Sudan University of Science and Technology

College of Graduate Studies

Design and Implementation of an Internet-based Remote Controlled Temperature Controller

تصميم وتنفيذ متحكم للتحكم عن بُعد في درجة الحرارة من خلال الإنترنت

A Thesis Submitted In Partial fulfillment for the Requirements of the Degree of M.Sc. in Mechatronics Engineering

Prepared by:

SALMA HISHAM BABIKER KHALAFALLA

Supervised by:

DR.FATH ELRAHMAN ISMAEL KHALIFA

April 2018

الإستهلال

بِسۡمِ ٱللَّهِ ٱلرَّحۡمَٰنِ ٱلرَّحِيمِ

قال تعالى: "وابتغِ فيما آتاكَ اللهُ الدارَ الآخرة ولا تنسَ نصيبكمن الدنيا وأَحْسِن كما أَحْسَنَ اللهُ إليكَ ولا تبغِ الفسادَ في الأرضِ ان الله لا يحب المفسدين"

صدق الله العظيم

سورة القصص الآية (77)

DEDICATION

To those who were very caring, helping and encouraging me for advancement and success.

To my mother and my father who supported me and believed in my capabilities.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my supervisor Dr.FathElrahman Ismael for keen supervision with his known open heart and mind and flooding generosity, I am also grateful for his encouragement and guidance

I also, special thanks to my husband and friend ***Ahmed*** *Muhammad Nimir for his strong support and encouragement.*

Lastly, I offer my regards and blessings to all of those who supported me in any aspect during the completion of the thesis.

المستخلص

الإنترنت هو شبكة اتصالات تستخدم لتوصيل جهاز الحاسوب بأي حاسوب آخر في أي مكان في العالم. في هذه الأطروحة قد تم بناء نظام تحكم عن بُعد للتحكم في درجة حرارة غرفة عن طريق الإنترنت بإستخدام تقنية المخدم والمستخدم.الغرض الأساسي من هذا النظام هو التحكم والمراقبة للعوامل البيئية (درجة الحرارة كمثال) عن بُعد.يتكون النظام من جزأين, الجزء الأول هو دائرة متحكم الأردوينووالذي يتم التحكم فيه بواسطة الإنترنت . اما الجزء الثاني فهو عبارة عن نظام البرمجيات الذي يحتوي على واجهات مستخدم رسومية تم ربطها مع دائرة متحكم الأردوينو بالانترنت عبر جهاز الخادم.جهاز الخادم يستقبل الاوامر من حاسب المستخدم لتمكين المستخدم من المراقبة والتحكم في درجة الحرارة بسهولة. إستهلاك الطاقة الإجمالي لدائرة متحكم الأردوينو ليس عالياً ويمكن إستبدال مصدر الطاقة المستخدم في هذه الأطروحة بمصدر جهد محمول او قابل لإعادة الشحن. يعتمد زمن إستجابة النظام بشكل كبير على نوع الإتصال بالإنترنت وسرعته.

ABSTRACT

The Internet is a telecommunications network for connecting a computer to any other computer anywhere in the world. It's approximately available in anywhere, so its accessibility is high and it's suitable to link industrial application over the grid. In this thesis a remote control system has been implemented to control a room temperature remotely through Internet by using client/server technique. The main purpose of the system is to remote control and monitor environmental parameter for example temperature degree. The system composed of two main parts, the first part is the Arduino control circuit which can be controlled through Internet. The second part is the software system which contains graphical user interfaces, which was linked with the Arduino control circuit which connected to Internet through the server computer. The server computer receives commands from the user computer to enable the user to control and monitor the room temperature. The overall power consumption is not high and can be spared by using rechargeable or portable voltage source, and the response time is considerably by the connection type and its speed.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AC	Alternating Current
ADC	Analog-to-Digital Converter
API	Application Programming Interface
ARM	Acorn RISC Machines
AVR	Advanced Virtual RISC
COM	Communication
DC	Direct Current
GLCD	Graphical Liquid Crystal Display
GND	Ground
GUI	Graphical User Interface
ICSP	In Circuit Serial Programming
IDE	Integrated Development Environment
IP	Internet protocol
LCD	Liquid Crystal Display
LED	Light Emitting Diode
MCU	MicroController Unit
PCB	Printed Circuit Board
PIC	peripheral interface controller
PID	Proportional Integral Derivative
PWM	Pulse Width Modulation
RISC	Reduced Instruction Set Computer
UART	Universal Asynchronous Receiver Transmitter
USB	Universal Serial Board
VB	Visual Basic
VS	Visual Studio
WLAN	Wireless Local Area Network

