

# Microcontroladores: Laboratorio 3

1<sup>st</sup> Mateo Lecuna

Ingeniería en Mecatrónica

Universidad Tecnológica (UTEC)

Fray Bentos, Uruguay

mateo.lecuna@estudiantes.utec.edu.uy

2<sup>nd</sup> Mateo Sánchez

Ingeniería en Mecatrónica

Universidad Tecnológica (UTEC)

Maldonado, Uruguay

mateo.sanchez@estudiantes.utec.edu.uy

**Resumen**—Se presenta el desarrollo de cinco sistemas embebidos implementados sobre el microcontrolador ATmega328P, correspondientes al Laboratorio 3 de la unidad curricular *Tecnologías de Microprocesamiento*. Los ejercicios abordaron distintos escenarios de control digital y adquisición de señales, incluyendo un plóter cartesiano de dos ejes, un sistema de control de temperatura, un control de motor por referencia analógica, una matriz RGB controlada mediante joystick y una cerradura electrónica basada en lector *RFID RC522*.

Cada proyecto integró periféricos de entrada/salida, sensores analógicos, comunicación serial UART e interfaces de visualización (LCD I2C y Python/MATLAB para gráficas). Se implementaron estrategias de control mediante modulación por ancho de pulso (PWM) y lógica secuencial programada en lenguaje C. Los resultados obtenidos demostraron el correcto funcionamiento de los sistemas tanto en simulación como en hardware físico, validando el aprendizaje y aplicación práctica de los conceptos de microprocesamiento y control embebido.

Este trabajo permitió integrar técnicas de control, comunicación y procesamiento digital en un entorno embebido real, consolidando competencias prácticas esenciales para el diseño mecatrónico.

**Index Terms**—Microcontroladores, ATmega328P, Sistemas embebidos, Modulación PWM, Control de motor, Joystick, Matriz RGB, RFID, UART.

## I. INTRODUCCIÓN

El presente laboratorio tuvo como objetivo aplicar los conocimientos adquiridos en la unidad curricular *Tecnologías de Microprocesamiento*, a través del diseño, programación e implementación de distintos sistemas embebidos basados en el microcontrolador ATmega328P. A partir de este dispositivo se desarrollaron cinco ejercicios experimentales que integran control digital, manejo de periféricos, adquisición de señales analógicas, y comunicación serial.

Los ejercicios planteados corresponden a los siguientes problemas:

- **Problema A – Control de Plotter:** Implementación de un sistema de control para un plóter cartesiano de dos ejes, empleando motores paso a paso y una válvula solenoide para el trazado de figuras. El objetivo principal consiste en coordinar los movimientos de los ejes X e Y respetando los límites definidos por sensores, logrando la representación precisa de figuras geométricas.
- **Problema B – Sistema de Control de Temperatura:** Desarrollo de un sistema de regulación térmica

mediante el sensor LM35. El microcontrolador mide la temperatura en intervalos regulares y controla un calefactor y un ventilador por modulación PWM, con posibilidad de ajustar el punto medio de temperatura a través de un menú UART.

- **Problema C – Control de Motor:** Implementación de un sistema de control de posición analógico en el que dos potenciómetros determinan la referencia y la posición actual de un motor de corriente continua. El microcontrolador regula el giro y la velocidad del motor mediante una señal PWM, buscando igualar ambos valores. El comportamiento se monitorea por puerto serial y se analiza mediante gráficas de evolución temporal.
- **Problema D – Matriz RGB con Joystick:** Diseño de una interfaz interactiva en la que el movimiento del joystick desplaza un LED encendido dentro de una matriz RGB. El sistema detecta las direcciones de movimiento mediante entradas analógicas (ADC) y genera un cambio de color aleatorio al presionar el pulsador integrado en el joystick.
- **Problema E – Cerradura RFID:** Construcción de una cerradura electrónica inteligente basada en un lector *RC522 RFID*, con almacenamiento del identificador autorizado en la memoria EEPROM, interfaz de visualización LCD I2C y señalización por LEDs y buzzer. El sistema gestiona el registro, validación y actualización de tarjetas, así como la comunicación UART para monitoreo externo.

Cada uno de estos ejercicios aborda distintos aspectos del control digital, incluyendo la comunicación UART, la modulación por ancho de pulso (PWM), la lectura de señales analógicas mediante el ADC interno y el manejo de periféricos de entrada/salida (I/O). En conjunto, el laboratorio permitió consolidar la comprensión del funcionamiento del microcontrolador ATmega328P y su aplicación en proyectos de automatización y control embebido.

En las siguientes secciones se presenta el desarrollo teórico, metodológico y experimental de los sistemas implementados, junto con los resultados obtenidos y las conclusiones correspondientes.

## II. MARCO TEÓRICO

### II-A. *Plotter*

Este dispositivo permite el trazado y graficación de figuras e ilustraciones bidimensionales en un espacio definido por su chasis. Posee un carro deslizante que se traslada horizontalmente sobre el eje Y, y un cabezal que varía su posición en X pero se encuentra enlazado al carro mediante rieles, permitiendo el movimiento en los ejes X e Y del solenoide (utilizado como actuador del eje Z). Estas piezas están unidas mediante una correa dentada a cada motor PAP, siendo el motor Y el que controla el carro y el motor X el que controla el cabezal.

Para el control y la alimentación de los motores, se cuenta con un controlador de motores PAP, cuya labor es servir como nexo entre el microcontrolador y los motores, tomando la señal de control del uC y transformándola a señal de potencia para los motores conectados.

El controlador se encuentra conectado al uC mediante los pines GPIO del IC. Acorde al conexionado interno, el pin PB3 corresponde al reloj del motor X, el PB4 controla la dirección de dicho motor, junto con PB5, que controla la activación o desactivación del motor X. De igual manera que con los componentes en X, los controles del eje Y son PC3, PC4 y PC5 para Reloj Y, Dirección Y y Enable Y respectivamente, con la inclusión del Solenoide como parte del registro C en el pin PC0.

Los límites están dados por dos sensores inductivos en los pines PD2 y PD3, siendo éstos YA y YD, y contando con el pin PD5 conectado a un LED indicador integrado para señalización.

Para mover un motor, se debe tener el pin Enable encendido, y posteriormente elegir el sentido de giro mediante el estado del pin Dir. Por último, se debe encender y apagar el pin Clk con un intervalo de tiempo entre conmutaciones definido por el usuario (se recomienda 1 ms, que equivale a 500 Hz).

