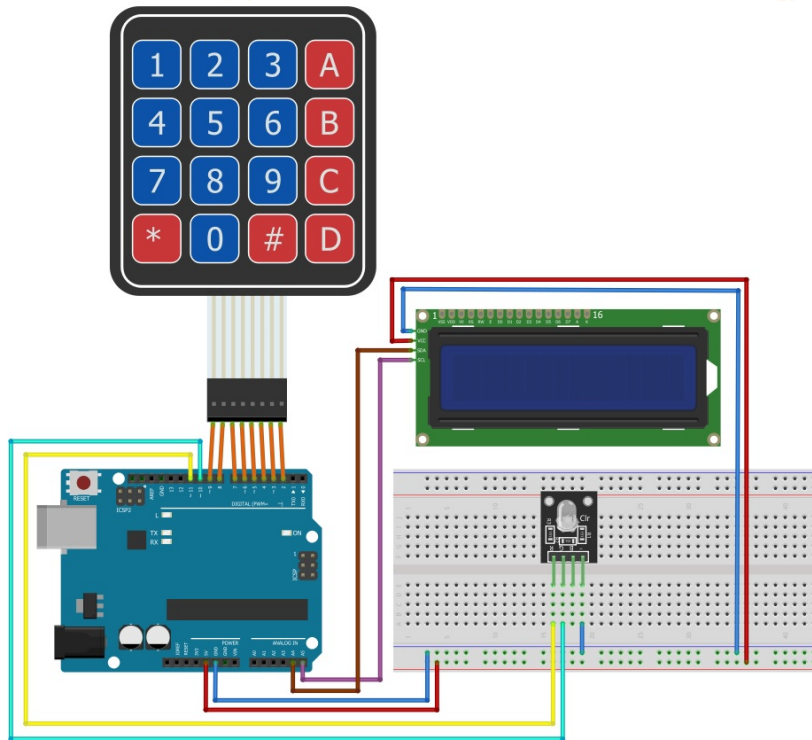
Conocimientos previos

El teclado matricial 4x4 está constituido por filas (A, B, C, D) y columnas (1, 2, 3, 4). Las 16 teclas necesitan sólo 8 pines del microcontrolador, en lugar de los 16 pines que se requerirían para la conexión de 16 teclas independientes.



Lección 17 Módulo de interruptor de membrana y Led RGB

Diagrama de Conexión



Lección 17

Módulo de interruptor de membrana y Led RGB

Código de Funcionamiento

Para trabajar con el siguiente código, será necesario que en el IDE de Arduino se descargue la librería para poder trabajar con el teclado de membrana 4x4, la cual se encuentra en el siguiente link:

<https://github.com/Chris--A/Keypad>

```
//Librerías para usar el teclado y la pantalla LCD
#include <Keypad.h>
#include <LiquidCrystal_I2C.h>

int rojoPin = 11; // El control del color Rojo por el pin 11
int verdePin = 10; //El control del color Verde por el pin 10
LiquidCrystal_I2C lcd(0x27, 16, 2); //Crear el objeto lcd dirección 0x27 y 16 columnas x 2 filas
//Declaración de variables para el control de Filas y Columnas de la matriz del teclado

const byte F = 4;
const byte C = 4;

//Declaración de valor de cada posición de la matriz 4x4
char keys [F][C] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'} };

//Declaración de que conexión se realizó entre la tarjeta UNO R3 y la matriz 4x4
byte pinesFilas[F] {9, 8, 7, 6};
byte pinesColumnas[C] {5, 4, 3, 2};

//teclado tendrá toda la información de pines y valores de la matriz 4X4
Keypad teclado = Keypad (makeKeymap(keys), pinesFilas, pinesColumnas, F, C);
//Variables para
char TECLA;
char CONTROL[5];

//Variable que contara qué número de teclas fue presionado
byte INDICE = 0;
void setup() {
  pinMode(rojoPin, OUTPUT); //Pin de salida 11
  pinMode(verdePin, OUTPUT); //Pin de salida 10
}
```