Chapter One

INTRODUCTION

Chapter One
Introduction

1.1 Preface

Temperature Control System is a type of control system that automatically controls the temperature of an object or an area. It can be implemented in different types of systems like Air Conditioning, Water Heaters, Refrigerators, etc. To accurately control temperature without extensive operator involvement, a temperature control system relies upon a controller which accepts temperature from a temperature sensor as input. It compares the actual temperature to the desired temperature or set point and provides an output to control element depending upon the system to be controlled. These days there is high demand in monitoring environmental parameters such as temperature. It's the most often-measured environmental quantity this might be expected since most physical, electronic, chemical, mechanical and biological systems are affected by temperature [1].

1.2 Problem Statement

A distant manufacture space that its' temperature has to be measured and controlled. And its user could be a rover that's far away thousands kilometers, the space got to be manipulated instantaneously.

1.3 Proposed Solution

The Client/server Technique over Internet is a viable, A remote access over Client/server Technique to Arduino control circuit is to be considered as a good solution for this problem, the control circuit connects to a computer server via Internet to provide the user with an interface to their device, it allows user full control over the target device from any computer with Internet access.

1.4 Aim and Objectives

The aim is to measure temperature then to send data to the user for controlling temperature according to the readings. Then transfer data into the control circuit in a secure link. All these actions are in both directions.

The main objective of this thesis is to design a remote control system through Internet. To achieve these objectives:

1. Design a proposed control circuit for the system.

2. Simulation of the proposed control circuit will be done using PROTEUS and visual basic.

3. Practical implement of the control circuit design in hardware.

4. Develop a user-friendly GUI to manipulate the temperature.

5. Linking the GUI and the control circuit over a secure link using Internet.

1.5 Methodology

To accomplish this project the work divided into three phases:

I. **Phase one:** the windows application over Internet had been developed by Visual Studio software. Serial interface had been tested by creating virtual ports. The user can send the commands for controlling the room temperature by using windows application through the Internet to the computer server which receive the command. Both applications secured with a password , the client application encrypts the commands before transfer it through the Internet , in the other side the server application receives the command and decrypt it.

II. **Phase two:** Implementation of the control circuit using Arduino and LM35 Celsius sensor (10 m volts/Celsius).

III. **Phase three:** communication between windows application and control circuit through serial port.

1.6 Thesis organization

This thesis composed of five chapters, their outlines are as follows: chapter one introduce the problem and the proposed solution. Chapter two gives an overview of the components used to implement the system and provides previous work. Chapter three describes the system design and the process in details. Chapter four represents results and its discussion and finally Chapter five concludes the thesis.

Chapter Two

LITERATURE REVIEW

Chapter Two

Literature Review

2.1Background

In manual temperature control system, an operator periodically reads the process temperature and adjusts the heat or cooling input up or down in such a direction as to drive the temperature to its desired value. Manual control may be used in non-critical applications where small changes in the manipulated variable cause the processes to change slowly and by a small amount. When several processes are involved, or temperature changes are too rapid for operator correction then accuracy is a must, the use of an automatic control system is indicated.

Typically these days the ability to simplify processes and improve efficiencies means that most temperature control systems are automatic, there are three general kinds of temperature controllers that are used to monitor temperature during manufacturing processes: on-off, proportional, and PID controls. An on/off temperature control is the least expensive of the control types, and also the most simple in terms of how it works. The control is either on or off, if the temperature drops below a certain point, the control signals to the machine to turn raise the temperature. Secondly proportional controls, they are designed to respond to temperature change before it slips out of the desired range. Essentially, proportional controls increase or decrease the power supply as the temperature reaches its upper or lower limit, or set point, which slows or speeds the heater and helps stabilize the temperature.

