IJEEC

International Journal of Electrical Engineering and Computing
Vol. 7, No. 2 (2023)

Original research paper
UDC 004.021:504.3.054
DOI 10.7251/IJEEC2302073B

HTTP embedded web server and its application for air parameters monitoring

Tijana Begović, Zorana Mandić, Nikola Kukrić, Slobodan Lubura

Faculty of Electrical Engineering, University of East Sarajevo, East Sarajevo, Bosnia and Herzegovina

E-mail address: tijana.begovic@etf.ues.rs.ba, zorana.mandic@etf.ues.rs.ba, nikola.kukric@etf.ues.rs.ba, slobodan.lubura@etf.ues.rs.ba

Abstract— **Embedded systems find extensive applications in various aspects of daily life and their presence in the highly connected world of today is growing. With a large number of remote devices and sensors, there is a growing need for remote data processing. Microcontrollers, often integral components of such devices, can enable remote data manipulation through the implementation of a web server called embedded web servers. The implementation of a server on such a system poses challenges, such as limited resources, so it is crucial for the web server implementation to be compatible with the available resources. This paper focuses on the requirements for the implementation of a web server. Also, an embedded web server has been designed for specific application of monitoring air parameters. The implementation utilizes the LwIP stack and allows for remote access to measurement results within a local network. The paper will delve into the operational principles of both the web server and the entire system.**

Keywords-embedded systems, HTTP web servers, STM32, ambient temparature application

I. INTRODUCTION

The origin of the embedded systems is linked to the year 1960s when the Apollo Guidance Computer was developed for the navigation and guidance of the Lunar Excursion Module to the moon at the Massachusetts Institute of Technology [1]. Its success led to the development of the first microcontroller by Texas Instruments, TMS1802NC, which served as four-function calculator [2]. From its beginnings to until now, embedded systems have been developed with emerging technologies. Internet of Things, one of the greatest pillars of moder technology, relies on embedded systems as one of the key drivers for development [3]. Some of the most important objectives of any system are connectivity and data exchanges throughout the Internet, which has led to state-of-the art communication protocols and network interfaces embedded into the microcontrollers. A web server which runs on embedded devices is called embedded web server and it enables remote access to this system from other devices via web browser. Servers like these differ from common web servers because they can run on limited resources without operating systems if it is necessary [4]. Web pages, generated in web servers, are mostly used for interface with embedded system. Some advantages of implementing web servers into the embedded devices are improving user interface with embedded device; interaction with other devices which contain web browser [5], the remote monitor and control mode provides complete exact real-time information and decision making [6]. Applications of EWS are not limited only to microcontrollers and can be implemented on the FPGA [7] and programmable logic controllers [8], with wide scope of applications: from agriculture [9,10], industry [11], home automation [12], etc. In achieving interface functionality, TCP/IP and HTTP protocols are commonly employed. Given the extensive and intricate nature of these protocols and the limited resources of embedded systems, it is advisable to include only essential components of the protocol stack [13].

This article will provide the introduction to the web server implementation in a special purpose embedded system, with implementation and applications of the HTTP web server in air parameters monitoring system, which enables remote review of measured parameters. This article is organized as follows: Section II gives an overview of the embedded web servers. The application of embedded web servers for remote monitoring of air parameters with its hardware components and protocols are discussed in Section III. Web server implementation principle is presented in Section IV, and Section V gives an overview of algorithm, working principles and web implementation of proposed system. Finally, Section VI provides comparison between proposed system and available solutions in literature.

II. EMBEDDED WEB SERVERS

Block diagram of the system with embedded web server is presented in Fig. 1. An embedded web server can be defined as a web server intended to be run on small devices, with request to be able to run on the limited resources. Industrial control systems and monitoring systems in wide scope of

This paper is a revised and expanded version of the paper presented at the XXII International Symposium INFOTEH-JAHORINA 2023 fields can incorporate web servers to facilitate web-based network communication. Utilizing embedded web servers, remote monitoring offers real-time information to managers, enabling them to make immediate decisions and act upon them [6]. In environments with EWS-equipped devices, users can employ web browsers to authorized devices, as well as send and receive data from browsers globally and locally. This allows for configuration, monitoring, and control from any location. Through web browsers, users can remotely manage individual devices, providing a straightforward yet powerful user interface [11].

