

PROJ 201 Project Final Report

## **Lab Monitoring with IoT via Raspberry Pi 3**

Deha Doğan 28065  
Emir Balkan 27787  
Hatice Merve Vural 27903

Supervised by: Melih Türkseven

29.12.2021

## **Abstract**

Internet of Things (IoT) is a predetermined system that links objects which receive data from the environment at which the system is built, with the system's decision algorithm through the Internet. In our case, we have decided upon creating an IoT system using a Raspberry Pi and a temperature sensor. We have developed a software that makes the transmission of the physical data to be transferred to an Internet database as a sequence of synchronous digital data possible.

In order to completely construct a so-called IoT system, we have decided to interconnect a temperature sensory device that synchronously sends digital data through a communication protocol called I<sup>2</sup>C to Raspberry Pi 3, which is a device that is responsible for processing the temperature data obtained through I<sup>2</sup>C every 500 milliseconds and hence sending the temperature values to an Internet interface called "Initial State" as a result of the pre-integrated Python code which was put into the Raspberry Pi itself. Therefore, through the integration of such an instance of an IoT system by following the steps mentioned above, our group members are going to be able to develop IoT systems alike in the future within the topics of communication, embedded electronics and hardware coding.

## Introduction

To completely establish a system that processes physical data into binary data, it is required to bring two main types of devices together. While one type of a device takes part as a “master”, the other one is going to be used as a “slave” (Sfuptownmaker, 2013). The master (controller) device is the Raspberry Pi 3, and the slave (peripheral) device is a digital thermometer called DS18B20. Raspberry Pi 3 is a controller which allows its developers to implement Python codes through an API into itself. The role of the interface called Initial State is to display the output that is sent by the web-based algorithm that processes the raw data which is sent from the Raspberry Pi 3 every 500 milliseconds. As a result of the implementation of these components which add up to an IoT system, remote monitoring of the Lab environment is going to be achieved through the transmission of the physical data (temperature of the Lab environment) into the processed digital data (the graph of the temperature values versus time) synchronously.

# **Materials & Methods**

## **1. Materials**

In order to accomplish our project, we have used:

- Raspberry Pi 3
- Micro-SD card with Raspbian Operating System
- Male to male Jumper
- Raspberry Pi power supply
- DS18B20 temperature sensor
- Monitor
- Computer
- MicroSD card reader
- Python 3
- Youtube
- Initial State account

## **2. Methods**

### **a. Setting up the Raspberry Pi**

First, we have downloaded the Raspbian Operating system image to our microSD via microSD card reader. It was downloaded from the official website of Raspberry Pi (Raspberrypi, n.d.). After that, we have inserted the microSD card into the Raspberry Pi 3.

Through plugging the Pi into power (5V) and connecting a monitor, we have successfully began utilizing the Raspberry Pi.

## **b. Connecting the Sensors**

In order to connect sensors to raspberry, we have watched tutorials about how to connect sensors to the correct pins (Circuit Basics, 2016). We connected ground to 6th pin, data to 7th pin and power to 1st pin as it can be seen in Appendix E. Then, we have written our codes to the terminal which we have accessed the Raspberry Pi from our computer via SSH connection (sentdex, 2017). In order to complete this step, we have enabled SSH from the configuration menu on the Raspberry Pi. Besides, we have also enabled I<sup>2</sup>C and 1-Wire. It can be seen how to enable them in Appendix F.

## **c. Codes**

Firstly, from the tutorial of Circuit Basics (2016) we edited config.txt by writing this command to the terminal: **sudo nano /boot/config.txt**, and added **dtoverlay=w1-gpio** to the bottom of the file. This step can be seen in Appendix H and I. Since we edited the Raspberry, we have rebooted it. Then **sudo modprobe w1-gpio** and **sudo modprobe w1-therm** was written to the terminal. Following this step, we entered **cd /sys/bus/w1/devices** in order to change directory and entered **ls** to list the devices. These steps are shown in Appendix F. If 28-XXXXXXXXXXXX is listed as the way it was listed in Appendix K, then it indicated that we have connected the sensor to the correct pins. By this interpretation, we continued to enter **cd 28-XXXXXXXXXXXX** and **cat w1\_slave**. We saw t=XXXX which was the temperature without decimal that the sensor has measured. In order to monitor the temperature in Celsius

and Fahrenheit, we have written a simple Python program, which can be seen in Appendix G, and we have run it with **sudo python temp2.py**. The last steps can be seen in Appendix L.