### II-B. *Sistema de Control de Temperatura*

### II-C. *Control de motor*

El presente experimento tiene como objetivo implementar un control proporcional de velocidad y dirección en un motor de corriente continua (DC). Este tipo de control se basa en ajustar el ciclo de trabajo (*duty cycle*) de una señal PWM (*Pulse Width Modulation*) en función de la diferencia entre un valor de referencia y la retroalimentación medida, generando así un comportamiento lineal y estable en la respuesta del motor [1].

Para lograrlo, se utilizó un **punto H L298N**, que permite invertir la polaridad del motor mediante dos entradas digitales de control, posibilitando el giro en ambos sentidos [2]. El punto H se alimentó con 5 V, mientras que las señales de control provinieron directamente del microcontrolador **ATmega328P** [3]. El pin ENA del L298N se controló con una señal PWM generada por el temporizador interno `Timer1`.

El `Timer1` fue configurado en modo *Fast PWM* de 8 bits, con salida no invertida sobre el pin OC1A (D9) y prescaler

igual a 1, alcanzando una frecuencia de conmutación de aproximadamente 62,5 kHz. Este modo permite generar un ciclo de trabajo variable entre 0 y 255, directamente proporcional al valor calculado por el control proporcional en el código.

El sistema empleó dos potenciómetros conectados a las entradas analógicas A0 (referencia) y A1 (realimentación), cuya lectura se realizó mediante el conversor analógico-digital (ADC) de 10 bits del microcontrolador [3]. La diferencia entre ambas señales ( $e = V_{\text{ref}} - V_{\text{act}}$ ) determina la dirección de giro y la magnitud del PWM aplicado. Para evitar oscilaciones cercanas al punto de equilibrio, se definió una zona muerta de  $\pm 10$  unidades ADC. De este modo, cuando la diferencia entre potenciómetros se mantiene dentro de ese rango, el motor permanece detenido y ambos LEDs indicadores se apagan.

El valor mínimo de PWM efectivo fue calibrado en 180 ( $\approx 70\%$  del ciclo de trabajo), ya que valores inferiores no lograban vencer la inercia del motor bajo carga. El sistema se comportó de forma estable, respondiendo proporcionalmente a los cambios de referencia y deteniéndose al alcanzar la coincidencia entre ambas señales.

### II-D. *Matriz RGB con joystick*

Este experimento tuvo como objetivo implementar el control de una matriz de LEDs RGB tipo **WS2812B** de 8x8, utilizando un **joystick analógico** para desplazar un píxel activo en la pantalla. El microcontrolador **ATmega328P** genera la señal digital codificada a una frecuencia de 800 kHz, respetando el protocolo de transmisión GRB (Green-Red-Blue) característico del WS2812B [4]. Este protocolo requiere una temporización precisa en el orden de cientos de nanosegundos, lo que se logró mediante retardos calibrados y la generación manual de pulsos de nivel lógico en el código.

El joystick utilizado es del tipo analógico-digital, compuesto por dos potenciómetros ortogonales (ejes X e Y) y un pulsador central (SW), permitiendo leer desplazamientos horizontales, verticales y acciones de selección [?]. Los ejes se conectaron a las entradas analógicas A0 y A1, mientras que el pulsador se conectó a un pin digital configurado como entrada con resistencia de *pull-up* interna.

Durante el desarrollo se consideró inicialmente implementar un mapeo matricial continuo, donde las coordenadas del LED activo dependieran de un cálculo analógico proporcional. Sin embargo, este enfoque resultó complejo de calibrar y sensible al ruido, por lo que se optó finalmente por un control basado en condiciones discretas, que permitió un funcionamiento más estable y predecible. Además, se incorporó una zona muerta en el eje central para evitar desplazamientos involuntarios del LED, y un retardo (*cooldown*) entre movimientos para suavizar la respuesta visual.

El resultado fue un sistema fluido, capaz de representar el movimiento del joystick con buena estabilidad y cambio de color del LED activo mediante el pulsador. La comprensión del protocolo WS2812B y la sincronización precisa de la

señal permitieron reforzar los conocimientos sobre control digital de periféricos de alta velocidad.

### II-E. Cerradura RFID

El objetivo de esta práctica fue desarrollar una cerradura electrónica que gestionara el acceso mediante tarjetas **RFID** del tipo **MIFARE 1K 13.56 MHz**, utilizando el lector **RC522** en conjunto con el microcontrolador **ATmega328P**. El sistema permite registrar y eliminar tarjetas, validar accesos y visualizar los estados mediante una pantalla LCD I2C y LEDs indicadores.

El lector RC522 se comunica con el microcontrolador a través del bus SPI (*Serial Peripheral Interface*), empleando una velocidad de reloj de 2MHz y operando en modo maestro [5]. Las señales SDA, SCK, MOSI, MISO y RST fueron conectadas a los pines digitales D10, D13, D11, D12 y D9, respectivamente. La pantalla LCD de 16x2 caracteres se controló mediante un expansor **PCF8574** conectado al bus I2C, utilizando las líneas SDA → A4 y SCL → A5 [6]. El LED verde y el LED rojo indicaron los estados de validación de acceso, mientras que un pulsador externo activó el modo de programación.

La lógica del sistema se implementó mediante un autómata de estados finitos (FSM), compuesto por las siguientes etapas: inicio (espera de tarjeta), lectura del UID, verificación, acceso permitido, acceso denegado y modo programación. En este último estado, el sistema permite dar de alta o baja tarjetas mediante la detección del pulsador. Los UID autorizados se almacenaron en la memoria EEPROM interna del microcontrolador, con un tamaño máximo de 64 registros de 4 bytes cada uno [3].

Durante las pruebas se observó que la lectura del RC522 era más sensible al tiempo de respuesta en modo de programación, lo cual se mitigó ajustando los retardos entre lecturas y el control de la bandera de detección. El sistema final demostró una lectura estable, almacenamiento persistente y una interfaz de usuario clara, validando el correcto funcionamiento conjunto de los periféricos SPI, I2C y UART en una misma arquitectura embebida.

## III. METODOLOGÍA

### III-A. Plotter

Este plotter CNC cuenta con dos motores paso a paso controlados por el ATmega328P mediante drivers de control. Esto permite que todo el sistema pueda ser controlado con 10 pines, tratándose estos del control directo de los motores (Enable, Step/Clock y Dir), el procesamiento de señales de entrada (Sensores inductivos YA e YD), y control de periféricos (LED, Solenoide). Para mover los motores, se debe activar el pin Enable, establecer la dirección de giro con Dir, y finalmente alternar el estado del reloj correspondiente, para que éste le envíe pulsos de señal al motor.