Lección 17

Módulo de interruptor de membrana y Led RGB

```

lcd.init();
lcd.backlight(); //Encender la luz de fondo.
lcd.print("Abre y Cierra"); //Escribimos el Mensaje en el LCD.
delay(1000);
lcd.clear(); //limpiamos la pantalla para agregar nuevo mensaje
}

void loop() {
  TECLA = teclado.getKey(); //dígitos que se presionen en el teclado, serán guardados en la //variable TECLADO
  if (TECLA) { //Dependiendo la cantidad de Teclas presionadas se realizará un //conteo
    CONTROL[INDICE] = TECLA; //que ayudará a limitar los caracteres requeridos
    INDICE++; //incremento de ese índice dependiendo las teclas presionadas
  }

  lcd.setCursor(0, 0); //colocamos el cursor en la posición columna, fila (0,0)
  lcd.print("Inserta Clave..."); //se visualiza el texto

  if (INDICE == 4) { //Si el índice anteriormente incrementado llega a 4...
    lcd.clear(); //limpia pantalla LCD
    if (!strcmp(CONTROL, "AB43")) { //Si la combinación es AB43
      lcd.clear();
      setColor(0, 255, 0); //El LED RGB Brillará en color Verde
      lcd.setCursor(0, 0);
      lcd.print("ABIERTO");
      delay(5000); //Instrucción que durará 1s (1000ms)
      Serial.println("VERDE");
    } else if (!strcmp(CONTROL, "C134")) { //Si la combinación es C134
      lcd.clear();
      setColor(255, 0, 0); //Brillará en color Rojo
      lcd.setCursor(0, 0);
      lcd.print("CERRADO");
      delay(5000); //Instrucción que durará 1s (1000ms)
      Serial.println("ROJO");
    } else { //Para todas las demás combinaciones de teclado
      lcd.clear();
      setColor(255, 255, 0);
      lcd.setCursor(0, 0);
      lcd.print("Intenta"); //pedirá que de nuevo se ingrese el código
      lcd.setCursor(0, 1);
      lcd.print("Nuevamente");
      delay(1000); //Instrucción que durará 1s (1000ms)
      INDICE = 0; //el indice se reiniciara
      lcd.clear();
    }
  }
}

//funcion para poder cambiar de color el led RGB dependiendo la combinación de dígitos en el //teclado
void setColor(int rojoValue, int verdeValue, int azulValue) {
  analogWrite(rojoPin, rojoValue);
  analogWrite(verdePin, verdeValue);
}

```

Lección 18

Módulo de Tiempo Real DS1302 y Led

Introducción

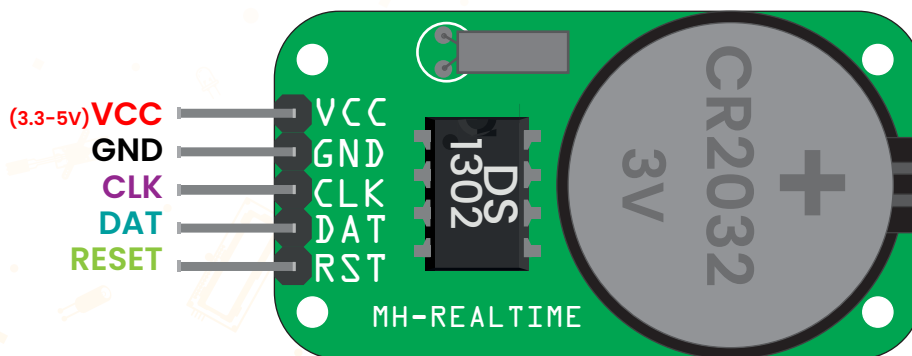
Realizaremos una sencilla alarma. la cual se activará a la hora y fecha deseada, gracias al módulo DS1302 y un led.

Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Led 5mm x 1
- 4 Resistencia de 220 Ohms x 1
- 5 Módulo de Tiempo Real DS1302
- 6 Cables Macho – Macho

Conocimientos previos

El módulo de tiempo real basa su funcionamiento en el circuito integrado DS1302 y con la alimentación de una batería de pastilla (3V), el cual registrará fecha (DD/MM/AAAA) y hora (HH:MM:SS), incluyendo una RAM estática de 31 bytes y trabaja con un cristal de 32.768 kHz. La comunicación que utiliza es por medio de I2C.

