Facultad de ingeniería

Materia: Laboratorio de Microcomputadoras

REPORTE DE LA PRÁCTICA 7

Título: Puerto Serie SCI (Asíncrono)

Integrantes:

- Martínez Pérez Brian Erik - 319049792
- Nuñez Rodas Abraham Enrique - 114003546
- Vicenteño Maldonado Jennifer Michel - 317207251

Profesor: Moises Melendez Reyes

Grupo: 1

Fecha de Entrega: 14 de mayo de 2025

Semestre: 2025-2

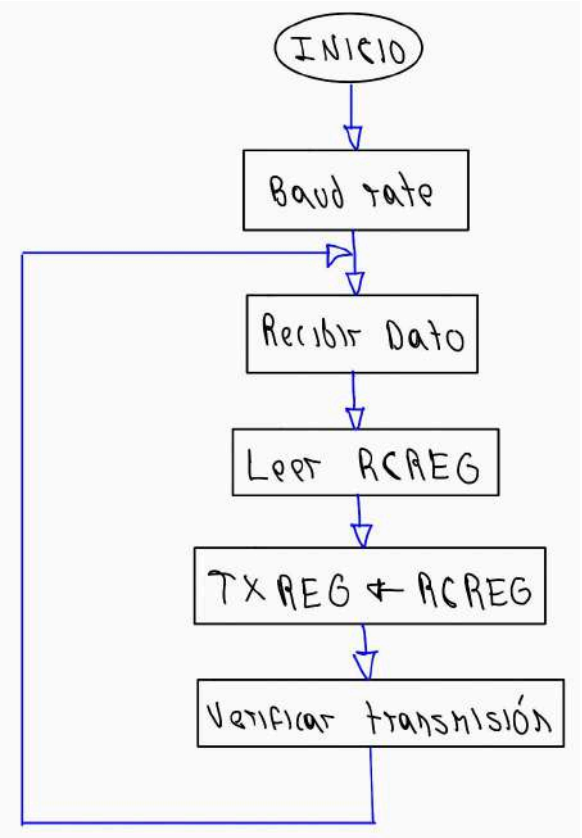
Objetivo: Familiarizar al alumno en el uso de una Interfaz de Comunicación Serie Asíncrona de un microcontrolador

Ejercicios

Ejercicio 1 y 2 : Escribir, comentar y ensamblar el siguiente código

```
processor 16f877
include<p16f877.inc>
ORG 0
GOTO inicio
ORG 5
BSF STATUS,RP0
BCF STATUS,RP1
BSF TXSTA,BRGH
MOVLW D'129'
MOVWF SPBRG
BCF TXSTA,SYNC
BSF TXSTA,TXEN

BCF STATUS,RP0
BSF RCSTA,SPEN
BSF RCSTA,CREN
RECIBE:
TRASMITE:
BTFSS PIR1,RCIF
GOTO RECIBE
MOVF RCREG,W
MOVWF TXREG
BSF STATUS,RP0
BTFSS TXSTA,TRMT
GOTO TRASMITE
BCF STATUS,RP0
GOTO RECIBE
END
```

Diagrama de flujo:



Código:

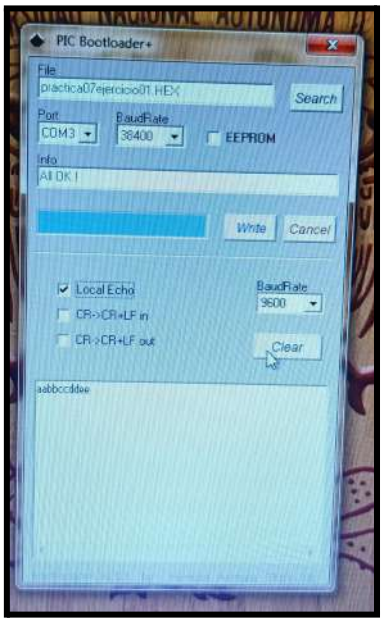
```
        processor 16f877
        include         <p16f877.inc>

        ORG 0
        GOTO INICIO
        ORG 5
INICIO:
        ;Cambio al banco 01
        BSF STATUS,RP0
        BCF STATUS,RP1

        BSF TXSTA,BRGH          ;SELECCIÓN DE ALTA VELOCIDAD DE
BAUDIOS
        MOVLW D'129'
        MOVWF SPBRG                     ;Asignar 9600 BAUDS

        BCF TXSTA,SYNC          ;Modo de comunicación=0. Asíncrona
        BSF TXSTA,TXEN          ;Activación de transmisión

        BCF STATUS,RP0  ;Cambio al banco 0

        BSF RCSTA,SPEN          ;Habilita el puerto Serie
        BSF RCSTA,CREN          ;Activa la recepción continua en modo de
comunicación asíncrona

RECIBE:
        BTFSS PIR1,RCIF         ;Revisa si la recepción ha sido completada
        GOTO RECIBE                     ;Si aun está en recepción de proceso repite

        MOVF RCREG,W            ;En la recepción completada,
                                        ;se puede obtener la información por medio
de RCREG
                                        ;W=RCREG
        MOVWF TXREG                     ;TXREG=W
                                        ;Registro para mandar a transmitir

        BSF STATUS,RP0          ;Cambio a banco 1

TRANSMITE:
        BTFSS TXSTA,TRMT        ;Revisa si ha transmitido el dato
        GOTO TRANSMITE          ;Si no repite el proceso
        BCF STATUS,RP0          ;Cambio al banco 0
        GOTO RECIBE                     ;Repite la recepción

        END
```

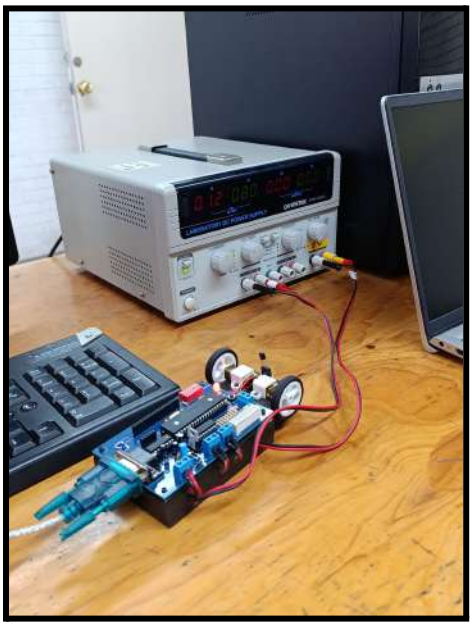
Análisis:

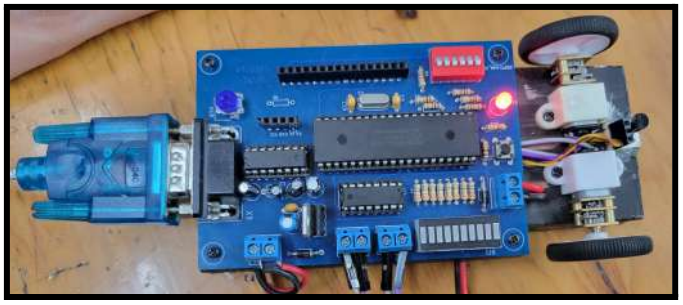
El código implementa una función de eco serial usando el módulo USART del PIC16F877. Configura la comunicación asíncrona a 9600 baudios, habilita la transmisión y recepción continua, y entra en un bucle donde espera un carácter recibido en RCREG, lo transfiere al registro TXREG y lo retransmite. Antes de enviar cada dato, verifica que la transmisión anterior haya finalizado consultando el bit TRMT. La solución es eficiente y cumple su propósito, demostrando correctamente el uso básico del puerto serial en modo asíncrono.