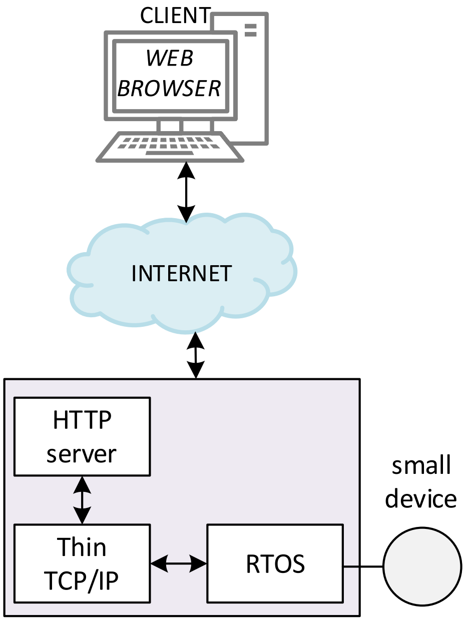


Figure 1. The architecture of the embedded web server [6]

The installed server on the device has an allocated IP address, which provides access via browser to the authorized clients, meaning that no special-use software is needed. Communication, between embedded web server and browser, is being carried out via thin HTTP. A client submits the request to the embedded server, and an operation is being executed by the embedded system application. Dynamically created web page by the server is returned to the client with the response of the operation requested.

The development of an embedded web server must address limited system resources, specifically meeting memory and processing power requirements while minimizing CPU impact during web request servicing. Additionally, high reliability, portability, and security, especially in applications involving equipment configuration, are crucial considerations for embedded web server functioning in diverse small devices environments with limited resource constraints [4].

III. AIR PARAMETERS REMOTE MONITORING SYSTEM

Block diagram of air parameters monitoring system, implementing embedded web server, is given in Fig. 2. The system presented includes hardware components: development board, sensors for monitoring and personal computer as programming station, as well as web server as software part of the system.

A. *STM32 Nucleo development board and STM32CubeIDE development tool*

STM32 microcontroller can be identified as the central part of the system. To enable simple, fast, and affordable work with STM32 microcontroller, a development board by *STMicroelectronic*, STM32Nucleo–F756ZG, is being used. This board is characterized as high-performance board (14), and it is equipped with various peripherals and a built-in programming tool (ST-LINK/V2-1), which significantly facilitates use and programming and provides fast prototyping.

The board can be powered via USB connector or by external power source. It contains several LEDs, some of them are user LEDs and others provide information about board components status. The board is equipped with peripherals for USART, USB and Ethernet communication [15]. The microcontroller is STM32F756ZG with 32-bit ARM Cortex - M7 core. It can operate at up to 216 MHz frequency and contains a memory protection unit (MPU). The microcontroller has an embedded flash memory of 1 MB. It is equipped with eighteen 16-bit or 32-bit timers, one hundred and sixty-eight I/O pins with interrupt capability, twenty-five communication interfaces (including I^2C, USART, UART and SPI), enhanced USB and Ethernet connectivity and analog - digital and digital - analog converters. The microcontroller has built-in additional functionalities such as random number generator, real time clock and CRC calculation unit [16].

A development tool intended for developing software for STM32 microcontrollers is STM32CubeIDE. It is a C/C++ platform, based on the Eclipse framework, that enables peripheral configuration, code generation and compilation, as well as debugging [17].

HAL *(Hardware abstraction layer*) driver, that provides high-level programming interfaces accessible to all peripherals, is available within this IDE. This driver covers generic functions (intended for implementing basic peripheral functionalities) and extended ones (intended for implementing special peripheral functionalities). The STM32CubeIDE also provides LL (*Low layer*) drivers and libraries, that execute instructions at the register level and are only available for specific microcontroller peripherals. The LwIP stack, representing an implementation of the TCP/IP protocol primarily designed for embedded systems, is also at disposal in this development environment group [18].

B. *Sensors for monitoring air parameters*

1) BME280

BME280 is a widely used integrated digital temperature, humidity and pressure sensor, that is suitable for use in battery driven systems because of small dimensions and low power consumption. The temperature sensor is used to measure the ambient temperature in the range of -40 °C to 85 °C, and for the humidity and pressure compensation. A humidity sensor measures full humidity range and has a very short response time and high accuracy over a wide temperature range. The pressure sensor measures absolute barometric pressure with high accuracy and resolution. It can be powered with a voltage

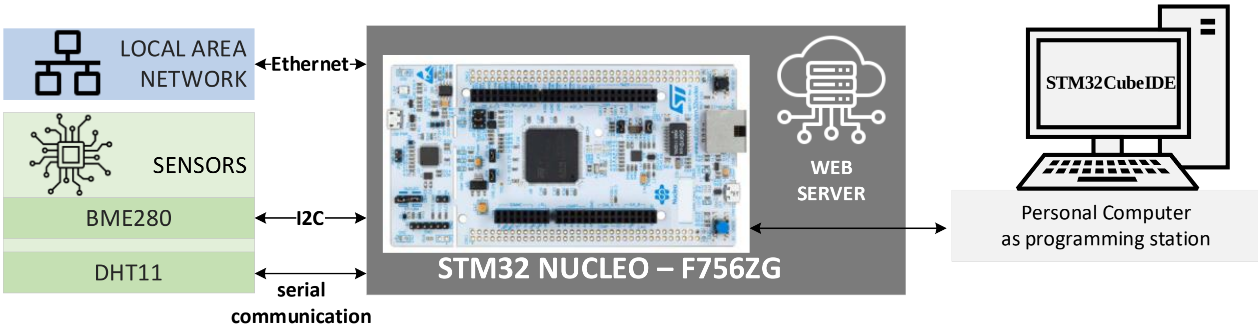


Figure 2. Embedded web server application schematic

between 1.7 V and 3.6 V. For communication the I^{2}C and SPI interface can be utilized.

