ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25may1890

Smart Soil-Less Greenhouse Management: A Desktop Application for Monitoring and Control

Nihad Fattahzada[1]; Umid Asgarzada[2]; Artoghrul Musayev[3]

[1,2,3]Department of Engineering Mathematics and Artificial Intelligence
Azerbaijan Technical University Baku, Azerbaijan

[1]ORCID: https://orcid.org/0009-0005-0124-2364).
[2]ORCID: https://orcid.org/0009-0002-9083-158X).
[3]ORCID: https://orcid.org/0009-0007-6618-0832).

Publication Date: 2025/06/05

Abstract: This paper proposes a simulation-based smart greenhouse management system specifically designed for soilless agriculture applications. Integrating an Arduino Uno microcontroller with a desktop application built in C#, the system provides real-time monitoring and precise control of critical environmental parameters including temperature, humidity, electrical conductivity (EC), pH levels, light intensity, and water management. A structured serial communication protocol facilitates robust data exchange, allowing immediate sensor feedback and dynamic adjustment of actuator operations. Using Proteus Design Suite, the system is thoroughly validated in a virtual simulation environment, enabling comprehensive testing of automated responses under various stress scenarios without physical hardware. Results demonstrate effective autonomous control, consistent performance, and reliable interaction between embedded systems and user interface components. The presented framework offers a cost-efficient, adaptable, and user-friendly solution, paving the way for future IoT integration and AI-driven adaptive strategies in precision agriculture.

Keywords: *Smart Greenhouse, Soilless Agriculture, Arduino Uno, Simulation-Based Validation, Desktop Application, Real-Time Monitoring, Automated Control, Proteus Simulation, Serial Communication, Precision Agriculture, Environmental Parameters, IoT Integration.*

How To Site: Nihad Fattahzada; Umid Asgarzada; Artoghrul Musayev (2025) Smart Soil-Less Greenhouse Management: A Desktop Application for Monitoring and Control. *International Journal of Innovative Science and Research Technology*, 10(5), 3387-3396. https://doi.org/10.38124/ijisrt/25may1890

I. INTRODUCTION

➢ *Background and Motivation*

Traditional agriculture methods, though foundational, are increasingly challenged by environmental factors, resource scarcity, and climate unpredictability [1]. To address these issues, modern agricultural practices have evolved toward controlled environment agriculture, notably through soilless techniques such as hydroponics and aeroponics. These innovative methods offer substantial benefits, including increased productivity, optimized resource use, and reduced environmental impact [2]. Consequently, the integration of automated systems within greenhouse management has gained considerable attention from researchers and industry professionals.

➢ *Problem Statement*

Managing the ideal conditions for plant growth within a greenhouse environment is inherently complex, requiring constant monitoring and adjustment of parameters such as temperature, humidity, nutrient concentration (pH and EC), and light intensity [3]. Manual management approaches often result in suboptimal crop yields due to human error, inconsistencies in control, and delayed responses to changing conditions [4]. Moreover, manual methods can significantly increase operational costs and labor requirements, highlighting the necessity for automated and precise control systems.

➢ *Importance and Relevance of Automated Greenhouse Systems*

The adoption of automation in greenhouse management enhances operational efficiency, reduces resource waste, and ensures consistently optimal environmental conditions [5]. Automated systems leverage sensors and actuators to precisely regulate essential parameters, thereby improving crop quality, increasing yields, and reducing overall management costs [6]. These systems also facilitate remote monitoring and control, further streamlining operational processes and enabling immediate responses to environmental changes [7].

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25may1890

➢ *Objective of the Study*

The primary objective of this study is to develop a comprehensive simulation-based automated greenhouse management system, integrating an Arduino-based embedded system with a desktop application for real-time monitoring and control. This system aims to provide an efficient and user-friendly platform for managing key environmental parameters within a simulated soilless greenhouse environment, thus demonstrating the feasibility and benefits of such technology prior to physical implementation.

➢ *Scope and Contribution*

The scope of this project encompasses the design and implementation of a simulated automated greenhouse control system, including:

- An Arduino-based embedded system to acquire data from sensors and control actuators based on predefined and user-configurable thresholds.
- A desktop application developed in C# (Windows Forms), providing a graphical user interface (GUI) for remote real-time monitoring and comprehensive control of the simulated greenhouse environment.

This study contributes significantly by offering a robust, fully integrated simulation environment that can be utilized for thorough testing and validation without the initial need for physical hardware investment, thus benefiting researchers and practitioners in precision agriculture.

II. LITERATURE REVIEW

➢ *Overview of Soilless Cultivation Systems*

In recent decades, the global agricultural sector has increasingly adopted soilless cultivation methods, particularly hydroponics, as a viable alternative to traditional farming due to land scarcity and resource constraints. Hydroponics leverages nutrient-rich aqueous solutions for plant cultivation, substantially enhancing water efficiency—reportedly reducing water usage by approximately 90% compared to conventional soil-based methods [8]. Urban agricultural initiatives have especially benefited from hydroponics, facilitating localized, sustainable food production that mitigates urban food insecurity [9].