Funcionamiento de la solución:
Este programa utiliza el Módulo USART, realiza las confirmaciones para recibir un carácter a través del pin RX y retransmitirlo a través del TX.
En general este programa funciona como un eco, dónde a través de la terminal le proporcionamls un carácter que recibirá el controlador y lo rebotara.

Imágenes:



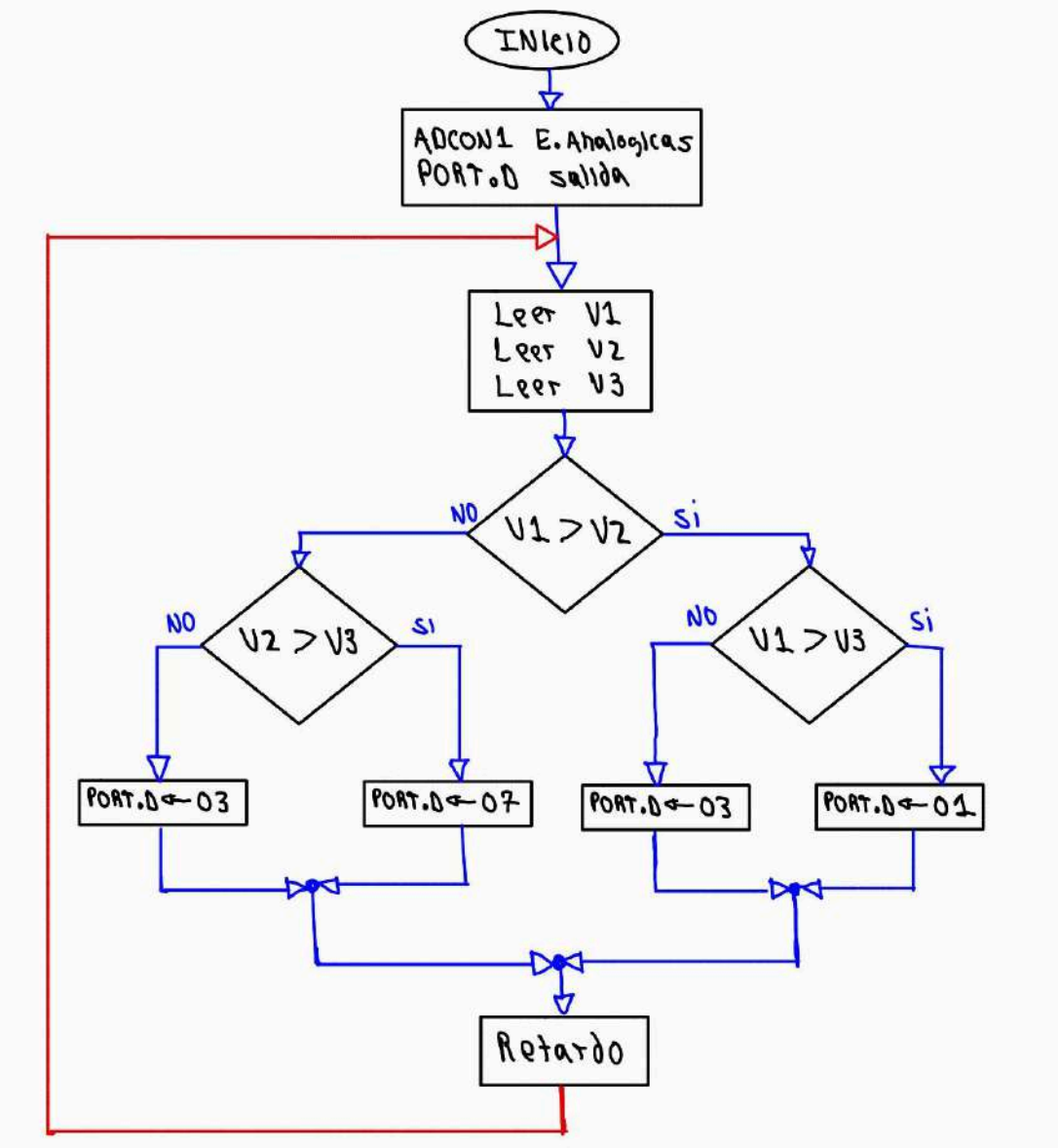




Ejercicio 3: Realizar un programa que despliegue la siguiente cadena en una terminal.

HOLA UNAM

Diagrama de flujo:



Código:

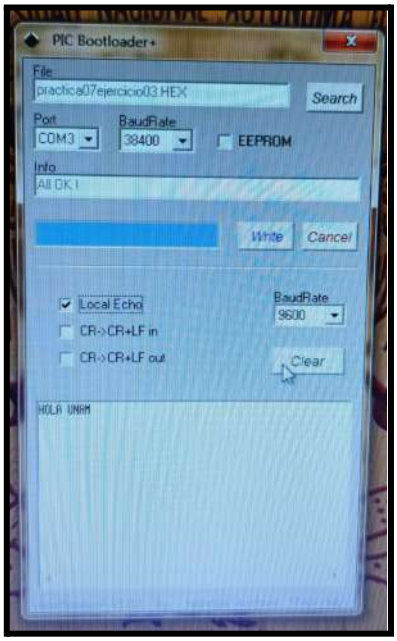
```
        processor 16f877
        include         <p16f877.inc>
VALOR EQU H'20'
SAL     EQU H'21'
        ORG 0
        GOTO INICIO
        ORG 5
INICIO:
        ;Cambio al banco 01
        BSF STATUS,RP0
        BCF STATUS,RP1
        BSF TXSTA,BRGH
;SELECCIÓN DE ALTA VELOCIDAD
DE BAUDIOS
        MOVLW D'129'
        MOVWF SPBRG
        ;Asignar 9600 BAUDS

        BCF TXSTA,SYNC
;Modo de comunicación=0. Asíncrona
        BSF TXSTA,TXEN
;Activación de transmisión
```

