UDC 638.1:621.3:004.8

REMOTE MONITORING SYSTEM FOR MICROCLIMATE PARAMETERS IN BEEHIVES

Yurii Zborivskyi**, Bohdan Koman***
Ivan Franko National University of Lviv,
50 Dragomanova St., UA–79005 Lviv, Ukraine

Zborivskyi, Y. A., Koman, B. K. (2025). Remote Monitoring System for Microclimate Parameters in Beehives. *Electronics and Information Technologies*, 31, 123–136. https://doi.org/10.30970/eli.31.11

ABSTRACT

Background. The health and productivity of bee colonies have a direct impact on global food security, as bees contribute to the pollination of approximately 50% of the world's plant-based food resources. Traditional hive observation methods lack the precision and continuity required for effective colony management. This creates an urgent need for sophisticated remote monitoring solutions capable of continuously tracking critical environmental parameters within beehive ecosystems, including temperature, humidity, and air quality.

Materials and Methods. This study presents the development of an integrated internet-based monitoring system centered on a microcontroller platform. The system incorporates a comprehensive sensor array for multi-parameter data acquisition: gas sensors for detecting harmful substances, a temperature sensor, a particulate matter detector, and an environmental parameter sensor. Real-time data processing is achieved through a multitasking architecture, enabling concurrent sensor polling, data validation, and wireless transmission.

Results and Discussion. Comprehensive system testing validated reliable data acquisition across all monitored parameters. Temperature measurements demonstrated operational ranges from 13.75°C to 32.69°C for internal hive conditions, while humidity levels varied between 51.70% and 88.10%. Gas concentration monitoring showed stable baseline readings with normal conditions at 1.71 parts per million. Statistical correlation analysis revealed significant environmental interdependencies, including a strong positive correlation between gas sensor measurements and a pronounced negative correlation between temperature and humidity parameters, confirming expected environmental relationships within the hive microclimate.

Conclusion. The developed IoT-based beehive monitoring system, built on the ESP32 microcontroller, demonstrated stable and energy-efficient performance during continuous field testing. The system recorded internal hive temperatures ranging from 13.75 °C to 32.69 °C, humidity levels from 51.70% to 88.10%, and gas concentrations of 1.71 ppm (MQ-6) and 0.37 ppm (MQ-135) under normal conditions.

Keywords: Beehive monitoring, IoT, embedded systems, microclimate parameters, cloud computing, real-time data.

INTRODUCTION

Monitoring physical and environmental parameters of inaccessible technological processes, remote devices, and biological systems remains a major technical challenge for researchers. One such application is monitoring the bioenvironment within beehives. It is estimated that over half of global plant-based food production depends, directly or indirectly, on bee pollination, yet human activities often negatively affect colony viability and productivity. Traditional empirical observation and manual inspection methods are insufficient to ensure reliable bee colony health and productivity. Modern beehive monitoring systems enable continuous observation and control of key environmental parameters such as temperature, humidity, air quality, and particle concentration that influence bee health and productivity. These systems help predict optimal conditions for beekeeping.

Recent studies have proposed various IoT-based beehive monitoring solutions that combine environmental sensing, data analytics, and wireless communication to support sustainable apiculture. Andrijević et al. [1] developed an IoT system capable of real-time monitoring and prediction of bee activity with integrated alarm functions. Zhang et al. [2] proposed an intelligent monitoring platform combining IoT data acquisition with colony state analysis to identify abnormal hive behavior. Kurdin and Kurdina [3] introduced a smart beehive network using homogeneous data modeling for forecasting honey robbing phenomena, while Zabasta et al. [4] presented a low-power IoT system emphasizing energy efficiency and safety in beekeeping. Earlier work by Ali and Raza [5] demonstrated a general IoT environmental monitoring concept, but without hive-specific optimization.

However, most existing IoT beehive systems are constrained by limited multitasking capability, high power consumption, or a lack of stable real-time data synchronization. The system proposed in this study overcomes these limitations by integrating multiple environmental sensors under a multitasking FreeRTOS architecture on the ESP32 platform. This approach enables parallel data acquisition, validation, and wireless transmission, achieving high temporal resolution and improved energy efficiency. The modular design and optimized firmware allow scalability, stability, and long-term autonomous operation, which are essential for remote or large-scale apiary deployments.

This work aims to develop the architecture and circuit implementation of a computerized system for remote monitoring of beehive environmental parameters, with real-time data transmission to mobile devices. The system targets apiary farms that require continuous monitoring of hives, research projects analyzing environmental impacts on bee colonies, and broader ecological studies, as bees serve as reliable indicators of ecosystem health [6].

MATERIALS AND METHODS

1. Functional Diagram of the Parameters Monitoring System

The monitoring system (see **Fig. 1**) integrates a suite of sensors (MQ-6, MQ-135, DS18B20, PMS5003, DHT21) [8-13] with the ESP32 microcontroller to collect, process, and transmit data. The hardware connects sensors to the ESP32, which powers them, reads signals, and transmits data wirelessly via Wi-Fi. The software implements algorithms for sensor polling, data processing, and transmission to an external interface for beekeeper analysis.

The ESP32 microcontroller [7] is selected for its built-in Wi-Fi module, low power consumption (especially in sleep mode), and dual-core processor supporting multitasking. These features enable efficient sensor management and parallel data processing.

The system operates through the following stages:

1. Initialization Stage: Upon powering on, the ESP32 configures all components, initializes data transmission interfaces (UART, SPI, 1-Wire), calibrates MQ-6 and MQ-135 gas sensors, and sets up the TFT display via SPI.
2. Data Collection Stage: The controller cyclically polls sensors at set intervals. Analog data from MQ-6 and MQ-135 are read via ADC, DS18B20 temperature via 1-Wire, PMS5003 dust data via UART, and DHT21 temperature/humidity via a single-wire interface.