➢ *Greenhouse Automation Technologies and Their Impact*

Automated greenhouse systems, which leverage sensors and actuators to maintain ideal growth environments, have revolutionized precision agriculture. These systems dynamically control critical environmental parameters—such as temperature, humidity, nutrient solution quality (pH and EC), and lighting—significantly boosting yields and product quality while simultaneously reducing resource waste [10]. Automation reduces dependency on manual monitoring, leading to improved labor efficiency, lower costs, and minimized errors associated with human intervention [11].

➢ *Embedded Systems for Agricultural Automation*

Microcontroller-based embedded systems have significantly enhanced the functionality and affordability of smart agriculture. Platforms such as Arduino have emerged prominently due to their open-source design, affordability, and ease of sensor integration, facilitating widespread adoption in greenhouse automation. Researchers have successfully demonstrated various Arduino-based control solutions that provide reliable environmental monitoring and real-time decision-making [12].

➢ *Communication Interfaces and Protocols*

Efficient greenhouse management requires robust and reliable communication protocols for seamless data transfer between embedded devices and monitoring interfaces. Commonly used wireless technologies include Bluetooth and Wi-Fi, whereas serial communication remains prevalent for its reliability and simplicity, particularly in simulation environments. Studies have indicated Bluetooth's suitability for short-range communication, due to its low-power consumption and straightforward implementation in embedded solutions [13].

➢ *Simulation-Based Validation of Greenhouse Systems*

Simulation environments are increasingly being adopted in agricultural technology development to validate system functionalities before physical implementation. Utilizing simulation tools such as Proteus significantly reduces development costs and accelerates testing cycles. These tools enable precise visualization and validation of interactions between embedded systems, sensors, actuators, and software interfaces, fostering more accurate system designs [14]. Desktop-based graphical interfaces enhance this simulation experience by allowing users to interact intuitively with the simulated environment, adjusting parameters and observing system responses in real-time.

➢ *Identified Research Gaps and Project Contribution*

While existing literature provides substantial evidence of advancements in automated greenhouse systems, there remain noticeable gaps, particularly in the integration of comprehensive hardware-software simulation environments. Few studies address the complete lifecycle of system development—from hardware simulation to fully integrated desktop application control interfaces—especially within the context of Arduino-based solutions. This research specifically addresses these gaps by developing a complete, simulation-based greenhouse automation system, combining robust Arduino hardware simulations with intuitive and interactive desktop software for effective monitoring and control.

III. SYSTEM DESIGN AND ARCHITECTURE

➢ *General System Architecture*

The greenhouse management system is centered around an Arduino Uno microcontroller, which continuously manages sensor readings, actuator controls, and data transmission. To maintain continuous operation and prevent system resets, it is recommended that the Arduino is always powered, preferably supported by an Uninterruptible Power Supply (UPS) and a backup generator. Power interruptions require the system to be rebooted, potentially disrupting greenhouse operations [15].

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25may1890

Sensor data is collected by the Arduino via analog or I²C inputs. These sensor readings are processed and displayed locally on a 16x2 LCD screen, providing immediate feedback regarding current environmental conditions inside the greenhouse. Simultaneously, this sensor data is transmitted in real-time to a desktop application, which is responsible for setting and managing the thresholds necessary for actuator operations. Actuators (e.g., pumps, lights, heating elements, humidifiers, fans) are subsequently controlled via relay modules, based on these predefined value ranges [16].

➢ *Hardware Subsystem*

The hardware subsystem is designed to precisely monitor and control greenhouse environmental parameters. Detailed information about the sensors used and their respective roles is given in Table 1.

Table 1 Sensor Specifications and Functions

Sensor Type	Measurement Purpose	Interface
SHT25	Greenhouse air temperature and humidity	I²C
PH Sensor	Water pH level (irrigation water quality)	Analog
EC Sensor	Electrical conductivity of water	Analog
HC-SR04 Ultrasonic	Water level in storage tank	Digital (Echo/Trig)
NTC Thermistor	Temperature of water in storage tank	Analog
DS1307 RTC	Real-time clock for scheduled operations	I²C
LDR	Ambient daylight intensity	Analog
Wind sensors	Wind speed and direction	Analog/Digital

The system actuators, controlled through relays, manage environmental conditions autonomously or manually based on set thresholds. Actuator types and their functionality include ventilation, humidification, water pumping (both irrigation and tank refilling), artificial lighting, heating, and manual valve control for water tank drainage [17].

Figure 1 shows a representation of the system connections on Proteus 8.13.