Para crear las figuras más adelante, es conveniente definir los movimientos básicos de control como funciones para posteriormente manejar la posición a un más alto nivel.

Por ejemplo, se define la función U (up) como la activación de los pines PC5 (Enable Y) y PC4 (Dir Y), con la intermitencia del pin PC3 (ClkY/StepY) para enviarle la señal al driver de activar en secuencia las bobinas del motor PAP.

Con estas funciones básicas (U,D,L,R), es posible crear todas las otras figuras que se deseen, haciendo uso de la posibilidad de ingresar la cantidad de pasos en cierto sentido para simplificar las funciones. Podemos, por ejemplo, crear un cuadrado con tamaño ajustable si definimos su función con una entrada de enteros, y luego establecemos 4 movimientos en sus correspondientes sentidos, y que tomarán como entrada de datos la propia entrada de datos de la función *cuadrado()*.

### III-B. Sistema de control de temperatura

### III-C. Control de motor

El desarrollo de este módulo se centró en la implementación de un sistema de control analógico-digital para un motor de corriente continua, utilizando el microcontrolador **ATmega328P** y un driver **L298N**. El objetivo fue lograr que la velocidad y dirección del motor se ajustaran de forma proporcional a la diferencia entre dos señales analógicas: una de referencia y otra de realimentación.

**Configuración del hardware:** El montaje se realizó sobre una placa Arduino Uno. Se emplearon dos potenciómetros lineales de 10 kΩ, conectados a las entradas analógicas A0 (referencia) y A1 (posición actual del eje). El motor DC fue acoplado mecánicamente a uno de los potenciómetros para obtener la señal de realimentación. El puente H L298N se conectó de la siguiente forma:

- ENA → pin digital D9 (salida PWM, OC1A)
- IN1 → pin digital D4
- IN2 → pin digital D5
- OUT1/OUT2 → terminales del motor
- +12V → fuente de alimentación del motor (batería o fuente externa)
- 5V y GND conectados al Arduino

Se incorporaron dos LEDs indicadores (en paralelo a las líneas IN1 e IN2) para representar visualmente la dirección de giro (**DER** o **IZQ**). El motor se alimentó desde una fuente externa de 12 V aislada del suministro lógico de 5 V, compartiendo masa común con el Arduino.

**Configuración del software:** El programa fue desarrollado en lenguaje C utilizando el entorno *Microchip Studio*. Se configuró el temporizador `Timer1` en modo *Fast PWM* de 8 bits, con prescaler igual a 1, alcanzando una frecuencia de conmutación de aproximadamente 62,5 kHz. Las señales analógicas de los potenciómetros fueron digitalizadas mediante el ADC interno de 10 bits, utilizando AVcc como referencia y prescaler 64 (frecuencia de conversión 250 kHz). Para reducir el ruido, cada lectura se promedió a partir de cuatro muestras consecutivas.

El control se implementó mediante un esquema proporcional simple, con una zona muerta de  $\pm 10$  unidades ADC. Si la diferencia entre la referencia y la realimentación

estaba dentro de esa banda, el sistema detenía el motor. En caso contrario, se ajustaba el ciclo útil del PWM según la ganancia  $K_p = 1/4$ , con un valor mínimo de 180 (de un total de 255) para garantizar el arranque bajo carga.

**Comunicación y monitoreo:** Se habilitó la UART a  $115\,200\text{ bits s}^{-1}$  para enviar por puerto serie los valores de referencia, realimentación, ciclo PWM y dirección del motor. Los datos se imprimieron en formato CSV con el encabezado `P3C,ref,act,pwm,dir`, lo que permitió registrar la evolución temporal y graficarla en Python o MATLAB para analizar la respuesta del sistema.

**Procedimiento experimental:** El procedimiento constó de tres etapas:

1. Calibración del umbral mínimo de PWM necesario para vencer el par de arranque del motor acoplado al potenciómetro.
2. Validación del control proporcional variando manualmente el potenciómetro de referencia.
3. Registro de las variables por UART para evaluar la linealidad y estabilidad del sistema.

Durante las pruebas se comprobó que el control respondía de manera fluida, con inversión inmediata del sentido de giro al cruzar el punto medio y sin oscilaciones perceptibles en la zona de equilibrio.

En conjunto, la metodología aplicada permitió obtener un control de posición analógica estable y reproducible, integrando adquisición de datos, procesamiento digital y modulación PWM en un circuito de bajo costo y fácil replicación.

### III-D. Matriz RGB con joystick

En este módulo se desarrolló una interfaz interactiva que permite controlar el desplazamiento de un LED encendido dentro de una matriz  $8 \times 8$  de LEDs RGB direccionables (**WS2812B**) mediante un joystick analógico. El objetivo principal fue integrar la lectura de señales analógicas, el manejo digital de una cadena de LEDs y la generación de colores pseudoaleatorios, explorando el control de movimiento dentro de un entorno bidimensional.

**Configuración del hardware:** El sistema se montó sobre una placa Arduino Uno (ATmega328P). Las conexiones físicas principales fueron las siguientes:

- PB0  $\rightarrow$  pin DIN de la matriz WS2812B (línea de datos)
- A0  $\rightarrow$  eje X del joystick (lectura ADC)
- A1  $\rightarrow$  eje Y del joystick (lectura ADC)
- PD2  $\rightarrow$  botón SW del joystick (entrada digital con *pull-up* interno)
- Alimentación: 5 V y GND comunes a matriz y joystick

La matriz RGB se alimentó directamente desde 5 V, asegurando que la corriente total no superara los 800 mA al limitar el brillo global a un 30 %. La comunicación con la cadena WS2812B se realizó a través de una única línea de datos a 800 kHz.

**Configuración del software:** El código se programó en C y utiliza una rutina de temporización precisa (*bit-banging*) para generar el protocolo de comunicación requerido por

los WS2812B. Durante la transmisión se deshabilitan las interrupciones (`cli()`/`sei()`) para garantizar la integridad del tren de bits GRB (24 bits por LED). Se implementó una escala global de brillo de 80/255 para evitar saturación visual.

El joystick se lee mediante el ADC de 10 bits, utilizando AVcc como referencia y prescaler 64 (frecuencia de conversión 250 kHz). Se definieron umbrales de decisión con una *zona muerta* de  $\pm 90$  cuentas alrededor del punto medio (512), lo que evita movimientos no deseados por ruido. Cada eje se evalúa independientemente para determinar desplazamientos discretos en X o Y, desplazando el LED activo una celda por evento. Se aplicó un *cooldown* de algunos milisegundos para limitar la frecuencia de actualización y dar una sensación de movimiento fluido.

