Microcontroladores: Laboratorio 1

1st Mateo Lecuna
Ingeniería en Mecatrónica
Universidad Tecnológica (UTEC)
Fray Bentos, Uruguay
mateo.lecuna@estudiantes.utec.edu.uy

2nd Mateo Sanchez
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.

Keywords: ATmega328P, sistemas embebidos, modulación PWM, control de motor, joystick, matriz RGB, RFID, UART, microcontroladores.

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

II-B. Sistema de Control de Temperatura

II-C. Control de Motor

El sistema de control de motor implementado tiene como propósito igualar el valor leído por un potenciómetro de referencia con el valor obtenido de otro potenciómetro acoplado mecánicamente al eje de un motor de corriente continua (DC). Para lograrlo, el microcontrolador **ATmega328P** compara ambas señales analógicas, determina la dirección de giro correspondiente y ajusta la velocidad del motor mediante modulación por ancho de pulso (*PWM*), utilizando un puente H **L298N** como etapa de potencia.

Motor de corriente continua (DC): El motor DC convierte energía eléctrica en mecánica mediante la interacción entre el campo magnético del estator y la corriente en el rotor. La velocidad de giro es proporcional al voltaje aplicado, mientras que el sentido de rotación depende de la polaridad. En este sistema, el motor opera dentro de un rango de alimentación de 6–12 V, con un comportamiento estable a partir de un ciclo útil mínimo equivalente a una señal PWM de aproximadamente 180/255. Este valor fue determinado experimentalmente, ya que con ciclos menores (120/255) el par resultaba insuficiente para vencer la carga impuesta por el acoplamiento con el potenciómetro del eje.

Puente H L298N: El circuito integrado L298N permite invertir la polaridad aplicada al motor mediante el control de cuatro transistores dispuestos en configuración de puente H. El microcontrolador gobierna las entradas `IN1` e `IN2` para establecer el sentido de giro, y la entrada `ENA` se conecta a un pin PWM para modular la velocidad. Esta arquitectura posibilita un control bidireccional eficiente, evitando el uso de relevadores o conmutadores mecánicos. El módulo integra además diodos de rueda libre para protección ante corrientes inversas generadas por la inductancia del motor.

Potenciómetros de referencia y realimentación: Se emplearon dos potenciómetros lineales de 10 kΩ. El primero, conectado al pin analógico `A0`, define la posición o velocidad de referencia (*setpoint*), mientras que el segundo, vinculado físicamente al eje del motor y conectado al pin `A1`, mide la posición actual. Ambos valores son convertidos por el conversor analógico–digital (ADC) de 10 bits del ATmega328P, proporcionando una resolución suficiente para el control analógico proporcional. Para mejorar la estabilidad de lectura, cada medición promedia cuatro muestras consecutivas, lo que reduce el efecto del ruido eléctrico.

Control por modulación PWM: El ATmega328P genera la señal PWM en el pin `OC1A` (D9) mediante el temporizador interno `Timer1` configurado en modo Fast PWM de 8 bits, con prescaler $N = 1$, alcanzando una frecuencia de conmutación de aproximadamente 62.5 kHz. Este valor resulta inaudible para el oído humano y proporciona una respuesta suave en el motor. En el código se implementó un control proporcional simple que ajusta el ciclo útil del PWM en función de la diferencia entre los valores de referencia y realimentación. Cuando la diferencia es menor a una zona muerta de 10 unidades ADC, el sistema detiene el motor y apaga los indicadores de dirección.

Comunicación serial y visualización: Durante la ejecución, el sistema transmite por UART los valores de ambos potenciómetros, el sentido de giro y el valor del PWM aplicado, en formato CSV (`ref, act, pwm, dir`). Esto permite representar las variables en tiempo real mediante Python o MATLAB, facilitando el análisis de la evolución temporal y la respuesta del control.

Observación experimental del umbral de arranque: Durante las pruebas se determinó empíricamente un umbral mínimo de ciclo útil para garantizar el arranque bajo carga. Sin acoplamiento mecánico, el motor iniciaba la rotación con PWM $\approx 120/255$; sin embargo, al acoplar el potenciómetro al eje, el incremento de par resistente elevó el umbral. Tras incrementos sucesivos, se fijó un mínimo operativo de **180/255**, que aseguró arranque fiable y evitó bloqueos intermitentes en transitorios de baja referencia. Este valor se adoptó como límite inferior del control para preservar la robustez del sistema.