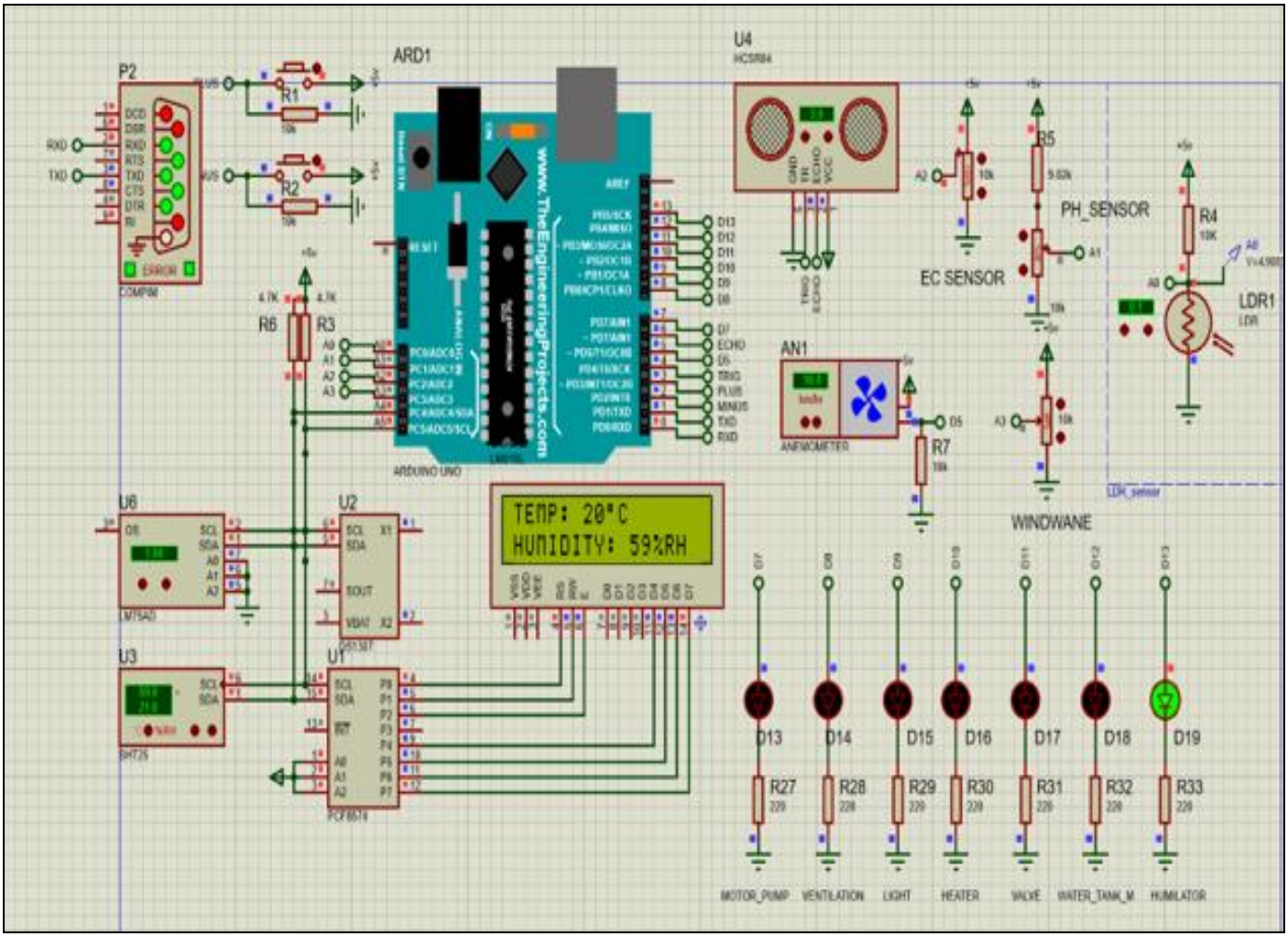

Fig 1 Schematic Connection of Arduino Uno's Sensors and Devices to Relays (Leds) on Proteus 8.13

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25may1890

➢ *Communication Protocol*

Communication between the Arduino and the desktop application is implemented via a structured serial communication protocol at a baud rate of 9600 bps. This protocol ensures consistent and reliable data transmission essential for real-time monitoring and control.

The structure of the transmitted data is specifically formatted for easy parsing, with clearly defined start and end markers for each sensor or actuator data group. An example data string transmitted from Arduino to the desktop software is as follows:

B8bT0.00tHovfhK273kE0.00eL277.44lP4pW51wJ50jD2dO0oQ20-25qR60-70rY400-0yU50-100uI0-10iF5-7fG6:30:0gZ15:30:0zX17:30:0xN14:3

➢ *Interpretation of Data Packet:*

- Sensor and actuator statuses are encapsulated between upper-case (start) and lower-case (end) letters.
- Time-based operation intervals and actuator thresholds are also included clearly within the packet structure [18].
- Table 2 shows a complete explanation of the data range from Arduino to the application.