Depending on the application, the sensor can operate in three different modes – sleep mode, forced mode and normal mode. It can be used indoors and outdoors, for permanent monitoring or single measurements. It is suitable for application in indoor navigation, fitness, home automation control and weather monitoring [19].

2) DHT11

The DHT11 is a digital temperature and humidity sensor with calibrated output signal, which is widely applicable indoors and outdoors, for equipment testing, in the automotive industry, medicine, weather monitoring stations and household appliances [20]. It is characterized by small dimensions and low power consumption, which makes it usable even for the most demanding applications. It can be powered with a voltage between 3.3 V and 5.5 V [20], while communication with the master device is achieved using only one wire [21]. The sensor includes an NTC temperature sensor, which measures temperature in the range of 0 °C to 50 °C, as well as a resistive humidity sensor for humidity in the range of 20% to 90%. This configuration together with the 8-bit microcontroller ensures excellent measurement quality, low response time and cost-effectiveness [21].

C. Protocols and technologies

1) I^{2}C communication protocol

I^{2}C (*Inter-Integrated Circuit*) is a serial communication protocol developed by Philips Semiconductors, used for synchronous communication between one master and one or more slave devices [20]. The number of slave devices that can be connected at the same time is only limited by bus capacitance. This protocol requires two bus lines: serial data line (SDA) and serial clock line (SCL), provided by the master. All slave devices are connected to a single bus and have unique 7-bit or 10-bit addresses. Bidirectional serial 8-bit data transfer is done in several modes, regarding its speed: Standard mode (100 kbit/s), Fast-mode (400 kbit/s), Fast-mode Plus (1 Mbit/s) and High-speed mode (3.4 Mbit/s). Unidirectional 8-bit data transfer can be done in Ultra-speed mode up to 5 Mbit/s.

Data is sent in 8-bit format; where the MSB bit is transmitted first, and the number of bytes being sent is arbitrary. The master generates and sends a start bit, which initiates the transfer. This is followed by the address of the target slave device and a bit that determines the direction of data exchange ($R/\overline{W}$ – 0 for reading, 1 for writing data). A device after receiving each byte of data, regardless of whether it is a master or a slave device, sends an ACK (*acknowledge*) bit to inform the transmitter whether the data has been received correctly. When the data is not received correctly, a NACK (*not acknowledge*) bit is being sent. The data transfer is terminated by the master by sending a stop bit.

The I^{2}C protocol is widely used in embedded systems where the microcontroller represents the master device. Various types of sensors, displays, I/O expanders, A/D and D/A converters or EEPROmories are used as slave devices [10].

2) HTTP protocol

HTTP (Hypertext Transfer Protocol) is an application – level network protocol, based on the exchange of requests and responses within servers and clients, that has been used by the World Wide Web since 1990. It is a generic protocol used for distributed and hypermedia information systems. It is also used for communication between users and proxy servers and some other Internet systems that use protocols such as FTP, SMPT or NNTP [22]. The HTTP protocol defines the form of the request that the client sends to the server, as well as the form of the response that the server sends to the client. It is not a completely secure protocol and it is vulnerable [23]. Communication is generally initiated by the user, an HTTP client, who sends a request to the server. Request consists of an URI, protocol version and message. After receiving the request, the server sends a response to the user. Response contains a status line, protocol version, information about the success of the message and a message containing information from the server.

HTTP can be implemented on the top of any other protocol on the Internet or any other network. In general, the connection is made using the TCP/IP protocol [22]. When the user sends a request in the form of a URL through the browser, the request is translated into the IP address of the server. After that, a TCP connection to the server is established and communication between the client and the server begins [24].

3) Ethernet

Ethernet represents a set of rules; a standard that defines the method of computer systems connection to a LAN network. Ethernet was declared standard IEEE 802.3, which defines the types of physical media for signal transmission, the Ethernet message frame, as well as a set of rules for controlling medium access by multiple devices connected to the network [25]. Ethernet is the base of the LAN implementation and remains the basic media technology as it is flexible, simple and cheap. Through the development and improvement of Ethernet technologies, many faults have been overcome and eliminated [26]. To improve transmission and eliminate shortcomings, several generations of Ethernet have been developed. These generations differ in transmission speed and type of used physical medium; standard Ethernet, fast Ethernet, gigabit Ethernet, 10-gigabit Ethernet, and 40-gigabit and 100-gigabit Ethernet [25].

