Práctica 10
Comunicación Serial utilizando un Microcontrolador

1. Objetivos de aprendizaje

a. Objetivo general.

- El objetivo de esta práctica es el diseño de una interfaz de usuario, para monitorear y controlar un hábitat para reptiles o terrario, a través de una comunicación serial.

b. Objetivos específicos.

- Diseñar y desarrollar una interfaz para el monitoreo de sensores analógicos a través de una interfaz puerto serie utilizando el componente *SerialPort.*

2. Introducción

2.1 Sensores.

El sensor, se define como un dispositivo capaz de detectar diferentes tipos de materiales recibiendo una señal o estímulo; éste responde a este con una señal eléctrica de salida que se puede cuantificar y manipular.

Los sensores analógicos están generalmente construidos con componentes pasivos y poseen características básicas como: calibración, rango de funcionamiento, confiabilidad, velocidad de respuesta, exactitud, precisión, sensibilidad, linealidad entre otros. Esto permite que el control de la variable a medir se lleve a cabo de la mejor manera y en el menor tiempo posible.

2.2 El Puerto Serie

El puerto serie es una interfaz de comunicación que se basa en un protocolo serial, en el cual los datos son transmitidos y recibidos uno por uno. Generalmente se le asocia con el uso del estándar RS232 (o EIA 232). La norma RS232 establece dos

tipos de conectores llamados DB25 (25 pines) y DB9 (9 pines), macho y hembras. En una computadora puede haber varios puertos series, normalmente denominados COM1, COM2, etc.

La forma de transmitir de comunicarse en un RS232 se detalla en la Tabla 10.1 continuación:

PIN	SEÑAL
1	Data Carrier Detect (DCD)
2	Received Data (RxD)
3	Transmitted Data (TxD)
4	Data Terminal Ready (DTR)
5	Signal Ground (SG)
6	Data Set Ready (DSR)
7	Request to Send (RTS)
8	Clear to Send (CTS)
9	Ring Indicator (RI)

Tabla 10.1 Pines del conector DB-9 (versión 1) Fuente(s): Construcción propia, 2018

Para la comunicación con un microcontrolador es suficiente con tres líneas:

1. Línea de transmisión (TxD), pin 3
2. Línea de recepción (RxD), pin 2
3. Pin de masa (SG), pin 5

La velocidad a la que pueden trabajar los puertos COM de una computadora está normalizada a 75, 150, 300, 600, 1200, 2400, 4800, 9600 baudios, etc. Los requisitos en cuanto a niveles lógicos que debe cumplir una transmisión serie son:

1. Los datos se transmiten con lógica negativa.
2. Para un “0” lógico la línea debe mantener un voltaje entre +3 y +15V
3. Para un “1” lógico debe estar entre -3 y -15V
4. Los voltajes más usados son +12V para el “0” y -12 V para el “1”
5. Cuando un puerto serie no está transmitiendo mantiene el terminal de transmisión a “1” lógico (-12V)

6. La región de transmisión donde los niveles lógicos no están definidos (entre +3V y -3V)

El protocolo establecido por la norma RS232 envía la información estructurada en 4 pares:

1. Bit de inicio o arranque
2. Bit de datos
3. Bit de paridad
4. Bit de parada

Es indispensable que todos los programas que lo formen sean parte de un proyecto. Los proyectos son un conjunto de parámetros y configuraciones que definen a los programas y facilitan el paso de parámetros al compilador; de lo contrario el proceso tedioso sobre todo si se trata de programas muy grandes.

Para el manejo de un puerto serie con un lenguaje de alto nivel como es Visual C# contiene una herramienta que es de gran ayuda para trabajar con los puertos: el *Componente SerialPort*. Modificando las propiedades del componente *SerialPort* se definen las características con las que se llevará a cabo la comunicación ya sea declarando el nombre del puerto COM a utilizar, el tamaño del buffer y el baudaje, entre otros.

Métodos y Propiedades útiles de *SerialPort*:
SerialPort.Open() – Abre un canal de comunicación con las características especificadas en las Propiedades de dicho Componente.
SerialPort.Close() – Cierra el canal de comunicación actual, impidiendo que se envíen o reciban nuevos datos
SerialPort.Write(string text) – Envía la cadena de caracteres *"text"* a través de un Puerto Serie.
SerialPort.ReadExisting() – Regresa una cadena de caracteres proveniente de un Puerto Serie.
SerialPort.IsOpen – Propiedad cuyo valor será true o false dependiendo del estado actual del canal de comunicación.

