LAB REPORT

MCTA 3203

SECTION 1

GROUP D

EXPERIMENT 3A

Parallel, Serial, and USB interfacing with microcontroller and computer-based system (1):

Sensors and actuators

MATIC NO.	NAME
2111633	MUHAMMAD FARID BIN JAFRI
2110765	MUHAMMAD ARIF SYAZWAN BIN MOHD RIZAL
2112969	MUHAMMAD HAZIQ HAIKAL BIN SUAIB
2115207	MUHAMMAD AMIRUL SYAFIQ BIN RUSLANI
2116281	MUHAMMAD FADHLUL WAFI BIN AHMAD NAIM

SUBMISSION DATE: 1/11/2023

INSTRUCTOR: DR. WAHJU SEDIONO

Abstract

The primary aim of this project is to develop a serial communication interface between the Python programming language and an Arduino microcontroller. This interface will facilitate the transmission of data from a potentiometer connected to the Arduino to a computer via a USB connection. This facilitates comprehension of the procedure for establishing serial connectivity and enables the utilisation of real-time data from the Arduino in Python programs. In order to establish serial communication, the Arduino will be programmed to provide potentiometer data in a continuous manner through a serial interface. Conversely, a Python script will utilise the pyserial package to create a serial connection and get the data transmitted by the Arduino. The project yielded significant results, demonstrating the successful development of a serial communication link between Python and an Arduino. This achievement facilitated the seamless real-time transfer of potentiometer readings. The project showcased the potential for establishing dependable connection between Arduino and Python, enabling the utilisation of the acquired data for many purposes, including data visualisation through plotting.

In summary, this experiment showcases the effective development of a serial communication interface between the Python programming language and an Arduino microcontroller. This interface enables the interchange of data, specifically pertaining to potentiometer readings. The efficacy of this communication architecture for real-time data usage in Python programs is demonstrated through the configuration of both ends. The results of this project contribute to the establishment of a fundamental comprehension of the process of serial communication between microcontrollers and computers, hence creating opportunities for diverse applications and projects in subsequent endeavours.

Table of Contents

Introduction

The goal of this project is to set up a serial communication link between an Arduino and Python so that data can be sent via USB from the Arduino's potentiometer to a computer. This makes it easy to understand how to connect to a serial port and lets Python programs use real-time data from an Arduino.

The microcontroller Arduino is used to read data from devices and run hardware. An Arduino would be used to connect to the gear and sensors and send data to the computer since the computer itself can't get to them. Plus, Arduino boards are a lot less expensive than most computers, so if a motor or device fails and destroys an Arduino, it's not as bad as if it were a computer.

Python is a flexible computer language that is often used in data science. The pyserial tool lets a Python script read and write to a serial port. So, a Python script can be used to set up contact between a computer and an Arduino. Also, Python's flexibility in changing data lets you do complicated calculations with access to more computing power.

Materials and Equipment

Materials Needed:

1. Arduino Board
2. Potentiometer
3. Jumper Wires
4. LED
5. 220 resistor
6. Breadboard

Equipment:

1. Computer with necessary software for programming and testing (Arduino IDE)
2. Screwdriver and pliers (for assembly and adjustments)

Experimental Setup

1. Microcontroller or Digital Logic IC: The microcontroller or digital logic IC (such as an Arduino or 74HC595) has been integrated into the circuit. It was connected to the breadboard and potentiometers, ensuring the appropriate establishment of input and output pin connections.

2. Potentiometer: The potentiometer was placed on the breadboard. One pin was connected to the ground. The other one is to the pin in Arduino and the last pin is to the 5V . Pull-up or pull-down resistors were implemented as needed, depending on our specific circuit design.

3. LED: The anode LED was connected to the ground and cathode to the resistor that is connected to the pin. All the jumper wires were connected tightly.

4. Power Supply: Power supply has been connected to provide the necessary voltage for the circuit. We ensured that the voltage levels align with the operating requirements of the components and took into consideration the current demands of the potentiometers.

5. Jumper Wires: Jumper wires were employed to establish electrical connections between the components. We have maintained an organised and clear wiring layout to prevent confusion or short circuits.

6. Programming: We programmed the microcontroller to interpret the signals from the potentiometers and read the values in Python. The values of potentiometers can be increased or decreased and the output in Python can be read as described in the experiment steps. The specific programming details depend on the microcontroller or IC in use.

7. Testing: Before proceeding with the experiment, the circuit has been verified to be correctly assembled and programmed. The potentiometers and the display in Python have been tested to ensure that they function as intended.