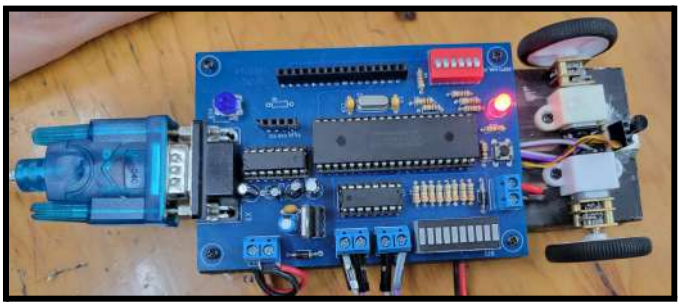
```
        BCF STATUS,RP0  ;Cambio al
banco 0

        BSF RCSTA,SPEN
;Habilita el puerto Serie
        BSF RCSTA,CREN
;Activa la recepción continua en modo
de comunicación asíncrona



INFO

        ;Transmisión de los caracteres
de 'HOLA UNAM'

        ;El valor en ASCII se envia a
partir del registo TXREG
        MOVLW   A'H'
        MOVWF   TXREG
        CALL  CONF_TRANSM
                ;Subtrutina del camvio de
banco 01
        MOVLW   A'O'
        MOVWF   TXREG
        CALL  CONF_TRANSM
        MOVLW   A'L'
        MOVWF   TXREG
        CALL  CONF_TRANSM
        MOVLW   A'A'
        MOVWF   TXREG
        CALL  CONF_TRANSM
        MOVLW   A' '
        MOVWF   TXREG
        CALL  CONF_TRANSM
        MOVLW   A'U'
        MOVWF   TXREG
        CALL  CONF_TRANSM
        MOVLW   A'N'
        MOVWF   TXREG
        CALL  CONF_TRANSM
        MOVLW   A'A'
        MOVWF   TXREG
        CALL  CONF_TRANSM
        MOVLW   A'M'
        MOVWF   TXREG
        CALL  CONF_TRANSM
LOOP
        GOTOLOOP

CONF_TRANSM
        BSF             STATUS,RP0
                ;Cambio al banco 1

TRANSMITIR
        BTFSS           TXSTA,TRMT
                ;Revisión de transmisión
exitosa
        GOTOTRANSMITIR
        BCF             STATUS,RP0
                ;Cambio al banco 0
        RETURN
                ;Regreso al último CALL

        END
```

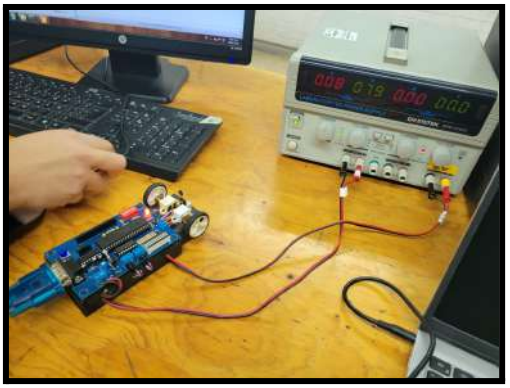
Análisis:

El programa configura el módulo USART del PIC16F877 para operar en modo asíncrono a una velocidad de 9600 baudios. Tras la inicialización, se procede a transmitir carácter por carácter la cadena "HOLA UNAM" utilizando el registro TXREG. Cada carácter se carga con MOVLW y se envía de manera secuencial, asegurando que la transmisión anterior haya finalizado mediante la verificación del bit TRMT en TXSTA. Para esto, se utiliza una subrutina que gestiona el cambio de banco y espera activa hasta que la línea de transmisión esté libre.

Imágenes





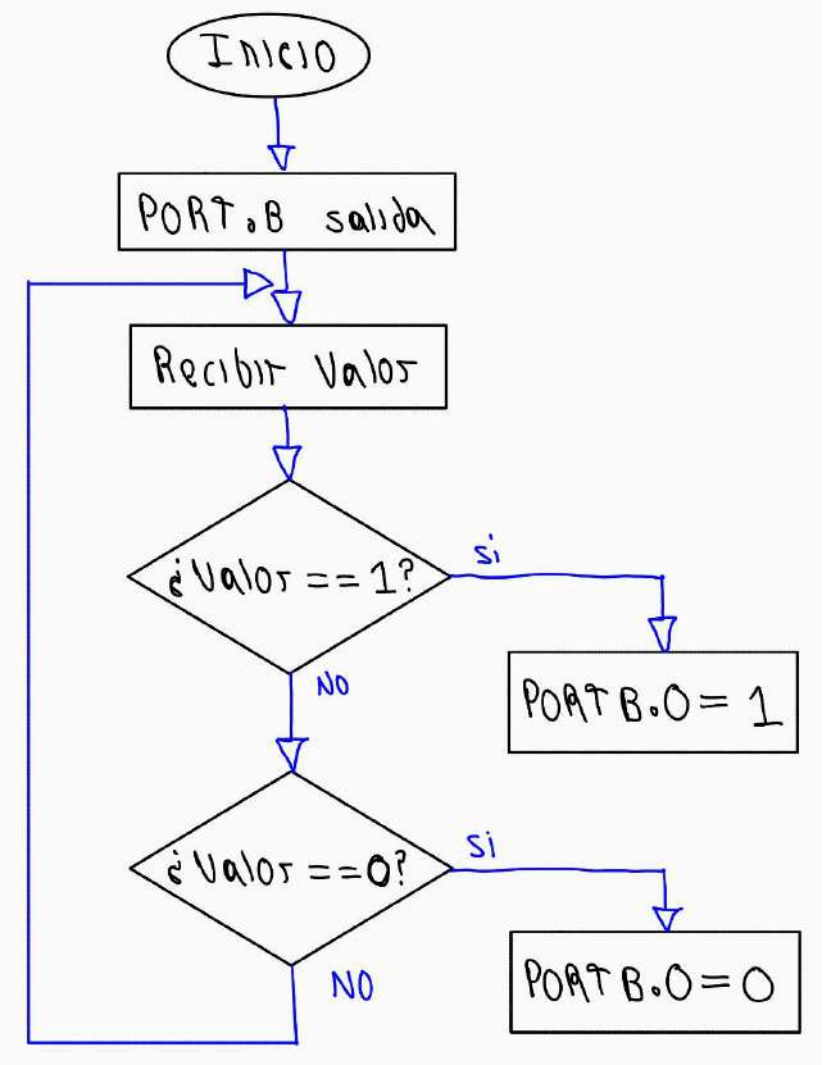


Ejercicio 4: Realizar un programa que ejecute el control indicado; el dato proviene a través del puerto serie:

DATO Puerto Serie	**ACCION** Terminal 0 del puerto B (PB0)
'0'	0
'1'	1

Tabla 7.1 Control para activar y desactivar una señal

Diagrama de flujo:



Código:

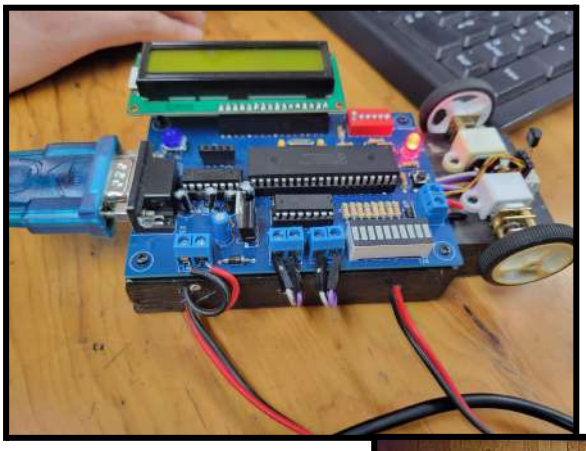
```
processor 16f877
include <p16f877.inc>
VALOR EQU H'20'
ORG 0
GOTO INICIO
ORG 5

INICIO:
BSF STATUS,RP0
BCF STATUS,RP1
MOVLW h'0'
MOVWF TRISB
CLRF PORTB
BSF TXSTA,BRGH
MOVLW D'129'
MOVWF SPBRG
BCF TXSTA,SYNC
BSF TXSTA,TXEN
BCF STATUS,RP0
BSF RCSTA,SPEN
BSF RCSTA,CREN
MOVLW 0X00
MOVWF PORTB

RECIBE:
BTFSS PIR1,RCIF
GOTO RECIBE
MOVF RCREG,W
MOVWF VALOR
MOVWF TXREG
BSF STATUS,RP0

TRASMITE:
BTFSS TXSTA,TRMT
```