**Intento de mapeo matricial:** Inicialmente se intentó implementar un control basado en un *mapeo matricial*, que asociaba directamente cada par (X, Y) de coordenadas analógicas con un índice de LED. Sin embargo, la complejidad del cálculo y la respuesta no lineal del joystick dificultaron su calibración. Finalmente se optó por un control condicional con límites discretos y desplazamientos unitarios, logrando un funcionamiento más predecible y sencillo de depurar. Esta decisión simplificó la depuración y mejoró la respuesta visual del sistema.

**Gestión del color y del botón SW:** Al detectar la pulsación del botón del joystick (SW), el sistema genera un nuevo color RGB pseudoaleatorio mediante un LFSR de 8 bits. El nuevo color se aplica únicamente al LED actual, permaneciendo activo hasta la siguiente pulsación. Los eventos se envían por UART a  $9600\text{ bit s}^{-1}$  para registro y diagnóstico.

**Procedimiento experimental:** El proceso de validación se realizó en tres etapas:

1. Verificación de la lectura estable del ADC y calibración de los umbrales de zona muerta.
2. Ensayo del movimiento del LED dentro de la matriz verificando los límites horizontales y verticales.
3. Prueba del cambio de color con pulsaciones consecutivas del botón SW.

El sistema presentó una respuesta fluida y estable, con detección confiable de los ejes y transición de color sin errores de sincronización.

En conjunto, la metodología implementada permitió integrar la adquisición analógica, la comunicación digital temporizada y la generación de color en un mismo entorno embebido, demostrando la versatilidad del ATmega328P en tareas de control interactivo.

### III-E. Cerradura RFID

El objetivo de este módulo fue desarrollar una cerradura electrónica que gestionara el acceso mediante tarjetas **RFID** del tipo **MIFARE 1K 13.56 MHz**, utilizando el lector **RC522** en conjunto con el microcontrolador **ATmega328P**. El sistema permite registrar y eliminar tarjetas, validar accesos y visualizar los estados mediante una pantalla LCD I2C y LEDs indicadores.

**Configuración del hardware:** El montaje se realizó sobre una placa Arduino Uno, conectando los módulos externos del siguiente modo:

- **RC522 → SPI:** SDA → D10, SCK → D13, MOSI → D11, MISO → D12, RST → D9.
- **LCD I2C (PCF8574):** dirección 0x27, líneas SDA → A4, SCL → A5.
- **LED Verde → D6, LED Rojo → D7.**
- Pulsador de modo programación → D2.
- Alimentación: 5 V y GND comunes.

El módulo RC522 se alimentó a 3.3 V y se cuidó la compatibilidad lógica entre niveles (5 V del ATmega328P y 3.3 V del lector). Se incluyeron resistencias de protección y un capacitor de desacoplo cercano a la alimentación del módulo para mejorar la estabilidad eléctrica [7].

**Configuración del software:** El programa se estructuró en lenguaje C, utilizando los periféricos SPI, UART e I2C del ATmega328P. La comunicación con el RC522 se realiza mediante una biblioteca propia (RC522.c/h) que implementa las funciones básicas de inicialización, detección de tarjeta y lectura del UID (*Unique Identifier*). La interfaz SPI opera en modo maestro con frecuencia de 2 MHz (prescaler = Fosc/8).

El flujo del programa se organizó como un **autómata de estados finitos (FSM)** con los siguientes estados principales:

1. **Inicio:** Muestra en el LCD “Listo para leer”.
2. **Lectura:** Detecta tarjeta y lee su UID.
3. **Verificación:** Compara el UID con los almacenados en EEPROM.
4. **Acceso:** Si coincide, enciende el LED verde y muestra “Acceso concedido”.
5. **Error:** Si no coincide, enciende el LED rojo y muestra “Acceso denegado”.
6. **Programación:** Permite agregar o eliminar UID mediante la pulsación del botón.

**Gestión de la memoria EEPROM:** Se utilizó la memoria EEPROM interna del ATmega328P para almacenar los UID autorizados. Cada tarjeta ocupa una posición de cuatro bytes, y el sistema puede almacenar hasta 64 identificadores. Las operaciones de lectura y escritura se realizaron con las funciones estándar de AVR LibC (`eeeprom_read_byte()` y `eeeprom_write_byte()`), evitando accesos repetitivos para prolongar la vida útil de la memoria. Mediante la UART se implementaron comandos para listar, borrar o limpiar registros, facilitando la administración sin necesidad de reprogramar el dispositivo.

**Interfaz LCD I2C y comunicación serial:** El módulo LCD de 16×2 caracteres, controlado por el expansor PCF8574, muestra mensajes informativos del sistema: detección de tarjeta, acceso concedido/denegado, modo programación y número de tarjetas registradas. La UART a 9600 bit s<sup>-1</sup> se utilizó para depuración y monitoreo en consola, mostrando el UID leído y el estado de las operaciones, además de permitir la recepción de comandos textuales.

**Procedimiento experimental:** El proceso de validación incluyó las siguientes etapas:

1. **Prueba de lectura RFID:** se verificó la detección estable de tarjetas y la correcta lectura del UID.
2. **Prueba de almacenamiento:** se registraron varias tarjetas y se comprobó la persistencia de datos tras reiniciar el sistema.
3. **Prueba de acceso:** se ensayó la validación de tarjetas autorizadas y el rechazo de no registradas.
4. **Prueba de interfaz:** se observó la coherencia de mensajes en LCD y UART, así como la correcta respuesta de los LEDs.

**Observaciones:** Durante la puesta a punto, se observó que la lectura del RC522 era más sensible en el modo de inicio que en programación, debido a pequeñas variaciones de temporización en el lazo de consulta. Se solucionó aumentando los tiempos de espera entre intentos de lectura y reforzando la condición de detección del UID. El sistema final demostró una operación confiable, con una lectura estable y almacenamiento persistente.

En síntesis, esta metodología combinó la comunicación SPI, I2C y UART en un mismo sistema embebido, aplicando control secuencial y almacenamiento no volátil para lograr una cerradura RFID funcional y extensible.

## IV. RESULTADOS

### IV-A. Plotter

El plotter puede dibujar figuras simples con un tamaño ajustable (cuadrado, cruz, triángulo), pero el círculo posee un tamaño fijo debido a que está hecho con una serie predefinida de movimientos a modo de LUT.