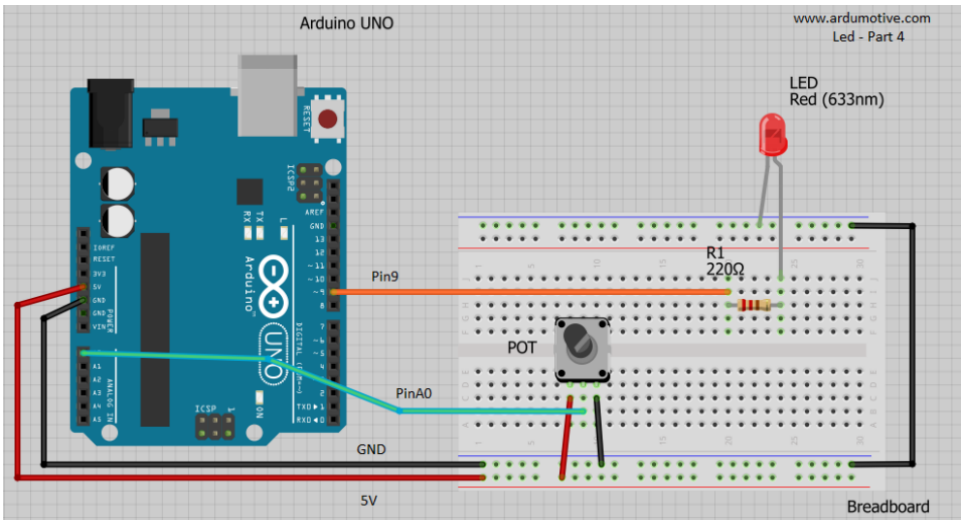

Figure 1

Methodology

Circuit Setup:

1. The one end of the potentiometer was attached to the 5V terminal on the Arduino.
2. The opposite end of the potentiometer was connected to the GND (Ground) terminal on the Arduino.
3. The middle leg, also known as the wiper, of the potentiometer has been wired to an analog input pin on the Arduino, like A0. We have referred to Figure 1 for the illustration of the circuit setup.

Experiment Steps:

1. The connection between the Arduino and the computer was established using a USB cable.
2. The Arduino has been activated by uploading the sketch to it through the Arduino IDE.
3. The Python script was executed on the computer.
4. When the potentiometer knob was adjusted, the readings of the potentiometer were observed in the Python terminal.
5. We have used these readings for various purposes, such as experiments, data logging, or control applications, tailored to our project's specific needs.

Results (Observation)

