Advances in Engineering Innovation Vol.16 Issue 7 EWA Publishing
Available Online: 18 July 2025 DOI: 10.54254/2977-3903/2025.25300

Design of a low-cost voice-controlled smart home hub with IoT connectivity

Tao Zheng[1], Xin Chen[1], Yunxin Chen[1], Hui Deng[1]*

[1]Geely University of China, Chengdu, China

*Corresponding Author. Email: zhengtao@guc.edu.cn

Abstract. This paper presents the design of a low-cost smart home hub based on the STM32F103C8T6 microcontroller. The system integrates local voice control using an offline ASR module, environmental sensing with a DHT11 sensor, and cloud communication via an ESP8266 Wi-Fi module using the MQTT protocol. Time synchronization is achieved through NTP, maintaining a ±2.5 ms accuracy. Performance evaluations demonstrate that the device provides reliable ambient data and stable voice interaction without internet dependency. The results confirm the system's feasibility as a compact, cost-effective solution for smart home applications.

Keywords: Internet of Things, STM32F103C8T6, Wi-Fi, NTP

1. Introduction

The development of the Internet of Things (IoT) is profoundly transforming modern lifestyles and accelerating the shift toward a fully interconnected society. Among consumer applications, smart home technologies are the most prominent representation of IoT, encompassing lighting, temperature control, security systems, and various home appliances. These systems form a unified ecosystem that significantly enhances convenience, energy efficiency, and safety for users [1]. The IoT infrastructure comprises a wide range of devices—from basic sensors to complex control units—that exchange data over networks, bridging the physical world with digital intelligence.

Against this backdrop, traditional household clocks are no longer limited to time display functions. Instead, they are evolving into multifunctional hubs that integrate environmental monitoring, user interaction, and device orchestration. The system presented in this paper embodies this new direction by proposing a low-cost smart home hub capable of providing centralized environmental sensing and control.

Designing low-cost IoT devices for the consumer market requires careful trade-offs among functionality, performance, and cost. Several key technical challenges must be addressed to achieve a practical and reliable implementation [2, 3]:

(1) Time Synchronization Accuracy

Precise and consistent timestamps are essential for data logging, event ordering, and coordinated control in distributed IoT systems. However, low-cost crystal oscillators used in microcontrollers are prone to frequency drift over time, degrading local clock accuracy. Robust synchronization mechanisms such as the Network Time Protocol (NTP) are therefore necessary to ensure temporal consistency across devices [4].

(2) Sensor Data Reliability

Low-Cost Sensors (LCS) are widely used for ambient monitoring due to their affordability and ease of integration. Nevertheless, these sensors often suffer from poor accuracy, limited sensitivity, and environmental susceptibility. To improve data reliability, calibration procedures and signal-processing techniques must be employed [5].

(3) Energy Efficiency

Power consumption is a critical constraint, especially for battery-powered IoT devices. Wireless communication (e.g., Wi-Fi) is one of the most power-hungry components. Therefore, a careful balance must be maintained between connectivity and device longevity, often requiring strategies such as duty cycling or deep sleep modes [6].

(4) Human-Machine Interface (HMI)

User experience depends heavily on intuitive and responsive interfaces. In smart home systems, a hybrid approach that combines cloud-based remote control with low-latency local operation—such as offline voice recognition—is often necessary.

Excessive reliance on cloud services introduces latency and network dependence, while purely local control limits system capability [7].

2. Methods

2.1. System architecture

The system architecture of the smart home hub aims to achieve an optimal balance between functional integration and cost efficiency. As illustrated in Figure 1, the STM32F103C8T6 microcontroller serves as the central controller, responsible for coordinating and managing all peripheral modules. The architecture intentionally decouples network-dependent tasks from core real-time operations: the ESP8266 Wi-Fi module exclusively handles network communications, including NTP time requests and MQTT message transmission, while the STM32 main controller focuses on sensor polling, user interface updates, and local command processing. This separation ensures that essential functions such as time display and local control continue to operate smoothly even in the presence of network latency or interruptions [8].

