Kristel & Prabhav – 26/09/2025

MSc lab Autumn term – Wireless & Wired Sensor System

1. Aims and objectives

There are four main themes in this lab.
a) build two fully autonomous sensor nodes from chip level up and establish wireless data transfer via an ESP32-S3 Feather MCU[1] to cloud;
b) program the MCU to read out multiple commercial sensor boards using I2C and/or SPI wired connections;
c) design a PCB to wire the commercial boards in a compact format without any other wires apart for power;
d) design & 3D print an enclosure for packaging that fits the PCB;
e) use all possible sensors and/or combination of sensors to estimate the height of the EEE building and critically analyse sensor system performance.

The objective is to build a complete sensor system using a combination of self-built and commercial sensor nodes and using multiple wired and wireless communication systems including WiFi, BLE, I2C and SPI. By building this system we aim to practice embedded systems approaches, investigate signal transfer methods and analyse the capabilities of different sensors for a certain application (in this case height measurement).

2. Organisation

Each week, starting from week 2, a 3-hr lab session is timetabled. Look at your personal timetable for the scheduling. During these timetabled sessions, instructions and explanations will be given and members of staff will be available to support you. You are expected to attend these sessions and work in your group.
The MSc lab is on level 1 of the EEE department building. You will have access to this lab outside of the timetabled sessions during department opening hours and when there is space available (other MSc courses in the department use the same lab space).

4 MSc students form a laboratory group who need to work as a team by dividing up the work while keeping each team member fully informed of the approaches and solutions taken. In the end, everyone needs to know everything even though they might not have fully contributed to each aspect of the work.

The members of staff involved in the lab are: Prof. Kristel Fobelets (room 714), Dr. Prabhav Reddy and Dr. Sara Ghoreishizadeh. Technical support is provided by Mr. Amine Halimi and Mr. Danny Harvey.

Each group will be given a lab box that contains microcontroller, sensor boards, wires, battery. Groups will also be supplied with one Picoscope that connected to your laptop can be used as a oscilloscope. Thus, this lab requires you to have a laptop. Laptops can be borrowed from the automatic laptop dispenser next to stores near the lower-level entrance to the building.

The list of items that will be given to each group is given in Table I. These will be handed out by stores on level 1.

Table I: components and equipment given to each group by Stores

Part number	Description	Website link
1528-5477-ND	Adafruit ESP32-S3 Feather with 4MB Flash 2MB PSRAM - STEMMA QT / Qwiic	ESP32-S3 Feather
1528-5046-ND	Adafruit BME688 - Temperature, Humidity, Pressure and Gas Sensor - STEMMA QT	BME688
1528-4415-ND	Adafruit MINI GPS PA1010D STEMMA QT	Mini GPS

[1] Microcontroller unit

1528-5425-ND	STEMMAQT VL53L4CX TIME OF FLIGHT	Time of Flight Distance Sensor
1528-1462-ND	MICROSD CARD BREAKOUT 5V OR 3V	MicroSD card
1528-1220-ND	Monochrome 0.96" 128x64 OLED Graphic Display - STEMMA QT	OLED display
N/A	Home developed accelerometer autonomous sensor node	N/A
N/A	Home developed PPG autonomous sensor node	N/A
RS	PicoScope 2405A	PICOSCOPE 2405A datasheet
	Rechargeable coin-cell batteries for autonomous sensor nodes.	
2987-DH-20UE0062-NH-ND	C-TYPE USB cable	
1528-4399-ND	STEMMA QT / Qwiic JST SH 4-Pin Cable - 50mm Long	
	Pinheader cables	

At the end of the Autumn term all items need to be returned in the box to stores.

All PCB and 3D print designs need to be submitted via the EEE Tech Services - Power Apps. If you do not have access, please contact Mr Amine Halimi. Before production, submitted designs will first be checked by the technical team.

3. Deliverables and assessments

Task 1 autonomous sensor nodes

To be submitted: The data sheet of both your sensors containing – circuit schematic, PCB design and performance parameters. The report should not be longer than 10 pages. One report per lab group.

Task 2 environmental sensor

To be demonstrated to lab assessors: The environment sensor system within its enclosure + demonstration of its operation. To be submitted: The data of the environmental sensor experiment compared to its control and critically analysed for impact on public health. The report should not be longer than 5 pages and should contact a link to github page with the code, enclosure design and the data. One report per lab group.

Task 3 How many ways to measure the height of the EEE building and with what accuracy

To be presented to the whole class: Powerpoint presentation with findings.

4. Tasks

Task 1: Design Custom Sensor Nodes

The aim of this part of the lab is to learn PCB design and layout, on-board power management approaches, antenna layout for limiting interference and a custom-made USB extension board for programming the MCU to

avoid USB connectors on the small boards. These board should transmit data wirelessly (BLE) to the ESP32-S3 Feather MCU. This hardware and software will need to be used in the other parts of the laboratory and thus implementation needs to be generalizable.

The deliverables of this section

- Design PCBs for two fully autonomous sensor boards and submit for production.
- Develop software to read out your sensors nodes and send the data wirelessly to a database.
- Devise a test protocol to analyse the performance of your autonomous sensor nodes.

The PCBs will be fabricated and populated by JLCPCB. You will need to take the design capabilities of JLCPCB into account in your design.

You need to design the PCBs for two autonomous sensor nodes from chip level. One board should contain and accelerometer and the other a PPG sensor (photoplethysmography).

