Microcontroller-based system for automated room temperature monitoring and actuation ⋆

Borys Zlotenko[1,†], Oleg Synuyk[2,†], Dmytro Statsenko[1,†], Serhiy Horiashchenko[2,†] and Olha Kravchuk[2,†]

[1] Kyiv National University of Technologies and Design, 2, Mala Shyianovska, Kyiv, 01011, Ukraine
[2] Khmelnytskyi national university, 11, Instytuts'ka str, Khmelnytskyi, 29016, Ukraine

Abstract

This paper explores the design and implementation of a wireless indoor climate control system leveraging Arduino microcontrollers and cost-effective sensor technology. The setup features a central control unit (Arduino UNO) and a peripheral actuator unit (Arduino Nano), which communicate using 433 MHz radio modules. Real-time monitoring of indoor conditions is achieved through a temperature sensing mechanism based on a thermistor, enabling the regulation of heating or cooling appliances via relay-controlled switching. To ensure precise temperature detection, the resistance-temperature behavior of an NTC thermistor (mf52-103 3435) was evaluated experimentally and modeled using both the Steinhart–Hart equation and a simplified logarithmic approach. Comparative analysis with the DHT11 digital temperature sensor revealed a deviation of less than 0.5 °C, confirming the thermistor's reliability for practical climate monitoring applications. The system's control logic, programmed in C++ using the Arduino IDE, integrates user interface navigation along with automated relay operations guided by predefined temperature thresholds. A simulated daily scenario demonstrated potential energy savings of up to 40% compared to manual operation of appliances. This study highlights that microcontroller-driven automation offers a user-friendly, energy-efficient, and adaptable solution for controlling indoor environments in residential settings.

Keywords

Climate, controller, control system, Arduino, radio, temperature thermistor

1. Introduction

Nowadays, automation systems of residential buildings are widely used within such areas of information technology development as "Smart Home" and "Internet of Things" [1]. As a rule, such systems are quite complex and control lighting, heating, safety devices, kitchen appliances, etc. [2]. In this regard, complex automation of home equipment requires significant material costs and qualified adjustment and service.

Partial automation systems in everyday life, for example, indoor microclimate control systems, may become more accessible to the mass consumer. To regulate the microclimate, various types of electrical appliances are used, such as air conditioners, fans, dehumidifiers, humidifiers, cleaners, combined appliances and others. All climate-controlled household appliances consume a lot of electricity due to the presence of heating or cooling elements.

One of the ways to save energy when using devices for heating and cooling the air in the room is to observe the set temperature regime. It is quite difficult to maintain the set parameters of the microclimate without an appropriate automation system, since a person cannot clearly react to a change in these parameters.

⋆ ITTAP'2025: 5rd International Workshop on Information Technologies: Theoretical and Applied Problems, October 22–24, 2025, Ternopil, Ukraine, Opole, Poland

* Corresponding author.
† These authors contributed equally.
kems@knutd.edu.ua (B. Zlotenko); oleg.synyuk@khmnu.edu.ua (O. Synuyk); statsenko.v@knutd.edu.ua (D.Statsenko); tnt7@ukr.net (S.Horiashchenko); kravchukoa2@gmail.com (Olha Kravchuk)
(0000-0002-0870-8535 (B. Zlotenko); 0000-0002-9615-0729 (O. Synuyk); 0000-0002-3932-792X (D.Statsenko); 0000-0001-6623-2523 (S.Horiashchenko); 0000-0001-6937-5001 (Olha Kravchuk)

When developing an automation system, it should be taken into account that electrical household appliances are portable, have already installed means for connecting to the electrical network, and can be used in different parts of the room or in different rooms.

Formulation of the problem. Therefore, in order to eliminate the "human factor" from the process of managing the microclimate in rooms using electrical household appliances, an automated control system with wireless communication should be used.

2. Analysis of existing temperature control system

Smart thermostats exemplify a contemporary evolution of conventional models, providing improved functionality and ease. Featuring sensors, Wi-Fi, and complex algorithms, these gadgets adjust to user choices and modify heating and cooling settings intelligently. By examining routines and patterns, they can automatically refine temperature settings to boost comfort and energy efficiency. Additionally, users have the flexibility to control them remotely through mobile apps, providing convenient access to settings from virtually anywhere. These thermostats additionally produce detailed energy usage reports, allowing users to observe and regulate consumption more efficiently. Compatible with voice assistants such as Alexa, Google Assistant, and Apple HomeKit.

Adapts to user schedules, provides energy-saving suggestions, and integrates with other smart home devices. There are various temperature control systems available, each with its own set of features and benefits. Smart thermostats offer advanced control and convenience, while zoned HVAC systems and programmable thermostats provide more basic, but still effective, temperature management. Hydronic radiant heating and ductless mini-split systems offer unique advantages in terms of comfort and efficiency, and geothermal heat pumps present a sustainable, long-term solution. The choice of system depends on factors such as budget, building design, energy efficiency goals, and personal preferences.

Equivalent table 1 is presented below. An analysis of existing systems showed that modern solutions are quite expensive and require additional equipment for installation and maintenance. The comparison was also based on energy-saving factors. In today's environment, we need a simple and energy-efficient system that can control the set temperature in a given room. Therefore, our further developments will be based on the above factors: simplicity and energy efficiency. Temperature Sensors: DHT11, DHT22, or DS18B20 are commonly used to measure room temperature.

The aim of this research is to develop and experimentally validate a wireless automated indoor climate control system based on Arduino microcontrollers. The system is designed to maintain a user-defined temperature regime by controlling household heating and cooling appliances, while ensuring energy efficiency, affordability, and ease of implementation for residential applications. Research objectives:

1. To analyze existing temperature control systems and identify limitations in cost, complexity, and accessibility for general users.
2. To design a modular wireless system for indoor temperature regulation using Arduino-based microcontrollers and radio communication modules.
3. To implement a prototype climate control system consisting of central and peripheral Arduino units, temperature sensors, and relays for equipment control.
4. To develop software algorithms for user interface, temperature data acquisition, and wireless command transmission between nodes.
5. To conduct experimental measurements of temperature using the NTC thermistor and validate its accuracy by comparison with a standard digital temperature sensor (DHT11).
6. To evaluate the effectiveness and reliability of the developed system in maintaining the desired indoor temperature through automatic control logic.

3. Main part

The wireless system is built on the basis of an Arduino microcontroller and a radio transmitter and radio receiver operating at a frequency of 433 MHz.

One controller - Arduino UNO is central, it will read the temperature from the NTC thermistor, and this controller will also have an interface for communicating with the user in the form of a display and navigation tact switches [3, 4].

The second controller, an Arduino Nano, is installed directly at the site of the climate equipment. Its compact design makes it the ideal choice for handling the system's requirements.

The Arduino Nano platform, built on the microcontroller ATmega328 (Arduino Nano 3.0) or ATmega168 (Arduino Nano 2.x), has small dimensions and can be used in laboratory stands. It has similar functionality to the Arduino UNO, but differs structurally. The difference lies in the absence of a DC power connector and operation via a Mini-B USB cable. The advantages of this controller include its small size, which allows it to be integrated into almost any device without losing its functional properties.

The system is equipped with two separate power supplies: a 6V supply dedicated to the Arduino Uno controller and another 6V supply for the Arduino Nano controller. The main task of the control system is to receive data from the temperature sensor, process them in the main controller and, according to the programmed logic, send a control signal to the receiver, and turn on and off the corresponding equipment. As soon as the temperature reaches the one set by the user, the transmitter will send a command to turn off the heating/cooling device. For such automatic control and regulation, it is most convenient to use a radio module, which has a number of advantages over other methods of wireless communication.

Numerous communication modules give you the option to select a module fitting the needs and functions of a specific system. Thus, for systems with remote communication (access to devices with a communication module over extensive distances through communication networks), it's preferable to utilize GPS or WiFi modules. By connecting the first to the mobile network, and the second to the Internet, you can receive or transmit data over long distances using the appropriate communication protocols.

The Bluetooth module can also be used remotely, but only in combination with a Bluetooth device that is within the range of the signal and sends or receives data via the Internet.

The best component for constructing an automated system within the house is a radio module that functions without supplementary network distribution equipment and can be utilized to control and adjust the microclimate settings in the room.

A radio module with a frequency of 433 MHz is used to ensure wireless communication. There are two types of such a module - transmitter MX-FS-03V and receiver MX-05V. The signal is dispatched from the central board of the Arduino UNO, thus the MX-FS-03V transmitter is linked to the central board, and the MX-05V receiver is connected to the Arduino Nano.

An electromagnetic relay is used to shut or open an electrical circuit with particular variations in electrical or non-electrical input impacts. Within the control system, an electromagnetic relay is employed to switch devices to a 220 V AC network with a power of up to 2 kW. To ensure these parameters, we will use the SONGLE SRD-05VDC relay.

To provide feedback to the user, a system employs a display and tact switches as an interface. As a display, a 1602 LCD module, founded on the HD44780 controller, is utilized. This is among the most straightforward, inexpensive and widely-used displays for the creation of different electronic devices. It is found in both laboratory setups and in industrial apparatuses, like, for example, coffee makers. To communicate the user with the control system, we employ 4 tact switches to create a navigation menu.

An NTC thermistor MF52-103 3435 was used to measure the temperature in the room, with a range of operating temperatures from -30 to +125 ℃. Compared with analog temperature sensors such as LM35, TMP36, digital ones such as DS18B20, or thermocouples, thermistors are much cheaper, more reliable and not require additional elements when using, which are undoubted advantages.

The temperature sensor reads the current value of the temperature in the room and, by converting the analog value of the voltage level to the temperature value in degrees Celsius, transmits this value to the Arduino Uno controller. The controller processes this signal and displays it on the display. Next, it compares the temperature value received from the sensor with the one set by the user through the menu. If the temperature set by the user is higher than the actual temperature, then the controller gives a signal to the communication module to transmit to receiver 1. If the user's temperature is lower than the current one, 0 is sent.

The received signal decides if the receiver sends a 1 or 0 to the Arduino Nano controller. When the controller gets a 1, it outputs a high logic level, activating the relay; if it receives a 0, it outputs a low logic level, deactivating the relay. This method allows for temperature control by alternately switching climate equipment on and off. The Arduino Nano has an integrated COM driver port that mimics a virtual port for interacting with a computer. To set this up, just connect the controller to a computer's USB port, and then download and install the needed driver. After installation, confirm the existence of the virtual port by checking the corresponding COM port in the device manager [5]. To set up the control system, a functioning algorithm for the user menu was designed. This involved linking four tactile switches and three resistors to a single analog input and measuring digital voltage levels for each button function. As well, algorithms for the transmitter and receiver were created. A system operation sketch was made leveraging external libraries for communication modules and an LCD display module, ensuring smooth operation of all parts.

Arduino controllers were programmed in the Arduino IDE development environment, in the Wiring programming language, which is a type of the C++ language, but much simpler in its structure [6].

The operational framework of the proposed automated temperature control system, sheme of it is illustrated in Figure 1.

Arduino UNO: receives data from the thermistor, compares with the set value, displays on the screen, sends a command .

Arduino Nano: receives a signal via the radio module and turns the device on/off via a relay.

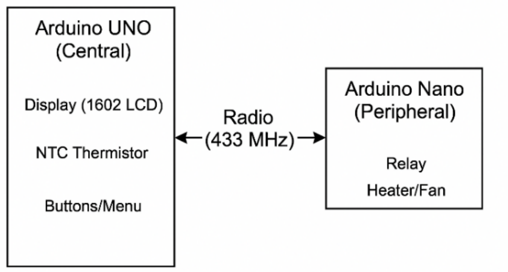

Figure 1: The scheme of operation automated temperature control system

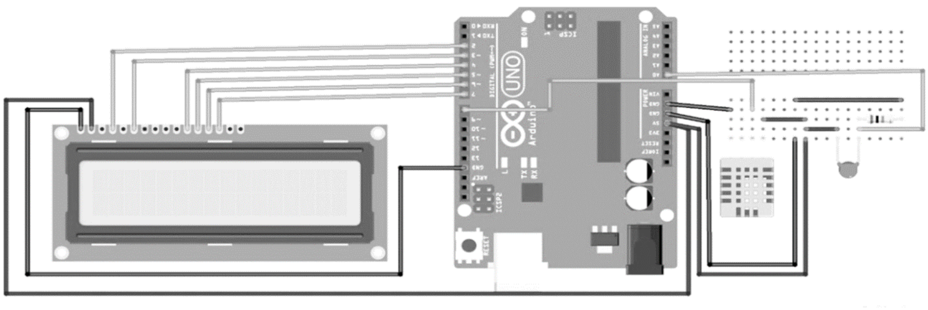

Figure 2: Simulated scheme of the laboratory bench for NTC thermistor study

A display is connected to digital pins 2 through 7, while a temperature sensor is attached to pin 8. The thermistor being analyzed is wired to the analog input A0. Both the display and the temperature sensor are additionally linked to their respective 5V and ground (GND) connections.

To convert the temperature into resistance, another resistor is connected in series (Fig. 3).

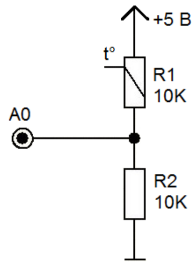

Figure 3: Example figure Connection diagram of the thermistor

Since a constant resistor with a nominal resistance of 10 kΩ is used, the output voltage (V_0), which is transmitted to the Arduino, is equal to:

$$V_0 = \frac{R}{(R+10000)*V_c}, \tag{1}$$

where V_c– power supply voltage (3.3 or 5 V).

When the ADC voltage is measured on the Arduino controller, a numerical value is obtained:

$$ADCvalue = \frac{V_0 * 1023}{V_c}, \tag{2}$$

Substituting (1) into (2), after transformations, we get:

$$ADCvalue = \frac{R}{\frac{(R+10000)*V_c*1023}{V_c}} = \frac{R}{(R+10000)*1023}, \tag{3}$$

In the final form, the expression for determining the resistance of the thermistor will look like this:

$$R = \frac{10000}{\frac{1023}{ADC} - 1}, \tag{4}$$

When measuring analog values, two approaches can be employed to enhance the accuracy of the readings. The first involves utilizing the 3.3 V pin for the analog signal, while the second entails gathering a small array of experimental values and calculating their average. In the conducted experiments, the first method was adopted. After measuring resistance during a temperature increase, the next phase is converting the resistance value to temperature. To accomplish this, the Steinhart-Hart equation can be employed, delivering a trustworthy approximation for the converted temperature readings:

$$\frac{1}{T} = a + b*\ln(R_T) + c*(\ln(R_T))^3, \tag{5}$$

where T – temperature in Kelvin; R_T – resistance at temperature T; a, b, c – coefficients of the Steinhart -Hart equation, which depend on the type and initial resistance of the thermistor.

The equation is quite complex and requires a large number of variable parameters, which may not be present for a specific thermistor. Instead of this equation, you can use a simplified equation [7]:

$$\frac{R_0}{R_T} = e^{B\left(\frac{1}{298\,K} - \frac{1}{T}\right)}, \tag{6}$$

where B - coefficient of temperature sensitivity. This coefficient is calculated on the basis of resistance values at two specific temperature values (in many cases, these temperatures are chosen as 25 °C and 100 °C); R_0 – resistance at room temperature. For the tested thermistor B = 3435K; $R_0 = 10\text{k}\Omega$.

Equation (6) can be rewritten as:

$$\frac{1}{T} = \frac{1}{298\,K} + \frac{1}{B} * \ln\left(\frac{R_T}{R_0}\right). \tag{7}$$

By substituting the numerical values of the quantities into equation (7), we get the formula for calculating the inverse temperature depending on the relative resistance of the thermistor mf52-103 3435:

$$\frac{1}{T} = 3{,}356 \times 10^{-3} + 2{,}911 \times 10^{-4} * \ln\left(\frac{R_T}{10}\right). \tag{8}$$

During the experiments, an LCD display was utilized to present the current temperature readings, derived from the thermistor's resistance as calculated using equation (8). These values were recorded in the microcontroller for processing. Table 1 presents the resistance measurements of the thermistor along with their matching temperature readings as displayed on the LCD screen. An examination of the derived connection allows the development of the subsequent formula:

$$\frac{1}{T} = 3{,}354 \times 10^{-3} + 2{,}907 \times 10^{-4} * \ln\left(\frac{R_T}{10}\right) + 2{,}317 \times 10^{-7} * \left(\ln\left(\frac{R_T}{10}\right)\right)^3. \tag{9}$$

The resulting expression can be written in the form:

$$\frac{1}{T} = 2{,}682 \times 10^{-3} + 2{,}907 \times 10^{-4} * \ln\left(R_T\right) + 2{,}317 \times 10^{-7} * \left(\ln\left(R_T\right)\right)^3. \tag{10}$$

Table 1
The temperature of the thermistor mf52-103 3435 according to its resistance

№	R кOm	t °C	№	R кOм	t °C
1	13,14	18,24	14	8,57	29,21
2	13,2	18,14	15	8,37	29,84
3	13,14	18,24	16	8,14	30,58
4	12,43	19,62	17	7,88	31,43
5	10,92	22,89	18	6,74	35,71
6	10,02	25,1	19	5,76	40,13
7	9,49	26,52	20	5,13	43,47

8	9,12	27,55	21	4,7	46,07
9	8,98	27,96	22	4,37	48,25
10	8,94	28,07	23	4,25	49,08
11	8,87	28,27	24	3,35	50,38
12	8,77	28,58	25	3,03	52,56
13	8,63	29	26	4,57	53,88

Dependence (10) is in good agreement with equation (5), in which the coefficients take values: $a = 2.682 \times 10^{-3}$; $b = 2.907 \times 10^{-4}$; $c = 2.317 \times 10^{-7}$

Graphical dependences corresponding to expressions (8) and (10) and the obtained data of the resistance measurement of the thermistor and the temperature shown on the display are presented in Fig. 4.

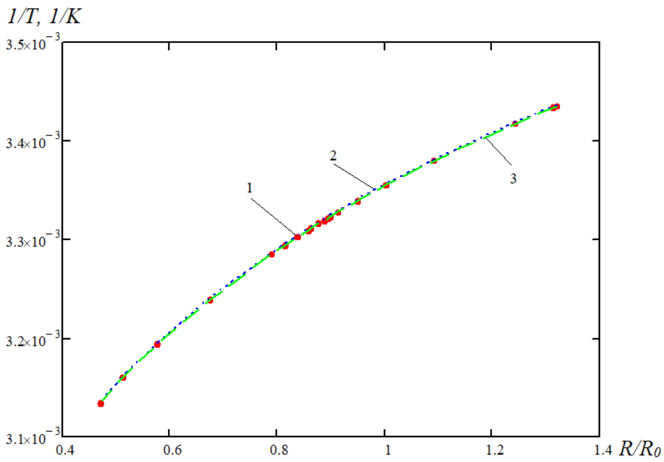

Figure 4: Dependencies between the inverse temperature and the relative resistance of the thermistor: 1 – display reading; 2 – calculation according to expression (8); 3 – calculation according to expression (10)

4. Experimental research

According to research [8, 9] a digital temperature sensor DHT11 together with a thermistor, it was utilized to guarantee the experiments' result dependability, which connects to a digital output of the Arduino. Data from this sensor is also displayed on the display.

Table 2 presents the temperature readings obtained from both a thermistor and a thermal sensor throughout the heating process.

Table 2
The temperature readings of the temperature sensor and the thermistor

T, min	1	2	3	4	5	6	7	8	9	10
t sen, °C	22	25	28	32	35	39	42	46	51	56
t ther, °C	22,29	25,10	28,40	32,30	35,50	39,50	42,50	46,32	51,46	56,32

The dependence, which reflects the dependence of temperature on time in the heating process, obtained as a result of using a thermistor has the form:

$$T = 51{,}38 * e^{0{,}054*t} - 31{,}86, \quad (11)$$

At the same time, a similar dependence obtained with the help of a thermal sensor has the form:

$$T = 45{,}092 * e^{0{,}059*t} - 25{,}664, \quad (12)$$

Dependencies of temperature on time during the heating procedure, acquired with the aid of a temperature sensor, a thermistor and calculation based on expressions (11) and (12), are presented in Fig.5.

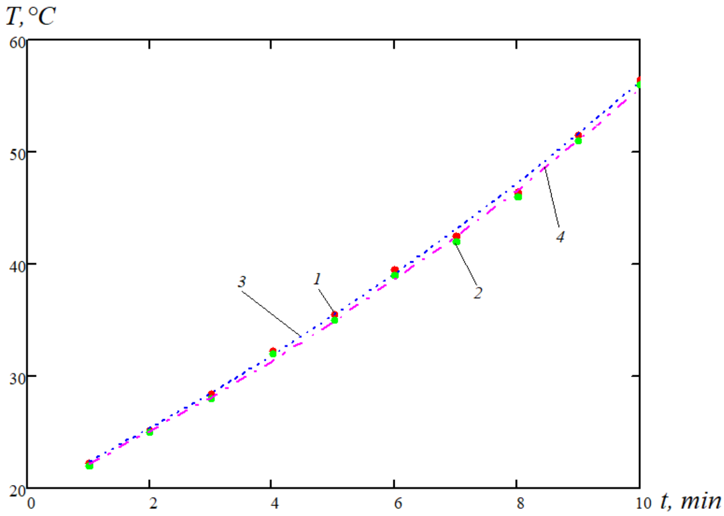

Figure 5: Dependence of temperature on time in the heating process: 1 – obtained with the help of the thermosensor; 2 – obtained with the help of the thermistor; 3 – calculated according to expression (11); 4 – calculated according to expression (12)

As is visible from tab. 2 and fig. 5, the disparity between the figures of the temperature determined by the thermistor and the temperature sensor does not exceed 0.5 degrees. Concurrently, all temperature values attained using the thermistor surpass the respective values obtained using the temperature sensor.

The climate control system comprises several Arduino nodes. Every node may function as a sensor node, actuator node, or a central control node. The sensor nodes collect environmental data and transmit it to the central control node using radio communication modules. The central control node processes this data and sends commands to actuator nodes to adjust the climate conditions.

The system consists of two Arduino UNO boards, each running its own program to simulate wireless communication via visible light. The connection schemes of the parts of automated temperature control system were developed.

The first Arduino (Automated Room Temperature Monitoring) reads temperature data from an NTC thermistor using the Steinhart–Hart equation to convert analog voltage into degrees Celsius. Automated Room Temperature Monitoring presented in Figure 6.

The transmitting system utilizes an Arduino UNO paired with an NTC thermistor to measure the ambient temperature. When the temperature is assessed against a predefined threshold, such as 25 °C, the Arduino determines whether the value falls below the setpoint. If it does, a digital signal is generated, activating a radiotransmitter that functions as a simulated wireless communication device. On the receiving end, another Arduino-based setup (depicted as Automated Actuation in Figure 7) incorporates a radiosensor to detect incoming signals. The system continuously monitors the signal strength and compares it to a set threshold. When a sufficiently strong signal is detected —indicating that the temperature at the transmitter is below the threshold—the Arduino engages a relay module, which then powers a connected heater. Conversely, if no signal is detected, the relay remains inactive, leaving the heater turned off. This arrangement demonstrates a fundamental wireless actuation mechanism driven by specific environmental conditions.

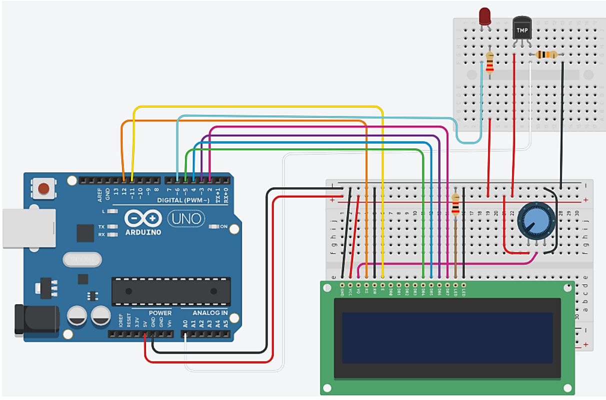

Figure 6: Scheme of Automated Room Temperature Monitoring

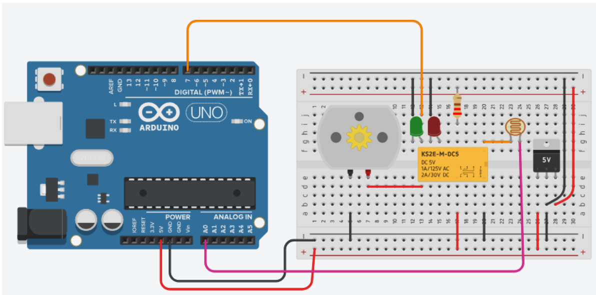

Figure 7: Scheme of Automated Actuation

Programs for each part have been developed. The codes of which are written in C++. Transmits the data using the VirtualWire library and MX-FS-03V transmitter. Code shown on Fig. 8.

```
#define THERMISTOR_PIN A0
#define LED_PIN 8

const float SERIES_RESISTOR = 10000.0;
const float NOMINAL_RESISTANCE = 10000.0;
const float NOMINAL_TEMPERATURE = 25.0;
const float B_COEFFICIENT = 3435.0;
const int ADC_MAX = 1023;

float threshold = 25.0;
unsigned long lastSend = 0;

float readThermistor() {
  int adc = analogRead(THERMISTOR_PIN);
  float voltage = adc * 5.0 / ADC_MAX;
  float resistance = SERIES_RESISTOR * (5.0 / voltage - 1.0);
  float steinhart;
  steinhart = resistance / NOMINAL_RESISTANCE;
  steinhart = log(steinhart);
  steinhart /= B_COEFFICIENT;
  steinhart += 1.0 / (NOMINAL_TEMPERATURE + 273.15);
  steinhart = 1.0 / steinhart;
  steinhart -= 273.15;
  return steinhart;
}

void setup() {
  Serial.begin(9600);
  pinMode(LED_PIN, OUTPUT);
}

void loop() {
  if (millis() - lastSend > 2000) {
    lastSend = millis();

    float temp = readThermistor();
    Serial.print("Temp: ");
    Serial.println(temp);

    if (temp < threshold) {
      digitalWrite(LED_PIN, HIGH);
    } else {
      digitalWrite(LED_PIN, LOW);
    }
  }
}
```

Figure 8: Code reads the temperature and humidity from the DHT11 sensor and transmits the data wirelessly using the MX-FS-03V transmitter.

Transmitter reads temperature and humidity from the DHT11 sensor. The central control node receives data from sensor nodes, processes it, and determines the necessary adjustments. For example, if the temperature is below a set point, it can turn on a heater.

Receives the data using the VirtualWire library and MX-05V receiver. Parses the received data and controls a relay based on the temperature. Code shown on Fig. 9.

Both codes include serial output for debugging and run in continuous loops with short delays to allow real-time monitoring and control.

This code assumes the relay turns on if the temperature exceeds 25°C.

```
#include <Wire.h>
#include <Servo.h>

#define RADIO_PIN A0
#define RELAY_PIN 7

int lightThreshold = 600;

void setup() {
  Serial.begin(9600);
  pinMode(RELAY_PIN, OUTPUT);
  digitalWrite(RELAY_PIN, LOW);
}

void loop() {
  int lightValue = analogRead(RADIO_PIN);
  Serial.print("Light: ");
  Serial.println(lightValue);

  if (lightValue > lightThreshold) {
    digitalWrite(RELAY_PIN, HIGH);
    Serial.println("HEATER ON");
  } else {
    digitalWrite(RELAY_PIN, LOW);
    Serial.println("HEATER OFF");
  }
  delay(500);
}
```

Figure 9: Code receives the data from the transmitter and controls a relay based on the received data

Actuator nodes receive commands from the central control node and adjust devices like heaters, fans, or air conditioners accordingly.

5. Estimation of Potential Energy Savings

To evaluate the effectiveness of the developed automated climate control system, a simplified estimation of daily energy savings was performed for a typical household heating appliance. Let us assume the following conditions: Electric convector heater with power consumption: 2000 W (2 kW).

Manual mode operation: 5 hours per day (e.g., turned on in the morning and evening regardless of room temperature)

Automated mode: using temperature control, the heater is switched off once the target temperature is reached and turned on only when necessary, reducing average runtime to 3.5 hours per day. Electricity cost: $0.10\, EUR / kW\, h$

Energy consumed in manual mode per day:

$$E_{manual} = 2\, kW \times 5\, h = 10\, kWh$$

Energy consumed in automated mode per day:

$$E_{autol} = 2\, kW \times 3.5\, h = 7\, kWh$$

Daily savings:

$$\Delta E = 10 - 7 = 3\, kW\, h$$

Financial savings per month (30 days):

$$90\, kW\, h \times 0.10\, EUR / kW\, h = 9\, EUR / mont\, h$$

Conclusions

Modern home automation systems aim to enhance comfort, energy efficiency, and convenience. One significant aspect of home automation is climate control, which includes the regulation of temperature, humidity, and air quality within a room. By integrating low-cost Arduino platforms with 433 MHz radio modules, the developed system provides a practical and scalable solution for regulating heating and cooling appliances in residential environments.

The measured resistance of the MF52-103 3435 thermistor under varying temperatures aligns well with the dependence described by the Steinhart-Hart equation. This makes the thermistor a suitable choice for determining air temperature in automated indoor microclimate control systems. Experimental testing of the temperature sensing mechanism based on the MF52-103 3435 thermistor demonstrated high precision, with deviations from a standard digital sensor remaining within 0.5 °C. Additionally, analytical modeling through the Steinhart-Hart equation and its simplified forms provided dependable real-time temperature estimates, ets.

The control algorithm successfully maintains user-defined temperature thresholds by enabling or disabling appliances via relay switches, ensuring energy efficiency and minimizing human intervention. A simplified energy consumption model suggests that the system can reduce electricity usage by up to 30%, depending on user behavior and appliance type.

The proposed solution demonstrates a high potential for integration into modular smart home systems, especially in scenarios where cost, flexibility, and simplicity are key design factors. Future enhancements may include the addition of humidity sensors, remote smartphone integration, or transitioning to Wi-Fi or MQTT-based communication protocols for broader IoT compatibility.

Acknowledgements

The work was completed thanks to many years of cooperation document Kyiv National University of Technologies and Design and Khmelnytskyi national university.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] Javed, A. (2016). Building Arduino projects for the Internet of Things: Experiments with real-world applications. Apress Media, LLC. DOI: 10.1007/978-1-4842-1940-9
[2] Elsenpeter, R., & Velte, T. (2003). Build your own smart home. McGraw-Hill Osborne Media.
[3] Margolis, M. (2011). Arduino cookbook. O’Reilly Media.
[4] Boloor, A. J. (2015). Arduino by example. Packt Publishing Ltd.
[5] Blum, J. (2013). Exploring Arduino: Tools and techniques for engineering wizardry. Wiley.
[6] Sommer, U. (2010). Arduino: Mikrocontroller-Programmierung mit Arduino/Freeduino (in German). Franzis.
[7] Hulzebosch, J. (2009). USB in der Elektronik: Die USB-Schnittstelle für praktische Anwendungen am PC einsetzen (in German). Franzis.
[8] Liubchyk, V., Kylimnik, A., & Horyashchenko, S. (2013, May 19–21). Application of the multi-frequency phase method of ranging to many objects for construction of ground penetrating radar. In Proceedings of the International Radar Symposium (IRS-2013) (pp. 835–840). Dresden, Germany.
[9] Horiashchenko, K., Horiaschenko, S., Golinka, I., & Drapak, G. (2022). Development of a synchronous motor with frequency control system for light industry equipment. In Proceedings of the 2022 IEEE 4th International Conference on Modern Electrical and Energy Systems (MEES 2022). https://doi.org/10.1109/MEES58014.2022.10005685