#### **d. Transferring Data to the Network**

We have used the website called Initial State to proctor the data i.e temperature changes in a live manner. We wrote IP address of raspberry pi to this website. Last step was open a .py file, wrote necessary program which can be seen in Appendix I. This file formed a data bucket for Initial State. Then, we started to proctor data from Initial State (InitialStateTech, 2019). In order to find IP address of raspberry, we entered **hostname -I** to the terminal and output was the IP address.

## **Result**

As a result, it was proved that a temperature sensor as a peripheral which is connected to Raspberry Pi 3 through I<sup>2</sup>C pinouts (see Figure 1.a) on Pi via jumper cables can transfer data to internet. In this case, it is possible to monitor temperature data from places where have stable wireless connection, which is the goal of the experiment (see Figure 1.b). As a concrete example

of IoT (Internet of Things) application, data obtained from the temperature sensor via the Raspberry Pi 3 shows that any peripheral which has I<sup>2</sup>C pins for connection can transfer the data to user through compatible Python code in Pi.

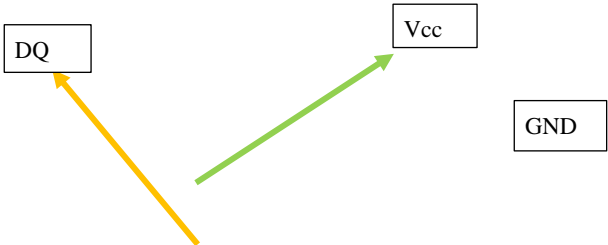
In this experiment, initial data was raw (i.e. unprocessed by Python code in Raspberry Pi's system), and recent data was processed and transferred to internet by Python code (see Picture 1.a). Therefore, there is no limit for processing data, in other words, Python code with libraries can be written as a multi-processing by using more than one peripheral.

## **Discussion and Conclusion**

Results obtained from the system shows that Raspberry Pi as a master of the system has a powerful feature to constitute a system which can collect data. Every component of the system, including keyboard, mouse, monitor, sensor and countless devices have wi-fi, are connected to Pi. Furthermore, it provides to store Python code to process data which is written by user. Because the Pi is considered as the center of the system, it can be embedded to any robotic system to run it for a specific purpose. From music machines to weather stations, Pi's have been using. In this experiment, A small system is designed to show basic structure of Pi and taught how to monitor data via Pi as an IoT application. In this sense, it is significant to understand the basic system to make it complicated and more functionalized. Thus, this experiment can be light to future robotic systems for everyone who understands the experiment.