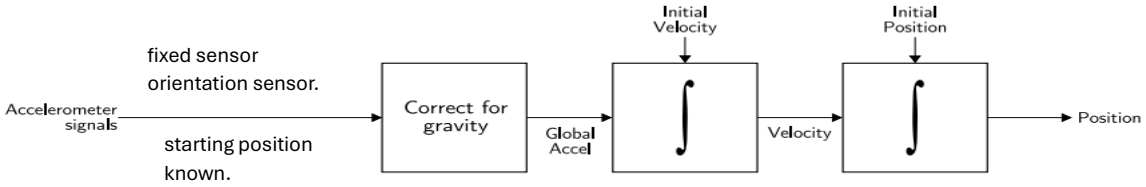
4.1 Brief description of the sensors

An accelerometer detects changes in velocity, *v* by measuring the force exerted on a small mass inside the device (Newton's second law).

$$\boldsymbol{F} = m \cdot \boldsymbol{a} = m \cdot \frac{d\boldsymbol{v}}{dt} \quad (1)$$

with ***F*** the force, *m* the mass, ***a*** the acceleration and ***v*** the velocity

The accelerometer used in the lab is based on MEMS (Micro-Electro-Mechanical-Systems) technology. A tiny proof mass is suspended by beams. When the device accelerates, the mass lags due to inertia which causes a displacement relative to the frame. This displacement is measured via capacitive, piezoelectric or piezoresistive sensors with the capacitive sensor the most applied. In capacitive accelerometers the proof



mass sits between two fixed plates and as the mass moves its distance to the plates changes resulting in a change in capacitance. These capacitance changes are converted in g-forces in three directions. MEMS accelerometers are small, light and have low power consumption and start-up times. Their main disadvantage is that they are not currently as accurate as accelerometers manufactured using traditional techniques. Linear accelerometers can be used as inertial navigation devices. Once an initial position is known, the accelerometer can track the change in position over time. Since the accelerometer used in this lab does not have a gyroscope, rotation cannot be measured directly[2]. The inertial navigation approach is sketched in Fig. 1.

Figure 1: Inertial navigation calculation methodology. Figure adapted from [3].

The position, *s* can be determined by:

$$v(t) = v(0) + \int_0^t (a(t) - g)dt \quad (2)$$

$$s(t) = s(0) + \int_0^t v(t)dt \quad (3)$$

with *a*(t) the temporal acceleration, *v*(t) the velocity, *s*(t) the position. *v*(0) and *s*(0) are the initial velocity and displacement, and g is the acceleration due to gravity that always acts on the sensor and thus the measurement needs to be corrected for the gravitation. 1 g = 9.81 m/s^2.

The integration scheme can be implemented on an MCU using the simple rectangular rule based on the rate with which the samples arrive.

[2] This means that if you want to use this node to navigate, the orientation of the sensor should not change during the recording. With some trigonometry you should be able to find the angle from the 3D acceleration information.

[3] An introduction to inertial navigation

$$v(t+\delta t) = v(t) + \delta t \cdot (a(t+\delta t) - g) \quad (4)$$

$$s(t+\delta t) = s(t0) + \delta t \cdot v(t+\delta t) \quad (5)$$

With δt the time between sample points.

To determine how the correction for the gravitational force needs to be applied look at the sketches of Fig. 2. The gravitational acceleration *g* will act against a weight going up and with a weight going down.

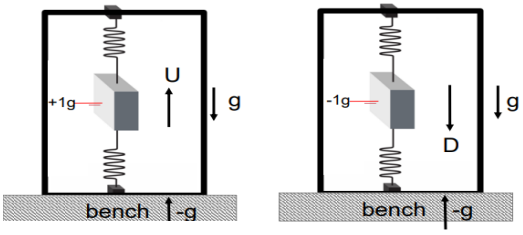


Figure 2: The sign of the correction by g when moving either up or down. Figure taken from [4].

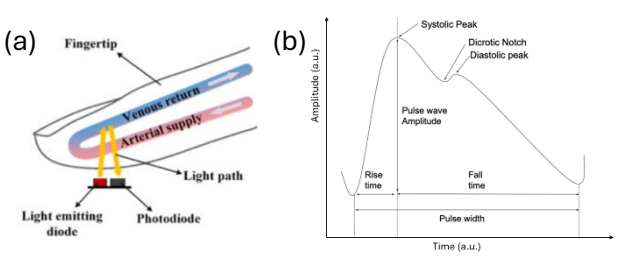
A PPG sensor (Photoplethysmography sensor) measures blood volume changes in the microvascular bed of tissue using a light-based technology. It's commonly used in smartwatches, fitness trackers, and medical devices to monitor heart rate, and sometimes blood oxygen levels (SpO_2). The PPG chip used in this lab uses red (~660 nm) and IR (~940 nm) LED light. These wavelengths are chosen because they penetrate tissue well. The sensor works by light absorption or reflection. As blood pulses through the blood vessels with each heartbeat the small veins expands and contract leading to variations in light absorption and reflection. A photodiode or photodetector measures the amount of light that is either reflected (in reflective PPG) or transmitted (in transmissive PPG) through the tissue. In reflective PPG (implemented here) the photodiode and photodetector are at the same side of the chip. The photodetector output over time will give a voltage waveform that varies over time with a pattern related to heart rate. Fig. 3 gives an illustration of the traditional position of a PPG sensor and the theoretical signal waveform that is produced.

SpO_2 (peripheral capillary oxygen saturation) can be derived from red and IR light reflection because oxygenated and deoxygenated haemoglobin absorb light differently at specific wavelengths. Oxyhaemoglobin (HbO_2) absorbs more IR light and less red light while deoxyhaemoglobin (Hb) absorbs more red light and less IR light. The ratio of absorption at these two wavelengths changes depending on the oxygen saturation of the blood. SpO2 can be calculated by:

$$R = \frac{(AC_{red}/DC_{red})}{(AC_{IR}/DC_{IR})} \quad (6)$$

with *AC* the pulsatile component (due to arterial blood) and *DC* the non-pulsatile component (due to skin, tissue, venous blood). This ratio *R* must then be mapped to an SpO2 value using a calibration curve derived from empirical data. SpO2 levels should be > 95% for healthy people.

