Internet of Things

Group 3: Maxim-Leonid Rezan, Tiago Caldas De Silva, Elena Maria Pesci

Winter semester 2023-2024

Project: Thingy52 connected to Raspberry PI through ESP32

The aim of the project is to transfer data collected by the sensors of the thingy52 through Bluetooth Low Energy to the ESP32. The information is then passed using the MQTT protocol to the Raspberry Pi and visualized in the ThingsBoard cloud service.

Hardware:

Thingy52

It is a compact, power-optimized, multi-sensor development kit device designed for collecting environmental data. Thingy can sense movement, orientation, temperature, humidity, air pressure, light, color, and air quality. The data can be transmitted via Bluetooth to Bluetooth-enabled devices.

J-Link EDU Mini

It is a minimalistic debug probe that has been used in the developing of the project to flash code into the Thingy52.

ESP32

ESP32 is a series of low-cost, low-power [system on a chip](https://en.wikipedia.org/wiki/System_on_a_chip) [microcontrollers](https://en.wikipedia.org/wiki/Microcontroller) with integrated [Wi-Fi](https://en.wikipedia.org/wiki/Wi-Fi) and dual-mode [Bluetooth](https://en.wikipedia.org/wiki/Bluetooth).

It includes a built-in support for Wi-Fi and supports Bluetooth Low Energy, which are both vital in our application.

Immagine che contiene testo, elettronica, Componente del computer, Componente elettrico

Descrizione generata automaticamente Immagine che contiene Componente elettrico, Componente di circuito, Componente di circuito passivo, Ingegneria elettronica

Descrizione generata automaticamenteImmagine che contiene elettronica, testo, Componente elettrico, Componente di circuito

Descrizione generata automaticamente

Figure 1: Thingy52 Figure 2: J-Link Figure 3: ESP32

Source: <https://www.nordicsemi.com/Nordic-news/2017/06/nordic-thingy>

Source: <https://www.segger.com/products/debug-probes/j-link/models/j-link-edu-mini/>

Source: <https://www.digikey.at/en/products/detail/espressif-systems/ESP32-DEVKITC-VIE/12091811>

Raspberry Pi

The Raspberry Pi is a series of small, affordable, single-board computers (SBCs). It encapsulates all essential components of a computer in compact manner.

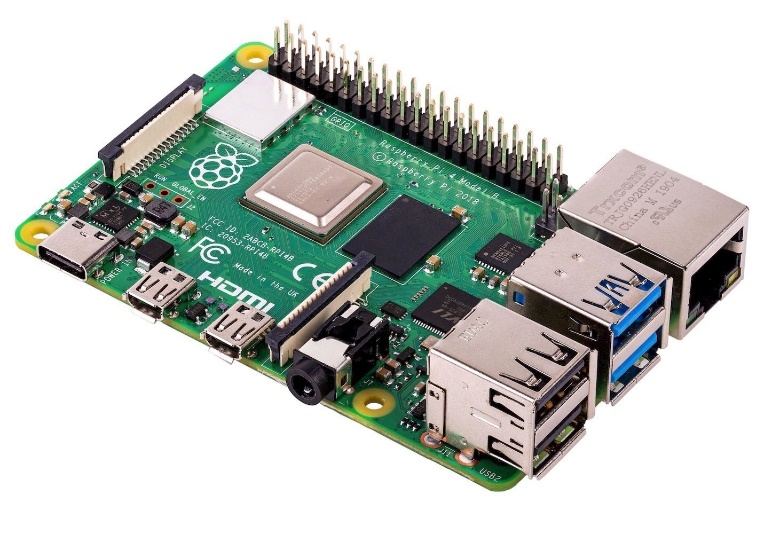


Figure 4: Raspberry Pi 4

Source: <https://www.berrybase.at/raspberry-pi-4-computer-modell-b-4gb-ram>

Development steps:

Note that whenever a Wi-Fi connection is mentioned in our report, we refer to the ‘max\_a53’ hotspot from the personal phone of one of use. The choice is based on the ease to access IP numbers and useful information about connected devices, besides the possibility to change location when working on the project.

* Raspberry Pi set up:

We started working on the Raspberry Pi first. Since it is a computer, to function it needs a running operating system. Raspbian is an operating system tailored to work on Raspberry Pi, we proceed by uploading it on microSD using a laptop. The microSD was then inserted into the computer.

We don’t dispone of a keyboard or monitor to use with the Raspberry Pi. Therefore, we built a headless configuration for the Raspberry Pi. This procedure allows to visualize the desktop of the computer directly on a laptop connected to the same Wi-Fi. This result was accomplished by the usage of TigerVNC and PuTTY.

The next step was to install the MQTT broker (mosquito) on the Raspberry Pi.

To be able to organize and manage the data from the communication, we installed ThingsBoard on the microcontroller. The platform runs on Java 11. The platform can be accessed by other devices connected to the same Wi-Fi, so we’ll show tables and graphs from our personal laptop.

* Communication between ESP32 and Thingy52:

To upload code to the Thingy52 we used VSCode with the extension nRF connect and Zephyr, while to load code on ESP32 we preferred Arduino IDE.

The connection is created exploiting the Bluetooth Low energy technology. This approach is particularly effective since we are using battery powered Thingy and the communication happens in short distances.

Things to verify based on the code: in general what does the code making reference to BLE

We are using a connecting topology (not broadcast) where ??what is central and what is pheripheral

A BLE device can be a master (initiate a connection), a slave (advertise its availability) or both.

The Thingy52 collects data regarding the temperature from the sensor and sends it to the ESP through BLE. We implemented two different functions to implement the communication: Nome1 is used to send the data and doesn’t expect an acknowledgment, to emphasize this transmission the ?? LED lights up. This functions is called each ?? seconds. Nome2 works differently, it’s usage is triggered by a variation in the temperature. It sends the information and waits for an acknowledgement by the ESP32, in this case the operation is accompanied by a ?? LED.

(Tiago please add the stuff about that code, to me it’s misterous. Like library used, important variables/ functions…)

* Communication between ESP32 and Raspberry Pi

The aim is to pass the data received by the Thingy52 to ThingsBoard using a MQTT transmission.

The MQTT protocol is based on a publish/subscriber messaging pattern

The code loaded on ESP32 is written in ArduinoIDE using the libraries PubSubClient.h and WiFi.h.

(Add details when code is ready)

Data Visualization:

The platform ThingsBoard allows the creation of a dashboard where the information received by selected devices can be visualized. We decided to used the widgets Entities Tables, that shows the latest value of a data key, and Timeseries Line Chart, to plot the sequence of temperatures collected.

(here screenshot of table and graph)

Final Conclusions:

It’s important to highlight a few of the main takeaways produced by the development of this project. The challenges have been many, mainly related to the correct installation of software into the hardware. We learned to persist, exploit the useful information of forums and websites and persue alternatives until the reach the desired goal.

The requests are met, the information collected from sensors of Thingy52 flows through the ESP32 using a BLE connection and reaches the ThingsBoard platform using the MQTT protocol, furthermore the number of Thingy52 devices that transmit data is dynamically managed.

The video attach shows our result in action.