	Laboratorio de Microcomputadoras

Profesor(a): Dra. Lourdes Angelica Quiñonez Juárez

Asignatura: Laboratorio de Microcomputadoras

Grupo Laboratorio: 5

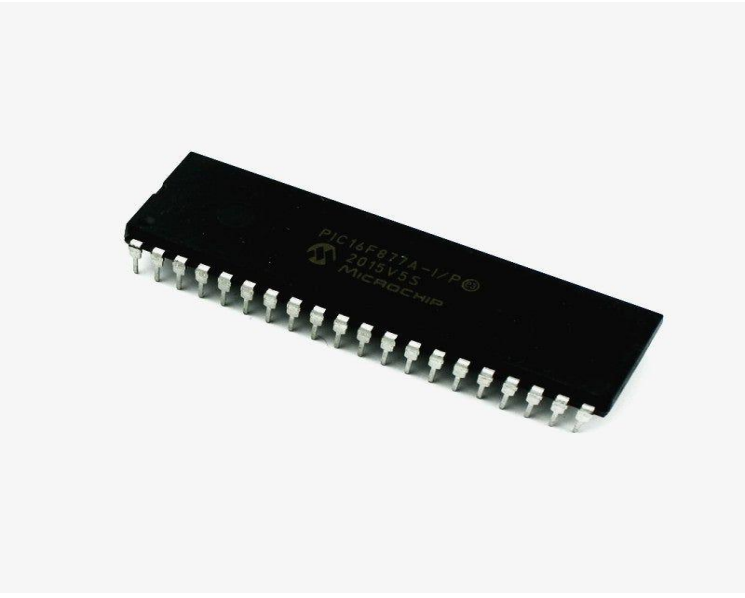
Grupo Teoría: 3

No. de Práctica(s): 4

Integrante(s): Hernández Diaz Sebastián

Semestre: 2025-2

Fecha de entrega: 17 de marzo del 2025



Hernández Diaz Sebastian

Laboratorio de Microcomputadoras

Practica No. 4

Puertos Paralelos III (Control de Motores de CD)

Objetivo. Emplear los puertos paralelos que contiene un microcontrolador, para controlar la operación de dos motores de corriente directa.

Desarrollo. Utilizando el circuito de potencia de motores de corriente directa y el sistema de desarrollo del microcontrolador PIC, realizar los programas solicitados.

1.- De acuerdo a la asignación de la tarjeta del driver de motores realizar un programa, el cual permita controlar el funcionamiento y sentido de giro de cada uno de ellos por separado, a través del puerto paralelo A, el puerto B deberá mandar las señales al driver, como se indica en la tabla 4.1.

Entrada binaria **PuertoA**	Motor		Sentido de giro **Puerto B**
	Izquierdo	Derecho	
000000	OFF	OFF	Paro
000010	OFF	ON	Horario
000100	OFF	ON	Antihorario
001000	ON	OFF	Horario
010000	ON	OFF	Antihorario

Tabla 4.1

2.- Considerando la información y los circuitos del ejercicio anterior, realizar un programa que de acuerdo a una señal de control ingresada por el puerto A, se genere la acción indicada en la tabla 4.2.

DATO Puerto A	ACCION	
	MOTOR M1	MOTOR M2
$00	PARO	PARO
$01	DERECHA	DERECHA
$02	IZQUIERDA	IZQUIERDA
$03	DERECHA	IZQUIERDA
$04	IZQUIERDA	DERECHA

Tabla 4.2

En el caso del laboratorio se decido hacer estas dos actividades en una sola, dando como resultado la siguiente tabla con los valores que debe hacer el motor.

PORTA	RC2 EN_M2	RB3 DIR2_M2	RB2 DIR1_M2	RC1 EN_M1	RB1 DIR2_ M1	RB0 DIR1_ M1	
000000	0	0	0	0	0	0	PASO 1
000001	0	0	0	1	1	0	PASO 2
000010	0	0	0	1	0	1	PASO 3
000100	1	1	0	0	0	0	PASO 4
001000	1	0	1	0	0	0	PASO 5
000011	1	1	0	1	1	0	PASO 6
000110	1	0	1	1	0	1	PASO 7
000111	1	1	0	1	0	1	PASO 8
000101	1	0	1	1	1	0	PASO 9

EN = 1 → Encendido

EN = 0 → Apagado

DIR1	DIR2	
0	0	→ paro
1	1	→ encendido
1	0	→ horario
0	1	→ antihorario

Siendo los pasos los siguientes:

PASO 1: Estará todo detenido.