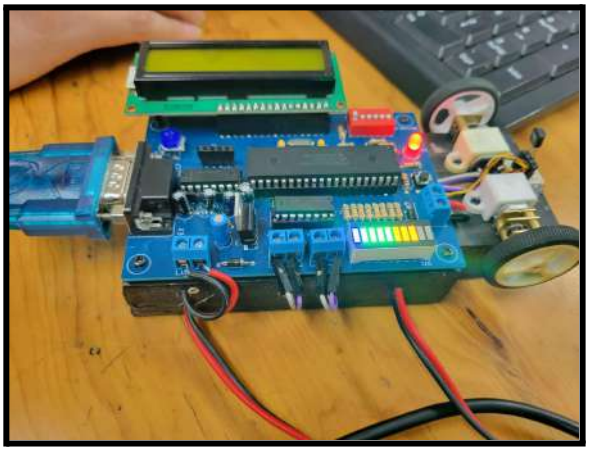
```
GOTO TRASMITE
BCF STATUS,RP0

SALIDA:
MOVLW A'1'
XORWF VALOR,W
BTFSC STATUS,Z
GOTO SAL1
MOVLW A'0'
XORWF VALOR,W
BTFSC STATUS,Z
GOTO SAL0
GOTO RECIBE

SAL1:
BSF PORTB,0
GOTO RECIBE

SAL0:
BCF PORTB,0
GOTO RECIBE

END
```

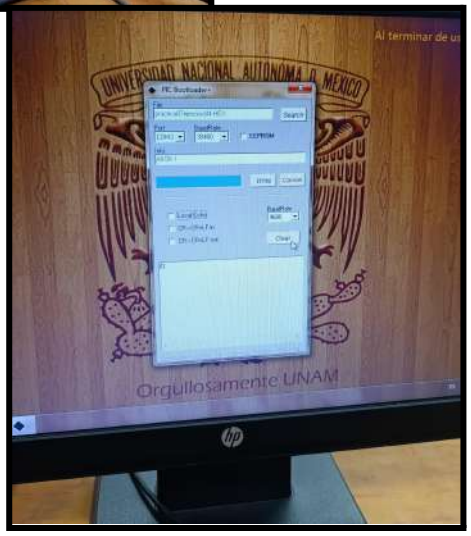
Análisis:

Implementamos un sistema de control digital mediante recepción de datos por comunicación serial asíncrona utilizando el módulo USART del PIC16F877. Una vez configurado el USART para operar a 9600 baudios, el microcontrolador queda en espera de un dato entrante por el pin RX. Cuando se recibe un carácter, este se almacena en el registro `VALOR` y se retransmite por `TXREG`. Posteriormente, se compara con los valores ASCII de los caracteres `'1'` y `'0'`. Si el valor recibido es `'1'`, se activa el bit 0 del puerto B (`PORTB,0`); si es `'0'`, el bit se apaga. Esto permite al usuario controlar directamente una salida digital desde una terminal serial enviando simplemente un carácter.

Imágenes:





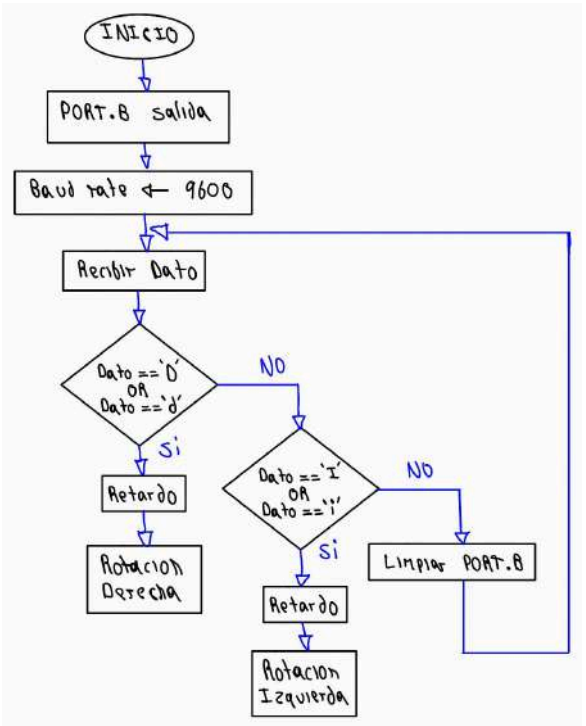


Ejercicio 5: Realizar un programa que ejecute el control indicado; la secuencia será reconocida cada que sea recibido el comando, usar retardos de ½ segundo entre cada estado generado:

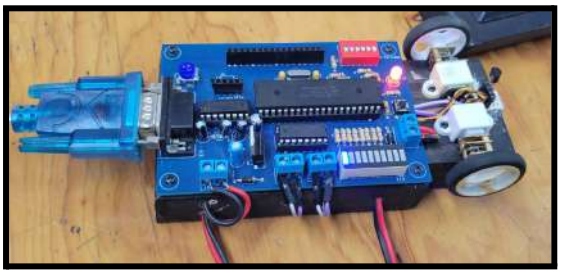
DATO Puerto Serie	**ACCION** Salida Puerto B
'D' ó 'd'	10000000 01000000 00100000 00010000 00001000 00000100 00000010 00000001
'I' ó 'i'	00000001 00000010 00000100 00001000 00010000 00100000 01000000 10000000

Tabla 7.2 Secuencia de control

Diagrama de flujo:



imágenes:



Código:

```
processor 16f877
include <p16f877.inc>

VALOR EQU H'20'
VALOR1 EQU H'51'
VALOR2 EQU H'52'
VALOR3 EQU H'53'
CTE1 equ 70h
CTE2 equ 70h
CTE3 equ 70h

ORG 0
GOTO INICIO
ORG 5

INICIO:
BSF STATUS,RP0
BCF STATUS,RP1
MOVLW h'0'
MOVWF TRISB
CLRF PORTB
BSF TXSTA,BRGH
MOVLW D'129'
MOVWF SPBRG
BCF TXSTA,SYNC
BSF TXSTA,TXEN
BCF STATUS,RP0
BSF RCSTA,SPEN
BSF RCSTA,CREN

RECIBE:
BTFSS PIR1,RCIF
GOTO RECIBE
MOVF RCREG,W
MOVWF VALOR
MOVWF TXREG
BSF STATUS,RP0

TRASMITE:
BTFSS TXSTA,TRMT
GOTO TRASMITE
BCF STATUS,RP0

SALIDA:
```

```
MOVLW A'D'
XORWF VALOR,W
BTFSC STATUS,Z
GOTO SAL_D
MOVLW A'd'
XORWF VALOR,W
BTFSC STATUS,Z
GOTO SAL_D
MOVLW A'I'
XORWF VALOR,W
BTFSC STATUS,Z
GOTO SAL_I
MOVLW A'i'
XORWF VALOR,W
BTFSC STATUS,Z
GOTO SAL_I

NONE:
CLRF PORTB
GOTO RECIBE

SAL_D:
MOVLW 0X80
MOVWF PORTB
BCF STATUS,C
CALL RETARDO

LOOP_D:
RRF PORTB
CALL RETARDO
BTFSS STATUS,C
GOTO LOOP_D
GOTO RECIBE

SAL_I:
MOVLW 0X01
MOVWF PORTB
BCF STATUS,C
CALL RETARDO

LOOP_L:
RLF PORTB
CALL RETARDO
BTFSS STATUS,C
GOTO LOOP_L
GOTO RECIBE

RETARDO:
MOVLW CTE1
MOVWF VALOR1

TRES:
MOVLW CTE2
MOVWF VALOR2

DOS:
MOVLW CTE3
MOVWF VALOR3

UNO:
DECFSZ VALOR3
GOTO UNO
DECFSZ VALOR2
GOTO DOS
DECFSZ VALOR1
GOTO TRES
RETURN

END
```

