**Task 1: Writing a Python program on RPi to communicate with Analog Discovery board via UART**

1-UART stands for Universal Asynchronous Receiver Transmitter. It is a serial communication protocol that allows two devices to exchange data. In this task, you will write a Python program on a Raspberry Pi to communicate with an Analog Discovery board via UART.

To do this, you will need to follow these steps:

Enable UART on the Raspberry Pi. You can do this by using the raspi-config tool. Open a terminal window and type the following command: **sudo raspi-config**

Navigate to the "Interfacing Options" menu and select "Serial". Choose "Enable" and then select "Finish".

2-Install the WiringPi library. The WiringPi library is a popular library for GPIO programming on Raspberry Pis. It provides functions for accessing and controlling GPIO pins, which is important for UART communication. You can install the WiringPi library by using the following command: sudo apt-get install wiringpi

3-Write the Python program. The Python program will need to open the UART port, read or write data to the Analog Discovery board, and then close the UART port. Here is an example of how to do this:

**import wiringpi**

**# Define the GPIO pins to use**

**uart\_rx\_pin = 10 # Receive pin**

**uart\_tx\_pin = 8 # Transmit pin**

**# Initialize WiringPi**

**wiringpi.wiringPiSetupGpio()**

**# Set the pins as input and output**

**wiringpi.pinMode(uart\_rx\_pin, wiringpi.INPUT)**

**wiringpi.pinMode(uart\_tx\_pin, wiringpi.OUTPUT)**

**# Open the UART port at 9600 baud**

**uart = wiringpi.Serial(uart\_rx\_pin, uart\_tx\_pin, 9600)**

**# Read data from the UART port**

**data = uart.read()**

**# Print the data**

**print(data)**

**# Close the UART port**

**uart.close()**

This program will read data from the UART port and print it to the console. You can modify this program to send data to the Analog Discovery board by using the write() method of the Serial object

**Task 2: Choose one sensor, either analog or digital, and connect it to RPi.**

In this task, you will choose an analog or digital sensor and connect it to the Raspberry Pi. You will then write a Python program to read data from the sensor and display it on the console.

<https://github.com/Seeed-Studio/grove.py/tree/master/grove>

**Task 3: Read data from the chosen sensor and store it in a file (type and extension of your choice).**

import wiringpi

import datetime

import csv

# Define the GPIO pins to use

temp\_sensor\_pin = 0 # Connect the temperature sensor to GPIO pin 0

# Initialize WiringPi

wiringpi.wiringPiSetupGpio()

# Set the pin as an input

wiringpi.pinMode(temp\_sensor\_pin, wiringpi.INPUT)

# Open the CSV file for writing

with open('temperature\_data.csv', 'w') as csvfile:

writer = csv.writer(csvfile)

# Write the header row

writer.writerow(['Date', 'Time', 'Temperature'])

# Read data from the temperature sensor and write it to the CSV file

while True:

# Read the raw temperature value

temperature\_raw = wiringpi.analogRead(temp\_sensor\_pin)

# Convert the raw value to temperature in degrees Celsius

temperature\_celsius = temperature\_raw \* 0.00392156862745

# Convert the temperature to degrees Fahrenheit

temperature\_fahrenheit = temperature\_celsius \* 1.8 + 32

# Get the current date and time

date = datetime.datetime.now().strftime('%Y-%m-%d')

time = datetime.datetime.now().strftime('%H:%M:%S')

# Write the data to the CSV file

writer.writerow([date, time, temperature\_celsius, temperature\_fahrenheit])

# Wait for 1 second before reading the temperature again

time.sleep(1)

This program will continuously read the temperature from the sensor and store the data in a CSV file called temperature\_data.csv. The file will have columns for the date, time, and temperature in both Celsius and Fahrenheit.

**Task 4: Send the file to your partner's device (Analog Discovery) and display it using simple GUI**

In this task, you will send the file you created in Task 3 to your partner's Analog Discovery board and display the data using a simple graphical user interface (GUI).

1. **Establish UART communication with Analog Discovery**

Use the UART communication you set up in Task 1 to send the file to the Analog Discovery board. You can do this by using the write() method of the Serial object to send the contents of the CSV file to the Analog Discovery board.

1. **Write the GUI program**

Create a GUI program on your partner's computer that can receive the file from the Analog Discovery board and display the data on the screen. You can use a GUI library like PyQt5 or Tkinter.

1. **Receive the file and display the data**

The GUI program should receive the file from the Analog Discovery board and display the data in a visually appealing format, such as a graph, chart, or table.

Here is an example of how to receive the file from the Analog Discovery board and display it in a graph: import pyqtgraph as pg

import serial

import csv

# Connect to the Analog Discovery board

ser = serial.Serial('COM3', 9600)

# Create a new plot window

app = pg.QtGui.QApplication([])

win = pg.GraphicsWindow()

win.show()

# Create a plot

p = win.addPlot()

# Read data from the Analog Discovery board

while True:

# Read the file from the serial port

data = ser.readline().decode('utf-8').strip()

# Parse the data into a list of timestamps and values

timestamps, values = zip(\*csv.reader(data.split(',')))

# Convert the timestamps to datetime objects

timestamps = [datetime.datetime.strptime(ts, '%Y-%m-%d %H:%M:%S') for ts in timestamps]

# Add the data to the plot

p.plot(timestamps, values)

# Update the plot

p.repaint()

# Wait for 1 second before reading the data again

time.sleep(1)