Table 2 Explanation of the Data Set Coming from the Arduino to the Application

Format	Description	Unit/Range
T...t	Air temperature	°C
H...h	Humidity	%
E... e	Electrical conductivity (EC)	mS/cm
P... p	pH level	pH
W... w	Water level	Liters
L... l	Light intensity	Lux
K... k	Tank temperature	°C
D... d	Wind direction	Degrees/Index
O... o	Wind speed	km/h
Q... q	Temperature range	°C (min-max)
R... r	Humidity range	% (min-max)
Y... y	Light range	Lux (min-max)
U... u	Tank water range	Liters/cm (min-max)
I... i	EC range	mS/cm (min-max)
F... f	pH range	pH (min-max)
G... g	Light ON time	HH:MM: SS
Z... z	Light OFF time	HH:MM: SS
X...x	Irrigation start time	HH:MM: SS
N... n	Irrigation stop time	HH:MM: SS
V... v	Valve status	0: Closed, 1: Open
S	Stop/end bit	End of data packet

IV. IMPLEMENTATION

➢ *Embedded System Implementation*

The core logic of the system is programmed into the Arduino Uno microcontroller. Upon initialization (setup()), all sensors and actuators are set up. This includes configuring analog or I2C inputs for temperature, humidity, light, pH, EC, wind, and water level sensors, as well as initializing the DS1307 RTC module for time-based automation.

Interrupts are assigned to two hardware buttons for local LCD navigation. The 16x2 LCD displays sensor values and system status (temperature, humidity, EC, etc.). Once the setup phase is completed, the Arduino enters a continuous loop (loop()), performing the following operations(Figure 2):

- *Sensor Reading:*
  Values are continuously collected from all sensors.
- *Climate Control:*
  Based on temperature and humidity thresholds set via the desktop application, the heater, fan, and humidifier are automatically activated or deactivated.
- *Irrigation Logic:*
  If EC and pH levels are within the acceptable range, irrigation starts based on scheduled watering times and tank level availability.
- *Light & Ventilation:*
  These are scheduled using the RTC and daylight readings from the LDR.
- *Serial Communication*:
  Sends current readings to the desktop application and receives new threshold or timing commands.