La resolución observada es aceptable para ilustraciones simples y líneas rectas, sin embargo puede ser considerablemente mejor si se activaran a la vez ambos motores, produciendo de esta forma un movimiento sin interrupciones, y por consiguiente, sin escalonado de trazo.

### IV-B. Sistema de control de temperatura

### IV-C. Control de motor

Durante las pruebas del sistema de control proporcional se registraron los valores de referencia, realimentación, ciclo de trabajo del PWM y dirección de giro, transmitidos mediante UART en formato CSV. Los datos fueron visualizados en tiempo real mediante una aplicación desarrollada en *Python*, lo que permitió verificar la respuesta dinámica del sistema.

Se observó una relación proporcional estable entre la diferencia de tensiones de los potenciómetros y la velocidad del motor. En la zona muerta ( $\pm 10$  unidades ADC) el motor permaneció detenido, y al aumentar la diferencia de referencia la velocidad respondió suavemente en ambos sentidos. El valor mínimo de PWM efectivo ( $180 \approx 70\%$ ) permitió superar el par de arranque bajo carga, evitando bloqueos y mejorando la estabilidad del control.

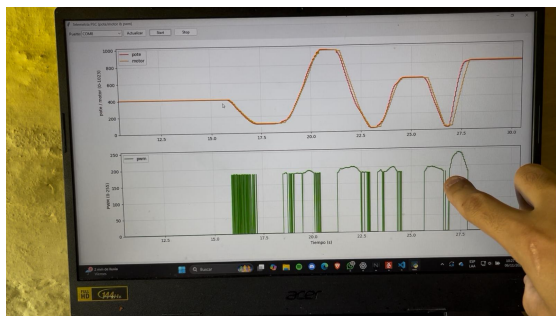


Figura 1. Gráfico obtenido en tiempo real mediante Python, mostrando la relación entre la posición de los potenciómetros y el valor del PWM aplicado al motor.

#### IV-D. Matriz RGB con joystick

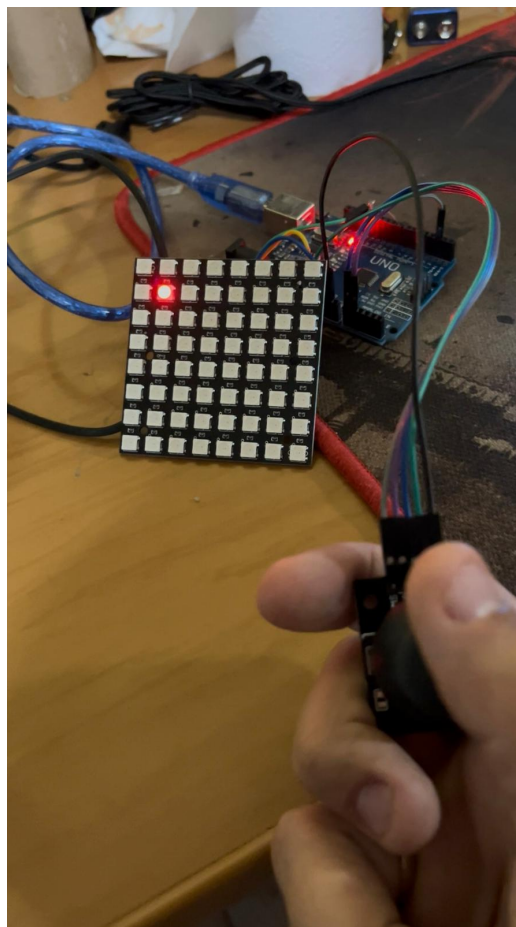


Figura 2. Desplazamiento del LED activo dentro de la matriz RGB controlado mediante el joystick.

El sistema respondió correctamente a los desplazamientos del joystick en ambos ejes, mostrando el movimiento del LED activo dentro de la matriz RGB de  $8 \times 8$ . La zona muerta definida permitió eliminar falsas detecciones y estabilizar la posición cuando el joystick se encontraba en reposo. El retardo de actualización (*cooldown*) aplicado entre lecturas contribuyó a un desplazamiento controlado y sin parpadeos.

Al presionar el botón del joystick, el color del LED activo cambiaba de forma pseudoaleatoria, generando distintos tonos visibles en pantalla. Aunque inicialmente se intentó implementar un mapeo matricial continuo, la solución final basada en condiciones discretas resultó más sencilla, robusta y predecible.



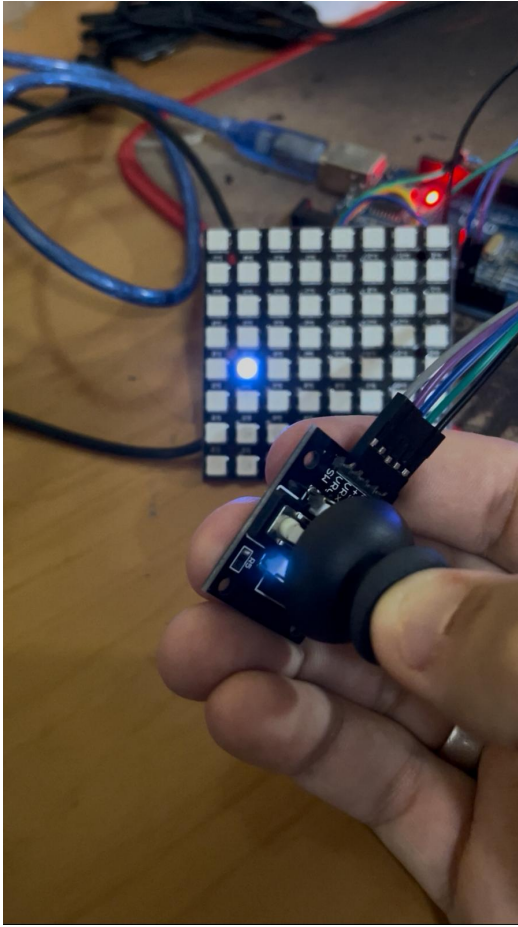


Figura 3. Cambio de color del LED activo al presionar el botón del joystick.

#### IV-E. Cerradura RFID

El sistema RFID demostró un funcionamiento estable en la lectura y verificación de tarjetas **MIFARE 1K**. Durante las pruebas, el LCD mostró correctamente los mensajes de estado (“Acerque tarjeta”, “Acceso concedido”, “Acceso denegado”), y los LEDs indicaron visualmente la validez o rechazo del acceso. Las tarjetas registradas permanecieron almacenadas en la memoria EEPROM incluso tras reiniciar el sistema, confirmando la persistencia de datos.

