Informática Industrial

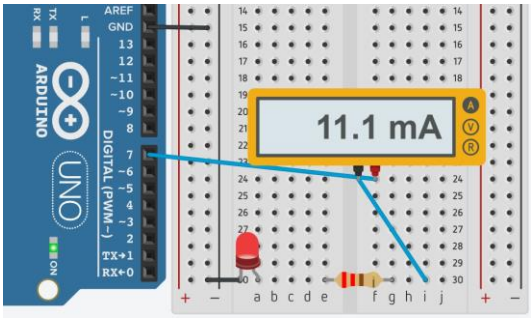
Práctica 1 Introducción a Arduino

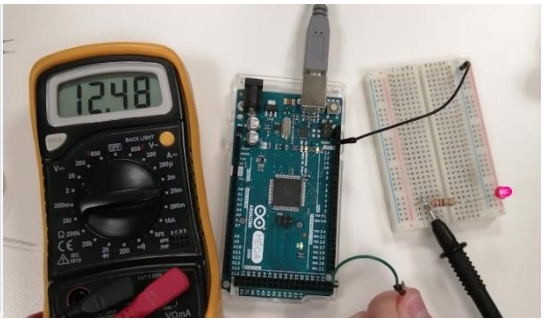
Ejercicio 1

Hay que realizar un esquema resistencia-led teniendo en cuenta las características del diodo y calculando la resistencia adecuada para que el LED funcione a 15mA. Luego aproximamos la resistencia para utilizar las incluidas en el kit comprobamos su funcionamiento. Incluye los cálculos realizados en el espacio en blanco que hay bajo el siguiente esquema y dibuja las caídas de tensión en los elementos del esquema.

Se ha seleccionado la resistencia de 220Ω (rojo-rojo-marrón-dorado) para realizar el experimento ya que es la que mejor se corresponde con el valor teórico obtenido según la ley de Ohm: $V = I * R$ donde el voltaje es 5V menos la caída de tensión que provoca el led (1.7V) y la intensidad deseada 0.015A, de esta manera se obtiene:

$$5 - 1.7 = 0.015 * R \rightarrow R = \frac{3.3}{0.015} = 220\Omega$$



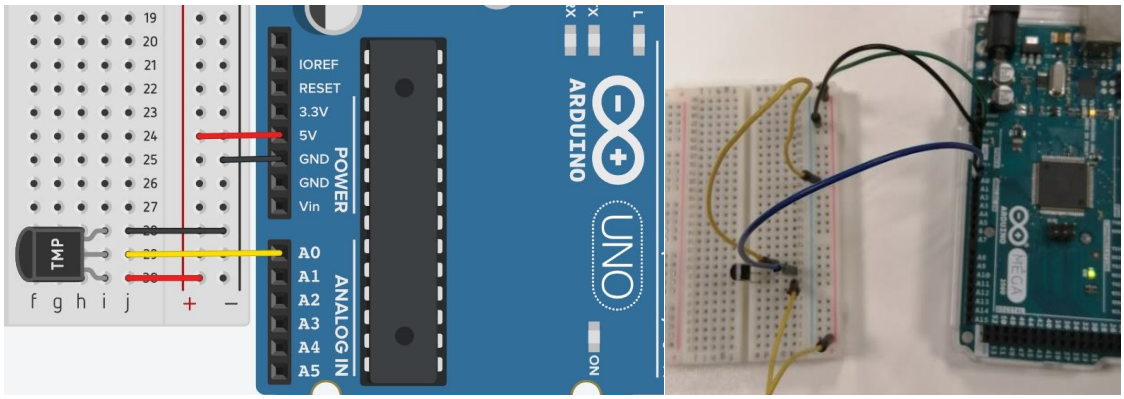


No obstante, con una resistencia de 220Ω se obtiene una corriente experimental de 12.48mA, esto se puede deber a varios factores: la alimentación no es 100% estable en 5V, la impedancia de los cables no es nula, los parámetros de cada led concreto pueden variar mínimamente…

Ejercicio 2

En este ejercicio hay conectar correctamente el sensor de temperatura LM35 al Arduino y leer la señal de salida a través de la entrada analógica A0, y luego hacer la conversión del valor leído a voltios, temperatura en Celsius y Fahrenheit, y enviarlos por puerto serie al ordenador. Dibuja el esquema creado en el siguiente espacio y pon las ecuaciones usadas para realizar la conversión.