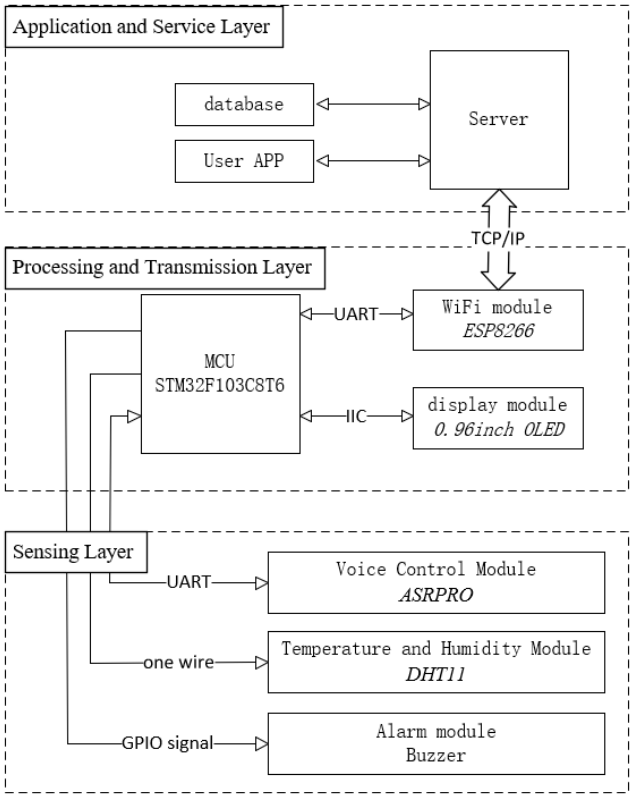

Figure 1. Block diagram of hardware structure design

The overall working logic of the system is that the STM32 master control unit acts as the overall commander and coordinates the peripheral modules. the ESP8266 module is responsible for connecting to the Internet, periodically calibrating the system time via NTP, and acting as a communication bridge between the device and the MQTT agent in the cloud. the DHT11 temperature and humidity sensors periodically collect the environmental data. the ASR offline voice module continuously listens to the user's voice commands. The OLED display serves as the main information output terminal, displaying time, date and environmental data in real time. Users can remotely monitor the status of the device and issue control commands via the cloud platform through mobile applications.

2.1.1. Microcontroller and interfaces

The STM32F103C8T6 microcontroller serves as the central control unit, offering a 72 MHz operating frequency and a rich set of GPIO pins. It seamlessly interfaces with the ESP8266 Wi-Fi module, ASRPRO offline voice recognition module, mini MP3 player, DHT11 temperature and humidity sensor, and 1.3-inch OLED display. This versatile microcontroller enables efficient data

exchange and control between all connected peripherals, ensuring coordinated system operation. Figure 2 shows a minimum system circuit design that enables the STM32F103C8T6 microcontroller to operate.

