IoT Firmware on Micropyhton Framework with Environmental Sensors and Network Functionalities

Major Project Report

Submitted in fulfillment of the requirements for the degree of

Master of Technology
in
Electronics & Communication Engineering
(Embedded Systems)

By

Devang Sharma

(16MECE22)

Electronics & Communication Engineering Department
Institute of Technology
Nirma University
Ahmedabad-382 481
May 2018

IoT Firmware on Micropyhton Framework with Environmental Sensors and Network Functionalities

Major Project Report

Submitted in partial fulfillment of the requirements

for the degree of

Master of Technology

in

Electronics & Communication Engineering

By

Devang Sharma
(16MECE22)

Under the guidance of

External Project Guide:	**Internal Project Guide:**
Mr. Sohil Patel	**Dr. N.P. Gajjar**
CTO	PG Coordinator, Embedded Systems,
Oizom Instruments Pvt. Ltd.,	Institute of Technology,
Ahmedabad.	Nirma University, Ahmedabad.

Electronics & Communication Engineering Department

Institute of Technology, Nirma University

Ahmedabad-382 481

May 2018

Declaration

This is to certify that

a. The thesis comprises my original work towards the degree of Master of Technology in Embedded Systems at Nirma University and has not been submitted elsewhere for a degree.

b. Due acknowledgment has been made in the text to all other material used.

- Devang Sharma
16MECE22

Disclaimer

NIRMA UNIVERSITY
INSTITUTE OF TECHNOLOGY
NAAC ACCREDITED 'A' GRADE

Certificate

This is to certify that the Major Project entitled **"IoT Firmware on Micropyhton Framework with Environmental Sensors and Network Functionalities"** submitted by **Devang Sharma (16MECE22)**, towards the partial fulfillment of the requirements for the degree of Master of Technology in Embedded Systems, Nirma University, Ahmedabad is the record of work carried out by him under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination.The results embodied in this major project, to the best of our knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Date: Place: Ahmedabad

Dr. N.P.Gajjar
Internal Guide

Program Coordinator

Dr. D. K. Kothari
Section Head, EC

Dr. Alka Mahajan
Director, IT

OIZOM™
REDEFINING RESOURCES

Certificate

This is to certify that the Major Project entitled **"IoT Firmware on Micropyhton Framework with Environmental Sensors and Network Functionalities"** submitted by **Devang Sharma (16MECE22)**, towards the partial fulfillment of the requirements for the degree of Master of Technology in Embedded Systems, Nirma University, Ahmedabad is the record of work carried out by him under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination.

Mr. Sohil Patel
CTO
Oizom Instruments Pvt. Ltd.
Ahmedabad

Acknowledgements

I would like to express my gratitude and sincere thanks to my internel project guide **Dr.N.P.Gajjar**, PG Coordinator of M.Tech Embedded Systems, for his constant support and guidance during the review process.

I would also like to thank **Mr. Sohil Patel**, external guide of my internship project from **Oizom Instruments Pvt. Ltd.**, for guidance, monitoring and encouragement regarding the project.

- Devang Sharma
16MECE22

Contents

List of Figures

Abstract

The air pollution is increasing along with the growth of cities and industries everyday. The quality of the air needs to be known and thus, to be measured. The device measures the environmental parameters such as Temperature, Humidity, Pressure, Light, Dust Particulate Matters, Carbon Dioxide and Carbon Monoxide using accurate, sustainable and reliable digital sensors. This device is designed to work in constrained embedded system which is achieved by integrating the sensors with low power micro-controller. Firmware of this device is built on the lightweight MicroPython framework. The device runs faster with the compact hardware and less internal memory usage. Moreover, Long Range(LoRa) technology is integrated to enable communication with low power and wider range between remote sensors and gateway connected to the network. The sensor data is sent to network server and is observed using various graphical representations on the web application.

Abbreviation Notation and Nomenclature

IoT	Internet of Things
LoRa	Long Range
LPWAN	Low Power Wide Area Network
LED	Light Emitting Diode
BLE	Bluetooth Low Energy
AES	Advanced Encryption Standard
UART	Universal Asynchronous Receiver Transmitter
I2C	Inter-Integrated Circuit
SPI	Serial Peripheral Interface
REPL	Read-Evaluate-Print Loop
URL	Uniform resource Locator
V (unit)	Voltage
MHz (unit)	MegaHertz
ug (unit)	microgram
ppm (unit)	parts per million

Chapter 1

Introduction

1.1 Motivation

The industrialism is growing rapidly in India which increases the air pollution. It has become a necessity to know the quality of the air that we breathe everyday. The device is designed to measure the air quality data accurately with minimal of memory and power usage. The device is able to transmit fetched data over long distances. It can also store the data on the server for future analysis.

1.2 Objective

The objective of the project is to integrate the various sensors to the device controller using serial communication protocols and to transmit the gathered data wirelessly to the remotely located gateway or application server using different communication technologies and networking protocol. The project fulfills the device management requirements for IoT(Internet of Things) such as authentication, configuration, control, monitoring and software maintenance.

1.3 Scope of Work

To build a firmware for the device controller(LoPy) such that the device should fetch sensor data continuously and send the average of the accumulated sensor data to the application server periodically. The device should show its current status using device status LEDs. It should handle all communication and networking protocols effectively. The data fetched should be monitored and maintained securely over the internet. Furthermore, the fundamental requirements for the IoT device management should be included.

1.4 Outline of Thesis

The outline of my thesis consists of seven chapters. Chapter 1 describes motivation, objective and scope of the project. Chapter 2 describes about the literature survey carried out during the project. Chapter 3 illustrates the hardware specifications of the micro controller and sensors. It also describes block diagram of the system. Chapter 4 gives details of software design flow, programming language and IDE(Integrated Development Environment) features. Chapter 5 explains about the detailed project work done at the company. Chapter 6 shows distinct issues faced during the project and their solution. Chapter 7 gives the conclusion of the project.

Chapter 2

Literature Survey

2.1 MicroPython Language Study

MicroPython is an open source programming language which runs directly on various bare-metal platforms. It provides interactive prompt to execute instructions immediately. It has the ability to run and import scripts form built-in file system. It includes some of the Python's core standard libraries. Moreover, it allows user to access low-level hardware by importing modules such as "machine". Micropython also provides advanced features such as an interactive prompt, arbitrary precision integers, generators, exception handlers and many more. Though it is compact enough to run within 16k of RAM and 256k of code space.

2.2 LoRa Technology

LoRa(**Lo**ng **Ra**nge) is a radio modulation technology used for **LPWAN**(Low-Power Wide-Area Network). It is based on CSS(Chirp Spread Spectrum) modulation for low power and high range communication. LoRa is the first implementation of CSS for commercial purposes. A single LoRa gateway can cover entire cities. It uses radio frequency bands(license-free) like **865-867 MHz(India)**, 868 MHz(Europe) and 915 MHz(USA).

Figure 2.1: LoRa Alliance[2]

2.2.1 Advantages of LPWAN

As number of IoT connected devices are increasing at significant pace, a single technology cannot serve all of the applications of IoT. Technologies such as WiFi and BLE are used for shorter ranges. Cellular technology is used for higher data throughput applications. LPWAN, on the other hand, offers very long battery life for communicating over wider ranges at low power.

List of Advantages

- Long Battery Life
- Low Power Consumption
- Long Range Coverage
- Secure Transmission
- Bi-directional
- Scalable Network
- Low Data Rate

2.3 LoRaWAN Protocol

LoRaWAN is a LPWAN specification intended for wireless battery operated things in a regional, national or global network. It is built on top of LoRa Technology by LoRa Alliance. It targets key requirements of Internet of Things such as secure bidirectional communication, mobility and localization services. It enables seamless interoperability among smart things without complex local installations. It follows starts-of-stars network topology. Gateways are transparent bridges between server and end-devices. The gateways and end-devices communicate using different frequency bands. LoRaWAN data rates range from 0.3 to 50kbps.

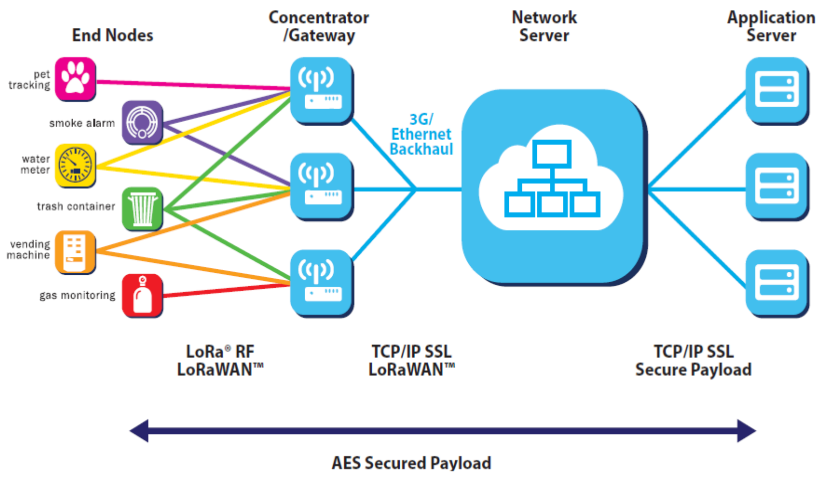


Figure 2.2: LoRaWAN Network Architecture[2]

LoRaWAN uses star topology which increases network capacity, reduces complexity, increases communication range and preserves battery lifetime. In LoRaWAN, each node transmits the data to multiple gateways. Each gateway then forwards data packet to the network server which will perform security checks, redundancy detection and message scheduling. Moreover, theres no need to handover, unlike cellular, in case of moving end-device(node)[2].