Detalles de implementación y consideraciones prácticas: El PWM se generó en el pin D9 (OC1A) mediante Timer1 en modo Fast PWM de 8 bits, con prescaler $N = 1$ (frecuencia $\approx 62{,}5$ kHz), evitando ruido audible en el motor. Dado que el L298N es un puente H con transistores Darlington, frecuencias tan elevadas pueden incrementar pérdidas de conmutación; en caso de calentamiento, es razonable reducir la frecuencia (por ejemplo, prescaler $N = 8$, $\approx 7{,}8$ kHz) sin afectar perceptiblemente el control. Para la adquisición analógica se utilizó el ADC interno a 10 bits con prescaler 64 (≈ 250 kHz) y promediado de cuatro muestras por canal, logrando lecturas estables a 1 kHz de lazo. Si se prioriza la precisión de 10 bits, el prescaler 128 (≈ 125 kHz) es una alternativa acorde a la recomendación del fabricante.

En síntesis, este módulo combina los principios de control analógico con el manejo digital de periféricos del microcontrolador, integrando adquisición de señales (ADC), procesamiento de control proporcional y generación de salida PWM, bajo una estructura de hardware sencilla y de bajo costo.

II-D. Matriz RGB con Joystick

Este módulo implementa una interfaz interactiva en la que un *cursor* luminoso (un único LED encendido) se desplaza sobre una matriz 8×8 de LEDs RGB direccionables **WS2812B**. La posición se controla con un *joystick analógico* (dos potenciómetros ortogonales) y un pulsador que, al activarse, asigna un color RGB pseudoaleatorio al LED activo. El sistema integra: (i) adquisición analógica por ADC, (ii) lectura digital del pulsador con resistencia de *pull-up* interna, y (iii) generación del protocolo de temporización de los WS2812B directamente por software (*bit-banging*), respetando el orden de bytes **GRB** y la ventana de *reset* ($\sim$80 μs) para el *latch* del tren de datos. :contentReference[oaicite:0]index=0

Joystick analógico y umbrales de decisión: Cada eje del joystick se lee mediante el conversor A/D de 10 bits del

ATmega328P, empleando AVcc como referencia y prescaler 64 (frecuencia de reloj del ADC $\approx$ 250 kHz), lo que otorga una resolución de 1024 niveles por canal. Se define un punto medio en MID $= 512$ y una *zona muerta* de amplitud DZ $= 90$ cuentas, generando umbrales TH_LO $= 512 - 90$ y TH_HI $= 512 + 90$. Valores por debajo/encima de esos umbrales se interpretan como movimientos discretos en $\{\uparrow, \downarrow, \leftarrow, \rightarrow\}$ (una celda por evento). Para evitar repeticiones espurias por ruido y velocidad de muestreo, se usa un *cooldown* entre pasos. En la implementación, el eje **X** está invertido (*mapping* físico), de modo que desplazamientos a un lado producen el índice decreciente y al contrario para el otro. :contentReference[oaicite:1]index=1

Pulsador con entrada digital y cambio de color: El pulsador del joystick (`SW`) se conecta a `PD2` como entrada con *pull-up* interno (*activo en bajo*). Al detectar un flanco de pulsación, el sistema genera un nuevo color mediante un generador pseudoaleatorio *LFSR* de 8 bits, actualizando los tres canales (R, G, B) del LED activo y registrando por UART el nuevo triplete RGB.

Matriz WS2812B y protocolo GRB a 800 kHz: Los WS2812B integran controlador por LED, por lo que *no requieren multiplexado externo*. La cadena de 64 LEDs se modela como un arreglo lineal de longitud $N = 64$ (8×8), direccionado por el índice $i \in [0, 63]$. Para transmitir color, se envían 24 bits por LED en orden **G-R-B**, con tiempos de $\mathrm{T_{0H}}$, $\mathrm{T_{0L}}$, $\mathrm{T_{1H}}$, $\mathrm{T_{1L}}$ compatibles con 800 kHz. Durante el envío se **deshabilitan interrupciones** (`cli()/sei()`) para garantizar la precisión de ciclo, y se deja un $\sim$80 μs de *reset* al final para *latch*. Además, se aplica una atenuación global (BR $= 80/255$) mediante una escala lineal por canal antes de transmitir el frame completo. :contentReference[oaicite:3]index=3

Comunicación UART para telemetría: El sistema inicializa la UART (9600 bps, 8N1) y reporta eventos (p. ej., cambio de color) con mensajes formateados, lo que facilita la verificación y documentación experimental desde consola y/o captura en PC.