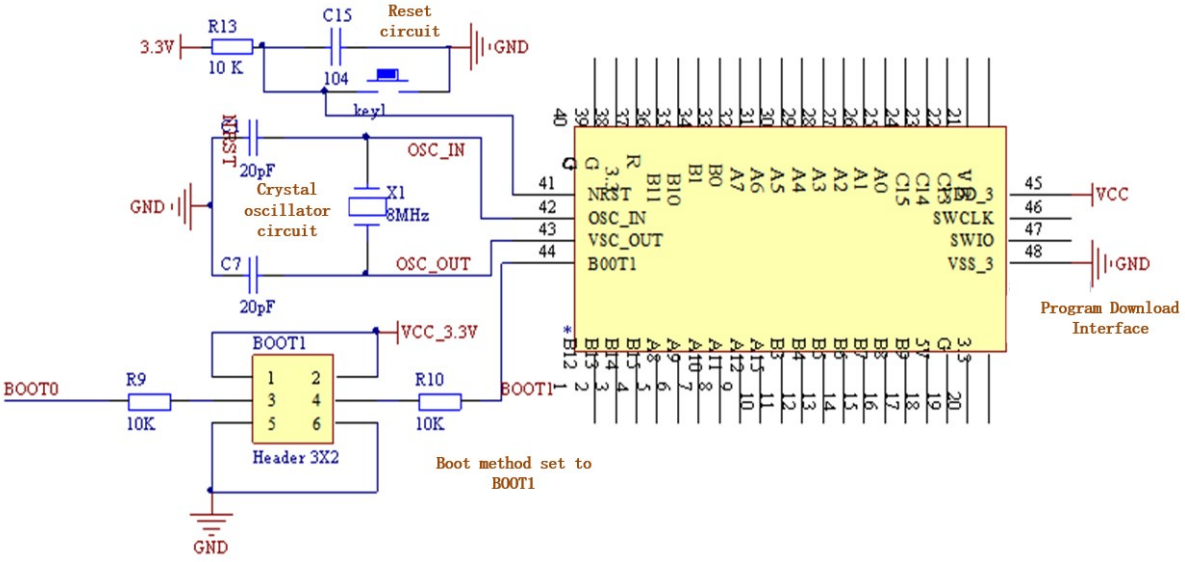

Figure 2. Microcontroller minimum system schematic

2.1.2. Communication and display design

The ESP8266 module connects to Wi-Fi networks to acquire standard time from NTP servers and communicates with the microcontroller via a UART interface. This setup ensures accurate time display and synchronization across different scenarios. The 1.3-inch OLED display utilizes an SPI interface to present real-time information such as time, date, temperature, and humidity parameters.

2.1.3. Temperature and humidity sensing design

The DHT11 temperature and humidity sensor features a simple digital interface and fast response, making it suitable for typical home environments. It provides measurement accuracies of approximately ±1.3 °C for temperature and ±10% for relative humidity, meeting the requirements for non-critical environmental monitoring applications.

2.1.4. Voice control and audio design

The ASRPRO offline speech recognition module delivers standardized control commands, which, combined with the Mini MP3 player module, enable simple voice interactions such as querying the time and playing alarm tones. This integration extends the functionality of the smart clock and enhances user interaction capabilities without relying on cloud-based voice services.

2.2. Hardware subsystem implementation

2.2.1. Control unity

The system adopts the STM32F103C8T6 microcontroller as the main control chip. Based on the high-performance ARM Cortex-M3 core, it operates at frequencies up to 72 MHz, providing strong computational power and real-time responsiveness. Its abundant peripheral interfaces, including multiple GPIOs, USART, SPI, and I2C, facilitate easy connection and management of various sensors and modules within the system. Moreover, its excellent low-power characteristics align well with the design requirements of IoT devices. Studies have demonstrated that compared to platforms like Arduino, STM32 offers superior processing capability and peripheral support, making it more suitable for developing complex multi-peripheral projects [9].

2.2.2. Network connection and time synchronization module

The design employs the ESP8266 Wi-Fi module to handle all network-related tasks. This module plays a dual role: firstly, it connects to the local wireless network, enabling internet access for the device; secondly, it includes a built-in Network Time Protocol (NTP) client function that periodically requests the standard time from public NTP servers (e.g., pool.ntp.org). This time data is used to calibrate the internal Real-Time Clock (RTC) of the STM32 microcontroller, compensating for clock drift and ensuring long-term time accuracy.

2.2.3. Environmental sensing module

The system utilizes the DHT11 digital temperature and humidity sensor for indoor environmental monitoring. This sensor connects to the STM32 microcontroller via a single-wire digital interface, simplifying the hardware design. According to the official datasheet, the DHT11 offers measurement accuracies of ±2°C for temperature and ±5% RH for relative humidity under standard conditions, which is adequate for general home environment monitoring applications.