2.3.1 End-device Classes

Class A: In class A, end-device initiates the communication(uplink), while server communicates via downlink during predetermined response windows. Class A devices consumes lowest power as it only requires downlink communication from server. Thus, Class A devices have longest battery life[1].

Class B: It provides bidirectional communication with scheduled receive windows. End-device receives periodic Beacon from the gateway at scheduled time slots. The server can know whenever the end-device is listening. Class B also provide extra receive window called *ping slot*[1].

Class C: It continuously provides bidirectional communication as receive windows are nearly open every time[2]. Server can initiate transmission at any time. So, Class C devices have no latency. But they consumes the most amount of power due to persistent transmission.

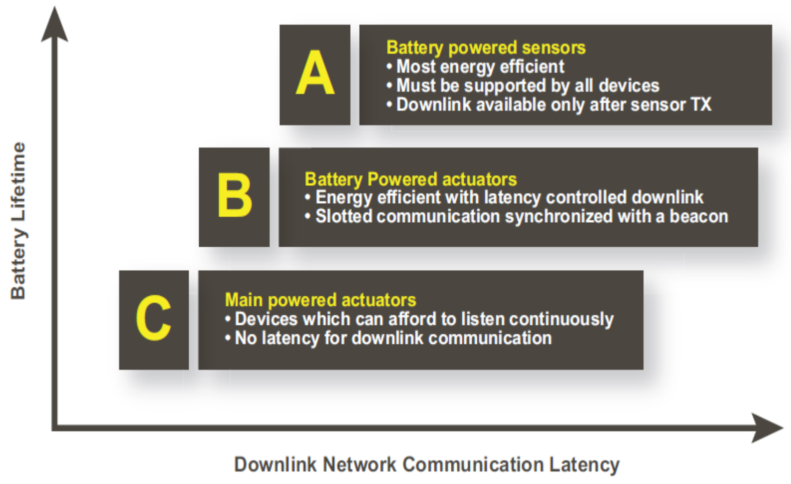

A
Battery powered sensors
• Most energy efficient
• Must be supported by all devices
• Downlink available only after sensor TX
B
Battery Powered actuators
• Energy efficient with latency controlled downlink
• Slotted communication synchronized with a beacon
C
Main powered actuators
• Devices which can afford to listen continuously
• No latency for downlink communication
Battery Lifetime
Downlink Network Communication Latency

Figure 2.3: LoRaWAN Device Classes[2]

2.3.2 Activation Methods

The end-device needs to be activated in order to communicate on the LoRaWAN network. For activation, end-device should provide the Device Address(DevAddr), Network Session Key(NwkSKey) and Application Session Key(AppSKey). DevAddr is a 32-bit unique identifier which is shared among end-device, network server and application server. It distinguishes network nodes and allows network to use and interpret data properly with the correct encryption keys[1]. NwkSKey is 128-bit AES encryption key is unique for each end-device which is shared between end-device and network server. It ensures message integrity and security for communication. AppSKey is also an AES encryption key which is unique per end-device. It is shared between device and application server to encrypt or decrypt data and for payload security. There are two activation methods to exchange these keys:

1. OTAA(Over-The-Air Authentication)

In OTAA, end-device sends Join Request to the application server. The request contains globally unique end-device identifier(DevEUI), application identifier(AppEUI) and authentication with application key(AppKey). Application server accepts the join request which is authenticated by device. It decrypts, extracts and stores DevAddr. It then obtains both NwkSKey and AppSKey.

2. ABP(Activation By Personalization)

In ABP activation method, the keys are locked to a specific network. The DevAddr, NwkSKey and AppSKey is configured at production time. It does not require any handshaking like OTAA method. Device can communicate on the network without any further process.

List of Applications

- IoT and M2M(Machine-to-Machine)

- Low Power Applications
- Industrial Automation
- Air Quality Monitoring
- Asset Tracking
- Smart Lighting, Metering, Agriculture
- Health Monitoring Devices
- Battery Powered Sensors and Actuators
- Infrastructure Management

Chapter 3

Hardware Design

3.1 Block Diagram

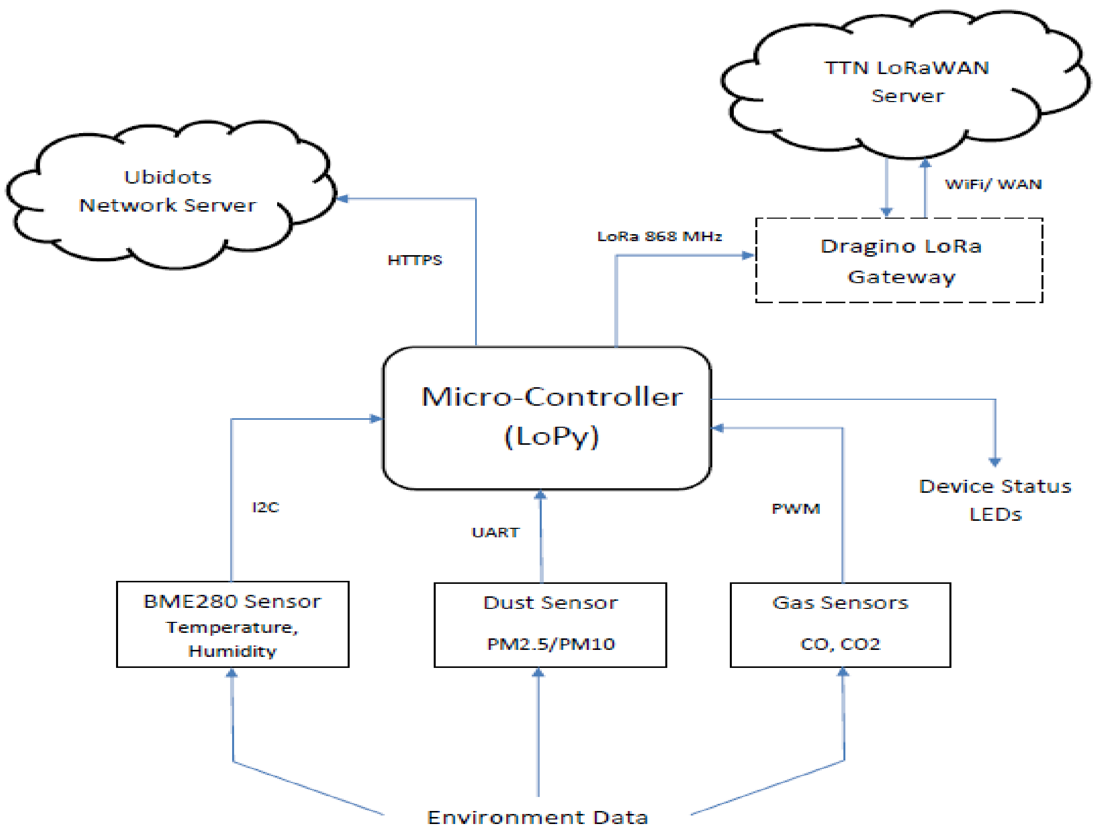


Figure 3.1: Block Diagram

3.2 LoPy Specification

LoPy is a MicroPython enabled micro-controller with LoRa(Semtech SX1272), WiFi and BLE technology. It has Espressif ESP32 chipset which achieves ultra-low power usage with greater flexibility. It has 512KB RAM, 32MB flash memory and 4MB external flash. It is programmable with Pymakr plugin for faster IoT application development. LoPy can either be used as LoRa node or as LoRa Nano-Gateway. The LoRa node can have range up to 40km, while LoRa Nano-gateway has range up to 22km. Nano-gateway has capacity to connect up to 100 nodes. LoPy has 24 GPIO interfaces which includes 2x SPI, 2x UART, I2C, I2S, micro SD card, 8*12 bit ADCs and 4*16 bit PWM timers. It supports SSL/TLS(Secure Socket Layer/Transport Layer Security) security. The LoPy normally runs *boot.py* and *main.py* on booting. It can be put into Safe mode to upgrade or degrade firmware.