Lógica de movimiento en retícula 8×8: El índice i se actualiza con reglas de borde para impedir *wrap-around* horizontal (columnas 0 y 7) y vertical (filas 0 y 7). Se emplea un *cooldown* de pasos para dar una cadencia estable al desplazamiento con el *stick* sostenido y un retardo de refresco de $\sim$10 ms por iteración, asegurando una respuesta perceptualmente fluida del cursor. Una vez actualizado i, se limpia el *framebuffer* y se enciende únicamente el LED de índice i con el color actual antes de transmitir el arreglo completo por PB0. :contentReference[oaicite:5]index=5

En síntesis, este ejercicio demuestra la integración entre adquisición analógica (ADC con umbrales y zona muerta), manejo de entradas digitales (botón con *pull-up* y detección de flanco), generación de color pseudoaleatorio, y transmisión temporalmente crítica del protocolo WS2812B por *bit-banging*, todo sobre el **ATmega328P**.

II-E. Cerradura RFID

El sistema de cerradura electrónica implementado permite la identificación y validación de tarjetas mediante la tecnología **RFID** (*Radio Frequency Identification*), utilizando el módulo **MFRC522** como lector principal. El objetivo del experimento es diseñar un sistema de acceso seguro que reconozca credenciales válidas, gestione nuevas altas o bajas y brinde retroalimentación visual mediante una pantalla LCD I2C, LEDs indicadores y comunicación UART.

Módulo RC522 y principio de funcionamiento: El módulo MFRC522 opera en la banda de 13.56 MHz y permite la lectura y escritura de etiquetas o tarjetas compatibles con el estándar **ISO/IEC 14443A** (MIFARE). Su funcionamiento se basa en la inducción electromagnética: el lector genera un campo alterno que alimenta la antena de la tarjeta pasiva, permitiendo el intercambio de información. El microcontrolador se comunica con el RC522 a través del bus **SPI** (pines MOSI, MISO, SCK y SS), enviando comandos y recibiendo respuestas estructuradas en registros internos del chip.

El código implementa una rutina de inicialización (`RC522_Init()`) que configura los registros de temporización y tasa de bits, y una función de detección (`RC522_Request()`) seguida de la lectura del UID con `RC522_AntiColl()`. El UID capturado se compara con los almacenados en memoria EEPROM para determinar si la tarjeta es válida. En caso positivo, se activa la secuencia de apertura, y en caso negativo, el sistema emite una señal de error visual y sonora.

Comunicación SPI: El bus **SPI** (Serial Peripheral Interface) fue configurado en modo maestro (`MSTR=1`) y frecuencia de reloj Fosc/8, lo que proporciona una velocidad de transferencia de 2 MHz para una operación confiable con el módulo RC522. La comunicación es síncrona y de tipo *full duplex*, utilizando los pines `PB3 (MOSI)`, `PB4 (MISO)`, `PB5 (SCK)` y `PB2 (SS)` del ATmega328P.

Almacenamiento en EEPROM: Para conservar las tarjetas autorizadas incluso tras el apagado, se emplea la memoria EEPROM interna del microcontrolador. Cada UID leído se guarda en posiciones consecutivas, permitiendo almacenar múltiples usuarios. El código utiliza las funciones de lectura y escritura de EEPROM estándar de AVR (`eeprom_read_byte`, `eeprom_write_byte`) y gestiona el espacio mediante un índice de direcciones. Se implementan comandos UART para listar, eliminar o limpiar las tarjetas registradas, lo que facilita la administración del sistema sin reprogramar el microcontrolador.

Interfaz LCD I2C: El sistema incorpora una pantalla LCD de 16×2 caracteres con interfaz **I2C** basada en el expansor **PCF8574**, la cual permite reducir el número de pines utilizados. Esta interfaz se encarga de mostrar los estados del sistema: detección de tarjeta, modo de programación, mensajes de autorización o error, y menús de administración. Los datos se envían en formato de 4 bits con temporización controlada, y la dirección esclava del módulo corresponde a `0x27`.