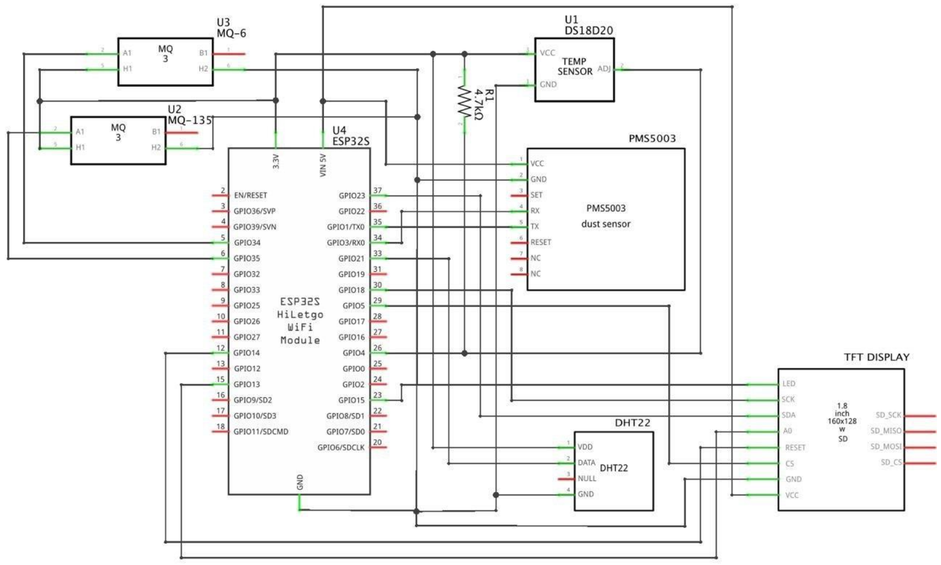

Fig. 1. Block diagram of the centralized data collection and processing system based on the ESP32 controller.

3. Initialization Stage: Upon powering on, the ESP32 configures all components, initializes data transmission interfaces (UART, SPI, 1-Wire), calibrates MQ-6 and MQ-135 gas sensors, and sets up the TFT display via SPI.
4. Data Collection Stage: The controller cyclically polls sensors at set intervals. Analog data from MQ-6 and MQ-135 are read via ADC, DS18B20 temperature via 1-Wire, PMS5003 dust data via UART, and DHT21 temperature/humidity via a single-wire interface.
5. Data Processing Stage: Data undergoes noise filtering, averaging, and validity checks. Gas sensor data is adjusted with calibration coefficients and temperature compensation, while PMS5003 data is verified using checksums.
6. Data Output and Transmission Stage: Processed data is displayed on the TFT screen in a user-friendly format and transmitted via Wi-Fi to a remote server. Critical value detection triggers warning signals.
7. Energy Saving Stage: Between polling cycles, the ESP32 enters low-power mode to maintain minimal activity for critical components and threshold monitoring.

This modular structure supports reliable operation, scalability, and the addition of new sensors as needed.

2. Main Interfaces

The beehive monitoring system relies on a variety of communication interfaces to connect the ESP32 microcontroller with sensors and a display. Each interface is tailored to the specific needs of the connected hardware, ensuring efficient and accurate data collection for monitoring beehive environmental conditions.

Analog Interface (MQ-6, MQ-135): These gas sensors change resistance upon gas exposure, altering output voltage. The ESP32's ADC converts this voltage (0–3.3V) to a digital value (0–4095). Software calibration uses sensor characteristic curves to compute gas concentration, enabling precise detection of gases like LPG, butane, and air pollutants.

1-Wire Interface (DS18B20): This single-wire bidirectional interface uses a 4.7kΩ pull-up resistor and time-slot-based data transmission. Each DS18B20 has a unique 64-bit code, enabling multiple sensors on one line, which simplifies wiring and supports scalable temperature monitoring inside the hive.

UART Interface (PMS5003): This asynchronous serial interface operates at 9600 baud, transmitting dust concentration data (PM1.0, PM2.5, PM10) in fixed-length frames with checksums for integrity, ensuring reliable measurement of particulate matter affecting hive air quality.

Single-Wire Digital Interface (DHT21): The DHT21 uses a proprietary protocol, sending 40-bit packets (16-bit humidity, 16-bit temperature, 8-bit checksum) after a request signal from the ESP32, providing accurate external temperature and humidity data with minimal wiring.

SPI Interface (TFT Display): This synchronous interface uses MOSI, MISO, SCK, and CS lines, with additional DC and RESET lines for data/command indication and display reset, facilitating high-speed data transfer for real-time visualization of sensor readings.

3. Software Architecture of the Monitoring System

3.1 General Overview of the Software Architecture

The software architecture leverages the ESP32 microcontroller’s dual-core processor to handle multiple tasks efficiently. It employs the FreeRTOS operating system to manage concurrent processes, ensuring reliable system performance. The main tasks include collecting sensor data, updating the TFT display, and transmitting data to a remote server via Wi-Fi. These tasks are distributed across the ESP32’s two cores to optimize performance: Core 1 manages sensor data collection and display updates to ensure timely user feedback, while Core 0 handles Wi-Fi communication to isolate potential network delays and prevent interference with other processes. A queue system facilitates asynchronous data exchange between tasks, enhancing system stability and responsiveness by allowing each task to operate independently without delays from other processes.

3.2 System Initialization

The initialization process begins with the setup() function executing upon microcontroller startup, systematically configuring all system components for operation.

Sensor Configuration. The sensor initialization process starts with configuring the DHT21 sensor using the DHT.h library to enable temperature and humidity data acquisition. The DS18B20 temperature sensor requires setup through both OneWire.h and DallasTemperature.h libraries, providing precise temperature measurements via the 1-Wire protocol. For gas detection, both MQ-6 and MQ-135 sensors are initialized using the MQUnifiedsensor.h library, with calibration performed by averaging 10 measurements in clean air to establish a baseline resistance (Ro). The PMS5003 dust particle sensor connects via the UART port to collect particulate matter data.