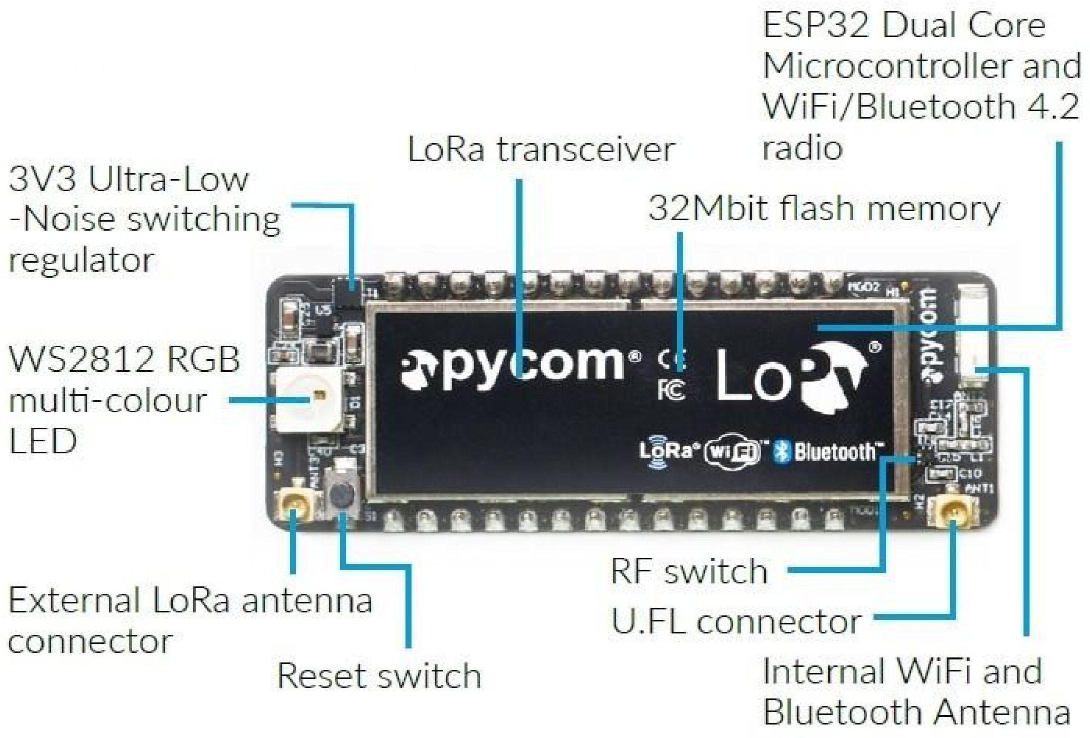

ESP32 Dual Core Microcontroller and WiFi/Bluetooth 4.2 radio
LoRa transceiver
3V3 Ultra-Low -Noise switching regulator
32Mbit flash memory
WS2812 RGB multi-colour LED
External LoRa antenna connector
Reset switch
RF switch
U.FL connector
Internal WiFi and Bluetooth Antenna

Figure 3.2: LoPy Controller

3.3 Why LoPy?

The LoPy micro-controller works on MicroPython programming language, which is developed from Python3. MicroPython is optimized to run in constrained environments. The code transfer from desktop to any controller or to any embedded system is done with ease. It also provides REPL(Read-Evaluate-Print Loop) environment to users for executing their code piece-wise.

Furthermore, We can use LoPy in either Station(STA) mode or Access-Point(AP) mode. LoPy can be connected via telnet using AP mode, while we can connect it to our wireless network router using STA mode. Moreover, LoPy has a small internal local file system called */flash*. It is accessible via an FTP(File Transfer Protocol) server using FTP client such as *FileZilla*. The FTP server runs on ftp://192.168.4.1. Also, LoRa nodes have long range which is very much beneficial for IoT devices with low power and low memory requirements. LoPy can be connected to a LoRaWAN network using TTN(The Things Network) or loriot as a Nano-gateway.

3.4 Sensors

3.4.1 BME280 Sensor

The BME280 is a combined digital sensor which measures humidity, pressure and temperature. The module is extremely compact and consumes low power. Due to this, it can be implemented in a battery driven devices. It measures data with high accuracy over a wider temperature range(-40 to +85 C), which makes it useful in applications such as Health Monitoring, Home Automation and Control and Internet of Things. The sensor can be interfaced using both I2C and SPI with VDD between 1.71 to 3.6 V.

3.4.2 Dust Sensor

The dust sensor uses laser scattering principle to get the particle concentration between 0.3 to 10um in the air. Light scattering is induced when dust particles go through detection area. The scattered light is transformed into electrical signals and will be processed further. The diameter and the amount of dust particles can be analyzed. It gives the digital PWM output and has a built-in fan. It can accurately measure PM2.5(Particulate Matter) and PM10. It uses UART(Universal Asynchronous Receive Transmit) communication interface to continuously send the data to the micro-controller. It works at 5V supply. The UART works at 9600 baudrate with 8 data bits, 1 start bit and 1 stop bit. The whole PWM output cycle is of 1004ms with 2ms of High level of output, 1000ms of corresponding time and 2ms of Low level output.

3.4.3 CO2 Sensor

The CO2 gas sensor works on non-dispersive Infrared (NDIR) principle to detect the amount of CO2 present in the air. The sensor has good sensitivity, high resolution, low power consumption and lifespan more than 5 years. The sensor has built-in temperature compensation and has UART as well as PWM output. It works at 5V power supply and measures the CO2 in the range of 0 2000 ppm. It is best used for indoor air quality monitoring systems such as smart home and schools.

3.4.4 Light Sensor

The light sensor converts light intensity into digital output with very high sensitivity. The device uses two ADCs to convert the photo-diode currents into digital signal output. The controller derives the illuminance in lux using formula to approximate the human eye response. It is operated just at 3.3V power supply. It can be directly interfaced with I2C protocol with data rates up to 400 kbps.

3.5 Neopixel WS2812B LEDs

WS2812B is the RGB LED with intelligent control circuit integrated inside single package. It operates at 5V power supply. The data transfer time is fast with reset time approximately around 50us. Each pixel of RGB colors can have 256 brightness with full color display[5]. The colors are consistent. The LEDs have reverse current protection, high brightness, low power consumption, small volume, better consistency and long lifetime. It uses NZR communication mode for transmitting data. The DIN pin receives data bits from LoPy which is fed to internal reshaping and amplification circuit[5]. Then, the processed data is displayed.

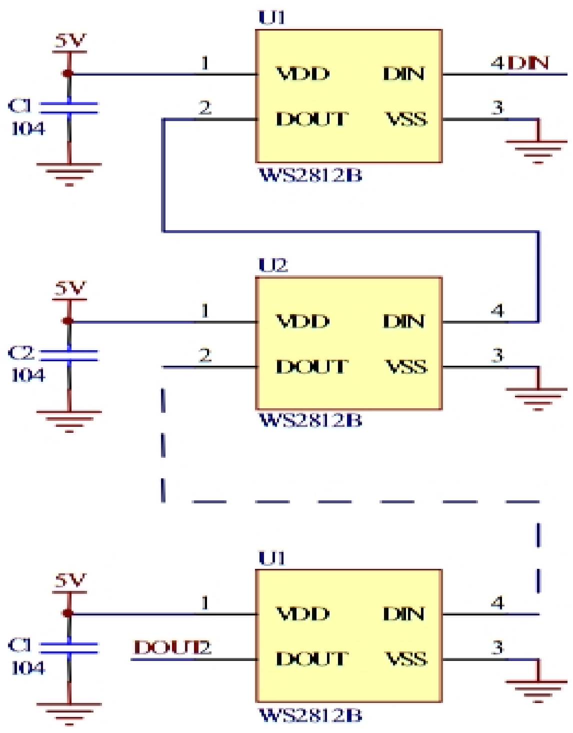


Figure 3.3: WS2812B Circuit[5]

3.6 Dragino LG01 LoRa Gateway

The Dragino LG01 is an open source single channel LoRa Gateway. It acts an a bridge between LoRa wireless network and IP network based on WiFi, Ethernet or Cellular[6]. It has open source Linux inside which allows users to modify firmware with their own requirement. It has built-in web server and is managed using Web GUI. Also, it supports both higher and lower frequency bands with high sensitivity down to -148dBm.

Chapter 4

Software Design flow

4.1 Micropython Features

Micropython programming language is implementation of Python3 which is optimized to run on micro-controllers with low power and low memory requirements. It provides set of the Pythons standard libraries, micropython specific libraries and system specific libraries. According to the system youre working on, the source code is available on the GitHub repositories. The code transfer is fast as it is done directly on the device controller without any compilation. Besides that, it has an interactive REPL prompt and modules for underlying hardware support.

4.1.1 REPL Prompt

The Micropython prompt is an REPL prompt which is useful for testing the code and executing commands easily. It allows user to run their code line-by-line. In addition to writing main.py, user can write the code in REPL. The REPL features which makes it interactive are as following:

- Halt any executing code with Ctrl-C
- Scroll through input history using up and down arrow keys

- Auto-indent while using python statements that end with a colon.
- Auto-completion of variables and module names with Tab Key
- Soft-reset with Ctrl-D
- Ctrl-E to enter the paste mode to to copy-paste chunks of code, Ctrl-D to exit.
- Assigning an underscore to a variable stores previous result in it.

4.1.2 Modules

The Micropython modules are the functions and class libraries. Micropython implements a subset of Python functionality to each of its module. To avoid contradictions, the Micropython modules derived from python uses u(micro) prefix. There modules are categorized as following:

a. Modules derived from standard Python libraries which cannot be expanded or modified by the user. Modules like *uos, ussl, utime* and *usocket* are not intended to be extended.

b. Modules which are functionally specific to the Micropython implementations. Modules such as *micropython, machine, network* are only used for Micropython.

c. Modules specific to a particular port or system and thus not portable. Modules like *pycom* and *pyb* are not portable to any other system.

4.2 Atom IDE features

Atom IDE(Integrated Development Environment) is an open source text editor made by GitHub. It allows users to create their own packages or to extend the existing packages. It has a smart and extensive auto-completion feature for every language. Atom workspace can be split into multiple panes to easily compare and edit code across files.The fuzzy finder feature helps to search any file in the current project. It has a package installer to install over 7000 packages in total. As it is made by GitHub, it is very easy to install and include the required libraries.