Retroalimentación visual y sonora: El estado de la cerradura se indica mediante dos LEDs y un buzzer piezoeléctrico. El LED verde confirma una tarjeta válida y activa brevemente el buzzer, mientras que el LED rojo se enciende ante un intento no autorizado. Esta respuesta multimodal brinda una interfaz intuitiva al usuario y permite verificar el funcionamiento sin necesidad de monitorear el puerto serie.

Comunicación UART y monitoreo externo: El microcontrolador establece una comunicación serial a 9600 bps (8N1) para registro de eventos, depuración y control remoto del sistema. Mediante esta interfaz se envían mensajes informativos como la detección de tarjetas, UID leído, estado del sistema y comandos de usuario. Las funciones `uart_send()`, `uart_receive()` y `uart_print()` facilitan la interacción con una terminal o aplicación de monitoreo externa, lo que resulta útil durante la programación y verificación del funcionamiento.

Arquitectura general del sistema: El flujo principal se estructura en un *autómata de estados finitos (FSM)*, donde cada estado corresponde a una etapa del proceso: `INICIO`, `ESPERA`, `PROGRAMACIÓN` y `VERIFICACIÓN`. Los eventos de entrada (detección de tarjeta, pulsación de botones o comandos UART) disparan las transiciones correspondientes. Este enfoque modular permite una gestión ordenada del control lógico y facilita la expansión futura del sistema, por ejemplo, agregando autenticación por contraseña o múltiples niveles de acceso.

En conjunto, el sistema de cerradura RFID combina varios subsistemas de hardware y software: la identificación inalámbrica mediante el RC522, la comunicación serial SPI y UART, la visualización por I2C, y el almacenamiento persistente en EEPROM, integrados todos bajo el control del microcontrolador ATmega328P.

III. Metodología

III-A. Plotter

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**).

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 $\approx$ 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 ± 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 115200 bps 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, y (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 8×8 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` → pin DIN de la matriz WS2812B (línea de datos)
- `A0` → eje X del joystick (lectura ADC)
- `A1` → eje Y del joystick (lectura ADC)
- `PD2` → 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 ($\mathrm{BR} = 80/255$) para evitar saturación visual.

El joystick se lee mediante el ADC de 10 bits, utilizando AVcc como referencia y prescaler 64 (frecuencia de conversión $\approx$ 250 kHz). Se definieron umbrales de decisión con una *zona muerta* de ±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.

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 bps 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, y (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.

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 (`eeprom_read_byte()` y `eeprom_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 bps 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, almacenamiento persistente y una interfaz de usuario clara e intuitiva.

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

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 se visualizaron en tiempo real mediante una aplicación 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 (±10 cuentas 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) permitió superar el par de arranque bajo carga, evitando bloqueos.

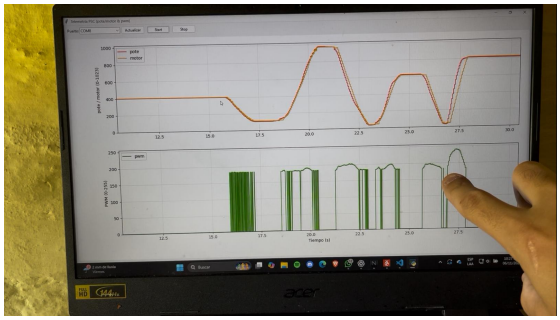

Figura 1. Gráfico obtenido en tiempo real mediante Python, mostrando la relación entre la posición de los potenciómetros y el valor del PWM aplicado al motor.

IV-D. Matriz RGB con Joystick

El sistema respondió correctamente a los desplazamientos del joystick en ambos ejes, mostrando el movimiento del LED activo dentro de la matriz 8×8. La zona muerta definida permitió eliminar falsas detecciones y estabilizar la posición cuando el joystick se encontraba en reposo. El retardo (*cooldown*) aplicado entre lecturas contribuyó a una sensación de desplazamiento controlada 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 se intentó implementar un mapeo matricial continuo, la solución final basada en condiciones discretas resultó más sencilla y confiable.