Fig. 3 shows frames, which are data structure transmitted using the Ethernet. CSMA/CD (*Carrier-sense multiple access with collision detection*) method is used for medium access control, which relies on checking the channel occupation. If the channel is free, the device can start sending data frames.

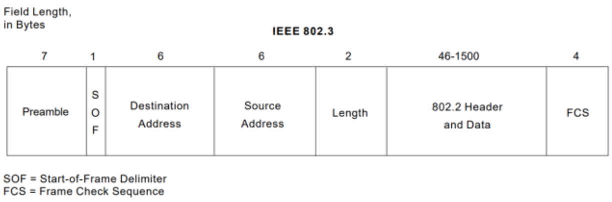


Figure 3. Ethernet package frame [25]

IV. WEB SERVER IMPLEMENTATION PRINCIPLE

The realization of the HTTP web server in embedded system with STM32 microcontroller is enabled by Ethernet HAL driver and LwIP middleware, both available within STM32CubeIDE [27]. The following will present a brief description of mentioned web server enablers.

A. LwIP stack

LwIP is an open – source stack which represents implementation of TCP/IP protocol. It was initially developed by Adam Dunkels and now is maintained by a worldwide network of developers. The main characteristics is RAM usage reduction while maintaining a full-scale TCP/IP stack, which makes it suitable for use in embedded systems. Main features include IP, IPv6, ICMP, ND, MLD, UDP, TCP, IGMP, ARP, PPPos, PPPoE protocols [28], and three application programming interfaces (APIs), that work at a low level and allow application to control data directly: Raw API (enables the development of applications using events callbacks), Netconn API (a high-level sequential API that requires a real-time operating system) and BSD Socket API (Berkeley-like Socket developed on top of the Nerconn API) [27]. The design of LwIP stack is based on layered protocol design so each protocol is implemented as a separate module. Also, for some actions, this is not strict, and layers could be mixed. In addition to the modules implemented for each protocol, there are also some additional auxiliary modules. The process model of LwIP stack is based on a single process. Thus, applications using this stack could be used without operating system. The advantage of this approach is portability over different operating systems, even the reduced once. All data transfer via LwIP is done through network connection [29].

In the implementation of applications with STM32 microcontrollers, the LwIP stack is used along with the Ethernet peripheral, since the TCP/IP protocol is mainly used in the work with this interface. LwIP stack is utilized with STM32Cube HAL API, including Ethernet API. TCP/IP stack application could be developed with or without an operating system. As the LwIP architecture relies on TCP/IP model, data transfer on four different levels is possible – the link layer, the internet layer, the transport layer and the application layer.

LwIP stack features are mostly used with Ethernet HAL driver. For interface between these two is used *ethernetif.c* file which contains functions needed for implementations of any application. Function implementation differs depending on whether the application is with or without an operating system. Application in standalone mode, without an operating system, is developed using raw API. In this mode, software needs to poll for recieved Ethernet package. After recieving message, it is copied from Ethernet buffer to LwIP buffer so it could be processed. If application uses callback functions, LwIP notify application layer. In the opposite case, only lower layers are required.

B. HTTP Web server implementation

Implementation of HTTP server belongs to LwIP features application group. These applications support HTTP and DHCP protocols which enable functionalities of web servers. Web servers could be based on raw APIs or may require real – time operating system.

Web server implemented with raw API enables access to STM32 microcontroller from clients within same network via IP assigned to microcontroller. Features which web server enables are URL parsing, CGI – Common Gateway Interface, SSI – Server Side Includes, Dynamic Header generation and HTTP post request.

Whether the user wants to assign dynamic or static IP addresses, it is necessary to enable or disable DHCP. Once the application is built and started, web clients via assigned IP address load html pages available within web server. These web pages could be both static and dynamic.

Dynamic web pages are loaded through SSI and CGI. SSI is method which enables including dynamic data in html code. Code contains specific tags which should be dynamically replaced during execution time. When a certain web page is requested, the application replaces the tag with value of variable that was pre-assigned to the tag. Web page is generated form file with 'shtml' extension. CGI is a technique which enables client request execution on the server side.

After the request is processed HTTP server sends corresponding file to client. This concept allows remote control of microcontroller itself, MCUs peripherals or any device connected to it [27].