Display and Network Configuration. The TFT display initialization utilizes the TFT_eSPI.h library, specifying essential pins including CS, DC, RST, SDA, and SCL while setting the text format to white on a black background with font size 3 for optimal readability. Network connectivity is established through the WiFi.h library, create a connection using predefined SSID and password credentials to enable server communication.

Task Management Setup. The system creates a sensorDataQueue of type QueueHandle_t capable of storing up to five SensorData structure instances, facilitating efficient data transfer between concurrent tasks. Three primary tasks are launched using the xTaskCreatePinnedToCore() function with strategic core allocation. The sensor reading task and display update task both operate on Core 1, handling data collection from sensors and updating the TFT display with current readings, respectively. The Wi-Fi communication task runs independently on Core 0, managing data transmission to the remote server. This core allocation strategy optimizes processor resources and ensures smooth operation by preventing interference between critical processes.

3.3 Data Structure

The SensorData structure organizes sensor readings for efficient storage and transfer between tasks, containing:

- temperatureDHT: External temperature in degrees Celsius from the DHT21 sensor.
- humidity: Humidity percentage from the DHT21 sensor.
- temperatureDS18B20: Internal temperature in degrees Celsius from the DS18B20 sensor.
- mq6_ppm: Gas concentration in parts per million from the MQ-6 sensor.
- mq135_ppm: Gas concentration in parts per million from the MQ-135 sensor.
- pm1_0, pm2_5, pm10: Dust particle concentrations in micrograms per cubic meter for 1.0, 2.5, and 10 micrometer particles from the PMS5003 sensor.

A separate PMSData structure isolates dust data (pm1_0, pm2_5, pm10) to enhance code modularity and simplify processing.

3.4 Sensor Read Task

The sensorReadTask() function operates as the core data acquisition module, continuously collecting sensor measurements at 6-second intervals. The task implements precise timing control through the vTaskDelay(pdMS_TO_TICKS(6000)) function, which temporarily suspends execution to allow other system processes to run efficiently, thereby preventing processor overload and maintaining optimal system performance.

Data Collection Process. The data acquisition cycle begins with environmental measurements from the DHT21 sensor, reading external temperature through dht.readTemperature() and humidity via dht.readHumidity() functions. Simultaneously, the DS18B20 temperature sensor performs internal hive temperature monitoring by initiating measurement with sensors.requestTemperatures() and subsequently retrieving the temperature value using sensors.getTempCByIndex(0).

Gas concentration monitoring involves updating the sensor states through MQ6.update() and MQ135.update() functions, followed by precise concentration readings using MQ6.readSensor() and MQ135.readSensor() methods after proper calibration adjustments. The PMS5003 particulate matter sensor contributes air quality data through UART communication via the readPMS5003() function, with built-in data integrity verification to ensure measurement reliability.

Data Management and Transfer. Upon completion of each measurement cycle, all sensor readings are systematically organized into a SensorData structure instance. This consolidated data package is then transmitted to the sensorDataQueue using the xQueueSend() function, enabling asynchronous access by other system tasks. This queue-based approach ensures efficient inter-task communication while maintaining data integrity and preventing bottlenecks in the overall system architecture.

3.5 Display Update Task

The displayUpdateTask() function displays sensor data on the TFT screen: Data Retrieval: Uses xQueueReceive() to fetch data from sensorDataQueue, activating only when new data is available to conserve resources.

Display Process: Clears the screen with tft.fillScreen(TFT_BLACK) and displays:

- Internal temperature in degrees Celsius (DS18B20)
- External temperature in degrees Celsius (DHT21)
- Humidity percentage (DHT21)
- Gas concentrations in parts per million (MQ-6, MQ-135)
- Dust levels in micrograms per cubic meter (PM1.0, PM2.5, PM10)

This ensures users receive clear, up-to-date environmental information.

3.6 WiFi Communication Task

The wifiCommunicationTask() function, executed on Core 0 of the ESP32, manages the transmission of collected sensor data to a remote server via Wi-Fi, enabling real-time storage and analysis outside the local system. It continuously monitors the Wi-Fi connection status using WiFi.status() == WL_CONNECTED, automatically reconnecting with WiFi.begin() if the connection is lost, ensuring resilience against network disruptions. The task retrieves the latest sensor readings from the sensorDataQueue using xQueueReceive(), maintaining efficient data access without interfering with other tasks. These readings, stored in the SensorData structure, are formatted into a JSON string, a universal format for seamless data exchange. Using the HTTPClient.h library, the task initiates an HTTP POST request to the server via http.begin(serverName), sends the JSON data, and awaits a response. Upon successful transmission (response code > 0), it logs a confirmation message with the server's response; otherwise, it displays the error code for diagnostics. This process ensures reliable and secure data delivery, supporting remote monitoring and analysis of beehive environmental conditions.

4. Server-Side Monitoring System

The server-side component of the system is a web application based on the Flask framework, deployed on the AWS EC2 platform. The application provides reception, processing, and visualization of beehive monitoring data.

According to the diagram in **Fig. 2**, the system consists of three main blocks:

- Sensor block, which includes a set of sensors for measuring various environmental parameters. Each sensor is connected to the controller via a corresponding interface: analog for gas sensors, digital for temperature and humidity sensors, UART for the dust particle sensor.
- Controller block, based on ESP32, it processes data from all sensors and ensures their transmission via the WiFi module. The controller also manages a local TFT display via the SPI interface for displaying current readings.
- Server block, deployed on AWS, it includes an EC2 instance with a Flask server for data processing, an AWS RDS database for storing information, and a web interface for visualizing monitoring data.