PASO 2: Motor derecho girara en sentido horario.

PASO 3: Motor derecho girara en sentido antihorario.

PASO 4: Motor izquierdo girara en sentido horario.

PASO 5: Motor izquierdo girara en sentido antihorario.

PASO 6: Los 2 motores giraran en sentido horario.

PASO 7: Los 2 motores giraran en sentido antihorario.

PASO 8: Los 2 motores giraran el izquierdo en sentido horario y el derecho en sentido antihorario.

PASO 9: Los 2 motores giraran el derecho en sentido horario y el izquierdo en sentido antihorario.

El código para esta práctica es el siguiente:

Hernández Diaz Sebastian

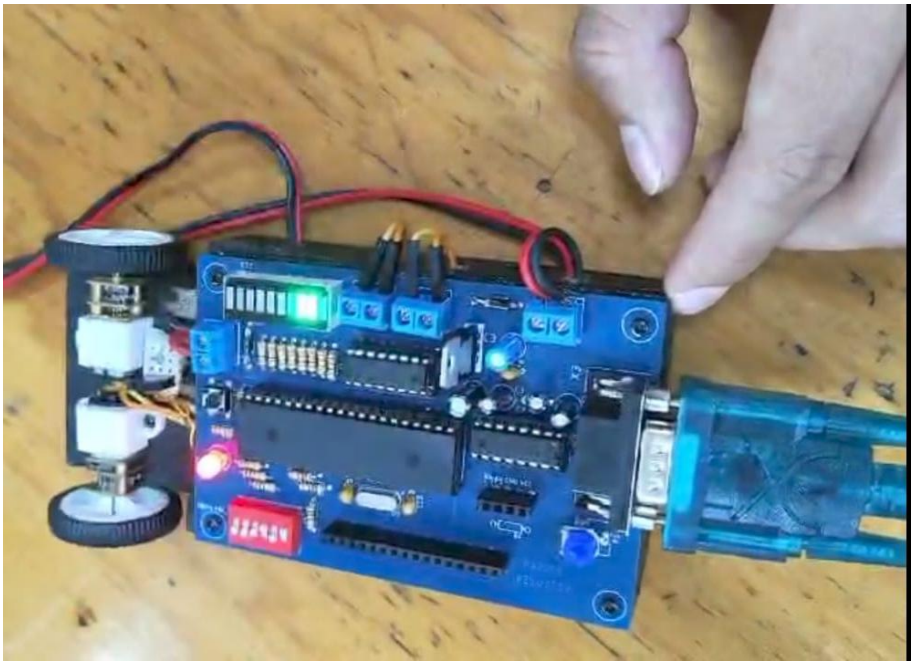
```
processor 16F877  ; Se especifica que el código es para el microcontrolador PIC16F877.
include<p16f877.inc>  ; Se incluye la librería con las definiciones del microcontrolador.

ORG 0
GOTO inicio  ; Salta a la etiqueta "inicio" para comenzar la ejecución.

ORG 5
inicio:
    bsf STATUS, RP0  ; Se activa el bit RP0 para cambiar al Banco 1 de memoria.
    BCF STATUS, RP1  ; Se limpia el bit RP1, asegurando que estamos en el Banco 1.
    MOVLW H'07'  ; Se carga el valor 07H en el registro W.
    MOVWF ADCON1  ; Se configura ADCON1 para deshabilitar los convertidores analógicos y usar los pines como digitales.
    MOVLW H'ff'  ; Se carga FFH en W.
    MOVWF TRISA  ; Se configura el Puerto A como entrada (1 en cada bit).
    CLRF TRISB  ; Se configura el Puerto B como salida (0 en cada bit).
    CLRF TRISC  ; Se configura el Puerto C como salida (0 en cada bit).
    BCF STATUS, RP0  ; Se vuelve al Banco 0 para acceder a los puertos.

switch:
    MOVF PORTA, 0  ; Se lee el valor del Puerto A (entrada de control).
    XORLW H'00'  ; Se compara con 00H (sin cambios en los motores).
    BTFSC STATUS, Z  ; Si es igual, se ejecuta "paso1".
    call paso1

    MOVF PORTA, 0
    XORLW h'01'  ; Se compara con 01H (girar motor derecho en sentido horario).
    BTFSC STATUS, Z
    call paso2

    MOVF PORTA, 0
    XORLW H'02'  ; Se compara con 02H (girar motor derecho en sentido antihorario).
    BTFSC STATUS, Z
    CALL paso3

    MOVF PORTA, 0
    XORLW H'04'  ; Se compara con 04H (girar motor izquierdo en sentido horario).
    BTFSC STATUS, Z
    CALL paso4

    MOVF PORTA, 0
    XORLW H'08'  ; Se compara con 08H (girar motor izquierdo en sentido antihorario).
    BTFSC STATUS, Z
    CALL paso5

    MOVF PORTA, 0
    XORLW H'03'  ; Se compara con 03H (ambos motores en sentido horario).
    BTFSC STATUS, Z
    CALL paso6
```