3. Equipo y material

- Microcontrolador con pines disponibles analógico, digitales y comunicación.
- Sensor de temperatura
- Resistencias.

- Resistencia variable.
- Luces indicadoras (cuatro al menos para las salidas).
- Protoboard y cables.
- Fuente de alimentación.
- Bocina
- Fotorresistor
- Potenciómetro

4. Metodología.

Se solicita al alumno que entienda la teoría sobre los registros asociados a la configuración de los pines de los puertos en un microcontrolador para emplearlos como entradas o salidas y terminales de comunicación. Adicionalmente se solicita que genere un programa donde se lea información del medio externo, se procese la información en forma digital dentro del microcontrolador y posteriormente se despliegue la información relacionada con las entradas utilizando el protocolo de comunicación.

5. Desarrollo

a. Actividad I.

Diseño del programa. Se recomienda el uso de una tabla de entradas y salidas, definir las variables internas que necesite el programa (en su caso), generar y discutir diagramas de flujo. Se recomienda el uso de computadora y los programas necesarios para la compilación del programa en el lenguaje seleccionado, además, se debe consultar la hoja de especificaciones del microcontrolador.

b. Actividad II

Diseño y simulación del circuito electrónico. Se consulta la hoja de especificaciones del microcontrolador para conectar los dispositivos de entrada y salida, así como los elementos de soporte, por ejemplo, el oscilador, la fuente de alimentación, el botón de reinicio (*reset*), por mencionar algunos. Se recomienda el uso de computadora y los programas necesarios para la simulación del circuito, empleando el programa diseñado en la actividad I.

c. Actividad III

Alambrado del circuito. Interconectar los elementos seleccionados siguiendo el diagrama electrónico en una tarjeta *protoboard* (placa que posee unos orificios conectados eléctricamente entre sí siguiendo un patrón horizontal o vertical.

Es empleada para realizar pruebas de circuitos electrónicos), previamente se debe grabar el código generado para el microcontrolador (archivo *.HEX), en un grabador, finalmente verificar que en el circuito no haya cortocircuitos antes de energizar el sistema.

6. Resultados

Para que el usuario de este manual pueda ver resultados, es necesario definir qué acción realizará la salida ante la entrada. Se verificará que se cumplan los valores de entrada en comparación al valor obtenido en la salida mediante luces indicadoras y puerto serie.

7. Aplicaciones

EL uso de la interfaz de usuario nos permite leer valores ingresados a nuestro PIC mediante una gran cantidad de sensores (sensor de temperatura, de proximidad, de presencia, etc.) y de esta forma poder interpretar y manipular estos valores de una mejor forma.

8. Bibliografía

- ANGULO J. (2000). Microcontroladores PIC. España. Paraninfo.
- GONZÁLEZ J. (1992). Introducción a los Microcontroladores. España. Graw-Hill.
- MYKE P. (2001). Programming and Customizing the PIC Microcontroller. McGraw-Hill. México.
- PALACIOS E. (2006). Microcontrolador PIC16F84, Desarrollo de proyectos México: Alfaomega Ra-Ma.
- TAFANERA A. (2000). Teoría y diseño con Microcontroladores PIC. México. Autores Editores.
- TORRES P. (1994). Microprocesadores y Microcontroladores Aplicados a la Industria. Madrid. Paraninfo.

- Microchip® (2019), PIC16F887 enero del 2019, de Microchip Sitio web: http://ww1.microchip.com/downloads/en/DeviceDoc/30292D.pdf

9. Posible Solución.

Lista de material sugerido

1. Microcontrolador: PIC16F887 o similar
2. Luces indicadoras: Barra de LEDs
3. Resistencias: 16 de 330 ∧ y 17 de 1K∧
4. Protoboard
5. Cables de colores: Calibre 22
6. Fuente de poder: 5V CD
7. Computadora: Programas de simulación y compilación
8. Grabador de microcontroladores.
9. Sensor de temperatura: LM35
10. Bocina: Bocina de 8∧
11. Fotorresistencias
12. Potenciómetro

DESARROLLO

a. Actividad I.

Diseño del programa y circuito electrónico.

Generando una tabla de entradas y salidas para las conexiones al microcontrolador. Se consultó la hoja de especificaciones.

