Lab Exercise

SoSe 2025 - Internet of Things Course

WLAN Communications in the Internet of Things (25 points)

Sustainable Communication Networks
Faculty of Electrical Engineering
University of Bremen

Submission deadline: 2nd of May 2025, 12 midnight

You are required to work in groups of 3. The report to be written at the end should be developed by all group members together, but each member must upload a copy of the report separately. The report should be in .PDF format and submitted using DoIT! at StudIP!.

Communication plays a pivotal role in the Internet of Things (IoT). The IoT landscape encompasses a diverse array of devices, all mandated to communicate with one another for various purposes. **MoleNet** is an extensible microcontroller platform (device) for the IoT, explicitly focusing on underground sensor networks. It is based on an Espressif ESP32 microcontroller and includes multiple connectivity, storage capabilities, built-in environmental sensing, and interfacing possibilities.

In this lab exercise, we will use the MoleNet platform to gather environmental data, such as temperature and humidity, and transmit it to an Internet-connected server. The goals of this lab exercise are as follows.

- Setup the built-in sensor of a **MoleNet** device to collect environmental data (temperature, humidity, pressure, and altitude);
- Establish a link between the **MoleNet** device and a WLAN Access Point to obtain a DHCP-based IP address;
- Setup a communication path between the **MoleNet** device and a server residing in the Internet;
- Communicate the collected environmental data at the **MoleNet** device to the server regularly.

The following components are required for you to complete this lab.

1. A **MoleNet** device (given by the lab supervisors),
2. A Laptop computer (your own),
3. A connecting USB cable,
4. The Arduino software development environment.

This lab exercise consists of **four parts**. The following list gives a summary of the lab work.

- Done during the lecture in class;
  - **Part 1** - Set up four Arduino programs and observe the output.
  - **Part 2** - Combine the functionality of the programs and send environmental data in **UDP packets**.
  - **Part 3** - Modify the program developed in **Part 2** to generate the altitude and send data in **TCP packets**.
- Done at home and submitted in a week;
  - **Part 4** - Submit a four-page report about the work.

The following sections provide the tasks involved in all parts and details of the report to be submitted. **After completing each part above, you must show your work to a supervisor.**

Part 1

Hardware Description

The MoleNet platform is used in this lab exercise. Figure 1 shows a picture of the MoleNet board with the essential components marked. The platform uses an Espressif ESP32-S3-WROOM-1 microcontroller with a built-in WLAN interface. The firmware embedded within the microcontroller boasts a comprehensive TCP/IP stack accessible through an API. This stack enables seamless WLAN communications via UDP and TCP protocols, providing versatility for various networking applications. Additionally, users have the flexibility to program the microcontroller using a range of integrated development environments (IDEs), including the popular Arduino IDE, ensuring a convenient and adaptable development experience.