Aunque los comandos UART de administración no se interpretaron correctamente, el sistema respondió con mensajes informativos (“Comandos: ADD, DEL, LIST”) ante cualquier entrada, verificando la comunicación serial básica. La detección en el modo de programación resultó más sensible a los tiempos de lectura, pero se estabilizó tras ajustar los retardos entre consultas.

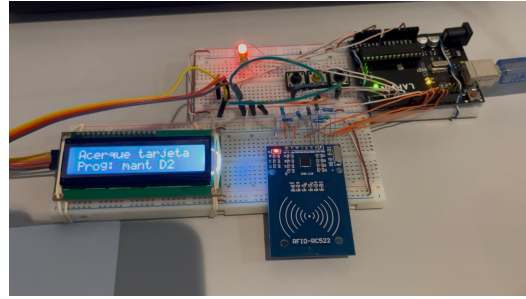


Figura 4. Pantalla inicial del sistema RFID con mensaje de espera para detección de tarjeta.

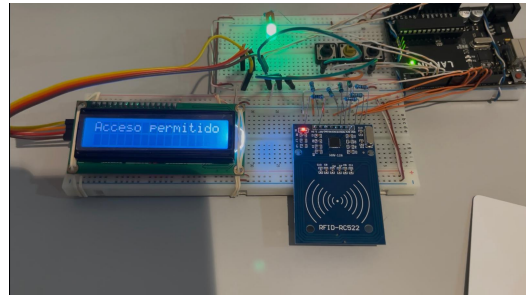


Figura 5. Validación correcta: acceso concedido y encendido del LED verde.



Figura 6. Lectura de tarjeta no registrada: acceso denegado y encendido del LED rojo.

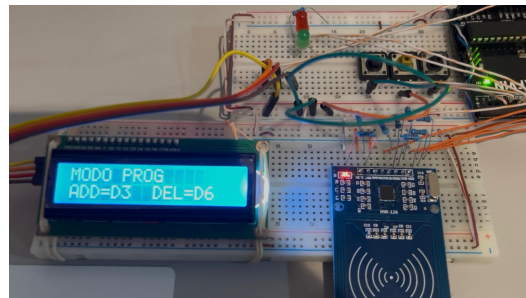


Figura 7. Modo de programación del sistema, con opción de alta y baja de tarjetas.

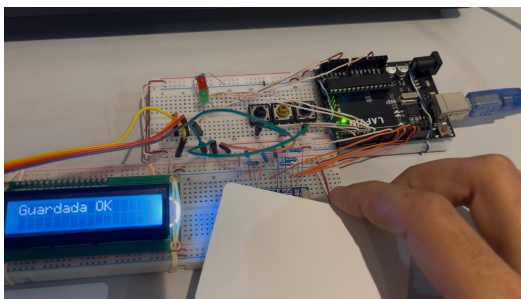


Figura 8. Pantalla correspondiente a la opción de agregar (ADD) una nueva tarjeta al sistema.



Figura 9. Pantalla correspondiente a la opción de eliminar (DEL) una tarjeta previamente registrada.

En conjunto, los resultados obtenidos demostraron el correcto funcionamiento de los tres módulos desarrollados: el control analógico proporcional del motor, la interfaz de usuario con joystick y matriz RGB, y la cerradura de usuario con joystick y matriz RGB, y la cerradura RFID con almacenamiento persistente. Cada sistema integró periféricos distintos del microcontrolador, validando su funcionamiento simultáneo y la estabilidad del hardware en condiciones reales de operación.

## V. CONCLUSIONES

El desarrollo de los ejercicios propuestos en este laboratorio permitió integrar y aplicar múltiples conceptos de electrónica digital, control y comunicación de periféricos en sistemas embebidos basados en el microcontrolador **ATmega328P**. Cada módulo abordó un enfoque distinto del control y la interacción hardware–software, consolidando la comprensión de los principios fundamentales de adquisición, procesamiento y respuesta dentro de un entorno de tiempo real.

En el **Ejercicio A**, Habiendo llevado a cabo la consigna solicitada, cabe destacar la experiencia en programación con C AVR adquirida, además del conocimiento absorbido respecto al manejo y programación de Máquinas de Control Numérico Computarizado. Además de obtener conocimiento práctico en resolución de problemas, fué posible realizar un diagnóstico mecánico de las piezas físicas y los problemas presentados por la máquina, así como explorar posibles mejoras al sistema (como la adición de un final de carrera en el eje X. Se lograron dibujar múltiples figuras haciendo uso de la máquina.

En el **Ejercicio B**, ...

En el **Ejercicio C**, se logró implementar un control proporcional simple de velocidad y dirección de un motor de corriente continua, utilizando señales analógicas de referencia y realimentación. El sistema presentó un comportamiento estable y predecible, con una respuesta lineal ajustada mediante modulación PWM. Este ejercicio permitió afianzar la comprensión del funcionamiento del ADC, del temporizador Timer1 en modo *Fast PWM* y del control proporcional como base para futuros controladores más complejos.

En el **Ejercicio D**, se implementó un control interactivo de una matriz de LEDs RGB mediante un joystick analógico, combinando lectura analógica, control digital temporizado y generación de color. La experiencia permitió comprender el funcionamiento de la comunicación unidireccional a alta velocidad (protocolo **WS2812B**) y la importancia de gestionar correctamente los tiempos de actualización, zonas muertas y saturación de color. Además, se exploró un enfoque alternativo de mapeo matricial que, si bien no se implementó exitosamente, aportó experiencia valiosa en la gestión de coordenadas discretas y calibración de entradas analógicas.

En el **Ejercicio E**, se desarrolló una cerradura electrónica mediante identificación **RFID**, integrando comunicación SPI, I2C, UART y almacenamiento no volátil en EEPROM. El sistema permitió registrar y verificar tarjetas, mostrando mensajes en un LCD y señalizando los estados mediante LEDs. A pesar de que los comandos seriales no se ejecutaron completamente, el prototipo demostró un funcionamiento estable y una correcta lectura de los UID, validando la integración de múltiples periféricos en una misma arquitectura embebida.