2.2.4. Human-Machine Interface (HMI) module

The display subsystem utilizes a 1.3-inch SPI-interface OLED screen as the primary information output interface. It is responsible for presenting key information such as current time, date, temperature, and humidity to the user in real time, facilitating convenient and intuitive interaction.

2.2.5. Voice control module

The voice control function utilizes the ASRpro offline speech recognition module. Its key advantage lies in the ability to process voice commands locally without relying on internet connectivity. The module is pre-programmed to recognize specific wake words and a set of control commands (e.g., "report temperature," "set alarm"). This design guarantees low latency and high reliability for core control functions, enabling basic voice operations even during network outages.

The system integrates a Mini MP3 player module capable of reading and playing user-customized alarm tones stored on an SD card, enabling personalized reminders. Additionally, a standard active buzzer serves as an auxiliary alarm device, emitting warning sounds during specific events such as temperature or humidity threshold breaches.

2.3. Software framework and cloud integration

The system firmware is developed using the Keil uVision5 integrated development environment, leveraging the STM32 HAL library for low-level hardware drivers. The main program follows a cyclic structure: it initializes all peripherals upon startup, then continuously reads sensor data, detects input signals, refreshes the display, and handles MQTT communication tasks. The system uses the lightweight MQTT protocol for bidirectional communication with the cloud, supporting data publishing and command subscription. This architecture ensures compatibility with various cloud platforms. A companion mobile application provides users with remote monitoring and control capabilities.

2.3.1. Cloud communication

The system employs the lightweight MQTT protocol for bidirectional communication with the cloud. MQTT, based on a publish/subscribe model, is well-suited for resource-constrained IoT devices. In this model, the device acts as an MQTT client that publishes collected temperature and humidity data to specific topics, such as /device/homehub/data. Simultaneously, it subscribes to command topics, for example, /device/homehub/cmd, to receive remote instructions from the mobile application, such as setting the alarm time.

This architecture offers high platform independence and is compatible with any standard public or private MQTT broker. Consequently, the system can easily integrate with major commercial cloud platforms like Alibaba Cloud IoT Platform, AWS IoT Core, as well as open-source IoT platforms such as ThingsBoard and EMQX, providing great flexibility for commercial deployment.

2.3.2. Mobile application

Figure 3 shows the Mobile Application that accompanies the experiment, which uses WxBit graphical programming and also functions as an MQTT client. It provides a Graphical User Interface (GUI) through which the user can remotely view real-time sensor data reported by the device and configure various device parameters (e.g., alarm settings). This enables easy and efficient remote management of smart home hubs.

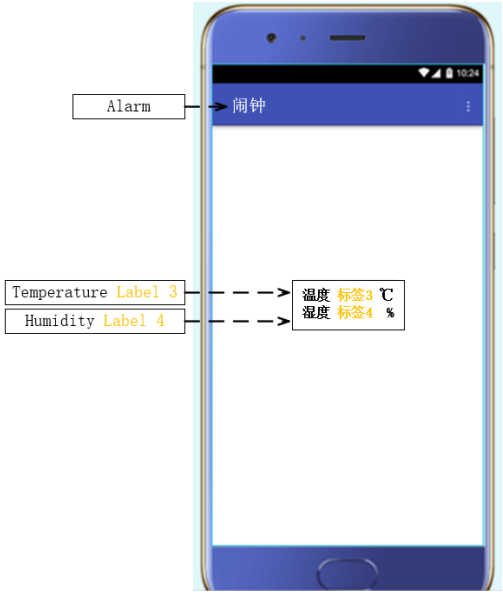

Figure 3. User app interface diagram

3. Results

3.1. System prototype and functional verification

This study successfully constructed a hardware prototype of the smart home hub, as shown in Figure 4. A series of functional tests verified the successful implementation of all designed features. These include: correct information display on the OLED screen, successful time synchronization and calibration with the NTP server via the ESP8266 module, temperature and humidity data acquisition from the DHT11 sensor, response to physical buttons and offline voice module commands, alarm and alert functionalities via the MP3 module and buzzer, as well as remote data monitoring and command issuing through the mobile application connected via the cloud platform. All functions operated stably and as expected.

