

Universidad Autónoma De Tamaulipas

Ingeniería en sistemas computacionales

Materia:

Programación de Interfaces y Puertos

Alumno:

De la cruz Márquez Rodrigo

Semestre:

6-G

1.-

int ldr = A0;

int led = 13;

void setup() {

  Serial.begin(9600);

  pinMode(led, OUTPUT);

}

void loop() {

  int valor = analogRead(ldr);

  if (valor > 500) {

    digitalWrite(led, LOW);

    Serial.println("LEDs Apagados");

  } else {

    digitalWrite(led, HIGH);

    Serial.println("LEDs Encendidos");

  }

  delay(100);

}

Diagrama

Descripción generada automáticamente

import serial  
import sys  
from PyQt5 import uic, QtWidgets, QtGui, QtCore  
qtCreatorFile = "LEDS y Fotorresistencias.ui" # Nombre del archivo aquí.  
Ui\_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)  
  
class MyApp(QtWidgets.QMainWindow, Ui\_MainWindow):  
 def \_\_init\_\_(self):  
 QtWidgets.QMainWindow.\_\_init\_\_(self)  
 Ui\_MainWindow.\_\_init\_\_(self)  
 self.setupUi(self)  
  
 # Área de los Signals  
 self.btn\_accion.clicked.connect(self.accion)  
  
 self.arduino = None  
  
 self.segundoPlano = QtCore.QTimer()  
 self.segundoPlano.timeout.connect(self.lecturaArduino)  
  
 #Area de Slots  
  
  
 def accion(self):  
 texto\_boton = self.btn\_accion.text()  
 com = self.txt\_com.text()  
 if texto\_boton == "CONECTAR" and self.arduino is None:  
 self.arduino = serial.Serial(port=com, baudrate=9600, timeout=1)  
 self.segundoPlano.start(100)  
 self.btn\_accion.setText("DESCONECTAR")  
 elif texto\_boton == "DESCONECTAR" and self.arduino.isOpen():  
 self.segundoPlano.stop()  
 self.arduino.close()  
 self.btn\_accion.setText("RECONECTAR")  
 else:  
 self.arduino.open()  
 self.segundoPlano.start(100)  
 self.btn\_accion.setText("DESCONECTAR")  
  
 def lecturaArduino(self):  
 if not self.arduino is None and self.arduino.isOpen():  
 if self.arduino.inWaiting():  
 cadena = self.arduino.readline()  
 cadena = cadena.decode()  
 cadena = cadena.strip()  
 #print(cadena)  
 if cadena != "":  
 self.datos.addItem(cadena)  
 self.datos.setCurrentRow(self.datos.count()-1)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app = QtWidgets.QApplication(sys.argv)  
 window = MyApp()  
 window.show()  
 sys.exit(app.exec\_())

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

2.-

.- #include <Servo.h>

const int pirPin = 4;

const int servoPin = 3;

Servo servo;

void setup() {

  Serial.begin(9600);

  pinMode(pirPin, INPUT);

  servo.attach(servoPin);

  servo.write(0);

}

void loop() {

  int pirState = digitalRead(pirPin);

  if (pirState == HIGH) {

    servo.write(90);

    Serial.println("Movimiento detectado! Moviendo servo a 90°");

    delay(2000);

  } else {

    servo.write(0);

    Serial.println("No hay movimiento. Servo en 0°");

  }

  delay(1000);

}

Diagrama

Descripción generada automáticamente

import serial  
import sys  
from PyQt5 import uic, QtWidgets, QtGui, QtCore  
qtCreatorFile = "ServoyPIR.ui" # Nombre del archivo aquí.  
Ui\_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)  
  
class MyApp(QtWidgets.QMainWindow, Ui\_MainWindow):  
 def \_\_init\_\_(self):  
 QtWidgets.QMainWindow.\_\_init\_\_(self)  
 Ui\_MainWindow.\_\_init\_\_(self)  
 self.setupUi(self)  
  
 # Área de los Signals  
 self.btn\_accion.clicked.connect(self.accion)  
  
 self.arduino = None  
  
 self.segundoPlano = QtCore.QTimer()  
 self.segundoPlano.timeout.connect(self.lecturaArduino)  
  
