



Universidad
del País Vasco

Euskal Herriko
Unibertsitatea

VITORIA-GASTEIZKO
INGENIARITZA
ESKOLA
ESCUELA
DE INGENIERÍA
DE VITORIA-GASTEIZ

Manual de Prácticas con Arduino

Ejercicios prácticos para las asignaturas Vehículos Híbridos y Eléctricos y Diseño de Sistemas Eléctricos

Escuela de Ingeniería de Vitoria-Gasteiz

Departamento Ingeniería Eléctrica

Josean Ramos Hernanz (josean.ramos@ehu.eus)

Mikel Armendáriz Pérez (mikel.armendarizp@ehu.eus)

La información recogida en estos apuntes ha sido elaborada y adaptada de
experiencia propia y de otras fuentes bibliográficas

Contenido

1.	Elementos principales usados en las prácticas	10
1.1.	Placa Arduino.....	10
1.2.	Placa de prototipado (protoboard).....	15
1.3.	Resistencias	17
1.4.	Cables jumpers o Dupont y Puentes.....	18
1.5.	Lista de elementos.....	20
1.6.	Bibliografía y direcciones de interés.....	23
2.	Lección 1 Monitor Serie y Monitor Serie Plotter	23
2.1.	Objetivo de la práctica	23
2.2.	Introducción al componente	23
2.3.	Componentes necesarios	26
2.4.	Esquema de conexión.....	26
2.5.	Códigos de los programas	27
2.6.	Bibliografía y direcciones de interés	27
3.	Lección 2 Blink	28
3.1.	Objetivo de la práctica	28
3.2.	Introducción al componente	28
3.3.	Componentes necesarios	28
3.4.	Esquema de conexión.....	29
3.5.	Códigos de los programas	29
3.6.	Bibliografía y direcciones de interés	29
4.	Lección 3 LED	29
4.1.	Objetivo de la práctica	29
4.2.	Introducción al componente	30
4.3.	Componentes necesarios	31
4.4.	Esquema de conexión.....	31

4.5.	Códigos de los programas	32
4.6.	Bibliografía y direcciones de interés.....	32
5.	Lección 4 RGB LED	32
5.1.	Objetivo de la práctica	32
5.2.	Introducción al componente	33
5.3.	Componentes necesarios	34
5.4.	Esquema de conexión.....	35
5.5.	Códigos de los programas	35
5.6.	Bibliografía y direcciones de interés.....	35
6.	Lección 5 Entradas Digitales.....	36
6.1.	Objetivo de la práctica	36
6.2.	Introducción al componente	36
6.3.	Componentes necesarios	39
6.4.	Esquema de conexión.....	39
6.5.	Códigos de los programas	40
6.6.	Bibliografía y direcciones de interés.....	40
7.	Lección 6 Zumbador Activo.....	40
7.1.	Objetivo de la práctica	40
7.2.	Introducción al componente	40
7.3.	Componentes necesarios	41
7.4.	Esquema de conexión.....	42
7.5.	Códigos de los programas	42
7.6.	Bibliografía y direcciones de interés.....	42
8.	Lección 7 Zumbador Pasivo	43
8.1.	Objetivo de la práctica	43
8.2.	Introducción al componente	43
8.3.	Componentes necesarios	44

8.4.	Esquema de conexión.....	44
8.5.	Códigos de los programas	44
8.6.	Bibliografía y direcciones de interés.....	44
9.	Lección 8 Interruptor de Bola de Inclinación.....	45
9.1.	Objetivo de la práctica	45
9.2.	Introducción al componente	45
9.3.	Componentes necesarios	46
9.4.	Esquema de conexión.....	46
9.5.	Códigos de los programas	47
9.6.	Bibliografía y direcciones de interés.....	47
10.	Lección 9 Servo	47
10.1.	Objetivo de la práctica	47
10.2.	Introducción al componente	47
10.3.	Componentes necesarios	52
10.4.	Esquema de conexión.....	53
10.5.	Códigos de los programas	54
10.6.	Bibliografía y direcciones de interés.....	54
11.	Lección 10 Módulo Sensor Ultrasónico HC-SR04.....	55
11.1.	Objetivo de la práctica	55
11.2.	Introducción al componente	55
11.3.	Componentes necesarios	57
11.4.	Esquema de conexión.....	58
11.5.	Códigos de los programas	59
11.6.	Bibliografía y direcciones de interés.....	60
12.	Lección 11 Teclado de Membrana.....	60
12.1.	Objetivo de la práctica	60
12.2.	Introducción al componente	60

12.3.	Componentes necesarios	62
12.4.	Esquema de conexión.....	62
12.5.	Códigos de los programas	62
12.6.	Bibliografía y direcciones de interés.....	63
13.	Lección 12 Sensor de Humedad y Temperatura DHT11.....	63
13.1.	Objetivo de la práctica	63
13.2.	Introducción al componente	63
13.3.	Componentes necesarios	65
13.4.	Esquema de conexión.....	65
13.5.	Códigos de los programas	65
13.6.	Bibliografía y direcciones de interés.....	65
14.	Lección 13 Módulo Joystick Analógico	66
14.1.	Objetivo de la práctica	66
14.2.	Introducción al componente	66
14.3.	Componentes necesarios	67
14.4.	Esquema de conexión.....	67
14.5.	Códigos de los programas	68
14.6.	Bibliografía y direcciones de interés.....	68
15.	Lección 14 Módulo de Receptor IR.....	68
15.1.	Objetivo de la práctica	68
15.2.	Introducción al componente	68
15.3.	Componentes necesarios	70
15.4.	Esquema de conexión.....	70
15.5.	Códigos de los programas	71
15.6.	Bibliografía y direcciones de interés.....	71
16.	Lección 15 Módulo de Matriz de Puntos LED MAX7219.....	72
16.1.	Objetivo de la práctica	72

16.2.	Introducción al componente	72
16.3.	Componentes necesarios	74
16.4.	Esquema de conexión.....	75
16.5.	Códigos de los programas	76
16.6.	Bibliografía y direcciones de interés.....	76
17.	Lección 16 Módulo GY-521 Medidor de Inercia	77
17.1.	Objetivo de la práctica	77
17.2.	Introducción al componente	77
17.3.	Componentes necesarios	80
17.4.	Esquema de conexión.....	81
17.5.	Códigos de los programas	82
17.6.	Bibliografía y direcciones de interés.....	82
18.	Lección 17 Sensor HC-SR501 PIR.....	82
18.1.	Objetivo de la práctica	82
18.2.	Introducción al componente	83
18.3.	Componentes necesarios	84
18.4.	Esquema de conexión.....	84
18.5.	Códigos de los programas	84
18.6.	Bibliografía y direcciones de interés.....	85
19.	Lección 18 Módulo Sensor Detección de Nivel de Agua.....	85
19.1.	Objetivo de la práctica	85
19.2.	Introducción al componente	85
19.3.	Componentes necesarios	87
19.4.	Esquema de conexión.....	87
19.5.	Códigos de los programas	88
19.6.	Bibliografía y direcciones de interés.....	88
20.	Lección 19 Módulo Reloj en Tiempo Real.....	89

20.1.	Objetivo de la práctica	89
20.2.	Introducción al componente	89
20.3.	Componentes necesarios	91
20.4.	Esquema de conexión.....	91
20.5.	Códigos de los programas	92
20.6.	Bibliografía y direcciones de interés	93
21.	Lección 20 Módulo de Sensor de Sonidos.....	93
21.1.	Objetivo de la práctica	93
21.2.	Introducción al componente	93
21.3.	Componentes necesarios	95
21.4.	Esquema de conexión.....	95
21.5.	Códigos de los programas	96
21.6.	Bibliografía y direcciones de interés	97
22.	Lección 21 Módulo RC522 RFID	97
22.1.	Objetivo de la práctica	97
22.2.	Introducción al componente	97
22.3.	Componentes necesarios	99
22.4.	Esquema de conexión.....	99
22.5.	Códigos de los programas	100
22.6.	Bibliografía y direcciones de interés	100
23.	Lección 22 Pantalla LCD1602.....	101
23.1.	Objetivo de la práctica	101
23.2.	Introducción al componente	101
23.3.	Componentes necesarios	104
23.4.	Esquema de conexión.....	104
23.5.	Códigos de los programas	105
23.6.	Bibliografía y direcciones de interés	105

24.	Lección 23 Termómetro con Termistor NTC	106
24.1.	Objetivo de la práctica	106
24.2.	Introducción al componente	107
24.3.	Componentes necesarios	108
24.4.	Esquema de conexión.....	108
24.5.	Códigos de los programas	109
24.6.	Bibliografía y direcciones de interés.....	109
25.	Lección 24 Control de Display de Ocho LED con 74HC595.....	110
25.1.	Objetivo de la práctica	110
25.2.	Introducción al componente	110
25.3.	Componentes necesarios	112
25.4.	Esquema de conexión.....	113
25.5.	Códigos de los programas	113
25.6.	Bibliografía y direcciones de interés.....	113
26.	Lección 25 Fotocélula LDR.....	114
26.1.	Objetivo de la práctica	114
26.2.	Introducción al componente	114
26.3.	Componentes necesarios	115
26.4.	Esquema de conexión.....	116
26.5.	Códigos de los programas	117
26.6.	Bibliografía y direcciones de interés.....	117
27.	Lección 26 74HC595 y Pantalla de 7 Dígitos Segmentado.....	118
27.1.	Objetivo de la práctica	118
27.2.	Introducción al componente	118
27.3.	Componentes necesarios	119
27.4.	Esquema de conexión.....	119
27.5.	Códigos de los programas	120

27.6.	Bibliografía y direcciones de interés	120
28.	Lección 27 Pantalla de 7 segmentos de cuatro dígitos	120
28.1.	Objetivo de la práctica	120
28.2.	Introducción al componente	120
28.3.	Componentes necesarios	121
28.4.	Esquema de conexión.....	121
28.5.	Códigos de los programas	122
28.6.	Bibliografía y direcciones de interés.....	122
29.	Lección 28 Motor de Corriente Continua	123
29.1.	Objetivo de la práctica	123
29.2.	Introducción al componente	123
29.3.	Componentes necesarios	126
29.4.	Esquema de conexión.....	127
29.5.	Códigos de los programas	128
29.6.	Bibliografía y direcciones de interés	128
30.	Lección 29 Relé	129
30.1.	Objetivo de la práctica	129
30.2.	Introducción al componente	129
30.3.	Componentes necesarios	132
30.4.	Esquema de conexión.....	133
30.5.	Códigos de los programas	134
30.6.	Bibliografía y direcciones de interés	135
31.	Lección 30 Motor Paso a Paso 28BYJ-48	135
31.1.	Objetivo de la práctica	135
31.2.	Introducción al componente	135
31.3.	Componentes necesarios	141
31.4.	Esquema de conexión.....	142

31.5. Códigos de los programas	143
31.6. Bibliografía y direcciones de interés.....	143

Arduino es una plataforma de electrónica abierta para la creación de prototipos basada en software y hardware flexibles y fáciles de utilizar.

La placa de Arduino va a tomar información del entorno, a través de sus pines de entrada y actuará sobre aquello que le rodea pudiendo controlar luces, motores y otros elementos mediante sus pines de salida. El microcontrolador en la placa Arduino se programa mediante el lenguaje de programación Arduino y el entorno de desarrollo Arduino. Los proyectos hechos con Arduino, una vez volcado a la placa el programa, se pueden ejecutar sin necesidad de conectar la placa a un ordenador.

Arduino tiene una página web (<https://www.Arduino.cc/>) en la que se puede encontrar todo lo necesario para programar en Arduino, desde la opción de descarga del entorno de desarrollo con sus instrucciones de instalación hasta una pequeña guía de programación. También se pueden encontrar ejemplos de programas y librerías que se pueden descargar.

1. Elementos principales usados en las prácticas

1.1. Placa Arduino

La placa elegida para estas prácticas es la placa Arduino Mega 2560 que es una actualización de Arduino Mega. Está basada en el microcontrolador ATmeg2560. Tiene 54 entradas/salidas digitales, de las cuales 14 proporcionan salidas PWM, 16 entradas digitales, 4 UARTS (puertos serie por hardware). El cristal oscilador es de 16MHz. Y posee conexión USB, entrada externa de alimentación, conector ICSP y botón reset integrado (también posee un pin de reset).

Tabla 1 Características básicas de la placa Arduino Mega/2560

Elementos	Valores
Microcontrolador	ATmega2560
Voltaje funcionamiento	5V
Voltaje entrada recomendado	7-12V
Voltaje entrada límite	6-20V
Pines E/S digitales	54 (14 son PWM)
Pines entrada analógica	16
Intensidad por pin	40mA
Intensidad pin 3.3V	50mA

Memoria flash	256KB (8KB usados para el arranque)
SRAM	8KB
EEPROM	4KB
Velocidad reloj	16MHz
Puertos serie	4

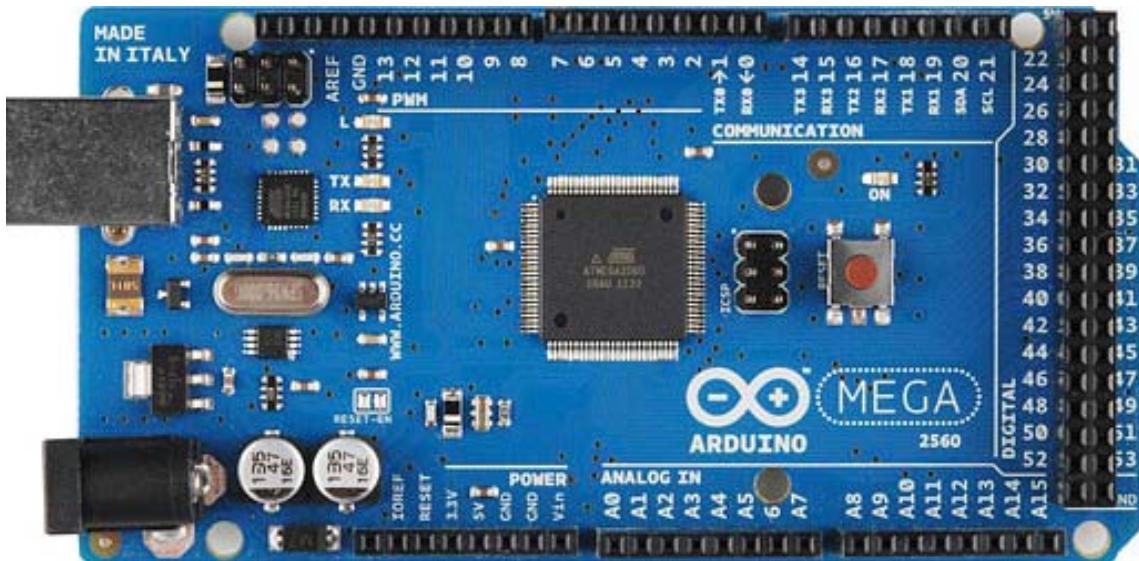


Fig. 1 Placa OFICIAL Arduino Mega 2560.

Pero debido a la utilización del kit ELEGOO, se puede adquirir en: (https://www.amazon.es/dp/B01MQPT9OD/ref=pe_3310721_185740151_TE_item), la placa que nosotros usaremos es:

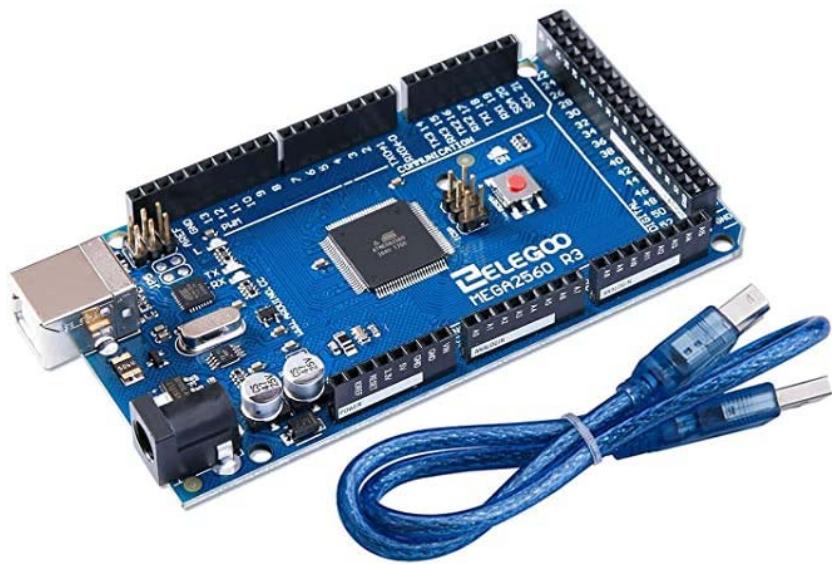


Fig. 2 Placa ELEGOO Arduino Mega 2560.

Alimentación

Arduino Mega puede ser alimentado vía USB o mediante una fuente de alimentación externa AC/DC o baterías. La fuente de alimentación se selecciona de forma automática, de tal manera que no es necesaria ninguna configuración adicional.

Arduino posee un conector macho de 2.1mm de centro-positivo. El rango de alimentación está entre 7 y 12V. Si el rango es superior a 12V, el regulador de tensión de Arduino podría dañarse. Si la tensión de alimentación es inferior a los 7V, el pin 5V de Arduino podría proporcionar menos de 5V y producirse inestabilidad.

Los pines de alimentación integrados en la placa son:

- **VIN:** Es la entrada de alimentación de Arduino, equivalente a la conexión por el conector de 2.1mm.
- **5V:** Los 5v proporcionados provienen del voltaje estabilizado por el regulador de tensión integrado y es el mismo que se utiliza para alimentar el microcontrolador. Esta tensión también puede provenir del USB u otra fuente externa estabilizada de 5V. El suministro de tensión a través de los pines de 5 V o 3.3 V no pasa por el regulador, y puede dañar la placa. No es aconsejable.
- **3.3V:** Es una fuente de voltaje de 3.3v generada en el chip FTDI de la placa. La corriente máxima que puede aportar son 50mA.
- **GND:** Pines de tierra. Tiene un total de 5: dos en la zona de alimentación, dos en la zona de los pines digitales y otro más junto a los pines de PWM.
- **IOREF.** Este pin en la placa proporciona la referencia de tensión con la que opera el microcontrolador.

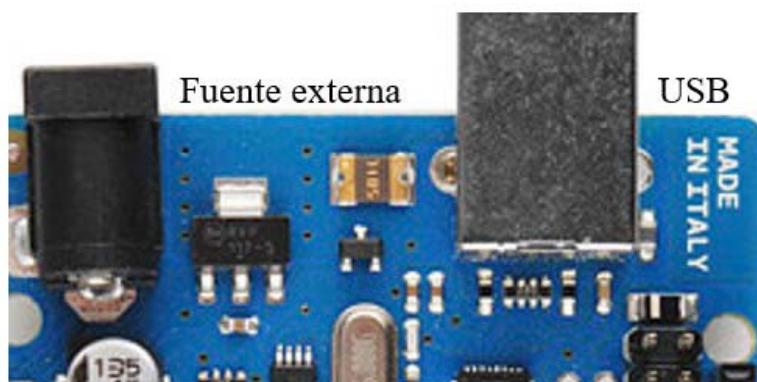


Fig. 3 Alimentación Arduino Mega 2560.

Memoria

El ATMEGA2560 posee 256KB de memoria flash para almacenar el código, de los cuales 8KB son utilizados para el proceso de arranque, 8KB de SRAM y 4KB de EEPROM. La memoria SRAM es volátil y la EEPROM no volátil.

Pines de Entrada y Salida

Cada uno de los 54 pines digitales de Arduino MEGA pueden utilizarse como entradas o salidas, usando las funciones pinMode(), digitalWrite(), y digitalRead(). Todas ellas operan a 5 voltios. Cada pin puede proporcionar o recibir un máximo de 40mA y tiene una resistencia interna (desconectada por defecto) de 20-50KΩ.

1. **Pines para transmisión serie RX/TX:** puede transmitir y recibir información a través de los puertos serie TTL de Arduino. Los pines serie 0 (RX, recibir) y 1(TX, transmitir) están conectados a los pines correspondientes del chip FTDI USB-to-TTL, de tal modo que el puerto USB se comporta como serie a través de estos pines.

El resto de los pines serie son:

- Serie: 0 (RX), 1 (TX);
- Serie1: 19 (RX), 18 (TX);
- Serie2: 17 (RX), 16 (TX);
- Serie3: 15 (RX), 14 (TX);



Fig. 4 Pines de las conexiones serie de Arduino Mega 2560.

2. **Interrupciones externas:** algunos pines se pueden configurar para lanzar una interrupción en valor LOW (0v), en flancos de subida o bajada o en cambios de valor:
 - Interrupción 0: Pin 2.
 - Interrupción 1: Pin 3.
 - Interrupción 2: Pin 21.
 - Interrupción 3: Pin 20.
 - Interrupción 5: Pin 18.
 - Interrupción 4: Pin 19.

3. **PWM:** pines 0 a 13: algunos pines pueden proporcionar una salida PWM (Pulse Wave Modulation, modulación de onda por pulsos) de 8 bits de resolución para obtener valores de 0 a 255 mediante la función analogwrite().



Fig. 5 Pines para generar señal PWM de Arduino Mega 2560.

4. **LED 13:** en la placa Arduino se ha integrado un led conectado al pin digital 13. Cuando éste tiene un valor alto, se enciende dicho LED.

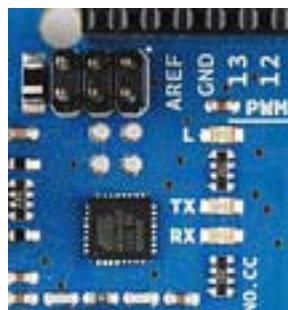


Fig. 6 Pines LED de Arduino Mega 2560.

5. **Entradas Analógicas:** Arduino MEGA posee 16 entradas analógicas; cada una de estas entradas proporciona una resolución de 10 bits (1024 valores). Por defecto se mide de tierra a +5V aunque es posible cambiar la cota usando el pin AREF y la función analogReference().



Fig. 7 Pines analógicos de Arduino Mega 2560.

6. **Reset:** suministrando un valor de 0V para reiniciar el microcontrolador.



Fig. 8 Pines Reset de Arduino Mega 2560.

7. **AREF.** Voltaje de referencia para las entradas analógicas. Usado por analogReference().



Fig. 9 Pines AREF de Arduino Mega 2560.

8. **TWI:** 20 (SDA) y 21 (SCL). TWI soporte de comunicación utilizando la biblioteca Wire.

1.2. Placa de prototipado (protoboard)

La placa de prototipos se emplea normalmente para realizar pruebas experimentales de circuitos electrónicos. Es muy sencilla de usar ya que los componentes simplemente se "pinchan" o insertan sobre la placa, sin necesidad de hacer soldaduras.

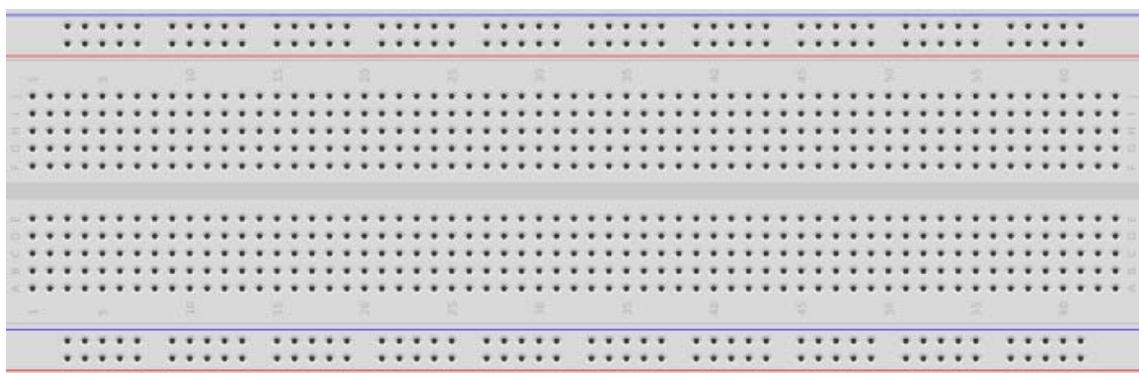


Fig. 10 Placa de prototipado (protoboard).

La placa tiene una serie de orificios en los que se introducen los terminales (patas) de los componentes. Debajo de estos orificios, en el interior de la placa, hay unas piezas metálicas con forma de pinza que sujetan el componente y además lo conectan eléctricamente con otros orificios, donde se puede introducir el terminal de otro componente, con lo que esos dos componentes electrónicos quedan conectados.

Los orificios de la placa de prototipos están conectados internamente con se ven en la siguiente imagen:

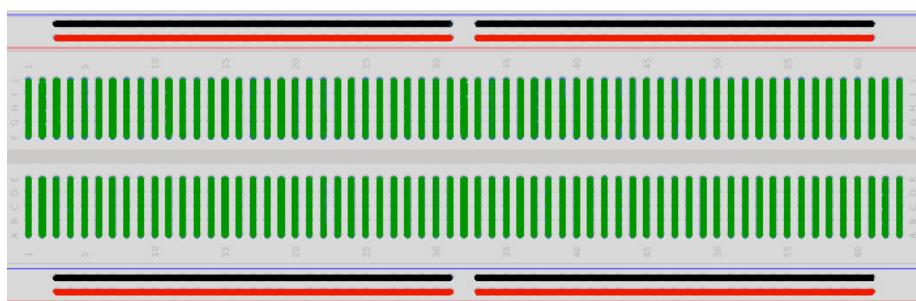


Fig. 11 Conexionado interno de los orificios de una placa de prototipado.

Las protoboard suelen venir acompañadas de unas letras y números de modo que cada orificio de la placa puede identificarse por la pareja de letra y número. Las protoboard tienen tres partes fáciles de identificar: el canal central, las pistas, y los buses.

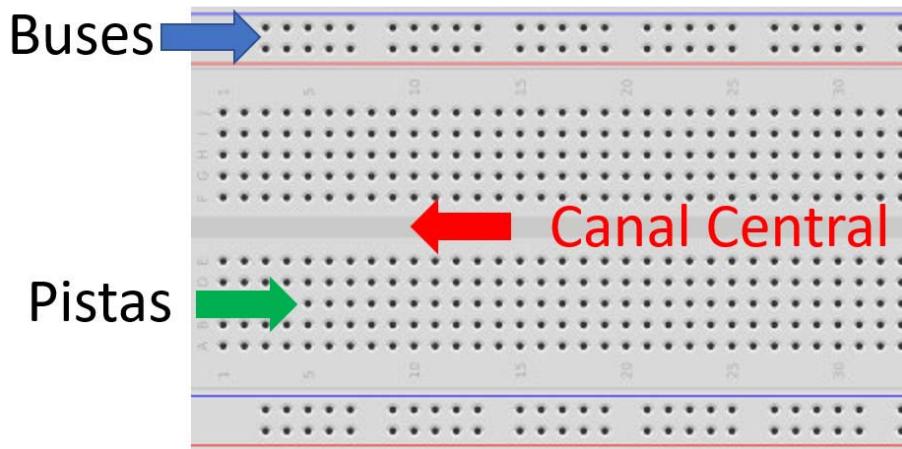


Fig. 12 Placa de prototipado (protoboard).

Buses

Los buses se encuentran a las partes superior e inferior de la placa Protoboard, y generalmente se emplean para conectar la tierra del circuito y sus voltajes de suministro. Los buses generalmente se indican con franjas negras o azules para marcar el bus de tierra, y franjas rojas para marcar el bus de voltaje positivo.

Pistas

El resto de los orificios de la Protoboard pertenecen a las pistas. Las pistas están separadas por filas de orificios conectados eléctricamente entre sí; cada fila (indicada con números) tiene conexión entre sí, y cada columna (indicada con letras) es independiente eléctricamente con las demás columnas, es decir, los orificios solo están conectados de forma horizontal.

Canal central

El canal central está ubicado en la parte central de la placa y está fabricado con un material aislante. Su función es separar las zonas de conexión superior e inferior de la placa. Es donde colocaremos los circuitos integrados y así se mantengan aislados los pines de ambos lados de dicho circuito integrado.

Conexión de componentes

Los protoboard tienen muchos enchufes pequeños (llamados ‘agujeros’) colocados en una cuadrícula de 2,54 mm. Los pines y cables de la mayoría de los componentes se pueden empujar directamente hacia los agujeros.

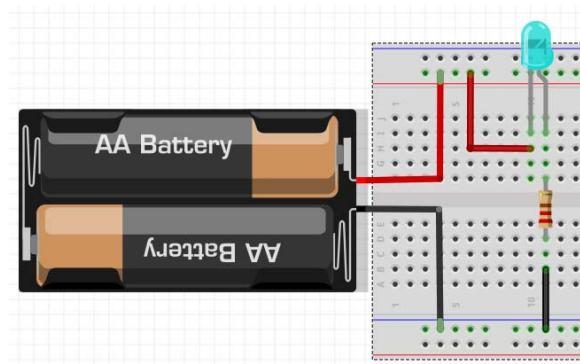


Fig. 13 Conexión de un circuito integrado en la protoboard.

Los circuitos integrados se insertan a través del canal central con su muesca o punto a la izquierda.

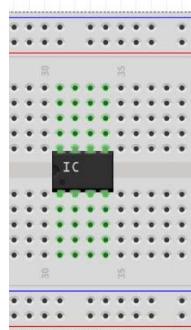


Fig. 14 Conexión de un circuito integrado en la protoboard.

1.3. Resistencias

La resistencia eléctrica es la propiedad física mediante la cual todos los materiales tienden a oponerse al flujo de la corriente. Por ello, se va a oponer al paso de la corriente eléctrica en un circuito, dando como resultado a un cambio en la tensión y en dicha corriente. El valor de las resistencias se mide en ohmios (se representa por la letra griega omega: Ω). Las bandas de colores en un lado de la resistencia indica su valor.

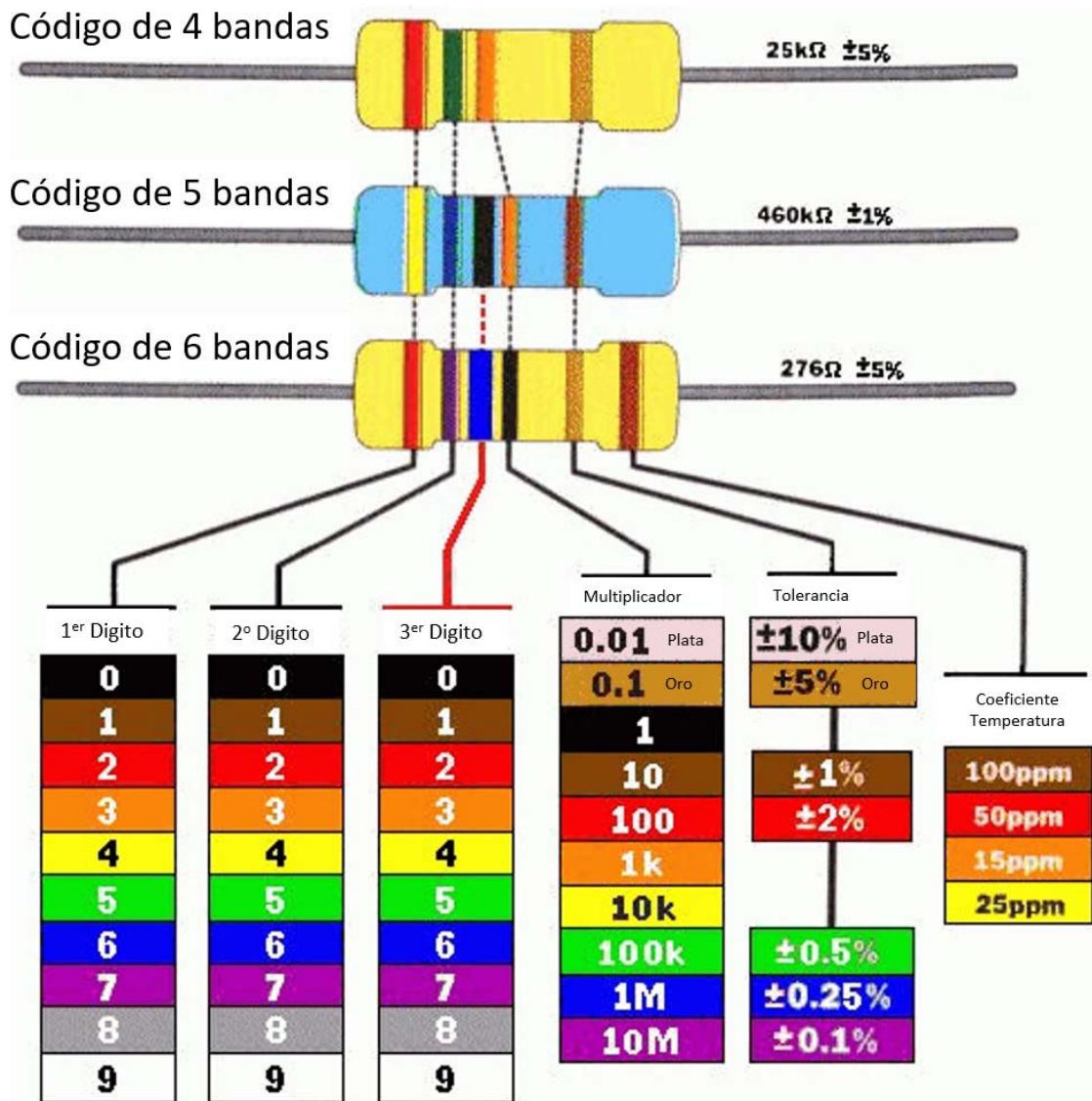


Fig. 15 Código de colores de las resistencias.

1.4. Cables jumpers o Dupont y Puentes.

Los cables jumpers, que son cables que tienen en sus terminales los jumpers que contienen sockets o pines, de ahí que se conozcan como cables jumper hembra o cables jumper macho. Cabe mencionar que estos cables también son conocidos como cables dupont.

Un cable puente para prototipos (o simplemente puente para prototipos), es un cable más o menos rígido con un conector en cada punta, que se usa normalmente para interconectar entre sí los componentes en una placa de pruebas. Por ejemplo, se utilizan de forma general para transferir señales eléctricas de cualquier parte de la placa de prototipos a los pines de entrada/salida de un microcontrolador.

Los cables o puentes se fijan mediante la inserción de sus extremos en los agujeros previstos a tal efecto en las ranuras de la placa de pruebas, la cual debajo de su superficie tiene unas planchas interiores paralelas que conectan las ranuras en grupos de filas o columnas según la zona. Los conectores se insertan en la placa de prototipos, sin necesidad de soldar, en los agujeros que convengan para el conexionado del diseño.

En el tipo con terminales aislados la disposición de los elementos y la facilidad de insertar los "conectores aislados" de los "cables puente" sobre la placa de pruebas permite el incremento de la densidad de montaje de ambos (componentes y puentes) sin temor a los cortocircuitos. Los cables o puentes varían en tamaño y color para distinguir las señales con las que se está trabajando.

Variación de cables con terminales esmaltados, según las combinaciones macho-hembra:

- Macho – macho
- Macho - hembra
- Hembra - hembra



Fig. 16 Cables Dupont y Puentes.

1.5. Lista de elementos.

Servo Motor (SG90) 1Elem



Placa de controlador de motor
paso a paso ULN2003 1Elem



Módulo de alimentación 1Elem



Placa controladora Mega 2560 1Elem



Modulo GY-521 1Elem



Módulo LCD 1602 (con encabezado
de pines) 1Elem



Módulo RFID RC522 1Elem



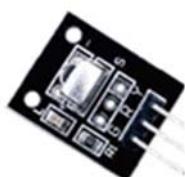
Motor paso a paso 1Elem



Placa de expansión para
prototipos 1Elem



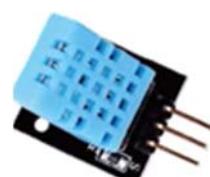
Módulo receptor de infrarrojos 1Elem



Módulo de joystick 1Elem



Módulo de temperatura y humedad
DHT11 1Elem



Sensor de ultrasonidos 1Elem



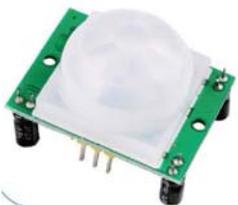
Modulo RTC DS1307 1Elem



Módulo codificador giratorio
Encoder 1Elem



Sensor de movimiento PIR HC-SR501 1 Elem

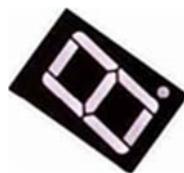


Módulo sensor de detección de nivel de agua 1Elem



Módulo de sensor de sonido 1Elem

Pantalla de 4 dígitos y 7 segmentos 1Elem



Pantalla de 1 dígito y 7 segmentos 1Elem

74HC5951Elem



Potenciómetro (10 K) 2Elem



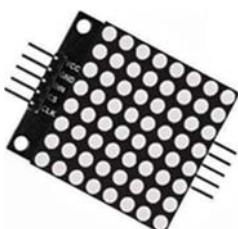
L2930 1Elem



Mando Remoto 1Elem



Modulo MAX7219 1Elem



Zumbador activo 1Elem



Zumbador pasivo 1Elem



65 Cables



Cable USB 1PCS



Cable Dupont de hembra a macho 20Ele



Motor de 3 Vcc y aspa del ventilador (con cable)
1Elem



Placa de Prototipado 1Elem



Teclado de Membrana 1Elem



Batería de 9V de CC 1Elem



Condensador Electrolítico (100uF 50V) 2Elem



Condensador Electrolítico (10uF
50V) 2Elem



Adaptador 9V 1A 1Elem



Transistor NPN (58050) 5Elem



Resistencias 120 Elem



Transistor NPN (PN2222) 5Elem



Botón Pulsador (Pequeño) 5Elem



Termistor 1Elem



Diodo Rectificador 5Elem



Sensor de inclinación 1Elem



LED Rojo 5Elem



LED Amarillo 5Elem



LED Azul 5Elem



Fotorresistor (Fotocélula) 2PCS



LED Verde 5Elem



LED Blanco 5Elem





1.6. Bibliografía y direcciones de interés.

- [1]. Elegoo. El kit más completo tutorial para mega2560.
- [2]. Arduino. Libro de proyectos. Scott Fitzgerald y Michael Shiloh.
- [3]. Curso Básico de Arduino: Misael Saenz Flores.
- [4]. Manual de prácticas con Arduino Uno R3. Roberto Patiño Ruiz.
- [5]. Arduino para jóvenes... y no tan jóvenes. Anaya Multimedia. Joan Ribas Lequerica.
- [6]. Guía del curso Arduino Kit. René Domínguez Escalona.
- [7]. Guía básica de Arduino. Tienda de Robótica. Cosas de Mecatrónica
- [8]. <https://www.Arduino.cc/>
- [9]. <https://mielecronicafacil.com/instrumentacion/protoboard/#conexion-de-componentes>
- [10]. <https://iesmiguelhernandez.es/moodle2/mod/page/view.php?id=9673>
- [11]. <https://intelectouniversal.com/electricidad/codigo-de-colores-de-resistencias/>

2. Lección 1 Monitor Serie y Monitor Serie Plotter

2.1. Objetivo de la práctica

Aprender a usar y visualizar la comunicación serial con Arduino. O sea, para ver en pantalla qué es lo que está haciendo un programa. Podremos leer un puerto analógico o digital con el ordenador en el monitor serial presente en el IDE de Arduino

2.2. Introducción al componente

Una de las maneras que Arduino puede comunicarse, es mediante sus puertos de entradas y salidas. El microcontrolador que contiene nuestra placa Arduino, posee un módulo receptor/transmisor Serie de tipo TTL-UART, por el cual podemos comunicarnos con otro Arduino, una placa Raspberry Pi, o cualquier sistema que posea un módulo similar, como nuestra computadora. El protocolo de comunicación UART (Universal

Asynchronous Receiver-Transmitter), cuenta con dos puertos, uno para emitir información (Tx) y uno para recibir información (Rx).

La comunicación serial entre dos dispositivos únicamente utiliza 3 líneas las cuales son:

- Línea de recepción de datos (RX)
- Línea de transmisión de datos (TX)
- Línea común (GND)

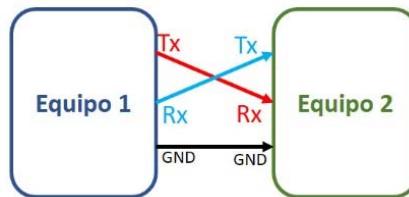


Fig. 17 Cableado comunicación serie.

El TX y RX del Arduino son los dos pines que emplea el dispositivo para realizar la comunicación por medio del protocolo serial. Los datos, por lo tanto, son transmitido en la línea o pin TX y son recibidos por la línea o pin RX.

Las placas de Arduino poseen unidades UART que operan a nivel TTL 0/5v, lo que las vuelve compatibles con la conexión USB. Las placas Arduino disponen de un conector USB que nos permite una conexión serie instantánea con nuestro ordenador.

Monitor Serie

Para realizar la conexión mediante puerto serie únicamente es necesario conectar nuestra placa Arduino empleando el mismo puerto que empleamos para programarlo. A continuación, abrimos el IDE Standard de Arduino que nos permite enviar y recibir fácilmente información a través del puerto serie. Haciendo clic en el "Monitor Serial" o en Herramientas → Monitor serie.

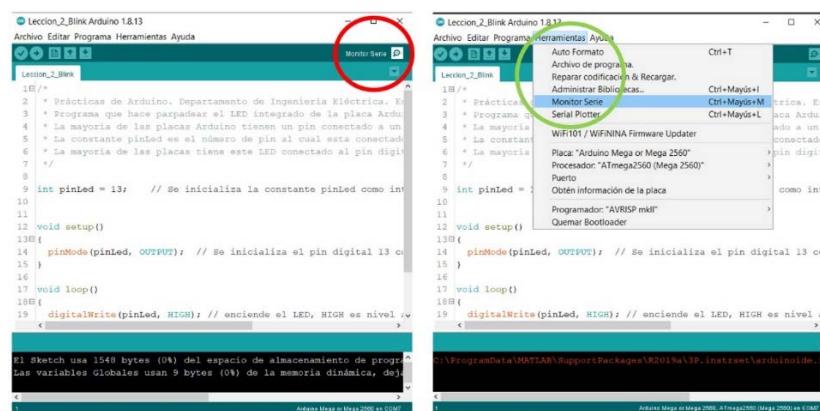


Fig. 18 Activación monitor serie.

En la parte superior escribiremos el dato que queremos mandar a la placa y pulsaremos Enviar. Y en el cuadrado grande veremos la información recibida. En la parte inferior podremos cambiar la velocidad de transmisión, que por defecto será de 9600 baudios



Fig. 19 Ventana del monitor serie para la comunicación con Arduino.

Principales instrucciones para la comunicación serie:

- `Serial.begin(velocidad);` Donde velocidad puede tomar los siguientes valores: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 o 115200. El 9600 indica el baud rate, o la cantidad de baudios que manejará el puerto serie. Se define baudio como una unidad de medida, usada en telecomunicaciones, que representa el número de símbolos por segundo en un medio de transmisión ya sea analógico o digital. Esta instrucción estará dentro de la función setup
- `Serial.print(dato);` Cuando queramos mandar un dato a la placa Arduino mediante la comunicación serie. Imprime los datos al puerto serie como texto ASCII. Si queremos que se escriba un texto o etiqueta, este debe ir entre comillas. `[Serial.print("Contador: ")]`, `Serial.print ("\\t")`; imprime un tabulador.
- `Serial.println(dato);` Hace lo mismo que la instrucción anterior, pero envía también un salto de línea (enter) al final de la comunicación.
- `Serial.write(dato);` Escribe datos en binario sobre el puerto serie. El dato es enviado como un byte o serie de bytes.
- `dato = Serial.read();` Leeremos un dato desde el puerto serie.
- `Serial.available();` Permite leer si existen datos disponibles.

Serie Plotter

El serial plotter es una herramienta disponible en el IDE de arduino que nos permite visualizar de forma gráfica el valor de una variable. Para hacerlo es tan fácil como enviar el valor a imprimir, en la función loop, con la instrucción `Serial.println()` y finalizando con `Serial.println()` para mostrar la última señal, entre cada señal es conveniente escribir `Serial.println()` y luego ir al menú herramientas y seleccionar serial plotter.