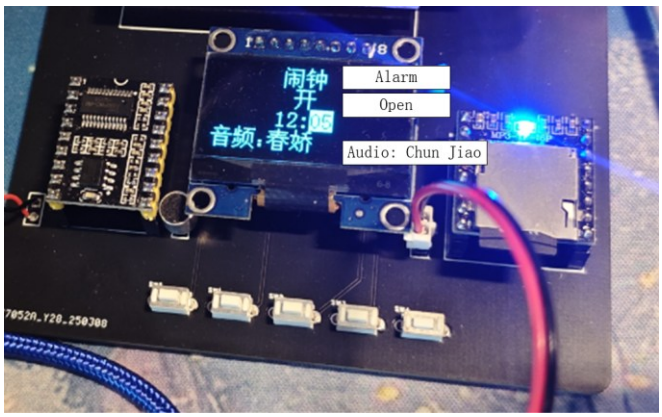

Figure 4. System realization in kind

3.2. Quantitative performance evaluation

To objectively evaluate the key performance indicators of the system, two quantitative tests were conducted.

3.2.1. Time synchronization performance

To assess the accuracy of the system's time synchronization, the clock offset after synchronization with a public time server (ntp.aliyun.com) via the NTP protocol was recorded every 6 hours over a 24-hour testing period. The results are summarized in Figure 5.

Figure 5 shows the measured delays and time offsets obtained at six-time intervals through NTP synchronization. Except for the test at 14:00, all attempts exhibited low delay (<0.03 seconds) and minimal time offset (<±2 milliseconds), indicating stable synchronization. However, at 14:00, the delay surged to 1.125 seconds with a corresponding time offset of -0.21 seconds, resulting in synchronization failure. This anomaly indicates possible network congestion or packet loss at that time, highlighting the importance of redundancy or fallback strategies in time-sensitive IoT applications.

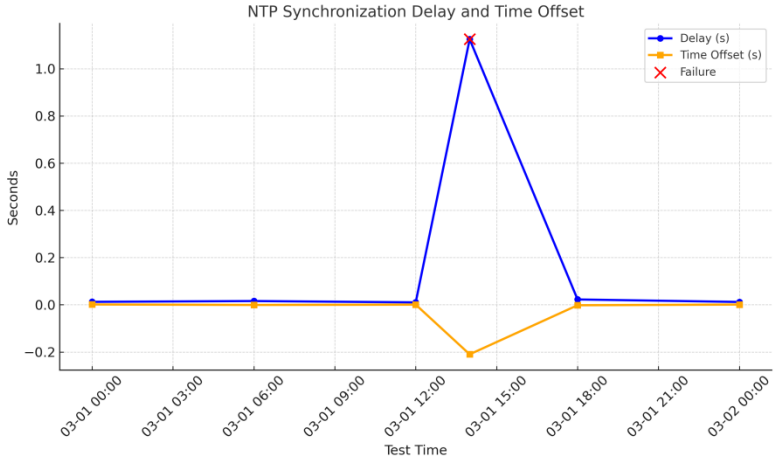

Figure 5. NTP grant delay test data

3.2.2. Environmental sensor performance

To evaluate the actual measurement accuracy of the DHT11 sensor, this study compared its readings with those of a high-precision reference temperature and humidity meter in both indoor and outdoor environments. The test lasted 12 hours, with data recorded every 3 hours. The error results are shown in Figure 6.