Análisis: El programa tiene como objetivo recibir datos vía UART a 9600 baudios y activar secuencias de desplazamiento en el puerto B según el carácter recibido. Se puede comprobar que la solución funciona correctamente, ya que el sistema recibe, compara, retransmite y ejecuta acciones según los comandos 'D', 'd', 'I' o 'i'. El flujo de datos inicia con la recepción en RCREG, luego se almacena en VALOR, se compara con valores ASCII y, si coincide, se activa una secuencia de desplazamientos a derecha o izquierda en PORTB, con retardos programados mediante ciclos anidados. Se usaron modos de direccionamiento inmediato (con MOVLW) y directo (con MOVWF, MOVF). Los periféricos utilizados son el módulo

USART para comunicación serial y el puerto B como salida digital. En resumen, los objetivos fueron alcanzados con una implementación lógica, funcional y eficiente del control serial y visual del microcontrolador.

Ejercicio 6: Descargar la aplicación practica7.apk e instalar en su dispositivo móvil (Android), realizar un programa para el microcontrolador, de manera que reciba el comando a través del puerto serie, con conexión inalámbrica (bluetooth), par que genere el control indicado en la tabla 7.4; usar la asignación mostrada en la tabla 7.3.

Notas importantes:

a. El dato que recibe es el código ASCCI del carácter transmitido.

b. Para vincularse con el dispositivo Bluetooth deberá comprobar su identificador.

c. Considerar la ubicación de las señales de control y los valores encontrados en la práctica 5.

MOTOR2		
PC2	**PB3**	**PB2**
ENABLE_M2	DIR1_M2	DIR2_M2

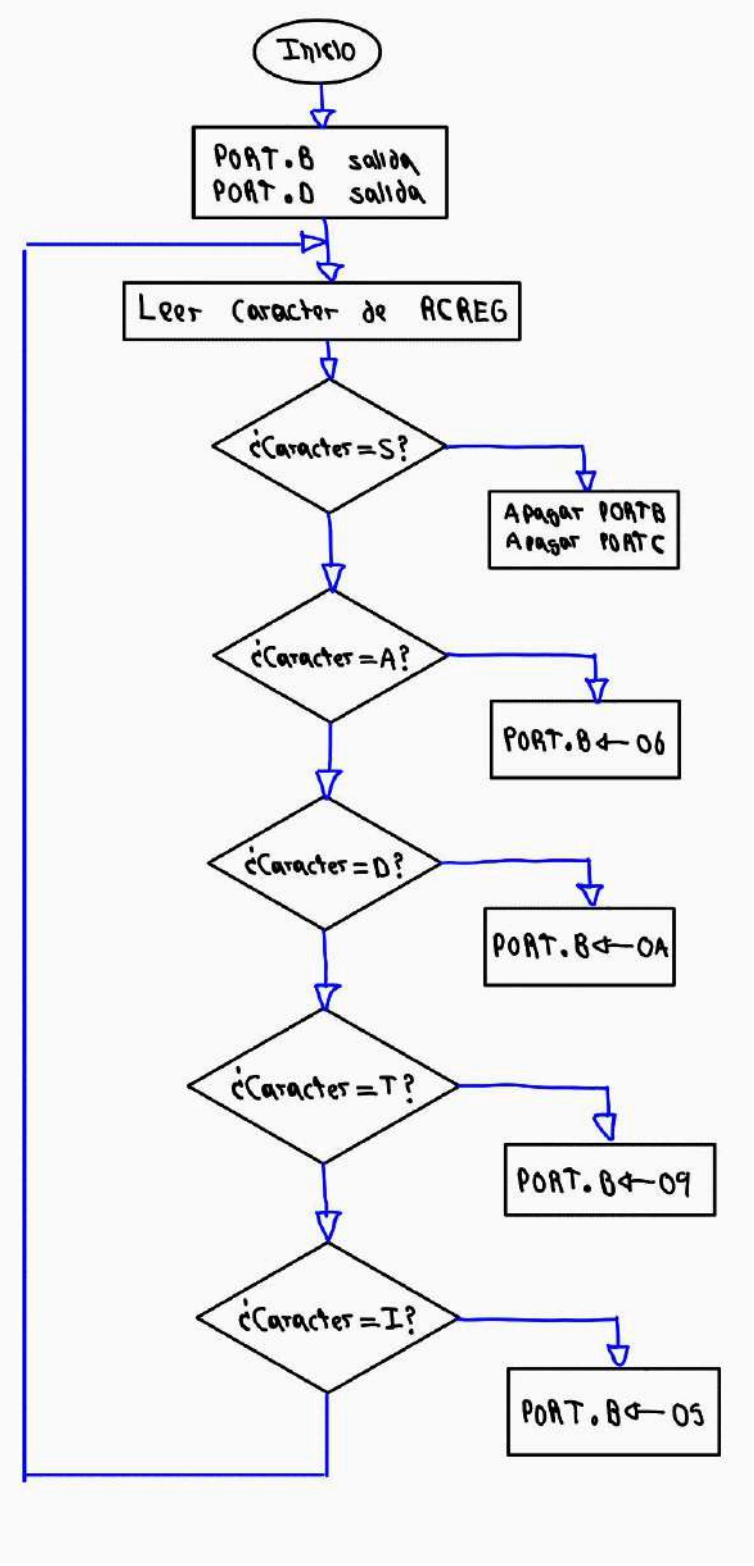
MOTOR1		
PC1	**PB1**	**PB0**
ENABLE_M1	DIR1_M1	DIR2_M1

Tabla 7.3. Asignación de señales de control de los motores de CD.

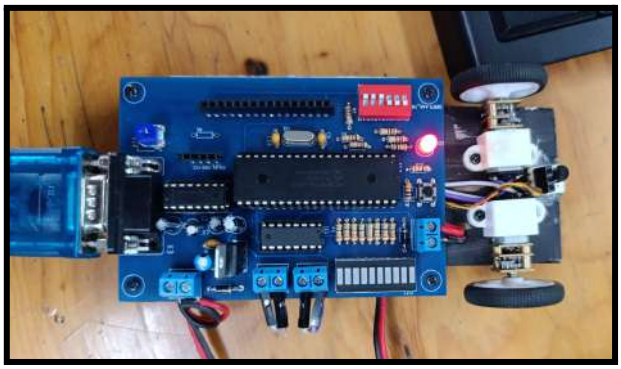
Comando Puerto serie	**ACCION**	
	MOTOR M1	**MOTOR M2**
'S'	PARO	PARO
'A'	DERECHA	DERECHA
'T'	IZQUIERDA	IZQUIERDA
'D'	DERECHA	IZQUIERDA
'I'	IZQUIERDA	DERECHA