Fig. 3(b) shows that the signal has two phases related to the mechanism behind the operation of the heart driving blood through the veins. The heart rate can be extracted from the distance between the peaks in the waveform.



[4] Slide 1

Figure 3: (a) Schematic position on the index finger of the LED and photodetector. (b) The part of the waveform that is repeated as a function of time. The signal in this figure does not have noise nor baseline wander.

Note that real sensors will have noise related to the electronic components as well as motion-related noise. For the PPG sensor, the signal will vary with any variation of the pressure between the sensor and the skin. In addition, low frequency motion noise that is a result of the sensor moving with respect to the skin surface with generate unwanted low frequency noise. Don't forget that mains noise (50 Hz in the UK) and other noise sources (lights, electronic switches, etc) will distort the signal.

Therefore, the signals from both sensors will require filtering (bandpass) to remove different unwanted noise components.

4.2 Building PCBs

You can use kiCAD, Altium Designer[5] or other software to design your PCBs. Before you design the PCB, you need to draw the circuit diagram in the same environment from chip level. The datasheet of the different chips you will be using contains recommended schematics and advice that are a good starting point for the designs. The following text takes you through the different chips that you must use and links them to their datasheet. Autonomous sensor nodes require an MCU and an antenna for wireless communication, as well as a battery with a battery management system. In this exercise, you will use the ESP32-C3FH4 as microcontroller (MCU). Click on the ESP32 link for the datasheet. A reference schematic that you will need to build your circuit around the ESP32 chip can be found by clicking here. The ESP32 is a family of System-on-Chip (SoC) controllers that have an integrated BLE (Bluetooth Low Energy) and WiFi transceiver, a lot of General-Purpose Input/Output (GPIO) pins, is capable of numerous transmission protocols and are generally easy to write firmware for, with a lot of online resources available. The C3 has a very small package footprint and can reduce its power significantly using its sleep modes.

BLE will be used as wireless communication protocol for the autonomous sensor nodes. A small compact low power chip antenna will be used - the ANT2012LL00R2400A. The PCB layout[6],[7] needs careful attention to limit interference between the hardware around the MCU and the antenna. Leave distance between the BLE antenna and the rest of the circuit and ensure good grounding of the antenna area via vias to ground. This implementation is illustrated in Fig. 4.

The PPG sensor uses the MAX30102 chip and the accelerometer uses the LIS2DE12. Click on the links to be redirected to the datasheet of the sensors. In these datasheets you will find a typical application circuit that identifies the peripherals that are required. Note that the PPG sensor communicates via I^2C (Inter-Integrated Circuit) with the on-board MCU while the accelerometer chip can use either I^2C or SPI (Serial Peripheral Interface). The comparison between the two is given in Table II.

Table II. Comparison between I^2C and SPI

Feature	**I^2C**	**SPI**
Wires	2 (SDA, SCL)	4 (MOSI, MISO, SCLK, SS)
Complexity	Simpler wiring	More pins needed, especially with multiple devices
Addressing	Devices have unique addresses	Each device needs a separate chip select (SS) line
Typical Speed	Up to 400 kHz (standard), 3.4 MHz (high-speed)	Up to tens of MHz
Performance	Slower	Faster, better for high-speed data transfer
Communication	Half-duplex	Full-duplex

[5] Also available via Imperial College London ICT software hub

[6] Chip Antenna Layout Tips for BLE, WiFi & Zigbee | Johanson Tech

[7] AN91445 Antenna Design and RF Layout Guidelines

Master/Slave	Multi-master possible	Single master, multiple slaves
Protocol	More complex (start/stop conditions, ACK/NACK)	Simpler, just clock and data lines
Overhead	Higher	Lower

Summary

- Use I^2C when you want simplicity and fewer pins, and speed isn't critical. Good for low-speed peripherals like temperature sensors, RTCs, EEPROMs.
- Use SPI when you need speed and performance and can afford more pins. Good for high-speed devices like displays, ADCs, and flash memory.

I^2C PCB implementation guidelines:

- Use a 4-layer PCB for improved grounding, signal – ground separation and easier routing.
- Proper impedance matching must be done.
- The total bus capacitance should be kept < 400 pF, thus keep traces short, direct, avoid large pads, stubs and branches and minimize via count. Run the SDA and SCL line not too close together.
- Use proper pull-up resistors: 2.2 kΩ to 10 kΩ[8], place them close to the master (MCU).
- Place a solid ground plane under the I^2C traces and use guard vias to ground for shielding.
- Add test points on SDA and SCL lines for oscilloscope probing and debugging.
- For more detailed guidance, you can refer to:
  - NXP I^2C-bus Specification and User Manual (UM10204)
  - Analog Devices – Designing Robust, Isolated I^2C Interfaces

A Battery Management System (BMS) is an electronic control system that manages rechargeable battery packs by monitoring their condition, controlling their operation, and ensuring safe performance. For lithium-ion batteries specifically, the BMS serves as a critical safety component that prevents dangerous conditions while optimizing battery performance. You must implement a BMS on your autonomous boards. The BMS prevents the battery from overcharging, over-discharging, and overcurrent, which can damage the battery or reduce its lifespan. It monitors and controls temperature to avoid overheating, tracks battery health, charge level, and remaining capacity. It optimizes energy usage to extend battery life. A BMS for a single coin cell battery is given here.

Batteries do not supply a constant voltage under load, therefore an LDO (low dropout regulator) needs to be used to keep the voltage stable and step down the battery voltage to the level required by the different components on the board. The LDO used in this work is the TLV755P 500mA, Low-IQ LDO.