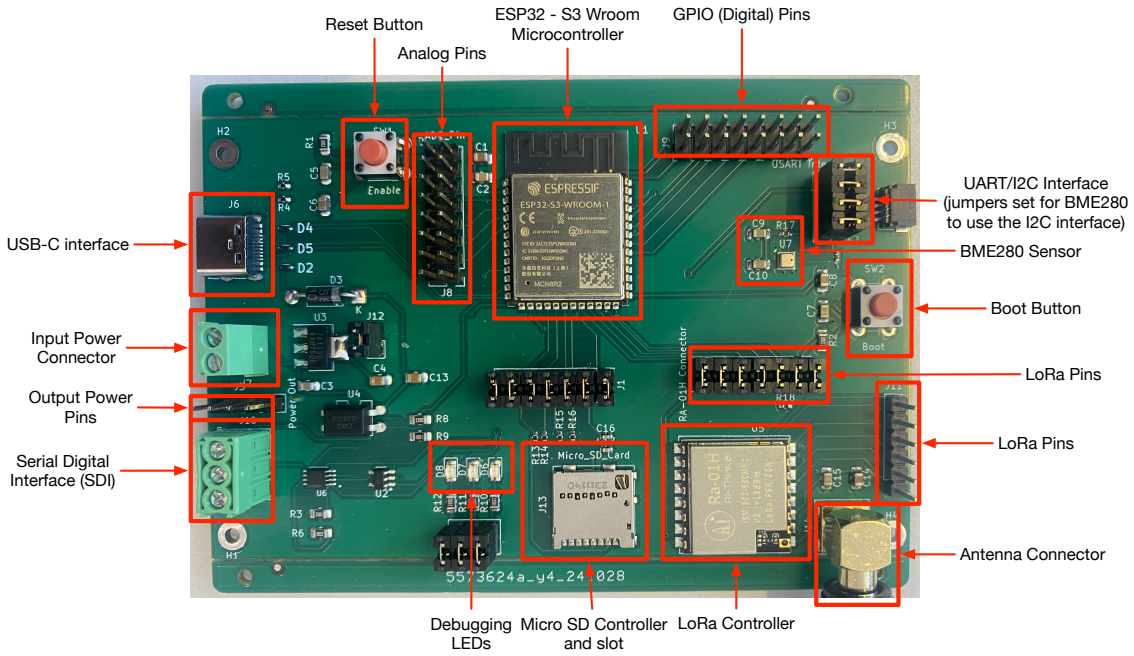

Figure 1: The MoleNet Platform with its essential components

The MoleNet platform has also integrated a BME280 Sensor accessible through an I2C interface, an SD storage slot, and a LoRa network interface. The BME280 sensor module is designed to capture data on barometric pressure, temperature, and humidity. Given the correlation between pressure and altitude, it also facilitates altitude estimation.

Step 1: Install & Configure the Arduino IDE

The lab code is developed, built, and installed on the MoleNet platform using the Arduino Integrated Development Environment (IDE). Download the Arduino IDE from the following link and install it on your computer.

`https://www.arduino.cc/en/software/`

Once installed, click on the **Boards Manager** to install the board packages required to use the ESP32 microcontroller on the MoleNet board. It requires the **esp32** package by **Espressif** to access several ESP32 modules. The MoleNet platform uses the module **ESP32S3 Dev Module**. Once the esp32 package is installed, connect the MoleNet platform to your computer and configure the **Board:** and **Port:** settings.

The **Board:** is the **ESP32S3 Dev Module**, and the **Port:** is the port that came up when the MoleNet platform was connected to the computer. Different ports are created at various times when the MoleNet Platform is connected to the computer. One port is to flash the firmware, while the second is to access

the MoleNet Platform's serial port. Use the **Reset** button on the MoleNet board to switch between the ports.

Additionally, set the **USB CDC On Boot: Enabled** and **USB Mode: USB-OTG (Tiny USB)** on the Arduino IDE to access the serial port on the MoleNet Platform.

Step 2: Using the BME280 Sensor

The built-in BME280 sensor senses pressure, temperature, and humidity. This sensor is accessed using a set of libraries. These libraries must be installed using the **Manage Libraries** option in the Arduino IDE. The libraries required to be installed are listed below.

- **Adafruit BME280 Library** - A library to access the functionality of the BME280 sensor.
- **Adafruit Unified Sensor Driver** - A library that provides an abstraction layer for accessing any Adafruit sensor

Once the above libraries are installed, download the following Arduino program, build it, and upload it to the MoleNet Platform to check the operation of the BME280 sensor. The output on the **Serial Monitor** shows the sensor readings.

- **Download:** ReadBME280.zip at `https://nc.uni-bremen.de/index.php/s/E9LQiMx5XWXJGTa`
- **Program:** ReadBME280.ino

Step 3: Using WLAN and UDP to Communicate

The built-in WLAN interface and the TCP/IP stack are used to communicate collected sensor data to a server on the Internet. Download the following Arduino program, build it, and upload it to the MoleNet Platform to check UDP communications. The output on the **Serial Monitor** of the Arduino IDE shows the communications.

- **Download:** WiFiUDPSend.zip at `https://nc.uni-bremen.de/index.php/s/E9LQiMx5XWXJGTa`
- **Program:** WiFiUDPSend.ino

The program connects to a WLAN Access Point and sends a string as UDP data to a server on the Internet. The supervisors install the WLAN Access Point and the Internet server. You must check and change the connection information in the code for the program to work successfully. The output is shown on the **Serial Monitor** of the Arduino IDE.

Step 4: Using WLAN and TCP to Communicate

Like the previous step, download the following Arduino program, build it, and upload it to the MoleNet Platform to check TCP communications. The output on the **Serial Monitor** of the Arduino IDE shows the communications.

- **Download:** WiFiTCPSend.zip at `https://nc.uni-bremen.de/index.php/s/E9LQiMx5XWXJGTa`
- **Program:** WiFiTCPSend.ino

The program connects to a WLAN Access Point and sends TCP data to a server on the Internet. The supervisors install the WLAN Access Point and the Internet server. You must check and change the connection information in the code for the program to work successfully. The output is shown on the **Serial Monitor** of the Arduino IDE.

Step 5: Retrieve a Unique Identifier

An ESP32 microcontroller has a unique identifier to identify itself uniquely. This identifier is accessed using a library. This library must be installed using the **Manage Libraries** option in the Arduino IDE. The library is listed below.

- **ArduinoUniqueID** - A library to obtain the unique identifier associated with a microcontroller.

Download the following Arduino program, build it, and upload it to the MoleNet Platform to check the operation. The identifier is a 16-digit hexadecimal number. The output is shown on the **Serial Monitor** of the Arduino IDE.

- **Download:** GetUniqueID.zip at `https://nc.uni-bremen.de/index.php/s/E9LQiMx5XWXJGTa`
- **Program:** GetUniqueID.ino

Part 2

Step 1: Combine Part 1 Programs to Develop an IoT Application

In this step, the programs run in **Part 1** must be combined to form a single program. This single program obtains the BME280 sensor readings and periodically sends them in a formatted string as UDP to the server. The string must be in a **CSV** format (as follows).

UniqueID,MatNum,SeqNum,Temperature,Pressure,Humidity

The **UniqueID** is a unique 16-digit hexadecimal microcontroller identifier. The **MatNum** is the 7-digit matriculation number of one of the students in your group. The **SeqNum** is a continuously incrementing number. The fields **Temperature**, **Pressure**, and **Humidity** are the readings that you obtain from the BME280 sensor. You **must** send this formatted string to the server every **5 seconds**. When combining the code, create a new Arduino program and then bring in the required parts of the code from the programs downloaded and run in **Part 1**.

Step 2: Check UDP data sent to the server

When you transmit your UDP data to the server, the server collects this information in a file. The supervisors show the contents of this file on the server's console. Please request a copy of your data by providing your **UniqID** or the **MatNum**.

Part 3

Step 1: Modify the IoT Application to Send TCP

Modify the program developed in **Part 2** to send the same information as TCP data. Additionally, compute and send the **correct Altitude** value. The format must be in **CSV**.

UniqueID,MatNum,SeqNum,Temperature,Pressure,Humidity,Altitude

Step 2: Check TCP data sent to the server

As before, when you transmit your TCP data to the server, the server collects this information in a file. The supervisors show the contents of this file on the server's console. Please request a copy of your data by providing your **UniqID** or the **MatNum**.

Part 4

Once you have completed setting up the components, developing your Arduino program, and running it successfully, you can request the dumps of the data (UDP and TCP) you sent to the server from the supervisors. You must use this data and other information to write and submit a **four-page** report of the lab exercise activities. The contents of this report must include the following information.

- Report title: WLAN Communications in the Internet of Things
- Administrative information
  - Your name(s) and your matriculation number(s)
- Section: Description of the Setup
  - A short description of the components used and how they are connected
  - A picture of your setup
- Section: Description of the Programs
  - A short description of the procedure of the programs you developed to send sensor data to the server.
- Section: Data Collection and Analysis
  - Generation of simple statistics from the sent data and an analysis of these statistical values. Create two separate analyses, one for the UDP and one for the TCP data.

Note: Since you must compute some fundamental statistical values in this report, make sure that you keep both of your programs running sufficiently long enough to collect at least 100 entries of your sensor readings at the server for the type of data sent (i.e., UDP and TCP).