Como, por ejemplo:

```
void loop() {
    Serial.print("460"); // Escribimos la constante 460
    Serial.print(",");
    Serial.print(x * x + 5 * x - 6); // Graficamos x2 + 5x - 6
    Serial.print(",");
    Serial.println("-30"); // Escribimos la constante -30
    x++;
    delay(5);
    if (x == 20) x = -20;
}
```

2.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB

2.4. Esquema de conexión

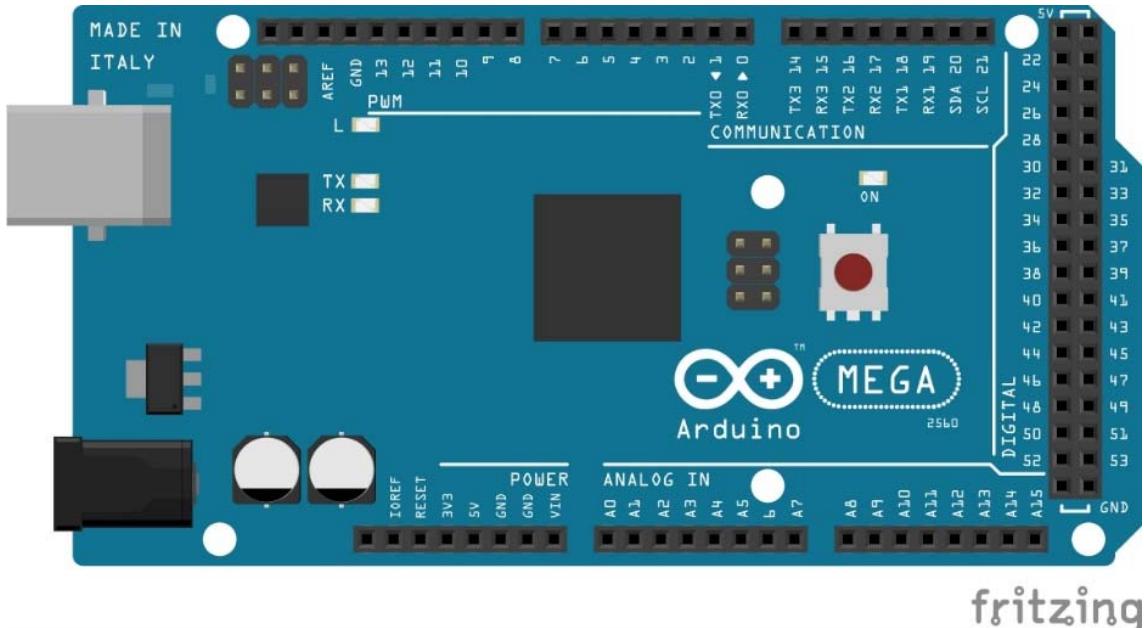


Fig. 20 Esquema de conexión Lección 1.

2.5. Códigos de los programas

Monitor Serie

1. Leccion_1_Monitor_Serial.ino: Programa que muestra el valor de un contador enviado desde la placa Arduino en el monitor serie. Este valor se incrementa cada segundo.

Serie Plotter

1. Leccion_1_Monitor_Serial_Plotter.ino: Programa que representa las ecuaciones $5x+1$ y x^2 el serial plotter.
2. Leccion_1_Monitor_Serial_Plotter_1.ino: Programa que representa la ecuación $x^2 + 5x - 6$ el serial plotter.
3. Leccion_1_Monitor_Serial_Plotter_2.ino: Programa que enciende y `paga el LED interno y muestra si está a 1 o a 0 en el serial plotter. Mostrando el pulso. En el monitor serie indica si está a 1 o a 0, dice si está a ON o a OFF o si está encendido o apagado.

2.6. Bibliografía y direcciones de interés

Monitor Serie

- [1]. <https://aprendiendoarduino.wordpress.com/category/monitor-serie/>
- [2]. <https://www.luisllamas.es/arduino-puerto-serie/>
- [3]. <https://www.zonamaker.com/arduino/intro-arduino/primeros-pasos-monitor-serial>
- [4]. <https://www.virtual-serial-port.org/es/articles/arduino-serial-monitor-alternative/>
- [5]. <https://www.manusoft.es/arduino/iniciacion-arduino/4-monitor-serie/>
- [6]. <http://panamahitek.com/comunicacion-serial-con-arduino/>
- [7]. <https://controlautomaticoeducacion.com/arduino/comunicacion-serial-con-arduino/>
- [8]. <https://aulaglaia.es/que-es-el-monitor-serie-de-arduino-y-para-que-sirve/>
- [9]. <https://www.rinconingenieril.es/lectura-potenciómetro-con-monitor-serie/>
- [10]. <https://www.automatizacionparatodos.com/puerto-serie-arduino/>
- [11]. <https://learn.adafruit.com/adafruit-arduino-lesson-5-the-serial-monitor/the-serial-monitor>
- [12]. <http://www.incb.com.mx/index.php/articulos/78-microcontroladores-y-dspes/3722-practicas-y-codigo-ejemplo-para-comunicacion-serie-y-monitor-serie-con-arduino-uno-mic043s>
- [13]. <https://zaragozamakerspace.com/index.php/lessons/curso-arduino-y-robotica-serial-monitor/>
- [14]. <https://aulaglaia.es/que-es-el-monitor-serie-de-arduino-y-para-que-sirve/>

[15]. [https://wiki.microduinoinc.com/Serial.print\(\)](https://wiki.microduinoinc.com/Serial.print())

Monitor Serie Plotter

- [1]. <https://www.luisllamas.es/arduino-serial-plotter/>
- [2]. <https://fidiasrodriguez.com/como-usar-el-monitor-serie-y-el-serial-plotter/>
- [3]. <https://aprendiendoarduino.wordpress.com/tag/serial-plotter/>
- [4]. https://github.com/jecrespo/aprendiendoarduino-Curso_Arduino_2017/blob/master/Ejercicio48-Blink_Plotter_Mejorado/Ejercicio48-Blink_Plotter_Mejorado.ino
- [5]. <https://www.prometec.net/serial-plotter/>
- [6]. <https://arduinogetstarted.com/tutorials/arduino-serial-plotter>
- [7]. <https://blog.bricogeek.com/noticias/arduino/como-utilizar-arduino-serial-plotter/>
- [8]. <https://www.instructables.com/Ultimate-Guide-to-Arduino-Serial-Plotter/>
- [9]. <http://diwo.bq.com/serial-plotter-conoce-la-nueva-caracteristica-del-ide-de-arduino/>

3. Lección 2 Blink

3.1. Objetivo de la práctica

Hacer parpadear el led interno de la placa Arduino.

3.2. Introducción al componente

Para esta práctica no se va a necesitar aprender ninguna teoría nueva. Lo único que hay que tener en cuenta es que el led interno que hay en la placa Arduino y que está conectado al PIN13 de Arduino ya lleva incorporada una resistencia interior, si no deberíamos colocar en serie con el led una resistencia de valor entre 220 y 500 ohmios dependiendo del consumo de corriente del diodo.

3.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB

3.4. Esquema de conexión

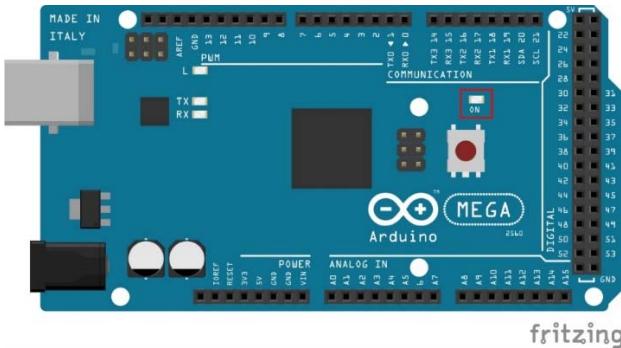


Fig. 21 Esquema de conexión Lección 2.

3.5. Códigos de los programas

- Las placas Arduino vienen siempre programadas de fábrica con el ejemplo Blink.
- Leccion_2_Blink.ino: Programa que hace parpadear el LED integrado de la placa Arduino MEGA.
- Leccion_2_Blink_Monitor_Serial_1.ino: Programa que enciende o apaga el LED integrado en la placa Arduino. Si se envía ‘a’ a la placa Arduino apaga el LED, y en caso de enviar ‘b’ lo enciende.
- Leccion_2_Blink_Monitor_Serial_2.ino: Programa que hace parpadear el LED integrado en la placa Arduino el número de veces que le mandas por el monitor serie.

3.6. Bibliografía y direcciones de interés

- [1]. <http://mecabot-ula.org/tutoriales/arduino/practical-encender-y-apagar-con-arduino/>
- [2]. <http://panamahitek.com/primeros-pasos-con-arduino-encender-un-led/>

4. Lección 3 LED

4.1. Objetivo de la práctica

Encender y apagar un LED externo con Arduino mediante la tarjeta Arduino Mega 2560. Y cómo cambiar el brillo de un LED usando diferentes valores de resistencia.

4.2. Introducción al componente

LED es un Diodo Emisor de Luz (Light Emitting Diode), y como todo diodo tiene un ánodo y un cátodo, siendo el ánodo el positivo y el cátodo el negativo. El cátodo se diferencia del ánodo mirando en el interior del cristal del LED.

Formas de determinar la polaridad de un LED:

1. La pata más larga siempre va a ser el ánodo. Irá conectada al terminal de valor más alto de tensión, normalmente al pin salida de Arduino.
2. En el lado del cátodo, la base del LED tiene un borde plano. Irá conectado al terminal más bajo de tensión, normalmente a tierra o GND.

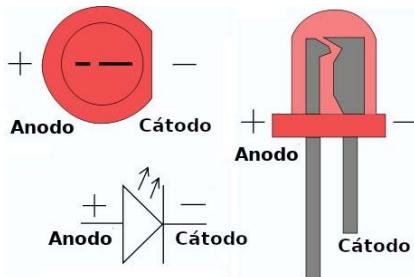


Fig. 22 Partes de un Led.

O sea, que la patilla larga debe ser conectada al voltaje positivo (ánodo), y la corta al voltaje negativo (cátodo).

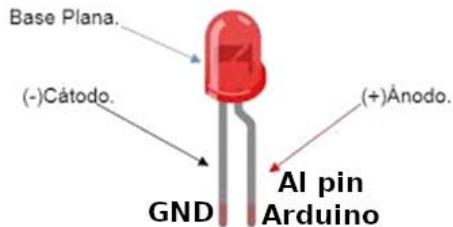


Fig. 23 Conexión a Arduino de un Led.

Los diodos, aunque estén conectados de forma correcta necesitan de un cierto valor de tensión al que llamamos tensión de polarización directa (V_d), que depende del tipo de diodo. A partir de esta tensión decimos que el diodo está polarizado y la corriente puede atravesarlo libremente con una resistencia casi nula. En el momento que superamos la tensión de polarización, y dado que la resistencia del diodo es muy pequeña, se genera una gran corriente que destruirá el diodo. Por ese motivo, necesitamos una resistencia que limite la cantidad de corriente que circula por el diodo.

Para calcular el valor de la resistencia que limitará la corriente que circulará por el diodo, con la finalidad de proteger a la salida digital del Arduino, necesitamos saber las características eléctricas del LED. Un LED trabaja aproximadamente con una tensión de

1,5 V y consume una corriente de 0,015 A. Entonces, la caída de tensión en la resistencia limitadora será de:

$$V_{\text{Arduino}} - V_{\text{led}} = 5 \text{ V} - 1,5 \text{ V} = 3,5 \text{ V}.$$

Luego, la resistencia que tendremos que poner en el circuito será $\Rightarrow R = V / I$

Donde la V serán los voltios sobre la resistencia y la I será la corriente que consume el LED.

$$R = 3,5 / 0,015 = 233,3 \Omega$$

Al menos deberíamos colocar en serie con el led una resistencia de valor entre 220 y 500 ohmios dependiendo del consumo de corriente del diodo. Nosotros usaremos una resistencia limitadora de 330Ω para todos los leds incluso para los RGB.

4.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- 1 LED de 5mm
- 1 resistencia de 330 ohmios
- 1 resistencia de $1 \text{ k}\Omega$
- 1 resistencia de $10 \text{ k}\Omega$

4.4. Esquema de conexión

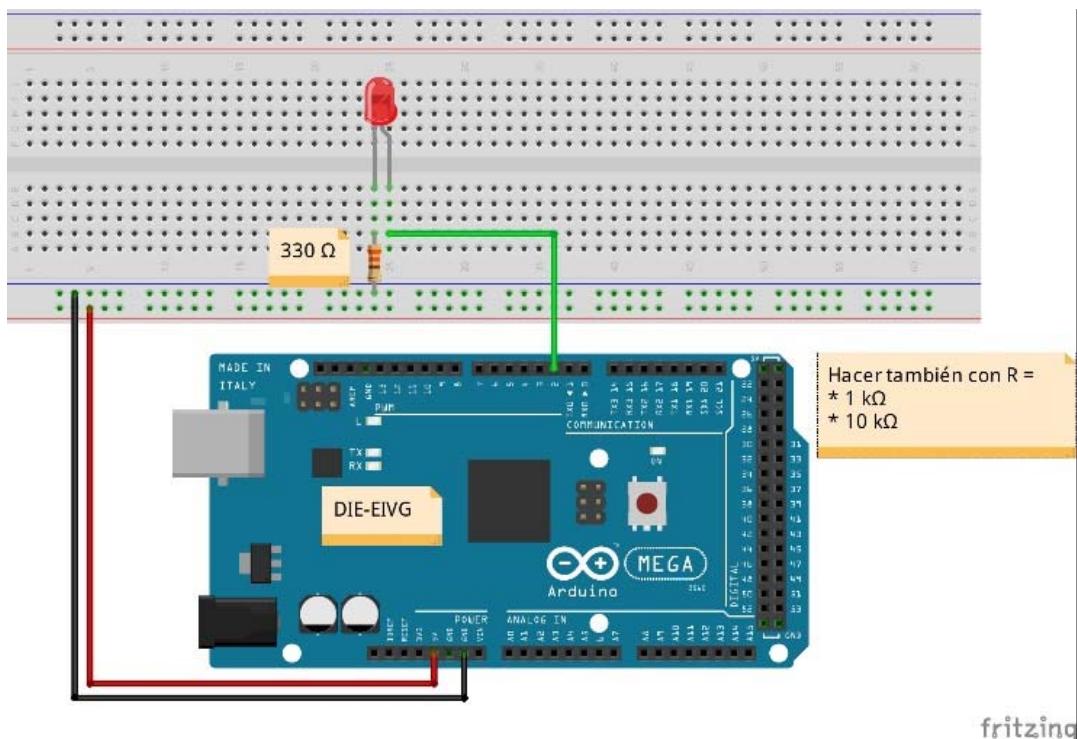


Fig. 24 Esquema de conexión Lección 3.

4.5. Códigos de los programas

1. Leccion_3_Blink: Programa que hace parpadear un el LED conectado al pin
2. Para proteger al led hay que colocar una resistencia en serie con el de 330 ohmios. Para variar el valor del brillo cambiaremos la resistencia en serie por una de 1 kΩ y luego de 10 kΩ

4.6. Bibliografía y direcciones de interés

- [1]. <https://www.luisllamas.es/encender-un-led-con-arduino/>
- [2]. <https://educaendigital.com/encender-un-led-con-arduino-programacion-2>
- [3]. <https://educaendigital.com/encender-un-led-con-arduino>
- [4]. <https://educaendigital.com/que-es-arduino-por-que-usarlo>
- [5]. <http://codigoelectronica.com/blog/arduino-blink>
- [6]. <https://arduino.cl/ejemplo-blink/>
- [7]. <https://roboticarduino.com/proyecto-arduino-led-blink/>
- [8]. <https://moviltronics.com/primer-programa-arduino-blink/>
- [9]. <https://www.reset.etsii.upm.es/servicios/tutoriales/tutorial-1-blink/>
- [10]. <https://blog.wokwi.com/5-ways-to-blink-an-led-with-arduino/>
- [11]. <https://learn.adafruit.com/adafruit-arduino-lesson-2-leds/blinking-the-led>
- [12]. <https://www.circuitbasics.com/arduino-basics-controlling-led/>
- [13]. <https://makeabilitylab.github.io/physcomp/arduino/led-blink.html>
- [14]. http://www.practicasconarduino.com/manualrapido/led_parpadeante.html
- [15]. <https://www.esploradores.com/practica-1-hola-mundo-parpadeo-de-un-led-blink/>

5. Lección 4 RGB LED

5.1. Objetivo de la práctica

Conocer la utilidad y manejo de un LED RGB. Aprender a programar los pines de Arduino como salidas analógicas (PWM). Se muestran los colores básicos Rojo, Verde, Azul, Amarillo, Cyan y Magenta. También aprenderemos a programar el led RGB para pasar de rojo a verde, de verde a azul, de azul a rojo, pasando por todas las tonalidades

5.2. Introducción al componente

El led RGB es un tipo especial de LED que tiene la capacidad de generar luz de cualquier color. El truco empleado para lograrlo es utilizar en realidad no uno sino RGB tres leds dentro del mismo encapsulado, de cada uno de los tres colores primarios: rojo (Red), verde (Green) y azul (Blue). Si se controla adecuadamente la cantidad de corriente que se suministra a cada led, se puede controlar la componente de luz de cada uno de estos tres colores y combinarlos para lograr el color que se desee.

Estos tres leds no tienen todos sus pines disponibles, sino que se conectan juntos todos los ánodos o todos los cátodos, dando lugar a dos tipos de leds RGB: ánodo común y cátodo común. El conexionado interno es el que muestra la siguiente imagen:

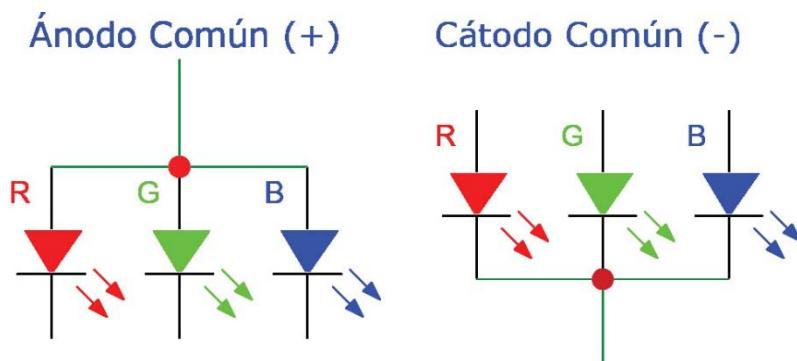


Fig. 25 Conexiones ánodo común y cátodo común de un LED RGB.

Antes de hacer el montaje hay que saber qué tipo de LED RGB estaremos utilizando para elegir correctamente su circuito de conexión. Una manera simple de conocer el tipo de LED RGB que tienes, es mediante un polímetro, utilizándolo en su modo de prueba de diodos. Conectaremos el terminal negativo del polímetro al pin común del LED RGB, y el terminal positivo del polímetro al pin del extremo. Si la configuración del LED es cátodo común, deberá encenderse una pequeña luz en el LED,

Los leds RGB se basan en la síntesis aditiva, pudiendo representar cualquier color mediante la mezcla por adición de los tres colores de luz primarios (rojo, verde y azul).

Siendo los colores secundarios (según el modelo aditivo RGB):

- rojo + verde = AMARILLO
- rojo + azul = MAGENTA
- verde + azul = CIAN

Como sabemos el color resultante es la suma de los tres primarios, para ello asignaremos valores numéricos en nuestro código a cada uno de los colores primarios. Sabiendo que:

- El resultado de la mezcla es un color monocromático.
- La intensidad de cada señal (o luz de color) va desde 0 (no interviene en la mezcla) hasta 255.
- De este modo, el rojo se obtiene con la siguiente mezcla $(255, 0, 0) = (R, G, B)$.
- El verde con $(0, 255, 0) = (R, G, B)$.
- El azul $(0, 0, 255) = (R, G, B)$.
- El color negro o, lo que es lo mismo, la ausencia de color se obtiene con $(0, 0, 0) = (R, G, B)$.
- El color blanco, con $(255, 255, 255) = (R, G, B)$.
- El color amarillo $(255, 255, 0)$; cian $(0, 255, 255)$ y magenta $(255, 0, 255)$.

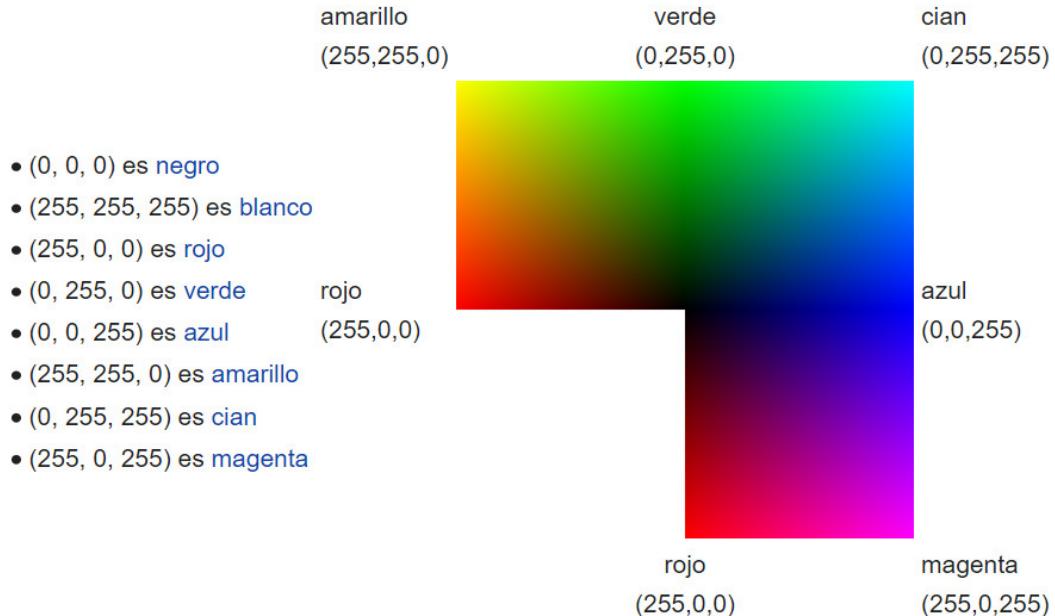


Fig. 26 Gama de colores RGB.

5.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 LED RGB de cátodo común (Ánodo común utiliza 5V en el pin común, mientras que el cátodo común se conecta a tierra.)
- 3 resistencias de 330Ω

5.4. Esquema de conexión

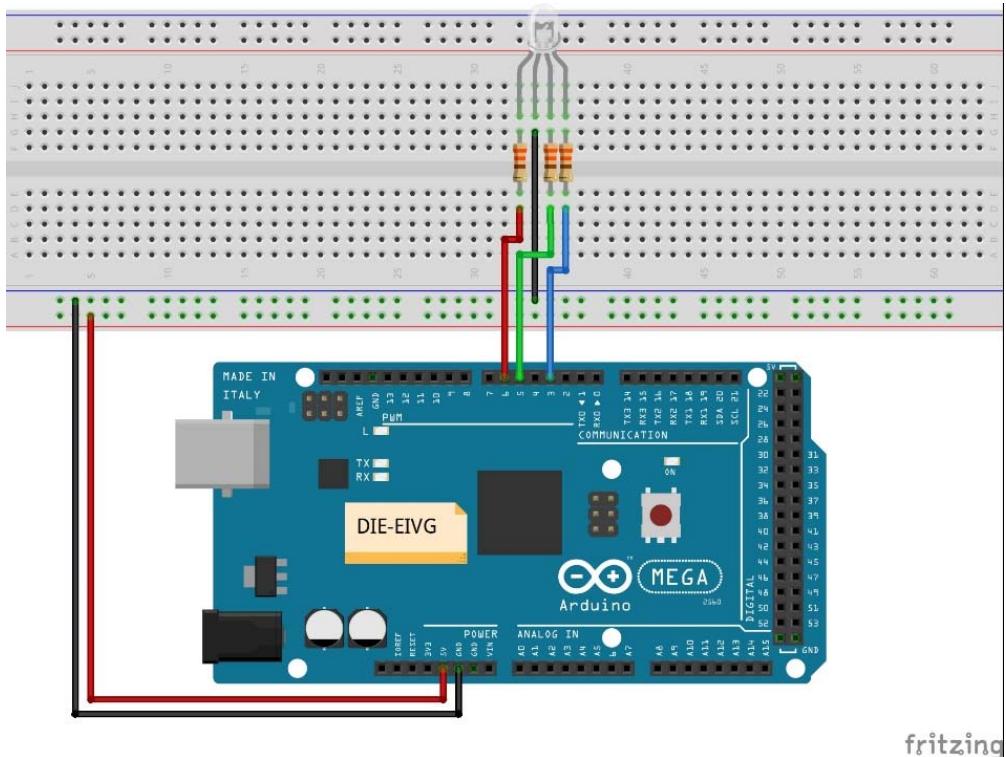


Fig. 27 Esquema de conexión Lección 4.

5.5. Códigos de los programas

1. Leccion_4_RGB.ino: Programa que enciende un LED RGB conectado a los pines 3, 5 y 6. Para proteger al led hay que colocar una resistencia en serie con el de 330 ohmios. Se muestran los colores básicos ROJO, VERDE, AZUL, AMARILLO, CYAN y MAGENTA.
2. Leccion_4_RGB_1: Programa que enciende un LED RGB conectado a los pines 3, 5 y 6. Para proteger al led hay que colocar una resistencia en serie con el de 330 ohmios. Se pasa de ROJO a VERDE, de VERDE a AZUL, de AZUL a ROJO, pasando por todas las tonalidades.

5.6. Bibliografía y direcciones de interés

- [1]. <https://www.programoergosum.com/cursos-online/arduino/255-salidas-analogicas-pwm-con-arduino/led-rgb-anodo-comun>
- [2]. <https://www.prometec.net/rgb-led/>
- [3]. <https://www.automatizacionparatodos.com/led-rgb-con-arduino/>
- [4]. <https://aprendiendoarduino.wordpress.com/2019/03/02/led-rgb-con-arduino/>

- [5]. <https://create.arduino.cc/projecthub/IATecNo/encender-rgb-led-d20e14>
- [6]. <https://create.arduino.cc/projecthub/muhammad-aqib/arduino-rgb-led-tutorial-fc003e>
- [7]. <https://educarparaelcambio.com/arduino/reto-10-led-rgb-jugando-con-los-colores-primarios-de-la-luz/>
- [8]. <https://techmake.com/blogs/tutoriales/empezando-con-arduino-1f-led-rgb>
- [9]. <https://blog.330ohms.com/2020/06/12/como-conectar-led-rgb-neopixel-ws2812-a-arduino/>
- [10]. <https://www.profetolocka.com.ar/2015/01/27/control-de-un-led-rgb-con-arduino/>
- [11]. <https://www.mecatronicalam.com/es/tutoriales/proyectos-con-arduino/arduino-led/>

6. Lección 5 Entradas Digitales

6.1. Objetivo de la práctica

Aprender a utilizar los interruptores o pulsadores de presión con entradas digitales para encender y apagar un LED. Siendo estas secuencias: al presionar el pulsador se encenderá el LED y cuando lo sueltes se apagará; al presionar el pulsador se encenderá el LED; pulsar otra vez el pulsador se apagará el LED; al presionar el pulsador se encenderá el LED; pulsar el otro pulsador se apagará el LED.

6.2. Introducción al componente

Una entrada digital es un pin del que disponemos en nuestro Arduino que leeremos al ejecutar el programa y que podrá tener dos estados dependiendo del funcionamiento del dispositivo al que está conectado. Estos 2 estados serán 0 o 1, bajo-alto o LOW-HIGH. dependiendo si el pin se conecta a la tensión de alimentación (usualmente 5V) o a masa (0V). Lo que hacemos en la práctica al leer una entrada digital es comparar su nivel de tensión con una referencia umbral. Si supera esta referencia la entrada estará a nivel alto y le asignamos el valor de 1. De lo contrario diremos que está a nivel bajo y la tomaremos como 0.

En general es razonable suponer que la tensión umbral es cercana al punto medio entre $-V_{cc}$ y $+V_{cc}$. No obstante, debemos evitar medir tensiones cerca de la tensión umbral porque pueden provocar mediciones incorrectas.

En Arduino los valores de alimentación habituales son 0V y 5V. En este caso la tensión umbral será muy cercana a 2'5V. Por tanto, si medimos una tensión con un valor intermedio entre 0 a 2'5V Arduino devolverá una lectura LOW, y si medimos un valor entre 2'5V y 5V, devolverá HIGH.

Es probable que muchas veces el elemento (led, sensor, ...) tenga un comportamiento inestable por no estar conectado a ninguna tensión (1 o 0), esto se conoce como alta impedancia. Para conseguir que el elemento quede en un estado determinado cuando el pulsador esté abierto, debemos hacer uso de las resistencias Pull-Up y Pull-Down.

Las resistencias de Pull-Down y Pull-Up se conectan entre el PIN digital y una de las tensiones de referencia (0V o 5V) y "fuerzan" el valor de la tensión a LOW o HIGH, respectivamente.

- La resistencia de Pull-Up fuerza HIGH cuando el pulsador está abierto. Cuando está cerrado el PIN se pone a LOW, la intensidad que circula se ve limitada por esta resistencia. Mientras el pulsador esté sin presionar, la referencia que tomará el elemento será de 5V, cuando se presiona el pulsador, la referencia que toma el elemento es 0V debido a que está conectado a tierra.
- La resistencia de Pull-Down fuerza LOW cuando el pulsador está abierto. Cuando está cerrado el PIN se pone a HIGH, y la intensidad que circula se ve limitada por esta resistencia. Esta es la configuración recomendada cuando queremos tener un valor HIGH al presionar el pulsador y un valor LOW al dejar de presionar el pulsador, el uso común de encender un elemento al presionar el pulsador.

Podemos hacerlo también por programación, `pinMode(buttonA, INPUT_PULLUP)`; el valor predeterminado de la entrada es un uno, a menos que se pulse el botón que pasará a ser cero.

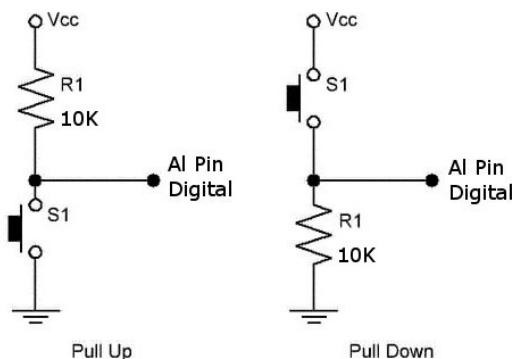


Fig. 28 Esquema de montaje de una resistencia Pull-Up y Pull-Down.

El interruptor que usaremos en las prácticas es del tipo pulsador lo que indica que necesitamos mantener una presión sobre él para que funcione. Por norma general, los pulsadores tienen un estado inicial donde no permiten pasar la corriente. Tenemos de dos tipos de tres o de cuatro patillas.

El pulsador de tres patillas su conexión es muy simple, usaremos siempre dos patillas la del centro que irá conectado al elemento y la patilla de la derecha VCC que irá conectado a 5 V si queremos que nuestro elemento se active.



Fig. 29 Tipos de pulsadores de presión.

En el caso del pulsador de cuatro patillas y que están conectadas a pares como se ve en el siguiente esquema.



Fig. 30 Esquema de conexión de los pulsadores de presión.

Cuando pulsamos el interruptor se cierra el circuito y dejamos pasar la corriente, unimos A y D. Esto nos permite, por ejemplo, activar un LED, un motor o cualquier otro elemento.

Instrucciones:

- `pinMode(pin,modo)`: Configura el pin especificado para comportarse como una entrada (INPUT) o una salida (OUTPUT). `pinMode(Pin9, OUTPUT)`
- `digitalWrite(pin,valor)`: Asigna el valor HIGH (5V) o LOW (0V) a un pin digital. `digitalWrite(Pin9, HIGH);`
- `digitalRead(pin)`: Lee el valor de un pin digital especificado, HIGH o LOW. `val = digitalRead(Pin9);`

6.3. Componentes necesarios

- Placa Arduino Mega
 - Cable USB
 - Protoboard
 - 1 LED de 5mm
 - 1 resistencia de 330 Ω
 - 2 resistencias de 10 k Ω
 - 2 interruptores o pulsadores de presión

6.4. Esquema de conexión

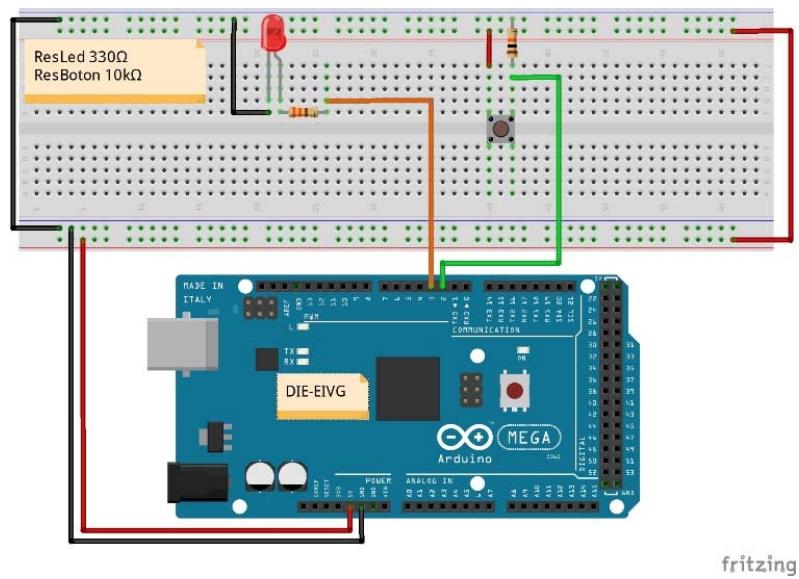


Fig. 31 Esquema de conexión Lección 5, ejercicio 1.

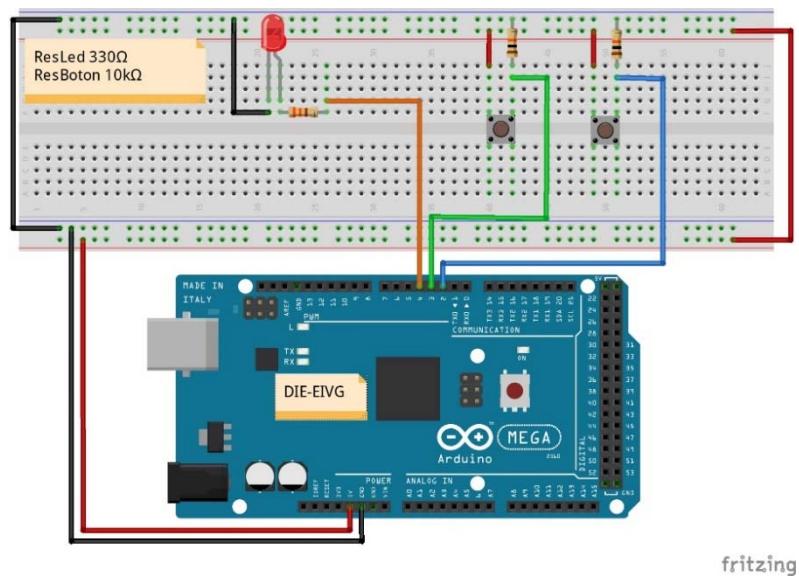


Fig. 32 Esquema de conexión Lección 5, ejercicio 2.

6.5. Códigos de los programas

1. Leccion_5_Entradas_Digitales.ino: Programa que enciende un LED conectado al pin 2 al presionar un pulsador y al soltar el mismo se apaga.
2. Leccion_5_Entradas_Digitales_1.ino: Programa que enciende un led al presionar el pulsador y lo apaga al presionarlo de nuevo.
3. Leccion_5_Entradas_Digitales_2.ino: Programa que enciende un LED conectado al pin 4 con el botón A y lo apaga con el botón B.

6.6. Bibliografía y direcciones de interés

- [1]. <https://www.hwlible.com/pulsador/>
- [2]. <https://programarfacil.com/blog/utilizar-pulsadores-en-arduino/>
- [3]. <http://yomaker.com/leer-un-pulsador-con-arduino/>
- [4]. <https://rufianenlared.com/pulsadores-arduino/>
- [5]. <https://www.luisllamas.es/leer-un-pulsador-con-arduino/>
- [6]. <https://www.geekfactory.mx/tutoriales/tutoriales-arduino/boton-o-pulsador-con-arduino/>
- [7]. <http://manueldelgadocrespo.blogspot.com/p/inpuy.html>
- [8]. <https://aprendiendoarduino.wordpress.com/tag/input-pullup/>

7. Lección 6 Zumbador Activo

7.1. Objetivo de la práctica

Aprender a generar un sonido con un zumbador activo.

7.2. Introducción al componente

Los buzzer activos, en ocasiones denominados zumbadores, son dispositivos que generan un sonido de una frecuencia determinada y fija cuando son conectados a tensión. Los buzzer o zumbadores activos disponen de un oscilador interno, por lo que únicamente tenemos que alimentar el dispositivo para que se produzca el sonido.

Un buzzer activo resulta muy sencillo de conectar y controlar. Pero, tienen la desventaja de que no podremos variar el tono del sonido emitido, por lo que no podremos realizar melodías. El buzzer activo es el que tiene la pegatina y al alimentarlo entre 5V y GND suena a una frecuencia fija.

Los zumbadores activos tienen polaridad normalmente la pata más larga es el positivo en el caso de tener solo dos patillas o también suele ir marcado en la pegatina.



Fig. 33 Patillas de un zumbador activo de montaje.

En el caso de tener tres patillas siempre viene señalado en la placa.



Fig. 34 Esquema de patillas de un zumbador en placa.

Generalmente se pone una resistencia entre el pin digital y el zumbador o buzzer. Esto se hace debido a que la intensidad máxima que puede suministrar por Arduino por cada uno de sus pines es de 40 mA.

Para activarlo todo lo que se necesita es un voltaje de CC, `digitalWrite(PinZumb, HIGH);` y para apagarlo `digitalWrite(PinZumb, LOW);`. Mientras no lo apaguemos el zumbará estará activo.

7.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 zumbador de Activo
- 1 resistencia de $1\text{ k}\Omega$

7.4. Esquema de conexión

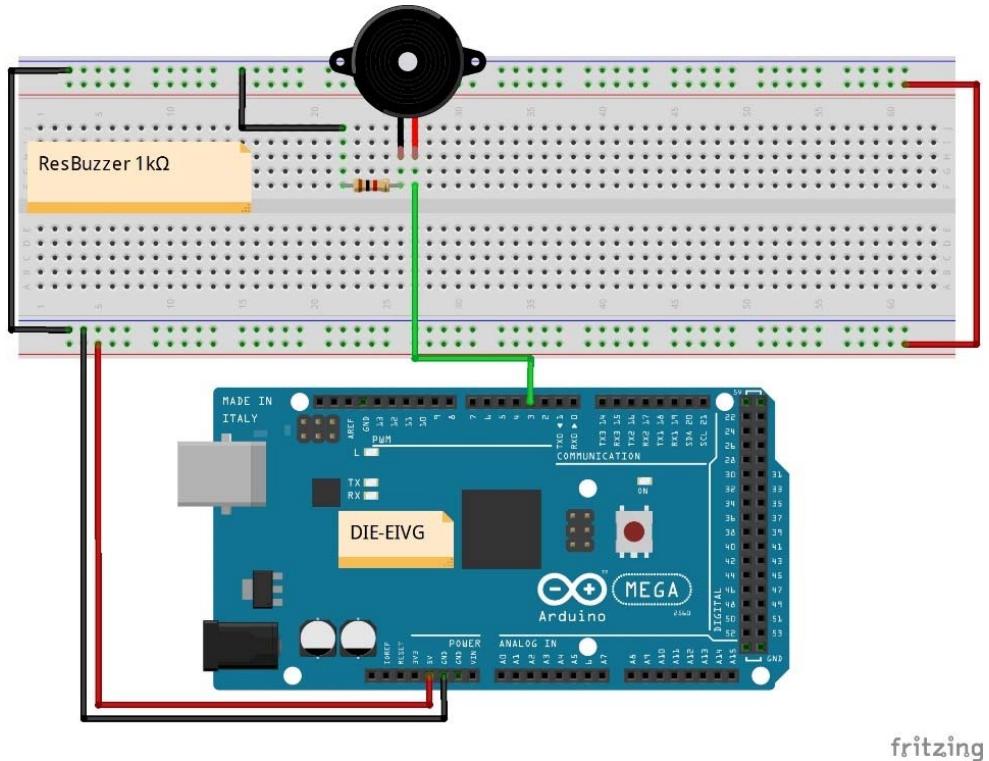


Fig. 35 Esquema de conexión Lección 6.

7.5. Códigos de los programas

1. Leccion_6_Zumbador_Activo.ino: Programa que hace vibrar a un buzzer o zumbador activo a una frecuencia.
2. Leccion_6_Zumbador_Activo_1.ino: Programa que hace vibrar a un buzzer o zumbador activo a dos frecuencias distintas.

7.6. Bibliografía y direcciones de interés

- [1]. <https://programarfácil.com/blog/arduino-blog/buzzer-con-arduino-zumbador/>
- [2]. <https://aprendiendoarduino.wordpress.com/2019/03/03/buzzer-con-arduino/>
- [3]. <https://www.luisllamas.es/arduino-buzzer-activo/>
- [4]. <https://desensores.com/sensores-arduino/proyectos-basicos-con-arduino-para-principiantes/buzzer-activo-arduino/>
- [5]. <http://www.zxs1688.com/info/difference-between-active-buzzer-passive-buzze-22548813.html>
- [6]. <https://respuestas.me/q/zumbador-activo-vs-pasivo-60298491963>

- [7]. <https://soloelectronicos.com/2018/04/21/disene-y-simule-circuitos-electronicos-facilmente-con-tinkercad/>
- [8]. <http://panamahitek.com/generando-sonidos-en-arduino-a-partir-de-pentagramas/>
- [9]. <https://openwebinars.net/blog/tutorial-de-arduino-sonidos-con-arduino/>
- [10]. <https://sites.google.com/site/portafolio201819pamela/actividad-61---arduino-y-bocina>

8. Lección 7 Zumbador Pasivo

8.1. Objetivo de la práctica

Aprender a conectar y programar el módulo de buzzer pasivo, lo prologándolo de tal manera que podamos sacar diferentes tonalidades.

8.2. Introducción al componente

Un buzzer pasivo o un altavoz son dispositivos que permiten convertir una señal eléctrica en una onda de sonido. Estos dispositivos no disponen de electrónica interna, por lo que tenemos que proporcionar una señal eléctrica para conseguir el sonido deseado.

Un buzzer pasivo es un componente de salida, es decir, su funcionamiento consiste recibir una señal de voltaje variable para convertirla en una señal de audio, la tarjeta Arduino Uno tiene la capacidad de generar señales cuadradas a distintas frecuencias en sus pines digitales utilizando la función `tone()`.

<https://www.arduino.cc/reference/en/language/functions/advanced-io/tone/>

```
//Activa un tono de frecuencia determinada en un pin dado
tone(pinBuzzer, frecuencia);
//Detiene el tono en el pin
noTone(pinBuzzer);
```

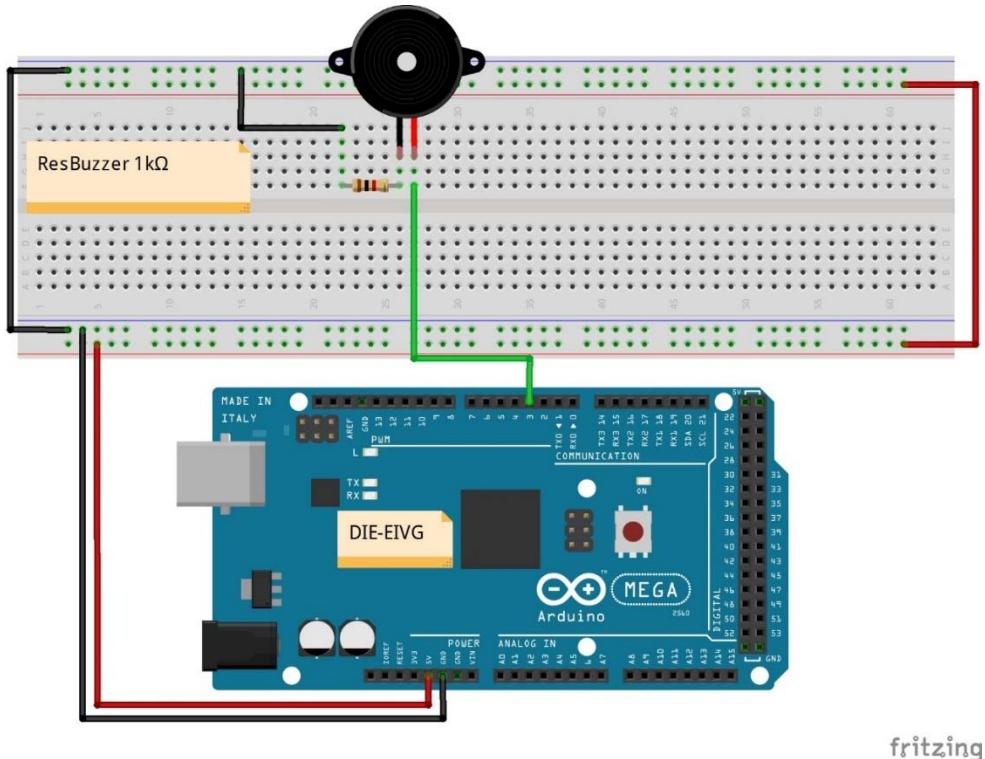
La manera más simple de distinguir un zumbador activo de uno pasivo es conectarlos al pin de 5V de Arduino. Los zumbadores tienen polaridad, es decir, tienen un terminal positivo y otro negativo. Normalmente, el terminal positivo es más largo que el negativo al igual que ocurre con los LEDs. Además, suelen llevar una inscripción que indica cual es el terminal positivo. Hay que recordar que el buzzer activo es el que tiene la pegatina.

La función Tone emplea el Timer 2, por lo que no podremos usar en Arduino mega los pines 9 y 10.

8.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 zumbador pasivo
- 1 resistencia de $1\text{ k}\Omega$

8.4. Esquema de conexión



fritzing

Fig. 36 Esquema de conexión Lección 7.

8.5. Códigos de los programas

1. Leccion_7_Zumbador_Pasivo.ino: Programa que al girar un potenciómetro genera las frecuencias que puede escuchar el oído humano y activa un zumbador pasivo.

8.6. Bibliografía y direcciones de interés

- [1]. <https://programarfácil.com/blog/arduino-blog/buzzer-con-arduino-zumbador/>

- [2]. <https://blog.330ohms.com/2020/06/02/como-conectar-un-buzzer-pasivo-a-arduino/>
- [3]. <https://www.taloselectronics.com/blogs/tutoriales/buzzer-pasivo>
- [4]. <http://cursoarduino.proserquisa.com/2016/10/15/tutorial-29-modulo-buzzer-pasivo-melodia-de-piratas-del-caribe/>
- [5]. <https://aprendiendoarduino.wordpress.com/2019/03/03/buzzer-con-arduino/>

9. Lección 8 Interruptor de Bola de Inclinación

9.1. Objetivo de la práctica

Aprender a conectar y programar un sensor de inclinación para detectar una inclinación.

9.2. Introducción al componente

Un sensor de inclinación es un dispositivo que proporciona una señal digital en caso de que su inclinación supere un umbral. Este tipo de sensor no permite saber el grado de inclinación del dispositivo, simplemente actúa como un sensor que se cierra a partir de una cierta inclinación. Los interruptores de inclinación pueden detectar el movimiento o la orientación simplemente.

El sensor consta de dos bolas, la de abajo hace contacto cuando está vertical, la de arriba tiene la misión de presionar a la de abajo. Cuando se inclina el sensor (a partir de 15° aproximadamente), la bola de abajo deja de hacer contacto con los terminales.

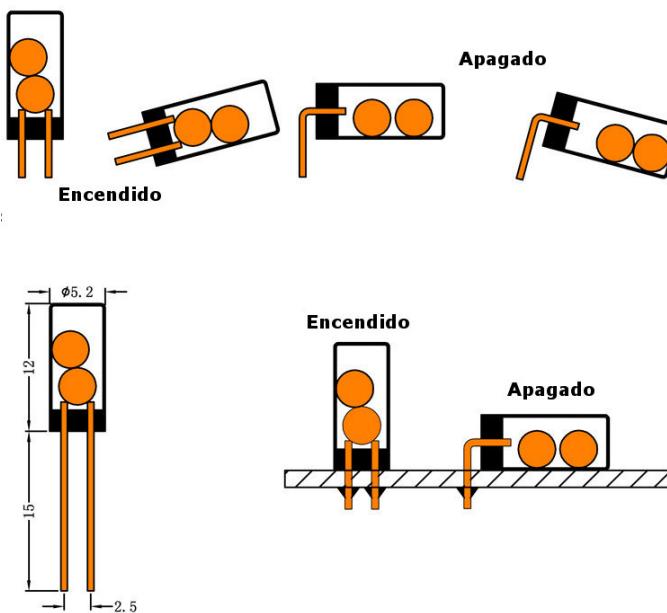


Fig. 37 Esquema de conexión Lección 7.

Cuando hacen contacto permiten el paso de la corriente y cierran el contacto exactamente igual que si fueran un interruptor (se manejan igual) pero que a partir de un cierto ángulo de inclinación dejan de hacer contacto y abren el contacto.

Podemos comprobar si el sensor funciona usando la función de continuidad del polímetro, que hace que éste suene si hay continuidad entre las dos puntas del polímetro. Si la punta negra y la roja están tocando cada uno una parte del circuito y el polímetro suena, quiere decir que ambas partes del circuito están conectadas. Seguirá sonando según vayas inclinándolo. Hasta que llegue a un punto que la inclinación sea mayor a la permitida y dejará de sonar.

9.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 interruptor de inclinación bola
- 1 LED de 5mm
- 1 resistencia de $330\ \Omega$

9.4. Esquema de conexión

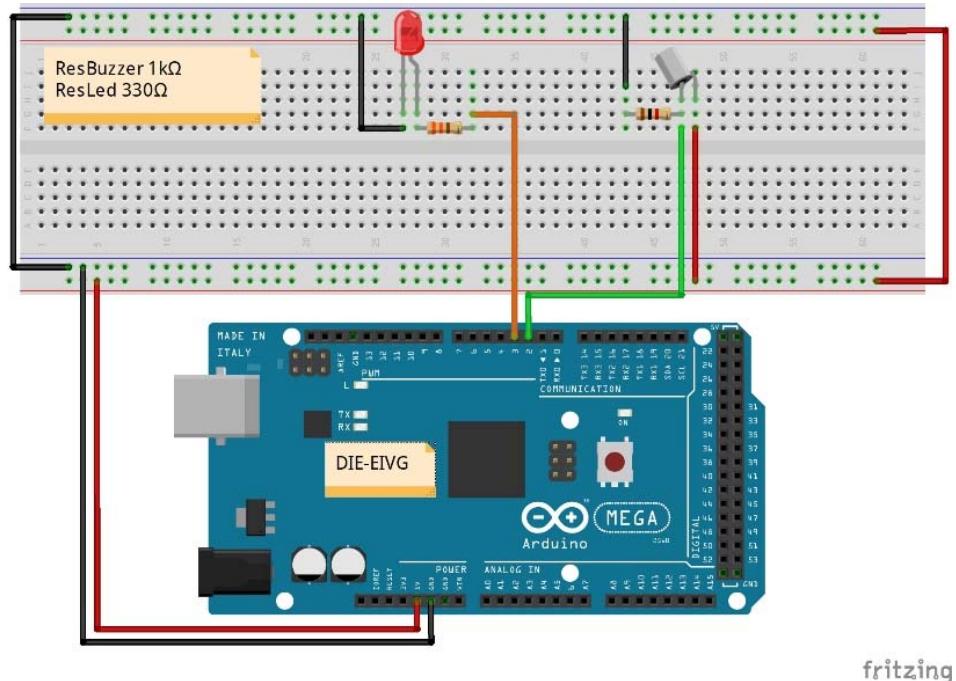


Fig. 38 Esquema de conexión Lección 8.

9.5. Códigos de los programas

1. Leccion_8_Interruptor_de_bola.ino: Programa que utiliza el módulo KY-002 para encender un LED cuando el sensor detecta un golpe o vibración fuerte. El sensor brinda un nivel lógico bajo cuando detecta la vibración.
2. Leccion_8_Interruptor_de_bola_1.ino: Programa que utiliza el módulo KY-002 para encender un LED cuando el sensor detecta un golpe o vibración fuerte. El sensor brinda un nivel lógico bajo cuando detecta la vibración. Y muestra por el Monitor serie si hay alerta por inclinación o está todo correcto.

9.6. Bibliografía y direcciones de interés

- [1]. <https://descubrearduino.com/tilt-switch-o-sensor-de-inclinacion-que-es-y-para-que-sirve/>
- [2]. <http://kio4.com/arduino/40interruptorinclinacion.htm>
- [3]. <https://www.prometec.net/tilt-switch/>
- [4]. <https://www.luisllamas.es/medir-inclinacion-con-arduino-y-sensor-tilt-sw-520d/>
- [5]. <https://rufianenlared.com/tilt-switch/>
- [6]. <https://eloctavobit.com/arduino/medir-la-inclinacion-con-arduino-y-el-sensor-sw-520d/>

10. Lección 9 Servo

10.1. Objetivo de la práctica

Aprender el funcionamiento y características de un servomotor y como programarlos desde Arduino.

10.2. Introducción al componente

Los servos son motores de corriente continua, pero en lugar de diseñarse para obtener un giro continuo que podamos aprovechar (para mover una rueda, por ejemplo), se diseñan para que se muevan un ángulo fijo en respuesta a una señal de control, y se mantengan fijos en esa posición. Tienen la capacidad de ubicar su eje en una posición o ángulo determinado, internamente tiene una caja reductora la cual le aumenta el torque y reduce la velocidad,

En un servo indicamos directamente el ángulo deseado y el servo se encarga de posicionares en este ángulo. Típicamente los servos disponen de un rango de movimiento de entre 0 a 180°. El ángulo de giro, en este caso nos permite hacer un barrido entre -90° y 90°. Aunque el servo puede moverse con una resolución de más de 1 grado, este es el máximo de resolución que vamos a conseguir debido a la limitación de la señal PWM que es capaz de generar Arduino.

Proporcionan un alto par y grado de precisión (incluso décimas de grado). Por contra, las velocidades de giro son pequeñas frente a los motores de corriente continua.

Estos motores funcionan con una señal PWM, con un pulso de trabajo entre 1 ms y 2 ms y con un periodo de 20 ms (50 Hz). Este dato nos indica la velocidad máxima a la que podemos mover el servomotor con Arduino. Solo podremos cambiar de posición cada 20 ms. Esto dependerá del tipo y marca de nuestro servo.

En el mercado existen diferentes modelos de servomotores, la principal diferencia entre ellos es el par. Este punto es bueno tenerlo claro para elegir el servomotor adecuado. Es mejor elegir un servo con par superior al que requerimos, pues el consumo de corriente es proporcional a la carga. En cambio, si sometemos un servomotor a cargas superiores a su par, corremos el riesgo de malograr tanto la parte mecánica (engranes) y eléctrica del servo, incluso podemos generar ruido en la fuente.



Fig. 39 Tipos de servos.

Para su funcionamiento necesitaremos tener instalada la librería servo. Servo es una librería estándar en Arduino. Eso quiere decir que viene incluida cuando instaláis el IDE. Al comienzo del programa incluiremos la librería y crearemos el objeto servo.

```
#include <Servo.h>          // Incluye librería de Servo  
Servo myServo;              // Crea objeto servo
```

Para referirse a la librería servo, es necesario crear un nombre de esta librería en una variable (`myServo`). Esto se conoce con el nombre de un objeto. Cuando se hace esto, se crea un nombre único que tendrá todas las funciones y capacidades que la librería servo posee. A partir de este punto en el programa, cada vez que se utilice el nombre `myServo`,

se podrá usar todas esas funciones y capacidades de la librería servo.

Principales las funciones de la librería Servo:

- `attach(Pin):` Establece el pin indicado en la variable servo.
`servo.attach(2);`
- `attach(Pin,min,max):` Establece el pin indicado en la variable servo, considerando min el ancho de pulso para la posición 0° y max el ancho de pulso para 180°. `servo.attach(2,900,2100);`
- `write(angulo):` Envía la señal correspondiente al servo para que se ubique en el ángulo indicado, ángulo es un valor entre 0 y 180°. `servo.write(45);`
- `writeMicroseconds(tiempo):` Envía al servo el ancho de pulso=tiempo en microsegundos. `servo.writeMicroseconds(1500);`
- `read ():` Lee la posición actual del servo, devuelve un valor entre 0 y 180. `angulo=read();`
- `attached(Pin):` Verifica si la variable servo está unido al pin indicado, devuelve true o false. `if(attached(2))`
- `detach(pin):` Separa la variable Servo del pin indicado. `servo.detach(2);`

Dentro de la función `setup()` es necesario decirle a Arduino que pin está unido al Servomotor. En el ejercicio del potenciómetro, dentro de la función `loop()`, se procede a leer la entrada analógica (A0) donde se conecta el terminal central del potenciómetro y a continuación mostrar este valor en el monitor serie y en la pantalla de un ordenador.

Para crear valores a partir de la entrada analógica que se puedan usar con el servomotor, se puede hacer de una forma muy fácil usando la instrucción `map()`. Esta instrucción trabaja con escalas de números. En este caso cambia la escala de valores entre 0-1023 a valores entre 0-179. Necesita cinco argumentos: el número que debe de ser escalado (en este caso dentro de la variable `valorPot`), el valor mínimo de la entrada (0), el valor máximo de la entrada (1023), el valor mínimo de la salida (0) y el valor máximo de la salida (179). El resultado se guarda dentro de la variable `valorAng`

El Servo tiene tres cables, marrón es el cable a masa y debe conectarse a GND, el rojo es el cable de corriente y debe conectarse al puerto de 5v y el naranja es el cable de señal.

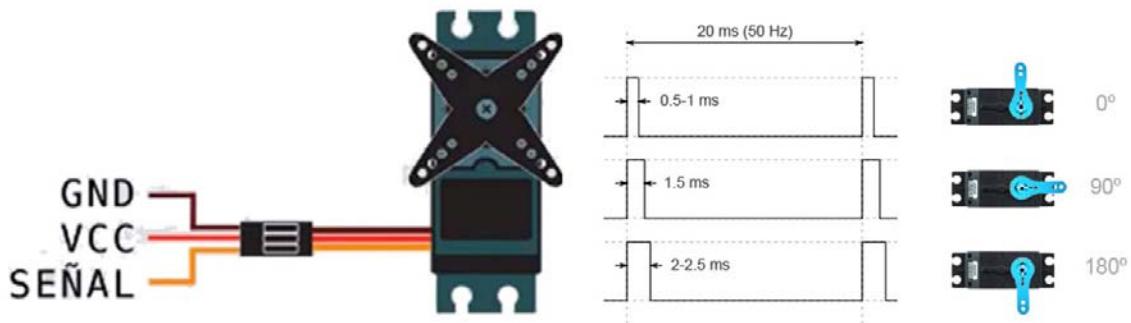


Fig. 40 Cableado de un servo.

La señal o dato que hay que enviarle al servo es una señal de PWM donde el tiempo en alto es equivalente al ángulo o posición del servo. Estos valores pueden variar y van desde 0.5 a 1 milisegundo para la posición 0° y 2 a 2.4 milisegundos para la posición de 180°, el periodo de la señal debe ser cercano a 20 milisegundos.

Señales PWM en Arduino.

La mayoría de los automatismos (y Arduino no es una excepción) no son capaces de proporcionar una auténtica salida analógica. Para salvar esta limitación y simular una salida analógica la mayoría de los automatismos emplean un "truco", que consiste en activar una salida digital durante un tiempo y mantenerla apagada durante el resto. El promedio de la tensión de salida, a lo largo del tiempo, será igual al valor analógico deseado.

Existe más de una forma de hacer esta aproximación. Una de las más sencillas, y por ello muy empleada en automatización, es la modulación de ancho de pulso (PWM). En esta modulación se mantiene constante la frecuencia (es decir, el tiempo entre disparo de pulsos), mientras que se hace variar la anchura del pulso. Variando en ancho del pulso, podemos obtener una señal promedio equivalente.

Para la primera señal, cuyo ancho de pulso es del 10%, nos daría una señal analógica de 0.5v (el 10% de 5V), para la segunda una señal de 1.5v, para la tercera una señal de 2.5v y para la cuarta señal obtendríamos una señal equivalente de 4.5v.

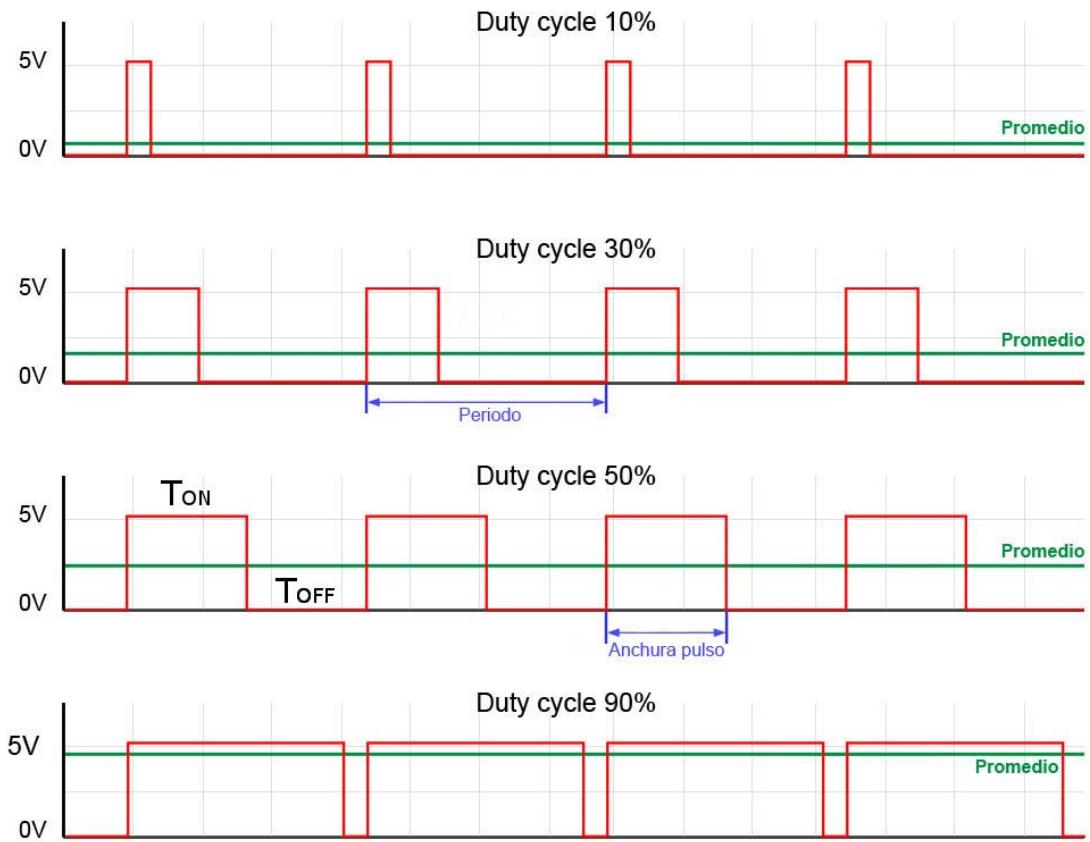


Fig. 41 Ciclos de trabajo para una señal PWM.

La proporción de tiempo que está encendida la señal, respecto al total del ciclo, se denomina "Duty cycle", ciclo de trabajo, y generalmente se expresa en tanto por ciento.

Hay que destacar tres valores de tiempo:

- TON: tiempo que la señal se mantiene en estado alto
- TOFF: tiempo que la señal se mantiene en estado bajo
- T: Periodo de la señal.

En la señal anterior se cumple: $T = T_{ON} + T_{OFF}$. Es lógico, el tiempo total (T) es igual al tiempo que está en estado alto (TON) más el tiempo que está en estado bajo (TOFF)

El ciclo de trabajo o duty cycle de la señal es uno de los conceptos más importantes de una señal PWM. Este se representa mediante la letra D y se define como la razón entre el tiempo en estado alto (TON) y el periodo de la señal (T):

$$D = \frac{T_{ON}}{T} 100\%$$

Se multiplica por 100 porque se suele expresar en tanto por ciento y representa el tiempo en el que la señal está en estado alto dentro de un periodo.

El voltaje promedio se calcula mediante la siguiente expresión:

$$V_{promed} = (V_H - V_L) \frac{D}{100}$$

Donde:

- V_H es el voltaje en estado alto.
- V_L es el voltaje en estado bajo.

Como en las placas Arduino el voltaje en estado bajo es normalmente 0V, la expresión se simplifica de la siguiente manera:

$$V_{promed} = V_{CC} \frac{D}{100}$$

Donde:

V_{CC} es el voltaje que manejan los pines digitales de la placa, 5V para Arduino Mega.

`analogWrite(pin, valor)`: Escribe un valor pseudo analógico usando modulación por ancho de pulso (PWM) a un pin de salida. **pin**: Es el pin en el cual se quiere generar la señal PWM. **valor**: El ciclo de trabajo deseado comprendido entre 0 (siempre apagado) y 255 (siempre encendido).

10.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 servo (SG90)
- 1 potenciómetro 10 k
- 1 LED de 5mm
- 1 resistencia de 330 Ω
- 1 interruptor o pulsador de presión

10.4. Esquema de conexión

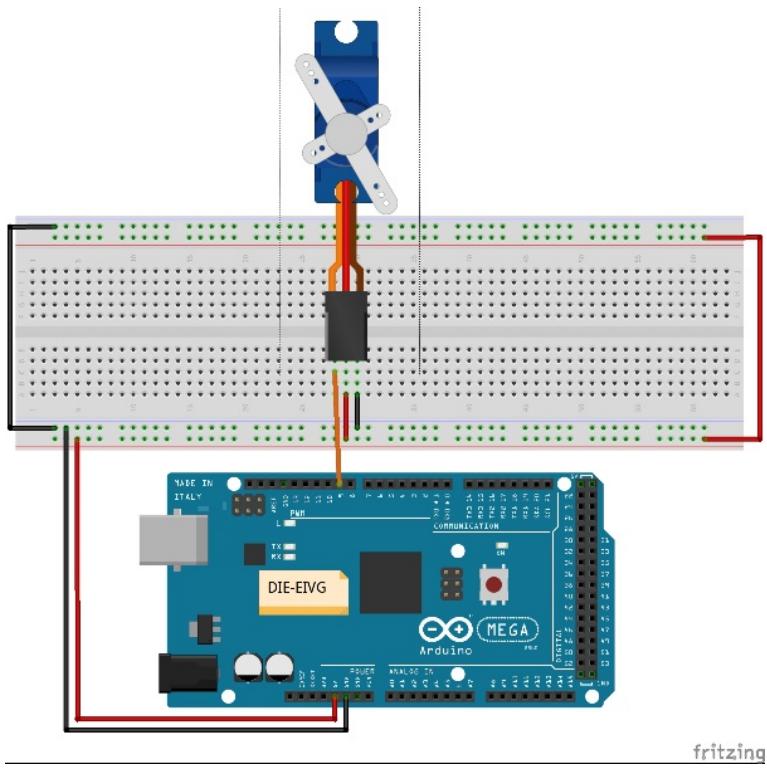


Fig. 42 Esquema de conexión Lección 9, ejercicio 1.

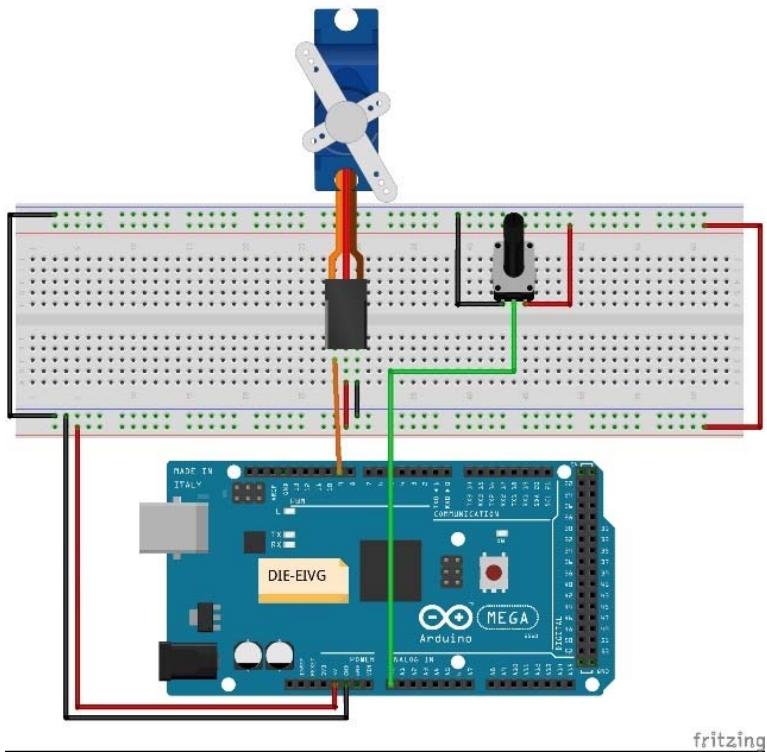


Fig. 43 Esquema de conexión Lección 9, ejercicio 2

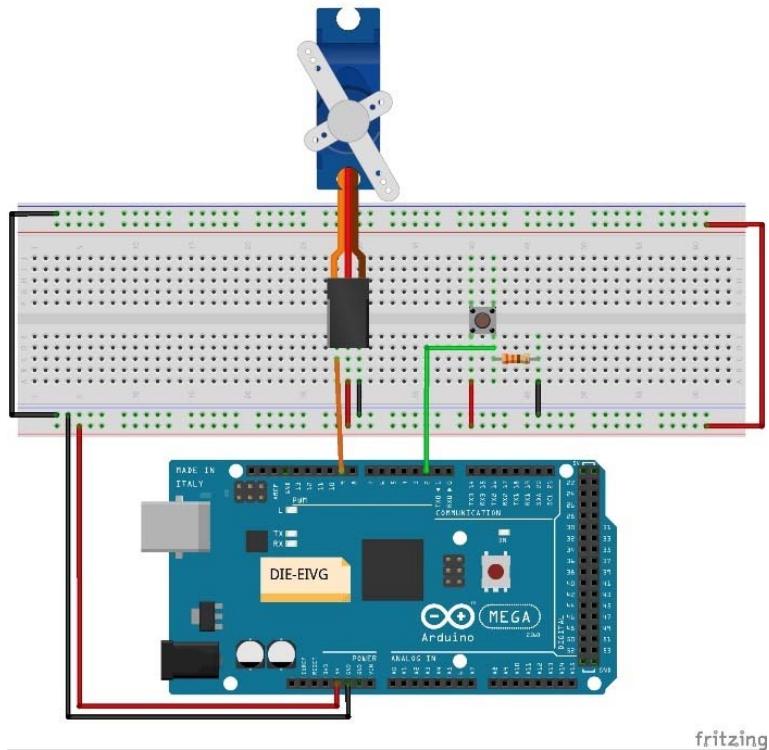


Fig. 44 Esquema de conexión Lección 9, ejercicio 3

10.5. Códigos de los programas

1. Leccion_9_Servo.ino: Programa que controla el servo desde el Monitor Serie. Donde introducimos el valor del ángulo al que queremos que vaya. Desde el Monitor Serie se pregunta la posición y se manda un valor entre 0 y 180, y lo convierte a valor de ángulo para mover al servo hacia el mismo.
2. Leccion_9_Servo_1.ino: Programa que lee el valor del potenciómetro conectado a entrada A0 y lo convierte a valor de ángulo para mover al servo hacia el mismo.
3. Leccion_9_Servo_2.ino: Programa que varía la velocidad de un servo según el número de la pulsación, p.e. la activación de un limpiaparabrisas. Hay tres velocidades de barrido, cada pulsación sube de velocidad y la cuarta pulsación lo para. Muestra en el Monitor Serie el número de la pulsación (0, 1, 2 o 3), que hayamos hecho el pulsador.

10.6. Bibliografía y direcciones de interés

- [1]. <https://www.luisllamas.es/controlar-un-servo-con-arduino/>
- [2]. <https://programarfácil.com/tutoriales/fragmentos/servomotor-con-arduino/>

- [3]. <https://aprendiendoarduino.wordpress.com/tag/servomotor/>
- [4]. <https://www.prometec.net/servos/>
- [5]. https://www.naylampmechatronics.com/blog/33_Tutorial-uso-de-servomotores-con-arduino-.html

11. Lección 10 Módulo Sensor Ultrasónico HC-SR04

11.1. Objetivo de la práctica

Conocer de forma detallada las características del sensor HC-SR04, como calibrar el sensor, como conectarlo y como utilizarlo con Arduino. Este sensor es ideal para todo tipo de proyectos que necesitan medidas de distancia, evitando los obstáculos como ejemplos. El HC-SR04 es barato y fácil de usar ya que vamos a usar una biblioteca diseñada específicamente para estos sensores.

11.2. Introducción al componente



Fig. 45 Conexiones del Sensor Ultrasónico HC-SR04

Este módulo tiene cuatro pines:

- VCC: + 5V de Corriente Continua
- Trig: El pin trigger recibe un pulso de habilitación de parte del microcontrolador, mediante el cual se le indica al módulo que comience a realizar la medición de distancia.

- Echo: En el pin echo el sensor devuelve al microcontrolador un pulso cuyo ancho es proporcional al tiempo que tarda el sonido en viajar del transductor al obstáculo y luego de vuelta al módulo.
- GND: GND

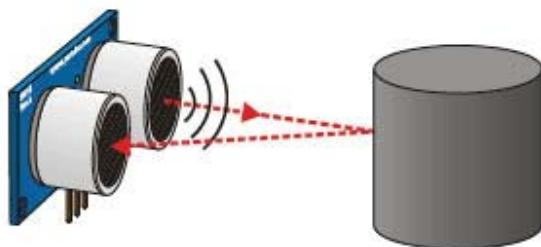
El sensor de distancia ultrasónico HC-SR04 ofrece un rango de mediciones de entre 2cm y 400cm, la precisión puede alcanzar los 3mm. Los módulos incluyen el transmisor ultrasónico, el receptor y el circuito de control.

El sensor se basa simplemente en medir el tiempo entre el envío y la recepción de un pulso sonoro. Sabemos que la velocidad del sonido es 343 m/s en condiciones de temperatura 20 °C, 50% de humedad, presión atmosférica a nivel del mar. Transformando unidades resulta:

$$343 \frac{m}{s} \cdot 100 \frac{cm}{m} \frac{1}{1000000} \frac{s}{\mu s} = \frac{1}{29,2} \frac{cm}{\mu s}$$

Es decir, el sonido tarda 29,2 microsegundos en recorrer un centímetro. Por tanto, podemos obtener la distancia a partir del tiempo entre la emisión y recepción del pulso mediante la siguiente ecuación.

$$\text{Distancia (cm)} = \frac{\text{Tiempo}(\mu s)}{29,2 \cdot 2} = \frac{\text{Tiempo}(\mu s)}{58,4}$$



$$\text{Tiempo} = 2 * (\text{Distancia} / \text{Velocidad})$$

$$\text{Distancia} = \text{Tiempo} \cdot \text{Velocidad} / 2$$

Fig. 46 Esquema de conexión Lección 10, ejercicio 1 y 2

El motivo de dividir por dos el tiempo es porque hemos medido el tiempo que tarda el pulso en ir y volver, por lo que la distancia recorrida por el pulso es el doble de la que queremos medir.

El sensor HC-SR04 es un módulo que incorpora un par de transductores de ultrasonido que se utilizan de manera conjunta para determinar la distancia del sensor con un objeto colocado enfrente de este. Un transductor emite una “ráfaga” de ultrasonido y el otro capta el rebote de dicha onda. El tiempo que tarda la onda sonora en ir y regresar

a un objeto puede utilizarse para conocer la distancia que existe entre el origen del sonido y el objeto.

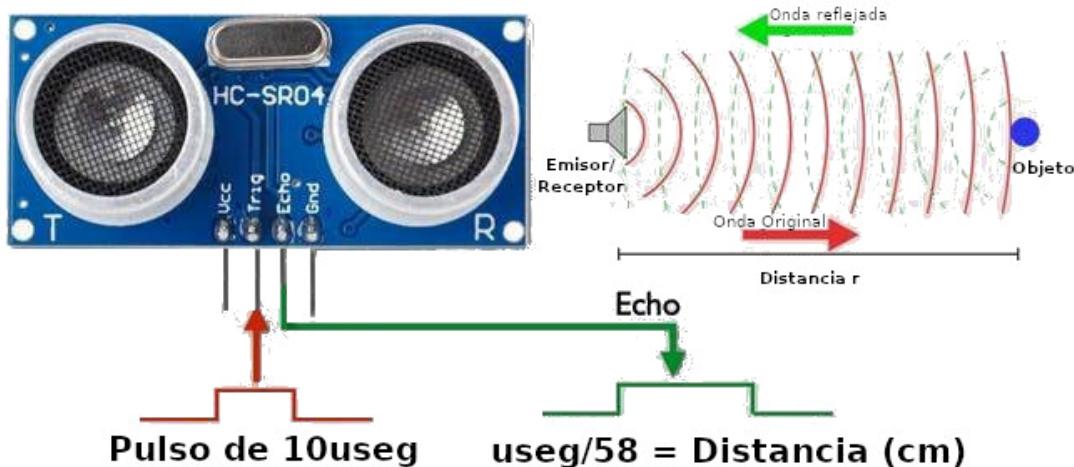


Fig. 47 Esquema de conexión Lección 10, ejercicio 1 y 2

Programación:

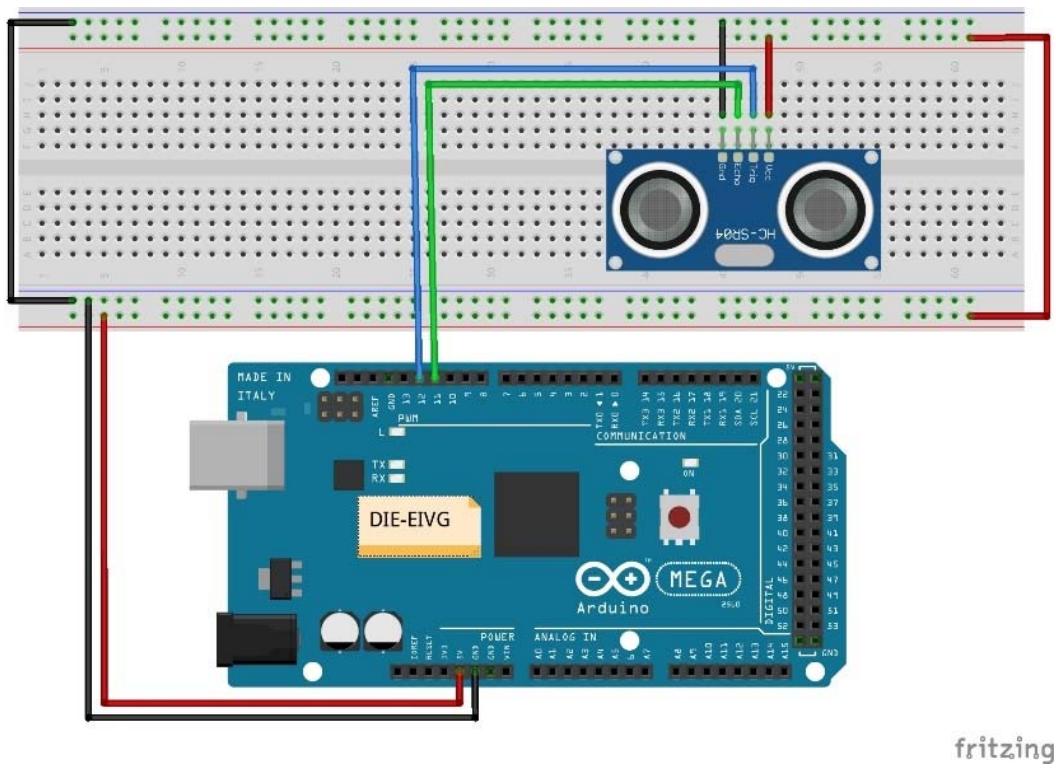
```
pinMode(pinTrig, OUTPUT); // trigger como salida
pinMode(pinEco, INPUT); // echo como entrada
// Recepcion del eco de respuesta
valDuracion = pulseIn(pinEco, HIGH); // con la función pulseIn se espera
un pulso alto en Echo
// Calculo de la distancia efectiva
valDistancia = valDuracion / 58.2; // Distancia medida en centímetros
```

Como se observa, el HC-SR04 genera un pulso en el pin marcado como “echo” cuya duración es proporcional a la distancia medida por el sensor. Para obtener la distancia en centímetros, solamente debemos dividir el tiempo en microsegundos entre 58,2 o para obtener la distancia en centímetros (148 para pulgadas).

11.3. Componentes necesarios

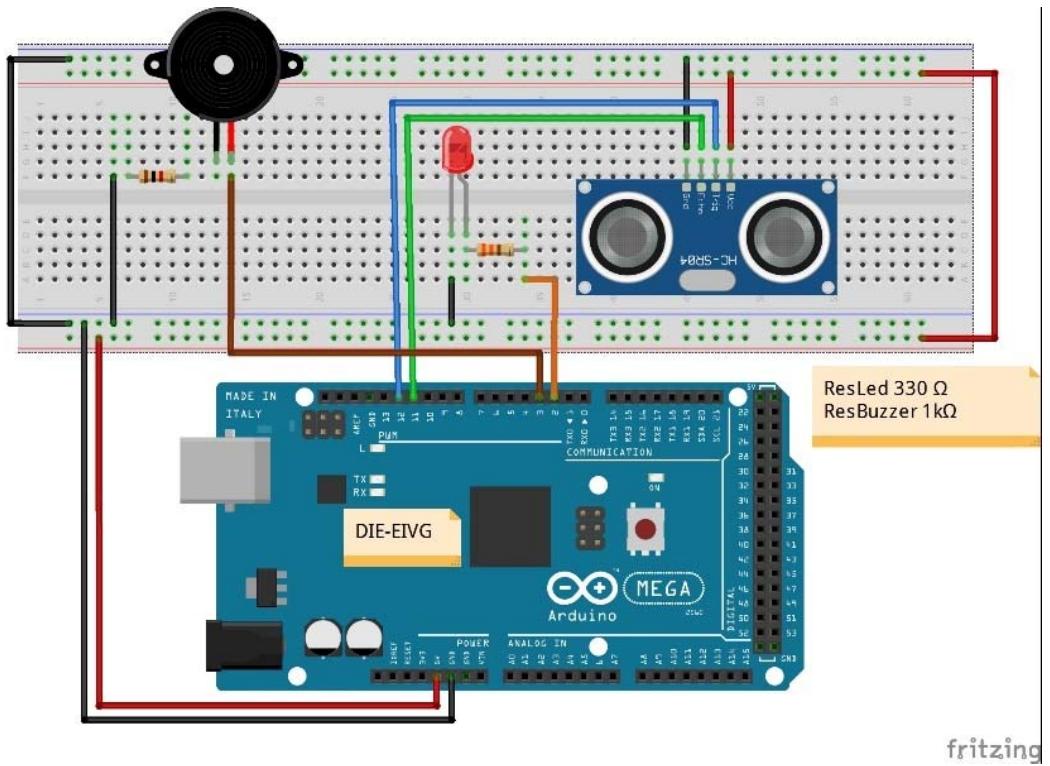
- Placa Arduino Mega
- Cable USB
- Protoboard
- Módulo de sensor ultrasónico
- 3 LEDs de 5mm
- 3 resistencias de 330 Ω

11.4. Esquema de conexión



fritzing

Fig. 48 Esquema de conexión Lección 10, ejercicio 1 y 2



ResLed 330 Ω
ResBuzzer 1kΩ

fritzing

Fig. 49 Esquema de conexión Lección 10, ejercicio 3

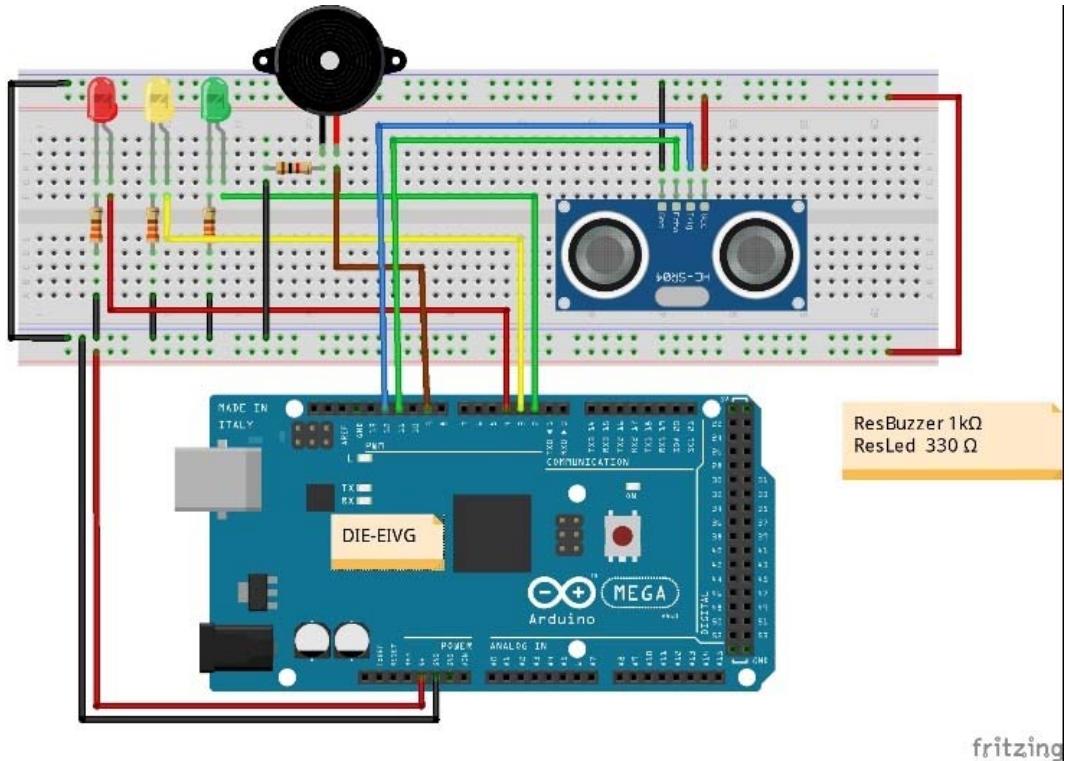


Fig. 50 Esquema de conexión Lección 10, ejercicio 4

11.5. Códigos de los programas

1. Leccion_10_Modulo_Sensor_Ultrasonico.ino: Programa que muestra en el Monitor Serial el valor de distancia leído por el sensor ultrasónico.
2. Leccion_10_Modulo_Sensor_Ultrasonico_0.ino: Programa que muestra en el Monitor Serial el valor de distancia leído por el sensor ultrasónico HC-SR04. Usando la librería SR04.h
3. Leccion_10_Modulo_Sensor_Ultrasonico_1.ino: Programa que envía mediante el Monitor Serial el valor de distancia leído por el sensor ultrasónico HC-SR04 y enciende y apaga un LED y un buzzer activo dentro del rango de 0 a 20 cms.
4. Leccion_10_Modulo_Sensor_Ultrasonico_2.ino: Programa que envía mediante el Monitor Serial el valor de distancia leído por el sensor ultrasónico HC-SR04 y nos permite visualizar si estamos cerca o lejos de un obstáculo. Con 3 LEDs (verde, amarillo y rojo) conseguimos determinar si estamos lejos, cerca o en zona de peligro.

11.6. Bibliografía y direcciones de interés

- [1]. https://www.naylampmechatronics.com/blog/10_Tutorial-de-Arduino-y-sensor-ultras%C3%B3nico-HC-S.html
- [2]. <https://www.luisllamas.es/medir-distancia-con-arduino-y-sensor-de-ultrasonidos-hc-sr04/>
- [3]. <https://www.geekfactory.mx/tutoriales/tutoriales-arduino/sensor-ultrasonico-hc-sr04-y-arduino/>
- [4]. <https://tecnopatafisica.com/tecnologia/teoria/robotica/27-hCSR04>
- [5]. <https://programarfacil.com/blog/arduino-blog/sensor-ultrasonico-arduino-medir-distancia/>
- [6]. <https://www.web-robotica.com/arduino/como-usar-el-sensor-de-distancia-ultrasonico-hc-sr04-con-arduino>
- [7]. <https://www.zonamaker.com/arduino/modulos-sensores-y-shields/ultrasonido-hc-sr04>
- [8]. <https://smelpro.com/blog/sensor-hc-sr04/>

12. Lección 11 Teclado de Membrana

12.1. Objetivo de la práctica

Aprender a integrar un teclado de membrana con una placa Arduino MEGA2560 R3 para que el MEGA2560 R3 pueda leer las teclas que está siendo presionado por un usuario. Usar otra librería externa: KeyPad.

12.2. Introducción al componente

Un teclado matricial es un conjunto de botones conectados en filas y columnas, de modo que se pueden leer varios botones con el mínimo número de pines requeridos. Un teclado matricial 4x4 solamente ocupa 4 líneas de un puerto para las filas y otras 4 líneas para las columnas, de este modo se pueden leer 16 teclas utilizando solamente 8 líneas de un microcontrolador. Si asumimos que todas las columnas y filas inicialmente están en alto (1 lógico), la pulsación de un botón se puede detectar al poner cada fila a en bajo (0 lógico) y chequear cada columna en busca de un cero, si ninguna columna está en bajo entonces el 0 de las filas se recorre hacia la siguiente y así secuencialmente.

Este tipo de teclado consiste en 3 membranas, de allí recibe su nombre. Dos membranas contienen material conductor de electricidad, y en medio de ellas hay una

membrana no conductora que sirve para separarlas, la cual contiene un agujero donde va cada botón. Bajo condiciones normales, el interruptor se encuentra abierto porque la corriente no puede circular, pero cuando se presiona una tecla, la membrana de arriba y la de abajo hacen contacto a través del pequeño agujero que hay en la membrana del medio, permitiendo que la corriente circule y se envíe la información.

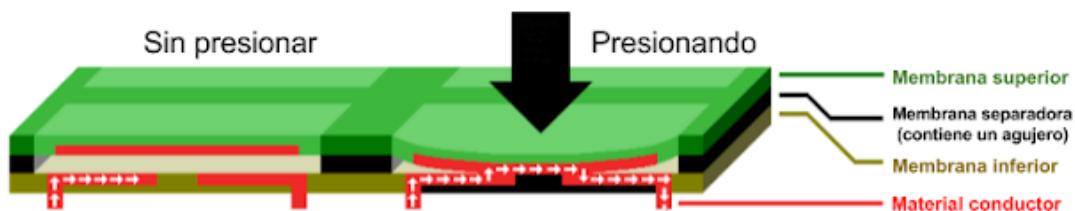


Fig. 51 Membranas de un teclado de membranas

Que sea un teclado matricial, significa que usa el esquema de una matriz para agrupar los pulsadores, es decir, los organiza en filas y columnas, haciendo que cada pulsador tenga una posición única dentro de la matriz. Esto tiene la ventaja de que se necesiten menos pines del Arduino a la hora de conectar el teclado, pues con este sistema se utiliza un pin por cada fila y un pin por cada columna. En esta práctica utilizaremos un teclado 4x4, esto quiere decir que se utilizarán 8 pines del Arduino cuando lo vayamos a conectar.

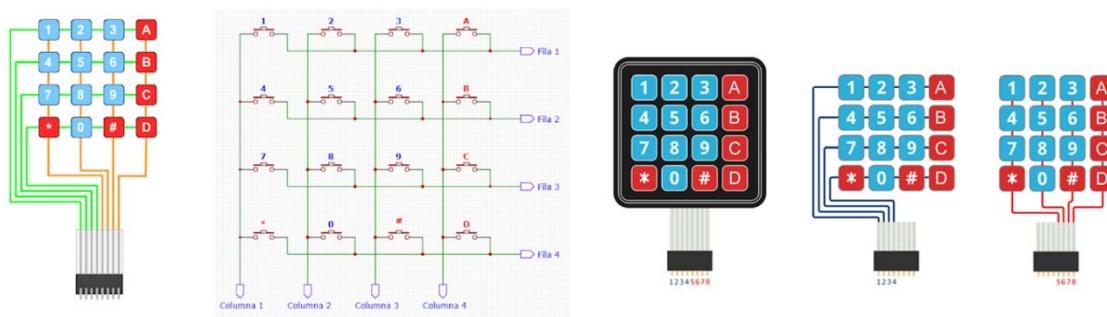


Fig. 52 Conexiones de un teclado de membranas

Librería Keypad de Arduino:

1. Incluimos la librería en nuestro proyecto: `#include <Keypad.h>`
2. Creamos el objeto de la instancia Keypad: `Keypad teclado = Keypad(makeKeymap(keys), pinesFilas, pinesColumnas, FILAS, COLUMNAS); // crea objeto`

3. Para leer cual tecla fue presionada usamos la siguiente instrucción: TECLA
`TECLA = teclado.getKey(); // obtiene tecla presionada y asigna a variable`

12.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 módulo teclado matricial

12.4. Esquema de conexión

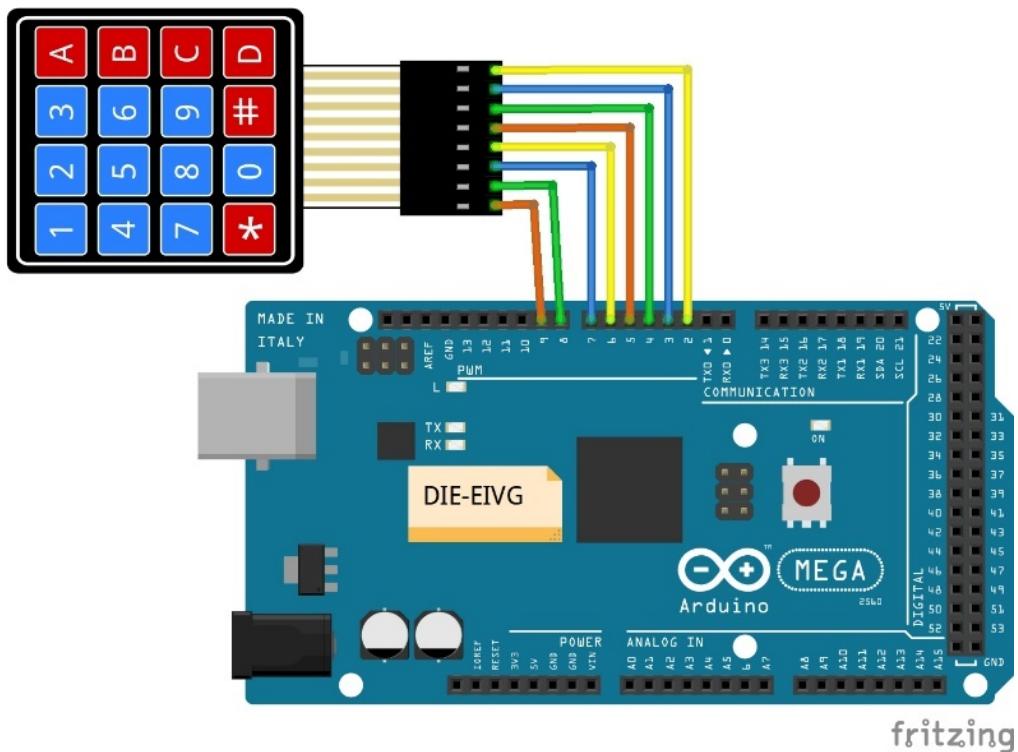


Fig. 53 Esquema de conexión Lección 11.

12.5. Códigos de los programas

1. Leccion_11_Teclado_de_Membrana.ino: Programa que permite conectar un teclado matricial de membrana de 4x4 y mostrar la tecla presionada en el Monitor Serie

12.6. Bibliografía y direcciones de interés

- [1]. <https://www.luisllamas.es/arduino-teclado-matricial/>
- [2]. <https://www.prometec.net/teclados-matriciales/>
- [3]. <http://arduparatodos.blogspot.com/2017/12/teclado-matricial-con-arduino-varios.html>
- [4]. <https://aprendiendoarduino.wordpress.com/tag/teclado-matricial/>
- [5]. <https://controlautomaticoeducacion.com/arduino/teclado-matricial-keypad/>
- [6]. <https://hetpro-store.com/TUTORIALES/teclado-matricial-con-arduino/>
- [7]. <https://electrocrea.com/blogs/tutoriales/18188479-teclado-matricial-4x4>

13. Lección 12 Sensor de Humedad y Temperatura DHT11

13.1. Objetivo de la práctica

Con esta práctica vamos a aprender cómo usar un Sensor de humedad y temperatura DHT11. Mostrar como leerlos utilizando la librería de control DHT.

13.2. Introducción al componente

El DHT11 y el DHT22 (o AM2302) son dos modelos de una misma familia de sensores, que permiten realizar la medición simultánea de temperatura y humedad.

Estos sensores disponen de un procesador interno que realiza el proceso de medición, proporcionando la medición mediante una señal digital, por lo que resulta muy sencillo obtener la medición desde un microprocesador como Arduino. Utiliza un sensor capacitivo de humedad y un termistor para medir el aire circundante y solo un pin para la lectura de los datos. Tal vez la desventaja de estos es la velocidad de las lecturas y el tiempo que hay que esperar para tomar nuevas lecturas (nueva lectura después de 2 segundos), pero esto no es tan importante puesto que la Temperatura y Humedad son variables que no cambian muy rápido en el tiempo.

Ambos sensores presentan un encapsulado de plástico similar. Podemos distinguir ambos modelos por el color del mismo. El DHT11 presenta una carcasa azul, mientras que en el caso del sensor DHT22 el exterior es blanco.

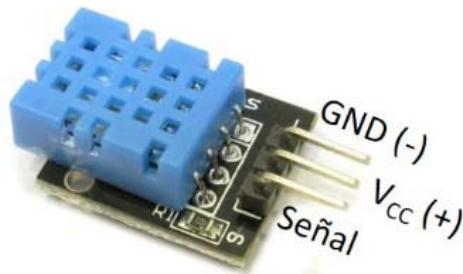


Fig. 54 Sensor de Humedad y Temperatura DHT11.

Las características del DHT11 son:

- Muy barato
- Funciona con 3,3 y 5V de alimentación
- Rango de temperatura: de 0º a 50º con 5% de precisión (pero solo mide por grados, no fracciones)
- Rango de humedad: de 20% al 80% con 5% de precisión
- 1 Muestra por segundo
- Bajo consumo
- Devuelva la medida en ºC

Los pines del DHT11 son:

- GND: conexión con tierra
- Vcc: alimentación 5 V
- Señal: transmisión de datos

Conectar el sensor es sencillo, simplemente alimentamos desde Arduino al sensor a través de los pines GND y Vcc del mismo. Por otro lado, conectamos la salida Output a una entrada digital de Arduino.

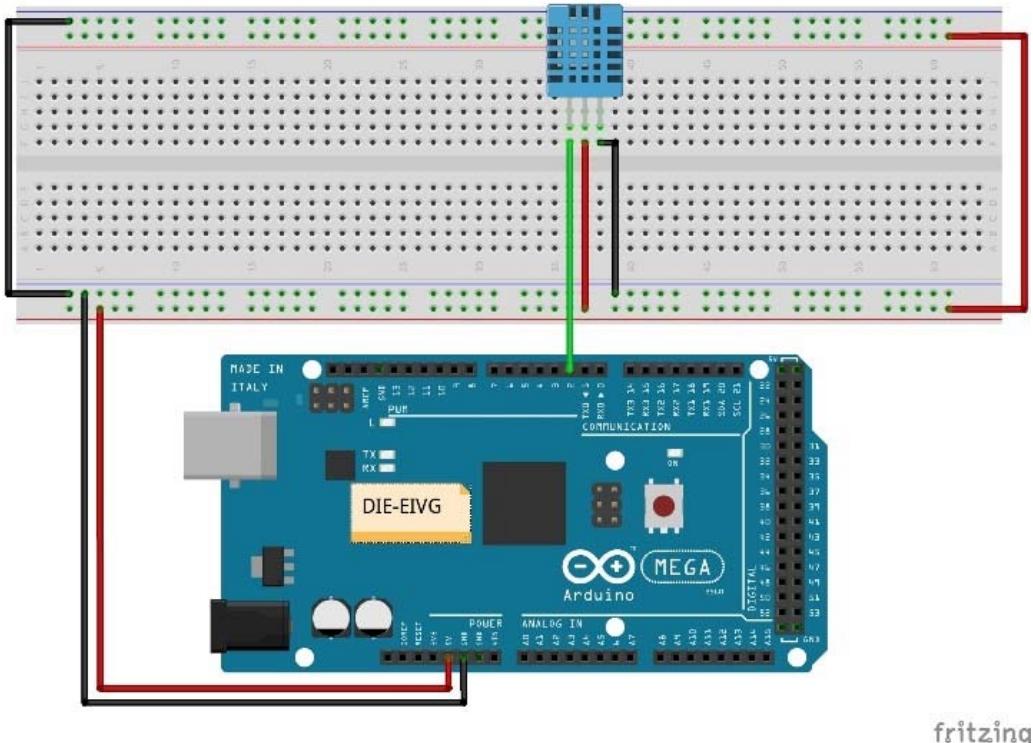
Librería DHT

- Importamos la librería DHT: #include <DHT11.h>
- Creamos el objeto DHT dht(pinSensor, DHT11);
- En el setup inicializamos el sensor dht.begin();
- En el loop obtenemos los valores:
 - valTemperatura = dht.readTemperature
 - valHumedad = dht.readHumidity();

13.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 módulo de humedad y temperatura DHT11

13.4. Esquema de conexión



fritzing

Fig. 55 Esquema de conexión Lección 12.

13.5. Códigos de los programas

1. Leccion_12_Sensor_DHT11.ino: Programa que envía al Monitor Serie el valor de la Temperatura y Humedad leído por el sensor DHT11. También calcula el índice de calor temperatura aparente, sensación térmica.

13.6. Bibliografía y direcciones de interés

- [1]. <https://www.luisllamas.es/arduino-dht11-dht22/>
- [2]. <https://programarfacil.com/blog/arduino-blog/sensor-dht11-temperatura-humedad-arduino/>
- [3]. <https://www.prometec.net/sensores-dht11/>

- [4]. <https://create.arduino.cc/projecthub/agriculturaelectronica/arduino-con-sensor-de-temperatura-y-humedad-dht11-f74a92>
- [5]. https://www.naylampmechatronics.com/blog/40_Tutorial-sensor-de-temperatura-y-humedad-DHT1.html
- [6]. <https://www.geekfactory.mx/tutoriales/tutoriales-arduino/dht11-con-arduino/>
- [7]. <https://www.instructables.com/Sensor-de-Temperatura-y-Humedad-DHT11-y-Arduino/>
- [8]. <https://www.hwlibre.com/dht11/>
- [9]. <https://aprendiendoarduino.wordpress.com/tag/dht11/>
- [10]. <https://www.instructables.com/Set-Up-a-DHT11-Temperature-and-Humidity-Sensor-Wit/>
- [11]. <https://pokelectro.wordpress.com/2017/07/21/sensor-de-temperatura-y-humedad/>
- [12]. <https://create.arduino.cc/projecthub/arcaegecengiz/using-dht11-b0f365>

14. Lección 13 Módulo Joystick Analógico

14.1. Objetivo de la práctica

Aprender a usar el módulo de joystick analógico.

14.2. Introducción al componente

Un joystick es una palanca que gira 360º y suele estar formado por dos potenciómetros a 90º que transforman el movimiento en X e Y del mando en una señal eléctrica proporcional a su posición y que además suele incluir un pulsador, lo que permite detectar cuando presionas la palanca hacia abajo. Si el potenciómetro se mueve de izquierda a derecha es como mover el potenciómetro de extremo a extremo,

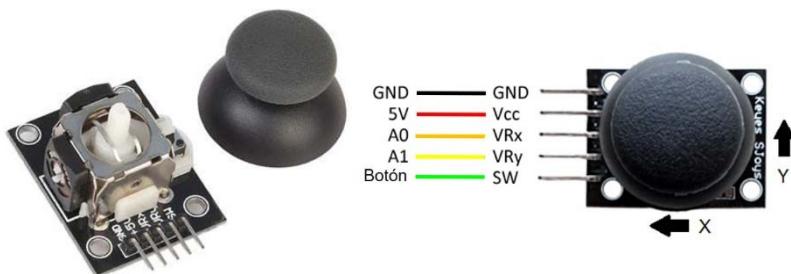


Fig. 56 Conexión del Módulo Joystick Analógico.

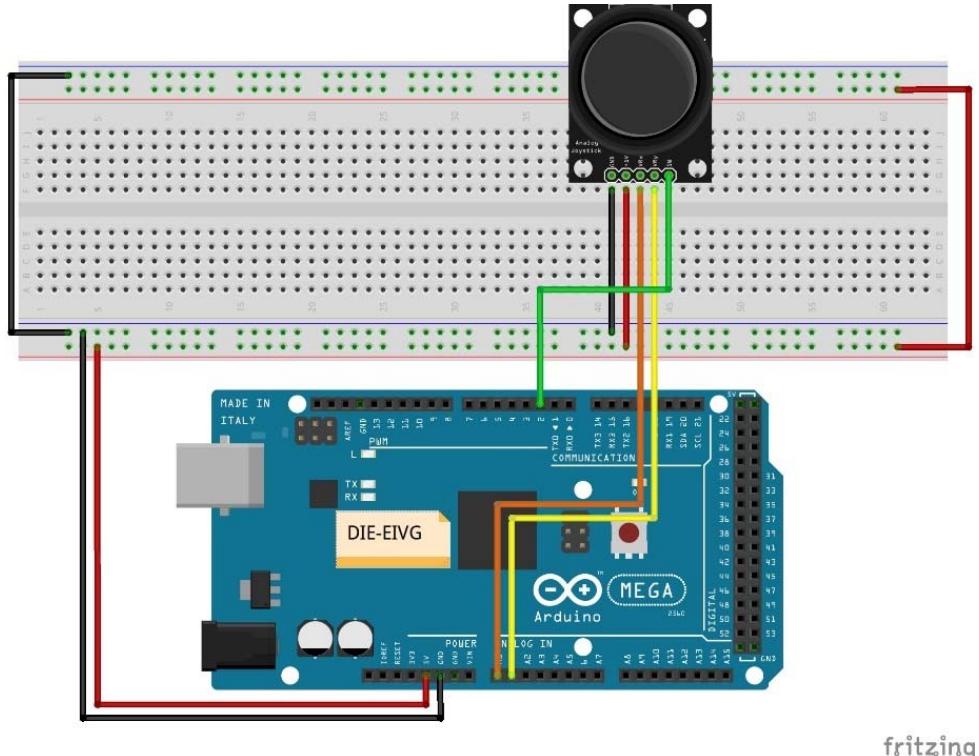
Este módulo de joystick cuenta con cinco pines los cuales se enumeran tomando de referencia de izquierda a derecha:

- GND: Pin conectado a tierra.
- +5V: pin de alimentación(5v).
- VRx: pin de lectura de potenciómetro para el eje x.
- VRy: pin de lectura de potenciómetro para el eje y.
- SW: es un pin adicional que se utiliza para un push button en la parte inferior.

14.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 módulo de Joystick analógico.

14.4. Esquema de conexión



fritzing

Fig. 57 Esquema de conexión Lección 13, ejercicios 1 y 2.

14.5. Códigos de los programas

1. Leccion_13_Joystick_Analogico.ino: Programa que envía al Monitor Serie los valores X e Y de un Joystick Analógico. Los ejes van desde 0 a 1023.
2. Leccion_13_Joystick_Analogico_1.ino: Programa que envía al Monitor Serie los valores X e Y de un Joystick Analógico. Los ejes van desde -512 a 0 y desde 0 a 512

14.6. Bibliografía y direcciones de interés

- [1]. <https://www.web-robotica.com/arduino/como-utilizar-el-modulo-joystick-con-arduino>
- [2]. <https://www.ebotics.com/es/actividad/proyecto-no-2-programar-un-joystick/>
- [3]. <https://aprendiendoarduino.wordpress.com/2018/10/16/joystick-arduino/>
- [4]. <https://programarfácil.com/blog/arduino-blog/joystick-con-arduino/>
- [5]. <https://www.prometec.net/joystick-servo/>
- [6]. <https://www.luisllamas.es/arduino-joystick/>
- [7]. <https://hetpro-store.com/TUTORIALES/joystick-analogico-programado-con-arduino/>

15. Lección 14 Módulo de Receptor IR

15.1. Objetivo de la práctica

Desmitificar la luz infrarroja. Presentar los receptores de infrarrojos y aprender a usar un mando IR con tu Arduino. Uso de la librería IRremote

15.2. Introducción al componente

La luz o radiación infrarroja, se encuentra dentro del espectro electromagnético, entre el espectro electromagnético visible y las microondas, su longitud de onda va desde los 0,7 a 1000 micrómetros, en función de su longitud de onda, los infrarrojos se subdividen en 3 categorías:

- Infrarrojo cercano (de 800 nm a 2500 nm)
- Infrarrojo medio (de 2.5 μm a 50 μm)
- Infrarrojo lejano (de 50 μm a 1000 μm)

Los detectores IR son pequeños microchips con una fotocélula que están sintonizados para detectar la luz infrarroja. Dentro del control remoto hay un LED IR, que emite pulsos IR para indicarle al receptor que actúe. La luz IR no es visible para el ojo humano, lo que significa que se necesita un poco más de trabajo para probar un experimento.

Estos sensores tienen un filtro interno para detectar solo frecuencias infrarrojas cercanas a 38KHz, lo que lo hace compatible con la mayoría de mandos infrarrojos, posee 3 pines de conexión GND, V_{CC} y Señal (SIG), el cual nos permite conectar directamente a un pin digital de nuestro Arduino o cualquier microcontrolador que deseemos usar.

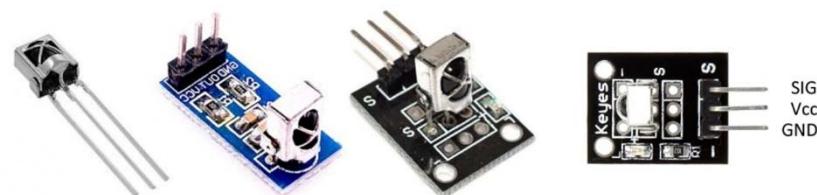


Fig. 58 Tipos de Módulos de Receptor IR y su conexión.

Un mando a distancia es un dispositivo de control que emplea un LED infrarrojo para enviar una señal al receptor. El alcance es limitado, típicamente inferior a 3m. La distancia depende fuertemente del ángulo de emisión, disminuyendo rápidamente a medida que nos desviamos de la dirección frontal.

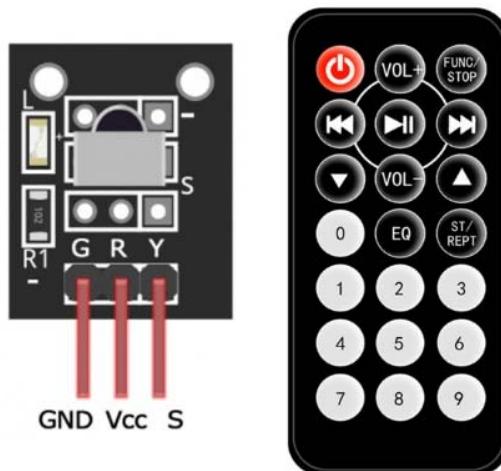


Fig. 59 Módulo de Receptor IR y mando a distancia.

Un mando a distancia transmite un cierto mensaje al receptor empleando luz infrarroja como sistema de transmisión. La luz empleada típicamente está en el rango de 940 nm.

15.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 módulo receptor IR
- 1 control remoto IR
- 6 LEDs de 5mm
- 6 resistencias de $330\ \Omega$
- 1 condensador de $10\ \mu F$
- 1 resistencia de $100\ \Omega$

15.4. Esquema de conexión

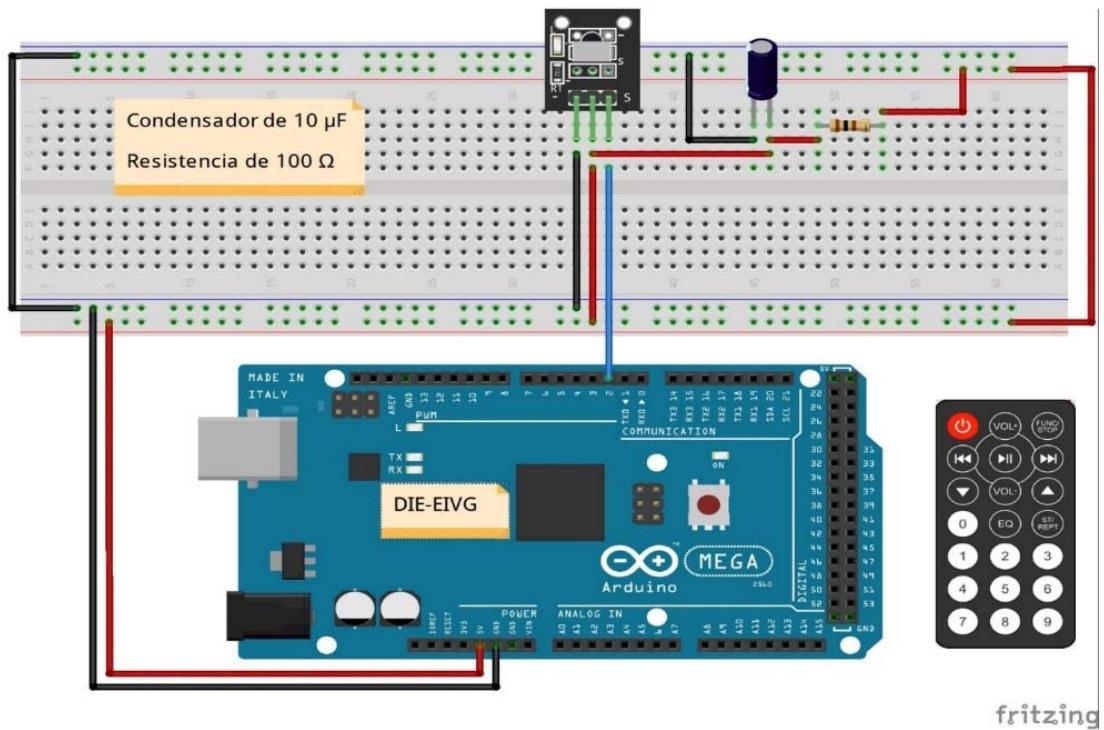


Fig. 60 Esquema de conexión Lección 13, ejercicios 1 y 2.

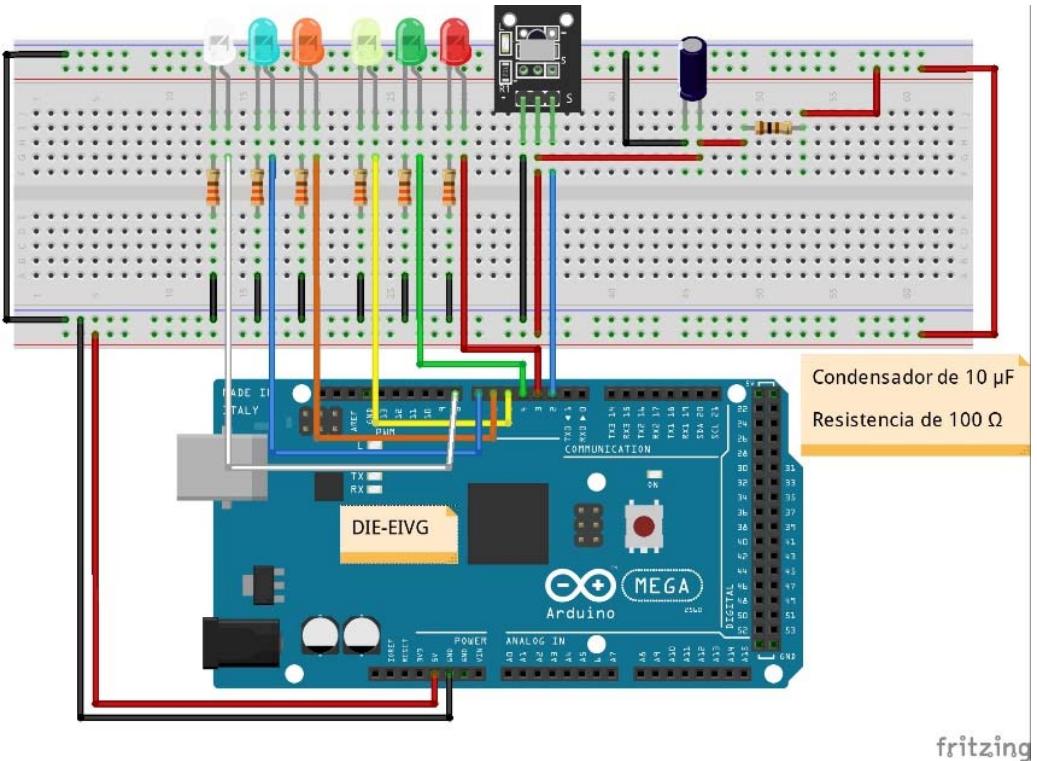


Fig. 61 Esquema de conexión Lección 13, ejercicio 3.

15.5. Códigos de los programas

1. Leccion_14_Receptor_IR.ino: Programa que envía al Monitor Serie la lectura del valor recibido por el mando, y lo muestra por pantalla. El código se muestra en formato hexadecimal.
2. Leccion_14_Receptor_IR_1.ino: Programa que envía al Monitor Serie la lectura del valor recibido por el mando, y lo muestra por pantalla. El código que se muestra es el nombre o número de la tecla presionada.
3. Leccion_14_Receptor_IR_2.ino: Programa que enciende el led correspondiente al número tecleado 1 rojo, 2 verde, 3 amarillo, 4 naranja, 5 azul, 6 blanco y el 0 apaga todos.

15.6. Bibliografía y direcciones de interés

- [1]. https://www.naylampmechatronics.com/blog/36_Tutorial-Arduino-y-control-remoto-Infrarrojo.html
- [2]. <https://www.prometec.net/infrarrojos/>
- [3]. <http://diegorys.es/2016/06/21/como-hacer-un-receptor-ir-con-arduino/>

- [4]. <https://desensores.com/sensores-arduino/proyectos-basicos-con-arduino-para-principiantes/modulo-transmisor-receptor-ir-arduino/>
- [5]. <https://www.zonamaker.com/arduino/modulos-sensores-y-shields/control-a-distancia-mediante-mando-infrarrojo-ir>
- [6]. <https://www.luisllamas.es/arduino-mando-a-distancia-infrarrojo/>
- [7]. <https://aprendiendoarduino.wordpress.com/2018/10/17/mando-a-distancia-ir-con-arduino/>
- [8]. <https://www.diarioelectronicohoy.com/blog/control-remoto-ir>
- [9]. <https://www.circuitbasics.com/arduino-ir-remote-receiver-tutorial/>

16. Lección 15 Módulo de Matriz de Puntos LED MAX7219

16.1. Objetivo de la práctica

Analizar las matrices LED de 8×8 controladas por MAX7219, lo que permite trabajar rápidamente con matrices led y utilizando solo 3 cables para la comunicación.

16.2. Introducción al componente

Una matriz LED es un display formado por múltiples LED en distribución rectangular. Existen distintos tamaños, siendo el más habitual los cuadrados de 8×8 LED.

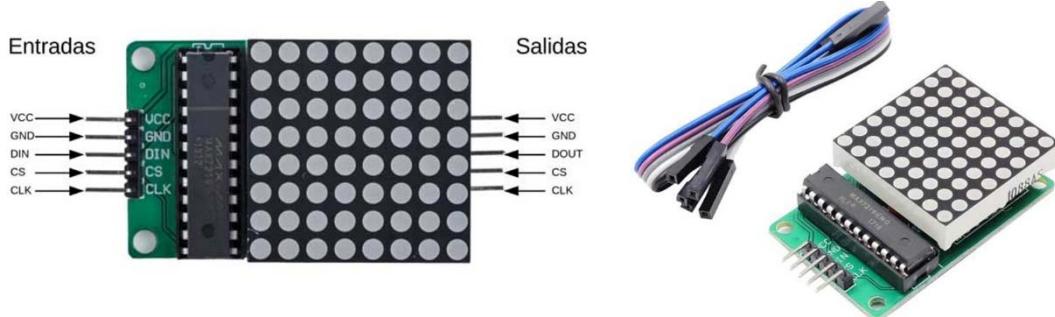


Fig. 62 Matriz 8x8 y sus conexiones

Conexiones

- El pin VCC de la matriz a los 5v del Arduino.
- El pin GND de la matriz al GND del Arduino.
- El pin DIN de la matriz al pin 11 del Arduino.
- El pin Cs de la matriz al pin 10 del Arduino.
- El pin CLK de la matriz al pin 13 del Arduino.

Las matrices de LEDs están conformadas por una serie de filas y columnas en donde hay un LED en cada una de las intersecciones. Para que un LED encienda se deberá recibir un «0» en la fila y un «1» en la columna simultáneamente.

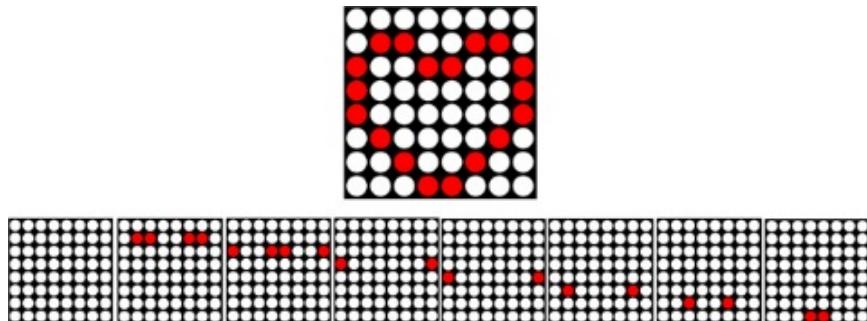


Fig. 63 Matriz 8x8 y sus conexiones

Para poder mostrar gráficos correctamente es necesario realizar un barrido por filas o columnas. Se iluminará sólo una fila a la vez. Si se quiere mostrar un corazón son necesarios 8 pasos, uno por cada fila. Las actualizaciones se realizan suficientemente rápido y con ayuda del efecto de persistencia visual el ojo humano no es capaz de notar el instante donde los LEDs se apagan. Por lo tanto, se ve los LEDs funcionando, pero realmente solo se activa una fila a la vez.

	1	2	3	4	5	6	7	8			1	2	3	4	5	6	7	8	
0	1								B00111000	1	1								B00010000
	2								B01000100		2								B00110000
	3								B01000100		3								B00010000
	4								B01000100		4								B00010000
	5								B01000100		5								B00010000
	6								B01000100		6								B00010000
	7								B01000100		7								B00010000
	8								B00111000		8								B00111000
	1	2	3	4	5	6	7	8			1	2	3	4	5	6	7	8	
Flecha arriba 2	1								B00011000	Flecha arriba 3	1								B00111100
	2								B00111100		2								B01111110
	3								B01111110		3								B11011011
	4								B11011011		4								B00011000
	5								B00011000		5								B00011000
	6								B00011000		6								B00011000
	7								B00011000		7								B00000000
	8								B00000000		8								B00000000

Fig. 64 Definición de caracteres

```

28 byte flecha_arriba_2[8] = { // array con segundo cuadro de animacion de flecha
29   B00011000,
30   B00111100,
31   B01111110,
32   B11011011,
33   B00011000,
34   B00011000,
35   B00011000,
36   B00000000
37 };
38 //DEFINO LOS CARACTERES DE LA MATRIZ
39 #define H { B01000010, B01000010, B01000010, B01000010, B01111110, B01000010, B01000010, B01000010}
40 #define o { B00000000, B00000000, B00000000, B00011000, B00100100, B00100100, B00100100, B00011000}
41 #define l { B00010000, B00010000, B00010000, B00010000, B00010000, B00010000, B00010000, B00001000}
42 #define a { B00000000, B00000000, B00000000, B00111000, B00000100, B00111100, B01000100, B00111010}
43 #define A { B00011000, B00100100, B01000010, B01000010, B01111110, B01000010, B01000010}
44 #define r { B00000000, B00000000, B00000000, B01011000, B01100000, B01000000, B01000000, B01000000}
45 #define d { B00000100, B00000100, B00000100, B00111100, B01000100, B01000100, B01000100, B00111100}
46 #define u { B00000000, B00000000, B00000000, B01000100, B01000100, B01000100, B01000100, B00111010}
47 #define in { B00000000, B00010000, B00000000, B00010000, B00010000, B00010000, B00010000, B00010000}
48 #define n { B00000000, B00000000, B00000000, B01011000, B01100100, B01000100, B01000100, B01000100}
49 #define esp { B00000000, B00000000, B00000000, B00000000, B00000000, B00000000, B00000000, B00000000}

```

Fig. 65 Declaración de caracteres

En la función loop del código del programa escribiremos el siguiente código para que se muestre en la pantalla los diferentes caracteres.

```

void loop() {
  printCaracter(flecha_arriba_2); //Imprime el carácter flecha_arriba_2
  delay(Tiempo); //Esperar Tiempo segundos
  printCaracter(flecha_arriba_3); //Imprime el carácter flecha_arriba_3
  delay(Tiempo); //Esperar Tiempo segundos

void loop() {
  printCaracter(H); //Imprime el carácter H
  delay(Tiempo); //Esperar Tiempo segundos
  printCaracter(o); //Imprime el carácter o
  delay(Tiempo); //Esperar Tiempo segundos

```

Conectando varios MAX7219 en cascada te permite controlar las matrices con solo tres pines del Arduino. Los pines CLK y LOAD de ambos chips se conectan juntos. Por otra parte, el pin DOUT del primer MAX7219 se ha conectado al pin DIN del segundo.

16.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 2 módulos de matriz de 8x8 Max7219

16.4. Esquema de conexión

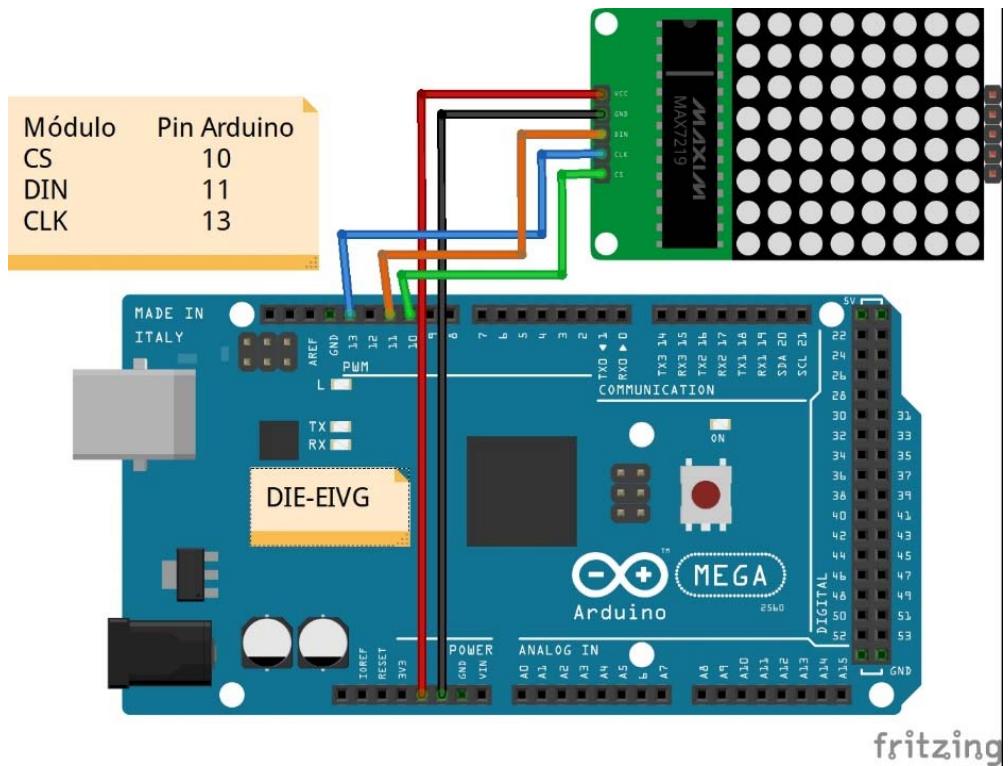


Fig. 66 Esquema de conexión Lección 15, ejercicios 1, 2, 3 y 5

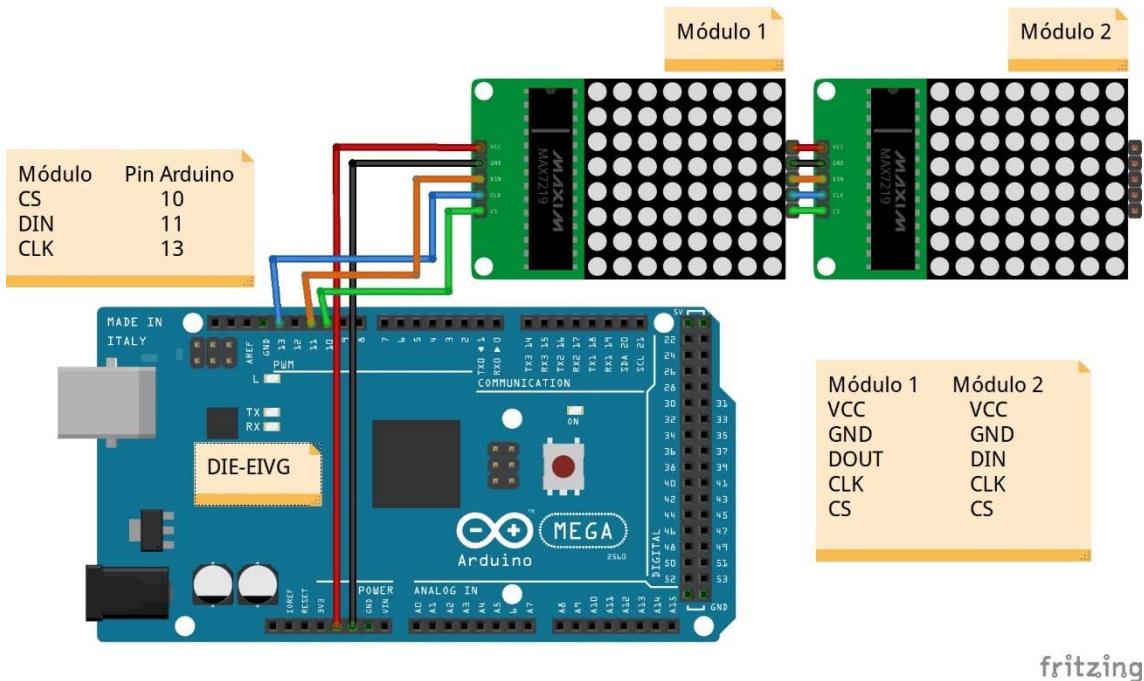


Fig. 67 Esquema de conexión Lección 15, ejercicio 4

16.5. Códigos de los programas

1. Leccion_15_Matriz_de_puntos_MAX7219.ino: Muestra los números de 0-9 en la matriz de 8x8 con MAX7219. El array se escribe en formato binario.
2. Leccion_15_Matriz_de_puntos_MAX7219_1.ino: Muestra el movimiento de una flecha hacia arriba en la matriz de 8x8 con MAX7219. El array se escribe en formato binario.
3. Leccion_15_Matriz_de_puntos_MAX7219_2.ino: Muestra el movimiento de una flecha hacia arriba en dos matrices de 8x8 con MAX7219. El array se escribe en formato binario.
4. Leccion_15_Matriz_de_puntos_MAX7219_3.ino: Muestra la frase "Hola a Arduino" en dos matrices de 8x8 con MAX7219. El array se escribe en formato binario.
5. Leccion_15_Matriz_de_puntos_MAX7219_4.ino: Muestra la frase "Hola a Arduino" en movimiento en una matriz de 8x8 con MAX7219. El array se escribe en formato binario.

16.6. Bibliografía y direcciones de interés

- [1]. <https://www.luisllamas.es/matriz-led-arduino-max7219/>
- [2]. <https://programarfacil.com/blog/arduino-blog/matriz-led-arduino-max7219/>
- [3]. <https://www.prometec.net/8x8-max7219/>
- [4]. <https://hetpro-store.com/TUTORIALES/matriz-led-max7219/>
- [5]. <http://kio4.com/arduino/15bmatriz8x8.htm>
- [6]. <https://www.instructables.com/Ejemplo-b%C3%A1sico-con-IC-MAX7219-Matriz-LED-y-visual/>
- [7]. <http://wayoda.github.io/LedControl/pages/software.html>
- [8]. <https://www.electrogeekshop.com/como-usar-una-matriz-led-con-arduino/>
- [9]. <https://mrchunckuee.blogspot.com/2017/07/mlabx-y-xc8-010.html>
- [10]. <https://www.neoteo.com/matriz-de-led-8x8/>
- [11]. <https://www.neoteo.com/matriz-de-led-8x8-parte-ii/>
- [12]. <https://playground.arduino.cc/Main/DirectDriveLEDMatrix/>
- [13]. <https://akirasan.net/jugando-con-matrices-led-8bits/>
- [14]. <https://www.taloselectronics.com/blogs/tutoriales/kit-matriz-led-de-8-8-con-max7219>

17. Lección 16 Módulo GY-521 Medidor de Inercia

17.1. Objetivo de la práctica

Conocer los conceptos básicos de los sistemas de medida inercial o IMUs. Aprender cómo usar el módulo GY-521 que es uno de los mejores sensores IMU (unidad de medida de la inercia), compatibles con Arduino. Conocer el manejo del MPU6050, Medida inercial de 6 grados de libertad (DoF) pues combina un acelerómetro de 3 ejes y un giroscopio de 3 ejes.

17.2. Introducción al componente

Sistemas de coordenadas

Para definir un punto en un espacio de 3 dimensiones necesitamos 3 números, que son la proyección del punto en los tres ejes ortogonales (perpendiculares entre sí).

De este modo, una vez definida la unidad de medida, la posición de cualquier punto del espacio queda definido mediante las 3 coordenadas que llamamos x, y, z, con respecto a nuestro sistema de referencia.

Pero en el caso de un avión. ¿Cómo definimos su posición en el espacio? Con un único punto, de 3 coordenadas en el espacio, podemos fijar por ejemplo su centro de gravedad, pero resulta evidente de que necesitamos además especificar otros parámetros para saber si el avión vuela boca arriba o boca abajo, o si tiene un Angulo ascendente o descendente.

Por eso para reflejar la situación y orientación de un avión o barco necesitamos más información: Los ángulos que cada uno de los ejes del aparato marca con respecto a la nuestra referencia. Estos ángulos de desviación respecto los ejes de referencia se llaman en inglés Yaw, Pitch y Roll, y en castellano corresponden a Guiñada, Cabeceo y Balanceo.

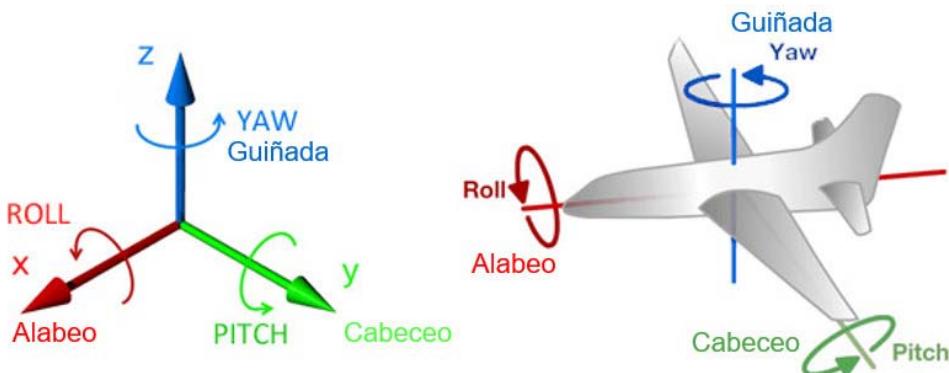


Fig. 68 Esquema de conexión Lección 16, ejercicios 1 y 2.