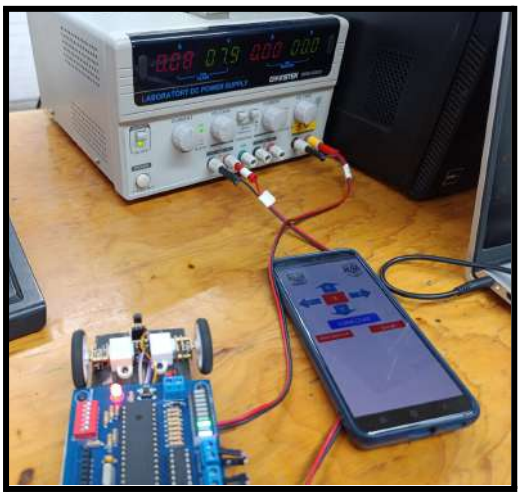
Tabla 7.4 Control de motores, comunicación serie

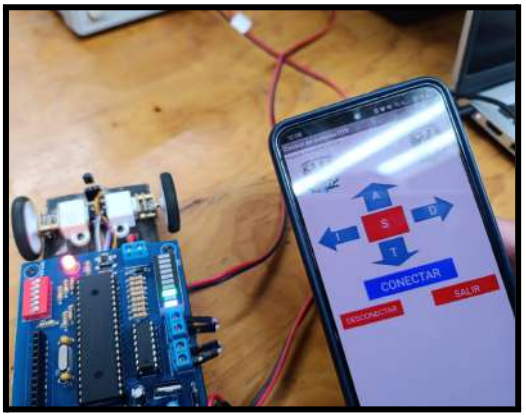
Diagrama de flujo:


Inicio
PORT.B salida
PORT.D salida
Leer Caracter de ACREG
¿Caracter=S?
Apagar PORTB
Apagar PORTC
¿Caracter=A?
PORT.B←06
¿Caracter=D?
PORT.B←0A
¿Caracter=T?
PORT.B←09
¿Caracter=I?
PORT.B←05

imágenes:







Código:

```
PROCESSOR 16F877A
INCLUDE <p16F877a.inc>

CARACTER EQU 0x20

ORG 0
GOTO INICIO_PROGRAMA
ORG 5

INICIO_PROGRAMA:
   CALL CONFIGURACION_INICIAL

ESPERA_DATO:
   BTFSS PIR1,RCIF
   GOTO ESPERA_DATO

   MOVFW RCREG
   MOVWF CARACTER

   MOVLW 0x53
   XORWF CARACTER,W
   BTFSS STATUS,Z
   GOTO $ + 5
   CLRF PORTD
   CLRF PORTB
   BCF PORTB,0
   GOTO ESPERA_DATO

   MOVLW 0x06
   MOVWF PORTD

   MOVLW 0x41
   XORWF CARACTER,W
   BTFSS STATUS,Z
   GOTO $ + 4
   MOVLW 0x06
   MOVWF PORTB
   GOTO ESPERA_DATO

   MOVLW 0x44
   XORWF CARACTER,W
   BTFSS STATUS,Z
```

```
    GOTO $ + 4
    MOVLW 0x0A                          CONFIGURACION_INICIAL:
    MOVWF PORTB                           BSF STATUS,RP0
    GOTO ESPERA_DATO                      BCF STATUS,RP1
                                          BSF TXSTA,BRGH
    MOVLW 0x54                            MOVLW 0x81
    XORWF CARACTER,W                      MOVWF SPBRG
    BTFSS STATUS,Z                        BCF TXSTA,SYNC
    GOTO $ + 4                            BSF TXSTA,TXEN
    MOVLW 0x09                            CLRF TRISB
    MOVWF PORTB                           CLRF TRISD
    GOTO ESPERA_DATO                      BCF STATUS,RP0
                                          BSF RCSTA,SPEN
    MOVLW 0x69                            BSF RCSTA,CREN
    XORWF CARACTER,W                      CLRF PORTB
    BTFSC STATUS,Z                        CLRF PORTD
    GOTO $ + 4                            RETURN
    MOVLW 0x05
    MOVWF PORTB                         END

    GOTO ESPERA_DATO
```

Análisis: El código tiene como objetivo controlar motores mediante comandos recibidos por comunicación serial usando el PIC16F877A. La solución funciona correctamente, ya que se configura el módulo USART para recibir datos a 9600 baudios, se interpreta el carácter recibido y se ejecuta la acción correspondiente encendiendo o apagando salidas en los puertos B y D. Se alcanzaron los objetivos propuestos, ya que los motores responden adecuadamente a los comandos ('S', 'A', 'T', 'D', 'I').

El flujo de datos inicia al recibir un carácter por RCREG, se guarda en CHR y se compara con valores ASCII. Según el resultado, se actualizan PORTB y PORTD para controlar la dirección o apagado de los motores. Se usaron modos de direccionamiento inmediato y directo. Los periféricos empleados fueron el módulo USART y los puertos B y D como salidas digitales. En resumen, el sistema responde correctamente y cumple con la funcionalidad esperada.

Ejercicio 7: Utilizando el termómetro LM35, mostrar la temperatura del ambiente en la terminal de la computadora. La variación del sensor de temperatura es de 10 mV/oC.

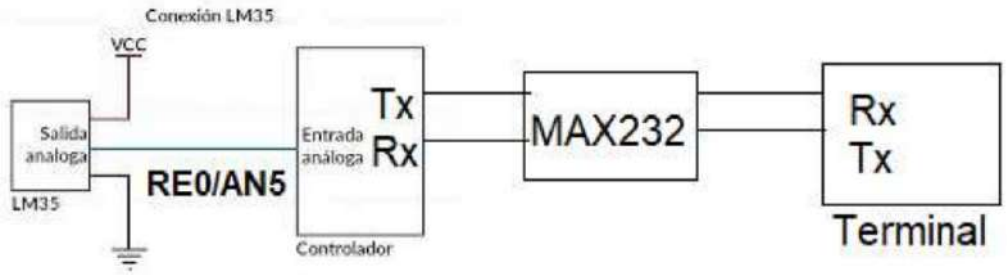
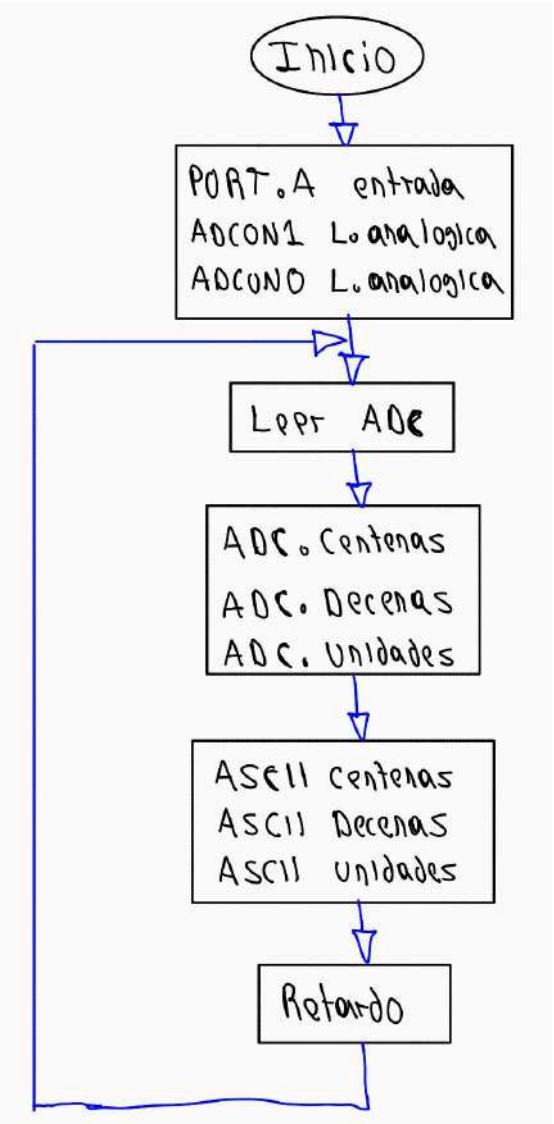
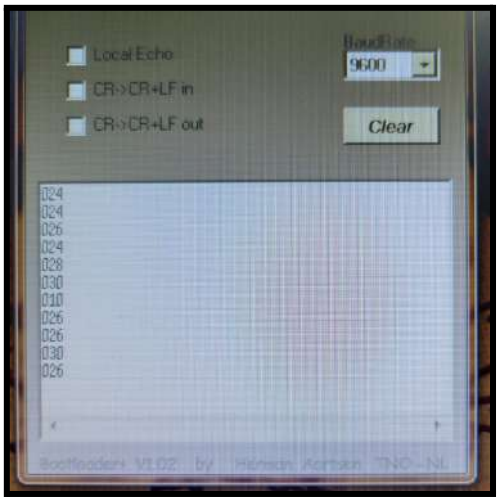
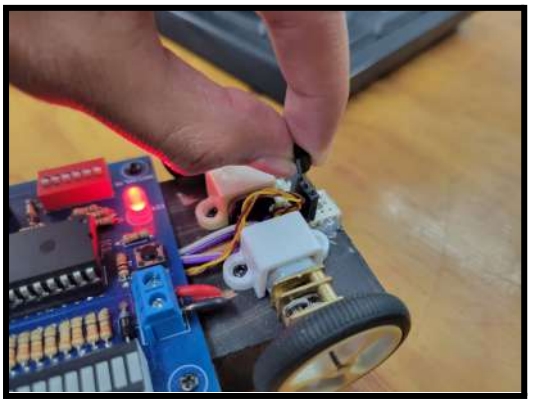