Figure 6 illustrates the error trends in indoor and outdoor temperature and humidity measurements. The errors gradually increased from 9:00 to 21:00, especially the outdoor humidity error, which rose from 8% RH to 12% RH. This persistent upward trend suggests that the sensor elements may be affected by thermal drift or environmental influences, particularly during the afternoon when external conditions fluctuate significantly. These results emphasize the necessity of dynamic calibration strategies to maintain sensor accuracy under varying conditions.

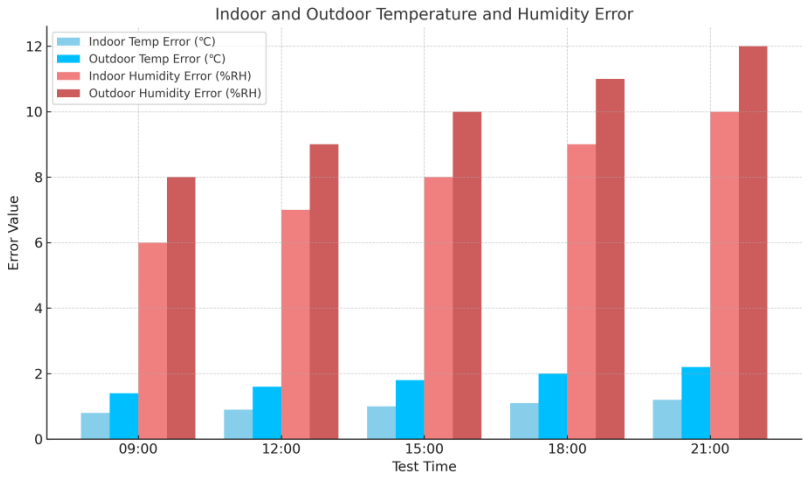

Figure 6. Indoor and outdoor temperature and humidity error data

4. Conclusion

4.1. System performance interpretation

The system achieved millisecond-level time synchronization accuracy, which is an excellent performance for a low-cost IoT device, demonstrating that a standard NTP client is sufficient for such applications. However, it must be acknowledged that this high performance heavily depends on a stable and low-latency local network environment. Under network congestion or instability, NTP performance may degrade, with increased delays and jitter causing larger synchronization errors. This is an inherent limitation of the NTP protocol.

The temperature measurement accuracy met expectations and can provide reliable indoor temperature references to users. However, the relatively large error observed in the humidity sensor is a significant finding that deserves in-depth analysis. Experimental results show that the humidity error reached ±12% RH, far exceeding the ±5% RH specified in the datasheet. The discrepancy between theory and practice may stem from several factors:

Low-cost sensors typically undergo only basic factory calibration. To achieve optimal performance in practical applications, users often need to perform multi-point calibration against reference instruments, a step not included in this design.

Sensors are highly sensitive to local environmental conditions. In this design, the DHT11 sensor was integrated on the main PCB, close to heat-generating components such as the STM32 microcontroller. The local heat generated during operation may slightly increase the air temperature around the sensor, causing the measured relative humidity to appear lower.

This result also highlights the general challenge of relying on low-cost sensors for accurate measurements without strict calibration. It clearly distinguishes between “indicative monitoring” and “metrological-grade monitoring.” For this application, providing a trend indication of environmental changes is sufficient. However, for more demanding scenarios, higher-grade sensors or rigorous calibration processes are necessary.

4.2. Design choice analysis

This system strategically chose an offline ASR module for voice control. This decision involved a trade-off: sacrificing the large vocabulary and natural language understanding capabilities offered by cloud-based ASR services in exchange for high reliability, low latency, and complete independence from internet connectivity. This design choice makes the device’s core control functions (such as time queries and mode switching) more robust. Additionally, processing voice data locally better protects user privacy.