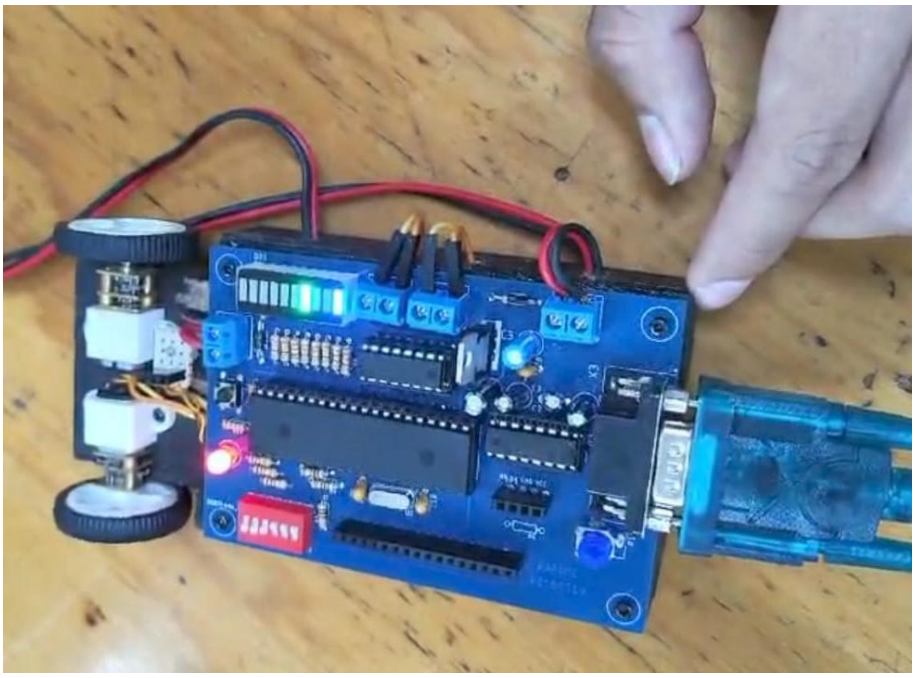
```
'
    MOVF PORTA, 0
    XORLW H'06'  ; Se compara con 06H (ambos motores en sentido antihorario).
    BTFSC STATUS, Z
    call paso7

    MOVF PORTA, 0
    XORLW H'07'  ; Se compara con 07H (motor izquierdo horario, motor derecho antihorario).
    BTFSC STATUS, Z
    call paso8

    MOVF PORTA, 0
    XORLW H'05'  ; Se compara con 05H (motor derecho horario, motor izquierdo antihorario).
    BTFSC STATUS, Z
    call paso9

    GOTO switch  ; Vuelve a leer el Puerto A para detectar cambios en la entrada.

paso1:
    MOVLW h'00'
    MOVWF PORTC  ; Se apagan ambos motores.
    MOVLW h'00'
    MOVWF PORTB  ; Se apagan ambos motores.
    RETURN

paso2:
    MOVLW h'02'
    MOVWF PORTC  ; Motor derecho gira en sentido horario.
    MOVLW h'02'
    MOVWF PORTB
    RETURN

paso3:
    MOVLW h'02'
    MOVWF PORTC  ; Motor derecho gira en sentido antihorario.
    MOVLW h'01'
    MOVWF PORTB
    RETURN

paso4:
    MOVLW h'04'
    MOVWF PORTC  ; Motor izquierdo gira en sentido horario.
    MOVLW h'08'
    MOVWF PORTB
    RETURN
```

```
paso5:
    MOVLW h'04'
    MOVWF PORTC  ; Motor izquierdo gira en sentido antihorario.
    MOVLW h'04'
    MOVWF PORTB
    RETURN

paso6:
    MOVLW h'06'
    MOVWF PORTC  ; Ambos motores giran en sentido horario.
    MOVLW h'0A'
    MOVWF PORTB
    RETURN

paso7:
    MOVLW h'06'
    MOVWF PORTC  ; Ambos motores giran en sentido antihorario.
    MOVLW h'05'
    MOVWF PORTB
    RETURN

paso8:
    MOVLW h'06'
    MOVWF PORTC  ; Motor izquierdo gira en sentido horario, motor derecho antihorario.
    MOVLW h'09'
    MOVWF PORTB
    RETURN

paso9:
    MOVLW h'06'
    MOVWF PORTC  ; Motor derecho gira en sentido horario, motor izquierdo antihorario.
    MOVLW h'06'
    MOVWF PORTB
    RETURN

END
```

