Parte A: Control de LED por Botones

Uso de Variables Predefinidas

El código utiliza las siguientes variables predefinidas del Controllino:

LEDs (Salidas Digitales):

- **CONTROLLINO_D0, CONTROLLINO_D1, CONTROLLINO_D2**: LEDs del lado izquierdo de la matriz
- **CONTROLLINO_D6, CONTROLLINO_D7, CONTROLLINO_D8**: LEDs centrales
- **CONTROLLINO_D12, CONTROLLINO_D13, CONTROLLINO_D14**: LEDs del lado derecho

Botones (Entradas Digitales):

- **CONTROLLINO_I16**: Botón 1 para espiral normal
- **CONTROLLINO_I17**: Botón 2 para espiral inversa
- **CONTROLLINO_I18**: Botón 3 para reiniciar y apagar todos los LEDs

Estructuras y Punteros Implementados

Array de LEDs:

```cpp
int leds[9] = {
  CONTROLLINO_D0, CONTROLLINO_D6, CONTROLLINO_D12,
  CONTROLLINO_D13, CONTROLLINO_D14, CONTROLLINO_D8,
  CONTROLLINO_D2, CONTROLLINO_D1, CONTROLLINO_D7
};
```

Los LEDs están organizados en orden espiral para facilitar la secuencia de encendido.

Uso de Punteros:

- **`int* puntero = leds`**: Puntero base que apunta al inicio del array
- **`int* led_actual = leds`**: Puntero que apunta al LED que debe encenderse en cada momento
- **`*(puntero + i)`**: Aritmética de punteros para acceder a cada LED del array
- **`*led_actual`**: Desreferenciación del puntero para obtener el pin del LED actual

Descripción del Funcionamiento del Sistema

Control de Modos:

El sistema opera en tres modos diferentes controlados por los botones:

- **Modo 0**: Sistema detenido (todos los LEDs apagados)
- **Modo 1**: Espiral normal (encendido secuencial en sentido horario)
- **Modo 2**: Espiral inversa (encendido secuencial en sentido antihorario)

Funciones Principales:

`espiralNormal()`:

- Enciende el LED actual usando `digitalWrite(*led_actual, HIGH)`
- Cada 500ms apaga el LED actual y avanza al siguiente usando aritmética de punteros
- Incrementa el contador y actualiza `led_actual = puntero + contador`
- El patrón se repite infinitamente mientras el modo esté activo

`espiralInversa()`:

- Similar a espiral normal pero decrementa el contador
- Cuando el contador llega a -1, se reinicia a 8 para mantener la secuencia inversa
- Actualiza el puntero de la misma manera: `led_actual = puntero + contador`

`reiniciar()`:

- Apaga todos los LEDs usando un bucle y aritmética de punteros: `digitalWrite(*(puntero + i), LOW)`

`reiniciarSecuencia()`:

- Llama a `reiniciar()` para apagar todos los LEDs
- Resetea las variables `contador = 0` y `led_actual = puntero`
- Actualiza el tiempo para comenzar inmediatamente la nueva secuencia

Características Técnicas:

- **Timing no bloqueante**: Utiliza `millis()` para controlar el tiempo sin detener la ejecución
- **Intervalo de 500ms**: Tiempo entre cada cambio de LED
- **Detección de cambio de modo**: Solo cambia de modo cuando se detecta una transición en los botones
- **Uso eficiente de memoria**: Los punteros permiten acceso directo a los pines sin copiar datos

El sistema responde inmediatamente a los botones y mantiene la secuencia de LEDs de manera fluida y continua.