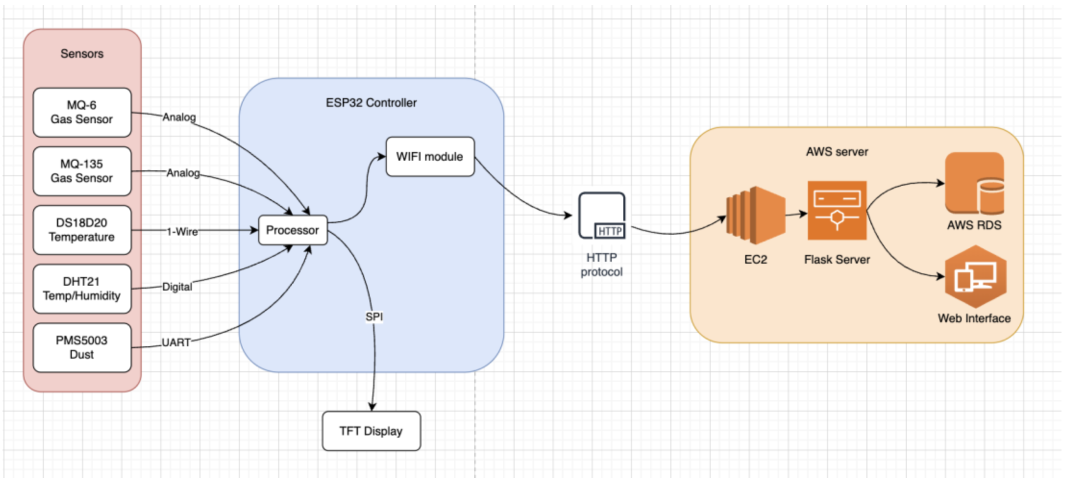

Fig. 2. System Architecture Diagram.

4.1 Principle of Server Architecture Operation

Deployed on an AWS EC2 virtual machine, the Flask-based web application in Python orchestrates the beehive monitoring system's data management with high efficiency. The server receives sensor data in JSON format from the ESP32 microcontroller via secure HTTP requests, ensuring reliable transmission over Wi-Fi. It validates and formats incoming data to eliminate errors, applying filters to ensure accuracy before processing. The processed data, tagged with precise timestamps, is stored in an AWS RDS database, which supports scalable, long-term storage and enables trend analysis for hive health monitoring. The Flask application retrieves this data to populate a dynamic, browser-based web interface, allowing beekeepers to access real-time and historical environmental parameters, such as temperature, humidity, and gas levels, from any internet-connected device. This architecture optimizes data flow from collection to visualization, with robust error handling and secure data transfer protocols to protect sensitive hive information. The server's design leverages AWS's scalability to handle multiple hives simultaneously, making it suitable for large-scale apiaries and research applications, while its modular structure supports future enhancements, such as integrating machine learning for predictive hive management.

4.2 AWS RDS (Relational Database Service)

AWS RDS is a managed relational database service used to store all monitoring data. It supports popular database management systems like MySQL, PostgreSQL, or MariaDB, providing flexibility in technology choice.

Role of the database in the system:

- Data Storage: All sensor readings (temperature, humidity, gases, dust) are stored in the database with timestamps. For example, a table might have columns: timestamp, temperature, humidity, gas_level, dust_particles.
- Historical Analysis: Thanks to the relational structure, users can perform queries to analyze trends (e.g., temperature change over a week) or detect anomalies.
- Fast Access: RDS provides efficient data access for the Flask server, allowing the web interface to quickly display information.

4.3 Web Interface

The web interface is a browser-based application that allows users to view monitoring data in a clear and convenient format. It is built using HTML5, TailWind, JavaScript, and the Chart.js library for visualization.

These three components together provide a complete cycle: from receiving data over the internet to displaying it on the screen of your phone or computer. The system is designed to be convenient, stable, and accessible from anywhere with internet access.

The system's web interface displays data in real-time using interactive graphs. A separate graph is built for each monitoring parameter (temperature, humidity, gases, particles). The system automatically calculates basic statistical indicators: mean values, minimums, maximums, and standard deviations for each parameter.

5. Examples of testing the developed system and the obtained results

This section presents examples of testing the results obtained and the developed monitoring system, illustrated by the following figures:

Internal Hive Temperature (DS18B20): Temperature readings from the DS18B20 sensor (see **Fig. 3**) initially showed an artificial increase to 24.63°C around 05:30 for testing purposes. The temperature then dropped to 13.75°C around 08:30 as the sensor began operating inside the hive. Subsequently, it increased steadily, reaching 32.69°C around 12:45.

External Humidity (DHT21): External humidity measurements from the DHT21 sensor (see **Fig. 4**) showed an artificial spike to 88.10% around 05:15 during the testing phase. The humidity later peaked at 65.90% around 08:30 and then gradually decreased to a minimum of 51.70% around 12:00, indicating typical sensor behavior in natural conditions.

External Temperature (DHT21): External temperature measurements from the DHT21 sensor (see **Fig. 5**) recorded an artificial temperature increase to 22.50°C around 05:30. Afterward, the temperature dropped to 14.40°C around 08:30 and gradually rose to 19.50°C by 13:00, representing realistic ambient dynamics.

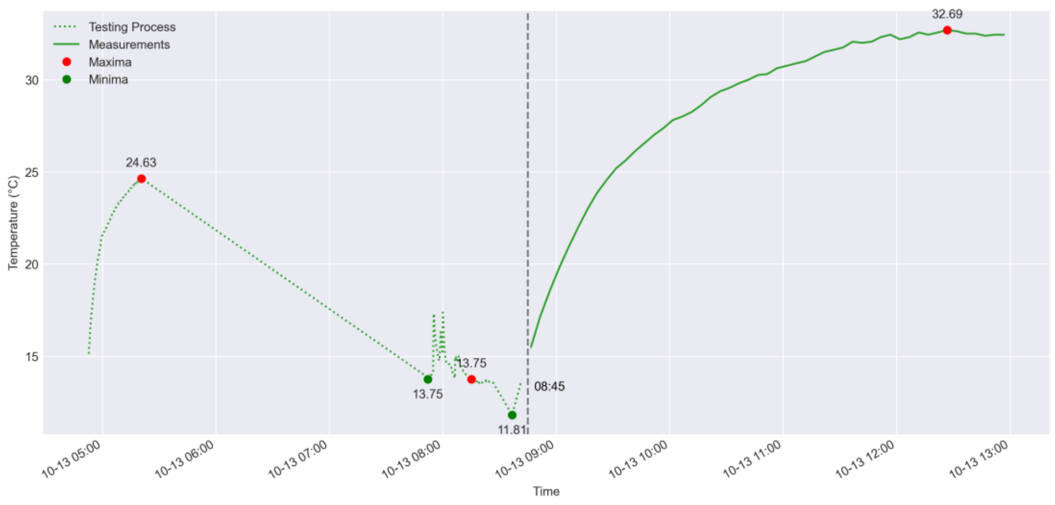

Fig. 3. Graph of internal hive temperature measurement (DS18B20).

