Lab session week 6: IoT Programming with sensors and databases

Aim

The aim of this tutorial is for students to be able to confidently connect the Arduino board via serial communication to the edge server, and to store data from the sensors to the edge server database. Finally, this tutorial asks students to work with the timer interruption in Arduino.

Important Information

- Please do not power the sensor nodes (Arduino) before getting the confirmation from your tutor.
- Please use your own laptop.
- If your laptop does not have a USB type A socket, you will need to bring an adaptor.
- Be gentle with the hardware.
- You can work with a physical Raspberry Pi board or a virtual desktop to complete this task.
- Do not connect your laptop or a physical Raspberry Pi via Ethernet to the Swinburne network.
- If you are using a physical Raspberry Pi, you will need to find a way to connect it to your laptop without using the Swinburne network.

Task 1: Reading data from the sensor

Write a program that reads the sensor and sends data via serial communication. The sensor can be a temperature sensor such as the LM35 (Figure 1) or the potentiometer (Figure 2), make sure you connect the sensor to an analog pin (e.g., A0) of the Arduino board.

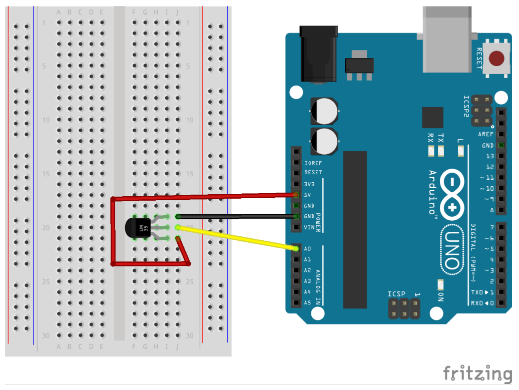


Figure 1: Wiring temperature sensor

Open the serial monitor of the Arduino IDE and check that the data is being sent.

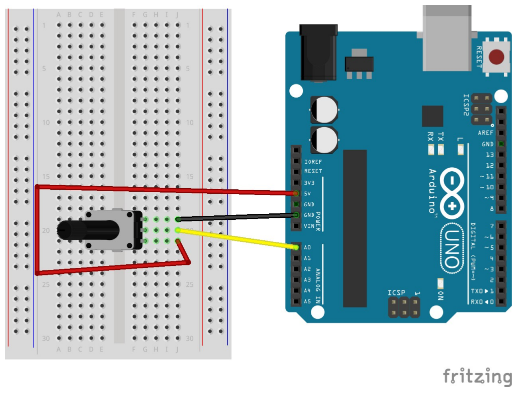
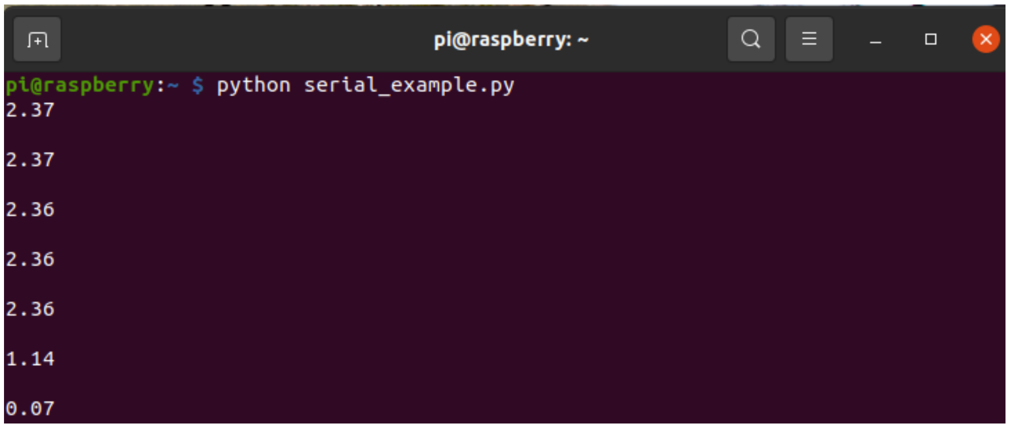


Figure 2: Wiring potentiometer (you can also refer to the figure (Task 1 Circuit Design. Jpg) given in the resources folder)

Task 2: Receiving data in the edge server

Connect via USB cable the Arduino board to the Raspberry Pi (or your computer if you are using the virtual desktop Raspberry Pi OS). Code a Python program that prints the information received via serial bus. The information received via serial bus should be printed in the terminal when running the python program. *Refer to the video file "Task 2 Demo. mp4" given in the resources folder.*



```
pi@raspberry:~ $ python serial_example.py
2.37

2.37

2.36

2.36

2.36

1.14

0.07
```

Figure 3: Python program printing serial data

Task 3: Storing data in a database

Now that the connection of Arduino with the edge server has been established, it's time to get ready the database. *Refer to the video file "Task 3 Demo. mp4" given in the resources folder.* First of all, we need to install MySQL in the edge server. Given that MySQL is not includedin the Debian repository, we can install MariaDB. MariaDB is a fork of MySQL maintained by the com-munity and compatible with MySQL. Make sure the edge server has internet connection before trying to install the database executing the following commands:

```
sudo apt-get install mariadb-server
sudo apt-get install python-pymysql python3-pymysql
```