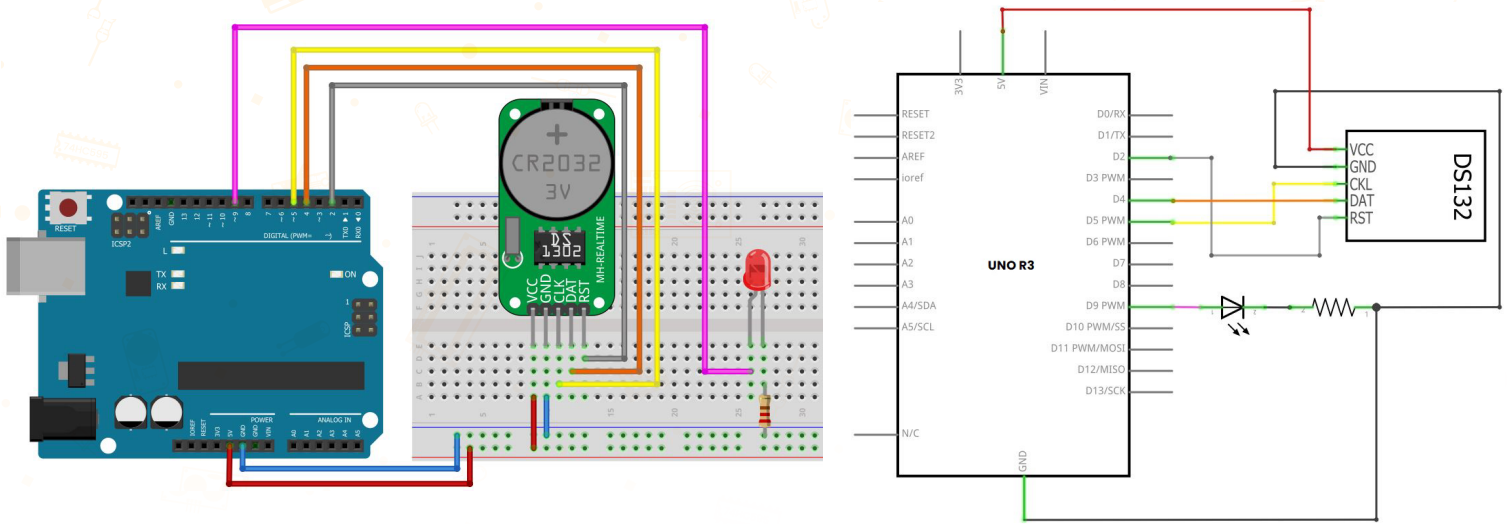


Lección 18

Módulo de tiempo real DS1302 y Led

Diagrama de Conexión

El siguiente diagrama muestra la conexión entre el módulo de tiempo real y un led que nos ayudará, para saber cuándo la alarma programada esté o haya ocurrido.



Código de Funcionamiento

Para realizar el siguiente código será necesario contar con las siguientes librerías en Arduino:

Rtc by Makuna

by Michael C. Miller (makuna@live.com) Versión 2.3.4 **INSTALLED**

A library that makes interfacing DS1302, DS1307, DS3231, and DS3234 Real Time Clock modules easy. Includes deep support of module features, including temperature, alarms and memory storage if present. Tested on esp8266.

[More info](#)

Seleccione versión ▾

Instalar

Actualizar

El siguiente código utilizaremos funciones propias de la librería previamente descargada para encender el led en determinada hora y fecha; y se desactiva también con una fecha predeterminada.

Para fines didácticos sólo se maneja en el código la hora, para que puedas modificarla al momento que realices la práctica prescindiendo del día que realices este programa.

Lección 18

Módulo de tiempo real DS1302 y Led

```

//Librerías para el uso del módulo de tiempo real
#include <ThreeWire.h>
#include <RtcDS1302.h>
//Espacio de matriz para definir datos de hora y fecha
#define countof(a) (sizeof(a) / sizeof(a[0]))
//Pines asociados a la conexión del módulo con la tarjeta UNO R3
ThreeWire myWire(4, 5, 2); // IO, SCLK, CE
RtcDS1302<ThreeWire> Rtc(myWire);
//variable bool para activar o desactivar alarma
bool alarma = true;
//variable de salida para encender el led
int led = 9;
//Definición de variables de salida y comienzo de monitor serial y variable RTC
void setup ()
{
  Serial.begin(57600);
  Rtc.Begin();
  pinMode(led, OUTPUT);
}
void loop ()
{
  //Variable now guardará el valor de la fecha y hora actual
  RtcDateTime now = Rtc.GetDateTime(); infoTiempo(now);
  //La variable ejecutará proceso en la función
  infoTiempo analogWrite(led, LOW); //El led siempre tendrá un valor bajo al iniciar el programa
}
//función infoTiempo que realizará la impresión de datos de hora y fecha
void infoTiempo(const RtcDateTime& dt)
{
  char datestring[20];
  sprintf_P(datestring, countof(datestring), PSTR("%02u/%02u/%04u %02u:%02u:%02u"),
    dt.Month(),
    dt.Day(),
    dt.Year(),
    dt.Hour(),
    dt.Minute(),
    dt.Second());
    //configuración de activación de alarma a las 14hr y 47 min
  if (dt.Hour() == 14 && dt.Minute() == 47) { if (alarma = true) {
    Serial.println("Alarma");
    analogWrite(led, HIGH);
    alarma = false; //desactiva alarma
  }
}
  Serial.println(datestring);
  delay(1000);
  //configuración desactivación de alarma a las 14hr y 49 min
  if (dt.Hour() == 14 && dt.Minute() == 49) analogWrite(led, LOW);
  alarma = true;
}

```


Lección 19

Led Receptor IR con Mando a Distancia y Control de Motor a Pasos

Introducción

En esta lección realizaremos el control de un motor de pasos por medio de la comunicación IR, en donde tendremos un receptor infrarrojo y emitiremos por medio de un control remoto señales para que el motor realice 3 acciones

Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Módulo de control de motor de pasos ULN2003
- 4 Motor de pasos
- 5 Módulo de alimentación para placas de circuitos
- 6 Receptor Infrarrojo VS1838B IR
- 7 Control inalámbrico con batería 3V
- 8 Cables Macho – Macho, Hembra-Hembra

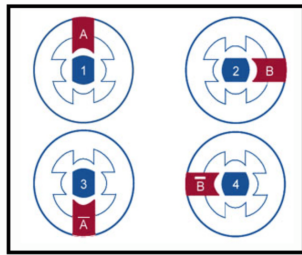
Conocimientos previos

Motor a pasos 28YJ-48 y control ULN2003

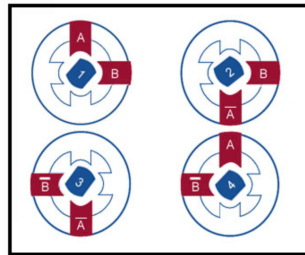
El 28YJ-48 es un motor a pasos engranado que funciona con una tensión de 5V. Son útiles para automatizar persianas, control de movimiento en robots e introducción al conocimiento de motor de pasos. Cuentan con una relación de reducción de 64:1, con aproximadamente 15 rotaciones por minuto (RPM), pudiendo variar este valor mediante programación en una tarjeta de desarrollo compatible con el controlador ULN2003.

El control del motor 28BYJ-48 por medio del ULN2003 es muy fácil de implementar con ayuda del UNO R3, ya que internamente el motor 28BYJ-48 cuenta con 4 bobinas, como se estará realizando un control mecánico a través de un sistema digital, es la justificación para usar el controlador ULN2003. Por medio de programación podremos realizar la modificación del movimiento de las bobinas.

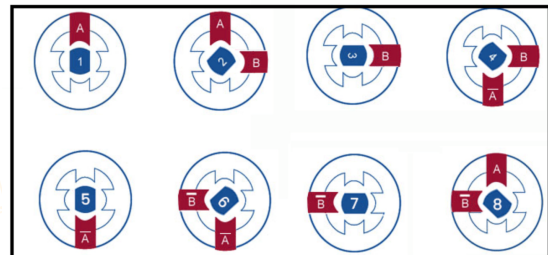
Lección 19 Led Receptor IR con Mando a Distancia y Control de Motor a Pasos



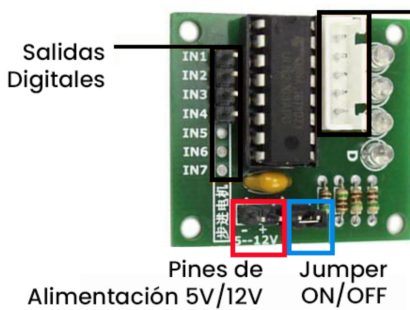
Movimiento Normal



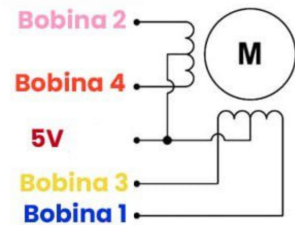
Movimiento por Olas



Medio Paso



Conexión hacia el motor dependiendo la configuración del movimiento



Transmisor y receptor IR

El módulo VS1838B es un dispositivo optoelectrónico teniendo la capacidad de recibir señales vía infrarrojo, convenientemente detecta pulsos a 38kHz, esta frecuencia es utilizada en la mayoría de los controles transmisores de TV, DVD, Sonido, etc. Y detectando señales en un ángulo de 45°. El funcionamiento del control remoto se basa en la emisión de secuencias de pulsos de luz IR, con un código que identifica la tecla pulsada.

Por lo cual, la dupla que tiene con el control nos ayudará a que solo nos ocupemos en identificar qué señal es la que emite en este caso el transmisor (Control Remoto), lectura que interpretaremos gracias a la conexión con la tarjeta UNO R3.