The battery charging circuit will not be implemented in the sensor node for health and safety reasons and also to keep the board as small as possible. However, you need to design a separate battery charging PCB to charge you battery safely.

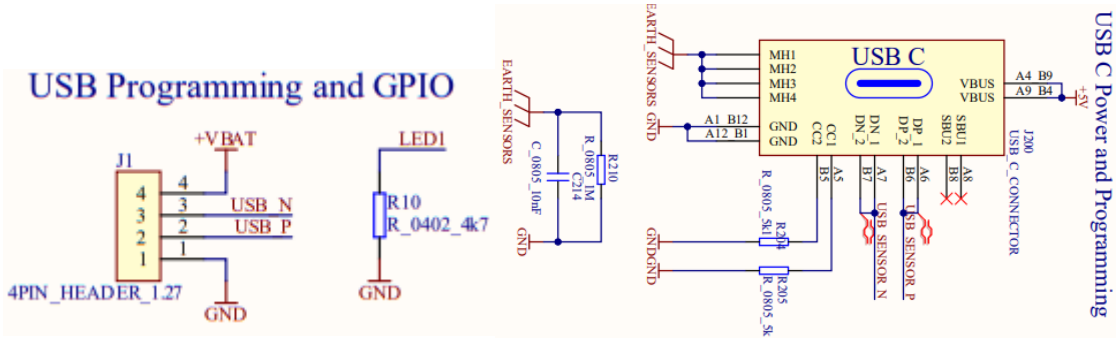
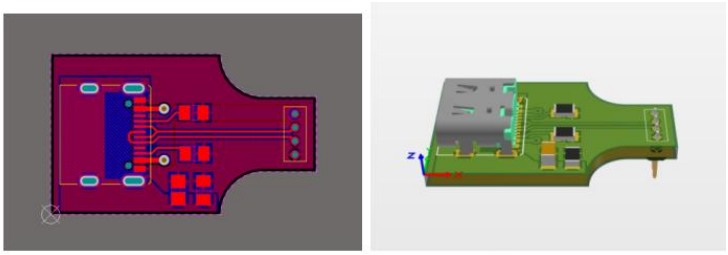


Figure 4a: left the schematic for the receptacle on the sensor node. Right: the schematic for the extension board.

[8] What are I2C Pull-Up Resistors and How to Calculate their Values - Total Phase

The ESP32 can be programmed directly via Universal Serial Bus (USB), however, adding the necessary connector or other programming logic on the board will significantly increase the size of the board. Therefore, the programming can be performed with a small 1.27 mm pitch 4-pin connector via a custom-made USB extension board, shown in figure 5b. This means that on the sensor nodes a receptacle needs to be defined that connects the USB extension board to the MCU on the boards for programming. The circuits related to both are given in Fig. 4a.



(a) Layout (b) 3D View

Figure 4b: Custom-made USB-to-sensor node connector board for programming the MCU on the node. (a) the PCB layout and (b) a 3D rendering.

A 3D rendering of a possible PPG PCB is shown in Fig. 5. The PPG signals from this board are given in Fig. 6.

Figure 5: 3D rendering of the PPG sensor node PCB. Left: the top view and right the bottom view of the PCB. Figure taken from the final year project report of Petar Barakov – EEE department, 2023.

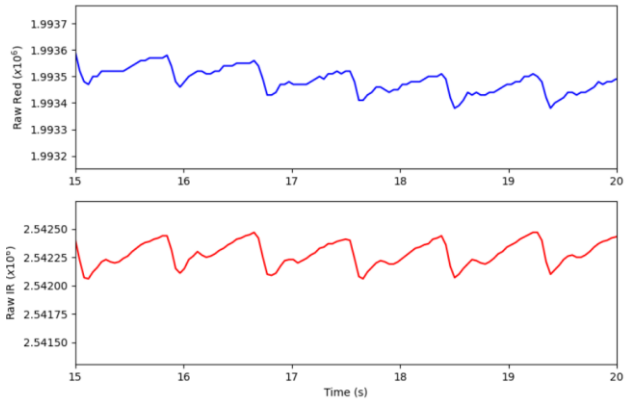


Figure 6: The output for the red (top) and infrared (bottom) light of the PPG sensor measured when sitting still.

Put the number of your group on the PCB for easy identification. After PCB design, they will need to be checked by Amine Halimi. When approved, they will be sent for fabrication in JCLPCB. Due to the use of very

small components, the soldering will also be done by the PCB manufacturer. During the wait time for the PCBs to be returned, the home-built sensor nodes can be used for implementing the wireless communication protocol to cloud or to the ESP32-S3 feather development board that can be used as a master. In the next section a communication architecture is explained. Keep in mind that your architecture should not only work for the autonomous sensor nodes but should also be able to integrate the commercial sensor system that will be explained in Task 2.

4.3 Communication Architecture

The architecture of the sensor system could be as shown in Fig. 7.

