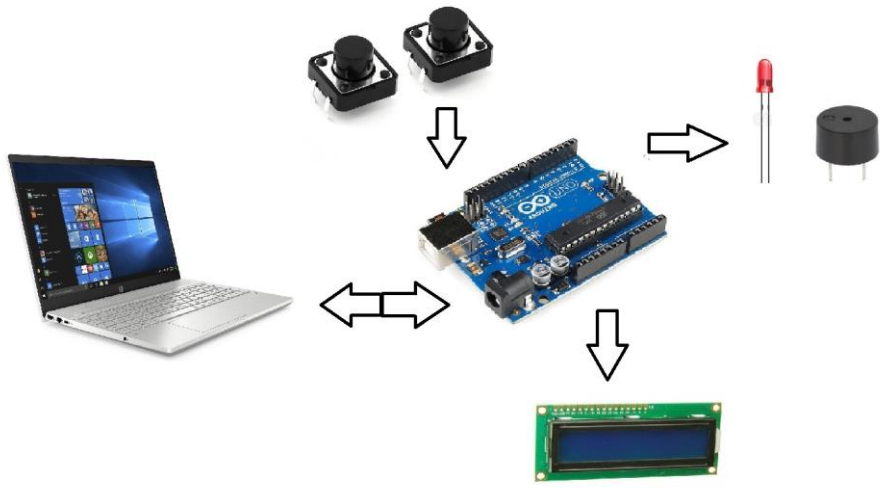
DATOS DE LOS MIEMBROS DEL GRUPO.

NOMBRE	APELLIDOS	Nº MATRÍCULA	EMAIL
Rafael	Rodríguez Palomo	55435	rafael.rodriguezp@alumnos.upm.es

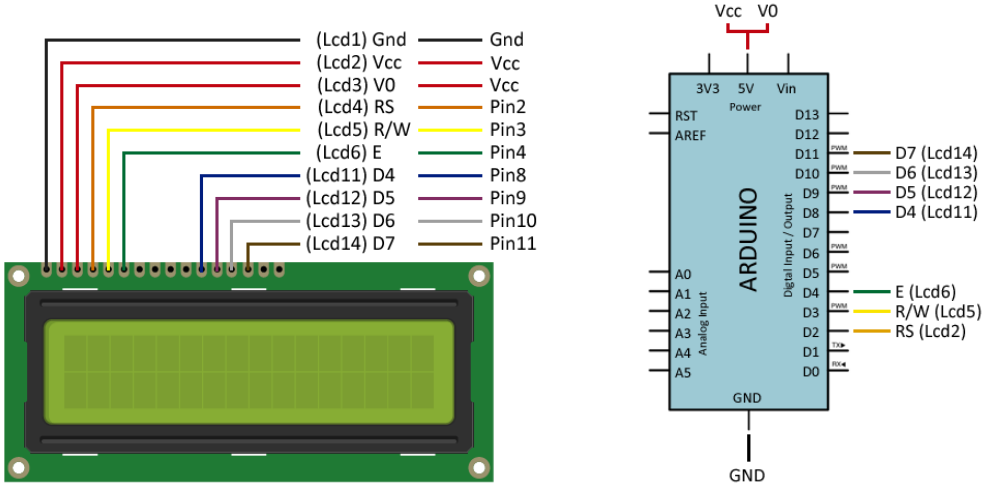
TÍTULO Y RESUMEN.

DECODIFICADOR/CODIFICADOR MORSE
Dispositivo electrónico basado en un microcontrolador capaz de recibir señales de varios pulsadores. El dispositivo debe diferenciar entre puntos o rayas según la duración de la pulsación y mostrará el mensaje traducido en una pantalla LCD. A su vez el dispositivo podrá traducir a código morse una frase introducida desde el ordenador y emitirla a través de un LED y un buzzer.

REQUISITOS FUNCIONALES
1. Es necesario un ordenador para el funcionamiento del dispositivo. **2.** Al iniciar el dispositivo es necesario elegir una opción (decodificador o codificador).

HARDWARE Y FUNDAMENTOS TÉCNICOS.

INTERFAZ – PANTALLA LCD
Las pantallas LCD (Liquid Cristal Display) son una de las formas más sencillas y económicas de dotar de un display a un autómata. El Hitachi HD44780 es uno de los controladores de LCDs más ampliamente extendidos por su sencillez y bajo precio. Está diseñado para controlar LCDs monocromos de hasta 80 caracteres alfanuméricos y símbolos. También dispone de una pequeña memoria RAM para configurar nuestros propios caracteres o dibujos. Las pantallas LCD se fabrican en distintos tamaños, siendo comunes 16x02 (2 líneas de 16 caracteres). Conectar directamente un LCD a Arduino requiere una gran cantidad de pines. Suele ser aconsejable emplear un adaptador a bus I2C. Por este motivo es posible que en el controlador de audio no sea posible incluir una pantalla LCD por el número de pines disponibles en la placa Arduino.