V. AIR MONITORING SYSTEM: ALGORITHM, WORKING PRINCIPLE AND WEB SERVER IMPLEMENTATION

Air parameters measurement system operation can be divided into two phases. At the beginning the initial phase is executed once. The second phase, called focal phase, is executed cyclically, in several steps which include processes of measurement, averaging and representation of results and connection check. The algorithm, made of mentioned two phases, is shown in Fig. 4.

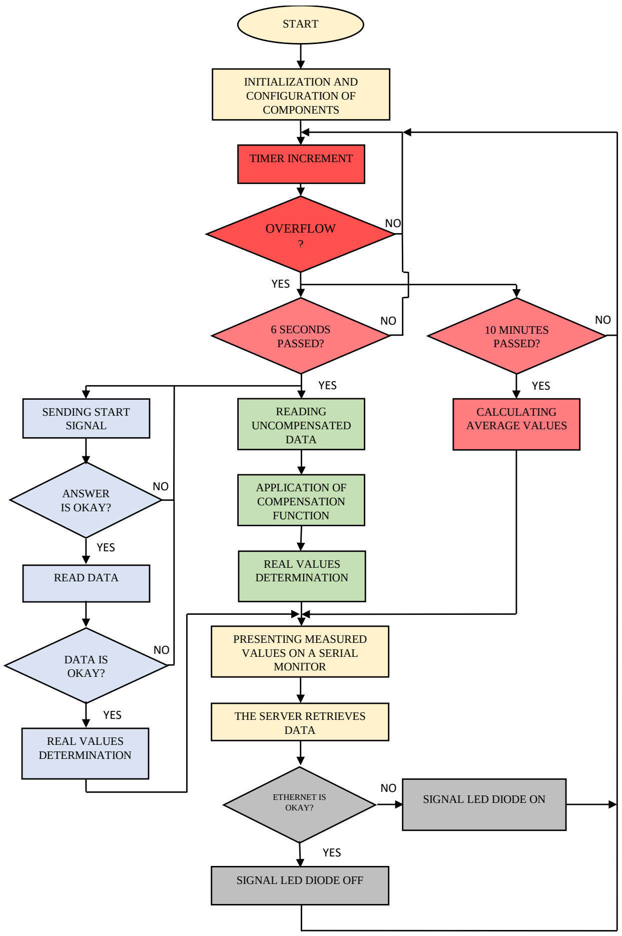


Figure 4. Algoritm of proposed air parameters monitoring system

A. Initial phase

After the system has been installed, start the execution of the initial phase, which includes microcontroller peripheral (RCC, GPIO, I^2C, UART, LwIP, timers) and sensors (modes, sample time, oversampling) configuration. Initialization of HTTP web server is done during this phase. Then, dynamic web pages are generated from files with 'shtml' extension which contains tags. The corresponding variables are assigned to these tags.

Initialization is preceded by the physical connection of the microcontroller to the LAN network with Ethernet, and the physical connection to sensors and a personal computer. The PC acts as programming station and it is not necessary for the system operation but can optionally be used for power supply and presentation of measured values on the serial monitor.

B. Focal phase

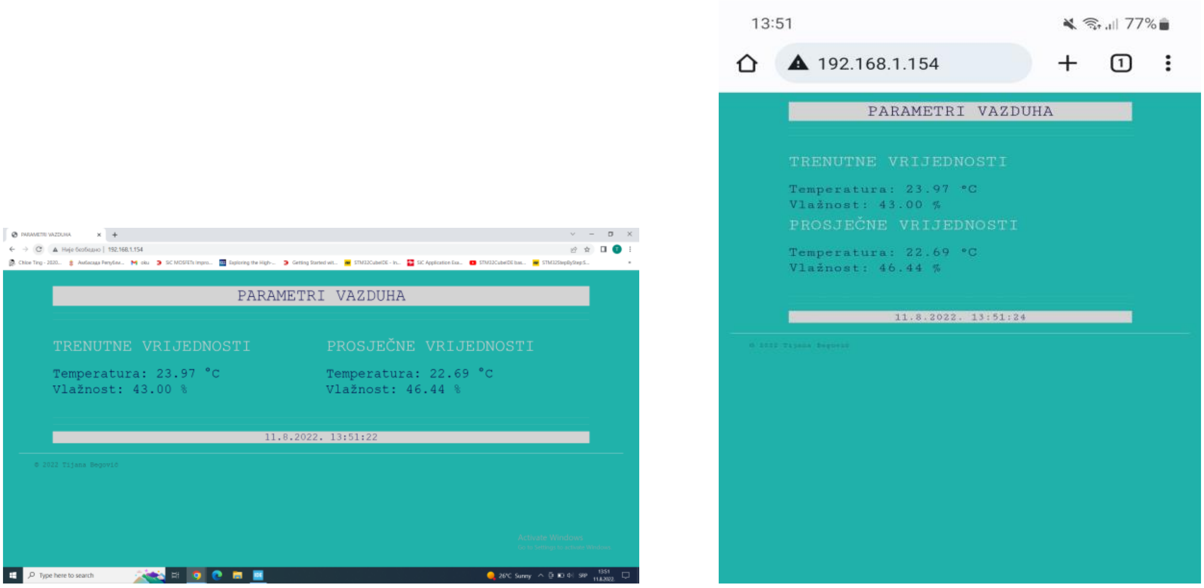
The second phase, or focal phase, comprises several cyclically repeated steps. The first step involves reading measuring data from the sensors, which occurs every six seconds. Firstly, BME280 data is read and compensated using the I^2C protocol for communication. Subsequently, data is read from the DHT11, and its accuracy is verified. The next step involves calculating average values of air parameters, performed every ten minutes. These average values represent the mean values based on a sample of 100 measurements. Following each measurement (data read), the results can be displayed on the serial monitor, available within STM32CubeIDE, providing a display of the measured data at the air parameter measurement location. To utilize this functionality, a PC is required. The fourth step consists of accessing the microcontroller and measured data via static IP address from other devices within LAN. This step will be discussed in detail later. The final step encompasses Ethernet checking. During this stage, the status of the Ethernet cable is examined. If the cable is unplugged, the signal red LED is turned on remains on until the cable is reconnected. Regardless of the connection status, the measurement process continues without interruption.