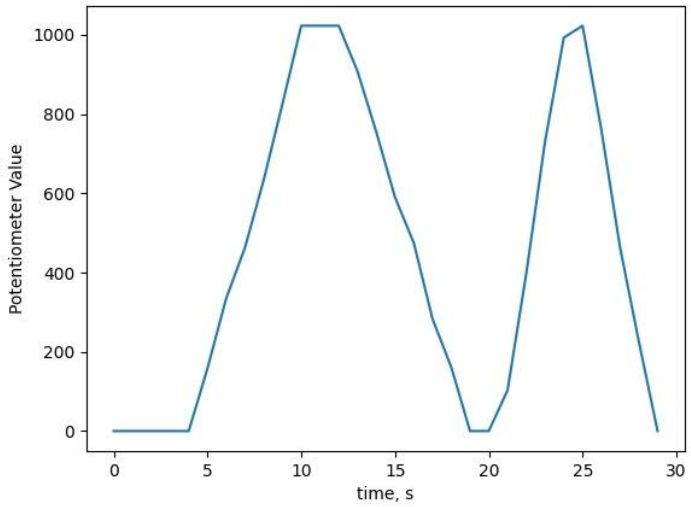

Figure 2

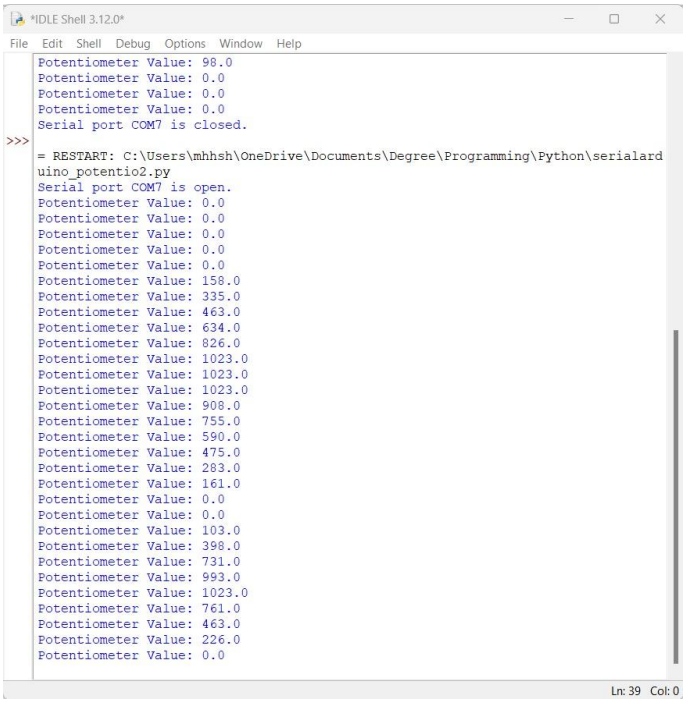

Figure 3

As the potentiometer is turned, the reading of potentiometer value is shown in Python script. This changing value indicates an established serial communication. When the potentiometer is rotated clockwise, the value of the potentiometer will increase while the value will decrease if it is rotated counterclockwise. The minimum value is 0 and the maximum value is 1023.

Link for the video:
https://drive.google.com/file/d/1gGuS0o6I7A0U_pYiOoQrOCiqT6tH9F3Y/view?usp=drivesdk

Discussions

The raw analogue value of the potentiometer varies from 0 to 1023 due to the long jumper wire and fragile connection. We convert the value from 0 to 1023 into a percentage from 0 to 100. This eliminates the tiny fluctuations and smoothes the value change.

A single programme can access the serial port. When we try to upload the code, we get warnings since we forgot to stop the Python script from accessing the port, and vice versa. This leads to a lot of frustration and wasted time. When we are finished with a programme that uses serial communication, we close it.

Conclusion

The objective of this experiment is to get the value reading sent by Arduino using Python via serial. It is also possible to visualise this data by altering the Python script that plots the values.

This experiment has built a stable and dependable serial communication system. Rather than the serial connection, any fluctuation can be traced to electrical noise in the potentiometer and wires. On both sides, this level of reliability can be attained using only basic code. Because of its simplicity, the reading has the potential for more sophisticated applications. In this experiment, we make use of the chance to depict the reading change over time on a line graph.

In conclusion, this experiment has proven to be successful and has created a window of opportunity for further exploration into the potential of this foundational understanding of serial communication. This information can be used in complex systems where using several computers and microcontrollers is typical. This is just the start of a much wider realm of communication.

Recommendations

The graphs do not accurately depict real-time data. To ensure no data is lost during reading, it is necessary to incorporate a one-second delay using an Arduino. Multithreading offers a solution, where one thread handles data reading while another is responsible for plotting and displaying the gathered information. Using this approach, the plotting and displaying of data will not impede the reading process.

References

1. *Matplotlib documentation — Matplotlib 3.8.0 documentation*. (n.d.). https://matplotlib.org/stable/index.html
2. *Welcome to pySerial's documentation — pySerial 3.4 documentation*. (n.d.). https://pyserial.readthedocs.io/en/latest/

Student's Declaration

Certificate of Originality and Authenticity

This is to certify that we are responsible for the work submitted in this report, that the original work is our own except as specified in the references and acknowledgment, and that the original work contained herein has not been untaken or done by unspecified sources or persons.

We hereby certify that this report has not been done by only one individual and all of us have contributed to the report. The length of contribution to the reports by Each individual is noted within this certificate.

We also hereby certify that we have read and understand the content of the total report and no further improvement on the reports is needed from any of the individual contributor to the report.

We, therefore, agreed unanimously that this report shall be submitted for marking and this final printed report has been verified by us

Signature: *arifsyazwan*	Read✔️
Name: Muhammad Arif Syazwan Bin Mohd Rizal	Understand ✔️
Matric Number:2110765	Agree ✔️
Signature: *haziqhaikal*	Read ✔️
Name: Muhammad Haziq Haikal bin Suaib	Understand ✔️
Matric Number: 2112969	Agree ✔️
Signature: *amirul*	Read ✔️
Name: Muhammad Amirul Syafiq Bin Ruslani	Understand✔️
Matric Number: 2115207	Agree✔️
Signature: *Wafi*	Read ✔️
Name: Muhammad Fadhlul Wafi Bin Ahmad Naim	Understand ✔️
Matric Number: 2116281	Agree ✔️
Signature: *faridjafri*	Read ✔️
Name: Muhammad Farid Bin Jafri	Understand ✔️
Matric Number: 2111633	Agree ✔️