Para esta práctica vamos a utilizar el modulo GY-201, que incluye un MPU650 y un regulador de tensión, con lo que podemos alimentar a tanto 3.3V como a 5V.

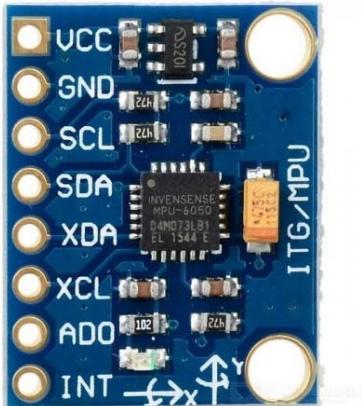


Fig. 69 Modulo GY-201, que incluye un MPU650.

EL módulo MPU6050 contiene un giroscopio de tres ejes con el que podemos medir velocidad angular y un acelerómetro también de 3 ejes con el que medimos los componentes X, Y y Z de la aceleración, el acelerómetro trabaja sobre el principio piezoelectrónico, posee además de un sensor de temperatura.

Los acelerómetros miden aceleración, es decir, la variación de velocidad por unidad de tiempo, internamente tienen un MEMS (MicroElectroMechanical Systems) que de forma similar a un sistema masa-resorte permite medir la aceleración.

Los giroscopios son unos dispositivos que mide la velocidad angular de un objeto, es decir, el desplazamiento angular por unidad de tiempo o lo rápido que gira un cuerpo alrededor de su eje. En este caso, también se emplean técnicas MEMS para medir dicha velocidad usando un efecto conocido como Coriolis. Gracias a eso se puede medir la velocidad angular o, integrando la velocidad angular respecto al tiempo, se puede obtener el desplazamiento angular.

Para entender el funcionamiento del acelerómetro, imaginemos que queremos fabricar un dispositivo que quiera medir aceleración. Podríamos construir un sensor formado por un cuerpo sólido, en cuyo interior suspendemos una masa sujetada por muelles al cuerpo exterior.

Al aplicar una aceleración al conjunto la masa suspendida ejercerá una fuerza sobre los muelles causando que uno se contraiga y otro se elongue, por lo que la posición relativa de la masa dentro del sensor variará.

Este desplazamiento de la masa libre interior puede ser medido para determinar la magnitud de la aceleración. El desplazamiento será proporcional a la aceleración soportada, y se mantendrá constante mientras la aceleración sea constante.

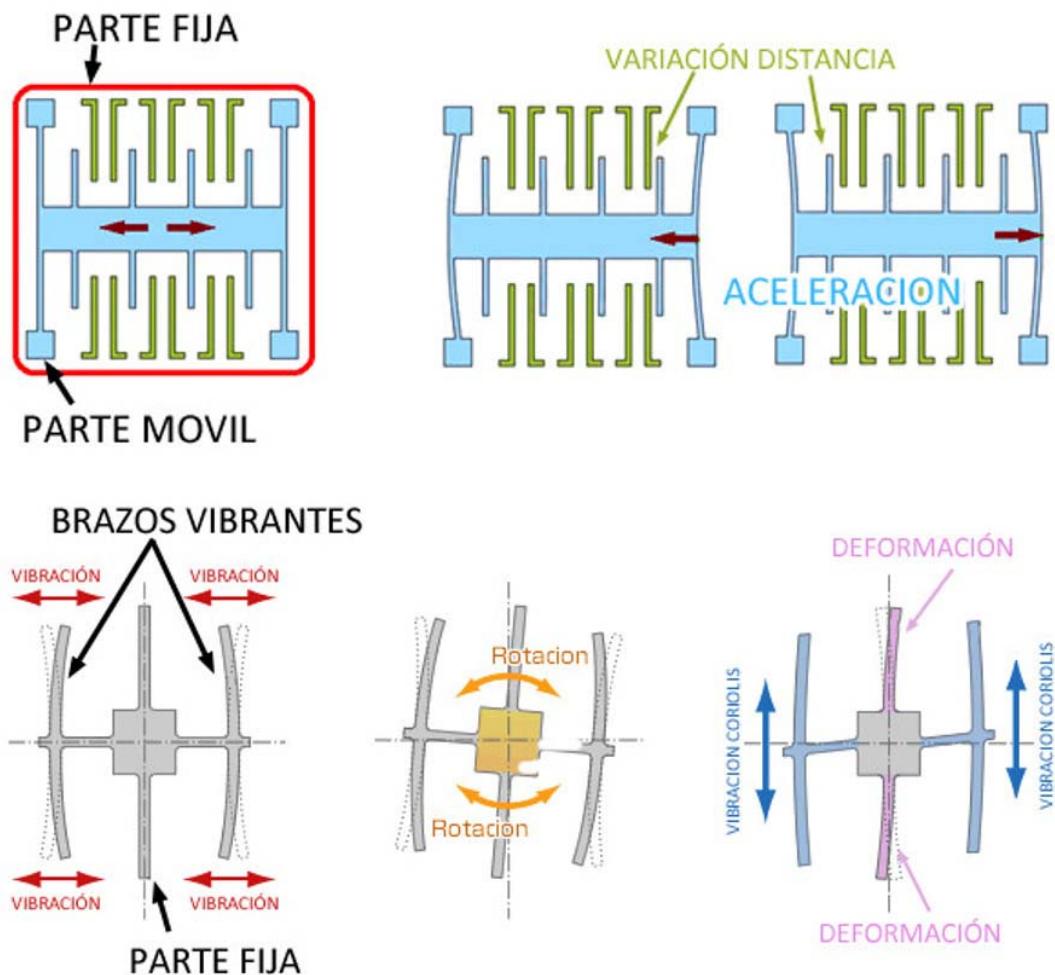


Fig. 70 Esquema interno del MEMS del giroscopio.

Para su conexión, basta con conectar los pines de alimentación y montar los pines necesarios para el control del bus I2C (el pin SDA y SCL de Arduino con los pines SDA y SCL correspondientes del sensor). Cuando el IMU dispone de alguna medida se lo informa a nuestro Arduino mediante una interrupción y por eso conectamos el pin INT a nuestra primera interrupción en el pin 2.

Librerías para el PMU6050

Principalmente usaremos librería wire.h

<https://www.arduino.cc/en/reference/wire>

Para esta práctica podemos usar otras dos librerías, la librería desarrollada por Jeff Rowberg, MPU6050, que la podemos descargar de:

<https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/MPU6050>

Y una librería adicional para la comunicación I2C:

<https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/I2Cdev>

Para ver bien el funcionamiento del módulo, es conveniente imprimir la silueta que está debajo y colocar bien el sensor y después de cargar el programa en la placa activar el serial plotter. O usar la maqueta de avión que se os dará en el laboratorio.

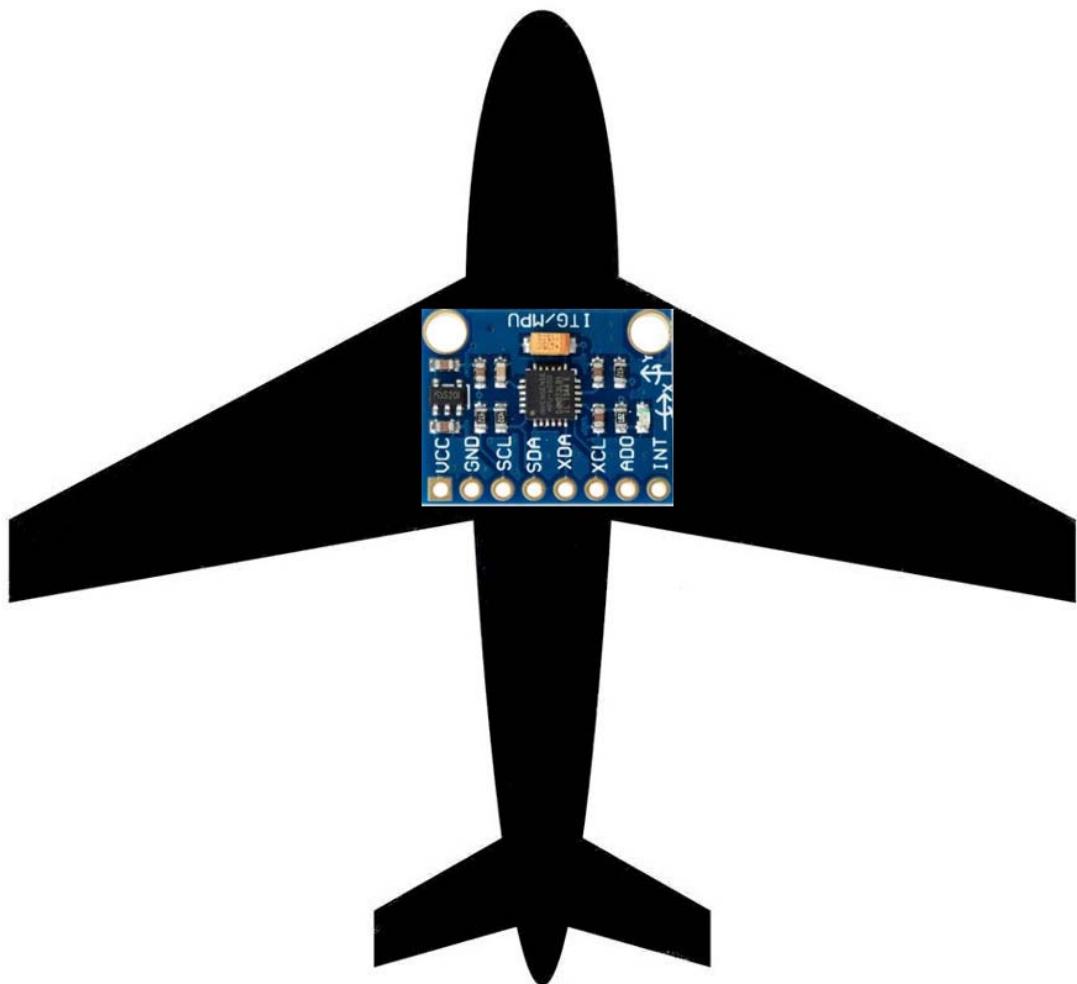


Fig. 71 Esquema de conexión Lección 16, ejercicios 1 y 2.

17.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 módulo GY-521
- 2 resistencias de 330 Ω

- 2 LEDs de 5mm

17.4. Esquema de conexión

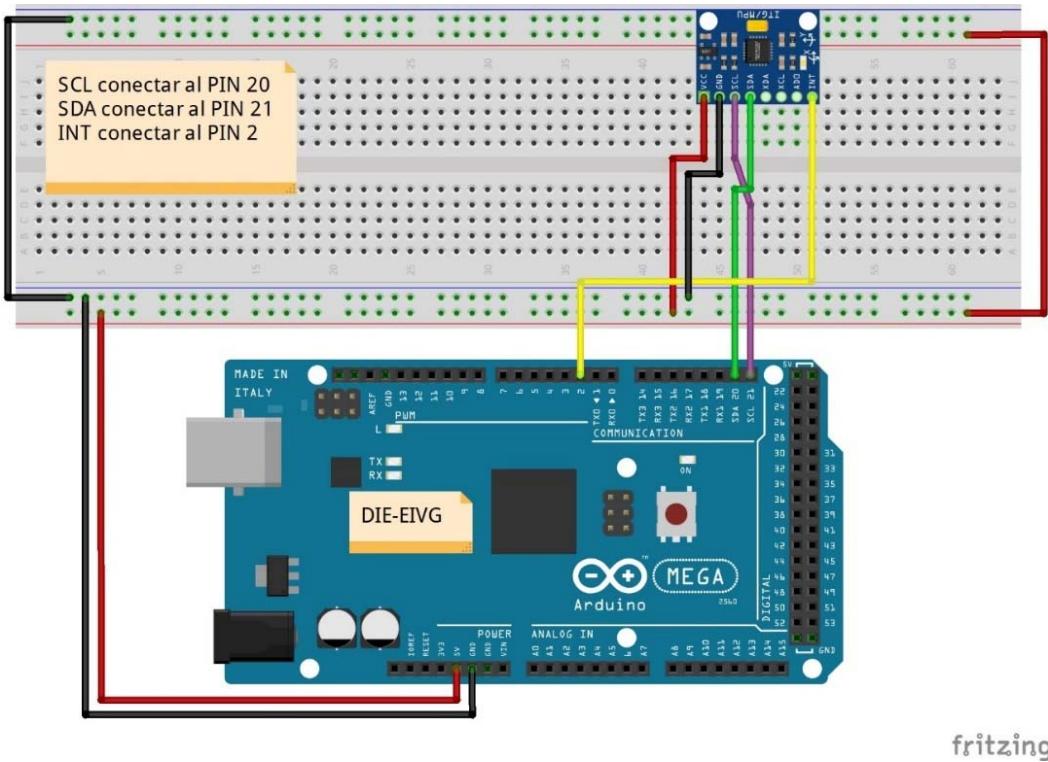


Fig. 72 Esquema de conexión Lección 16, ejercicios 1 y 2.

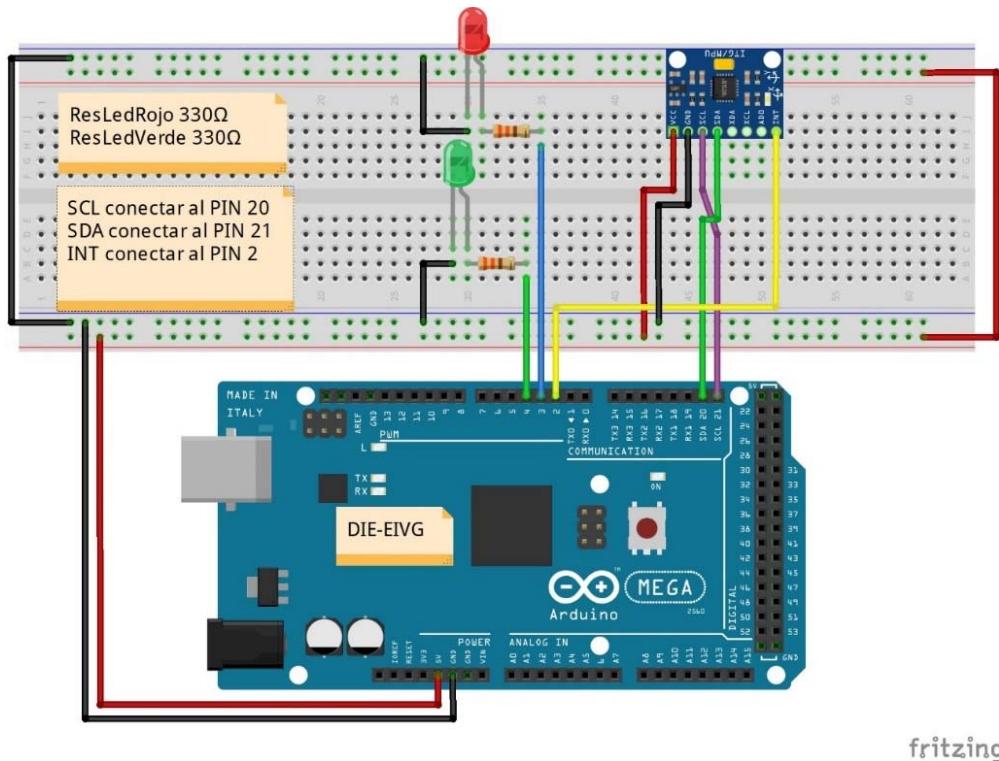


Fig. 73 Esquema de conexión Lección 16, ejercicio 3.

17.5. Códigos de los programas

1. Leccion_16_Modulo_GY_521.ino: Programa que muestra en el Serial Plotter los ángulos Roll-Alabeo(X), Pitch-Cabeceo(Y) y Yaw-Guiñada(Z).
2. Leccion_16_Modulo_GY_521_0: Programa que muestra los valores del acelerómetro y del giroscopio en el monitor serie.
3. Leccion_16_Modulo_GY_521_1.ino: Programa para mostrar el Serial Plotter los ángulos Roll-Alabeo(X), Pitch-Cabeceo(Y) y Yaw-Guiñada(Z). Enciende un led si el pitch-cabeceo es menor a -30 grados y enciende otro si el pitch es mayor a 30 grados.

17.6. Bibliografía y direcciones de interés

- [1]. <https://www.prometec.net/imu-mpu6050/>
- [2]. <https://www.prometec.net/usando-el-mpu6050/>
- [3]. <https://www.diarioelectronicohoy.com/blog/configurar-el-mpu6050>
- [4]. https://www.naylampmechatronics.com/blog/45_Tutorial-MPU6050-Aceler%C3%B3metro-y-Giroscopio.html
- [5]. <https://www.hwlibre.com/mpu6050/>
- [6]. <https://www.luisllamas.es/arduino-orientacion-imu-mpu-6050/>
- [7]. <https://www.luisllamas.es/como-usar-un-acelerometro-arduino/>
- [8]. <https://www.luisllamas.es/obtener-orientacion-y-altitud-ahrs-con-imu-9dof-y-rtimulib-arduino/>
- [9]. https://es.wikipedia.org/wiki/Ejes_del_av%C3%ADn
- [10]. https://es.wikipedia.org/wiki/%C3%81ngulos_de_navegaci%C3%B3n
- [11]. <https://www.aranacorp.com/es/usando-un-modulo-mpu6050-con-arduino/>
- [12]. https://create.arduino.cc/projecthub/Nicholas_N/how-to-use-the-accelerometer-gyroscope-gy-521-6dfc19
- [13]. http://electronoobs.com/eng_arduino_tut23_code_5.php

18. Lección 17 Sensor HC-SR501 PIR

18.1. Objetivo de la práctica

Aprender a cómo utilizar un detector de movimiento HC-SR501 PIR con una placa Arduino MEGA2560. Analizar su utilidad como sensores de presencia y movimiento.

18.2. Introducción al componente

Los sensores infrarrojos pasivos (PIR) son dispositivos para la detección de movimiento. Son baratos, pequeños, de baja potencia, y fáciles de usar.

Los sensores PIR se basan en la medición de la radiación infrarroja, al detectar un cambio en la radiación infrarroja que reciben y disparan una Señal/Alarma al percibirlo. Todos los cuerpos (vivos o no) emiten una cierta cantidad de energía infrarroja, mayor cuanto mayor es su temperatura. Los dispositivos PIR disponen de un sensor piro eléctrico capaz de captar esta radiación y convertirla en una señal eléctrica.

El otro elemento restante para que todo funcione es la óptica del sensor. Básicamente es una cúpula de plástico formada por lentes de fresnel, que divide el espacio en zonas, y enfoca la radiación infrarroja a cada uno de los campos del PIR.

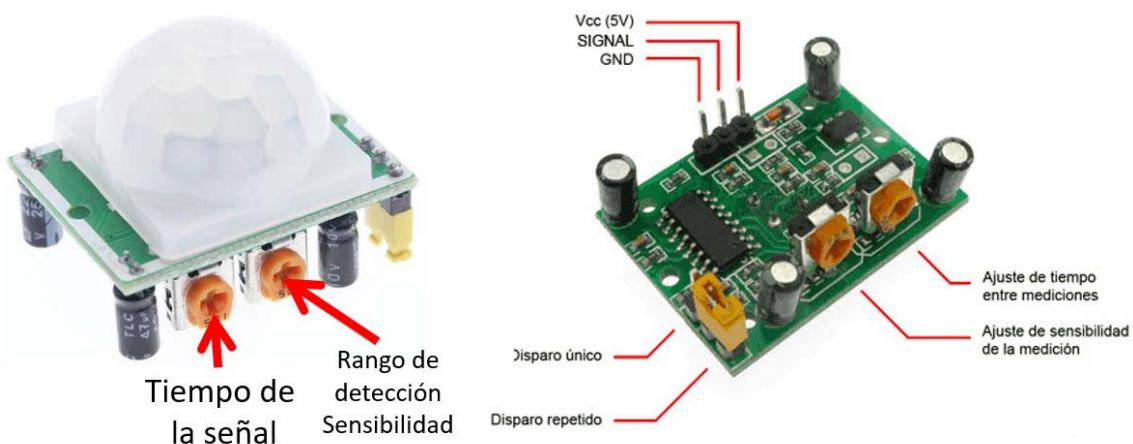


Fig. 74 Esquema del Sensor HC-SR501 PIR.

Para ajustar la sensibilidad podemos usar uno de los potenciómetros que incluye el sensor, fijaras en la imagen de arriba. (Girando a favor del reloj aumentamos la sensibilidad)

El segundo potenciómetro ajusta el tiempo que estará activa la señal de detección después de que esta haya desaparecido.

Puede detectar movimiento de 3 hasta 7 metros de distancia. y aperturas del PIR de hasta 90 y 110° Este sensor de movimiento PIR tiene 3 pinos, VCC, SEÑAL a conectar a pin de la placa y GND, 2 potenciómetros para ajustar la sensibilidad y el tiempo que dura la señal. El retardo se puede configurar entre 5 y 300 segundos mientras que el potenciómetro de sensibilidad ajusta el rango de detección de aproximadamente 3 metros a 7 metros.

18.3. Componentes necesarios

- Placa Arduino Mega
 - Cable USB
 - Protoboard
 - 1 resistencia de 330 Ω
 - 1 LED de 5mm
 - 1 sensor de movimiento HC-SR501 PIR

18.4. Esquema de conexión

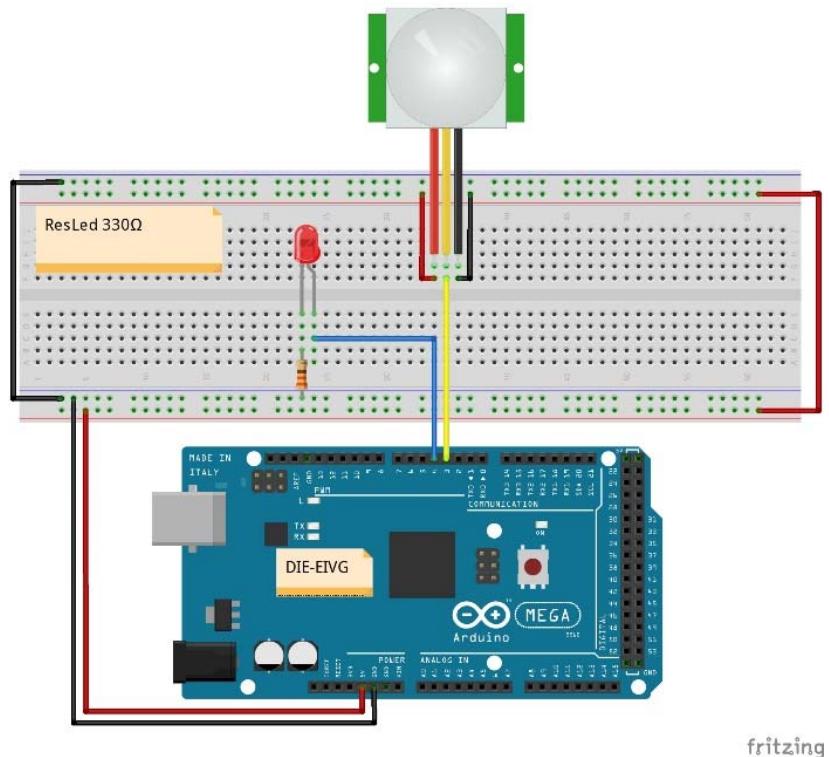


Fig. 75 Esquema de conexión Lección 17.

18.5. Códigos de los programas

1. Leccion_17_Sensor_HC-SR501_PIR.ino: Programa que permite encender un led mediante el sensor de movimiento PIR.
 2. Leccion_17_Sensor_HC-SR501_PIR_1.ino: Programa que permite encender un led mediante el sensor de movimiento PIR. Se tiene en cuenta los tiempos de calibración, según recomienda el fabricante. Y muestra por el monitor serie si se ha detectado movimiento

18.6. Bibliografía y direcciones de interés

- [1]. <https://www.luisllamas.es/detector-de-movimiento-con-arduino-y-sensor-pir/>
- [2]. <https://www.prometec.net/sensor-pir/>
- [3]. <https://www.hispavila.com/sensor-hc-sr501-con-arduino/>
- [4]. <https://www.hwlibre.com/hc-sr501/>
- [5]. http://www.arduinove.com/index.php?route=product/product&product_id=54
- [6]. <https://proyectosinteresantes.com/detector-movimiento-arduino-pir-hcsr501/>
- [7]. <https://steemit.com/spanish/@michelylopez/sensor-pir-hc-sr501-con-alarma-or-proyectos-con-arduino-uno>
- [8]. <http://arduparatodos.blogspot.com/2016/11/prueba-de-sensor-de-movimiento-hc-sr501.html>
- [9]. <https://unprogramador.com/sensor-de-movimiento-pir-con-arduino/>
- [10]. <https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor>

19. Lección 18 Módulo Sensor Detección de Nivel de Agua

19.1. Objetivo de la práctica

Aprender a utilizar un módulo de sensor de detección de nivel de agua.

19.2. Introducción al componente

Este módulo puede detectar la profundidad del agua, el componente principal es un circuito amplificador que está formado principalmente por un transistor y unas líneas metálicas en el PCB. Cuando está puesto en el agua, el elemento sensor presentará una resistencia que puede cambiar junto con el cambio de profundidad del agua. La señal de la profundidad del agua es convertida en señal eléctrica, y podemos conocer el cambio de profundidad del agua.

El sensor de agua está diseñado para la detección de agua y se puede usar para confirmar la presencia de lluvia, nivel de agua en un determinado punto o alerta por fuga de agua.

Este sensor de agua puede ser ampliamente utilizado en la detección de la precipitación, nivel del agua, incluso fugas. Las huellas del sensor tienen una resistencia

de pull-up débil de $1\text{ M}\Omega$. La resistencia tiene el valor de seguimiento alta hasta que una gota de agua corta la huella del sensor con conexión a tierra.

Sensor de agua (esquemático)

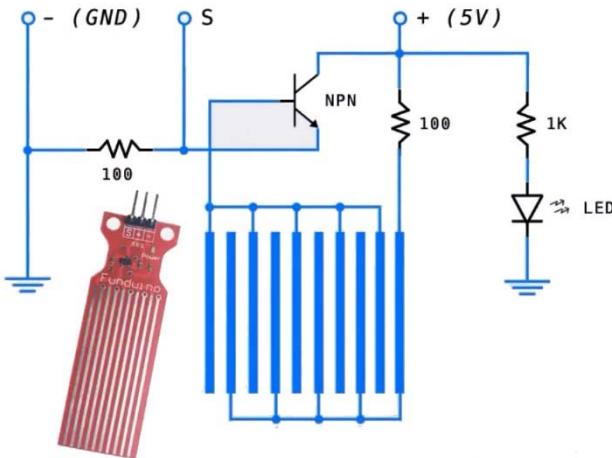


Fig. 76 Esquema eléctrico de un módulo de sensor de detección de nivel de agua.

Este sensor tiene bajo consumo de energía y alta sensibilidad. Características:

- Voltaje de funcionamiento: 3.3/5V
- Corriente de trabajo: < 20 mA
- Interfaz: analógica
- Ancho de detección: 40 mm × 16 mm
- Temperatura de trabajo: 10° C ~ 30 °C
- Señal de voltaje de salida: 0 ~ 4.2V

Pines de alimentación
y señal



Fig. 77 Módulo de sensor de detección de nivel de agua.

Estos módulos son muy simples y lo único que hay que hacer es conectar a tensión y GND (se alimenta a 5V o a 3,3V en los pines Vcc y GND) y el tercer pin es una señal analógica, proporcional a la cantidad de agua que detecta, comprendido entre Vcc y GND.

19.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 5 resistencias de $330\ \Omega$
- 5 LEDs de 5mm
- 1 módulo sensor de detección de agua

19.4. Esquema de conexión

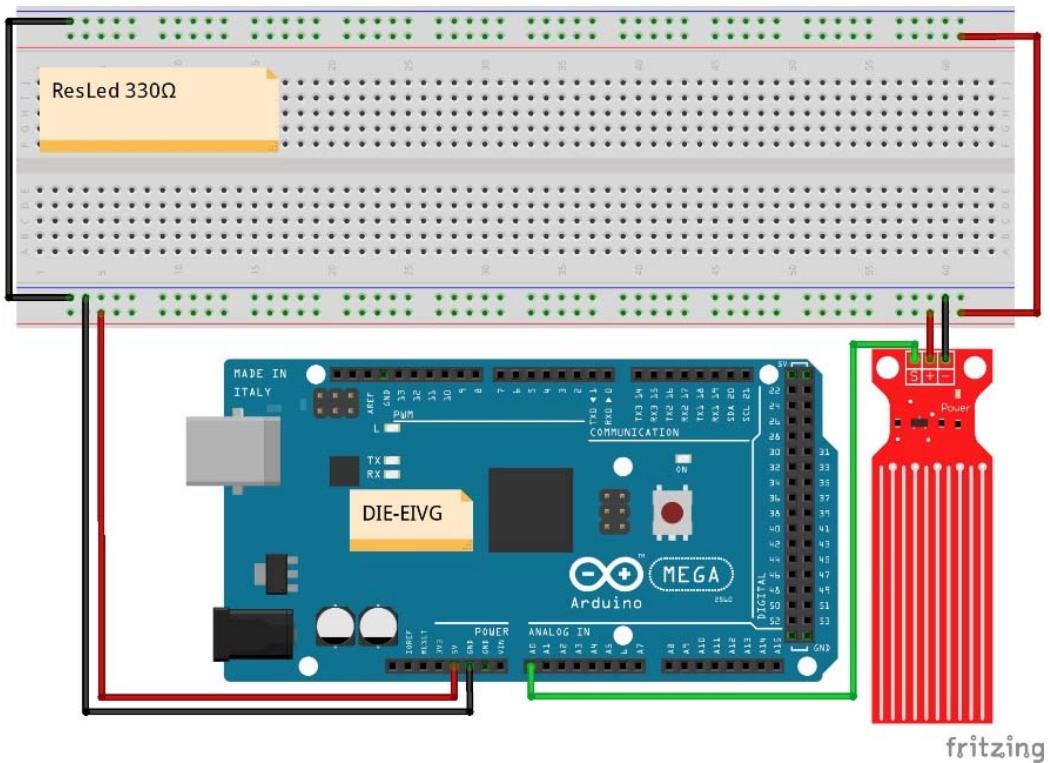


Fig. 78 Esquema de conexión Lección 18.

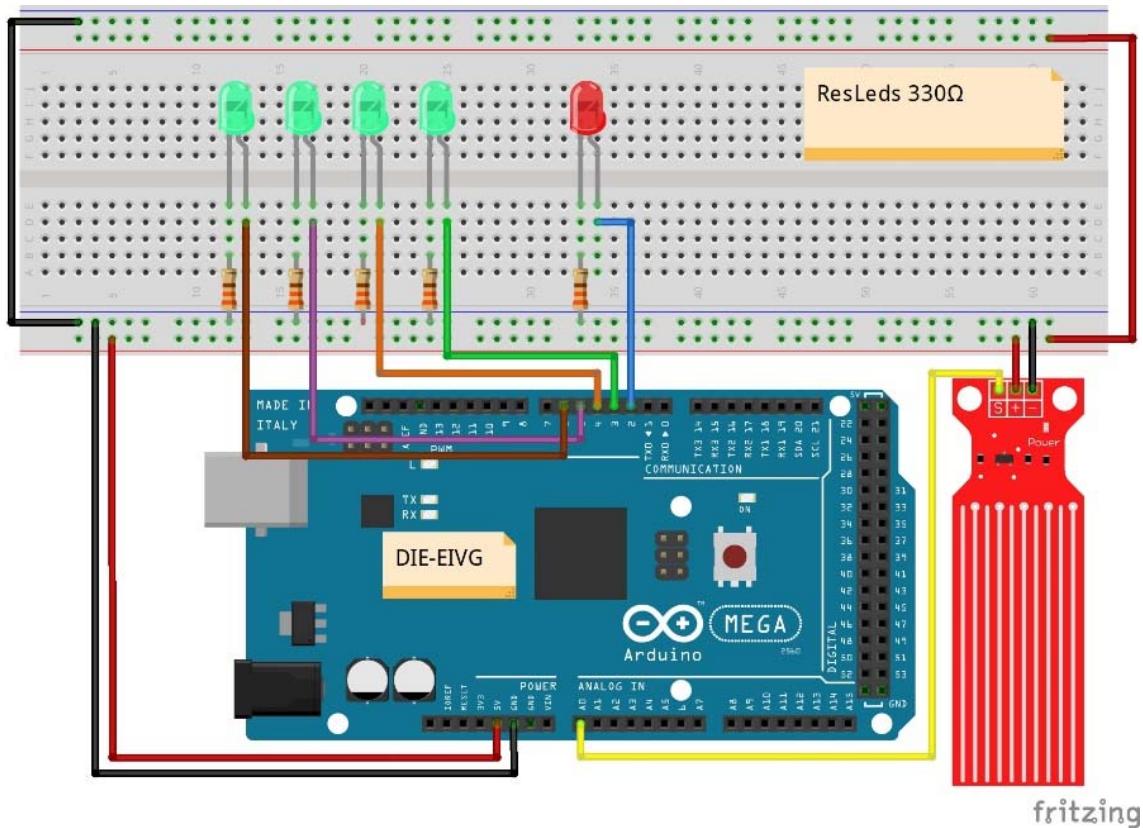


Fig. 79 Esquema de conexión Lección 18.

19.5. Códigos de los programas

1. Leccion_18_Modulo_Sensor_de_agua.ino: Programa que permite la lectura de la entrada analógica A0 a donde se encuentra conectado un sensor de agua y mostrando los valores mediante monitor serie.
2. Leccion_18_Modulo_Sensor_de_agua_1.ino: Programa que permite Lectura de la entrada analógica A0 a donde se encuentra conectado un sensor de agua y va apagando los leds según se va vaciando el depósito de combustible y enciende de un LED conectado a pin digital 2 si detecta que el combustible entra en reserva. Simula el depósito de combustible de un vehículo.

19.6. Bibliografía y direcciones de interés

- [1]. <https://www.prometec.net/sensor-agua/>
- [2]. <https://descubrearduino.com/sensor-arduino-detector-de-agua/>
- [3]. <https://scidle.com/es/como-usar-sensor-de-nivel-de-agua-con-arduino/>

- [4]. <https://aprendiendoarduino.wordpress.com/2018/10/17/sensor-deteccion-de-agua-para-arduino/>
- [5]. [Sensor de Nivel de Agua - Moviltronics](#)
- [6]. <https://juegosrobotica.es/sensor-de-agua/#>
- [7]. <https://programarfacil.com/blog/arduino-blog/sensor-de-nivel-de-agua-con-arduino/>
- [8]. <https://www.sensoresdepresion.top/2019/10/control-de-nivel-de-agua-en-un-tanque-con-arduino.html>
- [9]. <https://www.sysadminsdecuba.com/2020/03/controlador-de-nivel-de-agua-en-tanque-con-sensor-ultrasonico-y-arduino/>
- [10]. <https://www.geekfactory.mx/tutoriales/tutoriales-arduino/sensor-de-nivel-de-agua-con-arduino-y-modulo-ultrasonico/>

20. Lección 19 Módulo Reloj en Tiempo Real

20.1. Objetivo de la práctica

Aprender a usar el DS3231 y DS1307, módulos de reloj que muestra el año, mes, día, hora, minuto, segundo y semana.

20.2. Introducción al componente

Un reloj de tiempo real (RTC) es un dispositivo electrónico que permite obtener mediciones de tiempo en las unidades temporales que empleamos de forma cotidiana y que funcionan con segundos, minutos, horas, días, semanas, meses y años. La fecha al final del mes se ajusta automáticamente a los meses con menos de 31 días, incluidas las correcciones del año bisiesto. El reloj funciona en formato de 24 horas o de 12 horas con un indicador AM/PM

En el mundo Arduino existen dos RTC habituales el DS1307 y el DS3231, ambos fabricados por Maxim. El DS3231 tiene una precisión muy superior y puede considerarse sustituto del DS1307.

- En el modelo **DS1307** las variaciones de temperatura que afectan a la medición del tiempo de los cristales resonadores se traducen en errores en un desfase acumulado. Esto hace que el DS1307 sufra de un desfase temporal, que puede llegar a ser 1 o 2 minutos al día.

- Para solucionarlo, el **DS3231** incorpora medición y compensación de la temperatura garantizando una precisión de al menos 2ppm, lo que equivale a un desfase máximo 172ms/día o un segundo cada 6 días. En el mundo real normalmente consiguen precisiones superiores, equivalente a desfases de 1-2 segundos al mes.

La comunicación en ambos modelos se realiza a través del bus I2C, por lo que es sencillo obtener los datos medidos. La tensión de alimentación es 4.5 a 5.5 para el DS1307, y 2.3 a 5.5V para el DS3231.

También incorporan una batería CR2032 para mantener el dispositivo en hora al retirar la alimentación. Esta batería debería ser capaz de mantener alimentado durante varios años al DS1307, y durante meses al DS3231. La tensión de alimentación de batería es de 2.0 a 3.5 para el DS1307 y de 2.3 a 5.0 para el DS3231.

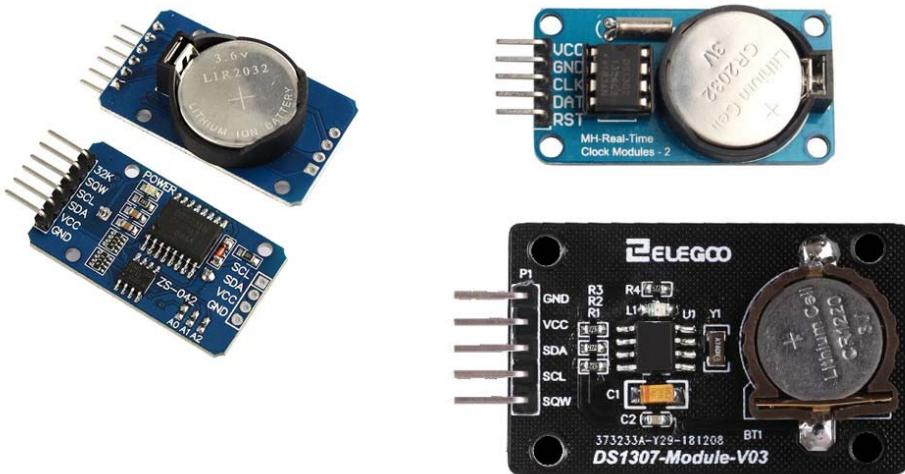


Fig. 80 Diferentes Módulos de Reloj en Tiempo Real.

La conexión es sencilla y similar tanto para el DS1307 como el DS3231. Simplemente alimentamos el módulo desde Arduino mediante 5V y GND. Por otro lado, conectamos los pines del bus I2C a los correspondientes de Arduino, Pin SCL al pin 21 de Arduino y el pin SDA al pin 20 de Arduino.

Las conexiones son las siguientes:

- VCC – Alimentación de 5 volts del módulo.
- GND – Tierra o común.
- SCL – Señal de reloj del bus I2C.
- SDA – Señal de datos del bus I2C.
- SQW – Salida de interrupción o señal de reloj.

Para este ejemplo usaremos la librería RTClib de Adafruit, librería que pueden descargarlo en:

<https://github.com/adafruit/RTClib>

Necesitamos descargar e importar dicha librería al IDE de Arduino.

```
#include <Wire.h>
#include "RTClib.h"
```

```
RTC_DS3231 rtc; // crea objeto del tipo RTC_DS3231
```

20.3 Componentes necesarios

- Placa Arduino Mega
 - Cable USB
 - Protoboard
 - 1 módulo Reloj en tiempo real DS3231 RTC
 - 1 resistencia de 330 Ω
 - 1 LED de 5mm

20.4. Esquema de conexión

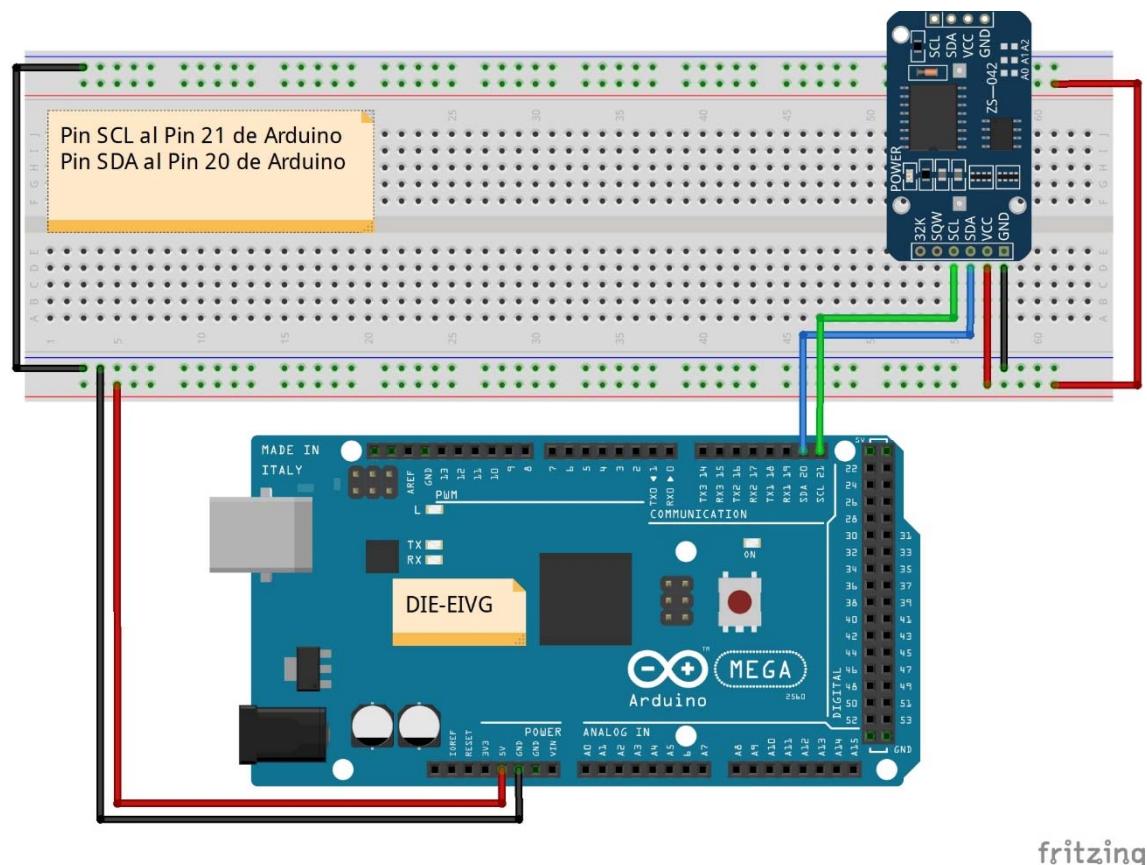
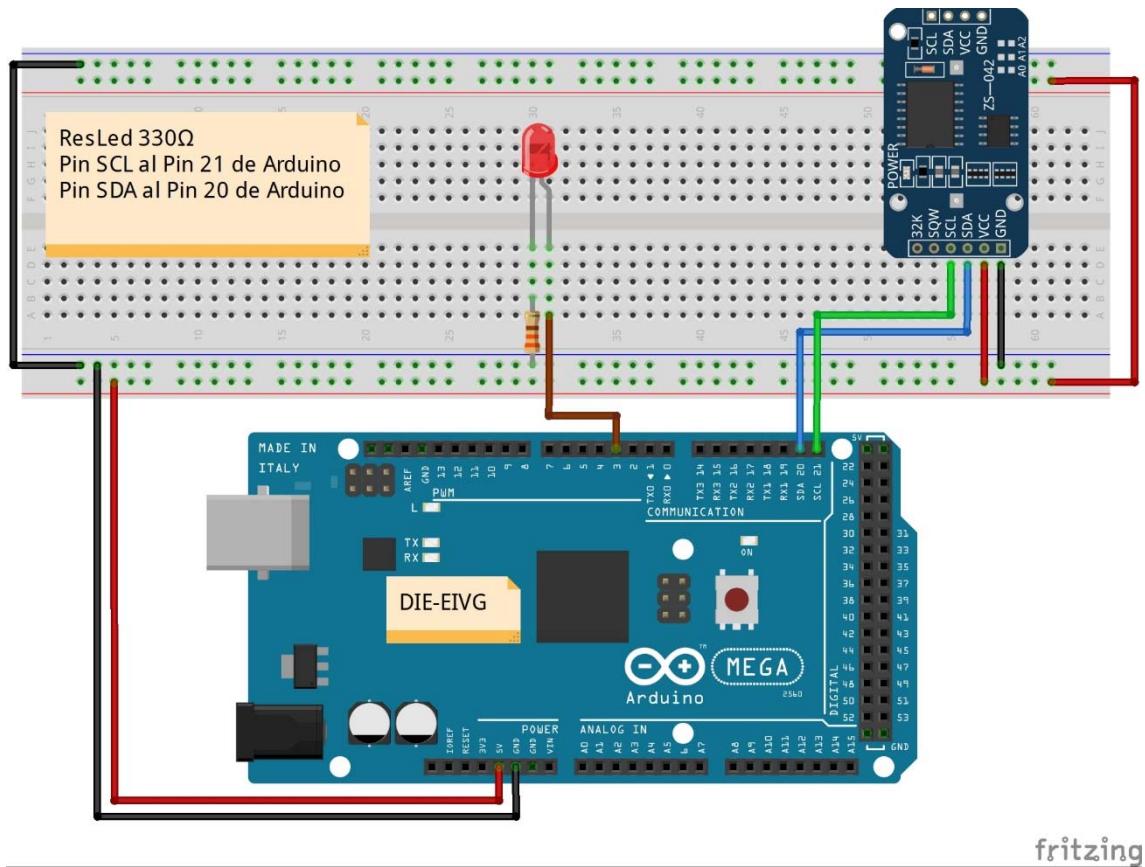


Fig. 81 Esquema de conexión Lección 19, ejercicio 1, 2 y 3.



fritzing

Fig. 82 Esquema de conexión Lección 19, ejercicio 4

20.5. Códigos de los programas

1. Leccion_19_Modulo_Reloj_en_tiempo_real.ino: Programa que establece la fecha y la hora iniciales para el módulo RTC modelo DS3231 y muestra dichos datos mediante el monitor serie. Requiere instalar librería RTCLib.
2. Leccion_19_Modulo_Reloj_en_tiempo_real_1.ino: Programa que establece la fecha y la hora iniciales para el módulo RTC modelo DS3231 y muestra la fecha, el día de la semana y la hora en el monitor serie. Requiere instalar librería RTCLib.
3. Leccion_19_Modulo_Reloj_en_tiempo_real_2.ino: Programa que establece una alarma para escribir en monitor serie el texto Alarma exactamente a las 14:30 h. todos los días. Para engañar al reloj y poder probar el programa cambiamos la hora a 30 s. antes de que salte la alarma, hay que activar un trozo de código.
4. Leccion_19_Modulo_Reloj_en_tiempo_real_3.ino: Programa que activa un Led a las 14:30 h y lo desactiva más tarde a las 14:31 h todos los días. Para

engaños al reloj y poder probar el programa cambiamos la hora a 30 s. antes de que salte la alarma, hay que activar un trozo de código.

20.6. Bibliografía y direcciones de interés

- [1]. <http://rogerbit.com/wprb/2020/03/reloj-de-tiempo-real-con-arduino/>
- [2]. <https://www.prometec.net/relojes-rtc/>
- [3]. <https://programarfacil.com/blog/arduino-blog/reloj-con-arduino-rtc/>
- [4]. <https://www.luisllamas.es/reloj-y-calendario-en-arduino-con-los-rtc-ds1307-y-ds3231/>
- [5]. <https://www.geekfactory.mx/tutoriales/tutoriales-arduino/ds3231-el-reloj-en-tiempo-real-de-gran-precision-para-arduino/>
- [6]. <https://www.aranacorp.com/es/usando-un-modulo-ds3231-con-arduino/>
- [7]. <https://areacom.altervista.org/arduino-rtc-ds1307-at24c32/>
- [8]. <http://ardinolearning.com/code/arduino-ds3231-rtc-example.php>
- [9]. <https://www.profetolocka.com.ar/2016/09/24/modulo-reloj-de-tiempo-real-para-arduino/>
- [10]. <http://codigoelectronica.com/blog/arduino-ds3231>
- [11]. <https://www.electrogeekshop.com/tutorial-rtc-ds1307-y-eeprom-at24c/>
- [12]. <http://howtomechatronics.com/tutorials/arduino/arduino-ds3231-real-time-clock-tutorial/>
- [13]. https://www.naylampmechatronics.com/blog/52_tutorial-rtc-ds1307-y-eeprom-at24c32.html
- [14]. <https://www.diarioelectronicohoy.com/blog/reloj-con-el-ds1307>

21. Lección 20 Módulo de Sensor de Sonidos

21.1. Objetivo de la práctica

Conocer el manejo del sensor de sonido (micrófono), y aprender a utilizar un módulo de sensor de sonidos KY038 y KY037. Utilizar el pin digital D0 del sensor y mostrar las lecturas del sensor utilizando el pin analógico A0 del sensor.

21.2. Introducción al componente

Este módulo permite detectar cuándo el sonido ha excedido el punto de ajuste que se ha seleccionado. El sonido se detecta a través de un micrófono y se alimenta a un

amplificador operacional LM393. El punto de ajuste del nivel de sonido se ajusta mediante un potenciómetro incorporado. Cuando el nivel de sonido excede el punto de ajuste, se ilumina un LED en el módulo y la salida se envía baja. El módulo KY-038 al incorpora un micrófono junto con un comparador LM393, permite obtener la lectura tanto como un valor analógico como de forma digital.

El uso habitual de este tipo de sensores no amplificados es emplear la salida digital para detectar el sonido cuando este supera un cierto umbral, regulado a través de un potenciómetro ubicado en la placa.

La salida analógica permite obtener una estimación del volumen registrado. Sin embargo, como hemos comentado, este tipo de módulos con micrófono no resultan adecuados para medir el sonido de forma analógica ya que carecen de amplificación.

Este tipo de sensores pueden ser útiles, por ejemplo, para encender un dispositivo cuando se detecte sonido, encender una lámpara con una palmada, o incluso orientar un robot o una torre con servos mediante sonido.

Los pines de conexión son:

- En el centro tenemos la conexión a 5V y a GND (+ y G).
- D0 es una salida digital que actúa a modo de comparador. Si el sonido captado por el micrófono supera un determinado nivel se pone a HIGH.
- A0 es una salida analógica que nos da un valor entre 0 y 1023 en función del volumen del sonido.
- También hay dos LEDs,
 - uno que nos indica si hay alimentación en el sensor
 - y otro que se ilumina si D0 está a HIGH.

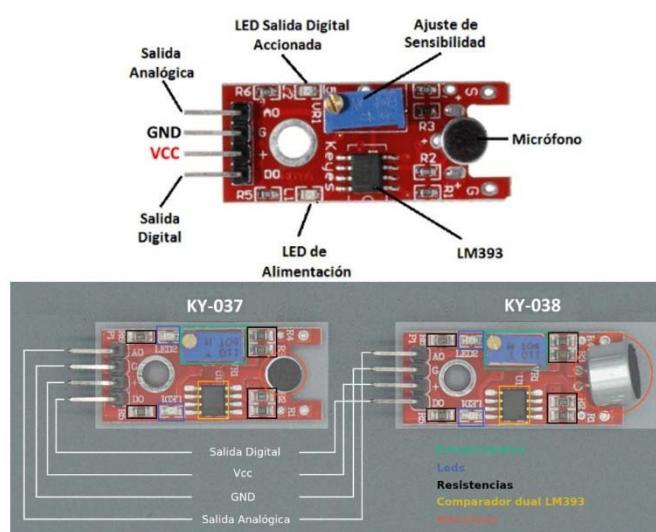


Fig. 83 Esquema de conexión módulo de sensor de sonidos KY038 y KY037.

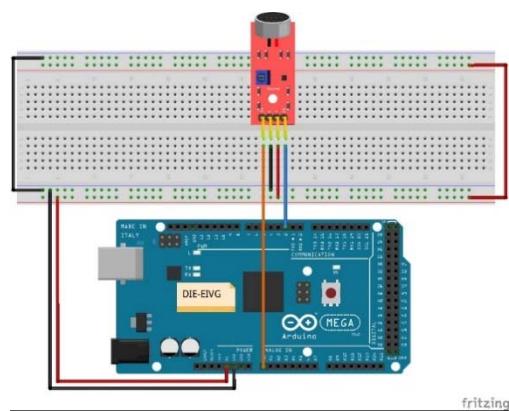
De fábrica viene regulado de tal manera que siempre está activo, hay que mover el preset multivuelta y girar en sentido contrarreloj hasta que se apague el led. De esa manera la salida digital tiene un nivel bajo y solo se activará con un sonido fuerte. Hay que verificar que el led de la izquierda del sensor esté apagado. Se activará el led exterior con un sonido fuerte, con un soplido cerca o con un pequeño golpe en el micrófono. Para ajustar el límite de disparo lo que hacemos es girar el potenciómetro con un destornillador. Tenemos que dejarlo de tal forma que el LED que marca si está accionada la salida digital esté apagado, pero lo más próximo posible al límite en el que se enciende.

- Si lo ajustamos mal y el LED se está encendido, no detectaremos ningún cambio y no podremos reaccionar a ningún estímulo sonoro.
- Si lo ajustamos de forma que esté apagado pero demasiado lejos del límite en el que se enciende, habrá que hacer un sonido muy fuerte para que se active.
- Un valor adecuado para una sensibilidad adecuada es que en la salida analógica haya un valor entre 530 y 600.

21.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 módulo sensor de sonido KY038
- 1 resistencia de $330\ \Omega$
- 1 LED de 5mm

21.4. Esquema de conexión



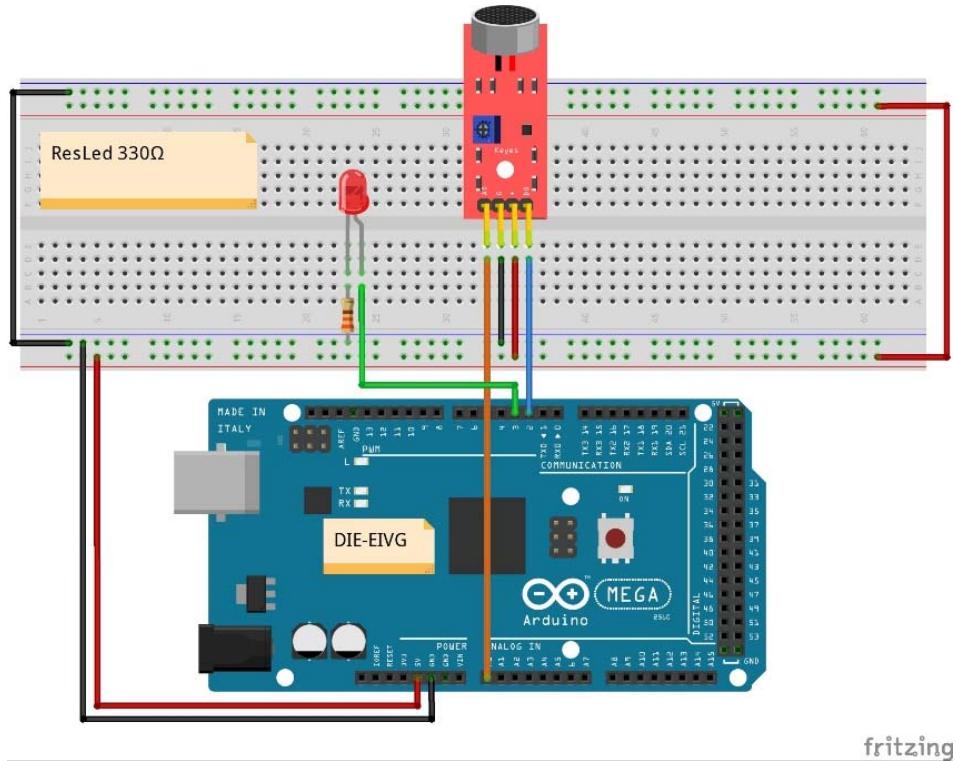


Fig. 85 Esquema de conexión Lección 20, ejercicios 2 y 3.

21.5. Códigos de los programas

1. Leccion_20_Modulo_de_sensor_de_sonidos.ino: Programa que utiliza la salida analógica A0 de KY-038 (o KY-037) para enviar el valor leído por monitor serial. El valor es cercano a 550.
2. Leccion_20_Modulo_de_sensor_de_sonidos_1.ino: Programa que utiliza la salida digital D0 del módulo KY-038 (o KY-037) para mantener encendido o apagado un LED. A partir de un toque en el MIC o una palmada se enciende o se apaga el led. De fábrica viene regulado de tal manera que siempre está activo, hay que mover el preset multivuelta y girar en sentido contrarreloj hasta que se apague el led. De esa manera la salida digital tiene un nivel bajo y solo se activará con un sonido fuerte. Hay que verificar que el led de la izquierda del sensor esté apagado. Se activará o se apagará el led exterior con un sonido fuerte, con un soplido cerca o con un pequeño golpe en el micrófono.
3. Leccion_20_Modulo_de_sensor_de_sonidos_2.ino: Programa que utiliza la salida digital D0 del módulo KY-038 (o KY-037) para mantener encendido o apagado un LED. A partir de dos palmadas se enciende o se apaga el led

21.6. Bibliografía y direcciones de interés

- [1]. <https://aprendiendoarduino.wordpress.com/2018/10/16/modulo-microfono-arduino/>
- [2]. <https://www.prometec.net/sensor-sonido-ky038/>
- [3]. <https://www.luisllamas.es/detectar-sonido-con-arduino-y-microfono-ky-038/>
- [4]. <http://ecuarobot.com/2020/03/18/sensor-de-sonido-con-led-arduino/>
- [5]. <http://www.iescamp.es/miarduino/2016/02/11/sensor-de-sonido-con-arduino/>
- [6]. <https://www.murkyrobot.com/guias/sensores/ky-038>
- [7]. <https://bitwisear.blogspot.com/2018/04/capitulo-13-sensor-de-sonido-ky-038-y.html>
- [8]. <https://uelectronics.com/producto/modulo-ky-038-sensor-microfono/>
- [9]. <https://diyi0t.com/sound-sensor-arduino-esp8266-esp32/>
- [10]. <https://www.flexbot.es/tutorial-control-del-volumen/>
- [11]. <http://codigoelectronica.com/blog/sensor-sonido-arduino>
- [12]. [https://sensorkit.en.joy-it.net/index.php?title=KY-037_Microphone_sensor_module_\(high_sensitivity\)](https://sensorkit.en.joy-it.net/index.php?title=KY-037_Microphone_sensor_module_(high_sensitivity))
- [13]. <https://makersportal.com/blog/2018/5/17/arduino-sound-level-meter>

22. Lección 21 Módulo RC522 RFID

22.1. Objetivo de la práctica

Aprender a usar el módulo lector de RFID RC522 en Arduino MEGA2560 R3. Este módulo utiliza el bus de interfaz periférico Serial (SPI) para comunicarse con controladores tales como Arduino.

22.2. Introducción al componente

El módulo de RFID es un lector de tarjeta inteligente que permite entre otros, para activar un mecanismo cuando la tarjeta correcta se presenta al lector. El RFID (Identificador por radiofrecuencia) es un conjunto de tecnologías diseñadas para obtener una información almacenada en un dispositivo denominado etiqueta (tag) a distancia de forma inalámbrica. Las etiquetas RFID están disponibles en una gran variedad de formatos, tales como pegatinas adheribles, tarjetas, llaveros.

El sistema RFID consta de dos componentes principales: una etiqueta RFID o transpondedor y un lector RFID o transceptor. Las típicas pegatinas RFID que van en muchos productos no tienen ninguna batería o fuente de alimentación. Pero el módulo RC522 debe estar conectado a la red eléctrica para funcionar.

El RFID puede operar en cuatro bandas de frecuencia, siendo la más frecuente 13.56 Mhz, Alta frecuencia o HF. Estos sistemas RFID funcionan en un rango de centímetros, aunque la lectura máxima podría ser en torno a unos 30-35 cm.

La conexión es sencilla. Simplemente alimentamos el módulo desde Arduino mediante **3.3V** y **GND**. Por otro lado, conectamos los pines del bus SPI a los correspondientes de Arduino.



Fig. 86 Esquema de conexión del módulo lector de RFID RC522.

- SDA/SS: el pin actúa como entrada de señal cuando la interfaz SPI está habilitada. Si la interfaz I2C está activa actúa como entrada de datos y como entrada de datos serie cuando la interfaz UART está habilitada.
- SCK: señal de reloj de la interfaz SPI.
- MOSI: entrada en la interfaz SPI.
- MISO: este pin tiene tres funciones. Cuando la interfaz SPI está habilitada funciona como salida de esclavo y entrada de máster. Cuando está activada la interfaz I2C funciona como señal de reloj y como salida serie cuando la interfaz UART está habilitada.
- RQ: pin de interrupción que alerta al microcontrolador cuando una etiqueta RFID se acerca al lector RFID RC522.
- GND: pin de tierra o GND.
- RST: es un pin para encender y apagar el módulo. Mientras el pin esté en estado LOW se mantendrá apagado sin apenas consumir. Cuando el estado cambia a HIGH el RC522 se reinicia.

- VCC: pin de alimentación del lector RFID RC522. Admite un voltaje de alimentación entre 2,5V y 3,3V.

Incluimos la librería RFID:

```
#include <MFRC522.h>
MFRC522 mfrc522(pinSS, pinRST); // crea objeto mfrc522
```

22.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 módulo RC522 RFID
- 1 lector MFRC522
- 1 LED RGB de cátodo común.
- 3 resistencias de 330Ω

22.4. Esquema de conexión

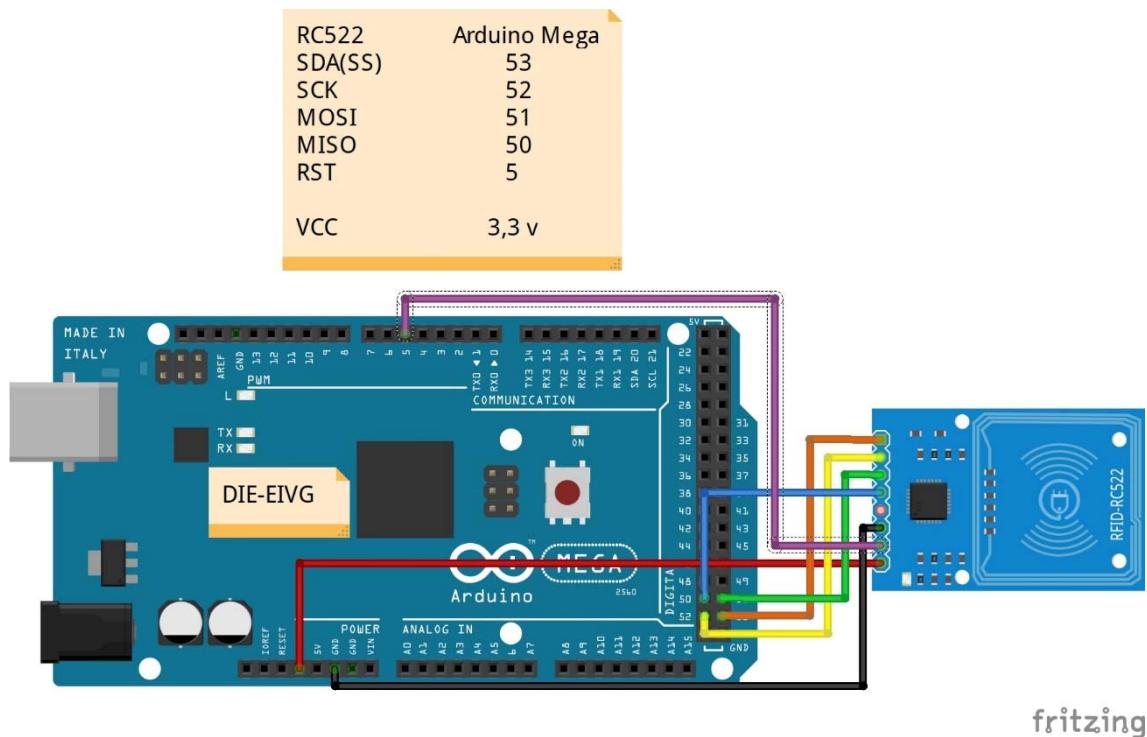


Fig. 87 Esquema de conexión Lección 21, ejercicio .

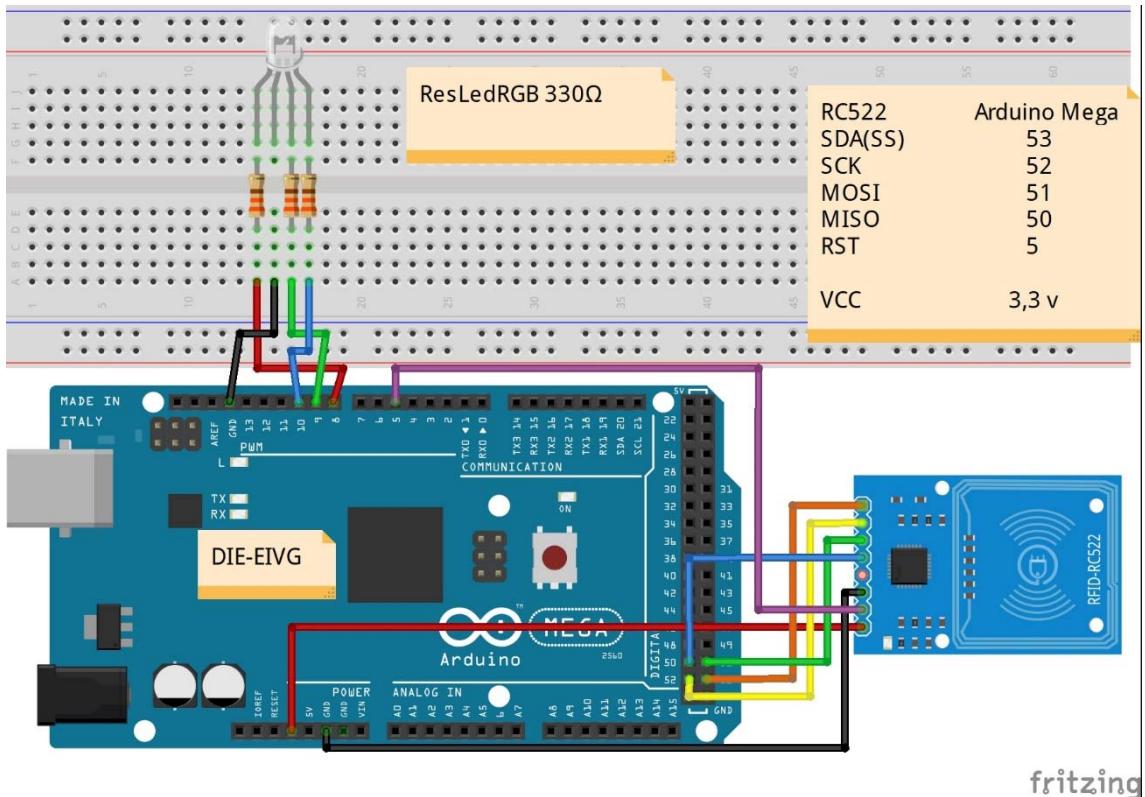


Fig. 88 Esquema de conexión Lección 21, ejercicio .

22.5. Códigos de los programas

1. Leccion_21_Modulo_RC522_RFID.ino: Programa que obtiene el UID de la tarjeta o llavero del kit RFID RC522 y lo muestra en monitor serie. Requiere instalar librería MFRC522.
2. Leccion_21_Modulo_RC522_RFID_1.ino: Programa que muestra en monitor serie un mensaje de bienvenida o de acceso denegado a la tarjeta o llavero del kit RFID RC522 activo. Se activa el UID de la tarjeta o del llavero con el valor leído en programa anterior Requiere instalar librería MFRC522.
3. Leccion_21_Modulo_RC522_RFID_2.ino: Programa que identifica el usuario (tarjeta o llavero del kit) y hace parpadear un led RGB con diferente color según qué usuario sea y parpadeo rojo si no tiene acceso. También muestra en el monitor serie un saludo de bienvenida.

22.6. Bibliografía y direcciones de interés

- [1]. <https://www.luisllamas.es/arduino-rfid-mifare-rc522/>
- [2]. <https://programarfacil.com/blog/arduino-blog/lector-rfid-rc522-con-arduino/>

- [3]. <https://bitwisear.blogspot.com/2018/08/capitulo-40-rfid-rc522-kit-lector.html>
- [4]. <https://www.prometec.net/arduino-rfid/>
- [5]. <https://hetpro-store.com/TUTORIALES/modulo-lector-rfid-rc522-rf-con-arduino/>
- [6]. <https://www.flexbot.es/tutorial-tarjetas-rfid/>
- [7]. <https://aprendiendoarduino.wordpress.com/tag/rc522/>
- [8]. <https://randomnerdtutorials.com/security-access-using-mfrc522-rfid-reader-with-arduino/>
- [9]. <https://www.aranacorp.com/es/uso-de-un-modulo-rfid-con-arduino/>
- [10]. https://naylampmechatronics.com/blog/22_tutorial-modulo-lector-rfid-rc522.html

23. Lección 22 Pantalla LCD1602

23.1. Objetivo de la práctica

Aprender a conectar y usar una pantalla LCD LCD1602 alfanumérica. La pantalla tiene una retroiluminación de LED y puede mostrar dos filas con hasta 16 caracteres en cada fila. Se puede visualizar todas las letras de alfabeto, letras de alfabeto griego, signos de puntuación, símbolos matemáticos etc. También es posible visualizar símbolos creados por el usuario. Entre otras características útiles es el desplazamiento automático de mensajes (a la izquierda y a la derecha), aparición del cursor, retroiluminación LED etc.

En esta práctica, utilizaremos la librería LiquidCrystal que viene integrada en la instalación del IDE de arduino y nos permite controlar toda clase de pantallas compatibles con el controlador HD44780 ya sean de 16×2 , 20×4 u otras configuraciones de caracteres.

23.2. Introducción al componente

Un display LCD (Liquid Crystal Display) es un display alfanumérico de matriz de puntos (dot-matrix) que sirve para mostrar mensajes a través de caracteres como letras, números o símbolos. La placa del display viene equipado con un microcontrolador (normalmente el modelo HD44780) que se encarga de generar los caracteres, polarizar la pantalla, desplazar el cursor... Además, también viene equipado con una memoria ROM donde están almacenados los caracteres a través de una matriz de puntos, y una memoria RAM donde se pueden almacenar caracteres creados por nosotros. Estos displays

disponen de unos pines para conectar un microcontrolador (en nuestro caso Arduino) para poder dar instrucciones al display.

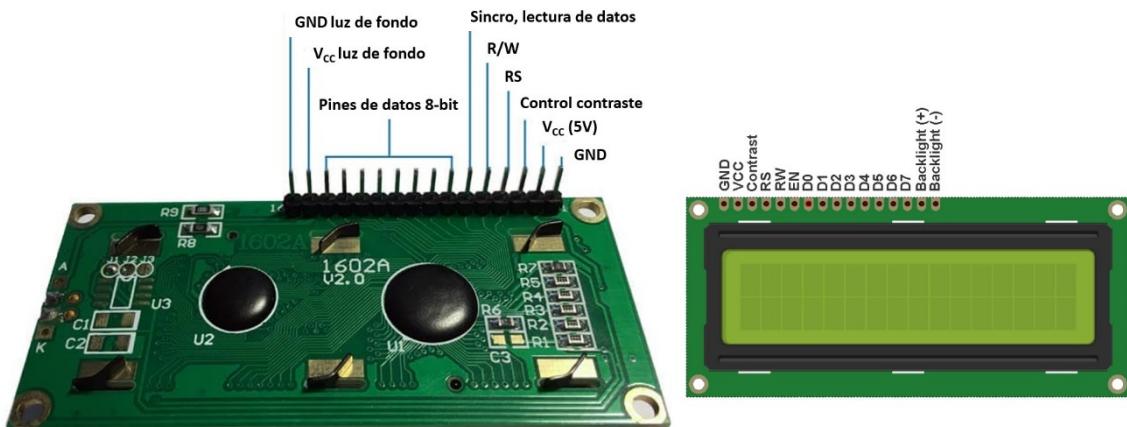


Fig. 89 Esquema de conexión Lección 22, ejercicios 1, 2, 3 y 4.

Los pines del LCD 1602a Arduino con controlador HD44780 se muestra a continuación:

- El pin RS controla en qué parte de la memoria LCD se están escribiendo los datos. Es para mandarle información al LCD de lo que se quiere hacer. Si deseamos mandarles comandos de control o mostrar caracteres en la pantalla. Es el selector de registro, commuta entre el envío de caracteres y el envío de comandos de control, como por ejemplo borrar la pantalla o desplazar el cursor.
- El pin RW de “lectura/escritura” (R/W) selecciona el modo de lectura o de escritura. Si el pin es conectado a +5V el modo de lectura es activado, si el pin es conectado a GND el modo de escritura es activado. Normalmente trabaja conectado a GND para indicarle en todo momento que queremos escribir.
- El pin EN para habilitar “enable”, este habilita los registros. Es para activar que la pantalla reciba información.
- 8 pines de datos “D00-D07” donde se envían bits para escribir en un registro, también pueden ser usados para leer un registro. Es un bus de datos de 8 bits, pero nosotros solamente necesitamos 4 bits. Del D4 al D7 del LCD conectados a cualquiera de los pines digitales del Arduino. Por este bus enviaremos los datos a la pantalla. Los pines restantes del bus de datos se dejan sin conectar. Normalmente se usa el modo de 4 bit's por necesitar menor cableado.

- Pin Contraste, es el contraste del display LCD Arduino. Conectando un potenciómetro de $10K\Omega$ podremos regular el contraste de la pantalla LCD. Si no conectamos este pin directamente no veríamos nada.
- Pin A y Pin K “de retro-iluminación” (Bklt+ y Bklt-) que le permiten controlar la retroiluminación, son para la luz de fondo del LCD. Los conectaremos igual que GND y Vcc.
- Pin de alimentación (+5V y GND). Lo conectaremos a 5V.

También podemos conectar la pantalla mediante el controlador de LCD I2C que es un dispositivo que nos permite controlar una pantalla a través del bus I2C, usando únicamente dos cables. Usando la pantalla directamente desde Arduino requería emplear una gran cantidad de pines de Arduino, lo que supone un enorme desperdicio de recursos, que deberían estar ocupados en cosas mucho más importantes que encender un simple display.

Librería LiquidCrystal de Arduino

A través de la librería de Arduino LiquidCrystal podemos controlar un display LCD con Arduino. En este enlace encontraréis todas las funciones de esta librería.

<http://arduino.cc/en/Reference/LiquidCrystal>

A continuación, se explica las funciones básicas:

- LiquidCrystal “nombre_variable” (rs, enable, d4, d5, d6, d7): A través de esta función se crea una variable de tipo LiquidCrystal. Entre paréntesis pondremos los pines del Arduino correspondientes a los pines del display (RS, Enable y los 4 pins usados del bus de datos). `LiquidCrystal lcd(7, 6, 5, 4, 3, 2);`
- begin(columnas, filas): Inicializa la interfaz del LCD y establece las dimensiones de la pantalla. `lcd.begin(16, 2);`
- home(): Mueve el cursor a la primera posición de la pantalla (0, 0). `lcd.home();`
- setCursor(columna, fila): Posicionar el cursor de la pantalla LCD. Mueve el cursor a la segunda línea (1) primera columna (0). `lcd.setCursor (0,1);`
- display(): Muestra el texto. `lcd.display();`
- noDisplay(): Oculta el texto. `lcd.noDisplay();`
- print(“mensaje”): Escribe texto a la pantalla LCD. `lcd.print("Pantalla activa!");`

- `clear()`: Limpia la pantalla LCD y posiciona el cursor en la parte superior izquierda.

23.3. Componentes necesarios

- Placa Arduino Mega
 - Cable USB
 - Protoboard
 - 1 módulo LCD1602
 - 2 potenciómetros $10k\Omega$
 - 1 resistencia $220\ \Omega$

23.4. Esquema de conexión

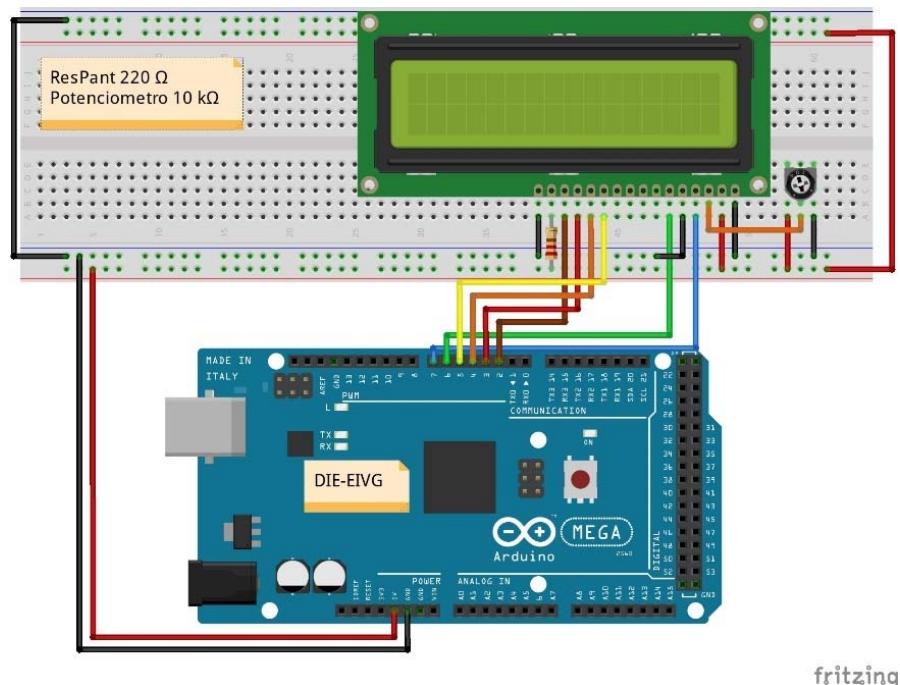
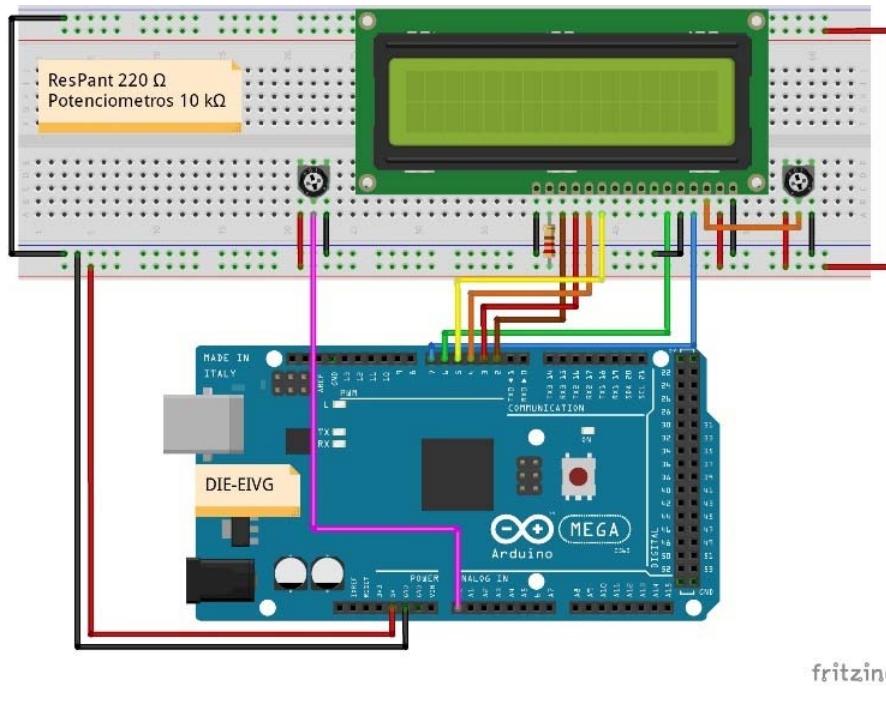


Fig. 90 Esquema de conexión Lección 22, ejercicios 1, 2, 3 y 4.



fritzing

Fig. 91 Esquema de conexión Lección 22, ejercicio 5.

23.5. Códigos de los programas

1. Leccion_22_Pantalla_LCD_1602.ino: Programa que muestra en el módulo LCD el tiempo transcurrido en segundos desde que se inició el programa. Requiere instalar librería LiquidCrystal.
2. Leccion_22_Pantalla_LCD_1602_1.ino: Programa que hace parpadear un texto y usa las funciones scroll para desplazar un texto horizontalmente en la pantalla.
3. Leccion_22_Pantalla_LCD_1602_2.ino: Programa que usa las funciones scroll para desplazar un texto en las dos líneas de la pantalla, horizontalmente.
4. Leccion_22_Pantalla_LCD_1602_3.ino: Programa que muestra un texto en la primera línea de la pantalla y unos caracteres definidos en la segunda.
5. Leccion_22_Pantalla_LCD_1602_4.ino: Programa que muestra el valor de un texto en la primera línea de la pantalla y el valor de un potenciómetro en la segunda.

23.6. Bibliografía y direcciones de interés

- [1]. <https://www.geekfactory.mx/tutoriales/tutoriales-arduino/pantalla-lcd-16x2-con-arduino/>

- [2]. https://naylampmechatronics.com/blog/34_tutorial-lcd-conectando-tu-arduino-a-un-lcd1602-y-lcd2004.html
- [3]. <https://programarfacil.com/tutoriales/fragmentos/arduino/texto-en-movimiento-en-un-lcd-con-arduino/>
- [4]. <https://www.prometec.net/displays-lcd/>
- [5]. <https://jorgesanz.es/conectar-pantalla-lcd-a-arduino-uno/>
- [6]. <https://controlautomaticoeducacion.com/arduino/lcd/>
- [7]. <https://www.instructables.com/Arduino-con-pantalla-LCD-1602/>
- [8]. <http://www.infotronikblog.com/2009/11/caracteres-especiales-lcd.html>
- [9]. <http://diymakers.es/aprender-usar-un-display-lcd/>
- [10]. <https://descubrearduino.com/como-usar-un-lcd/>
- [11]. <https://www.luisllamas.es/arduino-lcd-hitachi-hd44780/>
- [12]. <https://bitwisear.blogspot.com/2018/04/capitulo-10-modulo-lcd-1602a.html>
- [13]. <https://www.luisllamas.es/arduino-lcd-i2c/>
- [14]. <https://bitwisear.blogspot.com/2018/05/capitulo-35-adaptador-lcd-i2c.html>
- [15]. <https://www.geekfactory.mx/tutoriales/tutoriales-arduino/lcd-16x2-por-i2c-con-arduino/>
- [16]. https://naylampmechatronics.com/blog/35_tutorial-lcd-con-i2c-controla-un-lcd-con-solo-dos-pines.html
- [17]. <https://www.mikroe.com/ebooks/microcontroladores-pic-programacion-en-c-con-ejemplos/componentes-adicionales>
- [18]. <https://www.zonamaker.com/arduino/modulos-sensores-y-shields/tipos-de-lcd-para-arduino>

24. Lección 23 Termómetro con Termistor NTC

24.1. Objetivo de la práctica

Estudiar el sensor analógico, termistor NTC (resistencia variable con coeficiente de temperatura negativo, si aumenta la temperatura disminuye la resistencia). El termistor NTC es un sensor de temperatura resistivo, el cual, al cambiar su temperatura, varía su resistencia eléctrica.

24.2. Introducción al componente

El termistor es un transductor (sensor) que se fabrica con óxidos metálicos. Esto se traduce en que entre las patillas de un termistor siempre podremos medir una resistencia.

Un termistor está formado por el encapsulado y sus dos terminales, a efectos prácticos, lo podemos entender como una simple resistencia. Una resistencia que varía su valor en función de la temperatura a la que se ve sometido.

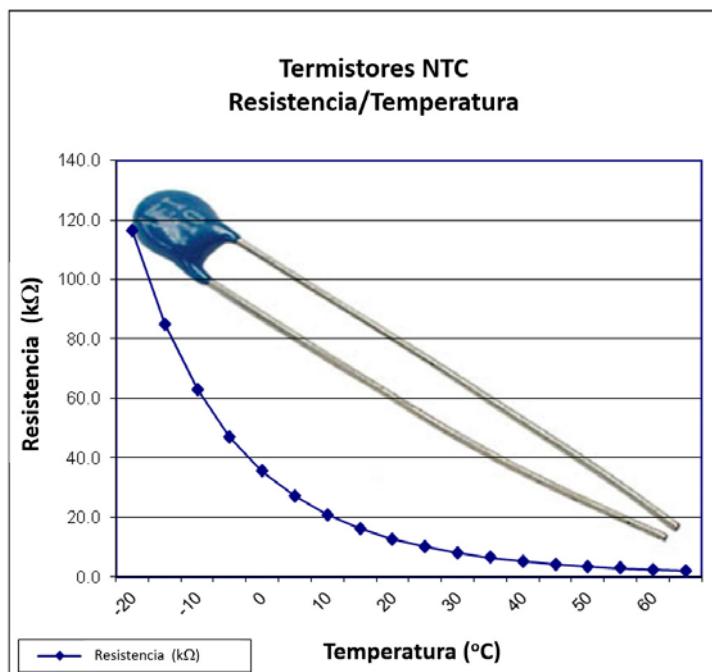


Fig. 92 Curva Resistencia vs Temperatura.

Podemos encontrar NTC que tienen el valor de 10K, esto quiere decir que cuando la temperatura es 25º su resistencia es de 10K, como por ejemplo la gráfica de arriba. Se toma como temperatura de referencia 25º.

El principal problema que plantean los termistores a la hora de emplearlos en aplicaciones de control de temperatura es sin duda la no linealidad de la curva R/T. Por contra, ofrecen gran sensibilidad, precisión y bajo coste frente a otro tipo de sensores. Su integración en sistemas domóticos, y en este caso en Arduino, exige una modelización de la curva R/T, teniendo en cuenta que, en este tipo de sensores, una pequeña variación de la temperatura provoca una gran variación de la resistencia asociada. En este caso, al ser NTC, un aumento de temperatura provoca una gran disminución de la resistencia y viceversa.

El método más corriente para modelizar la curva R/T de los termistores NTC consiste en aplicar la ecuación de Steinhart-Hart, que relaciona la temperatura con la resistencia de la forma:

$$\frac{1}{T} = a + b \ln(R) + c(\ln(R)^3)$$

a, b y c son parámetros específicos de cada tipo de sensor, y por tanto deben ser determinados o especificados por el fabricante

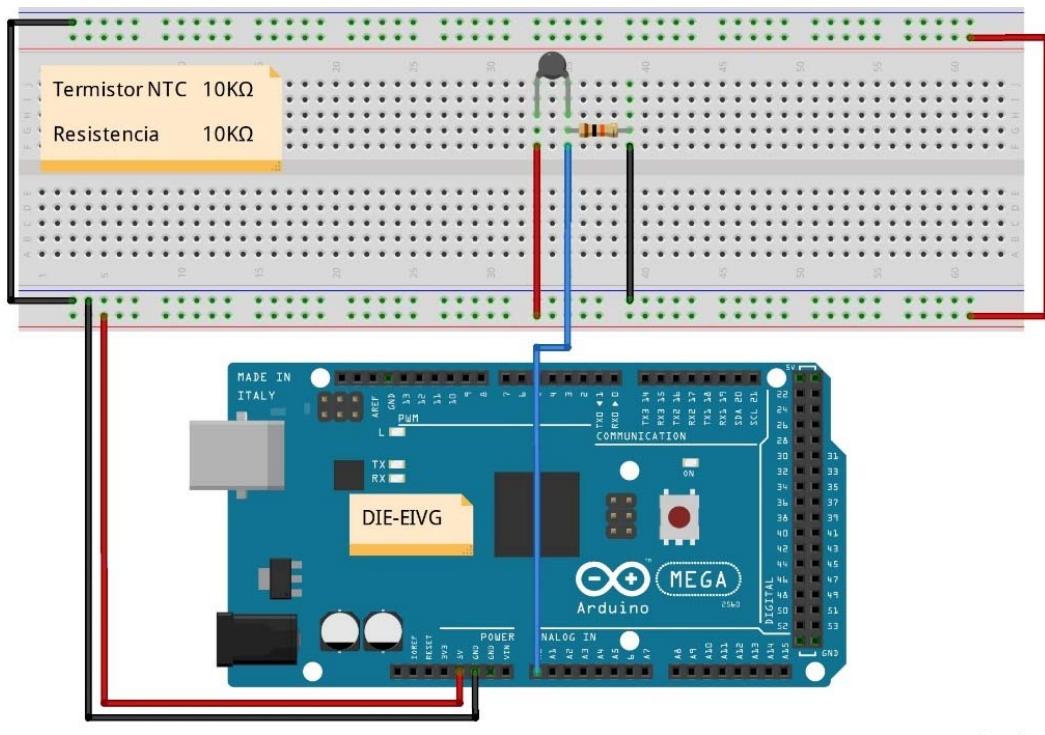
Coeficientes de S-H en pagina:

<http://www.thinksrs.com/downloads/programs/Therm%20Calc/NTCCalibrator/NTCcalculator.htm>

24.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 módulo LCD1602
- 1 resistencia 220 Ω
- 1 termistor NTC 10 kΩ
- 1 potenciómetro 10 kΩ

24.4. Esquema de conexión



fritzing

Fig. 93 Esquema de conexión Lección 23, ejercicio 1.

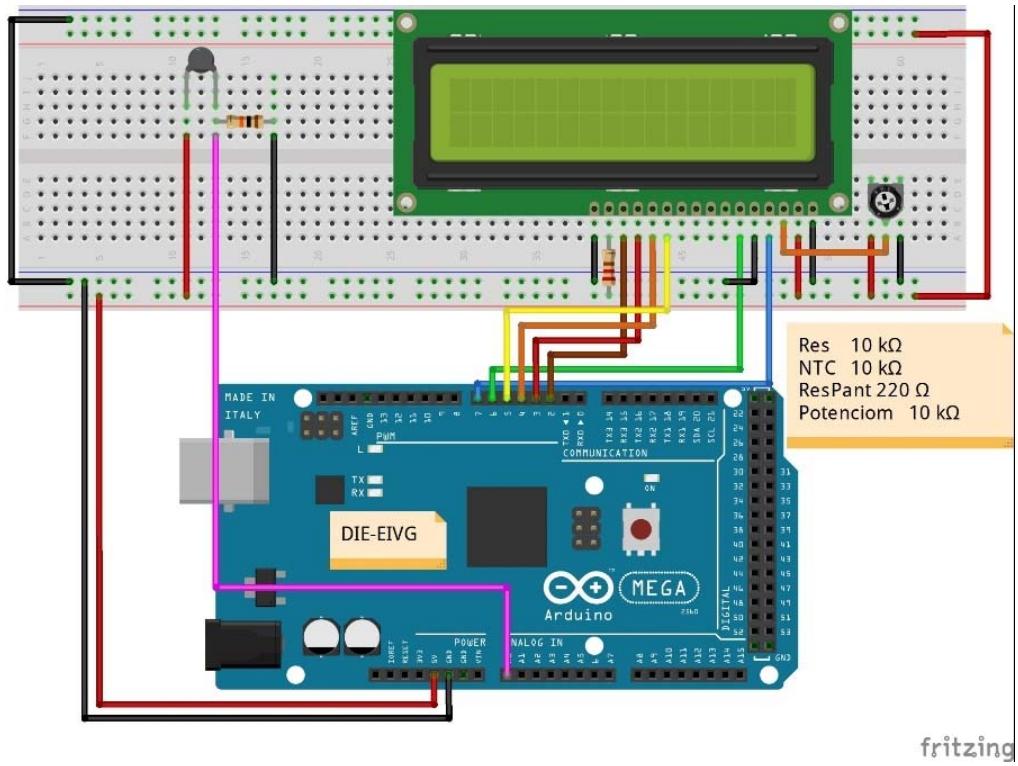


Fig. 94 Esquema de conexión Lección 23, ejercicio 2.

24.5. Códigos de los programas

1. Leccion_23_Termometro.ino: Programa que muestra en el monitor serie la temperatura en grados Celsius. Basándose en la ecuación de Steinhart-Hart:
$$1/T = C1 + c2 \ln(R) + c3[\ln(R)]^3$$
: Coeficientes de S-H en página:
<http://www.thinksrs.com/downloads/programs/Therm%20Calc/NTCCalibrator/NTCcalculator.htm>. Son los valores que hay por defecto
 2. Leccion_23_Termometro_1.ino: Programa que muestra en el módulo LCD la temperatura tomada desde un termistor NTC. En grados Celsius y Fahrenheit. Requiere instalar librería LiquidCrystal

24.6. Bibliografía y direcciones de interés

- [1]. https://en.wikipedia.org/wiki/Steinhart%E2%80%93Hart_equation
 - [2]. <https://bitwisear.blogspot.com/2018/05/capitulo-34-termistor-como-sensor-de.html>
 - [3]. <https://www.geekfactory.mx/tutoriales/tutoriales-arduino/termistor-ntc-con-arduino-como-sensor-de-temperatura/>

- [4]. <https://www.instructables.com/Temperatura-con-Termistor-NTC-10k-y-Arduino/>
- [5]. <https://www.instructables.com/Ejemplo-b%C3%A1sico-de-termistor-NTC-y-Arduino/>
- [6]. <https://www.luisllamas.es/medir-temperatura-con-arduino-y-termistor-mf52/>
- [7]. <https://electrocrea.com/blogs/tutoriales/18193247-sensor-de-temperatura-con-termistor-ntc-10k>
- [8]. <https://www.circuitbasics.com/arduino-thermistor-temperature-sensor-tutorial/>
- [9]. <https://www.rinconingenieril.es/como-usar-un-termistor-ntc/>
- [10]. <http://www.elec2.es/?p=580>
- [11]. <http://kio4.com/arduino/17NTC.htm>
- [12]. <https://learn.adafruit.com/thermistor/using-a-thermistor>
- [13]. <https://create.arduino.cc/projecthub/iasonas-christoulakis/make-an-arduino-temperature-sensor-thermistor-tutorial-b26ed3>

25. Lección 24 Control de Display de Ocho LED con 74HC595

25.1. Objetivo de la práctica

Aprender a utilizar un display de ocho LEDs con un 74HC595 (registro de desplazamiento). El 595 controla esencialmente ocho pines de salida separados, usando sólo tres pines de entrada.

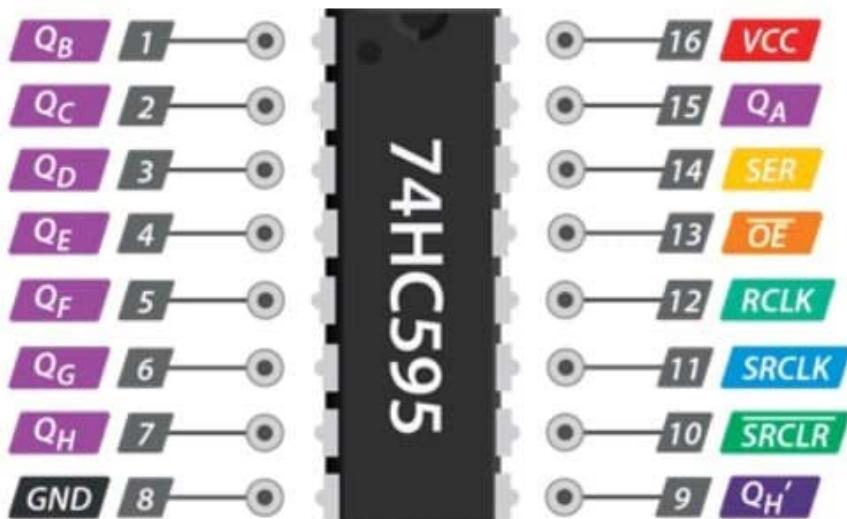
25.2. Introducción al componente

Un registro de desplazamiento te permite ampliar el número de pines de E/S que puedes utilizar de tu Arduino.

El 74HC595 es un circuito integrado bastante sencillo. Se trata de un registro de desplazamiento de 8-bit, es decir, tiene 8 biestables para almacenar 8 bits. Las patillas de este chip se pueden ver en la imagen inferior, con Vcc y GND para la alimentación, y luego las marcadas como Q que son las de datos. El resto corresponden a señales de reloj/control.

La entrada la tiene en serie y la salida en paralelo. Por tanto, con una sola entrada, se pueden controlar a la vez esas 8 salidas. Solo necesitarás tres pines del

microcontrolador usado (p.e.: Arduino) para manejarlo. Esas son Latch, Clock y Data. Latch es el pin 13 en este caso, aunque puede variar, por eso debes consultar el datasheet de tu fabricante. Clock puede estar en el 11 y el bit de datos es el 14.



74HC595 / Pinout

Fig. 95 Esquema de conexión Lección 24.

- GND debería estar conectado a la tierra de Arduino.
- VCC es la fuente de alimentación para el registro de cambio 74HC595 que conectamos al pin 5V de Arduino.
- El pin SER (Serial Input) se utiliza para alimentar los datos en el registro de cambio de un bit a la vez.
- SRCLK (Shift Register Clock) es el reloj del registro de cambio. El 595 es impulsado por el reloj en el borde ascendente. Esto significa que, para desplazar bits en el registro de desplazamiento, el reloj debe estar ALTO (HIGH). Y los bits se transfieren en el borde ascendente del reloj.
- RCLK (Register Clock / Latch) es un pin muy importante. Cuando se maneja en ALTO (HIGH), el contenido del Registro de Desplazamiento se copia en el Registro de Almacenamiento / Reloj; que finalmente se muestra en la salida. Así que la clavija del pestillo puede ser vista como el paso final en el proceso de ver nuestros resultados en la salida, que en este caso son los LEDs.

- La clavija SRCLR (Shift Register Clear) nos permite reiniciar todo el Registro de Desplazamiento, haciendo que todos sus bits sean 0, a la vez. Este es un pin de lógica negativa, así que, para realizar este restablecimiento, necesitamos poner el pin SRCLR en LOW. Cuando no se requiere un restablecimiento, este pin debe ser ALTO (HIGH).
- OE (Output Enable) también es de lógica negativa: Cuando el voltaje en él es ALTO (HIGH), los pines de salida se deshabilitan/se ajustan a un estado de alta impedancia y no permiten que la corriente fluya. Cuando OE tiene bajo voltaje, los pines de salida funcionan normalmente.
- QA-QH (Output Enable) son los pines de salida y deben ser conectados a algún tipo de salida como LEDs, 7 segmentos, etc.
- El Pin ‘QH’ da salida al bit 7 del ShiftRegister. Está ahí para que podamos encadenar 595s: si conectas este QH’ al pin SER de otro 595, y das a ambos ICs la misma señal de reloj, se comportarán como un solo IC con 16 salidas. Por supuesto, esta técnica no está limitada a dos ICs, puedes encadenar tantos como quieras, si tienes suficiente potencia para todos ellos.

```
int latchPin = 11; // (Pin 12) ST_CP [RCK-RCLK] en 74HC595. El pin latch del
74HC595 está conectado al pin digital 11
int clockPin = 9; // (Pin 11) SH_CP [SCK-SRCLR] en 74HC595. El pin del reloj
del 74HC595 está conectado al pin digital 9
int dataPin = 12; // (Pin 14) DS DATA [SER] en 74HC595. El pin de datos del
74HC595 está conectado al pin digital 12
```

25.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 8 resistencias de 330 Ω
- 8 LEDs de 5mm
- 1 módulo IC 74HC595

25.4. Esquema de conexión

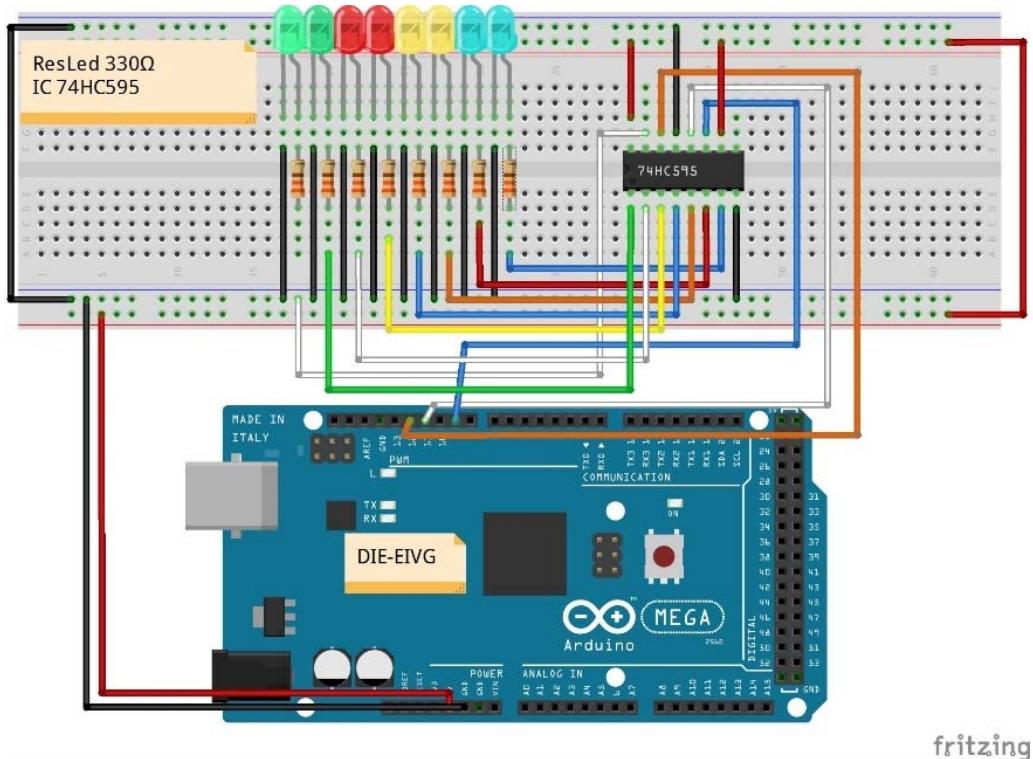


Fig. 96 Esquema de conexión Lección 24.

25.5. Códigos de los programas

1. Leccion_24_Control_de_ocho_LED_con_74HC595.ino: Programa que controla ocho leds con el integrado 74HC595.

25.6. Bibliografía y direcciones de interés

- [1]. <https://programarfacil.com/blog/74hc595-registro-de-desplazamiento-arduino/>
- [2]. <https://www.prometec.net/shift-registers/>
- [3]. <http://robots-argentina.com.ar/didactica/arduino-ampliar-cantidad-de-salidas-digitales-con-74hc595/>
- [4]. <https://descubrearduino.com/74hc595/>
- [5]. <https://www.hwlibre.com/74hc595/>
- [6]. <https://www.ti.com/lit/ds/symlink/sn74hc595.pdf>
- [7]. <https://parzibyte.me/blog/2017/11/07/tutorial-arduino-74hc595-leds/>
- [8]. <https://learn.adafruit.com/adafruit-arduino-lesson-4-eight-leds/the-74hc595-shift-register>

- [9]. <https://lastminuteengineers.com/74hc595-shift-register-arduino-tutorial/>
- [10]. <https://protostack.com.au/2010/05/introduction-to-74hc595-shift-register-controlling-16-leds/>
- [11]. <https://www.instructables.com/Arduino-16-LEDs-using-two-74HC595-shift-registers-/>

26. Lección 25 Fotocélula LDR

26.1. Objetivo de la práctica

Aprender a medir la intensidad de la luz utilizando una entrada analógica.

26.2. Introducción al componente

Una fotorresistencia o LDR (Light Depending Resistor, o resistencia dependiente de la luz) es un componente foto electrónico cuya resistencia varía en función de la luz que incide en él. Esta resistencia es muy baja, de unos pocos Ω s con una luz intensa incide en él y va creciendo fuertemente a medida que esa luz decrece (Mas luz = menor resistencia eléctrica; Menos luz = mayor resistencia eléctrica).



Fig. 97 Fotocélula LDR.

Una LDR disminuye su resistencia a medida que aumenta la luz sobre él. Los valores típicos son de $1 \text{ M}\Omega$ en total oscuridad, a $50-100 \Omega$ bajo luz brillante.

Por otro lado, la variación de la resistencia es relativamente lenta, de 20 a 100 ms en función del modelo. Esta lentitud hace que no sea posible registrar variaciones rápidas, como las producidas en fuentes de luz artificiales alimentadas por corriente alterna. Este comportamiento puede ser beneficioso, ya que dota al sensor de una gran estabilidad.

Finalmente, las LDR no resultan adecuados para proporcionar una medición de la iluminancia, es decir, para servir como luxómetro. Esto es debido a su baja precisión, su

fuerte dependencia con la temperatura y, especialmente, a que su distribución espectral no resulta adecuada para la medición de iluminancia.

Formas de conectar una LDR

Existen dos formas básicas para la conexión de nuestra LDR, pueden ser utilizadas dependiendo del fin deseado. Si disponemos de un controlador es posible modificar los resultados mediante programación.

- Mayor luz, mayor voltaje: Al conectar la LDR al nodo positivo de nuestra fuente de voltaje tendremos que, al incidir una mayor cantidad de luz provocará una menor caída de voltaje o diferencial de potencial entre la fuente y el pin de referencia (V_{salida}), por lo tanto, se tendrá una lectura mayor.
- Mayor luz, menor voltaje: La LDR se conecta al nodo de GND y provocará un comportamiento opuesto al punto anterior.

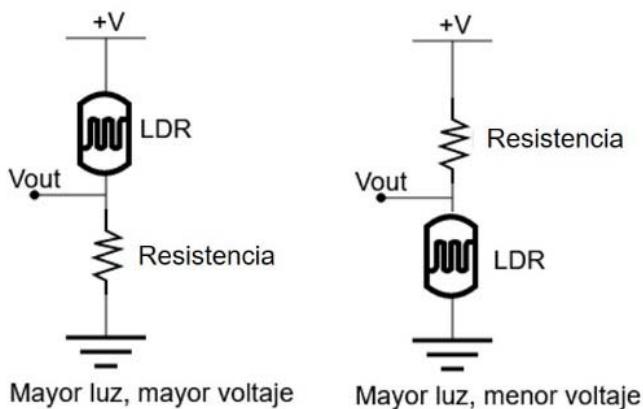


Fig. 98 Fotocélula LDR.

Se puede sustituir la resistencia por un potenciómetro si vamos a cambiar de un estado a otro, por lo tanto, la iluminación va a variar, con esto evitamos modificar el código de programación.

26.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 3 resistencias de $330\ \Omega$

- 1 resistencia de 10 k Ω
 - 3 LEDs de 5mm
 - 1 fotorresistor o fotocélula

26.4. Esquema de conexión

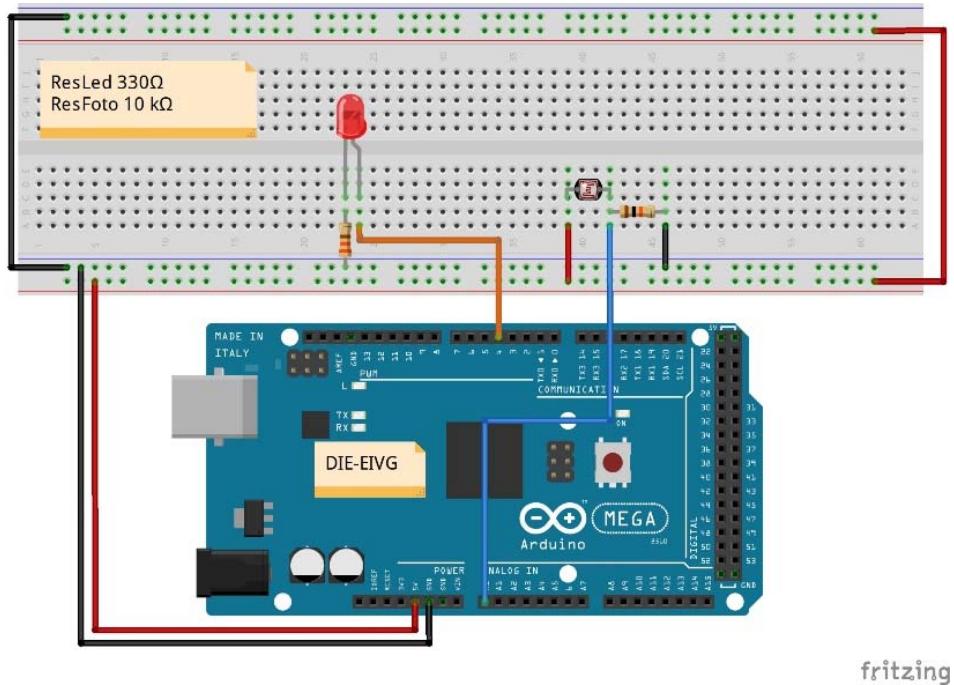


Fig. 99 Esquema de conexión Lección 25, ejercicios 1 y 2.

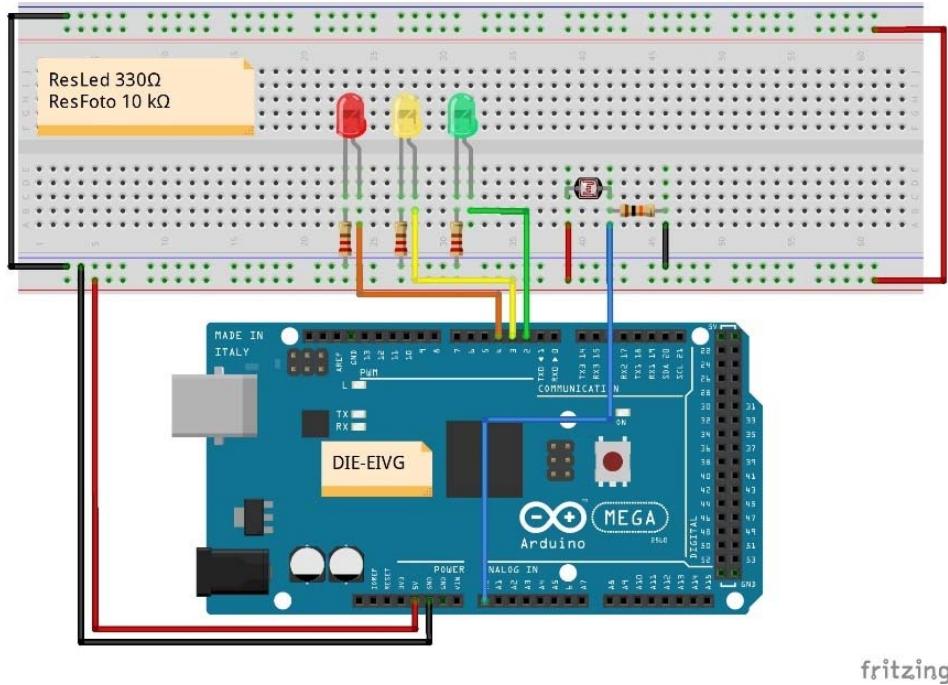


Fig. 100 Esquema de conexión Lección 25, ejercicios 3 y 4.

26.5. Códigos de los programas

1. Leccion_25_Fotocelula_LDR.ino: Programa que utiliza una fotocélula o una fotorresistencia para aumentar la luminosidad de un LED cuando hay baja intensidad de luz ambiente.
2. Leccion_25_Fotocelula_LDR_1.ino: Programa que utiliza una fotocélula o una fotorresistencia para cambiar el brillo del LED de forma inversamente proporcional a la luz captada.
3. Leccion_25_Fotocelula_LDR_2.ino: Programa que utiliza una fotocélula o una fotorresistencia para iluminar 1, 2 o 3 leds dependiendo de la luminosidad. Devuelve el valor leído al monitor serie en el IDE de Arduino
4. Leccion_25_Fotocelula_LDR_3.ino: Programa que enciende los leds apropiados de acuerdo con el valor obtenido, un led, dos leds o los tres. Mapeando este valor entre 0 y 255.

26.6. Bibliografía y direcciones de interés

- [1]. <https://www.geekfactory.mx/tutoriales/tutoriales-arduino/tutorial-arduino-con-fotoresistencia-ldr/>
- [2]. <https://www.luisllamas.es/medir-nivel-luz-con-arduino-y-fotoresistencia-ldr/>
- [3]. <https://www.programoergosum.com/cursos-online/arduino/257-entradas-analogicas-con-arduino/encendido-nocturno>
- [4]. <http://piensa3d.com/tutorial-programacion-arduino-ldr-sensor-luz/>
- [5]. <https://www.askix.com/como-utilizar-un-fotoresistor-o-fotocelula-arduino-tutorial.html>
- [6]. <https://www.askix.com/aprender-a-programar-un-arduino-fotocelula.html>
- [7]. <https://elsotanodemem.wordpress.com/2015/12/29/practica-22-rele-enciende-luz-con-fotocelula/>
- [8]. <https://bitwisear.blogspot.com/2018/04/capitulo-14-sensor-crepuscular-por.html>
- [9]. <https://bitwisear.blogspot.com/2018/04/capitulo-21-interrupciones-externas.html>
- [10]. <https://desensores.com/sensores-arduino/proyectos-basicos-con-arduino-para-principiantes/sensor-fotorresistencia-arduino/>

[11]. <https://www.mecatronicalam.com/es/tutoriales/sensores/sensor-de-luz/ldr/>

27. Lección 26 74HC595 y Pantalla de 7 Dígitos Segmentado

27.1. Objetivo de la práctica

Aprender a utilizar el registro de desplazamiento 74HC595 para controlar la visualización de los segmentos de una pantalla de 7 dígitos segmentados.

27.2. Introducción al componente

Una pantalla de 7 segmentos es un display formado por 7 LEDs para los segmentos y el LED del punto. Este display nos permite crear números e incluso algunas letras dándole la orden de que LEDs deben estar encendidos y de cuales apagados.

Los displays, pueden ser de cátodo común o de ánodo común, en este caso nos conviene que sean de cátodo común ya que nos permiten programar en lógica positiva.

Conexiones:

- Pin **Q0** del 74HC595 al pin del Segmento **b** del display.
- Pin **Q1** del 74HC595 al pin del Segmento **a** del display.
- Pin **Q2** del 74HC595 al pin del Segmento **f** del display.
- Pin **Q3** del 74HC595 al pin del Segmento **g** del display.
- Pin **Q4** del 74HC595 al pin del Segmento **e** del display.
- Pin **Q5** del 74HC595 al pin del Segmento **d** del display.
- Pin **Q6** del 74HC595 al pin del Segmento **c** del display.
- Pin **Q7** del 74HC595 al pin del Segmento **P** del display.

Para formar los patrones de los dígitos del LED, de 0 a 9:

```
1 = LED encendido, 0 = LED apagado, en este orden:  
pin 74HC595      Q0, Q1, Q2, Q3, Q4, Q5, Q6, Q7  
Asignación a      a, b, c, d, e, f, g, P del LED de siete segmentos  
O sea, que conectamos pin segmento-led con pin 74HC595 a(Q0), b(Q1),  
c(Q2), d(Q3), e(Q4), f(Q5), g(Q6) y P(Q7)  
Declarando la variable byte numero[10]  
//                                BabcdefgP  
byte numero[10] = { B11100110, // = 9  
                   B11111110, // = 8  
                   B10111110, // = 6  
                   B11100000, // = 7  
                   B10110110, // = 5  
                   B01100110, // = 4  
                   B11110010, // = 3  
                   B11011010, // = 2  
                   B01100000, // = 1
```

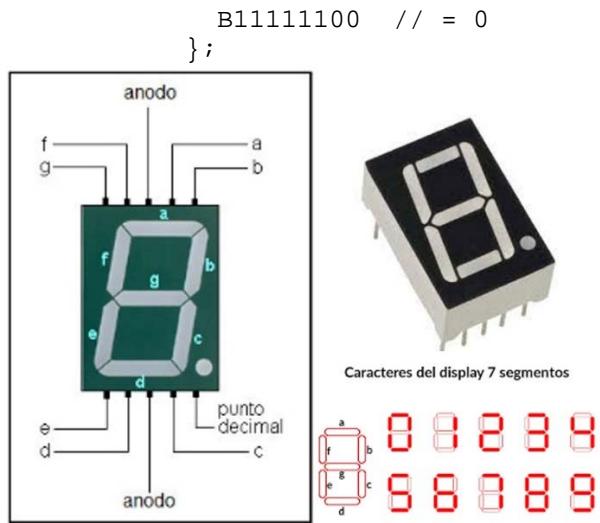


Fig. 101 Pantalla de 7 segmentos.

27.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 módulo IC 74HC595
- 1 pantalla de 1 dígitos y 7 segmentos
- 8 resistencias 220 Ω

27.4. Esquema de conexión

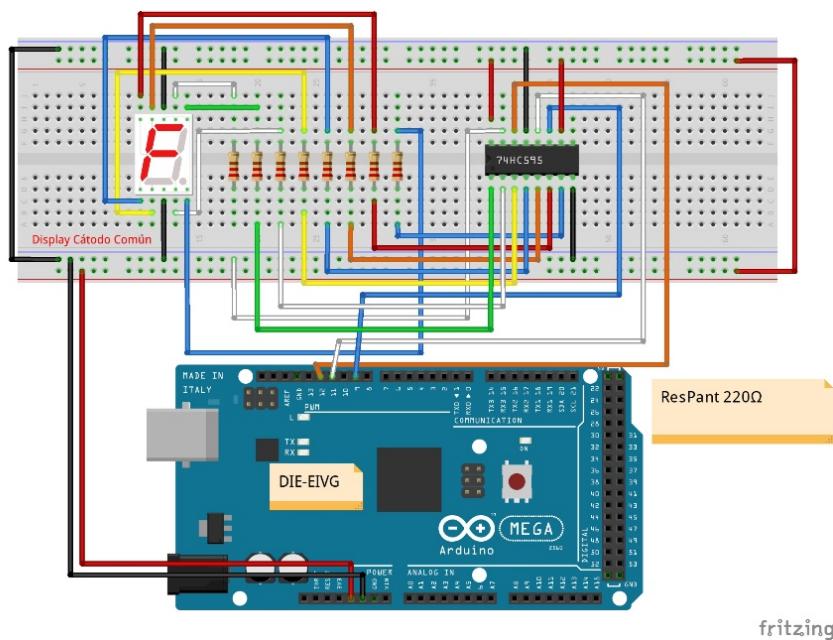


Fig. 102 Esquema de conexión Lección 26.

27.5. Códigos de los programas

1. Leccion_26_74HC595_y_display_Segmentado.ino: Programa que controla 1 pantalla de 7 segmentos de 1 dígito con el integrado 74HC595.

27.6. Bibliografía y direcciones de interés

- [1]. <https://descubrearduino.com/74hc595/>
- [2]. <https://www.hwlible.com/74hc595/>
- [3]. <https://www.electrontools.com/Home/WP/registro-de-desplazamiento-74hc595/>
- [4]. https://www.palimpalem.com/1/Alberto_Pumar_Tutoriales/index.html?bod y2.html
- [5]. <https://sites.google.com/site/angmuz/home/proyecto-36-2-clock-thermometer-with-arduino-5-digit-7-segment-dvd-display-74hc595>
- [6]. <https://www.programandoamedianoche.com/2014/05/como-utilizar-displays-de-7-segmentos-con-arduino/>
- [7]. <https://forum.arduino.cc/index.php?topic=668356.0>

28. Lección 27 Pantalla de 7 segmentos de cuatro dígitos

28.1. Objetivo de la práctica

Aprender a utilizar una pantalla de 7 segmentos de 4 dígitos. Multiplexando los dígitos.

28.2. Introducción al componente

Usaremos 8 pines para iluminar los 7 segmentos más el punto, y otros 4 pines para indicar a cuál de los dígitos corresponde la información que se pone en a, b, c, d, e, f, g y P.

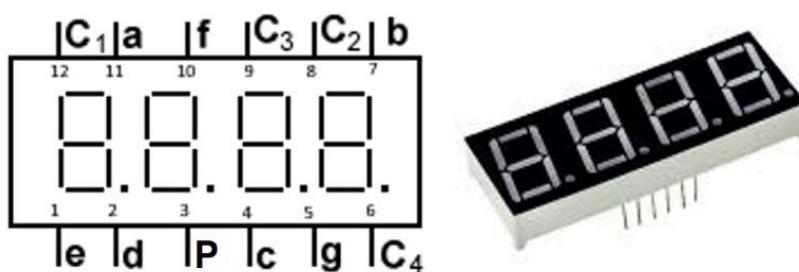


Fig. 103 Pantalla de 7 segmentos de cuatro dígitos.

La multiplexación de un Display simplemente consiste en encender un único display, mostrar el número y luego apagarlo, para encender el display que le sigue. El truco de encender y apagar el display a una alta velocidad, permite engañar al ojo humano, y tener la sensación de que todos los displays se encuentran energizados al mismo tiempo, pero en realidad, si lo vemos en cámara lenta, los displays de 7 segmentos de ánodo o cátodo común están encendiéndose mostrando el número y después se apagan cuando se enciende el siguiente.

Supongamos que quiero visualizar el número 1234. Para esto tenemos que hacer lo siguiente, encendemos el dígito de la unidad de mil y enviamos a visualizar el número 1, dejamos pasar 5ms lo apagamos y encendemos el dígito de la decena y enviamos a visualizar el número 2, dejamos pasar 5ms lo apagamos y encendemos la Decena y enviamos a visualizar el número 3, dejamos pasar 5ms lo apagamos y encendemos la Unidad y enviamos el número 4.

28.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 módulo IC 74HC595
- 1 pantalla de 4 dígitos y 7 segmentos
- 4 resistencias 220 Ω

28.4. Esquema de conexión

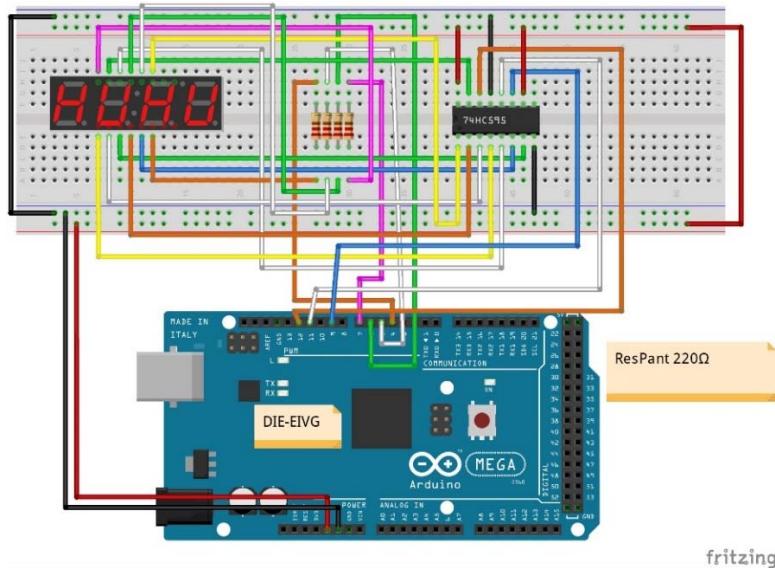


Fig. 104 Esquema de conexión Lección 27.

28.5. Códigos de los programas

1. Leccion_27_Display_de_7_seg_cuatro_digitos.ino: Programa que controla una pantalla de 4 dígitos de 7 segmentos con el integrado 74HC595. Escribe desde 0 a 9 en los dígitos diferentes de la pantalla.
2. Leccion_27_Display_de_7_seg_cuatro_digitos_1.ino: Programa que controla una pantalla de 4 dígitos de 7 segmentos con el integrado 74HC595. Escribe 0123 en la pantalla.
3. Leccion_27_Display_de_7_seg_cuatro_digitos_2.ino: Programa que controla una pantalla de 4 dígitos de 7 segmentos con el integrado 74HC595. Mostrando un contador de 0000 a 9999

28.6. Bibliografía y direcciones de interés

- [1]. <https://www.electrontools.com/Home/WP/multiplexar-display-7-segmentos/>
- [2]. <https://www.prometec.net/display-4digitos/>
- [3]. <https://controlautomaticoeducacion.com/arduino/multiplexar-display-7-segmentos/>
- [4]. <https://texolab.net/2017/10/27/arduino-display-7-segmentos-4-digitos/>
- [5]. <http://www.iescamp.es/miarduino/2016/04/14/display-de-7-segmentos-y-4-digitos-con-interface/>
- [6]. <https://www.diarioelectronicohoy.com/blog/display-multiple-de-7-segmentos>
- [7]. <https://create.arduino.cc/projecthub/goarray/4-digit-7-segment-shift-register-counter-b7338d>
- [8]. https://create.arduino.cc/projecthub/lagsilva/digital-clock-with-arduino-rtc-and-shift-register-74hc595-1141e6?ref=similar&ref_id=167235&offset=1
- [9]. <https://simple-circuit.com/arduino-7-segment-74hc595-shift-register/>
- [10]. <https://simple-circuit.com/arduino-7-segment-display-4-digit/>
- [11]. <https://www.instructables.com/Temperature-Displayed-on-4-Digit-7-segment-common/>
- [12]. <https://arduino.stackexchange.com/questions/35759/74hc595-with-4-digit-7-segment-display-any-way-to-get-rid-of-leading-zeros>
- [13]. <https://miguelpynto.github.io/ShiftDisplay/>

- [14]. <https://github.com/rmorenojr/ElegooTutorial/commit/2d3a450f8d27b62be10c77cb7da71a966da4b30c>
- [15]. https://create.arduino.cc/projecthub/mdraber/shift-register-tutorial-d9227e?ref=user&ref_id=1474727&offset=4
- [16]. <https://tronixstuff.com/2019/07/13/tutorial-arduino-and-four-digit-seven-segment-display-module/>
- [17]. <https://osoyoo.com/2017/08/08/arduino-lesson-4-digit-7-segment-led-display/>
- [18]. <https://www.askix.com/temperatura-aparece-en-la-4-digitos-7-segmentos-anodo-comun.html>

29. Lección 28 Motor de Corriente Continua

29.1. Objetivo de la práctica

Aprender a controlar un pequeño motor de CC (corriente continua)

29.2. Introducción al componente

Los motores de corriente continua son dispositivos que transforman energía eléctrica en energía mecánica en forma de movimiento angular de su eje. Un motor de corriente continua convierte la energía eléctrica en mecánica. Se compone de dos partes: el estator y el rotor. El estator es la parte mecánica del motor donde están los polos del imán. El rotor es la parte móvil del motor con devanado y un núcleo, al que llega la corriente a través de las escobillas.

Como su nombre lo indica se alimentan con corriente continua, el motor que utilizaremos en esta práctica se puede alimentar desde 3V hasta 5V. De ésta manera para que el eje del motor comience a girar solo es necesario aplicar alimentación (diferencia de potencial entre sus terminales), si las terminales de la fuente de alimentación se invierten, el motor gira en sentido contrario.

Para tener un control completo sobre el motor de corriente continua, tenemos que controlar su velocidad y dirección de rotación. Esto puede lograrse combinando estas dos técnicas.

- **PWM – Para controlar la velocidad:** PWM es una técnica en la que el valor medio del voltaje de entrada se ajusta enviando una serie de pulsos ON-OFF. El voltaje promedio es proporcional al ancho de los pulsos

conocido como Ciclo de Trabajo o Duty Cycle. Cuanto mayor sea el ciclo de trabajo, mayor será el promedio de voltaje que se aplica al motor de CC (alta velocidad) y menor será el ciclo de trabajo, menor será el promedio de voltaje que se aplica al motor de CC (baja velocidad).

- **H-Bridge (L293D)**– **Para controlar la dirección de rotación:** La dirección de giro del motor de corriente continua puede ser controlada cambiando la polaridad de su voltaje de entrada. Una técnica común para hacer esto es usar un Puente H. Un circuito de Puente-H contiene cuatro interruptores con el motor en el centro formando una disposición similar a la del Puente-H. Cerrar dos interruptores particulares al mismo tiempo invierte la polaridad del voltaje aplicado al motor. Esto causa un cambio en la dirección de giro del motor.

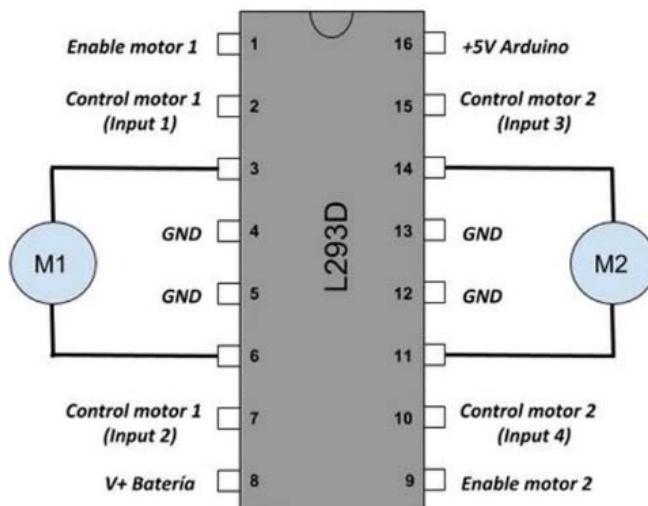


Fig. 105 Patillaje del L293D.

Los pines del L293D:

- El pin 16, son los 5V con los que alimentamos Arduino y el pin 8, es la tensión con la que alimentamos el motor.
- Los pines del 1 al 7 controlan el primer motor y los pines 9 a 15 controlan el segundo motor.
- El pin 1, Enable motor 1, Activa el uso del motor 1.
- Los pines 2 (a 1 el motor girará a favor de las agujas del reloj) y 7 (a 1 el motor girará en contra de las agujas del reloj) son los pines de control para el motor 1, e irán conectados a nuestros Arduino para controlar el sentido de giro.

- Los pines 3 y 6 son la salida a la que se conecta el motor 1, cuya polaridad se invierte en función los valores de 2 y 7.
- El pin 9, Enable motor 2, Activa el uso del motor 2.
- Los pines (11 y 14) y (10 y 15). Son los pines equivalentes para el control del motor 2.
- Los pines 4, 5, 12 y 13 van a GND.



Fig. 106 Patillaje del L293D.

HIGH = 5 V; LOW = 0 V; X = Indiferente

Pin 1	Pin 2	Pin 7	Comportamiento Motor 1
HIGH	LOW	HIGH	Motor gira en contra de las agujas del reloj
HIGH	HIGH	LOW	Motor gira a favor de las agujas del reloj
HIGH	LOW	LOW	Parada rápida del motor
HIGH	HIGH	HIGH	Parada rápida del motor
LOW	X	X	Parada rápida del motor

Pin 9	Pin 15	Pin 10	Comportamiento Motor 2
HIGH	LOW	HIGH	Motor gira en contra de las agujas del reloj
HIGH	HIGH	LOW	Motor gira a favor de las agujas del reloj
HIGH	LOW	LOW	Parada rápida del motor
HIGH	HIGH	HIGH	Parada rápida del motor
LOW	X	X	Parada rápida del motor

Cuando conectemos motor de corriente continua es probable que necesite más energía que pueda aportar la fuente de Arduino MEGA2560 R3 directamente. Si tratamos de conectar el motor directamente a un pin de placa MEGA2560 R3, hay una buena probabilidad de que se pueda dañar. Para ello usar un módulo de alimentación que proporcione la alimentación adecuada. La tensión de salida en los carriles de arriba y de abajo se pueden configurar independientemente. Podemos configurar los dos carriles con la misma tensión o distinta en cada carril. Para seleccionar la tensión de salida, hay que mover el puente a los pines correspondientes. Izquierda 5V y derecha 3.3V.

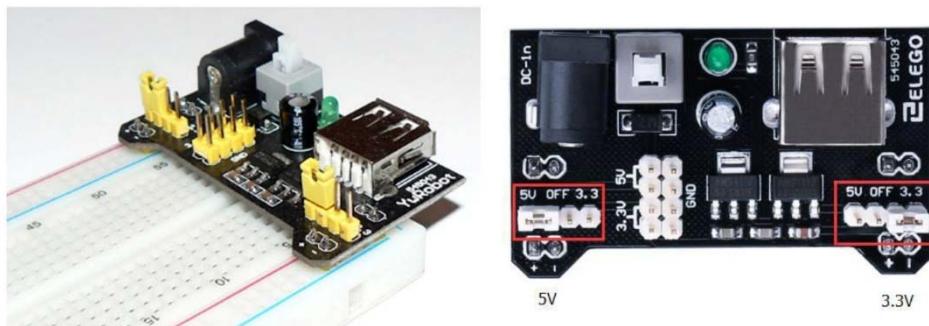


Fig. 107 Alimentación para protoboards.

Es la forma más simple de alimentar nuestras placas es usando una fuente de alimentación estable y conectarla al bus de la protoboard.

29.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 IC L293D
- 1 aspa de ventilador

- 1 motor CC de 3-6v
 - 1 módulo de alimentación
 - 1 adaptador corriente de 9V 1A.
 - 1 potenciómetro de 10kΩ

29.4. Esquema de conexión

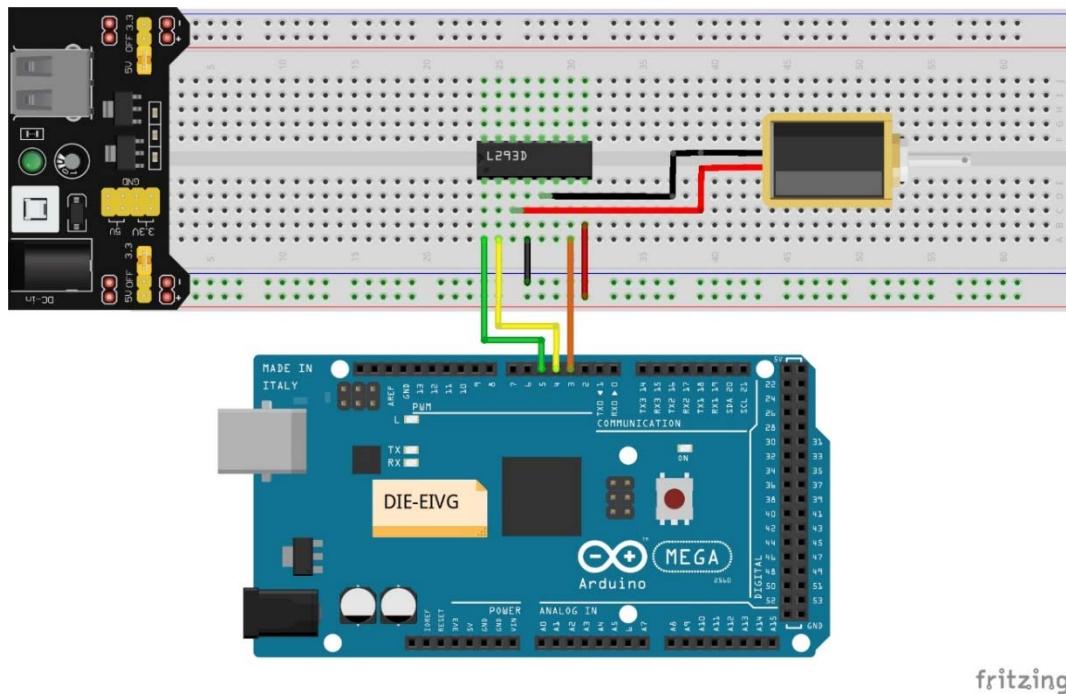


Fig. 108 Esquema de conexión Lección 28, ejercicios 1, 2 y 3.