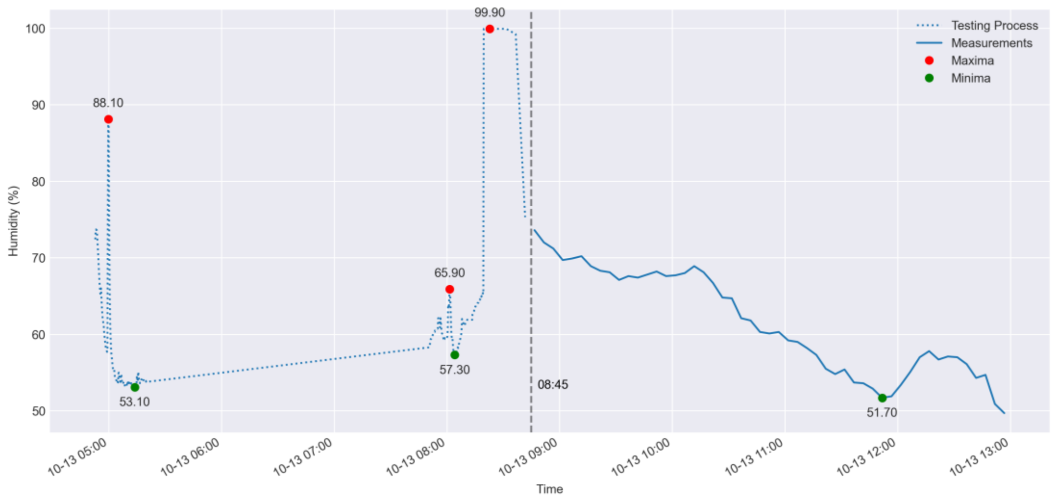

Fig. 4. Graph of external humidity measurement (DHT21).

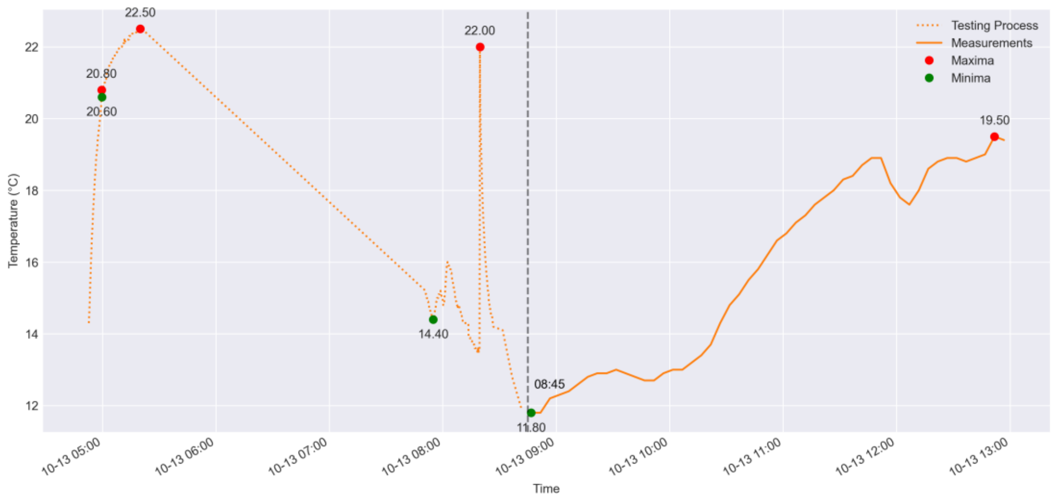

Fig. 5. Graph of external temperature measurement (DHT21).

Gas Concentration (MQ6): To verify the operation of the MQ6 sensor, an artificial gas concentration spike to 640.69 ppm was introduced around 05:15 (see **Fig. 6**). Following this test, the level dropped sharply to 3.37 ppm around 08:30 and stabilized at 1.71 ppm by 13:00, reflecting a return to normal environmental levels.

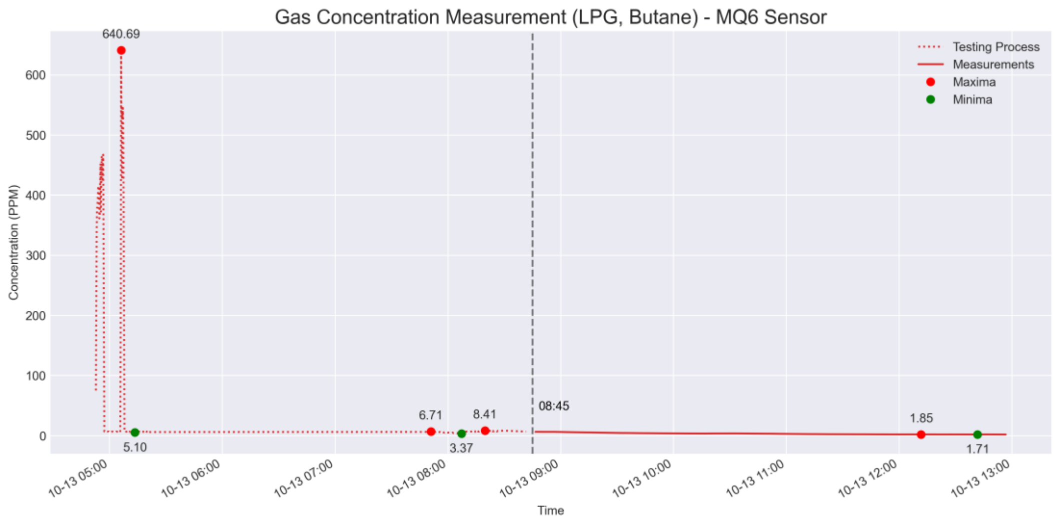

Fig. 6. Graph of gas concentration measurement from MQ6 sensor (LPG, butane).