4.2.1 Why Atom?

Apart from above features, Atom provides the number of plugins that increases the work productivity as plugins enables the customization. By installing the drag-and-drop plugin, we can simply highlight the text and drag and drop it somewhere else. The Pymakr plug-in enables communication with the Pycom board with REPL console. Pymakr plugin allows to run the code on the board, to synchronize the project files to the board or to type and execute commands directly using REPL console.

4.3 Flow Chart

The software flow charts illustrates about the programming flow.

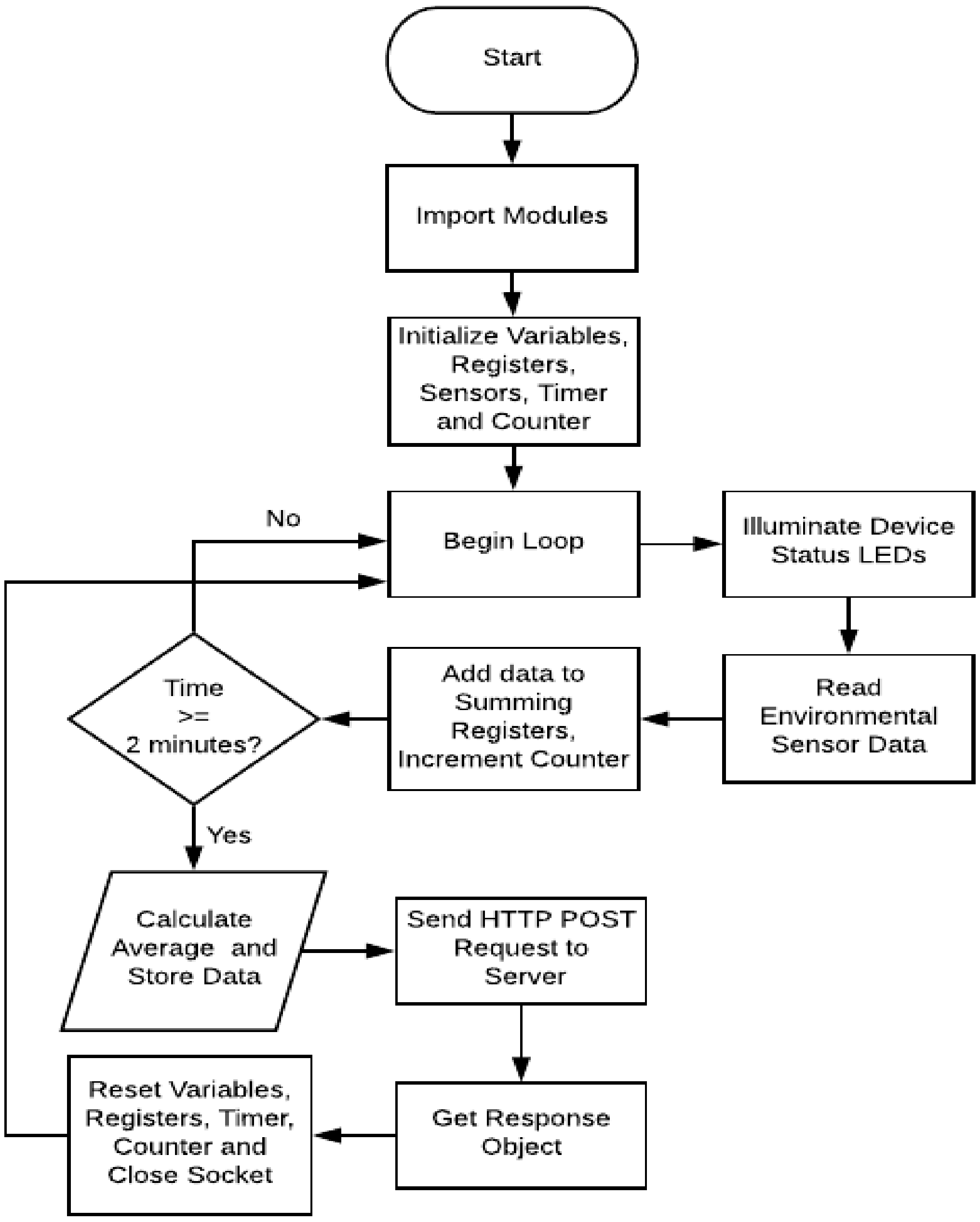

Start
Import Modules
Initialize Variables, Registers, Sensors, Timer and Counter
Begin Loop
Illuminate Device Status LEDs
Read Environmental Sensor Data
Add data to Summing Registers, Increment Counter
Time >= 2 minutes?
No
Yes
Calculate Average and Store Data
Send HTTP POST Request to Server
Get Response Object
Reset Variables, Registers, Timer, Counter and Close Socket

Figure 4.1: FlowChart(1)

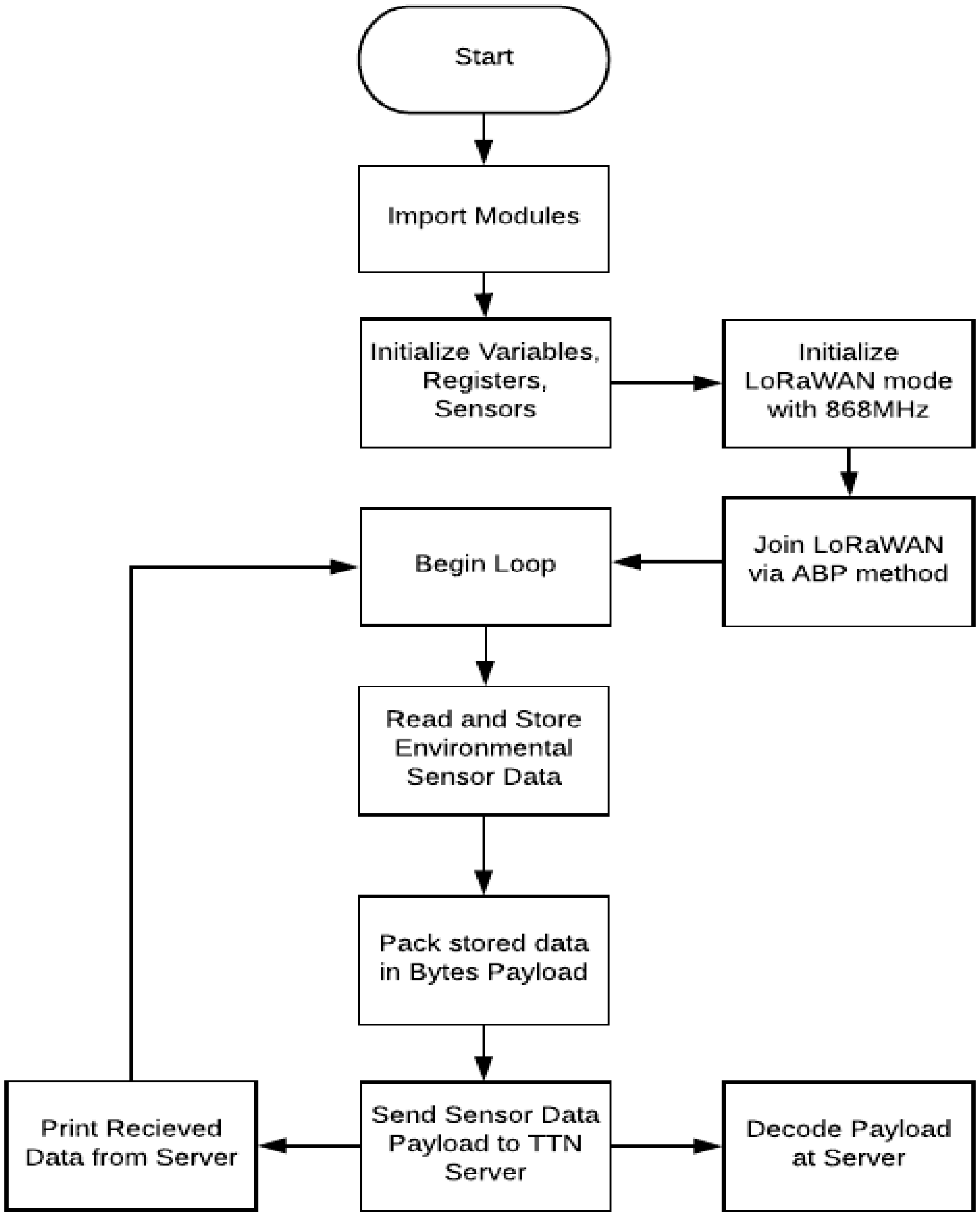

Start
Import Modules
Initialize Variables, Registers, Sensors
Initialize LoRaWAN mode with 868MHz
Join LoRaWAN via ABP method
Begin Loop
Read and Store Environmental Sensor Data
Pack stored data in Bytes Payload
Send Sensor Data Payload to TTN Server
Decode Payload at Server
Print Recieved Data from Server

Figure 4.2: FlowChart(2)

Chapter 5

Project Work and Outcomes

5.1 Integration of Sensors with LoPy

The BME280(*Temperature, Humidity and Pressure*), Dust sensor, CO2 gas sensor, CO sensor and Light sensor are integrated with LoPy micro-controller. The programming is done in Micropython language. The Atom IDE is used with Pymakr Plugin to sync project files and to view output on serial monitor continuously. The accumulated sensor data is sent either on the Ubidots Application Server over WiFi or on the TTN(The Things Network) Application Server over LoRa.