C. Web server implementation in proposed system

The previously described system for monitoring air parameters incorporates an HTTP web server, designed to facilitate access to measurement results from remote devices within the same LAN. The system for data measurement and processing is connected to LAN with Ethernet technology which enables web server implementation with LwIP stack.

The HTTP server is implemented as an SSI (Server Side Includes). To achieve this, a web page is generated from a file with a 'shtml' extension, containing tags that are dynamically replaced during program execution. The initialization of web server is done as the last step in the initial phase of execution, where a dynamic web page is generated which makes data stored on the server available. A function is called to assign corresponding variable to each tag in file. For this application, variables are current and average values of temperature and humidity. The microcontroller is assigned a static IP address earlier, which is later utilized by web clients.

The exchange of requests and responses between clients and server represents reception and transmission of the Ethernet packages as both server and clients are connected into the same LAN. The main program continuously polls for incoming Ethernet packages. After one or more packages are received, the message within the package is analyzed. If it contains a client request to access the web page, the tags are



(a) (b)

Figure 5. Web page with current and average values of the temperature and humifity (air parameters) when a) desktop version and b) mobile version is used

replaced with measured values of corresponding variables, so air parameter values become available to clients through the web page. The desktop and mobile versions of the web page displaying measured air parameter values are illustrated in Fig. 5a and Fig. 5b, respectively.

This implementation of web server in system for air parameters monitoring enables its application within a single LAN network, for monitoring air parameters within a several rooms or the entire building. It can also be used outdoors for weather or similar monitoring. Regardless of measuring system position within the network, it is possible to access the measurement results, available on the web page, from any device connected to the network. It is accessed via a static IP address assigned to the microcontroller during setup and initialization. Also, it is possible to set up several measuring systems for this purpose within one network.

Since the monitoring of air parameters is necessary in various spheres of life and work, this type of system can be widely used. It can be installed indoors, for measuring air parameters in factories, various facilities, healthcare institutions and hospitals, residential and commercial buildings. With the same efficiency, it can also work outdoors, within agricultural farms and fields and smaller public and private areas. In both cases, it is necessary to provide the physical connection, with Ethernet technology, of the measuring system into the LAN network.

The implementation of the HTTP web server stands out as the biggest advantage of the described system, due to the possibility of remote monitoring of measured values within a single network.

VI. COMPARING BETWEEN PROPOSED AND AVAILABLE SYSTEMS

In recent years, several systems for monitoring air parameters have been developed and introduced with evident differences in design and implementation compared to the system presented in the paper. In [30] the ATmega328P microcontroller was used that is characterized with significantly less flash memory compared to the STM32. This led to the system operating at a lower frequency and being somewhat slower. A significant difference is observed in the presentation of results, as the values in the mentioned system are displayed on a local LCD display, without utilizing embedded web server, making them readable only at the measurement location. A similar approach was presented in [31], with the addition of a fan controlled by a single-channel relay module which brings the possibility of automatic temperature regulation. Values can only be read at the measurement location, and the system operates somewhat slower due to the microcontroller type. A system designed for monitoring air temperature and humidity in various types of enclosed spaces, such as laboratories, hospitals, warehouses, and industrial facilities was presented in [32], which is based on the PIC16F877A microcontroller and DHT11 sensor. Although this system is simple and most cost-effective, it has its limitations, including a limited temperature measurement range of 0 to 60 °C, and the ability to locally read the measured values. Also, the PIC16F877A is an 8-bit microcontroller, which may limit its operating speed compared to 32-bit microcontrollers like the STM32-F756ZG. Monitoring the temperature and humidity of small fruit and vegetable storage refrigerators is presented in [33], with an Arduino development board, ZigBee wireless communication module, DHT11 sensor, and OLED display. The measured values are displayed locally but can also be monitored remotely via a mobile application that receives data via the Wi-Fi module. There is also an option to transmit data to any IoT platform via the ZigBee module. This system has the advantage of remote monitoring and data transmission to different platforms, making it adaptable for various applications.