La conexión del sensor LM35 es sencilla, consta de 3 pines, un pin de VCC conectado a 5V, un pin de tierra y un pin de salida analógica. El sensor LM35 es muy utilizado en los proyectos con Arduino y su curva de voltaje es muy sencilla, para leerla conectamos el pin de salida analógica a uno de los pines analógicos del Arduino, en este caso el A0, tal y como se muestra en la imagen.



El código para leer el sensor es muy sencillo, además se pide que la temperatura se exprese en grados Celsius y Fahrenheit, entre los cuales existe una conversión matemática que se muestra a continuación.

Ejercicio_2

```
// C++ code
//
void setup()
{
  Serial.begin(9600);
  pinMode(A0, INPUT);
}

void loop()
{
  int raw = analogRead(A0);
  float vin = raw*500.0/1023.0;
  Serial.print("Temperatura: " + String(vin) + "°C\t");
  Serial.println(String((vin*1.8)+32) + "°F");
  delay(1000); // Wait for 1000 millisecond(s)
}
```

COM10

```
Temperatura: 27.86°C    82.15°F
Temperatura: 26.88°C    80.39°F
Temperatura: 26.88°C    80.39°F
Temperatura: 27.37°C    81.27°F
Temperatura: 26.39°C    79.51°F
Temperatura: 25.90°C    78.63°F
Temperatura: 25.90°C    78.63°F
Temperatura: 25.42°C    77.75°F
Temperatura: 24.93°C    76.87°F
Temperatura: 25.42°C    77.75°F
Temperatura: 24.44°C    75.99°F
Temperatura: 24.44°C    75.99°F
Temperatura: 23.95°C    75.11°F
```

Autoscroll Mostrar marca temporal

Como se puede observar, el voltaje del sensor LM35 se corresponde con la temperatura en Celsius, para transformarla a Fahrenheit se aplica una regla de conversión muy sencilla y ambos se muestran por pantalla.

Ejercicio 3

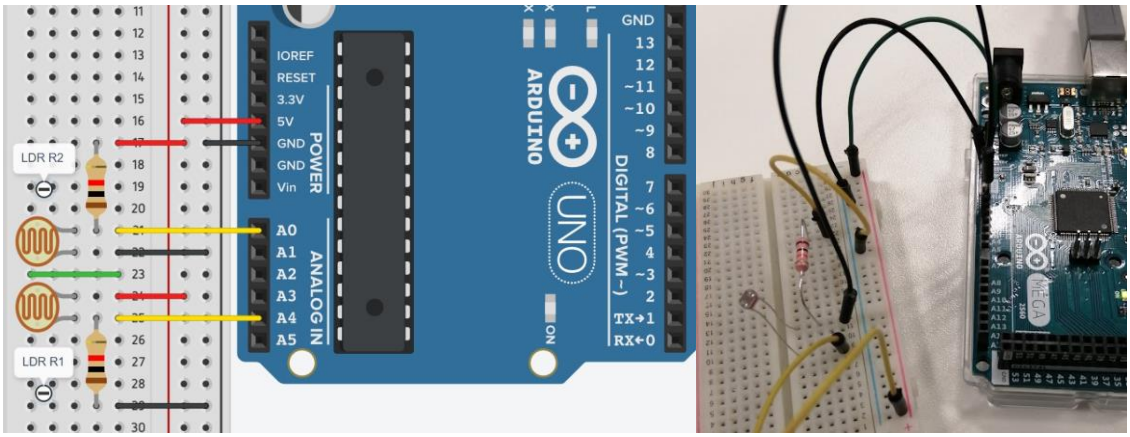
Se pide pensar y plantear los dos esquemas posibles para realizar un divisor de tensión empleando el LDR, proponer las ecuaciones de la intensidad y de la tensión V2 en función de V1, R1 y R2. Finalmente, calcula V2 en los casos en que R1 sea el LDR y R2 una resistencia fija de 10K, y viceversa, expresando V2 en función de la resistencia de la LDR. Calcula también los rangos de V2 para cada caso e implementa un código en Arduino que lea el puerto analógico A0 y envíe por el puerto serie el valor leído en V. Anota los valores obtenidos.