 #Area de Slots  
  
  
 def accion(self):  
 texto\_boton = self.btn\_accion.text()  
 com = self.txt\_com.text()  
 if texto\_boton == "CONECTAR" and self.arduino is None:  
 self.arduino = serial.Serial(port=com, baudrate=9600, timeout=1)  
 self.segundoPlano.start(100)  
 self.btn\_accion.setText("DESCONECTAR")  
 elif texto\_boton == "DESCONECTAR" and self.arduino.isOpen():  
 self.segundoPlano.stop()  
 self.arduino.close()  
 self.btn\_accion.setText("RECONECTAR")  
 else:  
 self.arduino.open()  
 self.segundoPlano.start(100)  
 self.btn\_accion.setText("DESCONECTAR")  
  
 def lecturaArduino(self):  
 if not self.arduino is None and self.arduino.isOpen():  
 if self.arduino.inWaiting():  
 cadena = self.arduino.readline()  
 cadena = cadena.decode()  
 cadena = cadena.strip()  
 #print(cadena)  
 if cadena != "":  
 self.datos.addItem(cadena)  
 self.datos.setCurrentRow(self.datos.count()-1)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app = QtWidgets.QApplication(sys.argv)  
 window = MyApp()  
 window.show()  
 sys.exit(app.exec\_())

Imagen que contiene Escala de tiempo

Descripción generada automáticamente

3.-

#include <Servo.h>

const int sensorPin = A0;

const int relayPin = 7;

const int motorPin1 = 5;

const int motorPin2 = 6;

const int enablePinA = 9;

const int enablePinB = 10;

Servo servo;

void setup() {

  pinMode(sensorPin, INPUT);

  pinMode(relayPin, OUTPUT);

  pinMode(motorPin1, OUTPUT);

  pinMode(motorPin2, OUTPUT);

  pinMode(enablePinA, OUTPUT);

  pinMode(enablePinB, OUTPUT);

  servo.attach(3);

}

void loop() {

  int sensorValue = analogRead(sensorPin);

  float temperature = (sensorValue \* 5.0 \* 100.0) / 1024.0;

  if (temperature > 25) {

    digitalWrite(relayPin, HIGH);

    digitalWrite(motorPin1, HIGH);

    digitalWrite(motorPin2, LOW);

    analogWrite(enablePinA, 255);

    analogWrite(enablePinB, 0);

    Serial.println("Temperatura alta! Encendiendo motor en una dirección.");

  } else {

    digitalWrite(relayPin, LOW);

    analogWrite(enablePinA, 0);

    analogWrite(enablePinB, 0);

    Serial.println("Temperatura normal. Motor apagado.");

  }

  delay(1000);

}

Interfaz de usuario gráfica, Diagrama

Descripción generada automáticamente

import serial  
import sys  
from PyQt5 import uic, QtWidgets, QtGui, QtCore  
qtCreatorFile = "Punte H y Relevador.ui" # Nombre del archivo aquí.  
Ui\_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)  
  
class MyApp(QtWidgets.QMainWindow, Ui\_MainWindow):  
 def \_\_init\_\_(self):  
 QtWidgets.QMainWindow.\_\_init\_\_(self)  
 Ui\_MainWindow.\_\_init\_\_(self)  
 self.setupUi(self)  
  
 # Área de los Signals  
 self.btn\_accion.clicked.connect(self.accion)  
  
 self.arduino = None  
  
 self.segundoPlano = QtCore.QTimer()  
 self.segundoPlano.timeout.connect(self.lecturaArduino)  
  
 #Area de Slots  
  
  
 def accion(self):  
 texto\_boton = self.btn\_accion.text()  
 com = self.txt\_com.text()  
 if texto\_boton == "CONECTAR" and self.arduino is None:  
 self.arduino = serial.Serial(port=com, baudrate=9600, timeout=1)  
 self.segundoPlano.start(100)  
 self.btn\_accion.setText("DESCONECTAR")  
 elif texto\_boton == "DESCONECTAR" and self.arduino.isOpen():  
 self.segundoPlano.stop()  
 self.arduino.close()  
 self.btn\_accion.setText("RECONECTAR")  
 else:  
 self.arduino.open()  
 self.segundoPlano.start(100)  
 self.btn\_accion.setText("DESCONECTAR")  
  
 def lecturaArduino(self):  
 if not self.arduino is None and self.arduino.isOpen():  
 if self.arduino.inWaiting():  
 cadena = self.arduino.readline()  
 cadena = cadena.decode()  
 cadena = cadena.strip()  
 #print(cadena)  
 if cadena != "":  
 self.datos.addItem(cadena)  
 self.datos.setCurrentRow(self.datos.count()-1)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app = QtWidgets.QApplication(sys.argv)  
 window = MyApp()  
 window.show()  
 sys.exit(app.exec\_())

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

Enlace repositorio: