

# Traductor de Código Morse – OLED + Botón Borrar

## Trabajo Práctico Final – Laboratorio de Computación I (UNSAM)

Grupo 8

• **Integrantes:**

- Matías Mantiñán
- Juan Fornes
- Santino Pelle
- Diego Mammana

• **Docentes:**

Matías José Gagliardo y Pedro Facundo Iriso

### Resumen del proyecto

Se implementa un sistema basado en Arduino UNO que recibe entradas mediante un pulsador y traduce dichas pulsaciones temporizadas al alfabeto Morse, reproduciendo puntos y rayas mediante un buzzer y mostrando el carácter reconocido en una pantalla OLED 128x32. El sistema opera empleando una máquina de estados de dos estados (UP/DOWN), un contador de flancos y control por tiempos para clasificar las pulsaciones (punto/raya) y los silencios (fin de letra / palabra). También se agregó un botón dedicado para borrar la pantalla. En Tinkercad se simula con LCD 16x2 I2C.

### Descripción técnica del hardware

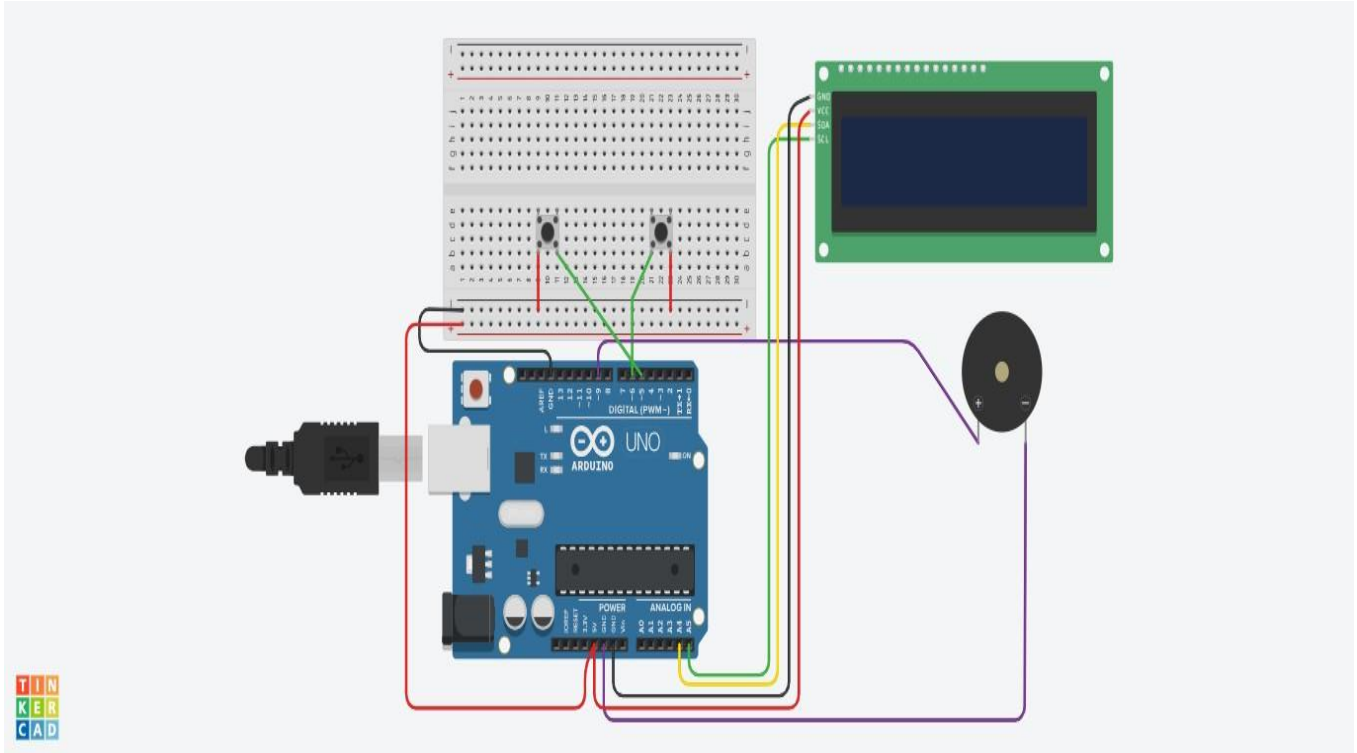
**Placa:** Arduino UNO R3.

**Entradas:** D6 (Morse) y D5 (Borrar), ambas con INPUT\_PULLUP;

**Salidas:** 1 buzzer en D9 controlado con tone(); Display OLED 128x32 por I2C (SDA→A4, SCL→A5, VCC 5V, GND).

Componente	Pin Arduino	Motivo
Pulsador Morse (a GND)	D6 – INPUT_PULLUP	1 en reposo, 0 al presionar; evita resistencias externas.
Pulsador Borrar (a GND)	D5 – INPUT_PULLUP	Limpia la pantalla por flanco.
Buzzer (+)	D9 (PWM)	Salida auditiva de pulsación.
Buzzer (–)	GND	Retorno común.
OLED VCC	5V	Alimentación.
OLED GND	GND	Referencia común.
OLED SDA	A4	Datos I2C.
OLED SCL	A5	Reloj I2C.

## Diagrama esquemático / conexiones



## Contador de flancos y control por tiempo

En cada iteración del `loop()` se leen los estados actuales de los botones y se comparan con sus estados anteriores (`estado_actual` y `estado_actual2`). Cuando `estado_nuevo != estado_actual` se detecta un flanco en el botón Morse (cambio `UP ↔ DOWN`) y se calcula la duración del estado anterior con `duracion = millis() - ultimo_cambio_t`.

En el flanco `DOWN → UP` esa duración se guarda en `down_duracion` y se usa en `LeerMorse()` para clasificar la pulsación como punto o raya (según `DOT_DURACION`, `DASH_DURACION` y `TOLERANCIA`). En el flanco `UP → DOWN` la duración se interpreta como tiempo de silencio y se compara con `MEDIUM_GAP` y `LONG_GAP` para decidir si se cierra una letra o se inserta un espacio entre palabras.

Cuando no hay flancos, el programa sigue midiendo el tiempo en estado `UP` para cerrar letras/palabras automáticamente por inactividad. El botón de borrado también se atiende por flanco: si `estado_nuevo2 != estado_actual2`, se limpia la pantalla con `clearDisplay()` y se reinicia el cursor.

**Fragmento central del código fuente donde se implementan la máquina de estados,  
el contador de flancos y el control por tiempo:**

**// Estados y tiempos**

```
enum Estado { UP, DOWN };
enum Estado2 { UP2, DOWN2 };
```

```
Estado estado_actual;
Estado2 estado_actual2;
unsigned long ultimo_cambio_t = 0;
unsigned long down_duracion = 0;
```

```
void loop() {
  Estado estado_nuevo = (digitalRead(BOTON_PIN) == HIGH) ? UP : DOWN;
  Estado2 estado_nuevo2 = (digitalRead(BOTON_LIMPIAR) == HIGH) ? UP2 : DOWN2;
```

**// Flancos del botón Morse**

```
if (estado_nuevo != estado_actual) {
  unsigned long duracion = millis() - ultimo_cambio_t;

  if (estado_actual == DOWN) { // DOWN → UP
    down_duracion = duracion;
    LeerMorse(); // punto / raya
  } else { // UP → DOWN
    unsigned long up_duracion = duracion;
    if (up_duracion >= LONG_GAP - TOLERANCIA ||
        up_duracion >= MEDIUM_GAP - TOLERANCIA) {
      char c = LeerCaracter(); // letra
      display.print(c ? c : '?');
      if (up_duracion >= LONG_GAP - TOLERANCIA) display.print(' ');
      display.display();
      LimpiaMorse();
    }
  }

  ultimo_cambio_t = millis();
  estado_actual = estado_nuevo;
}
```

**// Flanco del botón borrar**

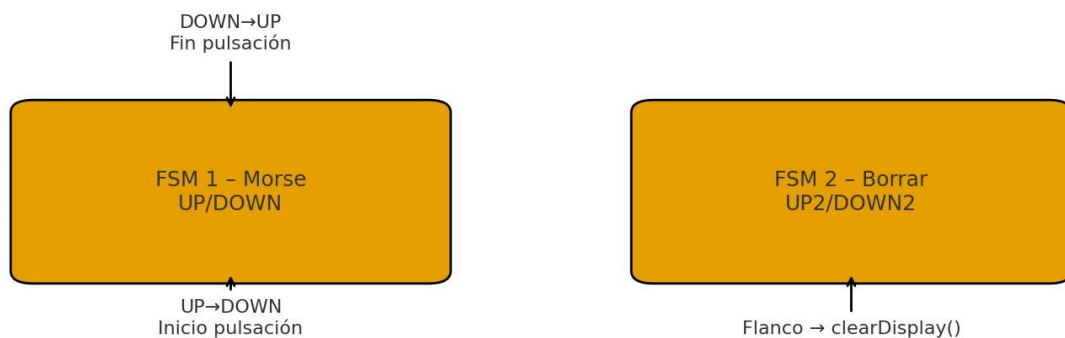
```
if (estado_nuevo2 != estado_actual2) {
  display.clearDisplay();
  display.setCursor(0, 0);
  display.display();
  estado_actual2 = estado_nuevo2;
}
}
```

## Descripción del software y su estructura

### Estructura:

- (1) **Definiciones de tiempo:** (DOT\_DURACION, TOLERANCIA, gaps de 1t/3t/7t).
- (2) **Entradas/Salidas:** (BOTON\_PIN, BOTON\_LIMPIAR, BUZZER\_PIN, OLED),
- (3) **Tabla del alfabeto Morse**
- (4) **Máquina de estados:** con enum Estado (UP/DOWN) + Borrar (UP2/DOWN2) + medida de tiempos por millis().
- (5) **Funciones de utilidad:** limpieza de buffer, clasificación punto/raya, decodificación de letra
- (6) **Loop principal:** detecta flancos, reproduce tono durante DOWN y escribe en LCD al cerrar letras/palabras.

## Máquina de Estados



Mediciones:  
DOWN:  $\approx 1t$  punto,  $\approx 3t$  raya  
UP: 1t sigue; 3t fin letra; 7t espacio

## Conclusiones del grupo

Se cumplieron los requisitos de E/S múltiples, contador de flancos, control lógico por tiempo y máquina de estados. El enfoque por tiempos parametrizables y tolerancias permitió una decodificación robusta del Morse ingresado manualmente. El OLED mejora la presentación y el segundo botón facilita la operación.

## Capturas del sistema funcionando