Entrada	BIT	PIN	Registro de configuración asociado	Salida	BIT	PIN	Registro de configuración asociado, (Asignar “ceros” lógicos)
Analógicos	AN0	2	ANSEL ANSELH	PORTC	0	15	TRISC
					1	16	
	AN1	3			2	17	
	AN2	4					

Entrada	BIT	PIN	Registro
Comunicación RS232	Rx Tx	25 26	#use RS232

Tabla 10.2. Registros de entradas y salidas (versión 1) Fuente(s): Construcción propia, 2018.

El siguiente programa sirve para observar el uso e implementación del puerto serial para permitir una comunicación entre nuestro PIC y algún equipo o dispositivo que nos permita observar los datos obtenidos. El resultado de la operación se despliega en el Puerto C a través de 3 luces (Tabla 10.2), también es posible observar en una hiperterminal los resultados de los sensores y acceder a las posibilidades mediante el teclado de la computadora. La Figura 10.3 muestra el diagrama de flujo. A continuación, se muestra una breve descripción del programa diseñado.

1. Inicio.
2. Para configurar los puertos de entrada y de salidas con ayuda de los registros relacionados indicados por el fabricante. (Ver Tabla 10.1).
3. En el puerto C observamos los valores binarios obtenidos mediante los puertos de entrada e interpretados en el microcontrolador.
4. Repite.

Figura 10.3. Diagrama de flujo del ejemplo de uso de puerto serial para la interpretación de datos (versión 1) Fuente(s): Construcción propia, 2018.

b. Actividad II

Simulación del circuito.

Se recomienda el uso de computadora y los programas necesarios para la simulación y la compilación del programa en lenguaje ensamblador.

El código mostrado en la Figura **10.4** se diseñó en el entorno de desarrollo de MPLAB para este documento.

```
#include <16F887.h>
#device ADC = 10
#fuses XT,NOWDT,NOPROTECT,NOLVP, NOMCLR, NOCPD, NOBROWNOUT, NODEBUG, NOLVP, NOIESO, NOFCMEN
#use delay(clock=4000000)
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)

int8 comando;

#define PIN_FAN  PIN_C0
#define PIN_LAMP PIN_C1
#define PIN_PUMP PIN_C2

#define CAPTURE    'A'  // Enviar valores de sensores
#define FAN_OFF    'B'  // Apagar ventilador
#define FAN_ON     'C'  // Encender ventilador
#define LAMP_OFF   'D'  // Apagar lampara
#define LAMP_ON    'E'  // Encender lampara
#define PUMP_OFF   'F'  // Apagar bomba de agua
#define PUMP_ON    'G'  // Encender bomba de agua

void main()
{
  //Bloque de configuracion
   int16  adc;
   float temp;
   float hum;
   float luz;
   setup_adc(ADC_CLOCK_INTERNAL);
   setup_adc_ports(ALL_ANALOG);
   comando = 0;
   while(true)
    {
     comando = getch();
     switch(comando)
     {
       case CAPTURE:
        set_adc_channel(0);                    //Canal 0 para sensor de calor
                adc=read_adc();
                temp=adc*0.019;                        //Guarda lectura de sensor en temp
                set_adc_channel(1);                    //Canal 1 para sensor de humedad
                adc=read_adc();
                hum=adc;                               //Guarda lectura de sensor en hum
                set_adc_channel(2);                    //Canal 2 para sensor de luz
                adc=read_adc();
                luz=adc;                  //Guarda lectura de sensor en luz
                printf("%f;%f;%f \n",temp,hum,luz); //Imprime lecturas
```

```
        break;
      case FAN_OFF:
        output_low(PIN_FAN);
        break;
      case FAN_ON:
        output_high(PIN_FAN);
        break;
      case LAMP_OFF:
        output_low(PIN_LAMP);
        break;
      case LAMP_ON:
        output_high(PIN_LAMP);
        break;
      case PUMP_OFF:
        output_low(PIN_PUMP);
        break;
      case PUMP_ON:
        output_high(PIN_PUMP);
        break;
    }
  }
    //comando = 0;
}
```

Figura 10.4. El programa en lenguaje C (versión 1) Fuente(s): Construcción propia, 2018.

En la Figura 10.5 se muestra una captura de pantalla del funcionamiento del PIC en el ambiente de simulación Proteus.