El IDE de Arduino incorpora de serie la librería LiquidCrystal, para controlar pantallas LCD HD44770 de forma sencilla. La librería LiquidCrystal varios ejemplos sobre su uso. Resulta aconsejable examinarlos, en general es sencilla de utilizar.



Ejemplo de código:

Por ejemplo, el siguiente sketch muestra el uso de las funciones de la librería para mostrar los textos "Linea 1" y "Linea 2" en una pantalla de 16x02.

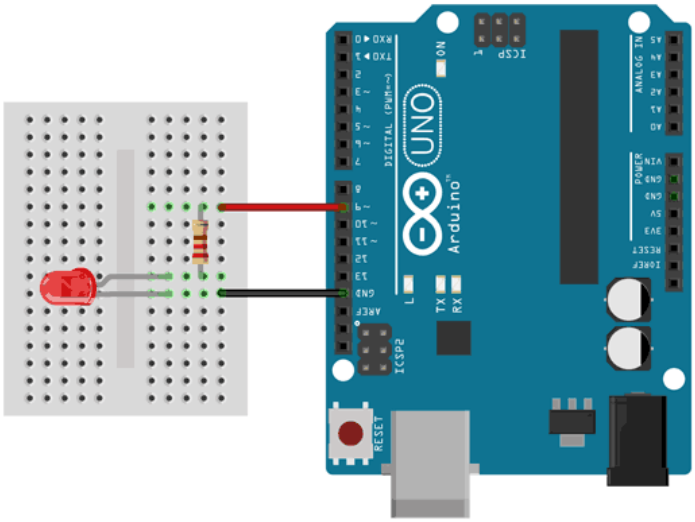
```
/* Conexión
PIN2 - LCD4 (RS)
PIN3 - LCD5 (R/W)
PIN4 - LCD6 (E)

PIN0 - LCD11 (D4)
PIN9 - LCD12 (D5)
PIN10 - LCD13 (D6)
PIN11 - LCD14 (D7)
*/

#include <LiquidCrystal.h>

LiquidCrystal lcd(2, 3, 4, 0, 9, 10, 11); //(RS,EW, E, D4,D5, D6, D7)

void setup()
{
   lcd.begin(16, 2);   // Inicia un LCD 16x02 (columnas,fila)
   lcd.setCursor(0, 0);   // Pone el cursor en las coordenadas (0,0)
   lcd.print("Linea 1");   // Escribe el LCD
   lcd.setCursor(0, 1);   // Ponemos el cursor en la segunda fila
   lcd.print("Linea 2");   // Escribe el LCD
}

void loop()
{
   //Apagar cursor
   lcd.noCursor();
   delay(500);

   //Apagar encender cursor
   lcd.cursor();
   delay(500);
}
```

LED

Un LED es un diodo emisor de luz, es decir, un tipo particular de diodo que emite luz al ser atravesado por una corriente eléctrica. Los diodos (emisor de luz, o no) son unos de los dispositivos electrónicos fundamentales.

Un diodo es una unión de dos materiales semiconductores con dopados distintos. Esta diferencia de dopado hace que genere una barrera de potencial, que como primera consecuencia hace que el paso de corriente en uno de los sentidos no sea posible.
Tienen polaridad, es decir, solo dejan pasar la corriente en un sentido. Por tanto, tenemos que conectar correctamente la tensión al dispositivo.
La patilla larga debe ser conectada al voltaje positivo (ánodo), y la corta al voltaje negativo (cátodo).

Es imprescindible el uso de una resistencia porque el LED puede quemarse si se supera el voltaje de polarización.



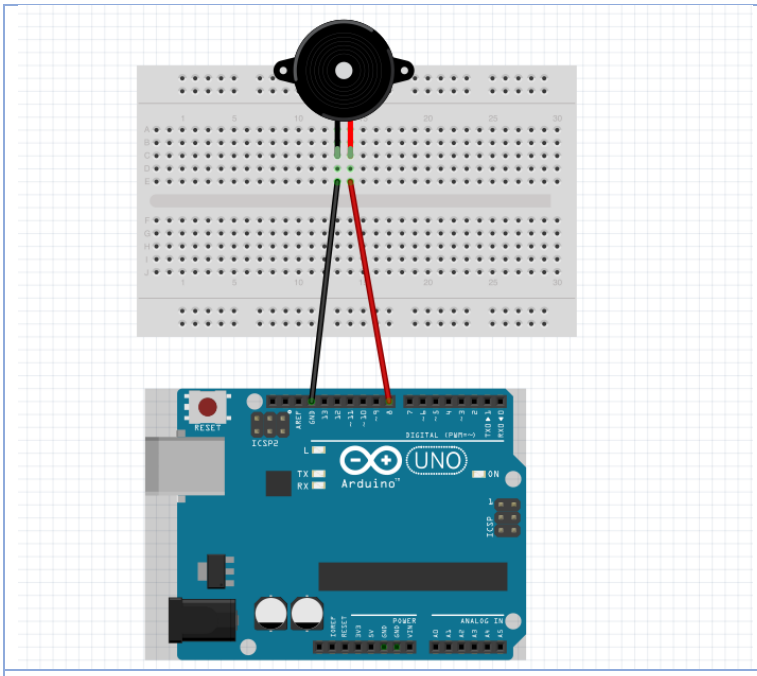
Ejemplo de código:

```
const int ledPIN = 9;

void setup() {
  Serial.begin(9600);    //iniciar puerto serie
  pinMode(ledPIN , OUTPUT);  //definir pin como salida
}

void loop(){
  digitalWrite(ledPIN , HIGH);   // poner el Pin en HIGH
  delay(1000);                   // esperar un segundo
  digitalWrite(ledPIN , LOW);    // poner el Pin en LOW
  delay(1000);                   // esperar un segundo
}
```