Comunicación IR

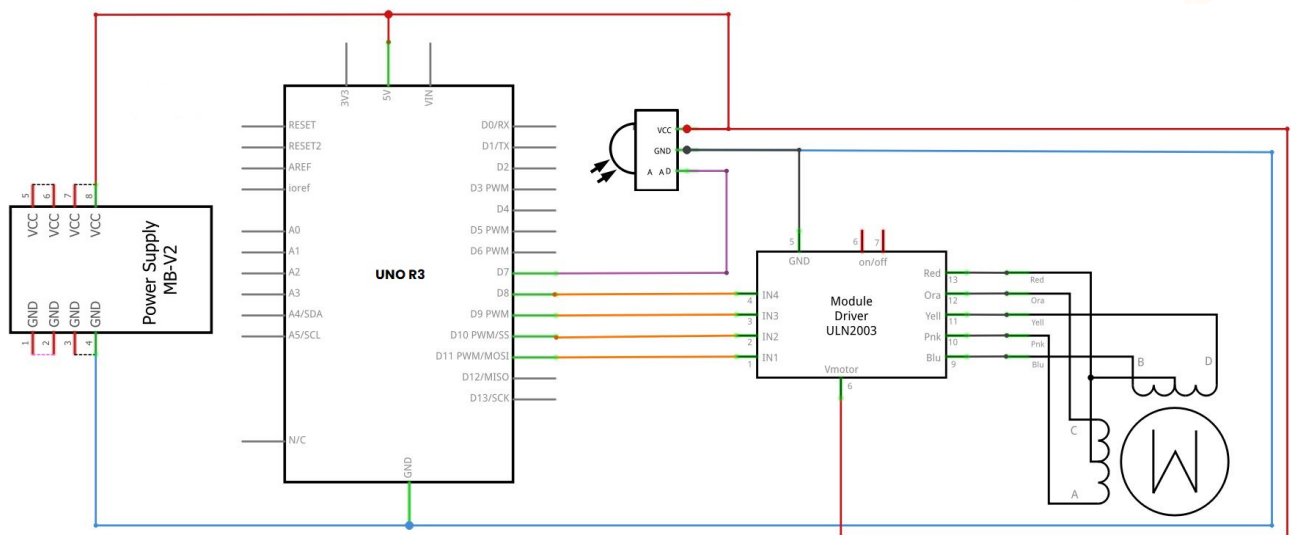
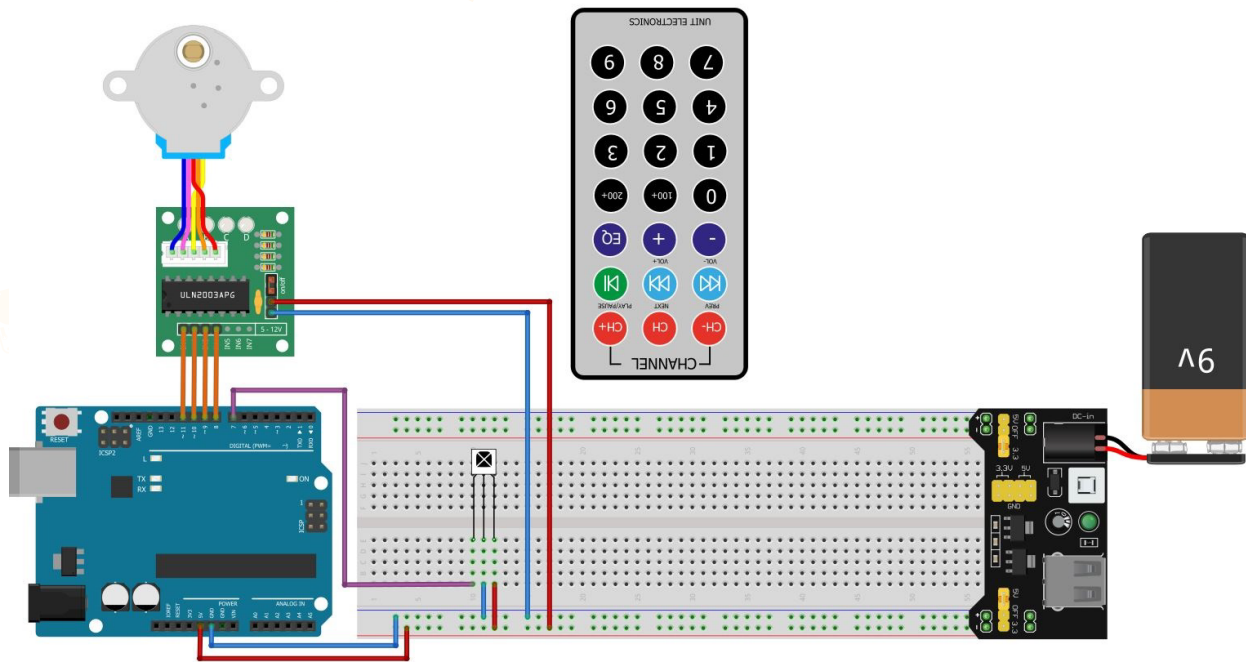
Un mando a distancia IR funciona conmutando el LED encendido y apagado en una secuencia particular. Alternando en dos estados ALTO y BAJO, cuando esté enviando una señal modulada (1) y apagada (0).

El mecanismo de funcionamiento de una transmisión por infrarrojos es por medio de un haz de luz, de alta frecuencia (en nuestro caso de 38kHz) se proyecta frente a un receptor. Este haz de luz no es más que una secuencia de bits (0 y 1) que son codificados por el receptor, a través de la polarización del fototransistor.

Lección 19 Led Receptor IR con Mando a Distancia y Control de Motor a Pasos

Diagrama de Conexión

En esta ocasión nos apoyaremos de una batería para alimentar al motor de pasos, ya que por la corriente de carga que el motor tendrá al ser activado es posible dañar nuestra tarjeta de desarrollo si lo conectamos directamente a ella.



Lección 19

Led Receptor IR con Mando a Distancia y Control de Motor a Pasos

Código de Funcionamiento

Para la realización del programa requerimos de 2 librerías, la primera Stepper que nos ayudará a trabajar con el motor a pasos e IRremote.

Stepper

by Arduino Versión 1.1.2 **INSTALLED**

Allows Arduino boards to control a variety of stepper motors. This library allows you to control unipolar or bipolar stepper motors. To use it you will need a stepper motor, and the appropriate hardware to control it.

[More info](#)

IRremote

by shirriff, z3t0, ArminJo Versión 3.5.2 **INSTALLED**

Send and receive infrared signals with multiple protocols Currently included protocols: Denon / Sharp, JVC, LG / LG2, NEC / Onkyo / Apple, Panasonic / Kaseikyo, RC5, RC6, Samsung, Sony, (Pronto), BoseWave, Lego, Whynter, MagiQuest.

New: RP2040 support and major refactoring of IRTimer.hpp.