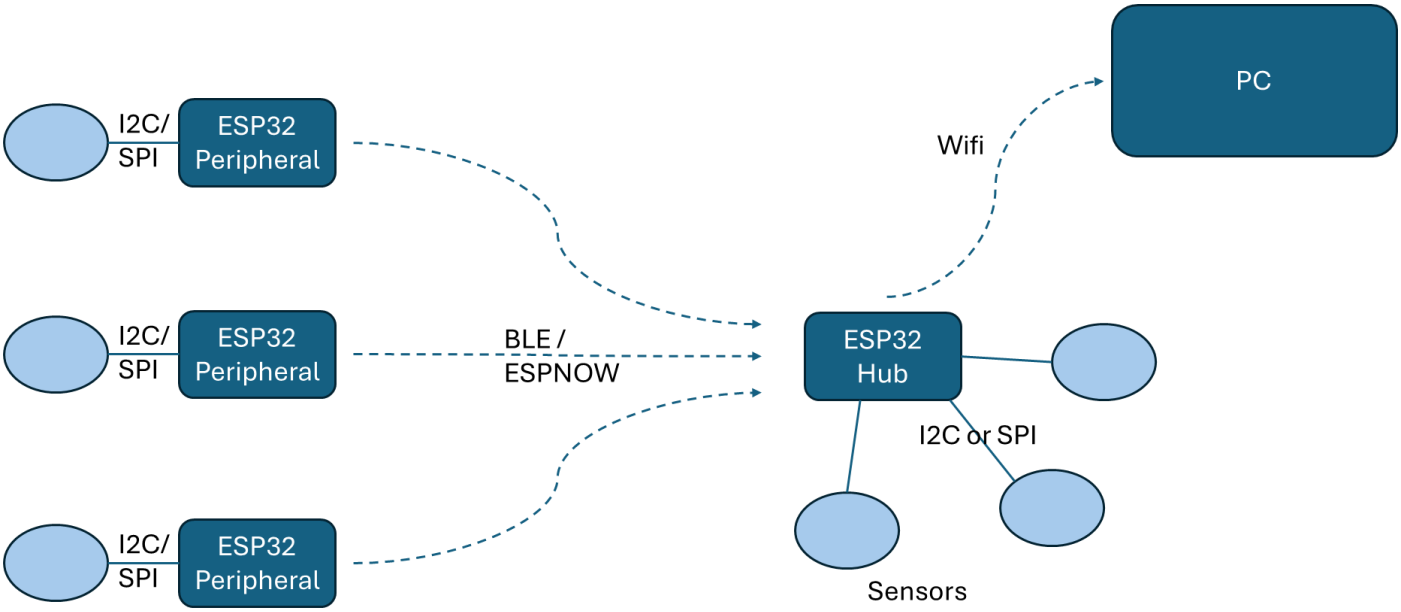


Figure 7. The architecture of the sensor system to be built.

As seen in Fig. 7, the sensor system has an ESP32 hub which collects data from commercial sensors which are physically connected to it – this is either done using I2C or SPI connections (see Task 2). In addition, the hub must be able to collect data from peripheral ESP32s – the autonomous sensor nodes. This would be via wireless communication using either bluetooth low energy (BLE) or ESP-NOW (a Wi-Fi based protocol). The hub then arranges the data and sends it to a PC where it is stored in a database. The connection to a PC is via Wi-Fi. Instead of sending the data to the PC, you may decide to store the data on a SD card – this makes sense if the PC is not available.

Description:

1. ESP32 Peripheral: The peripheral needs to collect data from a sensor and sends it to the hub. It is usual to configure this as the BLE server if you are using BLE.
2. ESP32 Hub: The hub will receive data from the peripherals. If you are using BLE, the hub is usually configured as a BLE client. It may need to collect data via I2C or SPI from other sensors. In addition, the hub will have to configured as a web server. At periodic intervals, the hub pushes data which is read by a code on the PC and stored in the database.
3. PC: You will need to listen for server-sent events and store the data in a database. We have worked with SQLite databases (Using sqlite3 library) but many other database and python libraries exist.
4. We have tested both home Wi-Fi (at home and through the phone hotspot) and enterprise Wi-Fi (ImperialWPA) and have been able to get the ESP32 to connect to them.

Helpful resources:

Kristel & Prabhav – 26/09/2025

1. ESP32 BLE Server and Client (Bluetooth Low Energy) | Random Nerd Tutorials - The tutorial shows how to send data between two ESP32 devices using BLE.
2. ESP32 Web Server with BME680 - Weather Station | Random Nerd Tutorials - The tutorial collects data from a BME680 sensor and displays it on a webpage.
3. ESP32: ESP-NOW and Wi-Fi Web Server Dashboard (Arduino) | Random Nerd Tutorials - The tutorial shows how sensor data collected by a peripheral is sent to a hub using ESP-NOW. The hub displays the data on a webpage.
4. sseclient · PyPI - The sseclient library in Python is used to handle Server-Sent Events (SSE), which allow servers to push real-time updates to clients over HTTP. It provides a simple way to consume event streams from web servers.
5. NimBLE-Arduino - The BLE library is quite large and we found that along with the HTTP Client library was exceeding the program memory of the ESP32. Because of this we replaced the BLE library with NimBLE, which is a light-weight version of the ArduinoBLE library. We found the migration guide (from ArduinoBLE to NimBLE) very helpful.

4.4 Testing

The deliverables of this section

- Design PCBs for two fully autonomous sensor boards and submit for production.
- Develop software to read out your sensors nodes and send the data wirelessly to a database.
- Devise a test protocol to analyse the performance of your autonomous sensor nodes.

When your sensor nodes are fabricated, implement a wireless communication protocol to save the data to an online database. If your sensor nodes are faulty or you decide to proceed with this part of the lab before you own nodes return, you can use the department sensors nodes for further development.

Different decisions on how to collect the data will need to be made at this stage. The sensor data must be sampled carefully to avoid inaccurate readings and excessive power consumption. Oversampling can be computationally and energy inefficient, while undersampling (sampling frequency below the Nyquist criterion < $2 \times f_{max}$) will lead to data loss and aliasing. Thus, sampling rates should be tailored to the specific application—e.g., body temperature sensors may only need a few readings per minute. Data processing also consumes power, especially on MCUs. Offloading computations to cloud servers can reduce local processing needs, but transmitting data wirelessly is often the most power-intensive task. Therefore, could be more efficient to process data onboard and transmit only essential results—e.g., calculating heart rate from PPG data locally and sending just the heart rate. Then finally, interference by noise sources should be considered. Typical noise sources are power grid interference (~50 Hz in the UK), electrical circuit noise (can be minimised through good PCB design) and body movement noise (e.g. breathing, muscle activity, friction) need to be filtered out. Digital filters are useful but computationally demanding, outsourcing them to data hubs can improve power efficiency if it doesn't increase transmission load. Analogue filtering will also increase power load on the nodes and increases the node's footprint.