La intensidad que pasa por las resistencias del divisor de tensión es la misma por lo tanto la ecuación del divisor de tensión se obtiene de la siguiente manera:

$$I_1 = \frac{V_1}{R_1} = I_2 = \frac{V_2}{R_2} \rightarrow \frac{V_1}{R_1} = \frac{V_2}{R_2} \rightarrow V_2 = \frac{R_2}{R_1 + R_2} * V_1 \begin{cases} V_2 = \frac{10 * 10^3}{LDR + 10 * 10^3} * 5 \\ V_2 = \frac{LDR}{LDR + 10 * 10^3} * 5 \end{cases}$$

El LDR reduce su resistencia con la incidencia de luz, cuanta mayor sea la luz incidente menor será la resistencia y viceversa, esto nos adelanta el comportamiento del divisor:

- Con el LDR en R1, al aplicar mas luz se reduce el denominador por lo que aumenta el voltaje de salida, tenemos un sensor de luz proporcional directo.
- Con el LDR en R2 obtenemos el resultado opuesto, se reduce el numerador obteniendo un sensor proporcional inverso.

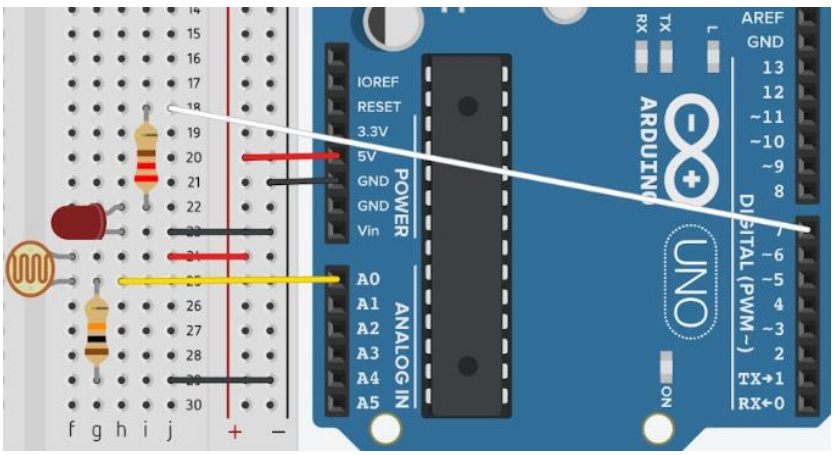


Durante la experimentación se han obtenido valores normales de luz cercanos al 600 (valor raw), en la primera configuración y alrededor de 450 en la segunda. Al incidir con la linterna del teléfono la primera configuración sube hasta los 1000 mientras la segunda baja por debajo de 100.

Ejercicio 4

Elige uno de los dos esquemas anteriores y realiza un programa que encienda un LED de manera automática al detectar que no hay suficiente luz. Para ello, primero habrá que calibrarlo con luz ambiente. Toma diferentes muestras y obtén el valor medio de ellas que se denominará valor base. Busca un umbral u con el cual realizar un ciclo de histéresis. Dibuja el esquema creado.