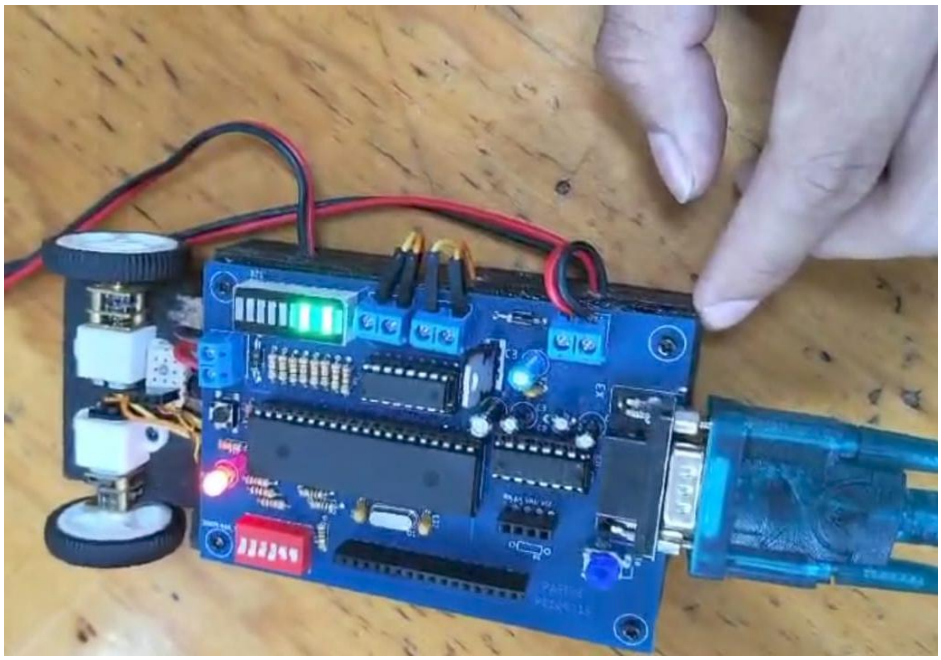
El código controla dos motores utilizando un PIC16F877, configurando PORTA como entrada para leer switches y PORTB y PORTC como salidas para activar los motores en diferentes direcciones. Dependiendo del valor leído en PORTA, se ejecuta una función que define el movimiento de los motores, permitiendo que giren en sentido horario, antihorario, de forma independiente o sincronizada. En total, hay nueve pasos: detener ambos motores, girar individualmente cada motor en ambas direcciones, mover ambos motores en el mismo sentido y hacer que giren en direcciones opuestas. Este sistema permite un control preciso del movimiento, útil para aplicaciones en robótica y vehículos autónomos.