En conjunto, las prácticas realizadas fortalecieron las habilidades de diseño, implementación y depuración de sistemas digitales, promoviendo la comprensión del trabajo modular y el uso coordinado de distintos buses de comunicación. Los resultados obtenidos evidencian el cumplimiento de los objetivos planteados y sientan las bases para el desarrollo de proyectos más avanzados, como sistemas de control en lazo cerrado, automatizaciones o aplicaciones de **IoT** basadas en microcontroladores AVR.

## REFERENCIAS

- [1] SparkFun Electronics. Pulse width modulation (pwm) tutorial. Fundamentos de modulación por ancho de pulso en microcontroladores. [Online]. Available: <https://learn.sparkfun.com/tutorials/pulse-width-modulation>
- [2] STMicroelectronics, L298 — Dual Full-Bridge Driver (H-Bridge) — Datasheet, 2022, puente H utilizado para el control bidireccional del motor DC. [Online]. Available: <https://www.st.com/resource/en/datasheet/l298.pdf>
- [3] Microchip Technology Inc., ATmega328P — 8-bit AVR Microcontroller Datasheet, 2023, documento técnico del microcontrolador utilizado en todos los ejercicios del laboratorio. [Online]. Available: [https://www1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P\\_Datasheet.pdf](https://www1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf)



- [4] Worldsemi, *WS2812B — Intelligent Control RGB LED Integrated Light Source*, 2020, especificaciones de temporización, protocolo GRB y control digital de LEDs direccionables. [Online]. Available: <https://cdn-shop.adafruit.com/datasheets/WS2812B.pdf>
- [5] NXP Semiconductors, *MFRC522 — Contactless Reader IC for ISO/IEC 14443A*, 2021, circuito lector RFID utilizado en el sistema de cerradura electrónica. [Online]. Available: <https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf>
- [6] Nexperia, *PCF8574 — Remote 8-bit I/O Expander for I2C-bus*, 2022, expansor I/O utilizado en la interfaz I2C del LCD. [Online]. Available: <https://www.nxp.com/docs/en/data-sheet/PCF8574.pdf>
- [7] Electronics Tutorials. Spi communication — basics and operation. Principios de comunicación SPI y configuración maestro-esclavo. [Online]. Available: [https://www.electronics-tutorials.ws/io/io\\_4.html](https://www.electronics-tutorials.ws/io/io_4.html)
- [8] —. H-bridge motor control theory. Conceptos de puente H y conmutación de polaridad. [Online]. Available: <https://www.electronics-tutorials.ws/blog/h-bridge.html>
- [9] Adafruit Industries. (2014) Brushed dc motors — adafruit motor selection guide. Conceptos de motores DC, control y selección de componentes. [Online]. Available: <https://learn.adafruit.com/adafruit-motor-selection-guide/dc-motors>
- [10] SparkFun Electronics. (2017) Thumb joystick hookup guide. Funcionamiento del joystick analógico, ejes, pulsador y lectura por ADC. [Online]. Available: <https://learn.sparkfun.com/tutorials/thumb-joystick-hookup-guide>
- [11] —. (2016) Rfid basics. Introducción a la tecnología RFID, fundamentos y flujo de lectura del UID. [Online]. Available: <https://learn.sparkfun.com/tutorials/rfid-basics/all>

## ANEXOS

### A. Evidencias experimentales

A continuación se presentan imágenes complementarias de los montajes y pruebas realizadas durante el Laboratorio 3, correspondientes a los módulos de control de motor, matriz RGB con joystick y cerradura RFID. Estas evidencias respaldan los resultados expuestos en la Sección 1 a la Sección 9.

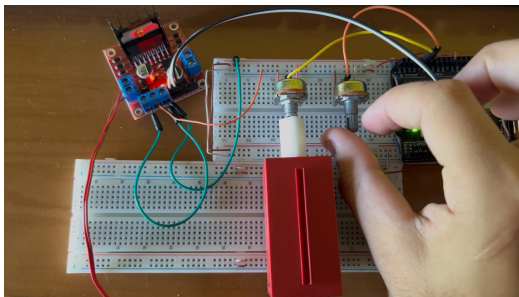


Figura 10. Conexión del puente H L298N con el motor DC y los potenciómetros de referencia y realimentación.

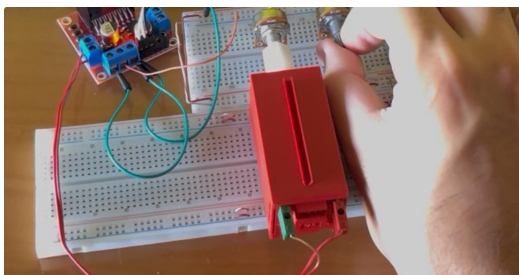


Figura 11. Detalle del acoplamiento mecánico entre el motor DC y el potenciómetro utilizado para obtener la señal de realimentación.



Figura 12. Captura del gráfico en Python mostrando la evolución temporal de las señales de referencia, realimentación y PWM.

### A.1 Control de Motor DC:

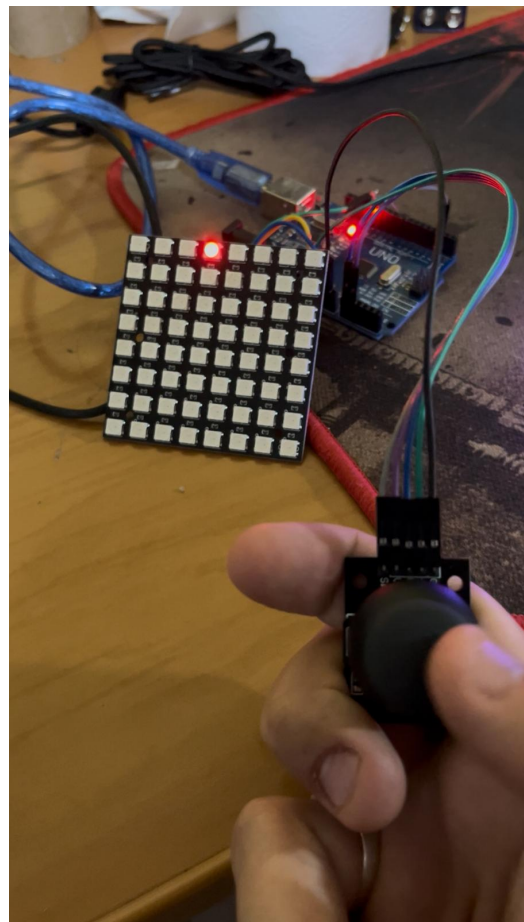


Figura 13. Montaje de la matriz WS2812B conectada al joystick analógico y al microcontrolador ATmega328P.