BUZZER

Los buzzer activos, en ocasiones denominados zumbadores, son dispositivos que generan un sonido de una frecuencia determinada y fija cuando son conectados a tensión.
Un buzzer activo incorpora un oscilador simple por lo que únicamente es necesario suministrar corriente al dispositivo para que emita sonido. En oposición, los buzzer pasivos necesitan recibir una onda de la frecuencia.

Al incorporar de forma interna la electrónica necesaria para hacer vibrar el altavoz un buzzer activo resulta muy sencillo de conectar y controlar. Además, no suponen carga para el procesador ya que no este no tiene que generar la onda eléctrica que se convertirá en sonido.
En contra posición, tienen la desventaja de que no podremos variar el tono del sonido emitido, por lo que no podremos realizar melodías, cosa que si podremos hacer con los buzzer pasivos. En este caso usaremos un buzzer pasivo.



Ejemplo de código:

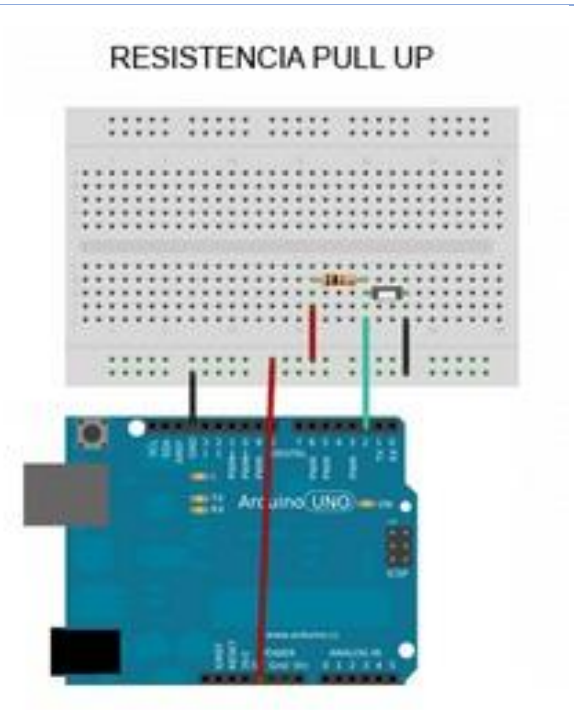
```
const int pin = 9;

void setup() {
  pinMode(pin, OUTPUT);  //definir pin como salida
}

void loop(){
  digitalWrite(pin, HIGH);   // poner el Pin en HIGH
  delay(5000);               // esperar 5 segundos
  digitalWrite(pin, LOW);    // poner el Pin en LOW
  delay(20000);               // esperar 20 segundos
}
```

PULSADOR

Para conectar un pulsador a arduino entra en juego el concepto de resistencias de Pull Down y Pull Up. Aunque ambos casos son muy similares, el montaje y el tipo de resistencia a usar dependerá de si queremos que al accionar el pulsador o interruptor leamos un valor LOW o HIGH.

Para medir un valor LOW al accionar el interruptor, podemos conectar el PIN de forma fija a 5V, y a 0V a través del interruptor. Con el interruptor abierto leeríamos HIGH, y al cerrar el interruptor se forzaría 0V en el PIN, por lo que leeríamos LOW.
Para medir un valor HIGH al accionar el interruptor, podemos conectar el PIN de forma fija a 0V, y a 5V a través del interruptor. Con el interruptor abierto leeríamos LOW, y al cerrar el interruptor se forzaría 5V en el PIN, por lo que leeríamos HIGH.

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.
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.



Ejemplo de código:

```
const int inputPin = 2;

int value = 0;

void setup() {
  Serial.begin(9600);
  pinMode(inputPin, INPUT);
}

void loop(){
  value = digitalRead(inputPin);  //lectura digital de pin

  //mandar mensaje a puerto serie en función del valor leído
  if (value == HIGH) {
      Serial.println("Encendido");
  }
  else {
      Serial.println("Apagado");
  }
  delay(1000);
}
```