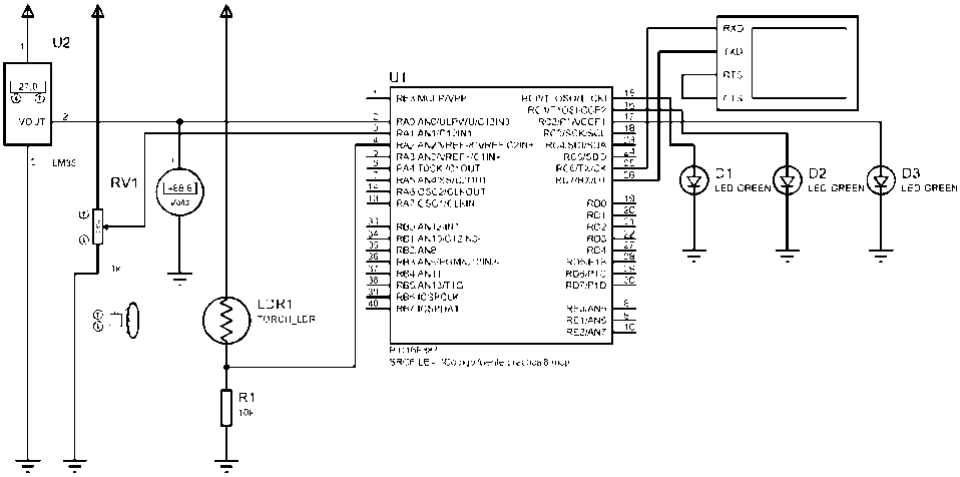


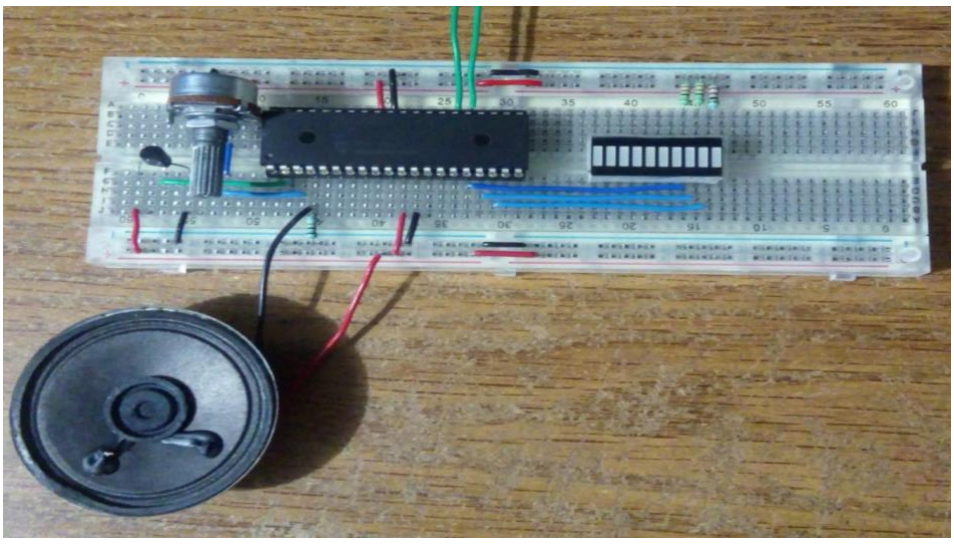
Figura 10.5. Circuito de la simulación, se observa que las salidas de la derecha reflejan los valores obtenidos por los puertos de entrada (versión 1) Fuente(s): Construcción propia, 2018.

c. Actividad III

Alambrado del circuito.

Se muestra en las figuras 10.6 un circuito armado de la solución con los componentes sugeridos.

Figura 10.6. El circuito real de la práctica (versión 1) Fuente(s): Construcción propia, 2018.



Resultados / Conclusión

Se configuraron puertos de un microcontrolador, entradas (analógicos) y salidas (digitales), se ingresó la información desde la entrada analógica de ocho bits donde, las entradas se reflejaron en las salidas del microcontrolador en forma digital. Además, se monitorea a través de una Interfaz puerto serie utilizando el componente *SerialPort*, para poder ser monitoreada desde cualquier interfaz de usuario.

10 AGRADECIMIENTOS

- **Trabajo realizado con el apoyo del Programa UNAM-DGAPA-PAPIME PE110618.**
- **Trabajo realizado con el apoyo de la Facultad de Estudios Superiores Aragón.**