Air Pollutants Concentration (MQ135): The MQ135 sensor initially detected an artificial pollutant concentration of 204.81 ppm around 05:15 (see **Fig. 7**). The value then sharply dropped to 0.95 ppm around 08:30 and continued to decrease, reaching 0.37 ppm by 13:00, indicating stable, low-level air quality conditions.

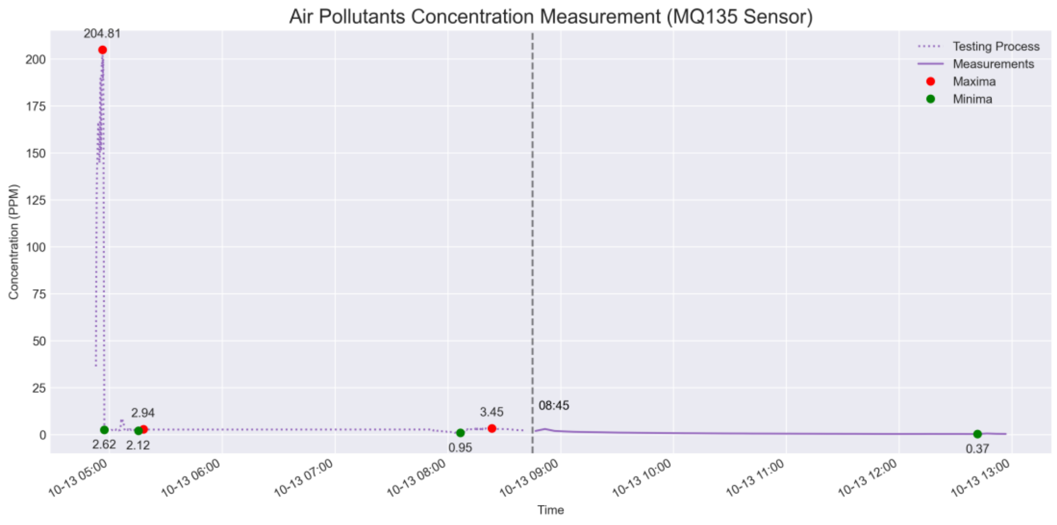

Fig. 7. Graph of gas concentration measurement from MQ135 sensor (air quality).

The photo shows (see **Fig. 8**) the installed beehive monitoring system in operation. The ESP32-based controller with connected sensors and antenna is powered by an external battery and is actively collecting environmental data inside the hive.

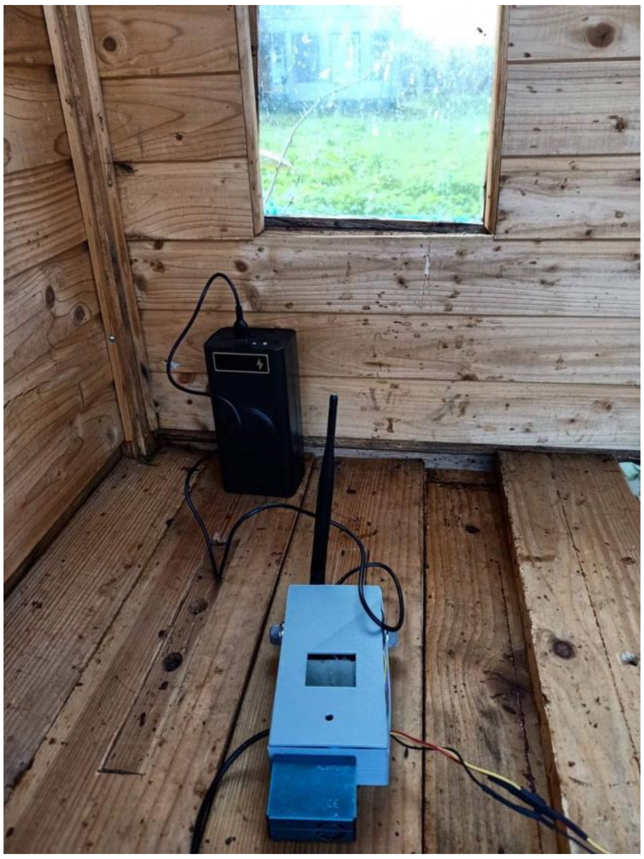

Fig. 8. General view of the beehive parameter monitoring system in operation.

5.1 Correlation analysis of sensor data

The analysis is based on the experimental data (see **Table 1**) collected during the "Measurements" period (After 08:45, October 13). The input dataset for the correlation matrix includes 51 data points collected over a duration of approximately 4 hours and 10 minutes (from 08:46:40 to 12:56:48 on October 13).

The Pearson correlation coefficient formula was used for calculations:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\ \sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}},$$

where x_i and y_i are the values of two variables, $\overline{x}$ and $\overline{y}$ are their mean values, and *n* is the number of observations.

For a deeper understanding of the relationships between various hive parameters, a correlation analysis (see **Fig. 9**) of the obtained data was conducted. The scientific purpose of this analysis is to confirm the expected physical and ecological interdependencies within the hive microclimate.

- Strong positive correlation is observed between mq6 ppm and mq135 ppm (0.918269), indicating a close relationship between gas concentrations, as well as between temperature DHT and temperature DS18B20 (0.859574), confirming consistent temperature measurements by both sensors.