[Release notes](#)

[More info](#)

En el caso de la librería IRremote nos permite enviar y recibir códigos con los que podremos manejar cualquier circuito con un control remoto de TV. La biblioteca referida, es compatible con Philips RC5, RC6 Philips, NEC, Sony SIRC y protocolos en bruto, se puede utilizar incluso para copiar los códigos de su mando a distancia y volver a transmitirlos, como un control remoto universal

Lección 19

Led Receptor IR con Mando a Distancia y Control de Motor a Pasos

```

#include <Stepper.h>
#include <IRremote.h>
//Declaración de variables
#define PASOS 32 // # de pasos por revolución en el eje
int AVANCE; // 2048 = 1 revolución
int IR = 7; // Pin 7 donde recibirá la información del IR
//Declaración de objetos para el motor e IR
//Declaración de pines que tendrán conexión al motor
Stepper small_stepper(PASOS, 11,10,9,8);
IRrecv irrecv(IR); // creación de instancia 'irrecv'
decode_results datos; // creación de instancia 'decode_datos'
void setup()
{
  Serial.begin(9600); //Iniciando puerto serial en 9600 bauds
  irrecv.enableIRin(); //Se habilita recepción de datos IR
}
void loop()
{
  Serial.println(datos.value, HEX); //Impresión de valor recibido por IR
  delay(2000); //espera de 2 seg
  if (irrecv.decode(&datos)) //se ha recibido
  {
    switch (datos.value) //operación switch/case para realizar una acción
    {
      case 0xFFA857: // Si se presiona el botón VOL+
        small_stepper.setSpeed(500); //Velocidad a la que rotara el motor
        AVANCE = 2048; // Rotara el motor en sentido de las manecillas del reloj
        small_stepper.step(AVANCE);
        delay(2000);
        break;
      case 0xFFE01F: // Si se presiona el botón VOL-
        small_stepper.setSpeed(500); //Velocidad a la que rotara el motor girara el motor en
        //sentido contrario de las manecillas del reloj
        AVANCE = -2048;
        small_stepper.step(AVANCE);
        delay(2000);
        break;
      case 0xFF6897: // Si se presiona el botón "0" cero
        digitalWrite(8, LOW); //se apagaran todos los motores
        digitalWrite(10, LOW);
        digitalWrite(9, LOW);
        digitalWrite(11, LOW);
        break;
    }
    irrecv.resume(); // recepción de nuevo valor IR
  }
}

```

Lección 20

RFID- RC522 y Servo Motor SG90

Introducción

Manteniendo el principio de comunicación vía IR, ahora realizaremos el control de giro un servomotor SG90 por medio del módulo RFID- RC522.

Materiales necesarios

- 1 Tarjeta UNO R3 x 1
- 2 Protoboard (Placa de pruebas) x 1
- 3 Servomotor SG90
- 4 Módulo lector RFID- RC522

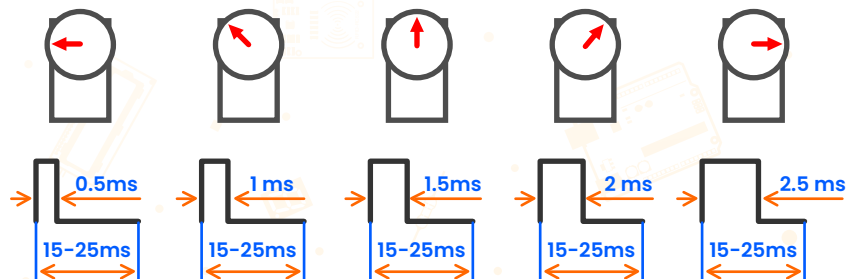
Conocimientos previos

Servomotor SG90

El SG90 es un pequeño actuador rotativo con especialidad de mantenerse fijo en una posición. Componiendo la parte mecánica, eléctrica y electrónica. Ya que se puede tener control preciso en posición angular, este servomotor puede rotar de 0° hasta 180° .



Para poder controlar cada ángulo del servomotor es necesario aplicar una señal PWM. Esta señal tiene la forma de una onda cuadrada. Dependiendo del ancho del pulso, el motor adoptará una posición fija.



Las señales que vemos en la imagen son las que permiten que el eje del motor adquiera determinada posición. Estas señales deben repetirse en el tiempo para que el motor mantenga una posición fija. La duración del ciclo de trabajo varía entre 15 y 25 milisegundos.

Lección 20

RFID- RC522 y Servo Motor SG90

Módulo Lector RFID RC522

El Lector RFID RC522 cuenta con un funcionamiento que consiste en pasar un TAG en forma de tarjeta o llavero cerca del lector RFID. El TAG enviará información al lector, ya sea un paquete de bytes o un simple código.

En el caso de este módulo necesita un sistema de modulación y demodulación de 13.56MHz. Y un protocolo de comunicación SPI.

Protocolo de comunicación SPI

Serial Peripheral Interface (SPI) es un protocolo de comunicaciones con una configuración full duplex <síncrono>, es decir, permite que dos dispositivos puedan comunicarse entre sí, en donde el maestro es el encargado de transmitir la información a sus esclavos, pero también el maestro puede recibir información de los esclavos.

Para el almacenamiento y envío de bits se requieren 2 registros de desplazamiento para realizar una conversión paralela a serial.