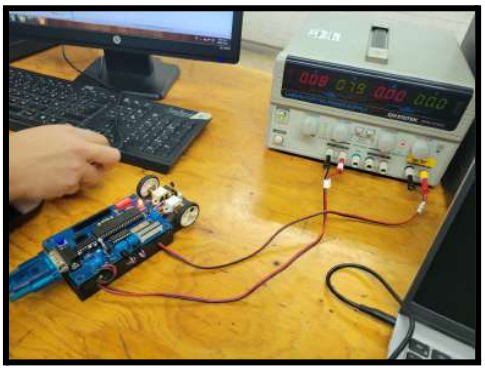
Diagrama de flujo:



imágenes:







Código:

```
        processor 16f877

include<p16f877.inc>

TEMPORAL    EQU H'30'
B_TEMPORAL  EQU H'31'
CONTADOR    EQU H'32'
valor1      equ h'21'
valor2      equ h'22'
valor3      equ h'23'
cte1        equ 50h
cte2        equ 50h
cte3        equ 60h

        ORG 0
        GOTO INICIO
        ORG 5
INICIO:
        CLRF PORTA
        BSF STATUS,RP0
        BCF STATUS,RP1
        MOVLW 00h
        MOVWF ADCON1
        BCF STATUS,RP0
        MOVLW B'11000001'
        MOVWF ADCON0
        BSF STATUS,RP0
        BCF STATUS,RP1
        BSF TXSTA,BRGH
        MOVLW D'129'
        MOVWF SPBRG
        BCF TXSTA,SYNC
        BSF TXSTA,TXEN
        BCF STATUS,RP0
        BSF RCSTA,SPEN
        BSF RCSTA,CREN
        CLRF TEMPORAL
        CLRF B_TEMPORAL
        MOVLW H'70'
        MOVWF CONTADOR

LECTURA:
        BSF ADCON0,2
        CALL RETARDO

ESPERA:
        CLRF TEMPORAL
        CLRF B_TEMPORAL
        BTFSC ADCON0,2
```

```
        GOTO ESPERA
        MOVF
ADRESH,W
        MOVWF
TEMPORAL
        CALL
COMP_CENTENAS
        CALL
COMP_DECENAS
        CALL
COMP_UNIDADES
        CALL FORMATO
        CALL
FORMATO_C
        CALL ENTER
        CALL RETARDO
        DECFSZ
CONTADOR
        GOTO LECTURA
        GOTO BUCLE

BUCLE:
        GOTO BUCLE

FORMATO:
        MOVLW H'F8'
        GOTO
CONFIGURAR

FORMATO_C:
        MOVLW A'C'
        GOTO
CONFIGURAR

ENTER:
        MOVLW H'0D'
        GOTO
CONFIGURAR

COMP_CENTENAS:
        MOVLW 0X33
        SUBWF
TEMPORAL,0
        BTFSS STATUS,C
        GOTO CIEN_0
        GOTO CIEN_1
COMP_DECENAS:
        MOVLW 0X05
        SUBWF
TEMPORAL,W
        BTFSS STATUS,C
        GOTO CERO
        MOVLW 0X0A
        SUBWF
TEMPORAL,W
        BTFSS STATUS,C
        GOTO UNO
        MOVLW 0X0F
        SUBWF
TEMPORAL,W
        BTFSS STATUS,C
        GOTO DOS
        MOVLW 0X14
        SUBWF
TEMPORAL,W
        BTFSS STATUS,C
        GOTO TRES
        MOVLW 0X19
        SUBWF
TEMPORAL,W
        BTFSS STATUS,C
        GOTO CUATRO
        MOVLW 0X1E
        SUBWF
TEMPORAL,W
        BTFSS STATUS,C
        GOTO CINCO
        MOVLW 0X24
        SUBWF
TEMPORAL,W
        BTFSS STATUS,C
        GOTO SEIS
        MOVLW 0X29
        SUBWF
TEMPORAL,W
        BTFSS STATUS,C
        GOTO SIETE
        MOVLW 0X2E
        SUBWF
TEMPORAL,W
        BTFSS STATUS,C
        GOTO OCHO
        GOTO NUEVE

COMP_UNIDADES:
        MOVLW 0X04
        XORWF
TEMPORAL,W
        BTFSC STATUS,Z
        GOTO EST_8_9
        MOVLW 0X03
        XORWF
TEMPORAL,W
        BTFSC STATUS,Z
        GOTO EST_6_7
        MOVLW 0X02
        XORWF
TEMPORAL,W
        BTFSC STATUS,Z
        GOTO EST_4_5
        MOVLW 0X01
        XORWF
TEMPORAL,W
        BTFSC STATUS,Z
        GOTO EST_2_3
        GOTO EST_0_1

EST_8_9:
        BSF STATUS,RP0
        MOVF
ADRESL,W
        MOVWF
B_TEMPORAL
        SWAPF
B_TEMPORAL
        RRF
B_TEMPORAL
        RRF
B_TEMPORAL
        BTFSS
B_TEMPORAL,1
        GOTO OCHO
        GOTO NUEVE

EST_6_7:
        BSF STATUS,RP0
```

```
        MOVF
ADRESL,W
        MOVWF
B_TEMPORAL
        SWAPF
B_TEMPORAL
        RRF
B_TEMPORAL
        RRF
B_TEMPORAL
        BTFSS
B_TEMPORAL,1
        GOTO SEIS
        GOTO SIETE

EST_4_5:
        BSF STATUS,RP0
        MOVF
ADRESL,W
        MOVWF
B_TEMPORAL
        SWAPF
B_TEMPORAL
        RRF
B_TEMPORAL
        RRF
B_TEMPORAL
        BTFSS
B_TEMPORAL,1
        GOTO CUATRO
        GOTO CINCO

EST_2_3:
        BSF STATUS,RP0
        MOVF
ADRESL,W
        MOVWF
B_TEMPORAL
        SWAPF
B_TEMPORAL
        RRF
B_TEMPORAL
        RRF
B_TEMPORAL
        BTFSS
B_TEMPORAL,1
        GOTO DOS
        GOTO TRES

EST_0_1:
        BSF STATUS,RP0
        MOVF
ADRESL,W
        MOVWF
B_TEMPORAL
        SWAPF
B_TEMPORAL
        RRF
B_TEMPORAL
        RRF
B_TEMPORAL
        BTFSS
B_TEMPORAL,1
        GOTO CERO
        GOTO UNO

CIEN_1:
        MOVLW 0X33
        SUBWF
TEMPORAL
        MOVLW A'1'
        GOTO
CONFIGURAR

CIEN_0:
        MOVLW A'0'
        GOTO
CONFIGURAR

NUEVE:
        BCF STATUS,RP0
        MOVLW 0X2E
        SUBWF
TEMPORAL,1
        MOVLW A'9'
        GOTO
CONFIGURAR

OCHO:
        BCF STATUS,RP0
        MOVLW 0X29
        SUBWF
TEMPORAL,1
        MOVLW A'8'
        GOTO
CONFIGURAR

SIETE:
        BCF STATUS,RP0
        MOVLW 0X24
        SUBWF
TEMPORAL,1
        MOVLW A'7'
        GOTO
CONFIGURAR

SEIS:
        BCF STATUS,RP0
        MOVLW 0X1E
        SUBWF
TEMPORAL,1
        MOVLW A'6'
        GOTO
CONFIGURAR

CINCO:
        BCF STATUS,RP0
        MOVLW 0X19
        SUBWF
TEMPORAL,1
        MOVLW A'5'
        GOTO
CONFIGURAR

CUATRO:
        BCF STATUS,RP0
        MOVLW 0X14
        SUBWF
TEMPORAL,1
        MOVLW A'4'
        GOTO
CONFIGURAR

TRES:
        BCF STATUS,RP0
        MOVLW 0X0F
```

```
        SUBWF
TEMPORAL,1
        MOVLW A'3'
        GOTO
CONFIGURAR

DOS:
        BCF STATUS,RP0
        MOVLW 0X0A
        SUBWF
TEMPORAL,1
        MOVLW A'2'
        GOTO
CONFIGURAR

UNO:
        BCF STATUS,RP0
        MOVLW 0X05
        SUBWF
TEMPORAL,1
        MOVLW A'1'
        GOTO
CONFIGURAR

CERO:
        BCF STATUS,RP0
        MOVLW A'0'
        GOTO
CONFIGURAR

CONFIGURAR:
        MOVWF TXREG
        BSF STATUS,RP0
TRANSMITIR:
        BTFSS
TXSTA,TRMT
        GOTO
TRANSMITIR
        BCF STATUS,RP0
        RETURN

RETARDO:
        MOVLW 0x20
        MOVWF valor1
UNO:
        DECFSZ valor1
        GOTO UNO
        RETURN
        END
```