Table 1. **The first 10 rows of raw experimental data were utilized to construct the correlation matrix.**

timestamp	temperatureDHT	humidity	temperatureDS18B20	mq6_ppm	mq135_ppm
2024-10-13 8:46:40	11.8	73.6	15.5	6.16	2
2024-10-13 8:51:41	11.8	72	17.19	6.06	3.04
2024-10-13 8:56:41	12.2	71.2	18.56	6.08	2.02
2024-10-13 9:01:41	12.3	69.7	19.81	5.79	1.76
2024-10-13 9:06:41	12.4	69.9	20.94	5.5	1.55
2024-10-13 9:11:41	12.6	70.2	22	5.26	1.44
2024-10-13 9:16:41	12.8	68.9	23	5.07	1.35
2024-10-13 9:21:41	12.9	68.3	23.87	4.74	1.28
2024-10-13 9:26:42	12.9	68.1	24.56	4.51	1.19
2024-10-13 9:31:42	13	67.1	25.19	4.25	1.1

- Strong negative correlation is found between temperature DHT and humidity (-0.974651), as well as between temperature DS18B20 and mq6 ppm (-0.979276), indicating an inverse relationship between temperature and humidity, and between temperature and LPG/butane gas concentration.
- Moderate negative correlation is observed between temperature DHT and mq135 ppm (-0.764955) and between temperature DS18B20 and mq135 ppm (-0.956536), indicating a decrease in pollutant concentration as temperature increases.

CONCLUSIONS

The developed beehive monitoring system based on the ESP32 microcontroller has proven effective for continuous data acquisition under real conditions. During testing, the system successfully recorded key operational ranges, including internal hive temperatures from 13.75 °C to 32.69 °C, humidity levels between 51.70 % and 88.10 %, and stable gas concentrations of approximately 1.71 ppm (MQ-6 sensor, LPG/butane) under normal conditions.

The multitasking FreeRTOS architecture ensured simultaneous sensor polling, Wi-Fi data transmission, and real-time display updates without observable data loss or performance degradation.

These results confirm the stability, accuracy, and practical applicability of the proposed IoT-based system for real-time environmental monitoring in apiaries.

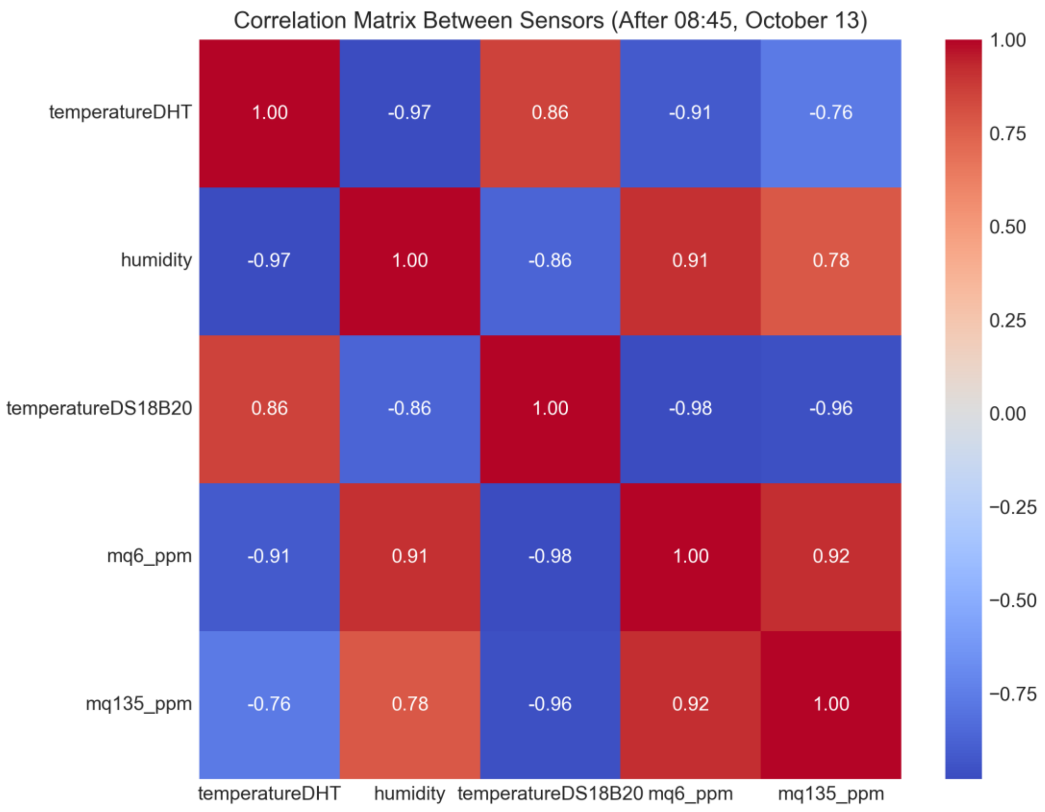

Fig. 9. Correlation matrix of beehive monitoring parameters.

Future research will focus on three main directions:

- Hardware enhancement — integrating additional sensors such as weight, CO_2, and TVOC to obtain a more comprehensive view of hive microclimate and colony health.
- Data analytics — applying machine learning methods for predictive modelling to forecast swarming events, detect disease and stress anomalies, and predict honey yield.
- Energy autonomy — implementing solar power solutions to achieve fully autonomous operation suitable for long-term, remote deployments.

ACKNOWLEDGMENTS AND FUNDING SOURCES

The authors received no financial support for the research, writing, and/or publication of this article.

COMPLIANCE WITH ETHICAL STANDARDS

The authors declare that they have no competing interests.

AUTHOR CONTRIBUTIONS

Conceptualization, [Y.Z.]; methodology, [B.K.]; validation, [Y.Z.]; formal analysis, [Koman B. H.].; investigation, [Y.Z.]; resources, [Y.Z.]; data curation, [Y.Z.]; writing – original draft preparation, [Y.Z.]; writing – review and editing, [B.K.]; visualization, [Y.Z.]; supervision, [B.K.]; project administration, [*Koman B. H*]; funding acquisition, [Y.Z.].

All authors have read and agreed to the published version of the manuscript.