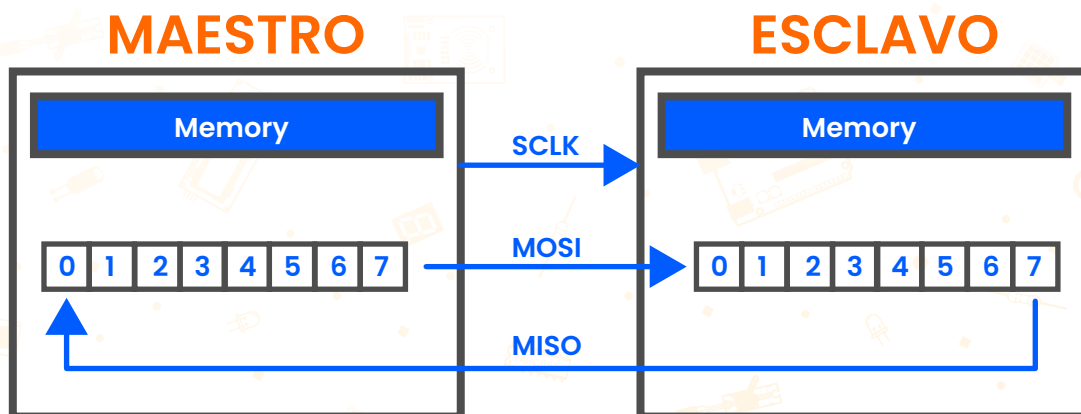
Existen cuatro líneas lógicas encargadas de realizar todo el proceso:

MOSI (Master Out Slave In): Línea utilizada para llevar los bits que provienen del maestro hacia el esclavo.

MISO (Master In Slave Out): Línea utilizada para llevar los bits que provienen del esclavo hacia el maestro.

CLK (Clock): Línea proveniente del maestro se encarga de enviar la señal de reloj para sincronizar los dispositivos.

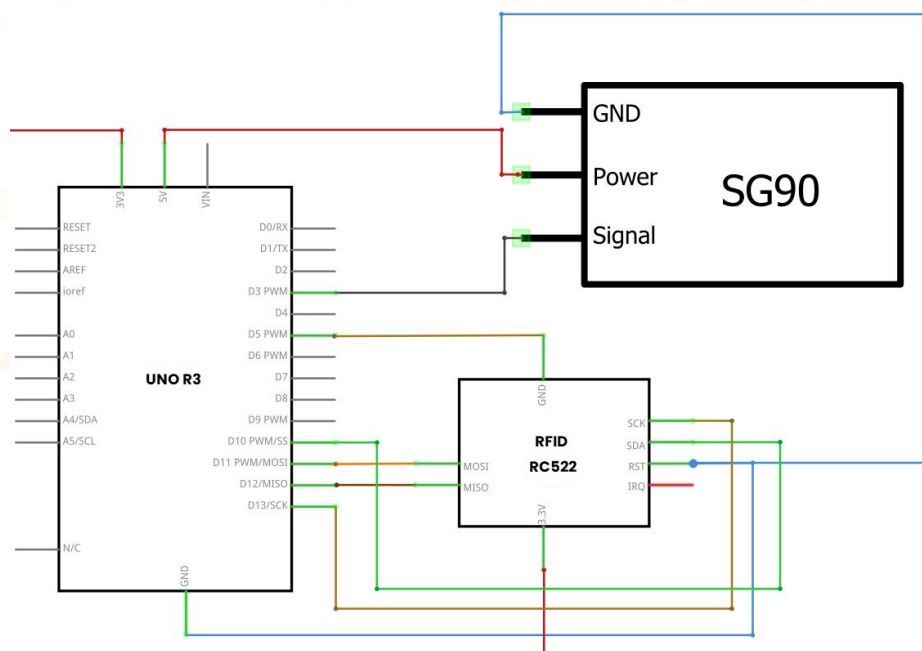
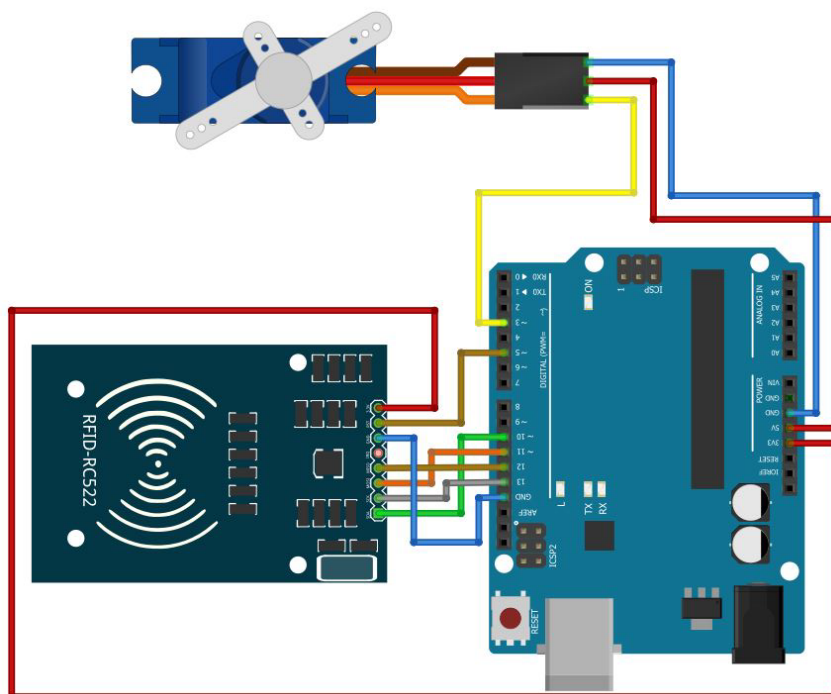
SS (Slave Select): Línea encargada de seleccionar y a su vez, habilitar un esclavo.



