Comunicacion serie

Configuración en Sistema Embebido

- Conecta los sensores a la placa según corresponda (por ejemplo, sensores de temperatura, luz, humedad, etc.).

- Escribe un programa que lea los valores de los sensores y los envíe por el puerto serie. Por ejemplo:

```
void setup() {
  Serial.begin(9600); // Inicia la comunicación serial a 9600 baudios
}

void loop() {
  // Leer valores de los sensores
  float temperatura = obtenerTemperatura(); // Funcion para obtener temperatur
  int luz = obtenerLuz(); // Funcion para obtener luz
  int humedad = obtenerHumedad(); // Funcion para obtener humedad

  // Enviar datos por puerto serie
  Serial.print(temperatura);
  Serial.print(",");
  Serial.print(luz);
  Serial.print(",");
  Serial.println(humedad);

  delay(1000); // Espera 1 segundo
}

float obtenerTemperatura() {
  // Código para obtener temperatura
  return (25.0) //En este ejmplo siempre se devuelve el mismo valor
}

int obtenerLuz() {
  // Código para obtener luz
  return(1000) //En este ejemplo siempre se devuelve el mismo valor
}

int obtenerHumedad() {
  // Código para obtener humedad
  return(70) //En este ejemplo siempre se devuelve el mismo valor
}
```

Programa en Python

- Instalación de PySerial, para instalar la librería PySerial, se utiliza el comando pip:

```
pip install pyserial
```

- Programa en Python que recibe los datos enviados por la placa a través del puerto serie.

```
import serial
import time

# Configura el puerto serial
puerto = '/dev/ttyUSB0'  # Usa '/dev/ttyUSB0' n Linux o 'COM3' en Windows. Aju
baudios = 9600
ser = serial.Serial(puerto, baudios, timeout=1)

def recibir_datos():
        if ser.in_waiting > 0:
                datos = ser.readline().decode('utf-8').strip()
                return datos.split(',')

while True:
        datos = recibir_datos()
        if datos:
                temperatura, luz, humedad = map(float, datos)
                print(f'Temperatura: {temperatura} °C, Luz: {luz}, Humedad: {h

        time.sleep(1)  # Espera 1 segundo antes de la próxima lectura

# Cierra el puerto serial al finalizar
ser.close()
```

- Ejecuta el programa de Python mientras el Sistema embebido envía datos por puerto serie. Recuerda que solo puede acceder a este puerto un programa a la vez, por lo que cierra el IDE de arduino u otro monitor serial.
- Ajusta el formato de los datos según sea necesario para tu Proyecto Integrador.

Graficar los datos recibidos

A partir de recibir los datos en Python, podemos utilizar esta información de múltiples formas, por ejemplo guarrdarlos en un archivo, enviarlos a un SGDB o graficarlos. Veamos como usar la librería Matplotlib para esto úlitmo.

Nuevamente, instalemos la librería si aun no la tenemos:

```
pip install matplotlib
```

Modifiquemos el programa usado anteriormente para graficar los datos que se van recibiendo:

```
import serial
import time
import matplotlib.pyplot as plt

# Configuración del puerto serial
puerto = 'COM3'  # Cambia esto por tu puerto si es diferente
baudios = 9600

# Intentar abrir el puerto serial
try:
    ser = serial.Serial(puerto, baudios, timeout=1)
    print(f'Puerto {puerto} abierto correctamente')
except serial.SerialException as e:
    print(f'Error al abrir el puerto: {e}')
    exit()

# Listas para almacenar los datos
tiempo = []
temperaturas = []
luces = []
humedades = []

# Configuración del gráfico
plt.ion()  # Habilitar modo interactivo
fig, (ax1, ax2, ax3) = plt.subplots(3, 1, figsize=(10, 8))

def recibir_datos():
    if ser.in_waiting > 0:
        datos = ser.readline().decode('utf-8').strip()
        return datos.split(',')

# Contador para el tiempo
contador = 0

try:
    while True:
        datos = recibir_datos()
        if datos:
            try:
```

```
                # Parsear los datos recibidos
                temperatura, luz, humedad = map(float, datos)

                # Añadir los datos a las listas
                tiempo.append(contador)
                temperaturas.append(temperatura)
                luces.append(luz)
                humedades.append(humedad)

                # Limitar el tamaño de las listas para que no crezcan indefini
                if len(tiempo) > 50:  # Solo mantener los últimos 50 puntos
                    tiempo.pop(0)
                    temperaturas.pop(0)
                    luces.pop(0)
                    humedades.pop(0)

                # Actualizar las gráficas
                ax1.clear()
                ax2.clear()
                ax3.clear()

                ax1.plot(tiempo, temperaturas, label='Temperatura (°C)', color
                ax2.plot(tiempo, luces, label='Luz', color='g')
                ax3.plot(tiempo, humedades, label='Humedad (%)', color='b')

                ax1.set_ylabel('Temperatura (°C)')
                ax2.set_ylabel('Luz')
                ax3.set_ylabel('Humedad (%)')
                ax3.set_xlabel('Tiempo (s)')

                # Mostrar leyendas
                ax1.legend()
                ax2.legend()
                ax3.legend()

                # Refrescar la gráfica
                plt.pause(0.1)

                contador += 1
            except ValueError:
                print("Error al convertir los datos.")

        time.sleep(0.1)  # Esperar 0.1 segundos antes de la próxima lectura

except KeyboardInterrupt:
    print("Interrupción manual detectada. Cerrando el programa...")

finally:
    ser.close()  # Asegurarse de cerrar el puerto serial
    print("Puerto serial cerrado.")
```

Podemos ampliar este programa, utilizando un menú que nos permita seleccionar que accion ocurre en cada momento, por ejemplo que variable se grafica. Recordemos que para crear un Menu de usuario podemos usar el bucle While y una estructura if-elif.

Autor

Ing. Lisandro Lanfranco

Fuentes

https://pythonhosted.org/pyserial/shortintro.html#opening-serial-ports

https://discuss.python.org/t/access-denied-error-in-serial-communications/19448

https://matplotlib.org/

Historial de Cambios

241007 - v01 - Creación
241014 - v02 - Agregado de graficos