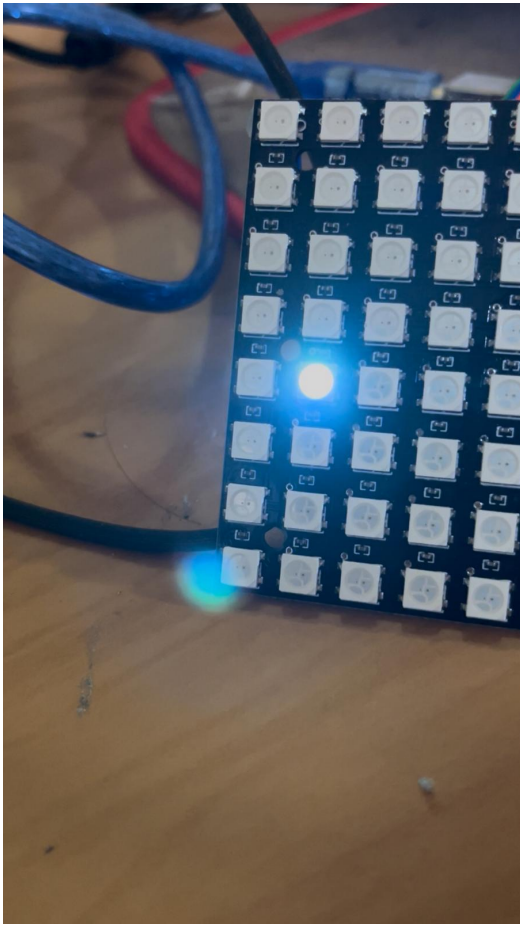


Figura 14. Cambio de color del LED activo al presionar el botón del joystick.

### A.2 Matriz RGB con Joystick:

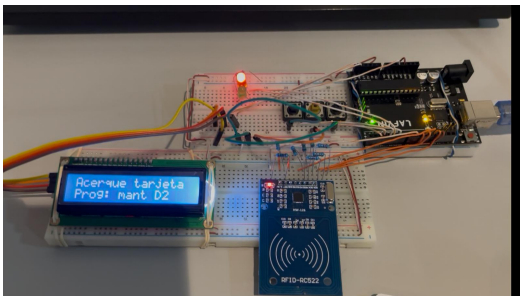


Figura 15. Montaje del sistema de cerradura RFID con lector RC522, pantalla LCD I2C y LEDs indicadores.

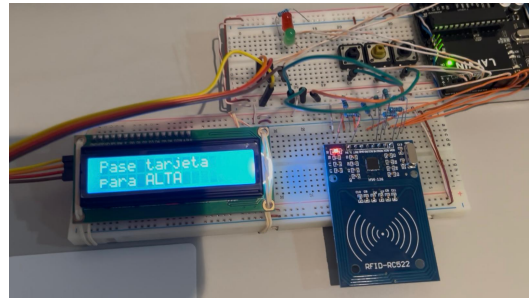


Figura 16. Pantalla LCD mostrando la opción de agregar (ADD) una nueva tarjeta al sistema.

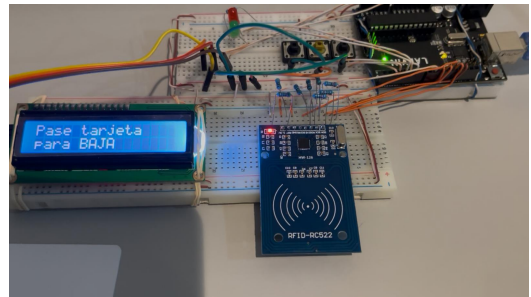


Figura 17. Pantalla LCD mostrando la opción de eliminar (DEL) una tarjeta registrada.

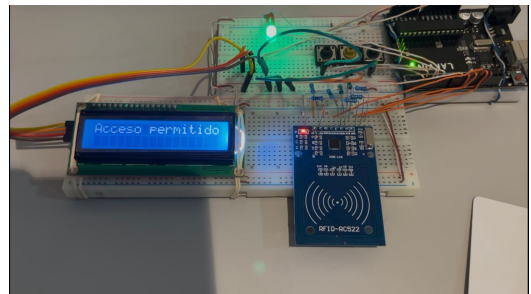


Figura 18. Estado de validación correcta: "Acceso concedido" con encendido del LED verde.

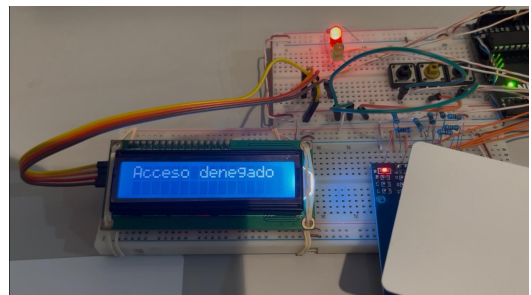


Figura 19. Estado de validación incorrecta: "Acceso denegado" con encendido del LED rojo.

### A.3 Cerradura RFID:

### B. Fragmentos de código relevantes

A continuación se incluyen fragmentos representativos del código implementado en lenguaje C, correspondientes a las rutinas principales de lectura, control y comunicación.

Listing 1. Inicialización del ADC y lectura promedio

```
static void adc_init(void) {  
    ADMUX = (1 << REFS0); // AVcc como referencia  
    ADCSRA = (1 << ADEN) | (1 << ADPS2) | (1 << ADPS1); // prescaler 64  
}  
  
static uint16_t adc_read_avg(uint8_t ch, uint8_t n) {  
    uint32_t acc = 0;  
    for (uint8_t i = 0; i < n; i++) acc += adc_read(ch);  
    return (uint16_t)(acc / n);  
}
```

Listing 2. Control de dirección y PWM del motor DC

```
if (mag <= DEAD_ADC) {  
    OCR1A = 0;  
    dir_leds_stop();  
} else {  
    uint16_t inc = (mag * KP_NUM) / KP_DEN;  
    uint16_t raw = MIN_PWM + inc;  
    if (raw > 255) raw = 255;  
    pwm = (uint8_t)raw;  
  
    if (e > 0) dir_leds_DER();  
    else      dir_leds_IZQ();  
  
    OCR1A = pwm;  
}
```

### C. Recursos complementarios

- **Repositorio GitHub del proyecto:** <https://github.com/MateoLecuna/Tec.Micro/tree/main/lab3>
- **Datasheets y documentación técnica:** disponibles en la bibliografía.
- **Videos de evidencia experimental:** carpeta lab3/Evidencias.