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25may1890

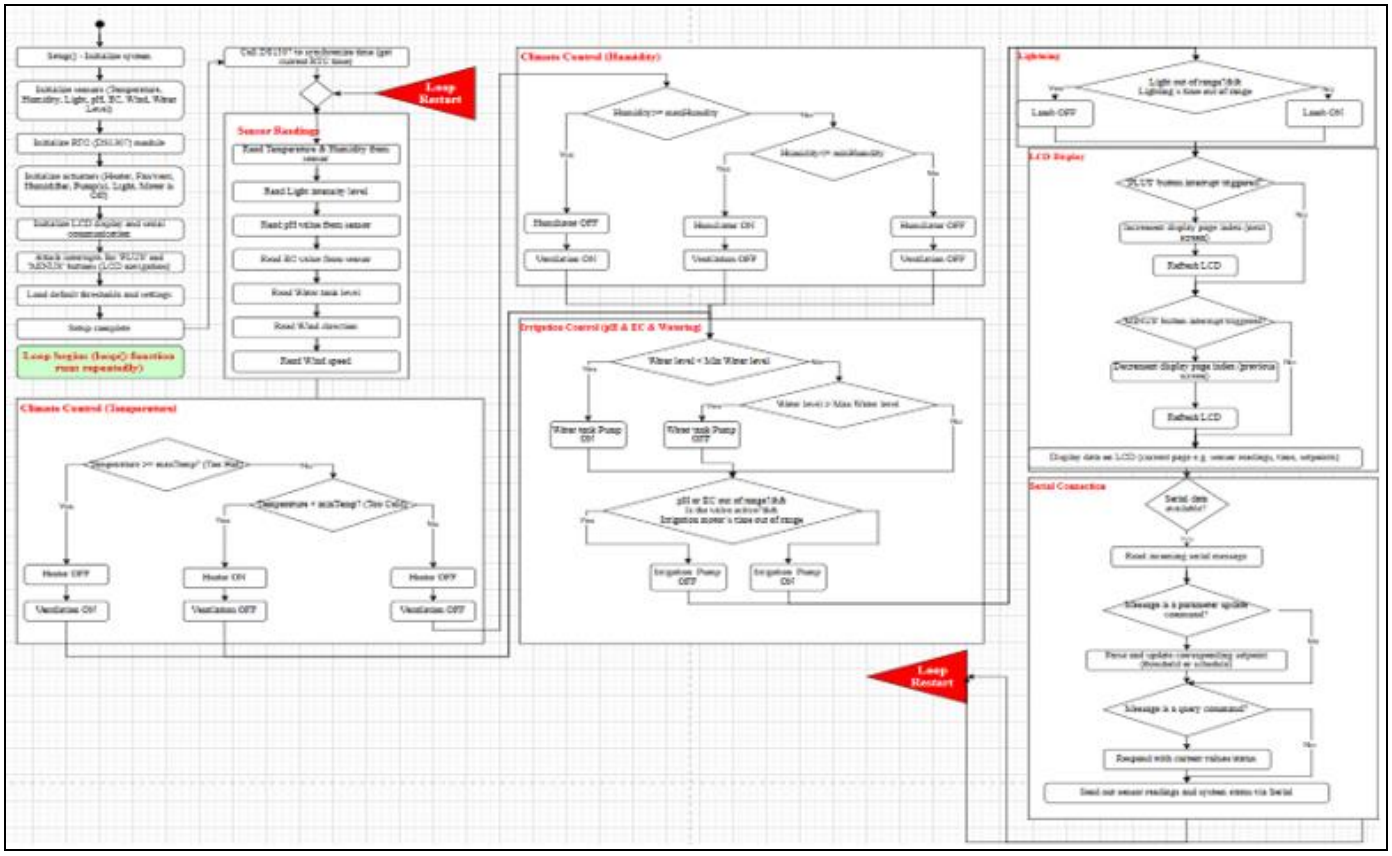

Fig 2 Flowchart Diagram of Code

➢ *Desktop Application Development*

The desktop application was developed using the C# Windows Forms framework. It consists of several key interfaces:

- *Home Tab:*

Displays real-time status of all actuators such as watering motor, heater, light, and ventilation (Figure 3).

Fig 3 "HOME" Page of Application

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25may1890

- *Port Config Tab:*
  Allows the user to select COM port and baud rate (9600) to establish a serial connection with the Arduino (Figure 4).

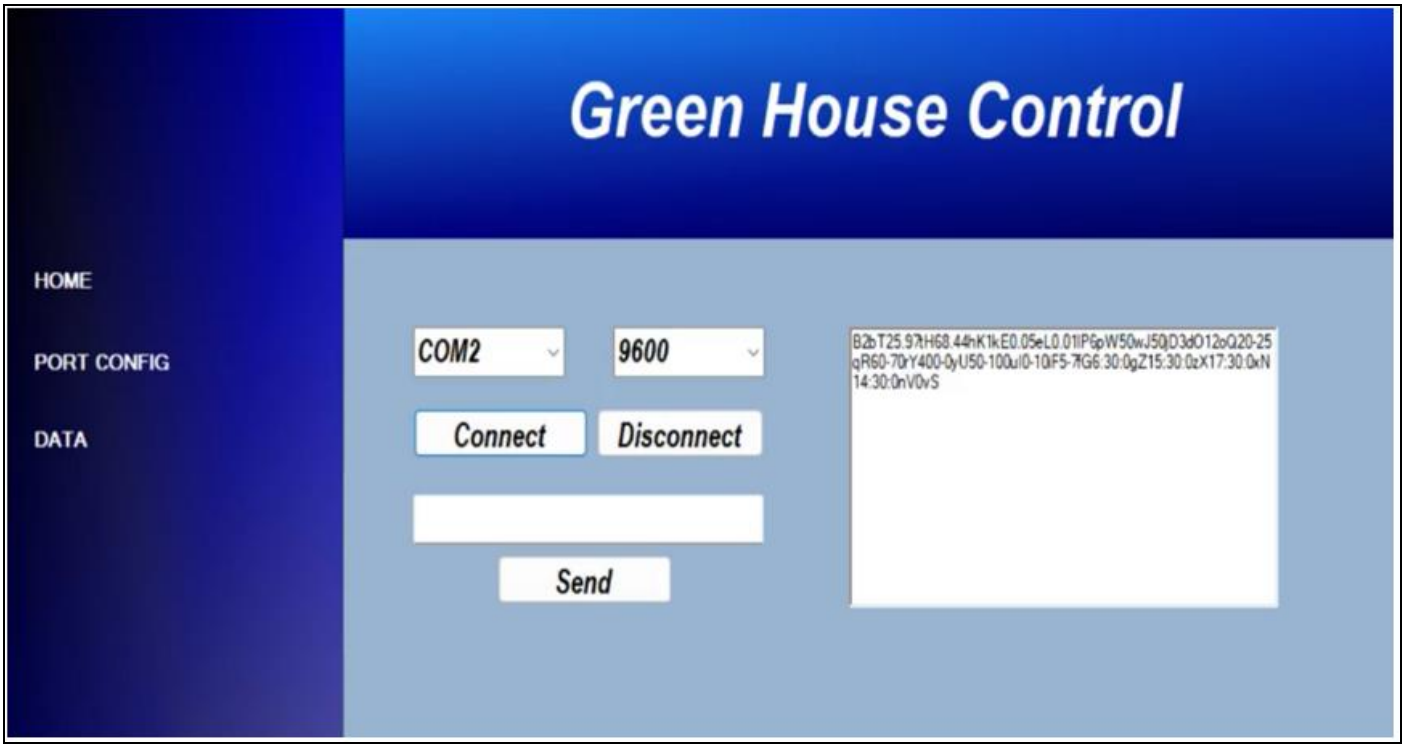

Fig 4 "PORT CONFIG" Page of Application

- Sensor *Data* Tab:
  Visualizes current sensor readings in a user-friendly dashboard style (Figure 5). Each module (temperature, pH, light, wind, water tank) displays the live value along with its timestamp.