Lección 20 RFID- RC522 y Servo Motor SG90

Diagrama de Conexión

La siguiente conexión, busca que cuando sea detectado el TAG en forma de llavero el servo gire en 90° para la derecha y en caso de detectar el TAG de Tarjeta, girara 90° en sentido contrario, esto para simular dos entradas de una posible puerta.



Lección 20

RFID- RC522 y Servo Motor SG90

Código de Funcionamiento

Para el siguiente programa requerimos de 3 librerías, SPI para la comunicación, MFRC522 para el módulo RFDI-RC55 y finalmente SERVO para usar el servomotor SG90.

MFRC522-spi-i2c-uart-async

by [GithubCommunity,miguelbalboa,dirkx@webweaving.org](#) Versión 1.5.1 **INSTALLED**

Arduino RFID Library for MFRC522 (SPI, I2C and UART) with asynchronous callbacks Read/Write a RFID Card or Tag using the ISO/IEC 14443A/MIFARE interface. Modified from the original miguelbalboa to also support I2C and UART connections and provide, in addition to normal constant polling, an option to do asynchronous callbacks. I.e. have a function called each time that a valid swipe has happend. Used at the <https://makerspaceleiden.nl>.

[More info](#)

Servo

Built-In by Michael Margolis, Arduino Versión 1.1.6 **INSTALLED**

Allows Arduino boards to control a variety of servo motors. This library can control a great number of servos. It makes careful use of timers: the library can control 12 servos using only 1 timer.

On the Arduino Due you can control up to 60 servos.

[More info](#)

```
//Librerías para poder utilizar el RFDI-RC522, ServoMotor SG90 y protocolo de comunicación SPI
#include <SPI.h>
#include <MFRC522.h>
#include <Servo.h>
#define RST_PIN    5           // Pin del RFDI-RC522 a Pin 5
#define SS_PIN     10          // Pin del RFDI-RC522 a Pin 10

Servo sg90;
MFRC522 mfrc522(SS_PIN, RST_PIN); // Creando instancia para RFID
byte i; //variable , elemento de matriz de lectura de datos por el lector
byte datoUID[4]; //matriz para guardar valores arrojados por el detector
//Valor hexadecimal al usar el llavero <previamente comprobado>, puede variar
byte llavero[4]={0xBC,0x73,0x9F,0x63};
//Valor hexadecimal al usar la tarjeta <previamente comprobado>, puede variar
byte tag[4]={0x33,0xF0,0x20,0x0D};
void setup() {
  Serial.begin(9600); // Inicializa comunicación serial
  SPI.begin(); // Iniciación de comunicación por bus SPI
  sg90.attach(3); //Pin para la señal al servo-Pin 3
  sg90.write(0); //Siempre que se inicie el programa el servo tendrá un valor de 0°
  mfrc522.PCD_Init(); // inicialización de MFRC522
  Serial.println(F("Coloque tarjeta para realizar escaner")); //mensaje para usuario
```

Lección 20

RFID- RC522 y Servo Motor SG90

```

}
void loop() {
  //Reinicia el ciclo si no hay detección de tarjeta
  if ( ! mfr522.PICC_IsNewCardPresent())
    return;
  // Lee el valor del ID presentado
  if ( ! mfr522.PICC_ReadCardSerial())
    return;
  Serial.println("UID:");
  //Se guarda los datos del ID
  for (byte i = 0; i < mfr522.uid.size; i++) {
    //limitando la lectura de los valores en la matriz hexadecimal a 10 dígitos
    if (mfr522.uid.uidByte[i] < 0x10) {
      Serial.print("0");
    } else {
      Serial.print("");
    }
  }
  //imprime en Monitor Serial, el valor leído por el detector
  Serial.print(mfr522.uid.uidByte[i], HEX);
  datoUID[i]= mfr522.uid.uidByte[i];    //Se guardan los valores en la variable dato UID
  }
  mfr522.PICC_HaltA();
  if (datoUID[i] != llavero[i]){ //Si la lectura corresponde a los valores de la matriz tarjeta[]
    Serial.println("Tarjeta");
    for (int j = 0; j <= 180; j++) { //sentido antihorario abre
      sg90.write(j);                //se dará el valor del ángulo de 0 a 180 incrementando
      delay(5);                      //realiza el movimiento más rápido 5 milisegundo
    }
  }
}

```



ELABORADO POR EL EQUIPO DE UNIT