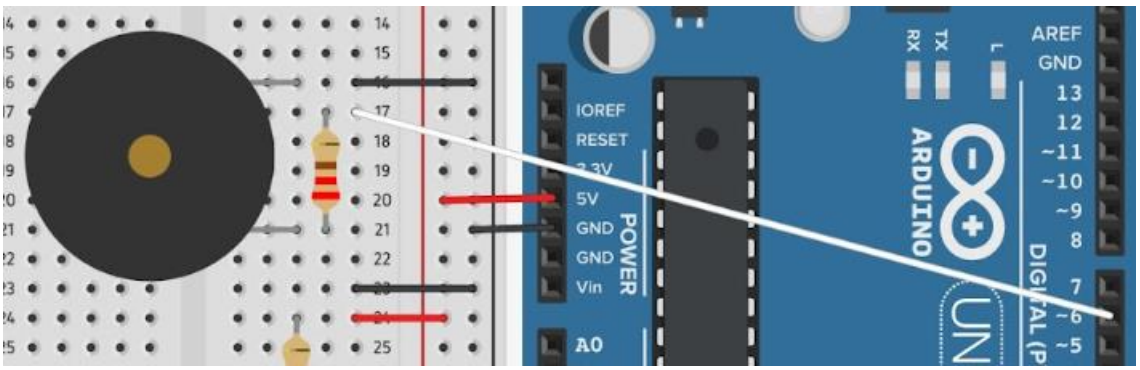
Para resolver este ejercicio se han empleado los datos de calibración del ejercicio anterior, el umbral de apagado se ha colocado en 600 y el de encendido en 300, creando un ciclo de histéresis como se pide. El esquema electrónico empleado es el que se aprecia en la siguiente imagen:



Ejercicio 5

Vamos a ver cómo incluir una nueva librería. Esta librería permite crear un manejador de tiempo, de forma que se programa una interrupción de tiempo que saltará cada "x" milisegundos y llamará a la función que le indiquemos. Ahora debes pinchar en tu protoboard el zumbador proporcionado por el profesor en el pin 6. Mira en la hoja de características del zumbador para determinar cómo se debe realizar su conexión. El funcionamiento de un zumbador es sencillo, cuando recibe una excitación a una determinada frecuencia (es decir un tren de pulsos) emite un pitido, deberás generar este tren de pulsos (usando el temporizador 2) en la salida 6 para hacer sonar el zumbador. Dibuja el esquema resultante.

En primer lugar, creamos una función que permita crear el tren de pulsos que se pide, esta función es muy sencilla, cada vez que se le llama cambia el estado del pin 6. Una vez creada esta función debemos inicializarla en el setup con la siguiente línea: `MsTimer2::set(ms, f)` En esta línea se especifica los milisegundos de período y la función (que debe ser tipo void sin parámetros), por último se inicia el contador con la orden `MsTimer2::start().`

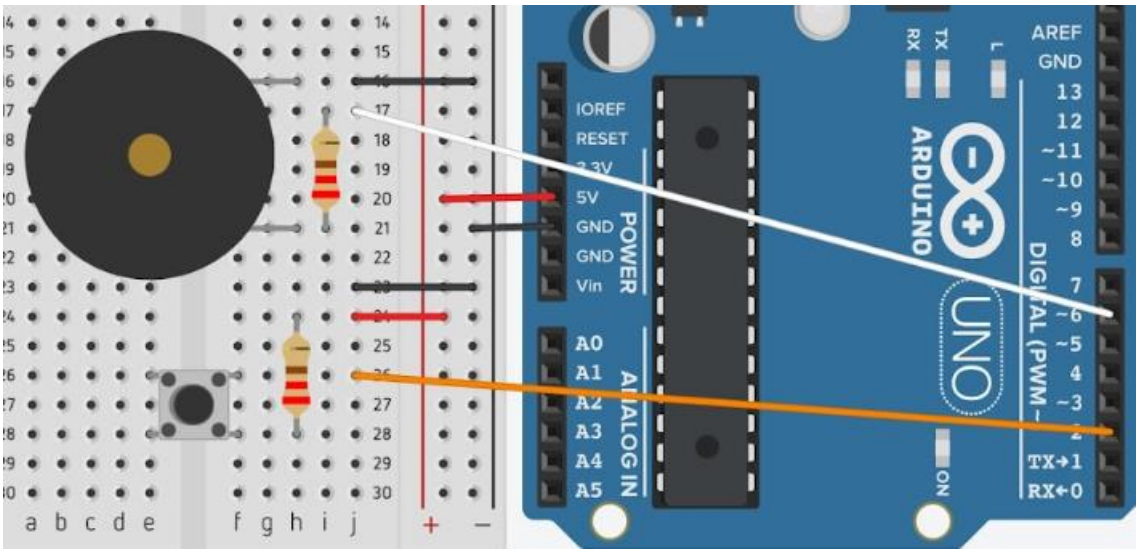


Ejercicio 6

Sustituye el LDR por un pulsador mecánico. Este pulsador debe estar configurado en pull-down. En la salida donde estaba el LED poner un zumbador. Dibuja el esquema y realiza un programa que haga sonar el zumbador al apretar el pulsador y que se detenga al soltarlo. Comenta los resultados.