5.1.1 BME280 Sensor Data Accumulation

The BME280 sensor works at 3.3 V. It is interfaced with controller using I2C communication protocol. The default I2C address is 0x77. The library class BME280 has functions to read values of temperature, humidity and pressure. The floating point sensor data is added to the summing registers for 2 minutes. After that, the average of the gathered data is calculated. This data is then sent securely to the server using HTTP request. The response object and the sent data can be shown in serial monitor or on the server. Temperature, humidity and barometric pressure are measured in degree Celsius, hPa(hecto Pascals) and percent respectively.

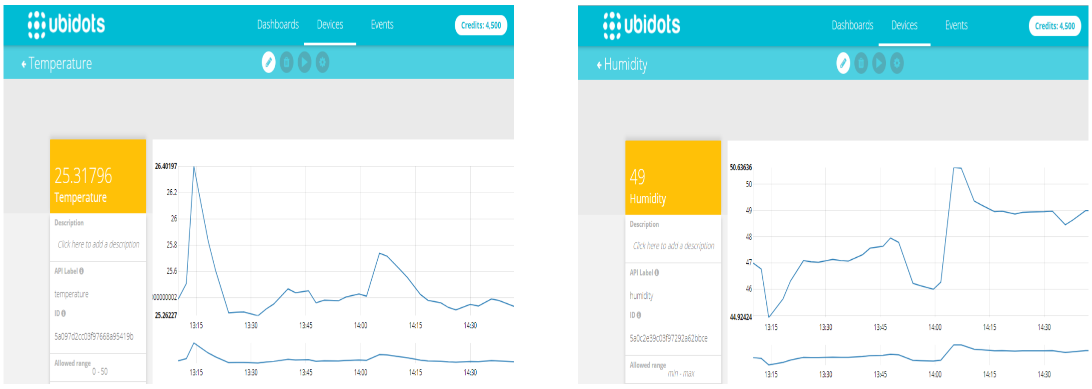


Figure 5.1: BME280 Sensor Data

5.1.2 Dust Sensor Data Accumulation

The Dust sensor works at 5V. It uses UART1 of the LoPy to read the sensor values. The values of dust particulate matters(PM2.5 and PM10) are measured in ug/m3. At booting, UART1 is initialized with 9600 baudrate. Then, the data is read using *uart.read()* function. The start and stop bytes are verified and data is added to summing registers. After 2 minutes, the average of accumulated data is taken which is sent securely on the server. The response object from server and sent data are displayed on serial monitor as well as on the server.

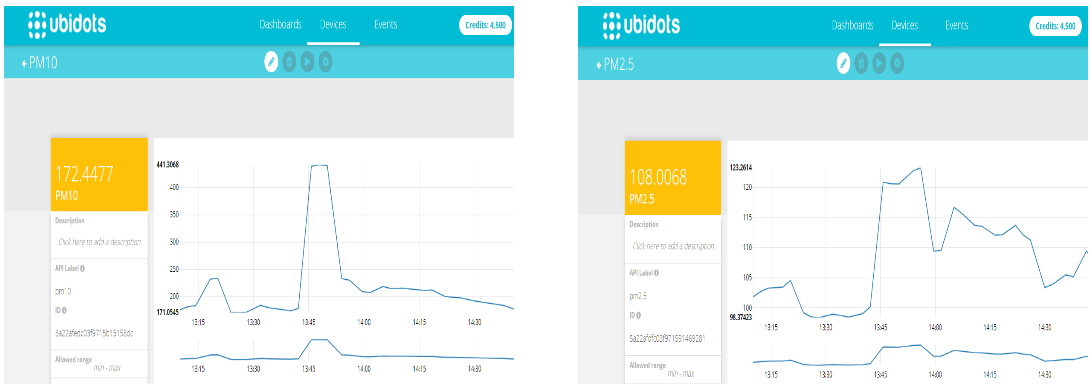
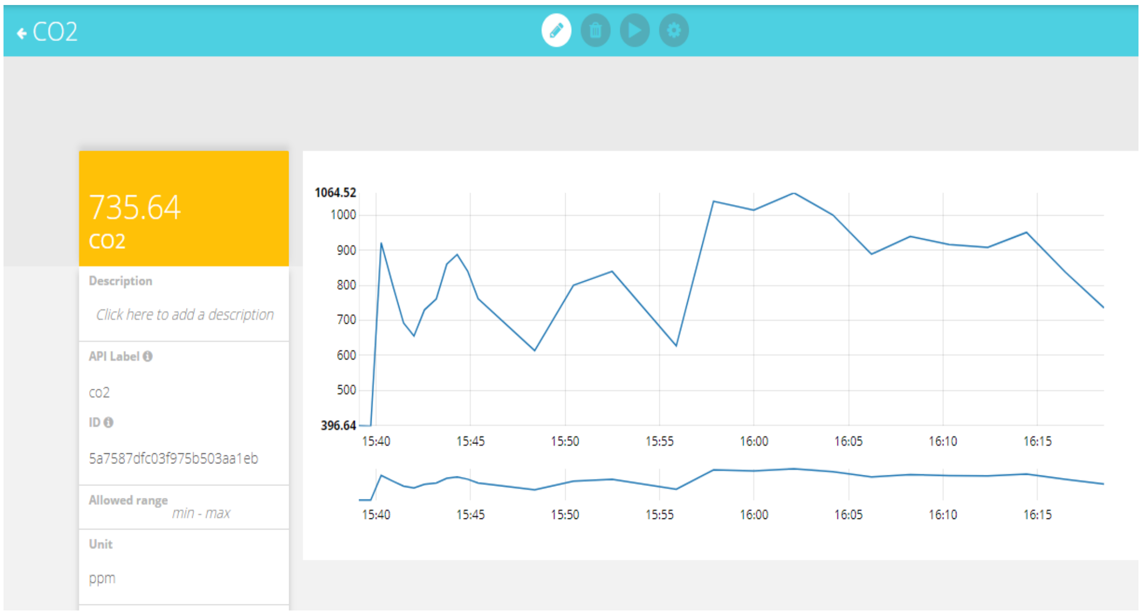


Figure 5.2: Dust Sensor Data

5.1.3 CO2 Sensor Data Accumulation

The CO2 sensor works at 5V and gives PWM output. The measured output range is between 0 2000ppm. The data is measured using PulseIN function. PulseIN function calls the handler when the PWM pin is held HIGH(or LOW). The handler function measures the length of the pulse in microseconds and calculate the time period for which the pin is HIGH. The result is stored and displayed on the application server.


CO2

735.64
CO2

Description
Click here to add a description

API Label
co2

ID
5a7587dfc03f975b503aa1eb

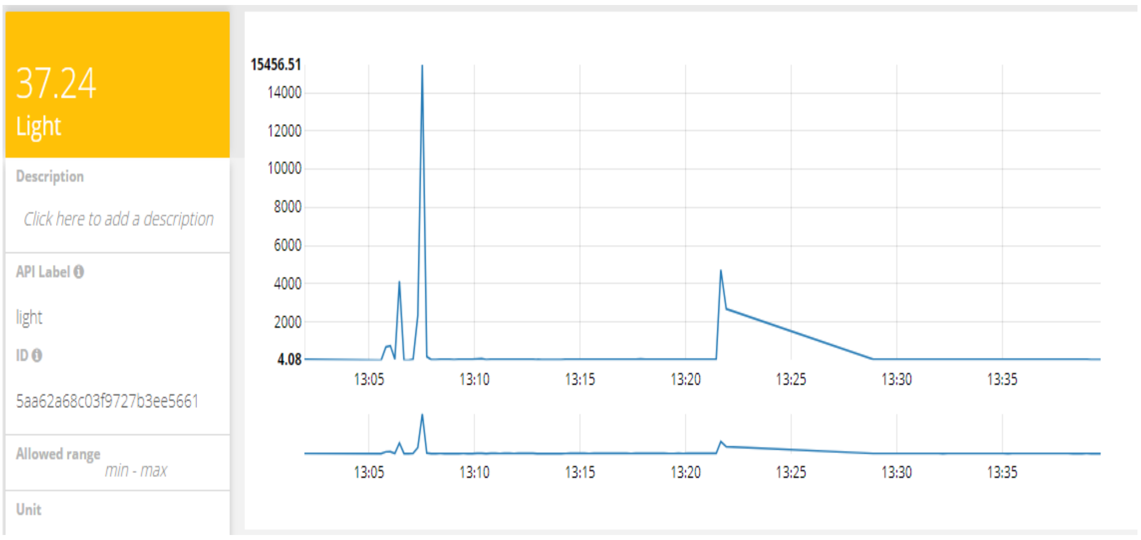
Allowed range
min - max

Unit
ppm

Figure 5.3: CO2 Sensor Data

5.1.4 Light Sensor Data Accumulation

The light sensor operates at 3.3V using I2C communication interface. The default I2C address is 0x29. The output obtained is digital and the measured value is in lux. The average accumulated data is sent to the application server after every 2 minutes and is also displayed using line graph.