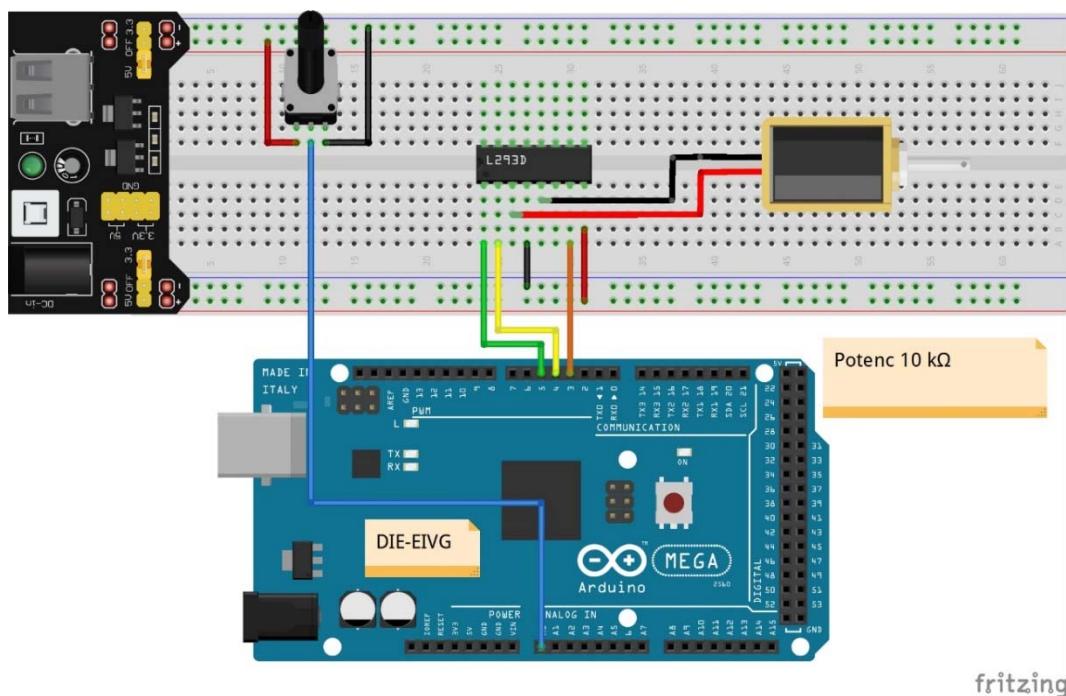


Fig. 109 Esquema de conexión Lección 28, ejercicio 4.

29.5. Códigos de los programas

1. Leccion_28_Motor_de_corriente_continua.ino: Programa que el mueve el motor de CC a favor de las agujas del reloj y luego en contra a velocidad máxima, 5 veces. Muestra en el monitor serie el sentido de giro.
2. Leccion_28_Motor_de_corriente_continua_1.ino: Programa que el mueve el motor de CC a favor de las agujas del reloj desde parado hasta velocidad máxima y luego en contra. Muestra en el monitor serie el sentido de giro y el valor de la variable PWM.
3. Leccion_28_Motor_de_corriente_continua_2.ino: Programa que el mueve el motor de CC según el sentido que se le mande por el monitor serie y también introduciremos la velocidad desde 0 a 9. Si introducimos un valor mayor que 9 el motor girará a la velocidad del último digito.
4. Leccion_28_Motor_de_corriente_continua_3.ino: Programa que el mueve el motor de CC dependiendo de la posición de un potenciómetro. La velocidad del eje del motor se modifica al girar el potenciómetro y el sentido de giro cambia cuando cruzamos por el punto medio del rango del potenciómetro aproximadamente. Muestra en el monitor serie el sentido de giro y el valor de la variable PWM.

29.6. Bibliografía y direcciones de interés

- [1]. <https://aprendiendoarduino.wordpress.com/tag/motor-dc/>
- [2]. <https://blog.330ohms.com/2020/06/15/como-conectar-un-motor-de-corriente-directa-a-arduino/>
- [3]. <http://diymakers.es/control-velocidad-y-sentido-de-motor-dc/>
- [4]. <https://descubrearduino.com/l293d/>
- [5]. <https://www.instructables.com/Arduino-How-to-Control-DC-Motors-With-L293D-Motor-/>
- [6]. <https://www.instructables.com/Control-a-DC-Motor-Using-Arduino-With-L293D/>
- [7]. http://www.practicasconarduino.com/manualrapido/montaje_con_l293.htm1
- [8]. <https://www.prometec.net/hbridge/>
- [9]. <https://www.prometec.net/motorcc/>

- [10]. <https://www.luisllamas.es/arduino-motor-corriente-continua-l298n/>
- [11]. <https://www.aranacorp.com/es/controla-un-motor-cc-con-arduino/>
- [12]. <http://www.iescamp.es/miarduino/2016/02/18/motores-de-corriente-continua/>
- [13]. <https://tecnopatafisica.com/tecnologia/robotica/73-dirverl298n>
- [14]. <http://mecabot-ula.org/tutoriales/arduino/practica-9-control-de-un-motor-de-corriente-continua/>
- [15]. <http://kio4.com/arduino/29Bmotordecontinua.htm>
- [16]. <https://soloarduino.blogspot.com/2016/04/mb102-alimentacion-para-breadboards.html>
- [17]. <https://bitwisear.blogspot.com/2020/10/capitulo-62-modulo-fuente-de-protoboard.html>
- [18]. <https://www.geekfactory.mx/tutoriales/tutoriales-arduino/alimentar-el-arduino-la-guia-definitiva/>
- [19]. <https://www.luisllamas.es/alimentar-arduino-baterias/>
- [20]. http://www.practicasconarduino.com/manualrapido/alimentacion_elctrica_de_arduino.html
- [21]. <https://aprendiendoarduino.wordpress.com/2016/11/09/alimentacion-arduino/>

30. Lección 29 Relé

30.1. Objetivo de la práctica

Aprender a utilizar un relé.

30.2. Introducción al componente

Un relé es un interruptor accionado eléctricamente. El relé (en inglés, relay) es un dispositivo que es utilizado para controlar el encendido y apagado de dispositivos que requieren altos voltajes o altas corrientes mediante una señal de corriente continua como las proporcionadas por los pines de un Arduino. Muchos relés utilizan un electroimán para operar mecánicamente un interruptor, pero otros principios de funcionamiento también se utilizan los relés de estado sólido. Los relés se utilizan cuando es necesario para controlar un circuito por una señal de baja potencia (con aislamiento eléctrico

completo entre el control y los circuitos controlados), o cuando varios circuitos deben ser controladas por una señal.

La principal diferencia entre ellos es que los electromecánicos cuentan internamente con partes móviles, de ahí que sean mecánicos. Los de estado sólido, por el contrario, están compuestos por dispositivos semiconductores: diodos, transistores, tiristores y triacs.

Físicamente un relé se comporta como un interruptor “convencional” pero que, en lugar de accionarse manualmente, es activado de forma electrónica. Los relés son aptos para accionar cargas tanto de corriente alterna como continua.

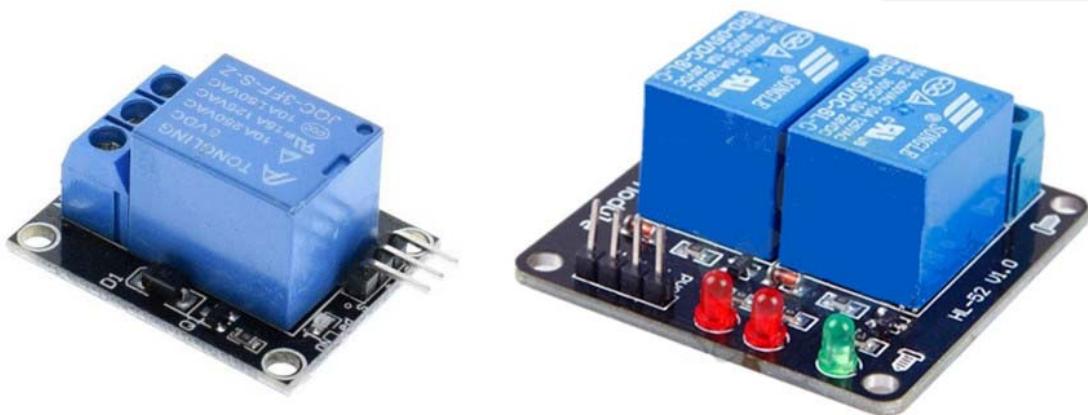


Fig. 110 Relés en pcb.

Estas placas comerciales montan relés con una limitación de 250V en corriente alterna (AC) o 30V en corriente continua (DC). La intensidad máxima que pueden soportar es de 10A. Esto es equivalente a una carga de 2.300W a 230V AC, y 300W en 30V DC.

Un relé dispone de dos circuitos:

- El circuito primario se conecta con la electrónica de baja tensión, en nuestro caso Arduino, y recibe la señal de encendido y apagado. Llamaremos a esta parte etapa de control y es la encargada de controlar la corriente que circula por la bobina o el inductor. Está formado por una bobina arrollada a un núcleo metálico, formando un electroimán. el relé tiene tres pines:
 - Tierra o GND, que deberá conectarse a uno de los pines marcados como GND.
 - Tensión, o V_{CC}, que deberá conectarse a uno de los pines marcados como 5 Voltios.

- Señal, o S, que es un pin digital a través del cual indicamos si el relé estará abierto o cerrado. Lo conectamos a cualquiera de los pines digitales.
- El circuito secundario es el interruptor encargado de encender o apagar la carga, está formado por unos contactos eléctricos instalados en láminas de metal flexible. Los relés normalmente disponen de tres contactos en el secundario C (común), NO (normalmente abierto) y NC (normalmente cerrado).
 - Un relé normalmente abierto (NO) es como un circuito abierto, no deja pasar la corriente, cuando no se aplica una corriente a la bobina, en estado de reposo. Al aplicar la corriente a la bobina el circuito se cierra y permite que fluya la corriente entre los contactos.
 - Un relé normalmente cerrado (NC) funciona, al contrario, en estado de reposo el circuito está cerrado. Al aplicar una corriente a la bobina abre el circuito e interrumpe el paso de la corriente entre los contactos.

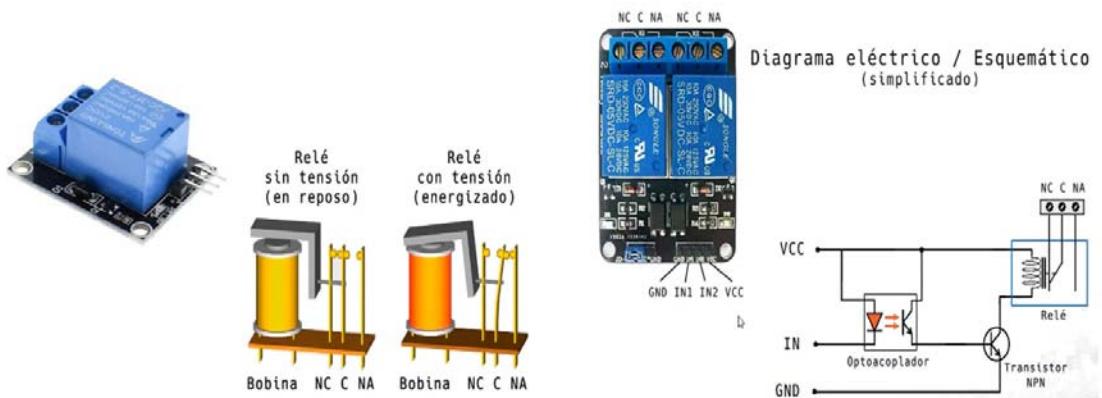


Fig. 111 Conexión Relé.

Vamos a tener placas que tienen desde un solo relé a 8 relés. La placa de la imagen tiene un conector de entradas (IN1 e IN2) y alimentación (GND es masa o negativo y Vcc es el positivo), dos leds que indican el estado de las entradas, dos optoacopladores del tipo FL817C, dos diodos de protección, dos relés de la marca SONGLE con bobinas de 5V y contactos capaces de controlar hasta 10 Amperes en una tensión de 250V y dos borneras, con tres contactos cada una (Común, Normal abierto y Normal cerrado), para las salidas de los relés.

El transistor del opto tiene su colector a Vcc y su emisor conectado al transistor NPN. Este es otro circuito serie por el cual circula corriente cuando el transistor del opto conduce al ser “iluminado” por su diodo, con lo que se introduce corriente en la base del transistor.

El transistor NPN está conectado en una típica configuración emisor común, con su emisor a masa (GND) y la bobina del relé como carga en el colector. Cuando circula corriente por la base desde el opto, el transistor NPN se satura permitiendo el paso de la corriente a través de la bobina del relé, lo que produce que se cierren los contactos de este (común con normal abierto). Al ponerse la entrada a nivel BAJO se pone a la saturación el transistor NPN a través del optoacoplador con lo que se cierra el contacto normal abierto del relé.

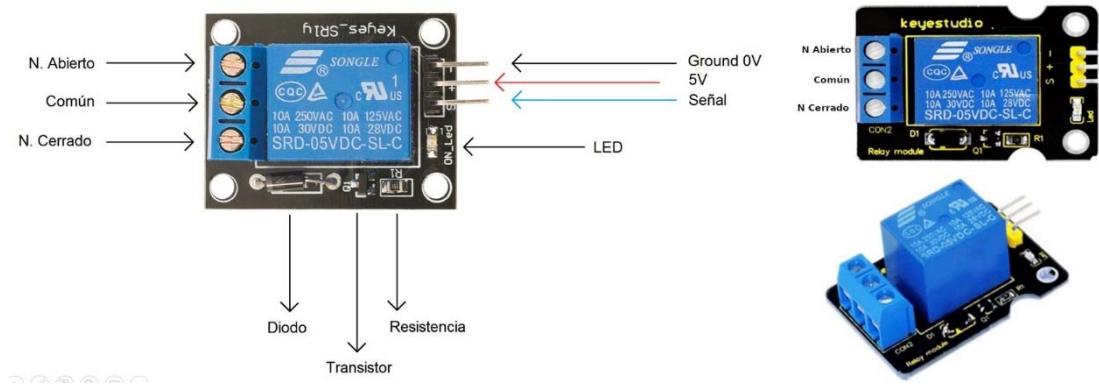


Fig. 112 Relés en pcb.

Las señales de entradas se activan con niveles BAJOS (0). Los pines de entrada del módulo funcionan a la inversa. Como podemos ver el relé se activará cuando el pin de entrada esté BAJO porque de esa manera la corriente podrá fluir desde el VCC al pin de entrada que está a tierra, y el LED se encenderá y activará el relé. Cuando el pin de entrada sea ALTO, no habrá flujo de corriente, por lo que el LED no se iluminará y el relé no se activará.

```
digitalWrite(pinRelé, LOW); // activación del módulo de relé con un nivel bajo
```

30.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard

30.4. Esquema de conexión

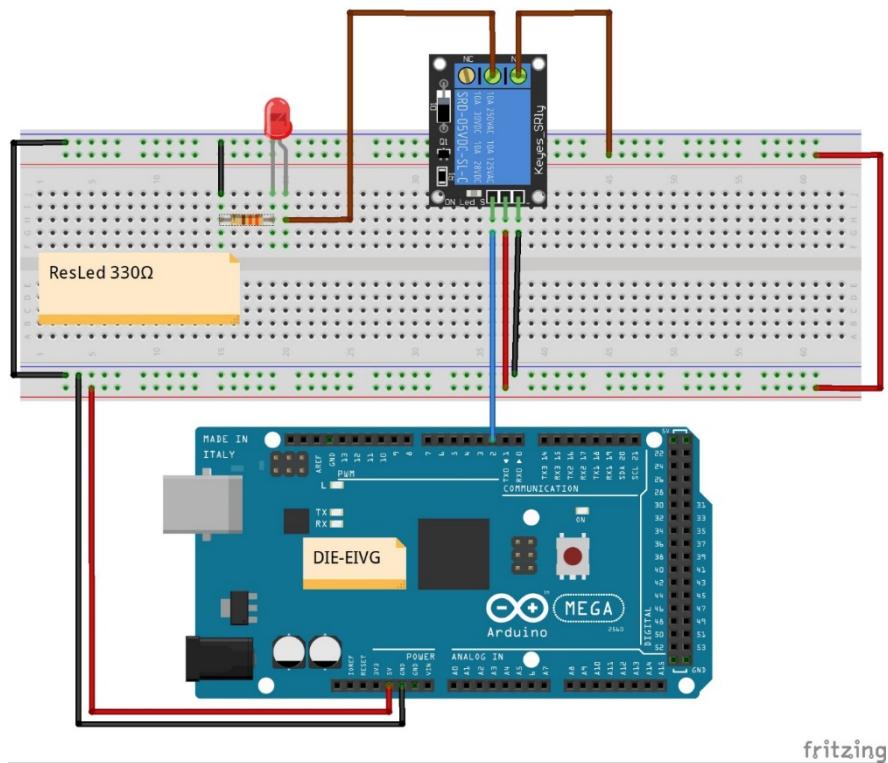


Fig. 113 Esquema de conexión Lección 29, Ejercicio 1.

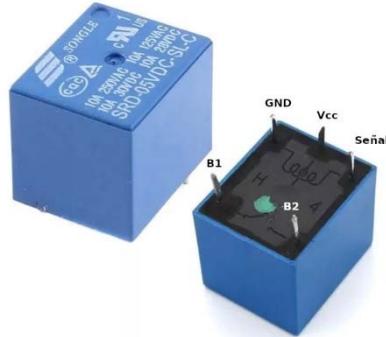


Fig. 114 Relé kit Elegoo.

La forma de cómo conectar un relé de 5 pines en protoboard consiste en colocar Vcc a positivo (+5v), GND a tierra y la señal al pin de Arduino, el cual es activado por el Arduino mediante programación con un CERO o nivel bajo. Y la carga debe ser conectado a los pines B1 y B2 que son pines Normalmente Abierto.

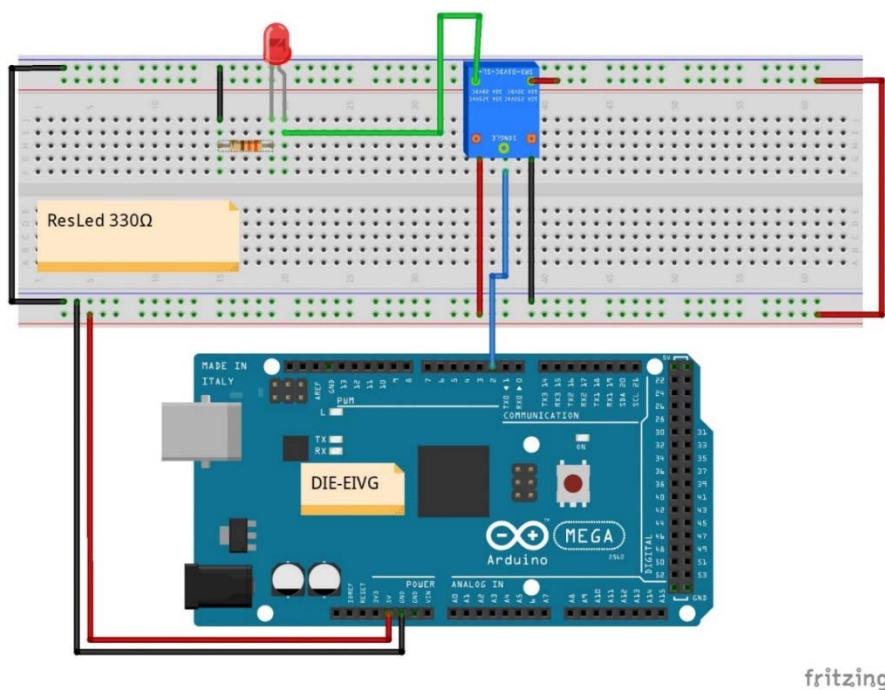


Fig. 115 Esquema de conexión Lección 29.

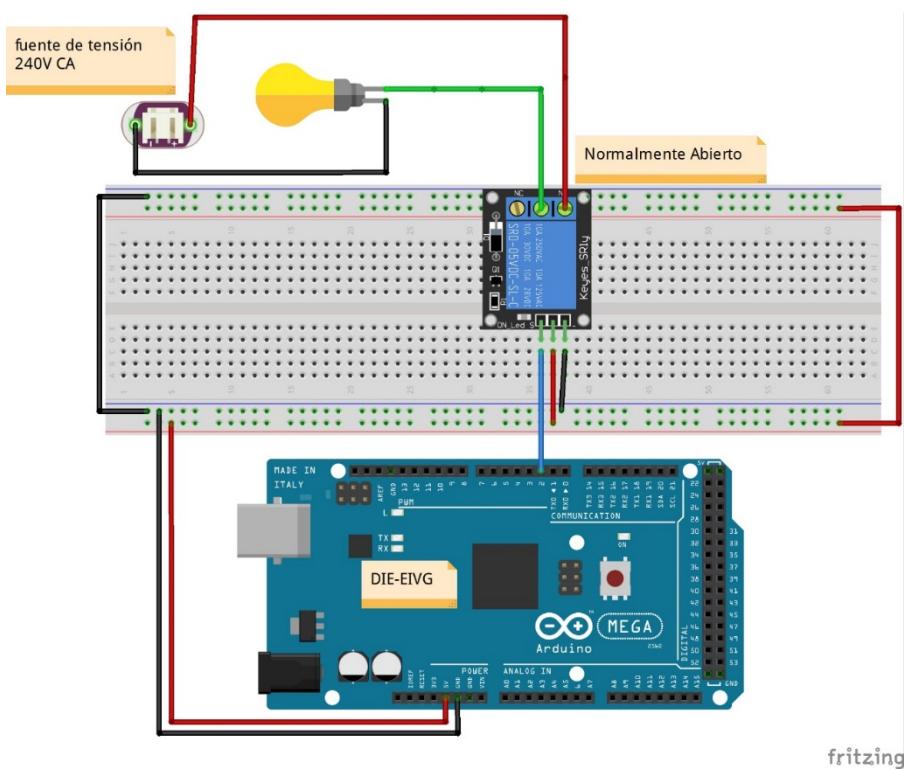


Fig. 116 Esquema de conexión Lección 29, con fuente de tensión de 220V.

30.5. Códigos de los programas

1. Leccion_29_Relay.ino: Programa que permite verificar el funcionamiento de un módulo de relé, simple o doble activándolo a intervalos de 5 segundos,

encendiendo y apagando un led. Y mostrando el estado del relé en el monitor serie.

30.6. Bibliografía y direcciones de interés

- [1]. <https://www.luisllamas.es/arduino-salida-rele/>
- [2]. <https://randomnerdtutorials.com/guide-for-relay-module-with-arduino/>
- [3]. <https://www.prometec.net/reles/>
- [4]. <http://diwo.bq.com/utilizar-rele-arduino-zum-core/>
- [5]. <https://aprendiendoarduino.wordpress.com/tag/rele/>
- [6]. <https://programarfácil.com/blog/arduino-blog/rele-con-arduino-lampara/>
- [7]. <https://programarfácil.com/blog/rele-y-arduino-mkr1000/>
- [8]. <https://www.profetolocka.com.ar/2015/05/09/modulo-de-4-reles-para-arduino/>
- [9]. <https://solectroshop.com/es/blog/guia-para-principiantes-sobre-modulos-de-reles-en-los-proyectos-de-arduino-n28>
- [10]. <https://controlautomaticoeducacion.com/arduino/relevador-con-arduino/>
- [11]. <https://www.altaruru.com/como-activar-un-rele-desde-arduino/>
- [12]. <http://sarrieta724.blogspot.com/2018/10/rele-para-arduino.html>
- [13]. <http://howtomechatronics.com/tutorials/arduino/control-high-voltage-devices-arduino-relay-tutorial/>

31. Lección 30 Motor Paso a Paso 28BYJ-48

31.1. Objetivo de la práctica

Aprender a manejar un motor paso a paso, 28BYJ-48. Y su controladora ULN2003.

31.2. Introducción al componente

Un motor paso a paso es un dispositivo electromecánico que convierte pulsos eléctricos en movimientos mecánicos discretos. El eje de un motor paso a paso gira en incrementos discretos cuando impulsos de mando eléctrico se aplican a él en la secuencia correcta.

La secuencia de los pulsos aplicados se relaciona directamente con la dirección de rotación de ejes motor. La velocidad de la rotación de los ejes motor está directamente

relacionada con la frecuencia de los pulsos de entrada y la duración de la rotación está directamente relacionada con el número de pulsos de entrada aplicada.

Los motores paso a paso (MPP) también conocidos como steppers, son similares en muchos sentidos a los de CC, pero con velocidades de giro y potencias bajas. Aquí lo que destaca es el posicionamiento del eje, es decir, la precisión para ponerlos en una posición concreta.

El 28BYJ-48 es un pequeño motor paso a paso unipolar de bajo precio. Las características eléctricas del 28BYJ-48 son modestas, pero incorpora un reductor integrado que lo convierte en un componente mucho más útil e interesante.

El 28BYJ-48 tiene un paso de 5.625 grados (64 pasos por vuelta). El reductor interno tiene una relación de 1/64. Combinados, la precisión total es de 4096 pasos por vuelta, equivalente a un paso de 0.088°, que es una precisión muy elevada. En realidad, la relación del reductor no es exactamente 1/64 por lo que el número de pasos es 4076 por vuelta (equivalente a un reductor de 1/63.6875).

El 28BYJ-48 tiene un par máximo tras el reductor de 3N . cm (0.3Kgf . cm). La frecuencia máxima es de 100Hz, lo que supone unos 40 segundos por vuelta, o equivalentemente una velocidad de giro máxima en torno a 1.5 rpm.

La resistencia y el consumo eléctrico varían con el modelo de 28BYJ-48. En los modelos de 5V es de 60 Ohm, lo que supone un consumo de 83mA. Los modelos de 12V tienen resistencias de 130-380 Ohm, y consumos de 71 a 32mA.



Fig. 117 Motor paso a paso, 28BYJ-48 y su controladora ULN2003.

El 28BYJ-48 es un motor unipolar (4 bobinas) y lo podemos mover de tres maneras diferentes, movimiento normal, movimiento por ola o paso completo y movimiento de medio paso. Veamos cómo funcionan cada uno de estos movimientos. Sin el uso de ninguna librería.

Movimiento por ola o paso completo (par menor)

Consiste en excitar una bobina cada vez. El consumo se reduce, pero también el par, por lo tanto, es un consumo y par moderados.

PASO	BOBINA A	BOBINA B	BOBINA C	BOBINA D
1	ON	OFF	OFF	OFF
2	OFF	ON	OFF	OFF
3	OFF	OFF	ON	OFF
4	OFF	OFF	OFF	ON

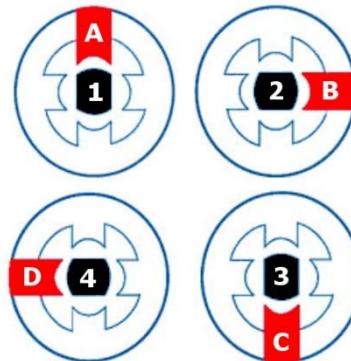


Fig. 118 Movimiento por ola o paso completo (par menor).

Movimiento normal (par máximo)

Consiste en excitar, administrar corriente, a dos bobinas a la vez en cada paso. Se consigue el máximo par pero también es el máximo consumo.

PASO	BOBINA A	BOBINA B	BOBINA C	BOBINA D
1	ON	ON	OFF	OFF
2	OFF	ON	ON	OFF
3	OFF	OFF	ON	ON
4	ON	OFF	OFF	ON

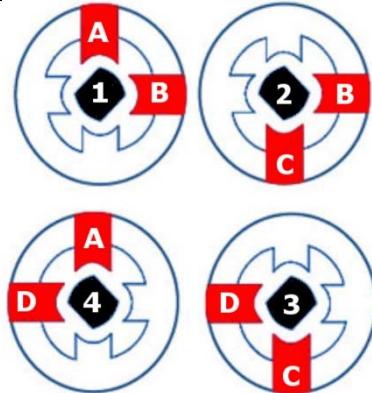


Fig. 119 Movimiento normal (par máximo).

Movimiento de medio paso

Se consigue un movimiento lento y suave con un par y consumo entre medias de los otros dos movimientos.

PASO	BOBINA A	BOBINA B	BOBINA C	BOBINA D
1	ON	OFF	OFF	OFF
2	ON	ON	OFF	OFF
3	OFF	ON	OFF	OFF
4	OFF	ON	ON	OFF
5	OFF	OFF	ON	OFF
6	OFF	OFF	ON	ON
7	OFF	OFF	OFF	ON
8	ON	OFF	OFF	ON

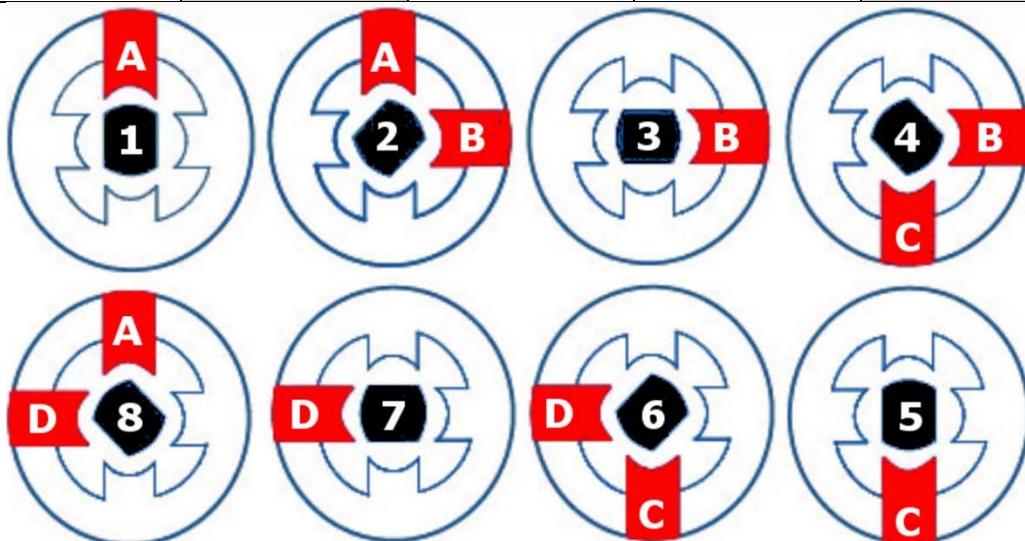


Fig. 120 Movimiento de medio paso

En la programación del ejercicio (Leccion_30_Motor_paso_a_paso_1.ino) habrá que definir unos arrays con estas tablas para secuenciar el movimiento que queramos elegir.

```
// paso completo simple
int paso [4][4] =      // matriz (array bidimensional) con la secuencia de pasos
{
{1, 0, 0, 0},
{0, 1, 0, 0},
{0, 0, 1, 0},
{0, 0, 0, 1}
};
const int numPasos = 4;
```

Fig. 121 Array para definir el paso completo simple

```

// paso completo con maximo torque
int paso [4][4] =      // matriz (array bidimensional) con la secuencia de pasos
{
{1, 1, 0, 0},
{0, 1, 1, 0},
{0, 0, 1, 1},
{1, 0, 0, 1}
};
const int numPasos = 4;

```

Fig. 122 Array para definir el paso completo con máximo par.

```

// medio paso
int paso [8][4] =      // matriz (array bidimensional) con la secuencia de pasos
{
{1, 0, 0, 0},
{1, 1, 0, 0},
{0, 1, 0, 0},
{0, 1, 1, 0},
{0, 0, 1, 0},
{0, 0, 1, 1},
{0, 0, 0, 1},
{1, 0, 0, 1}
};
const int numPasos = 8;

```

Fig. 123 Array para definir la secuencia de medio paso.

La otra manera de programar un motor 28BYJ-48 no es tan artesanal y reduce el código bastante. Se trata de utilizar la librería Stepper que viene incluida con el entorno de desarrollo oficial de Arduino. Esta librería nos facilita el uso de este tipo de motores.

La librería Stepper solamente contiene 3 métodos:

- Constructor Stepper: Hay que definir el número de pasos, y los pines a los que se conecta.
- setSpeed(): Establece la velocidad de movimiento
- step(): Determina cuantos pasos se desplaza en cada momento

```

#include <Stepper.h>          // Incluimos la librería Stepper.h

const int pasosPorVuelta = 2048;    // Valor de ajuste al número de pasos por vuelta
const int vueltasPorMinuto = 15;     // Adjustable range of 28BYJ-48 stepper is 0~17 rpm

// Se inicializa la biblioteca Stepper y construye un elemento en los pines 8 a 11:
Stepper myStepper(pasosPorVuelta, 8, 10, 9, 11);

```

En el setup:

```

void setup() {
  // Asignamos la velocidad en RPM (Revoluciones por Minuto)
  myStepper.setSpeed(vueltasPorMinuto);

```

En el loop:

```
void loop() {
    // Giramos el motor a favor de las agujas del reloj:
    Serial.println("El motor está girando en sentido horario");
    // Movemos el motor un número determinado de pasos (pasosPorVuelta)
    myStepper.step(pasosPorVuelta);
    delay(500);
```

Encoder rotativo

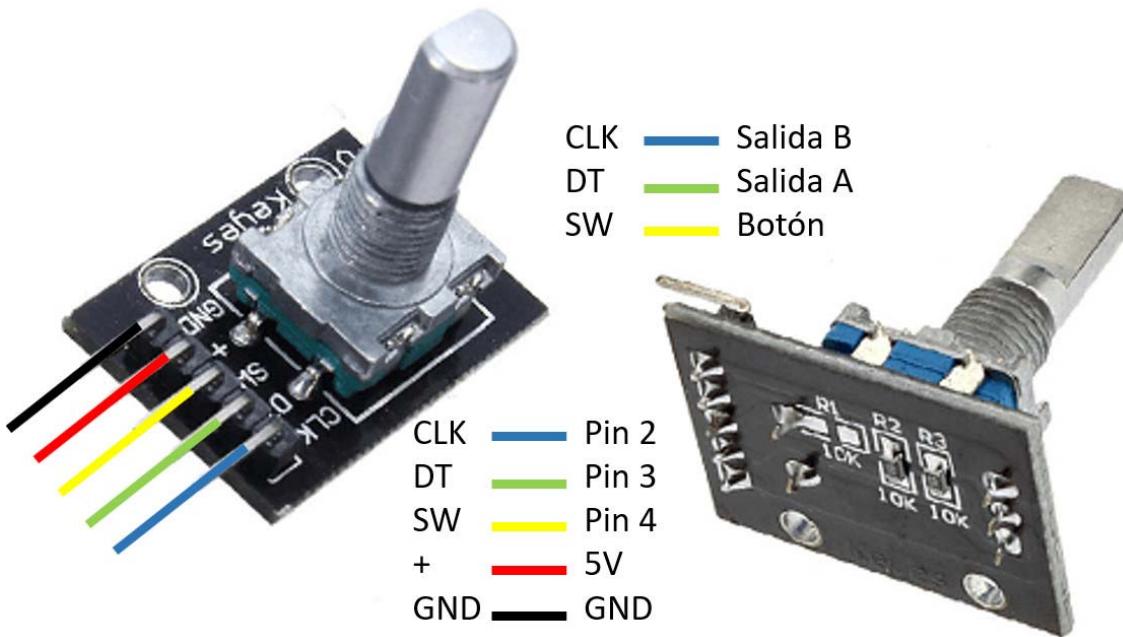


Fig. 124 Encoder rotativo.

Un encoder rotativo es un dispositivo genérico que permite determinar la posición y velocidad angular de un accionamiento, y registrar la medición desde un procesador o desde Arduino.

El encoder tiene los siguientes pines.

- + ⇒ Voltaje V_{CC}
- GND ⇒ Tierra
- CLK ⇒ CLOCK ⇒ Señal de reloj en el que se realiza cada medición.
- DT ⇒ DATA ⇒ Señal de datos en el que se detecta un cambio de movimiento.
- SW ⇒ SWITCH ⇒ Señal de presión. Esta señal solamente nos proporciona la presión sobre el botón.

Externamente estos encoders pueden ser parecidos a ciertos tipos modelos de potenciómetros, lo cual puede ser una ventaja porque hace que ciertos accesorios sean similares, e incluso sea posible sustituir uno por otro. Sin embargo, no hay que confundir

un encoder rotativo con un potenciómetro ya que tanto su electrónica como comportamiento son totalmente diferentes.

Este tipo de encoder rotativo es un dispositivo incremental que proporciona un pulso digital cada vez que el encoder gira un determinado ángulo. El número de pulsos por vuelta depende del encoder empleado, siendo habitual 256 pulsos/vuelta.

Al disponer de dos salidas mecánicas similares, obtenemos dos pulsos diferentes que debidamente estudiados nos indican el sentido de giro además del número de pulsos que se ha girado, porque la señal de una salida estará necesariamente desfasada con respecto a la otra ya que primero pasamos por una muesca y luego por la otra.

Comprobando la secuencia de las dos señales podemos decidir en qué sentido estamos girando el encoder, y contando los pulsos de cualquiera de las dos señales podemos calcular el número de clics girados una vez que conocemos el número de clicks por vuelta completa.

Dentro de sus características tenemos:

- No tienen tope a la derecha ni a la izquierda, por lo cual puede dar todas las vueltas que queramos.
- En su círculo interior hay contactos separados. Al girar el eje un par de pestañas van haciendo o no contacto.
- Cuando gira 360° ha realizado 20 contactos o 20 pasos (step), el usuario percibe ligeramente esos pasos cuando gira el eje.
- Se utilizan en equipos de audio, para controlar servos, control de iluminación, ...
- Tiene dos formas de trabajas, una en el void loop y otra mediante interrupciones.

31.3. Componentes necesarios

- Placa Arduino Mega
- Cable USB
- Protoboard
- 1 módulo controlador de motor paso a paso de ULN2003
- 1 motor paso a paso 28BYJ-48
- 1 módulo de alimentación
- 1 adaptador corriente de 9V 1A.

- 1 encoder rotativo

31.4. Esquema de conexión

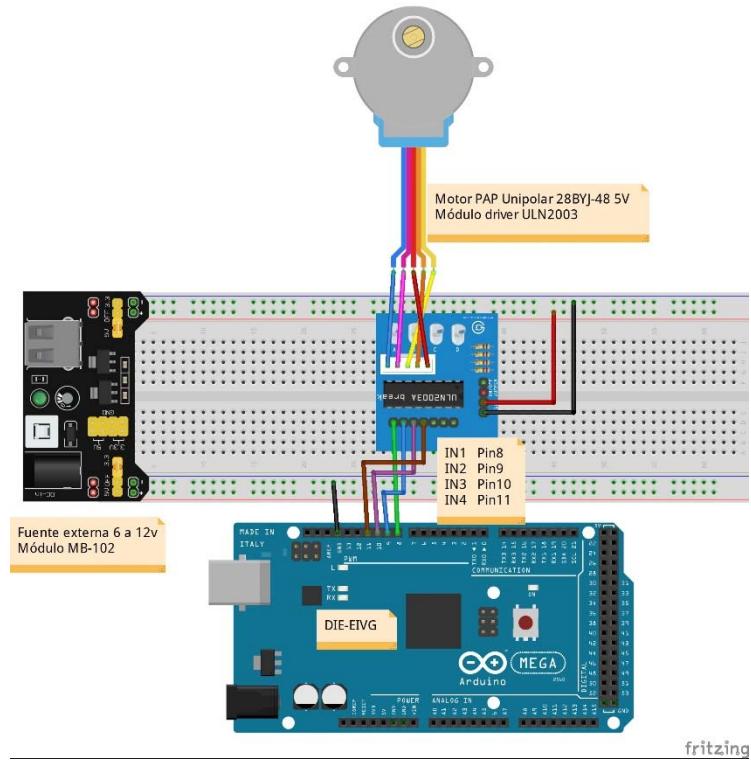


Fig. 125 Esquema de conexión Lección 30, ejercicios 1, 2 y 3.

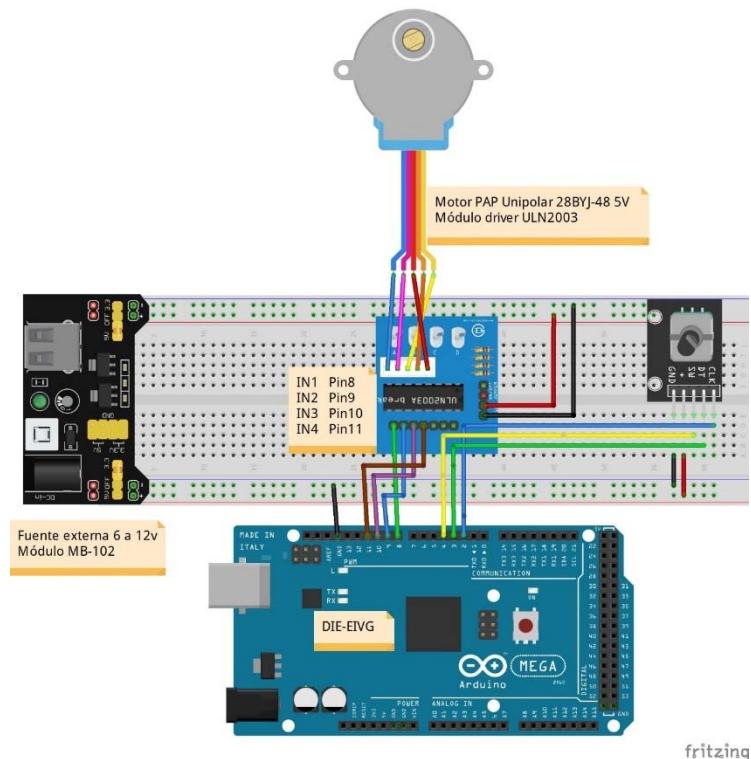


Fig. 126 Esquema de conexión Lección 30, ejercicio 4.

31.5. Códigos de los programas

1. Leccion_30_Motor_paso_a_paso.ino: Programa que realiza un giro completo del motor 28BYJ-48 en conjunto con el controlador basado en ULN2003, detiene 5 segundos y luego comienza nuevamente. La secuencia es la de paso completo simple (wave drive) energizando de a una bobina por vez. Hay que alimentar Arduino con fuente de alimentación externa de 6 a 12 V_{cc}.
2. Leccion_30_Motor_paso_a_paso_1.ino: Programa que realiza un giro completo del motor 28BYJ-48 en conjunto con el controlador basado en ULN2003, detiene 5 segundos y luego comienza nuevamente. La secuencia dependerá del paso elegido. Habrá que quitar el comentario y activar el paso elegido. Se puede elegir entre: [Paso completo simple, Paso completo con máximo par y Medio paso]. Hay que alimentar Arduino con fuente de alimentación externa de 6 a 12 V_{cc}.
3. Leccion_30_Motor_paso_a_paso_2.ino: Programa que realiza un giro completo del motor 28BYJ-48 en conjunto con el controlador basado en ULN2003, detiene 5 segundos y luego comienza nuevamente. El programa usa la librería Stepper.h. Hay que alimentar Arduino con fuente de alimentación externa de 6 a 12 V_{cc}.
4. Leccion_30_Motor_paso_a_paso_3.ino: Programa que realiza el giro de un motor 28BYJ-48 en conjunto con el controlador basado en ULN2003, a partir del giro de un encoder rotativo Modulo KY-040. El programa usa la librería Stepper.h. Hay que alimentar Arduino con fuente de alimentación externa de 6 a 12 V_{cc}.

31.6. Bibliografía y direcciones de interés

- [1]. <https://aprendiendoarduino.wordpress.com/tag/motor-paso-a-paso/>
- [2]. <https://programarfacil.com/blog/motor-paso-a-paso/>
- [3]. <https://www.luisllamas.es/motor-paso-paso-28byj-48-arduino-driver-uln2003/>
- [4]. <https://www.prometec.net/motor-28byj-48/>
- [5]. <https://www.aranacorp.com/es/controla-un-motor-paso-a-paso-con-arduino/>
- [6]. <https://controlautomaticoeducacion.com/arduino/motor-paso-a-paso-arduino/>
- [7]. <https://www.hwlibre.com/motor-paso-a-paso/>

- [8]. <https://bitwisear.blogspot.com/2018/05/capitulo-30-paso-paso-unipolar-con.html>
- [9]. <https://www.web-robotica.com/arduino/motor-de-pasos-28byj-48-con-driver-uln2003-y-arduino-uno>
- [10]. <https://www.aranacorp.com/es/controla-un-motor-paso-a-paso-con-arduino/>
- [11]. [http://www.geeetech.com/wiki/index.php/Stepper_Motor_5V_4-Phase_5-Wire %26 ULN2003 Driver Board for Arduino](http://www.geeetech.com/wiki/index.php/Stepper_Motor_5V_4-Phase_5-Wire_%26_ULN2003_Driver_Board_for_Arduino)
- [12]. <https://www.web-robotica.com/arduino/motor-de-pasos-28byj-48-con-driver-uln2003-y-arduino-uno>
- [13]. <https://github.com/ZGZMakerSpace/ArduinoCourse>
- [14]. <https://zaragozamakerspace.com/index.php/lessons/curso-arduino-y-robotica-motores-paso-a-paso-l298n-driver/>
- [15]. <https://texolab.net/2018/04/15/control-de-posicion-de-un-motor-pap-con-encoder-rotativo/>
- [16]. https://www.biwy-mecatronica.com/2017/11/mpp-encoder-arduino.html#_Toc30080422
- [17]. <https://descubrearduino.com/arduino-como-controlador-de-motor-paso-a-paso/>
- [18]. <https://www.luisllamas.es/arduino-encoder-rotativo/>