Comparing all the analyzed systems, the system based on the STM32-F756ZG microcontroller and two types of sensors from this work proved to be more functional and versatile, providing the ability for accurate measurement, remote access, and both local and remote monitoring of measured values. The microcontroller provides sufficient operating speed and the capability to communicate with other devices through various interfaces and technologies. Depending on the application area and the range of values to be measured, either individual or both sensors can be used. The implementation of an HTTP web server stands out as the greatest advantage due to the possibility of remotely monitoring the measured values within one network.

VII. Conclusion

Today, limited systems, deployed to oversee specific parameter values, with constrained resources find extensive use across various aspects of people's lives. Often, the need arises to provide the monitoring values in locations distinct from the measurement site, making the provision of such accessibility essential. This paper presents the implementation of an HTTP web server utilized in a specialized system for monitoring air parameters, applicable in both indoor and outdoor settings. The presented application offers the capability to monitor data from remote devices within the same local network. The web server implementation presented in this paper is straightforward and does not demand intricate hardware or software resources. Consequently, it proves great advantages compared to similar solutions previously mentioned, where the web server implementation on the microcontroller stands out as the primary advantage of the air parameter monitoring system, facilitating remote access to measurement results, and broadening the system's applicability.

Despite focusing on the web server's implementation in an air parameter monitoring system, this approach remains independent and can be employed, with unsignificant changes, in systems of diverse purposes. Future directions of work will involve expanding the system to enable access to air quality parameters beyond the LAN network. In other words, the data will be accessible to any user on the Internet. In this way, a system will be implemented that aligns with the Internet of Things paradigm.

References

[1] https://bluefruit.co.uk/processes/a-brief-history-of-embedded-operating-systems/

[2] https://www.eejournal.com/article/a-history-of-early-microcontrollers-part-2-the-texas-instruments-tms1000/

[3] F. Samie, L. Bauer and J. Henkel, „IoT technologies for embedded computing: A survey," 2016 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), Pittsburgh, PA, USA, pp. 1 – 10, 2016.

[4] Hong-Taek Ju, Mi-Joung Choi, J. W. Hong, „An Efficient Embedded Web Server for Web-based Network Element Management," International Journal of Network Management 10, pp. 261 – 275, 2000.

[5] M. Can Filibeli, O. Ozkasap, M. Reha Civanlar, „Embedded web server-based home appliance networks," Journal of Network and Computer Applications 30, pp. 499 – 514, 2007.

[6] X. Peng, J. W. Xia and J. M. Liao, „An Embedded Internet Interface System," Mobile Robots, vol. 8, no. 2, pp. 520 – 531. 2001.

[7] E. Magdaleno, M. Rodríguez, F. Pérez, D. Hernández, E. García, „A FPGA Embedded Web Server for Remote Monitoring and Control of Smart Sensors Networks," Sensors vol. 14, pp. 416 – 430, 2014. https://doi.org/10.3390/s140100416

[8] N. Kukrić, Z. Mandić, T. Begović, S. Lubura, „*WEB-BASED HMI FOR PLC SYSTEMS*, " X conference *IcETRAN*, East Sarajevo, Bosnia and Herzegovina, 2023.

[9] D. He, J. Wang, J. Bao, S. Lu, „A Web-Server-Embedded Information Monitoring System for Fruit Farm," Sensor Letters, vol. 11, pp. 1141 – 1144, 2013.

[10] S.M. Gawali, S.M. Gajabhiye, „A Design of Embedded based Web Server or Monitoring and Controlling Agricultural Field," International Journal of Current Engineering and Technology, vol. 4, no.3, pp. 1933–1935, 2014.

[11] J.Liu, Y. Zhang, L.Xu, J.Wang, „Design and Implementation of Embedded Web Server in Industrial Control Systems, " Communications and Network, vol. 05, pp. 65 – 68.

[12] D. C. Karia, V. Adajania, M. Agrawal and S. Dandekar, „Embedded web server application based automation and monitoring system," International Conference on Signal Processing, Communication, Computing and Networking Technologies, India, pp. 634 – 637, 2011, doi: 10.1109/ICSCCN.2011.6024628.