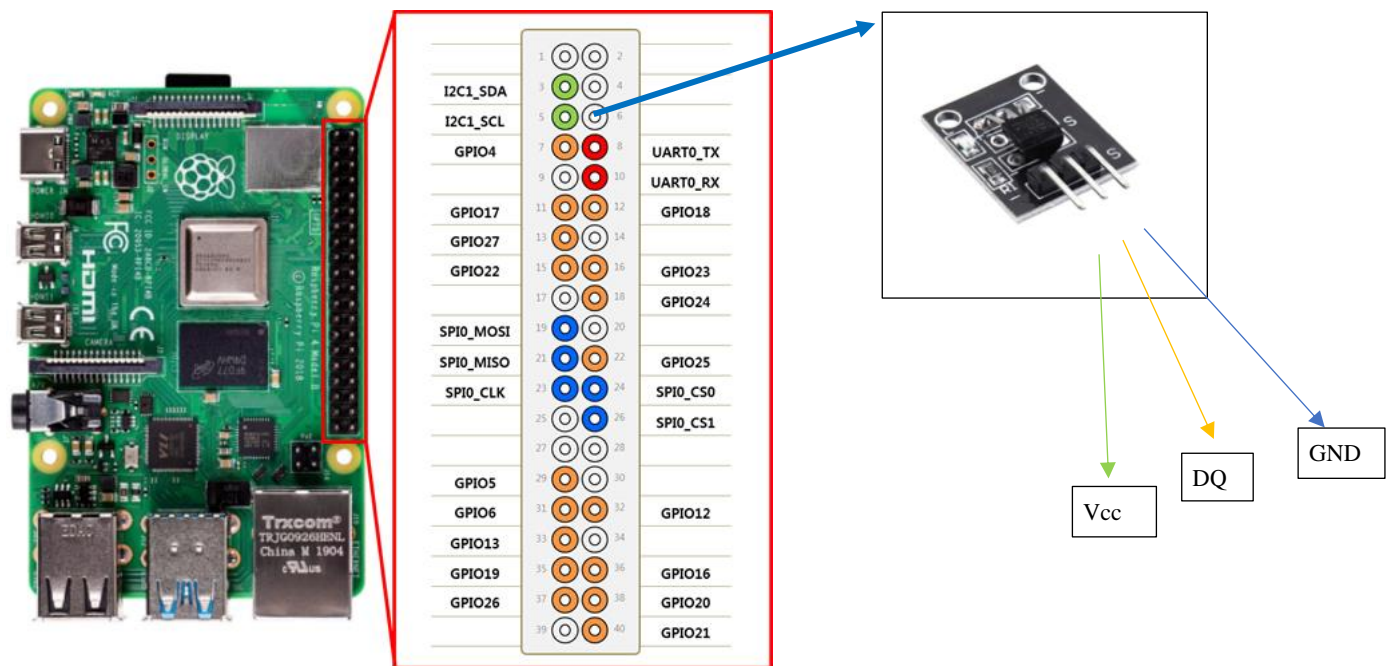
**APPENDICES**

**Appendix A**



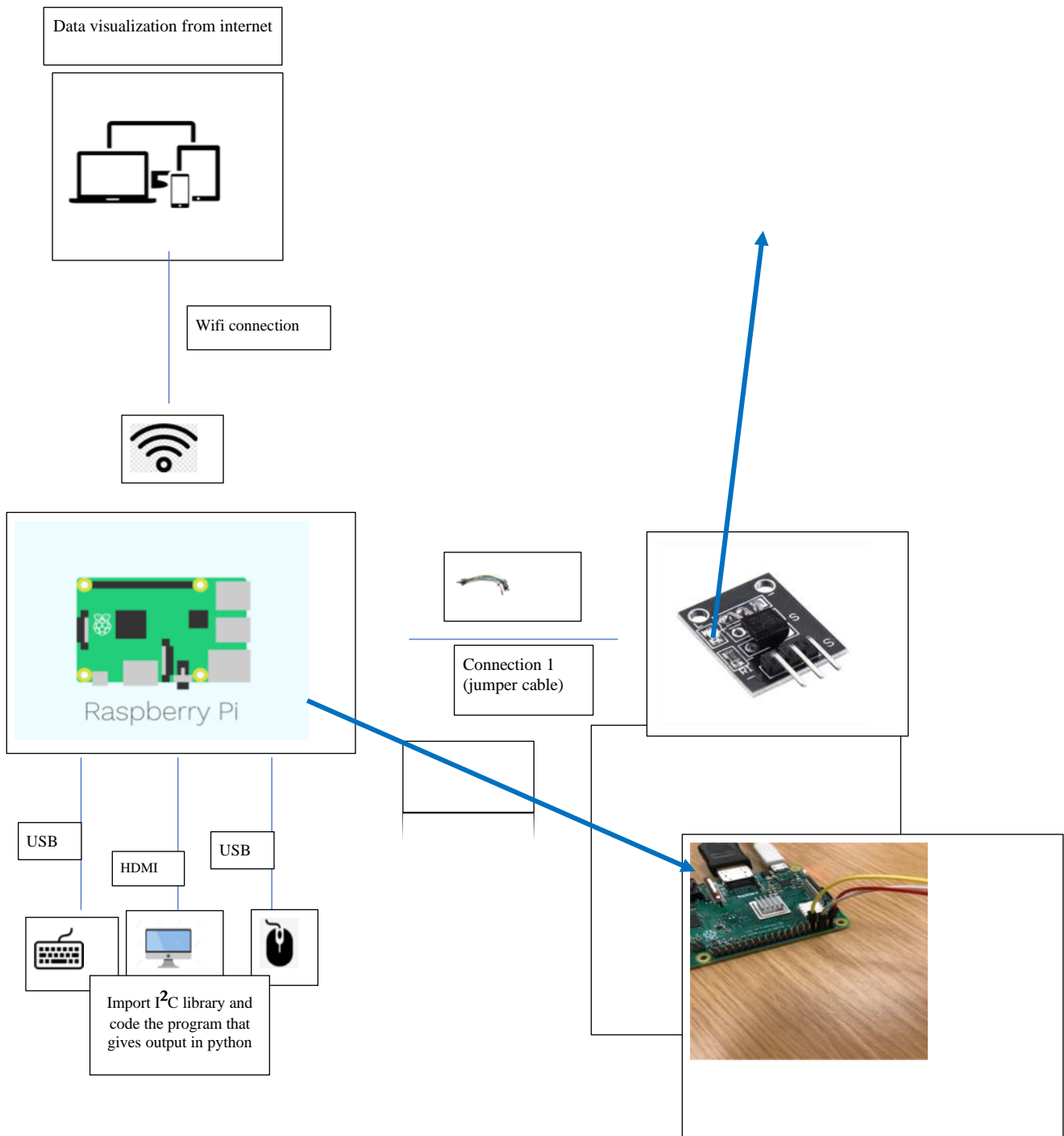
**Appendix B**



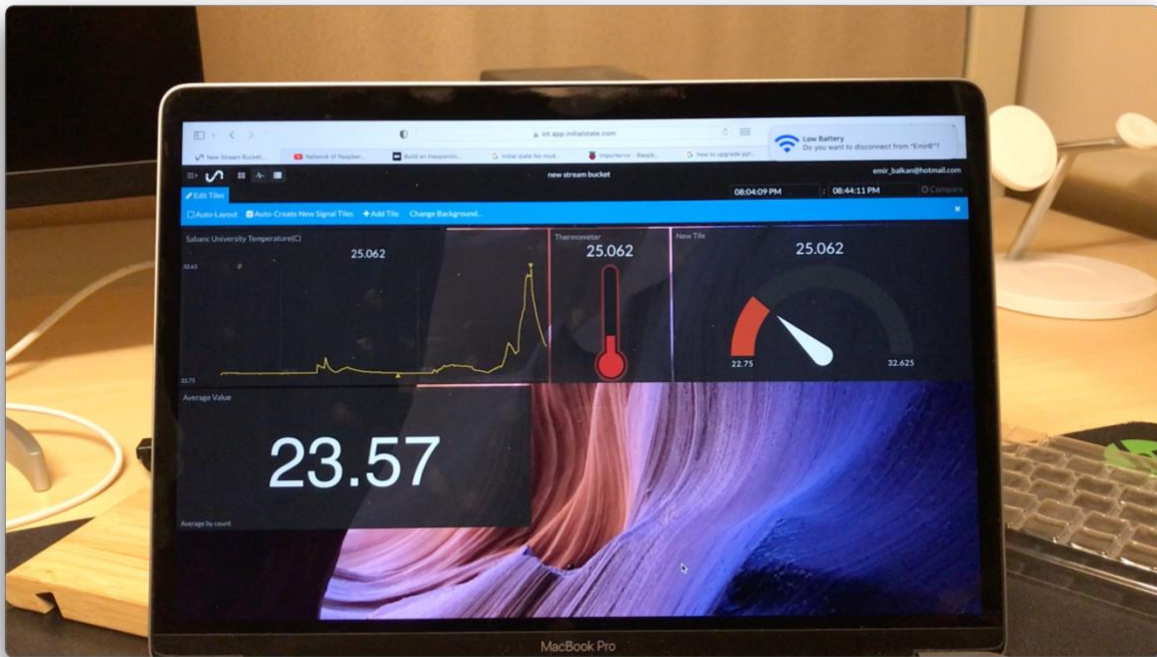


## Appendix C





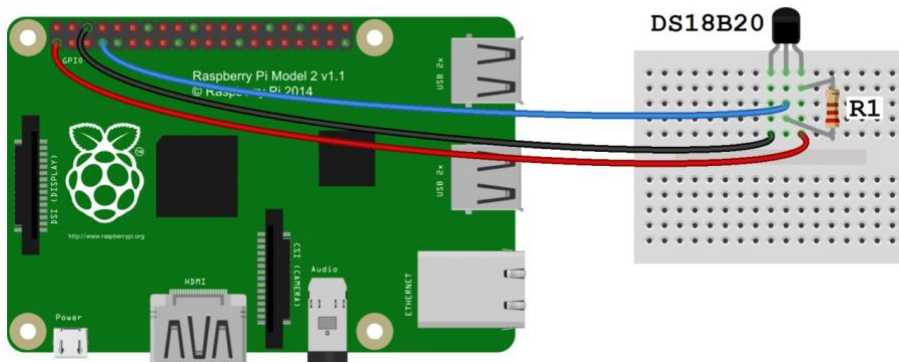
## Appendix D



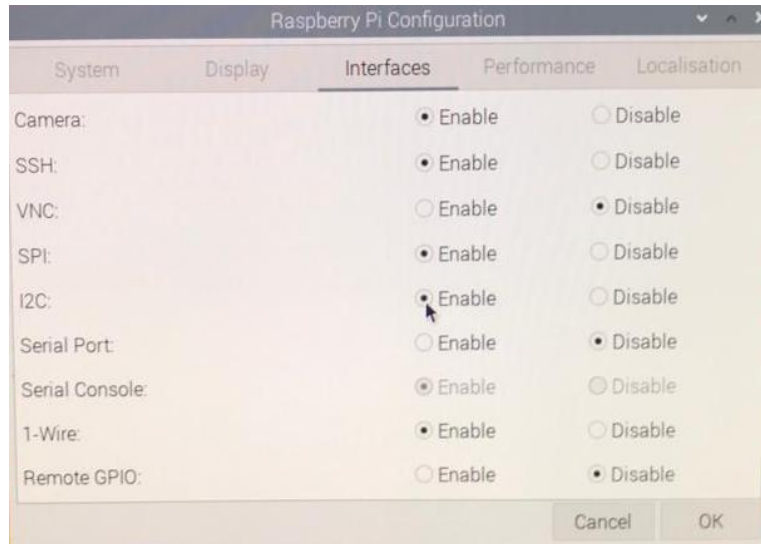
Visualization of Temperature data through IoT (Internet of Things)

(You can access last data which is obtained via <https://go.init.st/uvsszlp> by signing in)

## Appendix E



## Appendix F



## Appendix G

```
import os
import glob
import time

os.system('modprobe w1-gpio')
os.system('modprobe w1-therm')

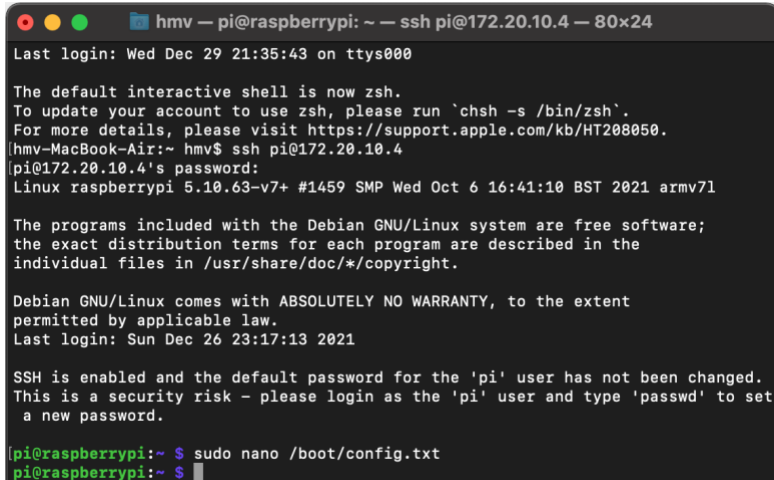
base_dir = '/sys/bus/w1/devices/'
device_folder = glob.glob(base_dir + '28*')[0]
device_file = device_folder + '/w1_slave'

def read_temp_raw():
    f = open(device_file, 'r')
    lines = f.readlines()
    f.close()
    return lines

def read_temp():
    lines = read_temp_raw()
    while lines[0].strip()[-3:] != 'YES':
        time.sleep(0.2)
        lines = read_temp_raw()
    equals_pos = lines[1].find('t=')
    if equals_pos != -1:
        temp_string = lines[1][equals_pos+2:]
        temp_c = float(temp_string) / 1000.0
        temp_f = temp_c * 9.0 / 5.0 + 32.0
        return temp_c, temp_f

while True:
    print(read_temp())
    time.sleep(1)
```