Once everything has been installed, we need to grant access to our user. Accessing as a root should be avoided as much as possible.

```
mysql
ERROR 1698 (28000): Access denied for user 'pi'@'localhost'
```

We will enter as admin and execute the following:

```
sudo mysql
CREATE USER 'pi'@'localhost';
CREATE DATABASE temperature_db;
GRANT ALL PRIVILEGES ON temperature_db.* TO 'pi'@'localhost';
exit
```

We created the temparature_db database, the username pi without password, and we granted permissions to pi to edit the database. We will open again mysql as the pi user to create a table in the temperature_db database:

```
mysql
USE temperature_db;
CREATE TABLE tempLog ( tempId int(11) AUTO_INCREMENT NOT NULL,
                       temperature VARCHAR(20) NOT NULL,
                       PRIMARY KEY (TempId) );
DESCRIBE tempLog;
exit
```

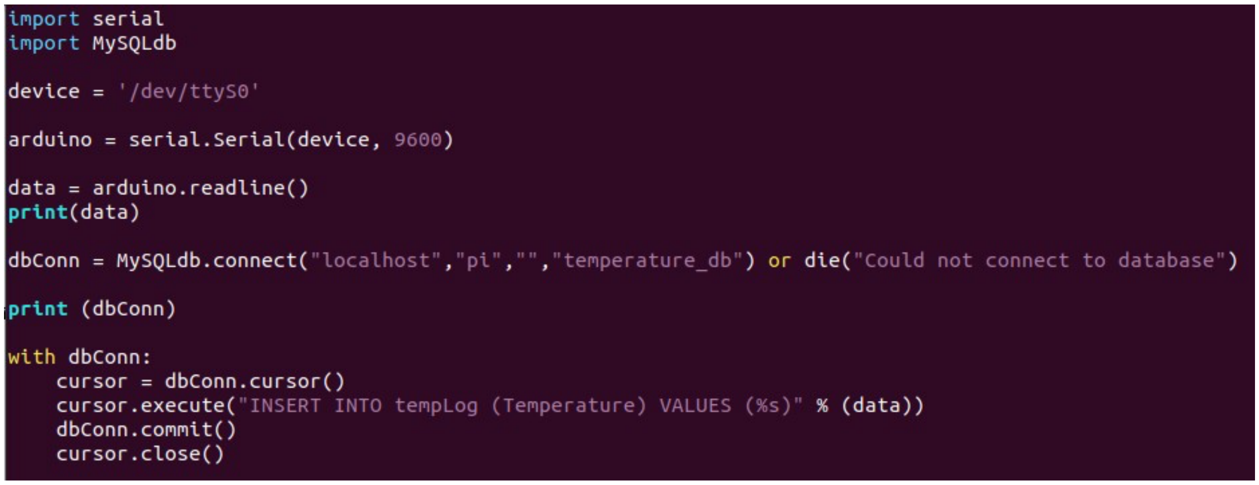
The final step is to modify your previous code to store data in the database. Your final code should be similar to the code shown in Figure 4. However, you need to modify it in order to store all the data being received via serial bus. WARNING: The code shown in Figure 4 imports MySQLdb, you need to import pymysql. Please check PyMySQL user guide provided in the resources section to modify your code accordingly.

You can check if the data is being stored in the database from mysql:

```
mysql
USE temperature_db;
SELECT * FROM tempLog;
```

Task 4: Reading data every second

Now that you have successfully connected the Arduino board to the database of the edge server, add a timer using interruptions that sends every 5 seconds data from the sensor and stores it in the database. *Refer to the video file "Task 4 Demo. mp4" given in the resources folder.*



```python
import serial
import MySQLdb

device = '/dev/ttyS0'

arduino = serial.Serial(device, 9600)

data = arduino.readline()
print(data)

dbConn = MySQLdb.connect("localhost","pi","","temperature_db") or die("Could not connect to database")

print (dbConn)

with dbConn:
    cursor = dbConn.cursor()
    cursor.execute("INSERT INTO tempLog (Temperature) VALUES (%s)" % (data))
    dbConn.commit()
    cursor.close()
```

Figure 4: Code to get serial data and store it in the database

Resources

Arduino

- Arduino Language Reference: https://www.arduino.cc/reference/en/
- Arduino Foundations: https://www.arduino.cc/en/Tutorial/Foundations
- Arduino Built-In Examples: https://www.arduino.cc/en/Tutorial/BuiltInExamples

Raspberry Pi

- https://www.raspberrypi.org/

GNU/Linux

- Linux Journey is a site dedicated to making learning Linux fun and easy. https://linuxjourney.com/
- Introduction to Linux: A Hands on Guide. https://tldp.org/LDP/intro-linux/intro-linux.pdf
- Introduction to Linux (LFS101), the Linux Foundation training course: https://training.linuxfoundation.org/training/introduction-to-linux/

Database

- MariaDB Server Documentation. https://mariadb.com/kb/en/documentation/
- MySQL 8.0 Reference Manual. https://dev.mysql.com/doc/refman/8.0/en/

Python

- Python for Beginners (Programmers). https://www.python.org/about/gettingstarted/
- PyMySQL user guide. https://pymysql.readthedocs.io/en/latest/