Este programa se ha realizado con la friolera de 1 linea de código en el loop, que es la que sigue: `digitalWrite(6,digitalRead(2))` con este comando el zumbador copiará el estado del botón, haciendo que pite cuando el botón es presionado y que se detenga al soltarlo.

El esquema de conexiones utilizado es el siguiente:



Ejercicio 7

Ahora sustituye el programa anterior, por un nuevo programa que realice un cambio de estado del zumbador según un flanco de subida en la entrada del pin 2 de Arduino mediante una interrupción. Implementa un contador que incremente en 1 cada vez que se ejecuta la interrupción y muestra su valor en la consola serie. ¿Por qué crees que ocurre esto? Propón una solución y coméntala con el profesor, por último, describe aquí la solución a la que hayáis llegado e incluye un esquema.

En teoría el resultado debería ser que se encienda el zumbador al pulsarlo y se apague al pulsarlo de nuevo.

No obstante, el comportamiento del sistema real no es tan perfecto como cabría esperar por la simplicidad de este, sino que muestra un comportamiento ligeramente errático, esto se debe a un defecto en el mecanismo del pulsador denominado *bouncing* que consiste en que, debido al mecanismo interno del botón, el cambio entre estados no se produce de forma limpia, sino que se genera una señal a modo de interferencia que produce falsas lecturas.

Una manera de solucionar este problema es añadir un filtro de paso bajo para evitar las interferencias de alta frecuencia, pero lo ideal sería utilizar un pulsador con mejores especificaciones que no produzca este problema como los pulsadores de membrana.

Ejercicio 8

En este apartado añade el sensor óptico de infrarrojos en la entrada 2. Para ello realiza el esquema de la figura 5, y determina la resistencia R_L al igual que en el primer ejercicio, para que el LED de infrarrojos emita luz con una excitación de **20mA**. Escribe un programa que haga que el zumbador se encienda cuando el detector óptico detecte algo próximo y deje de zumbar cuando no haya detección. Prueba el sistema y comenta los resultados. Si compruebas que no funciona, trata de averiguar el problema del sistema y resolverlo.

El sensor de infrarrojos esta configurado igual que un optoacoplador, pero orientado hacia la cara exterior en lugar de el receptor frente al emisor, de esta manera al acercarse un objeto que refleje los infrarrojos el circuito se cierra activando la salida del sensor.

La resistencia RL que necesita el led para no fundirse es de 220Ohmnios, al igual que el led anterior. Tras hacer pruebas pudimos observar que los cuerpos claros reflejan mejor los infrarrojos y con una hoja de papel pudimos leer valores de voltaje raw de hasta 1000, este dato es relevante ya que se necesitan por lo menos entre 500 y 600 para detectar un flanco de subida y poder leer la entrada como si fuera digital. Con cuerpos oscuros que tienen menor reflexión apenas se llegaba a los 500.