37.24
Light

Description

Click here to add a description

API Label

light

ID

5aa62a68c03f9727b3ee5661

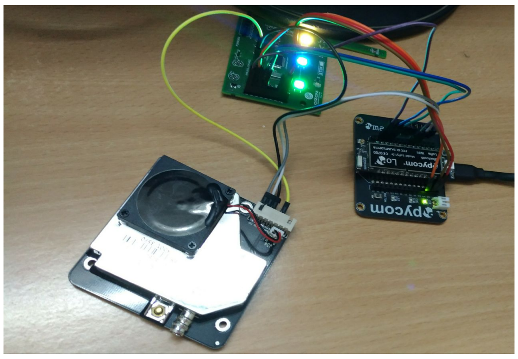
Allowed range
min - max

Unit

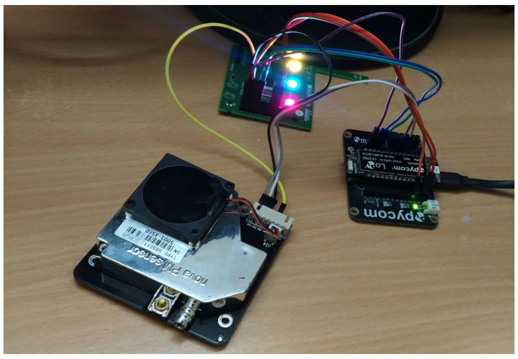
Figure 5.4: Light Sensor Data

5.2 Device Status LEDs

The Neopixel WS2812B LEDs are used as device status LEDs. The LEDs are integrated in a chain and are illuminated repeatedly. There are three LEDs which shows status of Data, Network and Battery. The network LED breaths Cyan continuously while reading data and is interrupted only while sending data on server. The battery status LED always displays Yellow color. The data status LED display stable Green color while reading the data. It changes to Magenta color when the data is being sent to the server.



(a) Reading Data

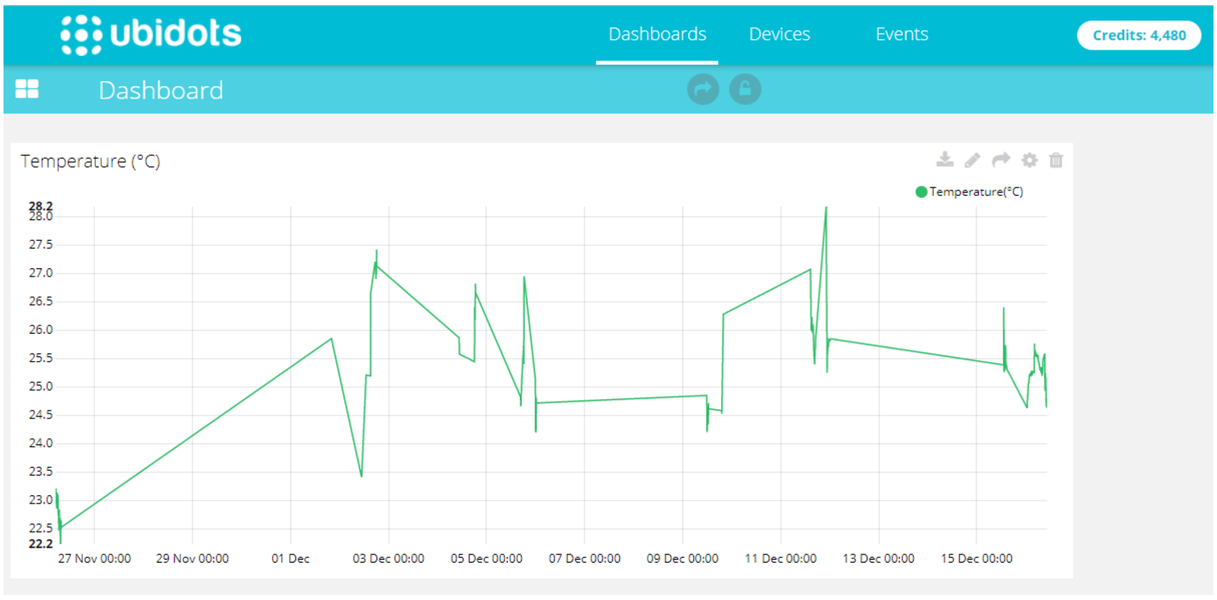


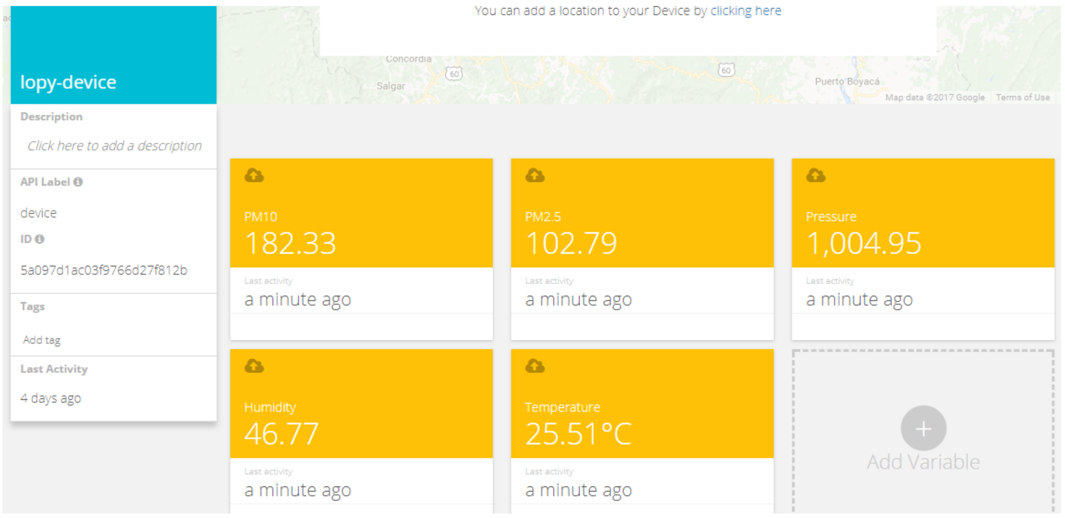
(b) Sending Data

Figure 5.5: Device Status LEDs

5.3 Ubidots Application Server

The data points are stored using the application server via HTTP protocol. The server used to store and plot data is *Ubidots Application server*. Every user is provided a unique Access Token. Firstly, the new device (lopy-device) is added in the dashboard. In lopy-device, the sensor variables are created. Each variable has a unique variable ID. The device uses urequest library to send the sensor data on server. The request() function uses arguments such as POST, URL, data and headers. The Access token is provided with the URL or in the headers for the authentication. The data contains variable IDs and their respective values. The Ubidots allows to see the hourly, daily, weekly and monthly data.





You can add a location to your Device by clicking here

lopy-device

Description

Click here to add a description

API Label

device

ID

5a097d1ac03f9766d27f812b

Tags

Add tag

Last Activity

4 days ago

PM10 182.33 Last activity a minute ago

PM2.5 102.79 Last activity a minute ago

Pressure 1,004.95 Last activity a minute ago

Humidity 46.77 Last activity a minute ago

Temperature 25.51°C Last activity a minute ago

Add Variable

Figure 5.6: Dashboard

5.4 LoRaWAN Network-Server

5.4.1 Dragino LG01 Registration with TTN

The LG01 gateway is configured as WiFi AP(Access Point) by default. It can be accessed and configured using its default IP address after connecting to its WiFi AP. Dragino is then connected to the internet by setting up either WiFi or WAN configuration. The Server Address, Server Port, Gateway ID, Bandwidth, Frequency, Spread Factor, etc are set as shown in Figure 5.7.

Now, the Gateway is registered on the TTN console by providing information such as Gateway ID, router and frequency plan. As in India, 865-867MHz frequency band is unlicensed to use, Europe 868MHz frequency plan is selected.