[13] C. Fu, Z. Zhu, X. Gao, P. Wang, „The design and implementation of a general reduced TCP/IP protocol stack for embedded Web server," International Conference on Industrial Electronics and Control Applications, pp. 1 – 5, 2005.

[14] https://www.st.com/en/evaluation-tools/stm32-nucleo-boards.html

[15] ST, "User manual STM32 Nucleo-144 boards (MB1137)," no. August, pp. 1–75, 2020.

[16] STMicroelectronics, "STM32F756xx Datasheet" vol. 15, no. 2, pp. 1–23, 2016.

[17] STMicroelectronics, "Data Brief - Integrated development environment for STM32 products," no. November, 2019, [Online]. Available: https://www.st.com/resource/en/data_brief/stm32cubeide.pdf.

[18] ST, "Description of STM32F7 HAL and low-layer drivers Introduction," Data Base, vol. 3304, no. January, pp. 1–148, 2012, [Online]. Available: https://www.st.com/resource/en/user_manual/dm00189702-description-of-stm32f7-hal-and-lowlayer-drivers-stmicroelectronics.pdf.

[19] BOSCH, "BME280 Combined humidity and pressure sensor", no. September, pp. 1-55, 2018.

[20] Aosong, "Temperature and Humidity Module DHT11 Product Manual", pp. 1-9, 2015.

[21] D-Robotics, "DHT11 Humidity & Temperature Sensor", pp. 1-10, 2018.

[22] R. Fielding и др, " Hypertext Transfer Protocol -- HTTP/1.1 " The Internet Society, pp. 7-10, 1999.

[23] M. Myers, "Ethernet Basics," CompTIA Network+ Certif. Exam N10-007, pp. 69–89, 2018.

[24] P. Chang, L. Mac Vittie, T. M. Manager, and A. Services, "The Fundamentals of HTTP," 2008.

[25] M. Maksimovic, „Prenos podataka", Istocno Novo Sarajevo, Zavod za udzbenike i nastavna sredstva, 2021, pp. 141-148.

[26] E. Engineers, - Ethernet Technologies -. 2012.

[27] ST, "STM32Cube_LwIP_user_manual," no. September, pp. 1–41, 2008.

[28] A. Dunkels, "Design and Implementation of the lwIP TCP/IP Stack," Swedish Inst. Comput. Sci., p. 46, 2001, [Online]. Available: http://www.sics.se/~adam/lwip/doc/lwip.pdf%0Ahttp://www.ece.ualberta.ca/~cmpe401/fall2004/labs/docs/lwip.pdf%0Ahttp://citeseerx.ist.psu.edu/viewdoc/summary?doi=1 0.1.1.109.1795.

[29] https://savannah.nongnu.org/projects/lwip/

[30] M. S. Wibawa, K. Putra, "Design of air temperature and humidity measurement based on Arduino ATmega 328P with DHT22 sensor", International Journal of Physical Sciences and Engineering, vol. 6, no. April, pp. 9–17, 2022, https://doi.org/10.53730/ijpse.v6n1.3065

[31] B. L. Samoila, R. Trif, S. Arad, I. Utu, "Temperature and humidity measurment device", Annals of the University of Petrosani, Electrical Engineering, 22, pp. 1-6, 2020.

[32] Ya Min Thaw Dar Aung, Cho Cho Myint2, "Microcontroller Based Room Temperature and Humidity Measurement System", International Journal of Trend in Scientific Research and Development, vol. 3, no. August, pp. 1-3, 2019.

[33] X. Tang, C. Tan, A. Chen, Z. Li, R. Shuai, "Design and implementation of temperature and humidity monitoring system for small cold storage of fruit and vegetable based on Arduino", Journal of Physics: Conference Series, pp. 1-7, 2020, https://doi.org/10.1088/1742-6596/1601/6/062010.

Tijana Begović received the B.Sc. degree in Electrical Engineering from the University of East Sarajevo, Bosnia and Herzegovina. She is working as a teaching assistant and researcher at the Faculty of Electrical Engineering. The field o research interest include embedded systems and IoT

Zorana Mandić received the B.Sc. and M.Sc. degree in Electrical Engineering from the University of East Sarajevo, Bosnia and Herzegovina. She is working as a teaching assistant and researcher at the Faculty of Electrical Engineering. The field of research interest include automation and electronics.

Nikola Kukrić received the B.Sc. and M.Sc. degree in Electrical Engineering from the University of East Sarajevo, Bosnia and Herzegovina. He is working as a teaching assistant and researcher at the Faculty of Electrical Engineering. The field of research interest include embedded systems and IoT

Slobodan Lubura is a Full Professor at Faculty of Electrical Engineering, University of East Sarajevo, Bosnia and Herzegovina. His current research interests are embedded systems, industrial automation and robotics.