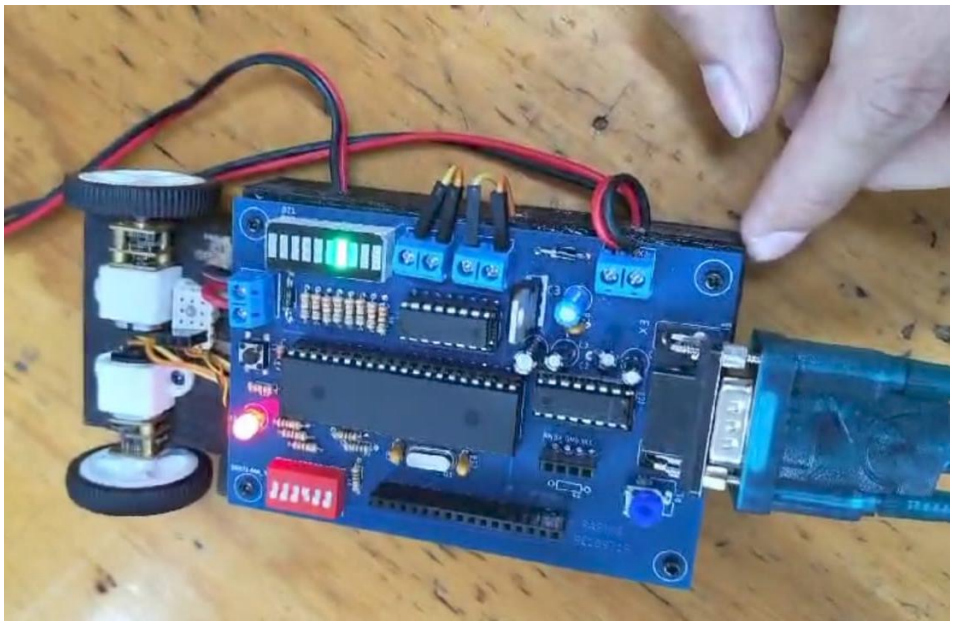
A continuación, se presentan unas imágenes de la ejecución de este código:

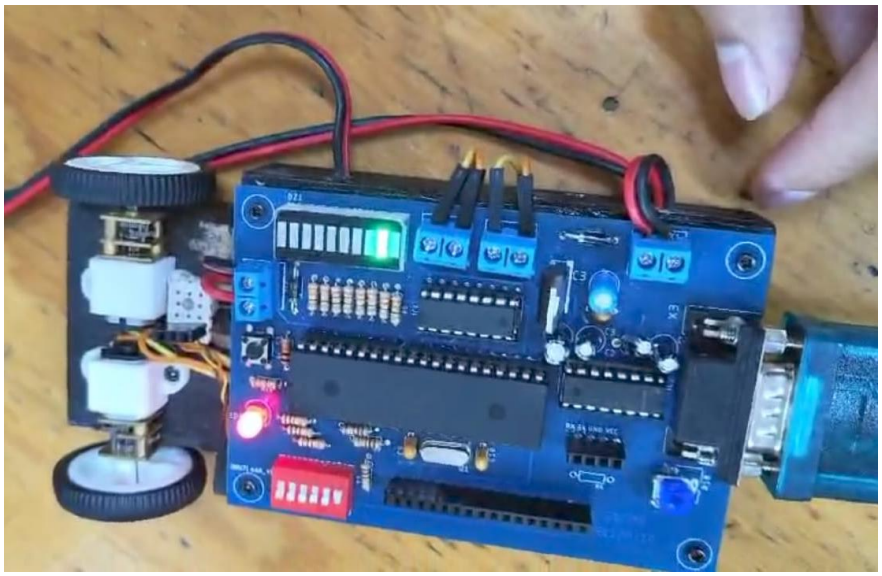


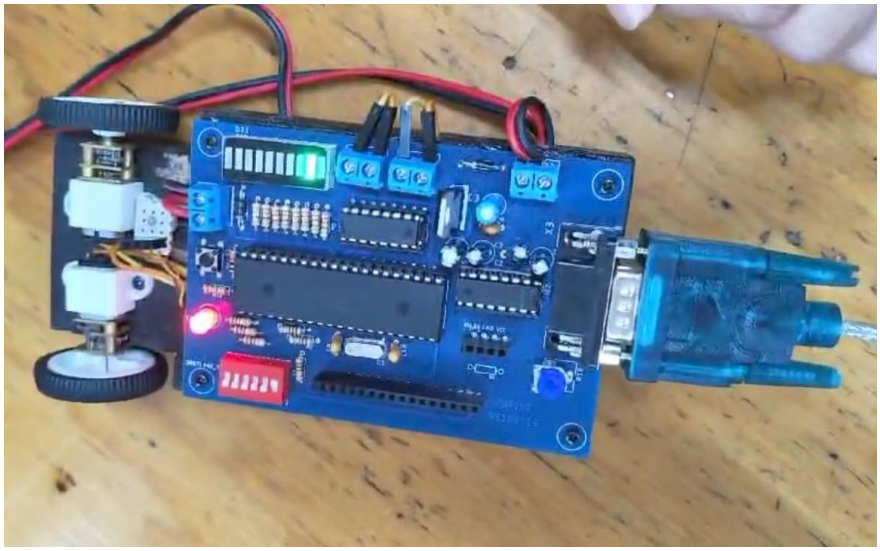


Hernández Diaz Sebastian









A pesar de que en las imágenes no se puede notar como rotaban las ruedas todo funciono de manera correcta en el laboratorio por lo que la practica fue llevada de manera correcta y se obtuvo el resultado esperado.

Hernández Diaz Sebastian

Conclusiones

Esta práctica permitió afianzar conocimientos sobre la programación y control de motores utilizando el microcontrolador PIC16F877. Se trabajó con la configuración de los puertos como entradas y salidas, donde PORTA se empleó como entrada para detectar señales de control, mientras que PORTB y PORTC se usaron como salidas para accionar los motores en distintas direcciones. Se utilizaron instrucciones como MOVLW y MOVWF para la carga y almacenamiento de valores en registros específicos, así como XORLW y BTFSC para la comparación de valores y la toma de decisiones en función del estado de PORTA. Gracias a esto, se logró controlar el giro de los motores en sentido horario y antihorario, tanto de manera individual como sincronizada.

Otro aspecto relevante de la práctica fue la estructuración del código en funciones separadas para cada operación de los motores. Esto facilitó la comprensión del programa y su mantenimiento, permitiendo modularizar el código en pasos bien definidos. Se destacó el uso de la lógica condicional para determinar el comportamiento de los motores según las señales de entrada, lo que resulta esencial en aplicaciones de control automático. Además, la programación basada en eventos, en donde la activación de un bit específico en PORTA ejecuta una función determinada, representa un principio fundamental en sistemas embebidos y automatización industrial.

Entre las prácticas relacionadas con este ejercicio, se pueden mencionar:

- **Control de velocidad de motores DC con PWM**: Implementando modulación por ancho de pulso para variar la velocidad de giro de los motores de forma eficiente.
- **Implementación de un control de dirección con sensores**: Para permitir que los motores respondan automáticamente a la detección de obstáculos o líneas en el suelo.
- **Automatización de un vehículo autónomo**: Usando principios similares de control de motores para seguir trayectorias predefinidas o reaccionar a entradas externas como sensores infrarrojos o ultrasonidos.
- **Manejo de motores paso a paso con PIC16F877**: Ampliando el conocimiento sobre el control de motores para aplicaciones de posicionamiento preciso, como en brazos robóticos o sistemas CNC.
- **Comunicación entre microcontroladores y sistemas de control remoto**: Aplicando comunicación serial o inalámbrica para controlar los motores desde un dispositivo externo, como una PC o un control remoto.

Bibliografía

Peatman, J. B. (1998). *Design with PIC Microcontrollers*. Pearson Education.

Predko, M. (2008). *Programming and Customizing the PIC Microcontroller*. McGraw-Hill Education.

Microchip Technology. (2024). *PIC16F877 Datasheet*. Recuperado de https://www.microchip.com

Gooligum Electronics. (2023). "PIC Tutorial – Mid-Range Microcontrollers". Recuperado de http://www.gooligum.com.au/tutorials