A sampling rate of 100 – 250 Hz for the PPG is sufficient for consumer wearable but must be much higher (~ 1000Hz for research or medical applications where signal fidelity is critical.

The sampling rate for the accelerometer depends on the application and will be limited by the chosen accelerometer technology. Some values are proposed in Table II.

Table II: Common sampling rates by use case.

Application	Typical Sampling Rate
Basic activity tracking	20–50 Hz

Application	Typical Sampling Rate
Pedometer / step counting	~100 Hz
Fall detection / motion events	100–400 Hz
Human activity recognition	50–100 Hz
Biomechanics / gait analysis	125–250 Hz
High-precision motion tracking	Up to 1000 Hz

An overview of filters typically used in IoT systems that can be implemented on the MCU, from the very simple to the complex:

- Moving Average Filters: Simple smoothing technique to reduce random noise.
- Polynomial fitting/detrending: fits a curve to the baseline and subtract it.
- Bandpass filters: remove high frequency noise and low frequency motion noise. Types:
    - Finite Impulse Response (FIR): Stable and linear phase; good for precise filtering.
    - Infinite Impulse Response (IIR): More efficient but can introduce phase distortion.
- Notch filters: target and remove specific frequencies like the 50 Hz power line interference.
- Kalman Filters: Advanced filters used for sensor fusion and motion tracking (e.g., combining accelerometer and gyroscope data).
- Adaptive Filters: Adjust their parameters based on signal characteristics (e.g., active noise cancellation).

Digital filters on the MCU must be lightweight, fast for real-time filtering and should preserve signal integrity.

Once you have designed and implemented your readout protocol you need to test the autonomous sensor nodes. The following parameters should be evaluated and reported on: resolution, accuracy, precision, power consumption, latency, transmission range. Devise a testing protocol and give evidence-based sensor performance.

To be submitted:
The data sheet of both your sensors containing – circuit schematic, PCB design and performance parameters. The report should not be longer than 10 pages. One report per lab group.

5. Task 2: Environmental sensing

The aim of this section is to make an environmental sensor system based on commercial sensor boards and gather data from different locations in London.

The deliverables of this task are:

- Develop the code to read all the sensor nodes, display them on the OLED and save the data to SD card.
- Design a compact PCB to replace wired connections. Use pin headers or sockets to connect the commercial boards to this PCB.
- Design and 3D print a box to contain your environmental sensor for safe deployment using the battery pack.
- Gather environmental data with your sensor board at different locations and compare to available data.

5.1 Introduction to Environmental Sensing

Kristel & Prabhav – 26/09/2025

Environmental sensing refers to the use of sensors and data acquisition systems to monitor physical, chemical, and biological conditions in the surrounding environment. These sensors can measure variables such as air quality, temperature, humidity, noise levels, radiation, and pollutant concentrations, among others. In urban settings like London, environmental sensing plays a crucial role in understanding and managing the impact of human activity on public health and ecological sustainability. Distributed sensor networks provide real-time, high-resolution data that can inform policy decisions, support scientific research, and empower communities to act. Breathe London is a city-wide air quality monitoring network managed by Imperial College London and offers real-time and historical data from over 400 sensor nodes. The data includes PM_1, $PM_{2·5}$, PM_{10}, NO_2, O_3, temperature, humidity and pressure and can be accessed via: Interactive maps, CSV/Excel downloads or via an API for developers.

Why is environmental sensing important?

1. Public Health Protection
   Monitoring air pollutants like NO_2 and $PM_{2·5}$ helps identify health risks and supports interventions to reduce exposure, especially for vulnerable populations.
2. Climate and Urban Planning
   Temperature and humidity data contribute to climate resilience strategies, such as designing green infrastructure and managing heat islands.
3. Regulatory Compliance
   Continuous sensing ensures that environmental standards are met and helps detect violations early.
4. Citizen Engagement
   Open access to environmental data fosters transparency and encourages public participation in sustainability efforts.
5. Smart City Development
   Integrating environmental sensors into urban systems enables adaptive responses to changing conditions, enhancing liveability and resource efficiency.

5.2 Building your environmental sensor system

This node will be relatively large as it will contain multiple sensors as well as data display and storage devices and will use an ESP32 development board. The boards can use I2C or SPI communication protocols.