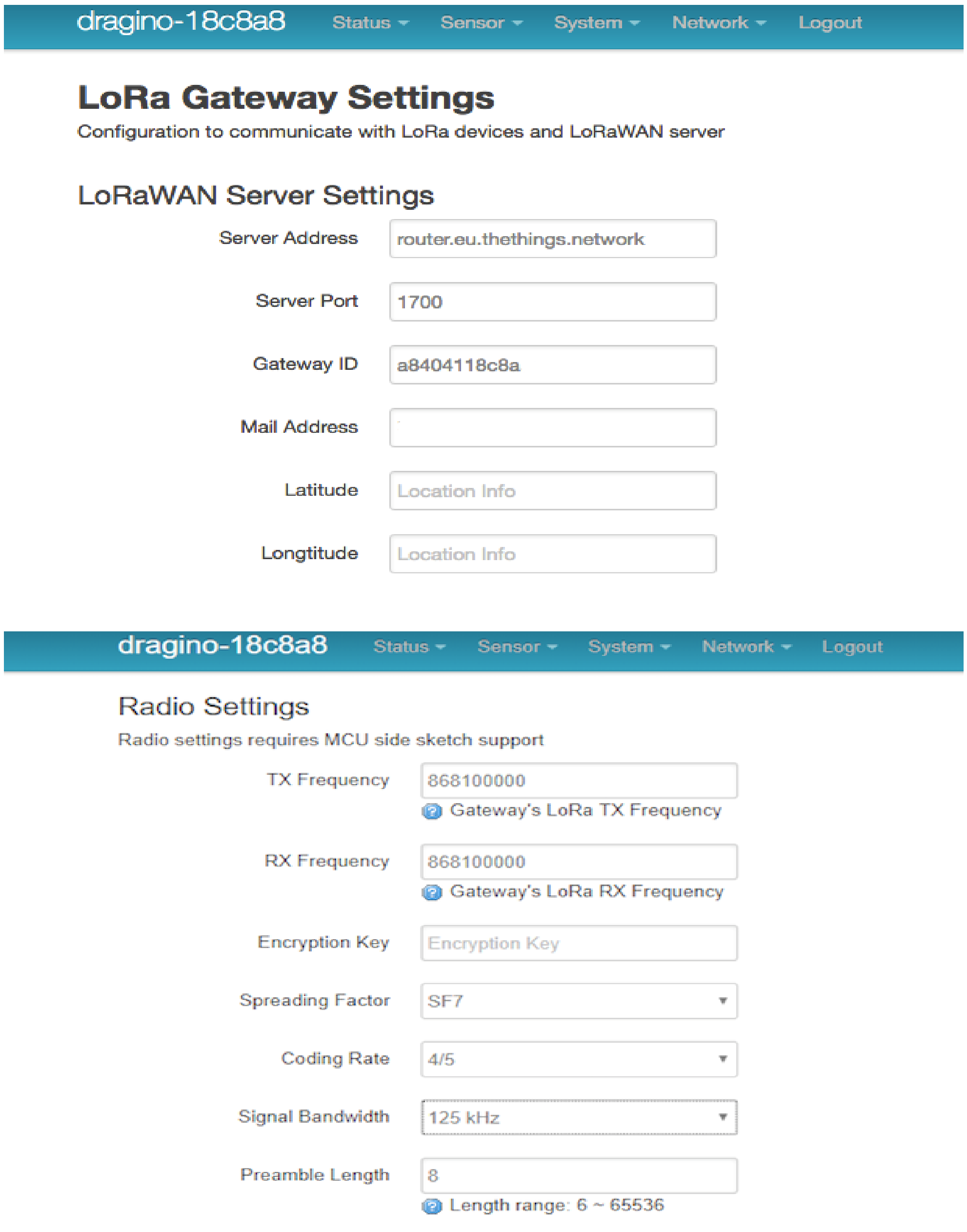
Once, the gateway is registered to the TTN console, we can see parameters such as gateway status, last seen and received/transmitted messages as shown in Figure 5.8.

5.4.2 LoPy Communication with Gateway and TTN

The LoPy micro-controller acts as LoRa node. In order to connect the node with Gateway and TTN LoRaWAN server, the node is configured. The configuration parameters to be set are frequency, region, Device Address(DevAddr), Network Session Key(NwkSkey), App Session Key(AppSkey), Activation Method and Data Rate. The Activation Method used is ABP(Activation By Personalization). To join LoRaWAN network server using ABP method, the DevAddr, NwkSkey and AppSkey are required. These activation keys are provided by the application server(TTN) once the device is registered with unique Device Identifier(DevEUI).

Every LoRa node has their unique DevEUI. LoPy's DevEUI can be fetched by running the command *binascii.hexlify(network.LoRa().mac())*Once DevEUI is provided to TTN, it will generate DevAddr, NwkSkey and AppSkey. After the node firmware is configured using these keys, the node will be successfully connected to the TTN Server. The device status is visible on Device Overview Panel (Figure 5.9).

The LoPy sends the LoRa Payload data(sensor data) which can be seen in the Application data on TTN. The payload is in the Hex Format which is decoded at the Server side using Payload Decoder Function to observe the received data properly(Figure 5.10).



LoRa Gateway Settings

Configuration to communicate with LoRa devices and LoRaWAN server

LoRaWAN Server Settings

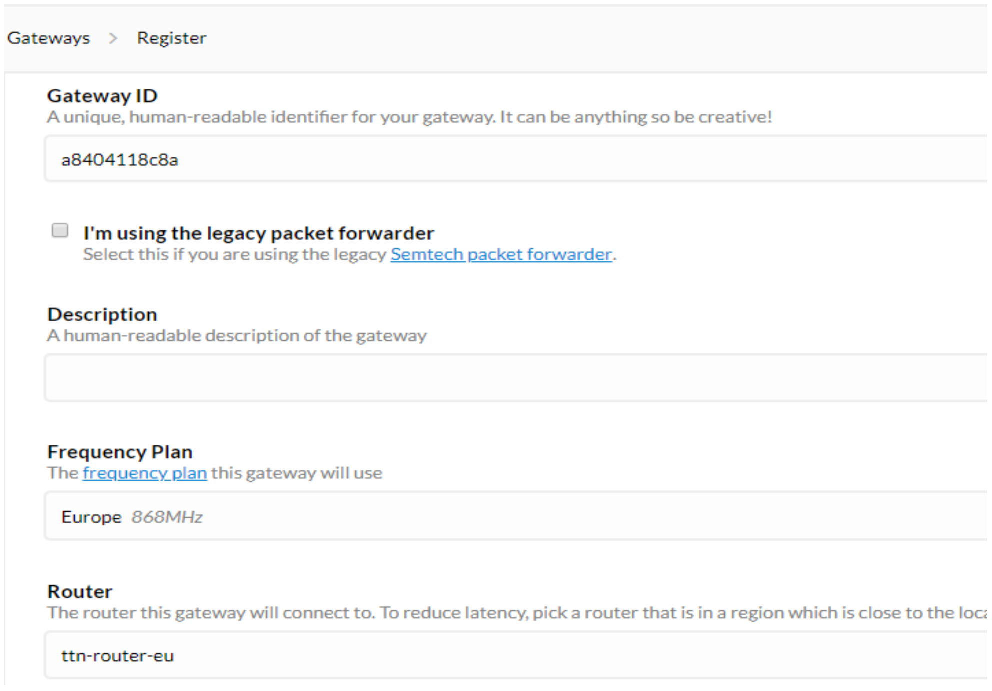
Server Address	router.eu.thethings.network
Server Port	1700
Gateway ID	a8404118c8a
Mail Address	
Latitude	Location Info
Longtitude	Location Info

Radio Settings

Radio settings requires MCU side sketch support

TX Frequency	868100000 — Gateway's LoRa TX Frequency
RX Frequency	868100000 — Gateway's LoRa RX Frequency
Encryption Key	Encryption Key
Spreading Factor	SF7
Coding Rate	4/5
Signal Bandwidth	125 kHz
Preamble Length	8 — Length range: 6 ~ 65536

Figure 5.7: LoRaWAN Server and Radio Settings


Gateways > Register

Gateway ID
A unique, human-readable identifier for your gateway. It can be anything so be creative!

a8404118c8a

☐ **I'm using the legacy packet forwarder**
Select this if you are using the legacy Semtech packet forwarder.

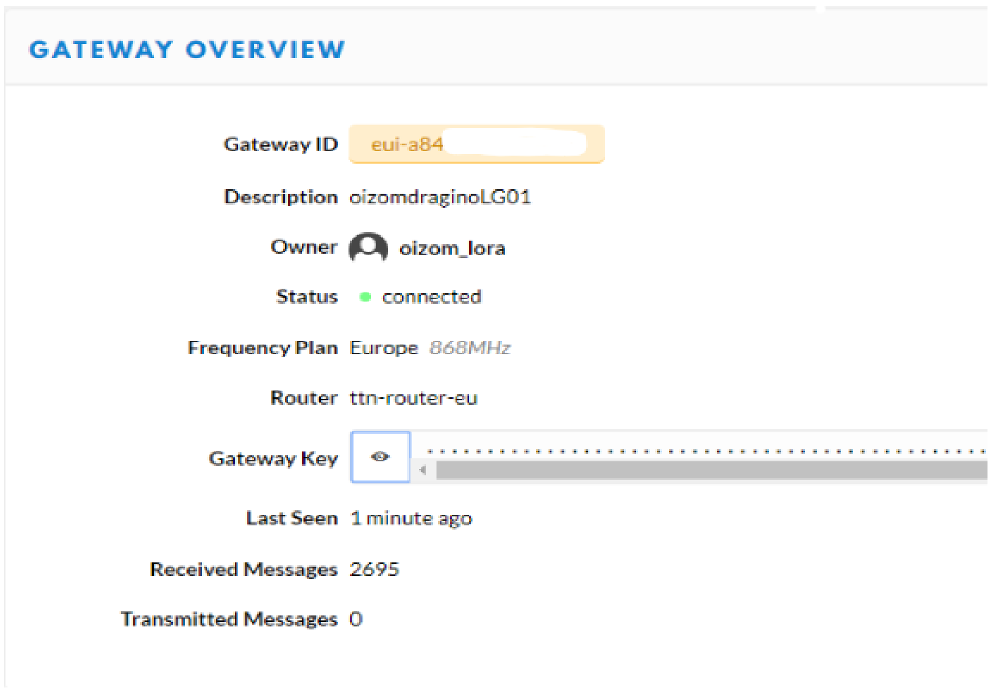
Description
A human-readable description of the gateway