## Appendix H



```
hmv — pi@raspberrypi: ~ — ssh pi@172.20.10.4 — 80x24
Last login: Wed Dec 29 21:35:43 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
hmv-MacBook-Air:~ hmv$ ssh pi@172.20.10.4
pi@172.20.10.4's password:
Linux raspberrypi 5.10.63-v7+ #1459 SMP Wed Oct 6 16:41:10 BST 2021 armv7l

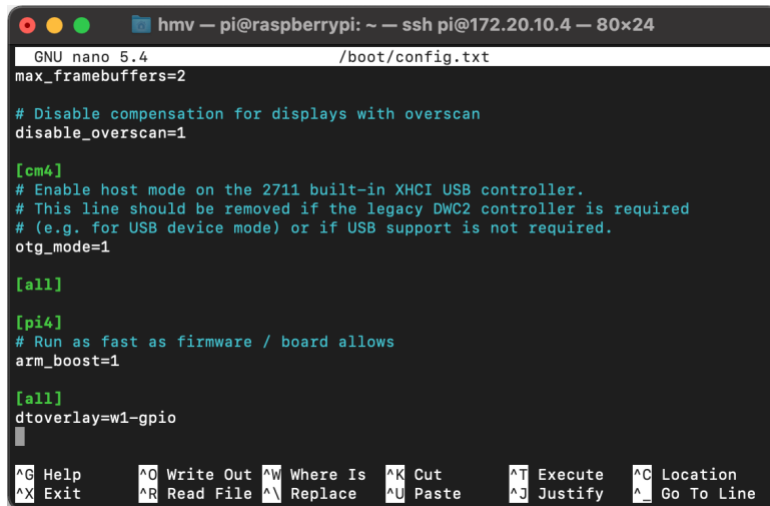
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Dec 26 23:17:13 2021

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

pi@raspberrypi:~ $ sudo nano /boot/config.txt
pi@raspberrypi:~ $
```

## Appendix I



The screenshot shows a terminal window with a title bar indicating an SSH session to a Raspberry Pi. The nano text editor is open, editing the file /boot/config.txt. The content of the file includes configuration for framebuffers, overscan, USB controller, and ARM boost. The bottom of the window displays a row of keyboard shortcuts for nano.

```
hmv — pi@raspberrypi: ~ — ssh pi@172.20.10.4 — 80x24
GNU nano 5.4 /boot/config.txt
max_framebuffers=2

# Disable compensation for displays with overscan
disable_overscan=1

[cm4]
# Enable host mode on the 2711 built-in XHCI USB controller.
# This line should be removed if the legacy DWC2 controller is required
# (e.g. for USB device mode) or if USB support is not required.
otg_mode=1

[all]

[pi4]
# Run as fast as firmware / board allows
arm_boost=1

[all]
dtoverlay=w1-gpio

```

<b>^G</b> Help	<b>^O</b> Write Out	<b>^W</b> Where Is	<b>^K</b> Cut	<b>^T</b> Execute	<b>^C</b> Location
<b>^X</b> Exit	<b>^R</b> Read File	<b>^_\</b> Replace	<b>^U</b> Paste	<b>^J</b> Justify	<b>^_</b> Go To Line

## Appendix J

```
hmv — pi@raspberrypi: /sys/bus/w1/devices — ssh pi@172.20.10.4 — 80x...
For more details, please visit https://support.apple.com/kb/HT208050.
hmv-MacBook-Air:~ hmv$ ssh pi@172.20.10.4
pi@172.20.10.4's password:
Linux raspberrypi 5.10.63-v7+ #1459 SMP Wed Oct 6 16:41:10 BST 2021 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Dec 26 23:17:13 2021

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

pi@raspberrypi:~ $ sudo nano /boot/config.txt
pi@raspberrypi:~ $ sudo modprobe w1-gpio
pi@raspberrypi:~ $ sudo modprobe w1-therm
pi@raspberrypi:~ $ cd /sys/bus/w1/devices
pi@raspberrypi:/sys/bus/w1/devices $
```