However, it must be admitted that power consumption optimization was not a primary focus during prototype design. The frequent network activities of the ESP8266 Wi-Fi module (for NTP synchronization and MQTT heartbeat) are the main sources of energy consumption. While this is not an issue in continuously powered scenarios, it poses a critical drawback for future battery-powered versions. This limitation reflects the inherent contradiction between rich functionality and energy efficiency in IoT device development. Future iterations must incorporate power management strategies, such as implementing deep sleep modes for the processor and duty-cycling peripheral wake-ups in firmware. Regarding time synchronization, more advanced energy-saving algorithms could be adopted, such as ecoSync or reinforcement learning-based adaptive synchronization interval adjustment, to minimize Wi-Fi module wake-ups and operation time while maintaining acceptable clock accuracy.

4.3. Limitations and future work

Based on the above discussion, this study has the following main limitations and indicates directions for future improvements:

1.Sensor Accuracy

The accuracy of the DHT11 humidity sensor is the primary performance bottleneck of the current system. Future work could consider replacing it with higher-precision sensors (e.g., DHT22 or SHT series) or introducing calibration algorithms based on reference data at the software level to compensate for system errors.

2.Energy Efficiency

The system lacks effective power consumption optimization mechanisms. Future research should focus on implementing deep sleep modes, optimizing communication protocol energy use, and adopting smarter energy-saving synchronization strategies.

3.System Security

The current prototype does not adequately address IoT security concerns. Future development should enforce Transport Layer Security protocols (TLS) for MQTT communication and implement device authentication and firmware protection measures to defend against common network attacks.

5. Conclusion

This study successfully designed and implemented a fully functional, low-cost smart home hub prototype. The system effectively integrates multiple core IoT technologies—including high-precision time synchronization, environmental sensing, offline voice control, and cloud-based remote management—into a unified platform. Through empirical performance evaluation, we quantified

the system's key metrics: achieving millisecond-level time synchronization accuracy and providing indoor environmental data suitable for indicative monitoring.

The value of this project lies not only in building a practical device but also in serving as a concrete case study that reveals fundamental design trade-offs faced when developing IoT products for the consumer market. The research findings clearly demonstrate the interdependent constraints among cost, accuracy, functionality, and power consumption. The practical experience gained provides important guidance for the design and development of future low-cost, high-performance IoT systems.

References

[1] Al-Ali, A. R., Zualkernan, I. A., Rashid, M., Gupta, R., & Alikarar, M. (2017). A smart home energy management system using IoT and big data analytics. *IEEE Transactions on Consumer Electronics, 63*(4), 426–434. https://doi.org/10.1109/TCE.2017.015014

[2] Sethi, P., & Sarangi, S. R. (2017). Internet of things: architectures, protocols, and applications. *Journal of Electrical and Computer Engineering, 2017*, 9324035. https://doi.org/10.1155/2017/9324035

[3] Rayes, A., & Salam, S. (2019). *Internet of Things from Hype to Reality: The Road to Digitization (2nd ed.)*. Springer. https://doi.org/10.1007/978-3-319-99516-8

[4] Mills, D. L. (1991). Internet time synchronization: The Network Time Protocol. *IEEE Transactions on Communications, 39*(10), 1482–1493. https://doi.org/10.1109/26.103043

[5] Srivastava, D., Kesarwani, A., & Dubey, S. (2018). Measurement of Temperature and Humidity by using Arduino Tool and DHT11. *International Research Journal of Engineering and Technology (IRJET), 5*(12), 876-878.

[6] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials, 17*(4), 2347-2376.

[7] Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... & Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine, 29*(6), 82-97.

[8] Zanella, A., Bui, N., Castellani, A., Vangelista, L., & Zorzi, M. (2014). Internet of things for smart cities. *IEEE Internet of Things Journal, 1*(1), 22-32.

[9] STMicroelectronics. (2018). *STM32F10xxx Cortex-M3 programming manual (PM0056)*. STMicroelectronics. https://www.st.com/resource/en/programming_manual/pm0056-stm32f10xxx-cortexm3-programming-manual-stmicroelectronics.pdf