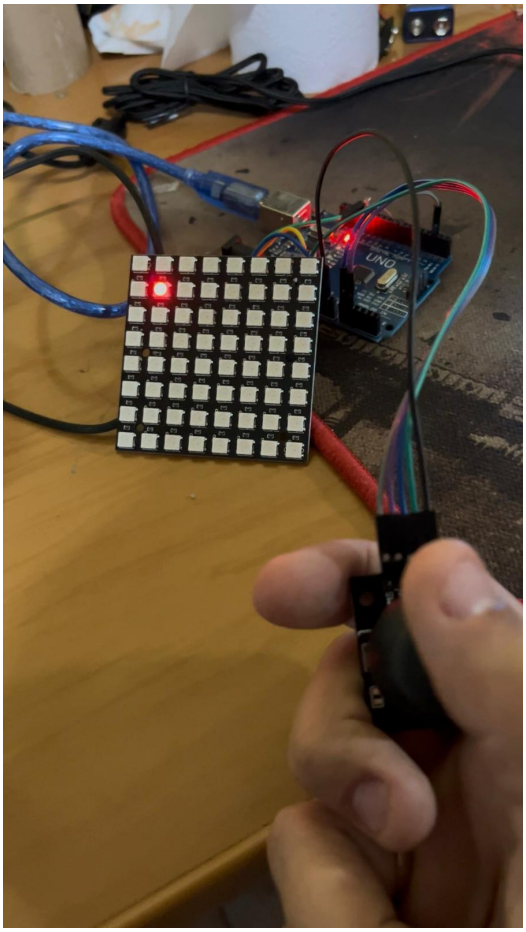

Figura 2. Desplazamiento del LED activo dentro de la matriz RGB controlado mediante el joystick.

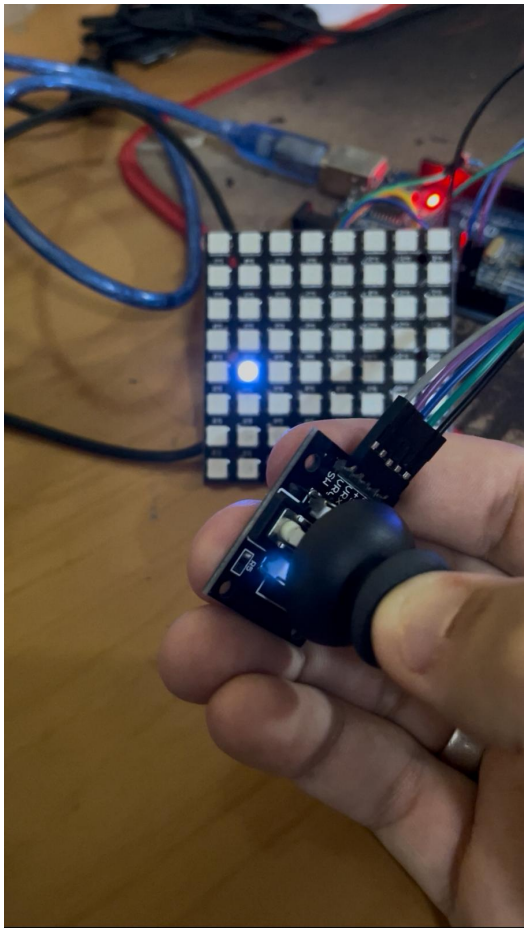

Figura 3. Cambio de color del LED activo al presionar el botón del joystick.

IV-E. Cerradura RFID

El sistema RFID demostró un funcionamiento estable en la lectura y verificación de tarjetas MIFARE. 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, comprobando 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, lo que permitió verificar la comunicación serial básica. La detección en el modo de programación resultó más sensible a los tiempos de lectura, pero se estabilizó ajustando los *delays* entre consultas.

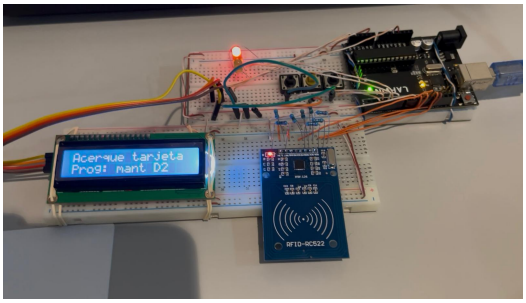

Figura 4. Pantalla inicial del sistema RFID con mensaje de espera para detección de tarjeta.