Fig 5 "SENSOR DATA" Page of Application

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25may1890

- *Controls Tab:*

Users can input threshold values for each sensor parameter and define active periods for lighting and watering (Figure 6). Upon pressing SEND, a formatted string is transmitted to Arduino to update its control logic.

Fig 6 "Controls" Page of Application

The GUI ensures real-time monitoring and parameter updates through structured serial parsing, and updates reflect instantly based on the Arduino's feedback.

➢ *Serial Communication Protocol*

A crucial component in the operation of the greenhouse simulation system is the serial communication interface between the Arduino Uno and the desktop application. This interface facilitates two-way data exchange, enabling the application to retrieve sensor values and send control parameters to the embedded system.

The communication is established via a USB serial port using a baud rate of 9600 bps, which ensures reliable transmission without significant delays or data loss. The serial protocol follows a structured message format wherein the application sends concatenated command strings to define operational thresholds for various subsystems, such as lighting, irrigation, temperature, and pH control.

The desktop application transmits data in predefined textual formats that are parsed on the Arduino side. Each message string begins with a keyword representing the target parameter (e.g., Temp, Motor, Valve), followed by delimiters such as colons (:), hyphens (-), and semicolons (;) to segment the control values.

➢ *Typical Formats İnclude:*

- Temperature/Humidity Range: Temp:20-25;
- Lighting Schedule: Lumen:06-30_00+14-30_00;
- Irrigation Timing: Motor:15-30_00+17-30_00;
- Valve Control: Valve:1;
- These messages are parsed using string-handling functions which extract numeric values for the control logic. The Arduino code employs routines such as findValue() and find_start_stop() to interpret these strings and store the values in configuration structures(Figure 7).

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25may1890

```
time_calibration[0]=include_time(Light_Motor_ST[0].start,":", ";", "-", "_");
time_calibration[1]=include_time(Light_Motor_ST[1].start,":", ";", "-", "_");
time_calibration[2]=include_time(Light_Motor_ST[1].stop,":", ";", "-", "_");
time_calibration[3]=include_time(Light_Motor_ST[0].stop,":", ";", "-", "_");



if (Serial.available()) {
  String text = Serial.readString();

  controls_calibration[0]=findValue(text,"Temp",":",";","-",controls_calibration[0]);
  controls_calibration[1]=findValue(text,"Humd",":",";","-",controls_calibration[1]);
  controls_calibration[2]=findValue(text,"Light",":",";","-",controls_calibration[2]);
  controls_calibration[3]=findValue(text,"WTank",":",";","-",controls_calibration[3]);
  controls_calibration[4]=findValue(text,"EC",":",";","-",controls_calibration[4]);
  controls_calibration[5]=findValue(text,"PH",":",";","-",controls_calibration[5]);

  Light_Motor_ST[0]=find_start_stop(text, "Lumen", ":", ";", "+", Light_Motor_ST[0]);
  Light_Motor_ST[1]=find_start_stop(text, "Motor", ":", ";", "+", Light_Motor_ST[1]);

  if(text.startsWith("Valve")){
    String selection_str = text.substring(text.indexOf(":") + 1, text.indexOf(";"));
    controls_start[1]=atoi(selection_str.c_str());
  }
  else{
    controls_start[1]=controls_start[1];
  }
}
```

Fig 7 Parsing Text from the Program

This allows real-time parameter updates without reprogramming the microcontroller. The modularity and transparency of this method enhance usability and system flexibility, which are essential for adaptive greenhouse environments [19],[20].

In the opposite direction, Arduino transmits real-time data readings back to the application, formatted similarly with symbol-paired tags (e.g., T...t, H...h, etc.). This returned data includes current sensor values and device statuses. These values are visualized on the graphical dashboard, while the predefined settings are reflected for validation (Table 2).

This two-way architecture ensures synchronization between hardware control and application-level interface, enhancing the automation, monitoring, and fine-tuning of all environmental parameters within the greenhouse[5],[6].

V. SIMULATION ENVIRONMENT

The proposed greenhouse control system was validated using a simulation-based methodology developed in Proteus Design Suite. This approach enabled virtual emulation of microcontroller hardware and facilitated direct testing of firmware with simulated sensors and actuators. The simulation included a virtual Arduino Uno, environmental and nutrient sensors, and actuator modules interfaced through digital output pins.