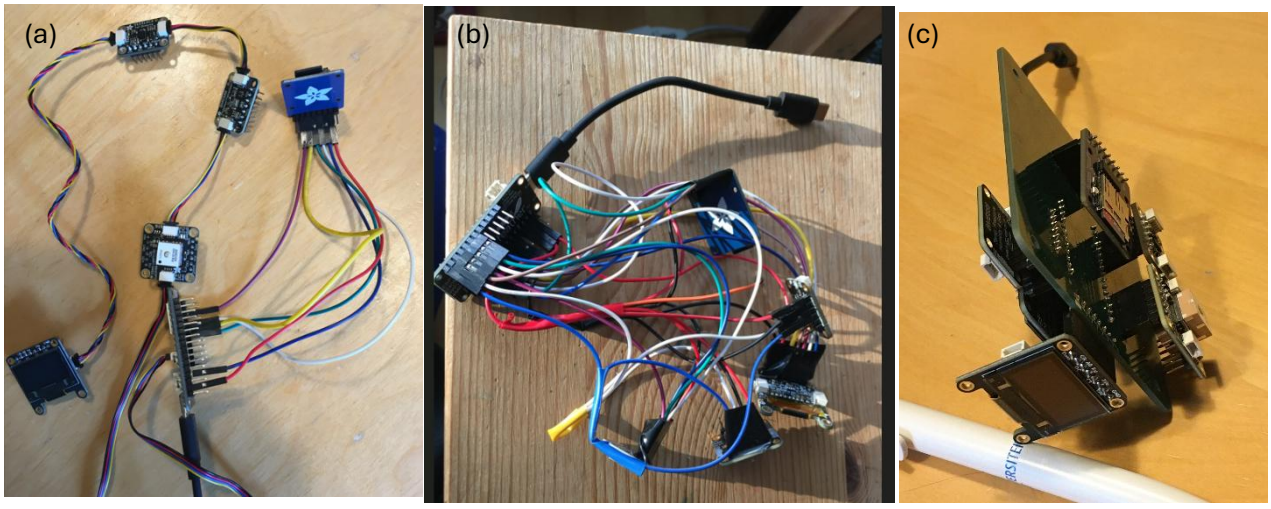


Figure 8: all boards are connected to the ESP32 MCU development board (a) using the qwiic connectors to daisy chain all possible boards, (b) using wires for I2C and SPI communication with the ESP, (c) using a custom-made board.

You have been given multiple commercial breakout boards: an environment sensor BME, a GPS module and a time-of-flight module. You have also been given an OLED display and a microSD card reader. You are strongly

advised to read up on how these sensors work. All these boards need to be connected to the ESP32 development board. All the boards are capable of I2C communication, apart from the SD card reader that operates using the SPI communication protocol. Some of the boards have both I2C and SPI capability. Most boards, including the MCU are fitted with a qwiic/stemma QT connector that supports I2C and allows you to daisy chain your different boards to the MCU with the special wires supplied in your pack, as long as they have different addresses (see Fig. 8a). In Fig. 8b the qwiic connectors are not used and all the boards are wired to the MCU GPIO pins by either I2C or SPI. In Fig. 8c the boards are all connected via a custom-made PCB on which the routing between boards and MCU is implemented. This is surely the least messy, most compact and most robust implementation. You need to implement your environment sensor system on PCB as shown in Fig. 8c. The PCB in Fig. 8c is only for illustration purposes and should not be copied.

A) Program the ESP32 to read out all sensors, display the readings on the OLED and save it to the SD card. This can be done by using Arduino IDE or PlatformIO or other proprietary software. This can be started from wired connections, but you must ensure that you can transfer your implementation to the system on PCB where are these wires are ansemt (cfr. Fig. 8c).
   Example code is available online for all commercial boards. See the Adafruit website for info as the boards are made by them. You will need to upload all the necessary libraries for the different Adafruit boards as well as the libraries for I2C and SPI communication. The library for the correct ESP32 development board also needs to be uploaded.
   It is probably best to take an approach of implementing each sensor separately and when it's working increase the number of sensors. Test by writing info to the serial monitor when using Arduino IDE. This will need to be removed in the final implementation.
   The OLED can be a bit slow in starting up and you should take that into account in your programming. The OLED screen is too small to display all the sensor information in one go and thus you will need to implement a solution to show all data.
   Write csv files to the SD card reader. USB-microSD cards convertors to download the data to your computer can be bought if you do not have one. Use excel or MATLAB to read and analyse the data.
   The GPS board comes across as slow since in the beginning it needs to localise different satellites. Depending on where you are and what time of the day, this can be either fast or slow. With the available battery the last satellite position can be retained for fast start-up. The GPS will not work indoors.
B) Design a compact PCB to route all connections between boards and MCU. The boards will be checked by Amine Halimi and will be ordered from JLCPCB via stores. You will need to solder the pin headers/sockets or other connectors yourself. Thus, connectors should be ordered via stores (who might have some connectors available). Since an enclosure needs to be fabricated for this PCB and the power bank, you are strongly encouraged to add the 3D CAD designs of all parts you are using to the PCB design such that you can visualize the geometry of the design. CAD designs can be downloaded from different sources such as the Adafruit github page, SnapEDA, GrabCAD, UltraLibrarian, 3DContentCentral. In the absence of available CAD designs, you will need to design your own for which you can use SolidWorks that will also be used to design the enclosure. An explanation of SolidWorks can be found in Section 5.3.
C) Design and print a box to house your PCB. Ensure that openings are made to allow access to USB, SD card, etc. Remember the time-of-flight sensor is optical. (see section 5.3).
D) Use the battery pack and the boxed environmental sensor system to take measurements at different locations inside and outside the department and compare the readings from your sensors to known data. An atmospheric sensor will be made available in the lab and data from environment sensors in London can be downloaded. Analyse your readings and any discrepancies.

5.3 Design and 3D print enclosure

To protect your circuit while monitoring the environment in different places in and around College you need to design and print an enclosure to hold your PCB with all sensor boards connected as well as your battery pack.

Kristel & Prabhav – 26/09/2025
Most sensors cannot be fully covered by the enclosure as they need access to the environment. Thus you enclosure will need to accommodate for this.
SOLIDWORKS is a 3D design software that is used to design all sorts of products. It is a great resource for teaching 3D mechanical CAD, design validation, and much more. You can get access to SOLIDWORKS via College ICT. Learning guides are available via the Solidsolutions website. Your print job will need to be submitted to the EEE Tech Services print queue. Before printing, Amine Halimi will check the feasibility of the print (but not check whether everything is correct for your particular PCB layout).