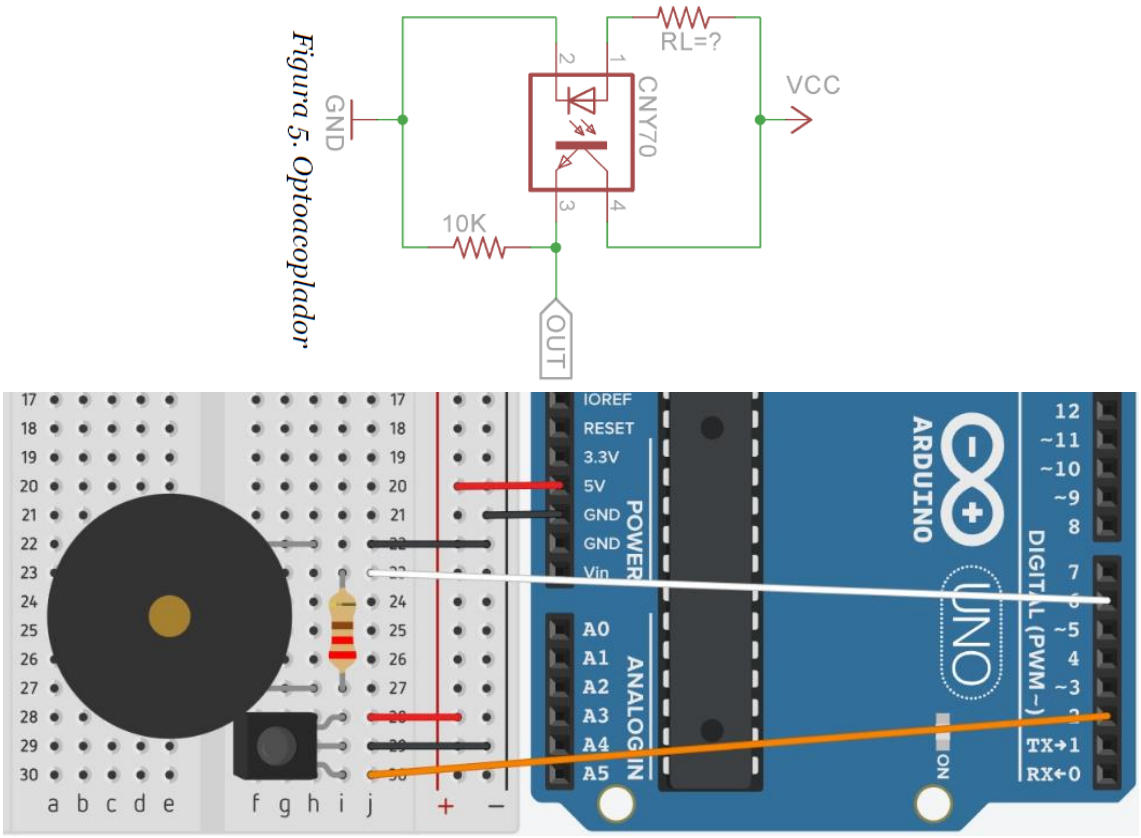


Figura 5. Optoacoplador

Ejercicio 9

Se propone la programación de un juego para ARDUINO utilizando 5 LEDs. El juego parte de los LEDs situados formando un círculo. El juego comienza haciendo brillar el primer LED (el principal), de forma que pasado un determinado tiempo dejará de brillar para encenderse el segundo LED, este proceso debe continuar indefinidamente, de forma que cuando se apague el quinto LED se encenderá de nuevo el primero (el principal).

El juego consiste en parar la ruleta de LEDs justo en el LED principal. Para ello, cuando el jugador pulse un botón, se evaluará si el LED que está encendido es el LED principal. Si es así, se habrá acertado, y se incrementará la velocidad del juego. Si se falla, se bajará la velocidad. El juego sólo acabará cuando el número de aciertos haya incrementado la velocidad hasta una velocidad que se establezca como máxima. En cada acierto se puede decrementar el tiempo de parpadeo en 50 ms. Cuando el juego termina puedes hacer sonar un zumbador durante 0,5 segundos y el tiempo vuelve a su valor inicial.

El esquema realizado en tinkercad y el programa para el juego de la ruleta se pueden consultar en el siguiente enlace con su cuenta de tinkercad. En el esquema se ven los leds conectados directamente, pero lo ideal sería utilizar una resistencia de 220 Ohmnios para cada uno, sin embargo, como los leds no se mantienen encendidos, sino que parpadean para crear el efecto de movimiento, no es peligroso. El botón debe ir colocado en el pin 2 para poder utilizar la interrupción y por último en el pin 6 colocamos el zumbador.