All sensors—temperature, humidity, pH, EC, light, and water level—were represented by adjustable analog or digital inputs. Outputs were simulated using LEDs to reflect device activation. The system's firmware, written in C/C++, was embedded into the microcontroller block, allowing real-time behavioral observation.

Key test cases included environmental stress scenarios such as temperature spikes, humidity drops, light deficiency, dry substrate, and nutrient imbalance. These cases verified that the system could respond autonomously by activating the relevant actuators, such as fans, heaters, lights, pumps, and valves. The simulation supported scenario tuning, enabling dynamic modification of input signals and observation of control response.

Proteus's virtual UART terminal was connected to the PC via a virtual COM port bridge. This enabled live interaction between the Arduino and the desktop GUI, simulating real-time data exchange. Commands from the PC updated sensor thresholds or triggered manual overrides, while the Arduino continuously sent back sensor readings. This two-way communication validated the robustness of the system's architecture in a non-physical environment[21], [22], [23].

VI. RESULTS AND DISCUSSION

Simulation tests confirmed that the embedded system operated as intended. For each threshold breach, appropriate actuator responses were triggered without delay. Temperature control, for instance, engaged ventilation when ambient temperature exceeded the configured maximum. Similarly, humidity control activated the humidifier under dry conditions, maintaining a stable internal climate.

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25may1890

The light control module demonstrated correct scheduling and light compensation behavior. The grow lights activated during low LUX readings or per user-defined time intervals. For pH and EC, the system reacted by stopping irrigation when values exceeded permissible ranges. Once corrected, it resumed scheduled operations, avoiding potential harm to plants.

All communication between the Arduino and desktop interface was consistent. Sensor data was parsed accurately by the GUI, and user-defined setpoints were successfully received and applied by the microcontroller. No data loss or command failures were observed. The round-trip latency between the desktop interface and actuator response remained within real-time tolerances.

Overall, the simulation outcomes confirmed that the proposed system was functionally stable, responsive, and adaptable. While threshold-based control inherently limits fine-grained regulation, it proved sufficient for the system's purpose. The behavior observed aligns with established models of autonomous greenhouse management [24], [25], [26], [27].

VII. CONCLUSION

This study presents a complete simulation framework for the autonomous control of a soilless greenhouse. The system integrates an Arduino Uno microcontroller, a suite of environmental and nutrient sensors, and a desktop interface for supervisory control. Through simulation, the system demonstrated real-time monitoring, autonomous actuation, and robust serial communication.

Key contributions include the design of a dual-layer architecture combining local control with user oversight, validation of control strategies through scenario-based testing, and demonstration of simulation-based development in smart agriculture.

Future directions involve integrating IoT connectivity for cloud-based monitoring, applying AI for predictive and adaptive control, and deploying the system in a physical greenhouse for empirical validation. These enhancements aim to increase automation, improve resource efficiency, and support scalable smart farming solutions.

REFERENCES

[1]. R. Ramin Shamshiri *et al.*, "Advances in greenhouse automation and controlled environment agriculture: A transition to plant factories and urban agriculture," *International Journal of Agricultural and Biological Engineering*, vol. 11, no. 1, pp. 1–22, 2018,[CrossRef]

[2]. N. S. Mattson, *Controlled Environment Agriculture: Principles and Practices*, 2022. [Online]. Available: [CrossRef]

[3]. S. Mauget, M. Ulloa, and J. Dever, "Planting Date Effects on Cotton Lint Yield and Fiber Quality in the U.S. Southern High Plains," *Agriculture*, vol. 9, no. 4, p. 82, Apr. 2019, [CrossRef]

[4]. H. G. Jones, *Plants and microclimate: a quantitative approach to environmental plant physiology*, Third edition. Cambridge ; New York: Cambridge University Press, 2014. [CrossRef]

[5]. E. Lichtfouse, Ed., *Sustainable Agriculture Reviews*, vol. 22. in Sustainable Agriculture Reviews, vol. 22. Cham: Springer International Publishing, 2017. [CrossRef]

[6]. A. Kochhar and N. Kumar, "Wireless sensor networks for greenhouses: An end-to-end review," *Computers and Electronics in Agriculture*, vol. 163, p. 104877, Aug. 2019, [CrossRef]

[7]. M. Herrero-Huerta, D. González-Aguilera, P. Rodriguez-Gonzalvez, and D. Hernández-López, "Vineyard yield estimation by automatic 3D bunch modelling in field conditions," *Computers and Electronics in Agriculture*, vol. 110, pp. 17–26, Jan. 2015, [CrossRef]

[8]. A. AlShrouf, "Hydroponics, Aeroponic and Aquaponic as Compared with Conventional Farming," vol. 27, no. 1, 2017. [CrossRef]