REFERENCES

[1] Andrijević, N., Urošević, V., Arsić, B., Herceg, D., & Savić, B. (2022). IoT monitoring and prediction modeling of honeybee activity with alarm. Electronics, 11(5), 783. https://www.mdpi.com/2079-9292/11/5/783

[2] Zhang, Y., Wang, Z., Liu, Y., & Zhang, Y. (2024). Intelligent beehive monitoring system based on internet of things and colony state analysis. Smart Agricultural Technology, 8, 100489. https://www.sciencedirect.com/science/article/pii/S2772375524001898

[3] Kurdin, I., & Kurdina, A. (2025). Internet of things smart beehive network: Homogeneous data, modeling, and forecasting the honey robbing phenomenon. Inventions, 10(2), 23. https://www.mdpi.com/2411-5134/10/2/23

[4] Zabasta, A., Kunicina, N., Kondratjevs, K., & Ribickis, L. (2019). An internet of things-based low-power integrated beekeeping safety and conditions monitoring system. Inventions, 4(3), 52. https://www.mdpi.com/2411-5134/4/3/52

[5] Ali, M., & Raza, B. (2014). Internet of things based intelligent monitoring system for environment. Procedia Computer Science, 37, 376–381. https://www.sciencedirect.com/science/article/pii/S1877050914015804

[6] Seeley, T. D., & Towne, W. F. (2000). Tactics of dance choice in honey bee recruitment. Proceedings of the National Academy of Sciences, 97(16), 8629–8632. https://www.pnas.org/doi/full/10.1073/pnas.262413599

[7] Espressif Systems. (2023). ESP32 series datasheet v4.9 [Datasheet]. https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

[8] Winsen Electronics. (n.d.). MQ-6 gas sensor datasheet [Datasheet]. https://www.winsen-sensor.com/d/files/semiconductor/mq-6.pdf

[9] Hanwei Electronics. (n.d.). MQ-135 gas sensor datasheet [Datasheet]. https://www.olimex.com/Products/Components/Sensors/Gas/SNS-MQ135/resources/SNS-MQ135.pdf

[10] Analog Devices. (n.d.). DS18B20 programmable resolution 1-wire digital thermometer [Datasheet]. https://www.analog.com/media/en/technical-documentation/data-sheets/ds18b20.pdf

[11] Plantower Technology. (2016). PMS5003 series manual [Datasheet]. https://www.aqmd.gov/docs/default-source/aq-spec/resources-page/plantower-pms5003-manual_v2-3.pdf

[12] Aosong Electronics. (n.d.). DHT21 (AM2301) digital temperature and humidity sensor [Datasheet]. https://mikroshop.ch/pdf/DHT21.pdf

[13] Principles of construction of hybrid microsystems for biomedical applications. Dzundza , B.S., Kohut, I.T., Holota, V.I., Turovska, L.V., Deichakivskyi, M.V. – Physics and Chemistry of Solid State, 2022, 23 (4), pp.776-784. https://doi.org/10.15330/pcss.23.4.776-784

СИСТЕМА ДИСТАНЦІЙНОГО МОНІТОРИНГУ ПАРАМЕТРІВ МІКРОКЛІМАТУ У ВУЛИКАХ

Юрій Зборівський*, *Богдан Коман
Львівський національний університет імені Івана Франка
вул. Драгоманова, 50, Львів, Україна

АНОТАЦІЯ

Вступ. Здоров'я та продуктивність бджолиних колоній безпосередньо впливають на глобальну продовольчу безпеку, оскільки бджоли сприяють запиленню приблизно 50% рослинних харчових ресурсів у всьому світі. Традиційні методи спостереження за вуликами не мають точності та безперервності, необхідної для ефективного управління колоніями, що створює нагальну потребу в досконалих системах дистанційного моніторингу, здатних безперервно відстежувати критичні параметри навколишнього середовища, включаючи температуру, вологість та якість повітря в екосистемах вуликів.

Матеріали та методи. Дане дослідження представляє розробку інтегрованої IoT системи моніторингу, що базується на платформі мікроконтролера ESP32. Система включає комплексний масив датчиків, що складається з газових сенсорів MQ-6 та MQ-135, датчика температури DS18B20, детектора твердих частинок PMS5003 та екологічного сенсора DHT21 для багатопараметричного збору даних. Обробка даних в режимі реального часу досягається завдяки багатозадачній архітектурі на основі FreeRTOS, що забезпечує одночасне опитування сенсорів, перевірку даних та бездротову передачу.

Результати. Комплексне тестування системи підтвердило надійний збір даних по всіх контрольованих параметрах. Вимірювання температури продемонстрували робочі діапазони від 13,75°C до 32,69°C для внутрішніх умов вулика, тоді як рівні вологості варіювалися між 51,70% та 88,10%. Моніторинг концентрації газу показав стабільні базові показники, при цьому датчики MQ-6 реєстрували 1,71 ppm за нормальних умов. Статистичний кореляційний аналіз виявив значущі взаємозалежності навколишнього середовища, включаючи сильну позитивну кореляцію ($r = 0{,}918269$) між вимірюваннями газу MQ-6 та MQ-135 і виражену негативну кореляцію ($r = -0{,}974651$) між параметрами температури та вологості, підтверджуючи очікувані екологічні взаємозв'язки в мікрокліматі вулика.

Висновки. Розроблена IoT-система моніторингу вуликів на базі мікроконтролера ESP32 продемонструвала стабільну та енергоефективну роботу під час безперервних польових випробувань. Система зафіксувала внутрішню температуру вулика в діапазоні від 13,75 °C до 32,69 °C, рівень вологості від 51,70 % до 88,10 %, а також концентрації газів 1,71 ppm (MQ-6) і 0,37 ppm (MQ-135) за нормальних умов.

Ключові слова: моніторинг вуликів, IoT, вбудовані системи, параметри мікроклімату, хмарні обчислення, дані реального часу.

Received / Одержано	Revision / Доопрацьовано	Accepted / Прийнято	Published / Опубліковано
25 May, 2025	06 October, 2025	10 October, 2025	31 October, 2025