Frequency Plan
The frequency plan this gateway will use

Europe 868MHz

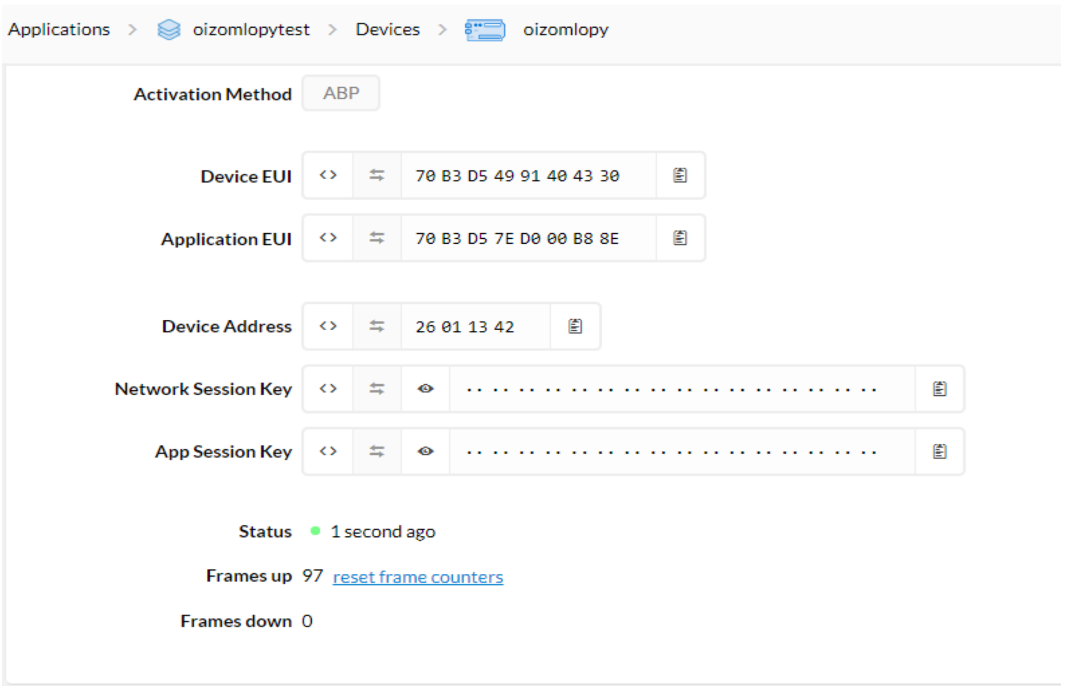
Router
The router this gateway will connect to. To reduce latency, pick a router that is in a region which is close to the loca

ttn-router-eu


GATEWAY OVERVIEW

Gateway ID	eui-a84
Description	oizomdraginoLG01
Owner	oizom_lora
Status	● connected
Frequency Plan	Europe 868MHz
Router	ttn-router-eu
Gateway Key	••
Last Seen	1 minute ago
Received Messages	2695
Transmitted Messages	0

Figure 5.8: Gateway Registration



Applications > oizomlopytest > Devices > oizomlopy

Activation Method: ABP

Device EUI: 70 B3 D5 49 91 40 43 30

Application EUI: 70 B3 D5 7E D0 00 B8 8E

Device Address: 26 01 13 42

Network Session Key: •• •• •• •• •• •• •• •• •• •• •• •• •• •• •• ••

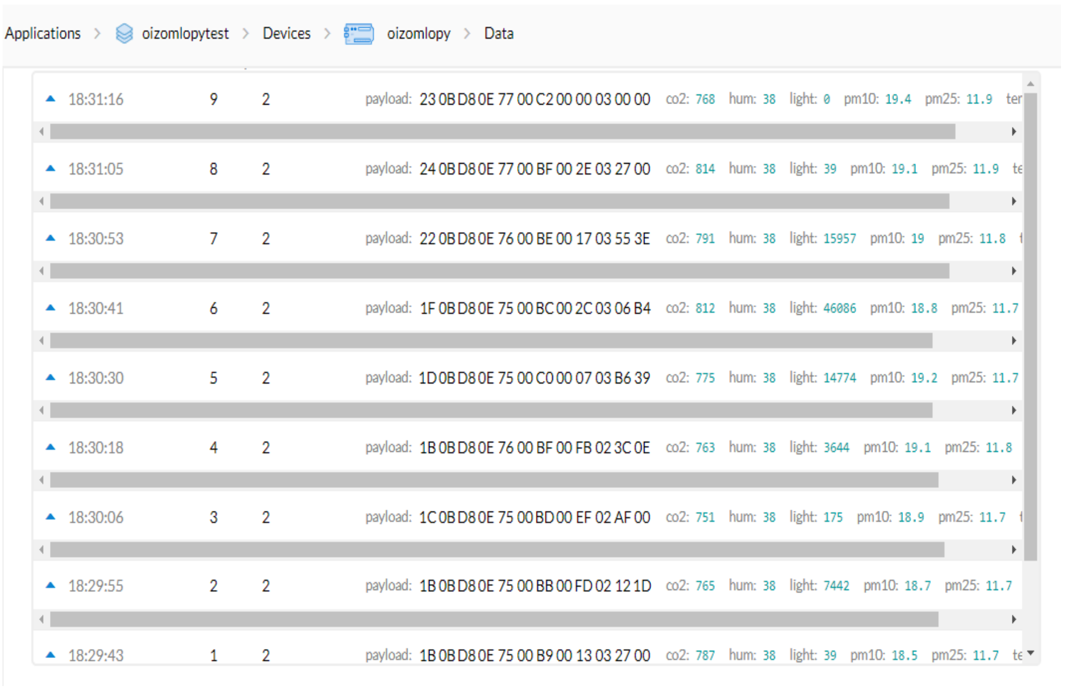
App Session Key: •• •• •• •• •• •• •• •• •• •• •• •• •• •• •• ••

Status: ● 1 second ago

Frames up: 97 reset frame counters

Frames down: 0

Figure 5.9: Device Overview



Applications > oizomlopytest > Devices > oizomlopy > Data

```
▲ 18:31:16   9   2   payload: 23 0B D8 0E 77 00 C2 00 00 03 00 00   co2: 768   hum: 38   light: 0       pm10: 19.4   pm25: 11.9   ter
▲ 18:31:05   8   2   payload: 24 0B D8 0E 77 00 BF 00 2E 03 27 00   co2: 814   hum: 38   light: 39      pm10: 19.1   pm25: 11.9   te
▲ 18:30:53   7   2   payload: 22 0B D8 0E 76 00 BE 00 17 03 55 3E   co2: 791   hum: 38   light: 15957   pm10: 19     pm25: 11.8
▲ 18:30:41   6   2   payload: 1F 0B D8 0E 75 00 BC 00 2C 03 06 B4   co2: 812   hum: 38   light: 46086   pm10: 18.8   pm25: 11.7
▲ 18:30:30   5   2   payload: 1D 0B D8 0E 75 00 C0 00 07 03 B6 39   co2: 775   hum: 38   light: 14774   pm10: 19.2   pm25: 11.7
▲ 18:30:18   4   2   payload: 1B 0B D8 0E 76 00 BF 00 FB 02 3C 0E   co2: 763   hum: 38   light: 3644    pm10: 19.1   pm25: 11.8
▲ 18:30:06   3   2   payload: 1C 0B D8 0E 75 00 BD 00 EF 02 AF 00   co2: 751   hum: 38   light: 175     pm10: 18.9   pm25: 11.7
▲ 18:29:55   2   2   payload: 1B 0B D8 0E 75 00 BB 00 FD 02 12 1D   co2: 765   hum: 38   light: 7442    pm10: 18.7   pm25: 11.7
▲ 18:29:43   1   2   payload: 1B 0B D8 0E 75 00 B9 00 13 03 27 00   co2: 787   hum: 38   light: 39      pm10: 18.5   pm25: 11.7   te
```

Figure 5.10: Application Data Payload

Chapter 6

Issues faced and Solutions

6.1 Data Overflow during Serial Communication

While reading the Dust sensor data serially form UART, the data bytes were getting overflowed and displaying undesirable output. In serial communication, it is necessary to flush the registers at particular intervals. But, the Micropython modules for LoPy does not provide any such function. Also, the internal registers which stores the data cannot be known.

6.1.1 Solution: RESET Controller

To solve this issue, a condition is put in the code. It checks the start and stop bytes every time after reading data from UART. If the condition is not met, then it resets the controller and start reading data again.

6.2 OSError: Sending Data Failed

While sending the data on the server using HTTPS, the device was facing an error at the time of wrapping the socket with SSL(Secure Socket Layer). There was no issue related to the *urequest* library or the server.

6.2.1 Solution: Close Socket

The MicroPython platforms such as LoPy does not have a full-fledged OS. Thus, it causes the malfunctioning and resource leaks if response objects are not handled manually. Closing the socket manually after sending the data solved this issue.

Chapter 7

Conclusion

7.1 Conclusion

By building the IoT firmware using lightweight MicroPython framework, the device measures environmental sensor data with high accuracy and resolution. Networking technology such as LoRa sends digital sensor output at ultra-low power easily to wider communication ranges over the internet.

References

[1] Technical Marketing Workgroup, “LoRaWAN 101, A Technical Introduction”, LoRa Alliance.

[2] “A technical overview of LoRa and LoRaWAN”, LoRa Alliance.

[3] BME280 Environmental Sensor Datasheet, Bosch Sensortec.

[4] Laser PM2.5 Sensor Datasheet, Nova Fitness Co., Ltd.

[5] WS2812B Intelligent control LED Datasheet, Worldsemi.

[6] LG01 LoRa Gateway User Manual, Dragino.