Análisis: El código tiene como objetivo leer una señal analógica mediante el ADC del microcontrolador PIC16F877, convertirla a formato decimal (centenas, decenas y unidades), transformarla a caracteres ASCII y enviarla por el puerto serial. La solución propuesta funciona de forma adecuada, ya que se realiza correctamente la configuración del ADC y del módulo USART, se inicia la conversión analógica, se espera su finalización y se realiza el procesamiento de los datos. El flujo interno de datos comienza con la activación del ADC, la lectura de ADRESH y ADRESL, el almacenamiento del valor en registros temporales, su descomposición mediante comparaciones sucesivas y su transmisión a través del registro TXREG. Se usaron modos de direccionamiento inmediato y directo. Los periféricos empleados fueron el ADC para la conversión analógica-digital y el módulo USART para la transmisión serial. Se puede afirmar que los objetivos se cumplieron porque el sistema logra transformar y enviar el valor analógico como una representación digital legible.

Conclusiones:

Martínez Pérez Brian Erik

En esta práctica utilizamos el PIC para poder realizar programas, donde utilizamos un sensor de temperatura ambiental y un módulo Bluetooth para el cual comunicamos nuestra señales desde una aplicación móvil, la comunicación serial nos sirve para poder comunicar otros dispositivos, este caso lo hacíamos desde los sensores y módulos, hacia el microcontrolador. En el caso particular de la comunicación serial tenemos la característica de que no es necesario tener un reloj compartido que mantiene en sincronía los dos dispositivos que se comunican

Nuñez Rodas Abraham Enrique

Durante esta práctica reforcé mis conocimientos sobre la comunicación serial asíncrona, aprendí a configurar el módulo USART del PIC16F877 y comprendí cómo recibir y transmitir datos entre el microcontrolador y otros dispositivos. Además, me permitió aplicar lo aprendido en ejemplos reales como el control de motores y la lectura de sensores**.**

Vicenteño Maldonado Jennifer Michel
Durante esta práctica utilizamos el puerto serie SCI, recibimos y enviamos datos mediante la interfaz USART, utilizando sensores o incluso la terminal.
También aprendimos a usar el microcontrolador con el módulo Bluetooth que nos permitió manejar el funcionamiento de las llantas a través de nuestro dispositivo móvil.

Referencias:

- del PIC, 2. 1-La Familia. (s/f). 2.- Descripción General del PIC16F877. Edu.ar., de https://exa.unne.edu.ar/ingenieria/sysistemas/public_html/Archi_pdf/HojaDatos/Microcontrolador es/PIC16F877.pdf
- (S/f). Newark.com. Recuperado de https://mexico.newark.com/microchip/pic16f877a-i-p/microcontroller-mcu-8-bit pic16/dp/69K7640?srsltid=AfmBOorWLTceQMTppGk0OMGmmjB6UpIiiw55U 28F2qBZH2pUYET 3EInu
- *Descripción General del PIC16F877 1 2.-Descripción General del PIC16F877 2.1.-La Familia del PIC16F877*. (n.d.). https://exa.unne.edu.ar/ingenieria/sysistemas/public_html/Archi_pdf/HojaDatos/Microcontroladores/PIC16F877.pdf
- *Manual de usuario Microchip PIC16F877A (280 páginas)*. (2025). Manual.cr. https://www.manual.cr/microchip/pic16f877a/manual?utm_
- *PIC16F87XA Devices Included in this Data Sheet: High-Performance RISC CPU*. (n.d.). https://ww1.microchip.com/downloads/en/DeviceDoc/39582C.pdf
- *puertos-de-entradasalida - MIKROE*. (2024). MIKROE. https://www.mikroe.com/ebooks/microcontroladores-pic-programacion-en-c-con-ejemplos/puertos-de-entradasalida