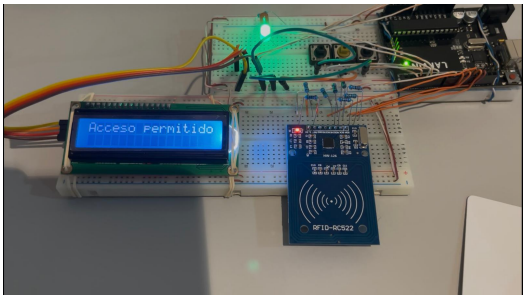

Figura 5. Validación correcta: acceso concedido y encendido del LED verde.

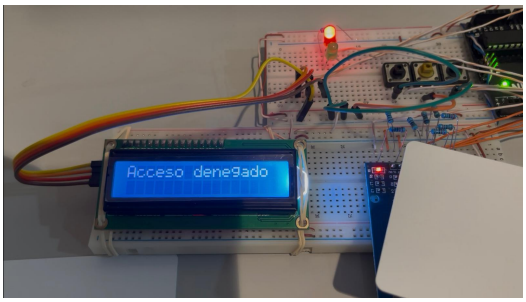

Figura 6. Lectura de tarjeta no registrada: acceso denegado y encendido del LED rojo.

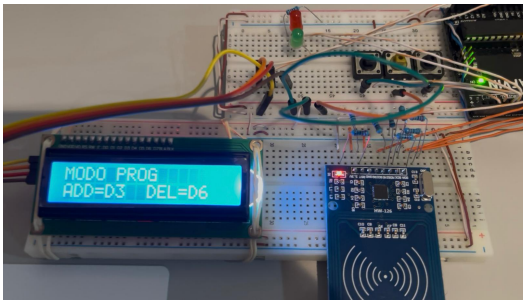

Figura 7. Modo programación: interfaz para alta y baja de tarjetas.

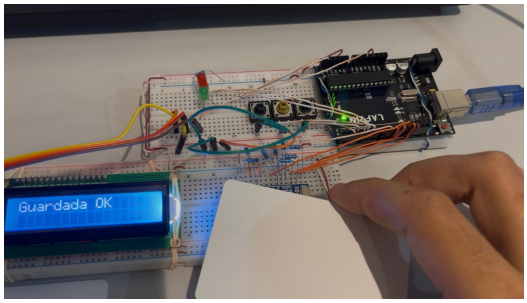

Figura 8. Pantalla correspondiente a la opción de agregar (`ADD`) una nueva tarjeta al sistema.

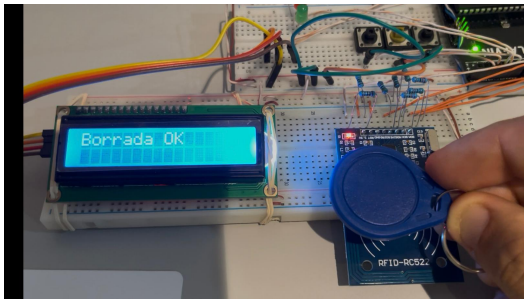

Figura 9. Pantalla correspondiente a la opción de eliminar (`DEL`) una tarjeta previamente registrada.

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 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**, ...

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

En el **Ejercicio C**, se logró implementar un control proporcional simple de velocidad y dirección de un motor DC, 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 ADC, del 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 fue implementado exitosamente, aportó valiosa experiencia en la gestión de coordenadas discretas y calibración de entradas analógicas.

En el **Ejercicio E**, se desarrolló un sistema de 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 cerrados, automatizaciones o aplicaciones de IoT basadas en microcontroladores AVR.

REFERENCIAS

[1] 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

[2] Electronics Tutorials. H-bridge motor control theory. Conceptos de puente H y conmutación de polaridad. [Online]. Available: https://www.electronics-tutorials.ws/blog/h-bridge.html

[3] 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

[4] 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

[5] NXP Semiconductors, *PCF8574/PCF8574A — Remote 8-bit I^2C I/O Expander*, 2023, datasheet oficial del expansor I/O para interfaz I2C de LCD. [Online]. Available: https://www.nxp.com/docs/en/data-sheet/PCF8574_PCF8574A.pdf

[6] SparkFun Electronics. (2017) Thumb joystick hookup guide. Funcionamiento del joystick analógico, ejes, SW y lectura por ADC. [Online]. Available: https://learn.sparkfun.com/tutorials/thumb-joystick-hookup-guide

[7] ——. (2016) Rfid basics. Introducción a RFID, fundamentos y flujo de lectura UID. [Online]. Available: https://learn.sparkfun.com/tutorials/rfid-basics/all

VI. ANEXOS