## Appendix K

```
pi@raspberrypi:~ $ sudo modprobe w1-gpio
pi@raspberrypi:~ $ sudo modprobe w1-therm
pi@raspberrypi:~ $ cd /sys/bus/w1/devices
pi@raspberrypi:/sys/bus/w1/devices $ ls
28-8000002b2b6b w1_bus_master1
pi@raspberrypi:/sys/bus/w1/devices $
```

## Appendix L

```
pi@raspberrypi:/sys/bus/w1/devices $ cd 28-8000002b2b6b
pi@raspberrypi:/sys/bus/w1/devices/28-8000002b2b6b $ cat w1_slave
c5 01 ff ff 7f ff ff ff 7b : crc=7b YES
c5 01 ff ff 7f ff ff ff 7b t=28312
pi@raspberrypi:/sys/bus/w1/devices/28-8000002b2b6b $ cd
pi@raspberrypi:~ $ sudo python temp2.py
1 27.4
2 27.3
3 27.2
4 27.1
5 71.2
6 76.3
7 96.4
8 87.3
9 72.9
10 63.5
11 57.0
12 52.3
13 48.8
14 46.0
15 43.9
16 42.1
17 40.5
18 39.1
19 38.1
20 37.2
21 36.4
```

## Appendix M

```
GNU nano 5.4
import Adafruit_DHT
from ISStreamer.Streamer import Streamer
import time

# ----- User Settings -----
SENSOR_LOCATION_NAME = "Sabanci University"
BUCKET_NAME = ":partly_sunny: Room Temperatures"
BUCKET_KEY = "3EG3N3SV3CTL"
ACCESS_KEY = "ist__FSTs_iEHORF15_ptwRY8bbuFXmn9tIE"
MINUTES_BETWEEN_READS = 10
METRIC_UNITS = True
# -----
streamer = Streamer(bucket_name=BUCKET_NAME, bucket_key=BUCKET_KEY, access_key=ACCESS_KEY)

DS18B20="/sys/bus/w1/devices/28-8000002b2b6b/w1_slave"

r = 0

while True:

    r += 1

    f = open(DS18B20, "r")
    data = f.read()
    f.close()

    (discard, sep, reading) = data.partition(' t=')

    temp_c = float(reading) / 1000.0

    streamer.log(SENSOR_LOCATION_NAME + " Temperature(C) ", temp_c)

    streamer.flush()
    time.sleep(0.5)
```



## References

Circuit Basics. (2016, April 30). *Raspberry Pi DS18B20 Temperature Sensor Tutorial [Video]*.

YouTube. <https://www.youtube.com/watch?v=aEnS0-Jy2vE>

InitialStateTech. (2019, September 9). *Network of Raspberry Pi Temperature Sensors / GIT*

*TECH D [Video]*. YouTube. <https://www.youtube.com/watch?v=XWRQdPAqEIU>

Jimblom. (2012, December 18). Serial Communication. Retrieved October 30, 2021, from

<https://learn.sparkfun.com/tutorials/serial-communication>

Raspberrypi. (n.d.) . *Operating System Images*. <https://www.raspberrypi.com/software/operating-systems/>

*Raspberry Pi DS18B20 Temperature Sensor Tutorial*. 2021, December 29). CircuitBasics.

Retrieved December 29, 2021, from <https://www.circuitbasics.com/raspberry-pi-ds18b20-temperature-sensor-tutorial/>

Sentdex. (2017, April 3). *Remote Access with SSH and Remote Desktop - Raspberry Pi and Python tutorials p.3 [Video]*. YouTube.

<https://www.youtube.com/watch?v=IDqQIDL3LKg&list=LL&index=3&t=288s>

Sfuptownmaker. (2013, July 8). I2C. Retrieved October 30, 2021, from

<https://learn.sparkfun.com/tutorials/i2c/all#introduction>.

Wikimedia Foundation. (2021, October 30). Raspberry Pi. Wikipedia. Retrieved October 30, 2021, from [https://en.wikipedia.org/wiki/Raspberry\\_Pi](https://en.wikipedia.org/wiki/Raspberry_Pi).