Some items to consider when designing an enclosure for an environmental sensor PCB that also holds a battery.

1. Initial Setup
    - Import accurate PCB dimensions (DXF, STEP, or manually measured)
    - Include all component placements and heights (OLED, sensors, connectors, battery)
    - [Set correct units (mm or inches) and tolerances for 3D printing
2. Enclosure Geometry
    - Define overall enclosure dimensions with clearance for components and wiring
    - Add mounting standoffs for PCB (check hole sizes and spacing)
    - Include internal supports or compartments for battery pack
    - Design snap-fit features or screw bosses for assembly
3. Openings and Interfaces
    - OLED display window or cutout with bezel/lip
    - Time-of-flight sensor aperture or transparent window
    - SD card slot with finger clearance
    - BME680 vent holes or mesh for airflow
    - USB or power connector access
4. Thermal and Environmental Considerations
    - Ventilation holes or slots near heat-generating components
    - Avoid enclosing environmental sensors in sealed compartments
    - Consider printing material thermal properties
5. Cable and Component Clearance
    - Check clearance for headers, and connectors
    - Avoid sharp corners near cables or battery leads
    - Include strain relief features if cables exit the enclosure
6. Assembly and Maintenance
    - Ensure easy insertion/removal of PCB and battery
    - Avoid tight fits that require force
    - Design for tool access if screws are used
    - Consider modularity for future upgrades
7. 3D Printing Preparation
    - Orient parts to minimize overhangs and supports
    - Add fillets/chamfers to reduce stress and improve print quality
    - Check wall thickness (≥1.2 mm)
    - Export STL with correct resolution and units
8. Labelling and Aesthetics
    - Add embossed or engraved labels for ports/sensors
    - Ensure intuitive orientation (e.g., display facing up)
    - Consider surface texture or branding elements

To be demonstrated to lab assessors: The environment sensor system within its enclosure + demonstration of its operation.

To be submitted:
The data of the environmental sensor experiment compared to its control and critically analysed for impact on public health. The report should not be longer than 5 pages and should contact a link to github page with the code, enclosure design and the data. One report per lab group.

6. Task 3: Measuring the height of the EEE building

In this last task you need to integrate all sensors you made, the environmental sensor and the autonomous sensor nodes and send the data of all these sensors wirelessly to a database (see Fig. 9 for a possible implementation). Of course, you can also consider sending all sensor data via BLE to your phone. Your smartphone will then act as a gateway to a cloud database. Ready made apps are available online, such as nRF Connect, Blynk or IoT platforms that can be used.

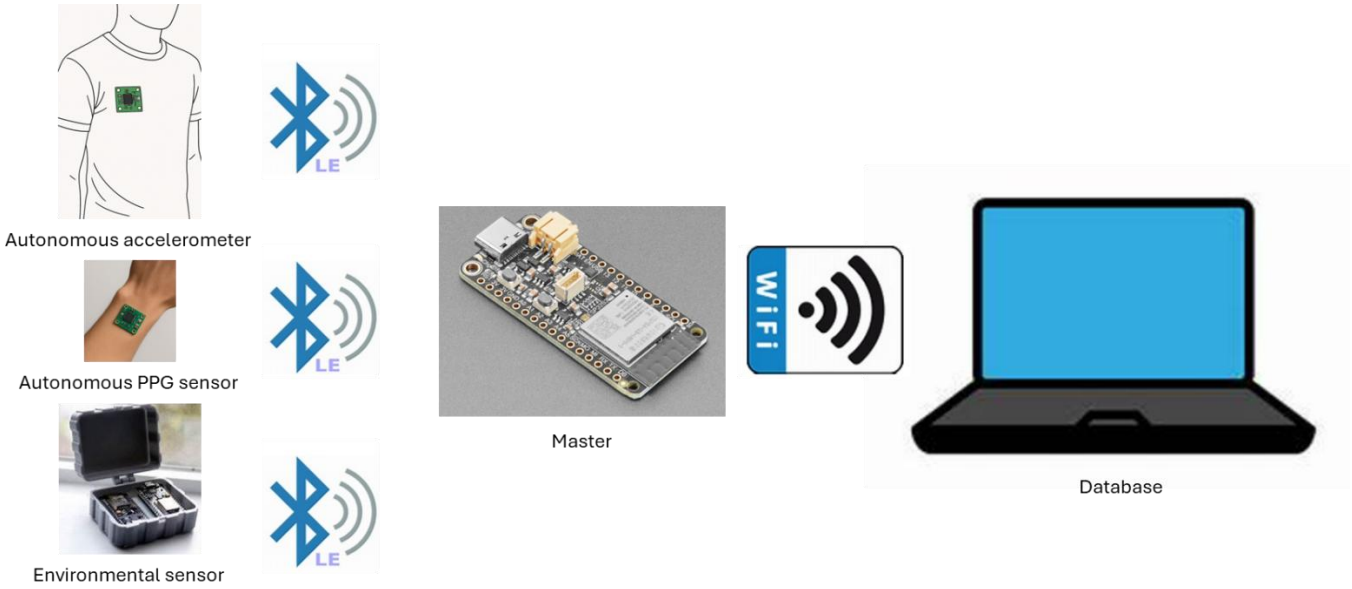


Figure 9: A possible architecture for transferring the sensor data of all sensor nodes to a computer.

Your task is now to use as many sensors and sensor combinations as possible to estimate the height of the EEE building. List all the approaches you can imagine, apply them to estimate the height of the building and give the pros and cons of the approach clearly explaining why a certain approach might give accurate/inaccurate results.

The group with the most appropriate approaches will win the MSc Sensor System lab prize.

To be presented to the whole class:
Powerpoint presentation with findings.