[9]. C. Eigenbrod and N. Gruda, "Urban vegetable for food security in cities. A review," *Agron. Sustain. Dev.*, vol. 35, no. 2, pp. 483–498, Apr. 2015, [CrossRef]

[10]. Y. Liu and C. Bi, "The Design of Greenhouse Monitoring System Based on ZigBee WSNs," in *22017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, Guangzhou, China: IEEE, Jul. 2017, pp. 430–433. [CrossRef]

[11]. M. Srbinovska, C. Gavrovski, V. Dimcev, A. Krkoleva, and V. Borozan, "Environmental parameters monitoring in precision agriculture using wireless sensor networks," *Journal of Cleaner Production*, vol. 88, pp. 297–307, Feb. 2015, [CrossRef]

[12]. Y. A. Badamasi, "The working principle of an Arduino," in *2014 11th International Conference on Electronics, Computer and Computation (ICECCO)*, Abuja, Nigeria: IEEE, Sep. 2014, pp. 1–4. [CrossRef]

[13]. T. Ojha, S. Misra, and N. S. Raghuwanshi, "Wireless sensor networks for agriculture: The state-of-the-art in practice and future challenges," *Computers and Electronics in Agriculture*, vol. 118, pp. 66–84, Oct. 2015, [CrossRef]

[14]. D. Ibrahim, *Microcontroller-based temperature monitoring and control*. Oxford: Newnes, 2002. [CrossRef]

[15]. N. K. Nawandar and V. R. Satpute, "IoT based low cost and intelligent module for smart irrigation system," *Computers and Electronics in Agriculture*, vol. 162, pp. 979–990, Jul. 2019, [CrossRef]

[16]. Computer Engineering Department, LPU-Cavite, Philippines *et al.*, "Development of IoT Smart Greenhouse System for Hydroponic Gardens," *IJCSR*, vol. 7, pp. 2111–2136, Jan. 2023, [CrossRef]

[17]. S. Saha, Md. M. Hemal, A. Rahman, and K. Nur, "An Iot and Machine Learning-Driven Advanced Greenhouse Farming System for Precision Agriculture," 2023, *SSRN*. [CrossRef]

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25may1890

[18]. M. Roopaei, P. Rad, and K.-K. R. Choo, "Cloud of Things in Smart Agriculture: Intelligent Irrigation Monitoring by Thermal Imaging," *IEEE Cloud Comput.*, vol. 4, no. 1, pp. 10–15, Jan. 2017, [CrossRef]

[19]. J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, Aug. 2008, [CrossRef]

[20]. J. P. Lynch, "A Summary Review of Wireless Sensors and Sensor Networks for Structural Health Monitoring," *The Shock and Vibration Digest*, vol. 38, no. 2, pp. 91–128, Mar. 2006, [CrossRef]

[21]. D. Cika and D. Grundler, "Proteus Virtual System Modelling used for microcontroller education". [CrossRef]

[22]. I. S. Laktionov, O. V. Vovna, A. A. Zori, and V. A. Lebedev, "Results of Simulation and Physical Modeling of the Computerized Monitoring and Control System for Greenhouse Microclimate Parameters," *International Journal on Smart Sensing and Intelligent Systems*, vol. 11, no. 1, pp. 1–15, Jan. 2018, [CrossRef]

[23]. A. Sagheer, M. Mohammed, K. Riad, and M. Alhajhoj, "A Cloud-Based IoT Platform for Precision Control of Soilless Greenhouse Cultivation," *Sensors*, vol. 21, no. 1, p. 223, Dec. 2020, [CrossRef]

[24]. H. Hu, L. Xu, R. Wei, and B. Zhu, "Multi-Objective Control Optimization for Greenhouse Environment Using Evolutionary Algorithms," *Sensors*, vol. 11, no. 6, pp. 5792–5807, May 2011, [CrossRef]

[25]. Department of Crop Science, Laboratory of Vegetable Crops, Agricultural University of Athens, Athens, Greece, D. Savvas, N. Gruda, and Department of Horticulture, University of Bonn, Germany, "Application of soilless culture technologies in the modern greenhouse industry – A review," *Europ.J.Hortic.Sci.*, vol. 83, no. 5, pp. 280–293, Nov. 2018, [CrossRef]

[26]. A. Lytos, T. Lagkas, P. Sarigiannidis, M. Zervakis, and G. Livanos, "Towards smart farming: Systems, frameworks and exploitation of multiple sources," *Computer Networks*, vol. 172, p. 107147, May 2020, [CrossRef]

[27]. M.-H. Lee, M.-H. Yao, P.-Y. Kow, B.-J. Kuo, and F.-J. Chang, "An Artificial Intelligence-Powered Environmental Control System for Resilient and Efficient Greenhouse Farming," *Sustainability*, vol. 16, no. 24, p. 10958, Dec. 2024, [CrossRef]