2.2Arduino

An Arduino is an open-source microcontroller (a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals) development board based on easy-to-use hardware and software. Over a long time Arduino has been the brain of thousands of projects, from regular objects to complex logical rebellious. It can be used to read sensors and control instruments [2].

2.2.1 Arduino Uno

The most common version of Arduino is the Arduino Uno. "Uno" means one in Italian.Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins, 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started [3].

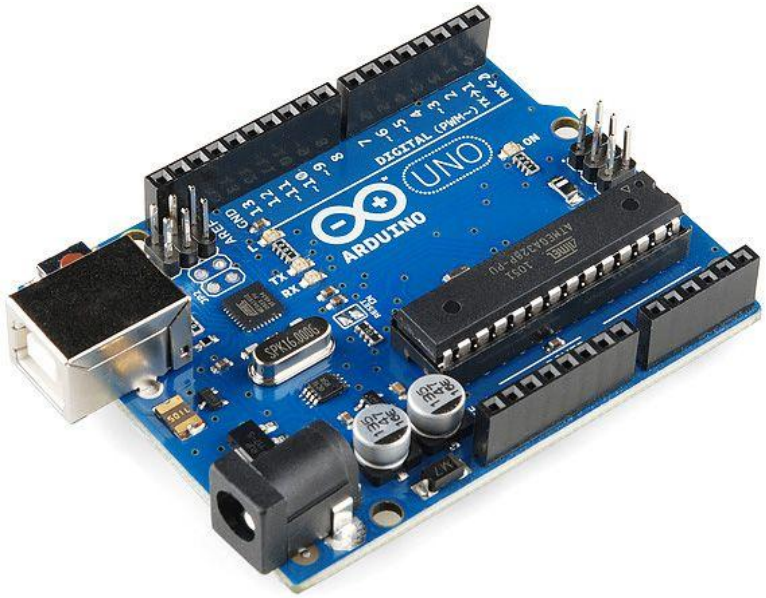


Figure 2-1: Arduino UNO

2.2.2 Arduino Mega

The Arduino Mega is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button [4].

2.2.3 Pin Description

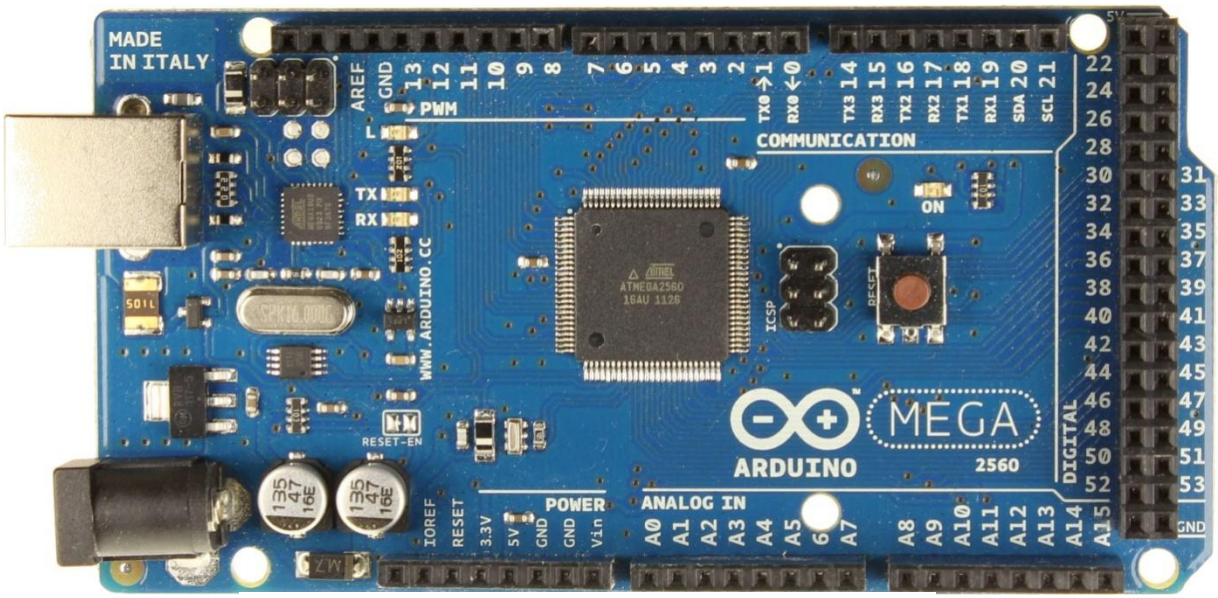


Figure 2-2: Arduino Mega pinout

I. Power Pins:

- **3.3 V**: Supply 3.3 output volt.
- **5V**: Supply 5 output volt.
- **GND**: There are several ground pins on the Arduino, any of which can be used to ground your circuit.

- **Vin**: This pin also can be used to power the Arduino board from an external power source.

II. **Analog Pins:** The Arduino Mega board has sixteen analog input pins A0 through A15. These pins can read the signal from an analog sensor like the humidity sensor or temperature sensor and convert it into a digital value that can be read by the microprocessor.

III. **Digital Pins:** These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc.

IV. **Communication Pins:** Used for communication between the Arduino board and a computer or other devices.

V. **PWM Pins:** Pulse Width Modulation pins, it is a technique for getting analog results with digital means.

2.3 Serial Communication

The serial communication process consists of transmitting or receiving one bit at a time, sequentially on a computer bus or communications channel, the opposite of parallel communications where several bits are sent as a whole.

The asynchronous serial protocol incorporates a number of built-in rules and mechanisms that help ensure robust and error-free data exchange. They are four mechanisms: data bits, synchronization bits, parity bits, and baud rate [5].

The baud rate specifies how fast data is sent over a serial line. It's usually expressed in units of bits-per-second (bps).In the serial port context, "9600 baud" means that the serial port is capable of transferring a maximum of 9600 bits per second.

Synchronization bits are two or three bits transferred with each frame. They are the start bit and the stop bit or bits to define the beginning and the end of the frame. Parity bits used for checking errors.

2.4 LM35 Temperature Sensor

LM35 temperature sensor series are exactitude integrated-circuit temperature devices with associate output voltage linearly-proportional to the Centigrade temperature. It provides associate output voltage of 10.0mV for every degree Centigrade of temperature from a reference voltage, the output of this device will be fed to analog-to-digital convertor. To induce the temperature value accurately, output voltage should be increased with one hundred. as an example if you browse 0.20V that may be 20 degrees Centigrade, as showed in Figure 2-3 the pin layout, pin 1 for supply voltage (4-20v), pin 2 is the output and pin 3 is the ground.

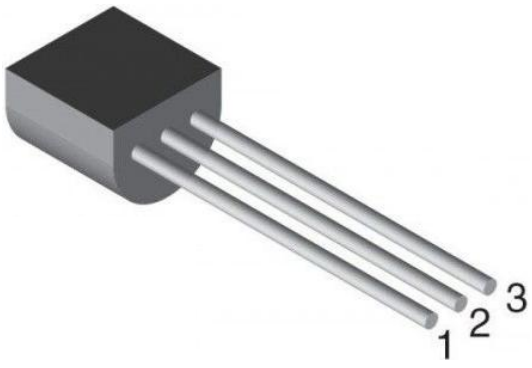


Figure 2-3: LM35 Temperature Sensor

2.5 Liquid Cristal Display (LCD)

LCD is a type of screen, it is associate electronic display module and realize a wide vary of applications. LCD has many types, one of them is LCD 16x2 named so because it has 16 Columns and 2 Rows with total of 28 characters. The pins layout is as follow as illustrated in Figure 2-4.

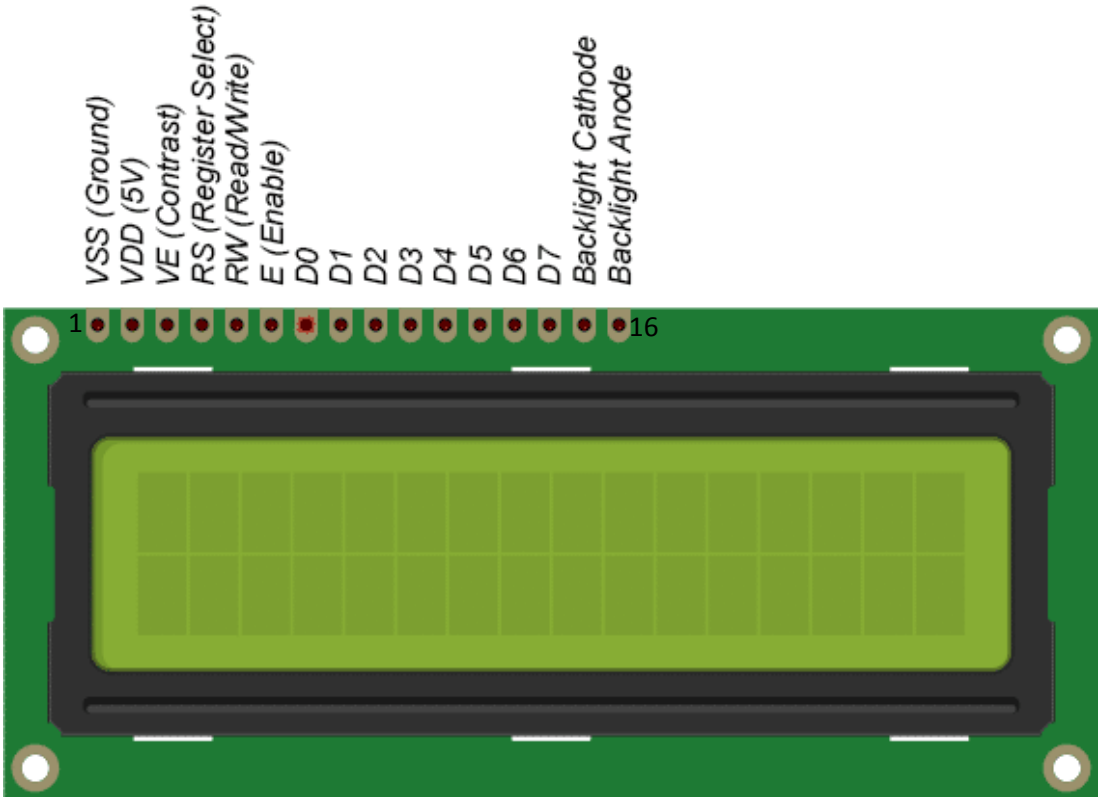


Figure 2-4: LCD 16x2

- pin 1: Ground (0V)
- pin 2: Supply voltage; 5V (4.7V – 5.3V)
- pin 3:Contrast adjustment; through a variable resistor
- pin 4:Selects command register when low; and data register when high

- pin 5:Low to write to the register; High to read from the register
- pin 6: Sends data to data pins when a high to low pulse is given
- pin7 to pin 14:8-bit data pins
- pin 15:Backlight Supply voltage (5V)
- pin 16:Backlight Ground (0V)

2.6 Arduino software

A program for Arduino is written in any programming language with compilers that turn out binary machine language for the target processor. ATmel provides a development environment for their microcontrollers, AVR Studio and therefore the newer Atmel Studio.

The Arduino project provides the Arduino Integrated Development Environment (IDE), which could be a cross-platform application written within the programming language Java. It originated from the IDE for the languages process and wiring. It includes a code editor with options like text cutting and pasting or copying, searching and exchanging text, automatic indenting, brace matching, and syntax highlight, and provides straightforward one-click mechanisms to compile and transfer programs to an Arduino board. It additionally contains a message space, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus as showed in Figure 2-5.

A program written with the IDE for Arduino is named a sketch. Sketches are saved on the development computer as text files with the file

extension .ino. Arduino software package (IDE) saves sketches with the extension .pde. The Arduino IDE supports the languages C and C++ exploitation special rules of code structuring. The Arduino IDE provides a software package library from the wiring project, which provides several common input and output procedures. User-written code solely needs two basic functions, for beginning the sketch the setup() function and the main program loop using loop() function. The Arduino IDE employs the program avrdude to convert the viable code into a text file in hexadecimal coding that's loaded into the Arduino board by a loader program within the board's firmware. The open-source nature of the Arduino project has expedited the publication of many free software package libraries that alternative developers use to enhance their projects [5].

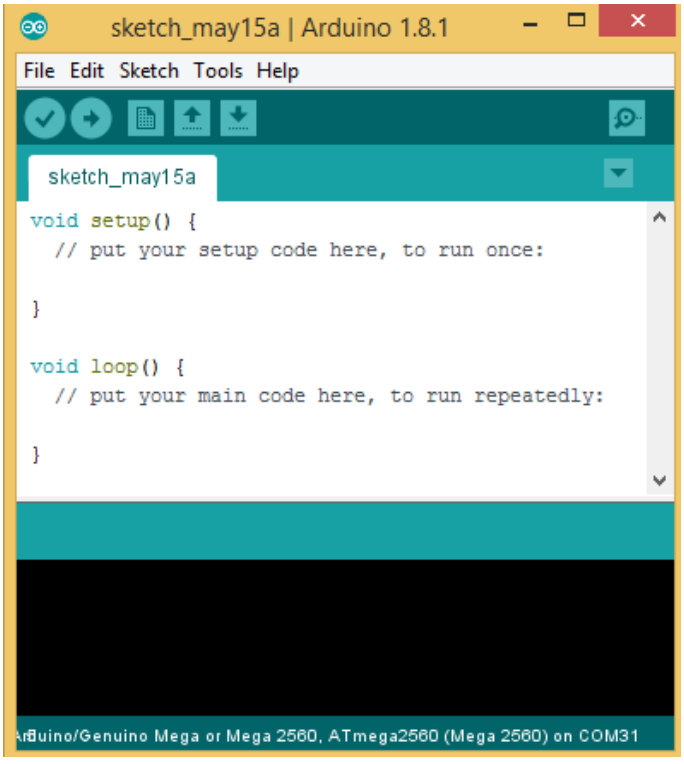


Figure 2-5: Arduino software

2.7 Proteus software

Proteus is a simulation and design software tool developed by way of Labcenter Electronics for electrical and electronic circuit design. It includes a schematic capture, simulation and PCB (Printed Circuit Board) Layout modules. It also combines animated components and microprocessor models to co-simulate the complete microcontroller based design. It's a perfect tool to test design before constructing a physical hardware prototype. Proteus is a real time simulation, also it saves time and money. Figure 2-6 shows Proteus software.

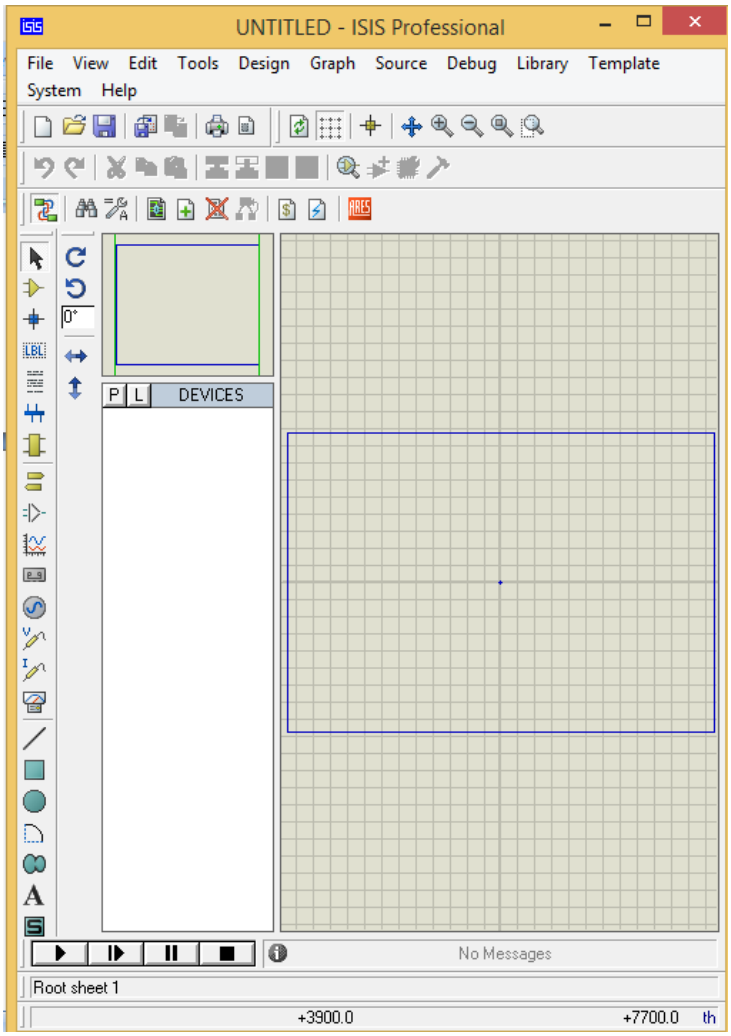


Figure 2-6: Proteus software

2.8 Visual Studio

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It uses Microsoft software development platforms (windows presentation foundation, windows forms, Microsoft Silverlight, windows API and windows store). Also it can deliver both managed code and native code. It's used to create and build windows applications, mobile applications, web applications, web services or even web sites.

VS includes suite of project types that can be chosen from. When creating a new project, VS will automatically generate skeleton code that can compile and run instantly. Each project type has project items that it can be added, and project items include skeleton code. Many parts of the VS environment, including colors, editor options, and layout can be customized.

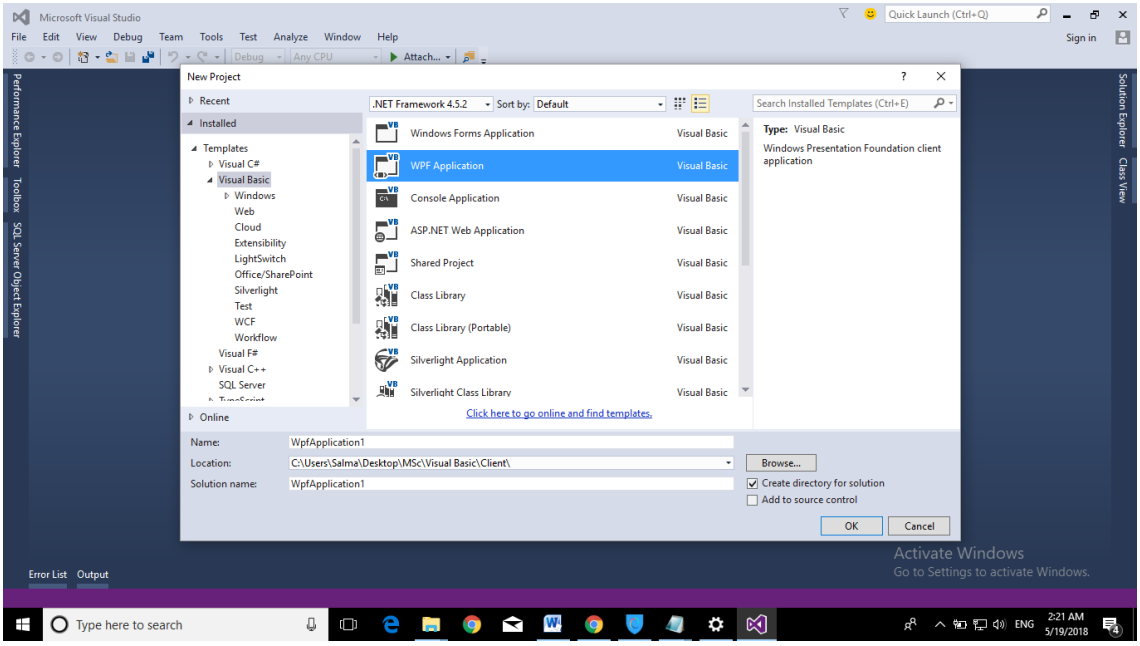


Figure 2-7: Visual Studio 2015 software

3.8.1 Visual Basic Language

Visual Basic .NET, the next era of the Visual Basic language, it is a fast and easy way to create .NET-based applications, including XML Web services and Web applications. Visual Basic .NET has many new and improved features to make it effective object-oriented programming language, including inheritance, interfaces, and overloading.

VB.NET is a multi-paradigm programming language. Multi-paradigm is a way to classify programming languages based on their features.

2.8.2 Winsock

Winsock is a programming interface and the supporting program that handles input/output requests for Internet applications in a Windows operating system. It's called Winsock because it's an adaptation for Windows of the Berkeley UNIX sockets interface. Socket is a particular convention for connecting with and exchanging data between two program processes within the same computer or across a network.

2.9 Pervious Work

In [6] an automatic cooling system according to the room temperature had been developed. The system is designed with Arduino (microcontroller). The main objective of their work is to display the temperature and when it goes beyond certain limit then control it to bring it

back into desired level and reduce waste of energy. Also to assist people who are disabled and are unable to control the speed of fan. It may also be used for monitoring changes in environment. In near future, it can also be used in different industries and electronic devices. Another objective is to study and build an automatic system using microcontroller and its interfacing with other device.

A scheme of wireless communication system for temperature and humidity monitoring had been designed. The system is divided into two main parts, data acquisition and system control. The data acquisition system consists of ATmega328 MCU based on Arduino platform, temperature & humidity sensor and wireless transceiver module. Control system is through single-chip microcomputer which controls the wireless transceiver module and then displays the received information in real-time. The control part also provides the flexibility of setting the upper and lower limits of temperature and humidity data at any time. The received data is checked for the upper and lower limits, if the received values are not within the original data range, the buzzer will sounded for alarming the people's attention, to make a timely dispose [7].

A remote temperature monitoring system using ARM, Arduino and ZigBee had been presented. The aim is to design and develop a system which fulfills the requirements of an environment monitoring system which is accurate, easily operated, simple in working, uncomplicated construction, cost-effective, comfortable to carry and lightweight. Temperature sensor LM35 is used to sense the environmental temperature. ARM microcontroller is used to convert temperature sensor analogue signal into digital data. The digital data was transmitted

wirelessly using ZigBee and the data values are displayed using ARM graphics liquid crystal display (GLCD. In ARM there are inbuilt 12-bit ADC which are used for Analog to digital conversion of data, and serial data transfer from a remote location. At receiver ZigBee with Arduino microcontroller is used to decode the serial data, which is transmitted by ZigBee of the transmitter and it was displayed using graphics LCD as well as the temperature data values are stored in PC [8].

[9] focused in their paper in 2016 on controlling of temperature using PID (Proportional, Integral, Derivative) controller (PIC 16F873A), they represented control of constant temperature according to the desired value (set point) in a closed loop using PID controller system. For this, they used a microcontroller, a temperature sensor for sensing the temperature of the closed loops. By using the microcontroller they compared the desired value with current value and it is displayed in the LCD. Also to provide the constant temperature, Fan or Heater is turned on or off according with the variations of current temperature in ^{o}C from desired set point. The main objective is maintaining the constant temperature in a particular area using PID controller.

In [10]a design, construction, development and control of an automatic temperature controller system that applies to the smart TudungSaji had been presented. The idea is based on the problems that happen in human life nowadays by improving the traditional TudungSaji. Their work consists of two major parts, which are hardware and software design and development. In the hardware design, this project chassis is developed by using bamboo TudungSaji. It was chosen due to its nature as an insulator and better material compared to plastic. The bulbs will

automatically switch on when environmental temperature in TudungSaji changes up to 60^{o}C. They used a microcontroller as a brain to control the bulbs according to temperature variation. The system measures the temperature of the LM35 sensor detection, where it will control the bulbs according to the setting in the programming. The system indicates the temperature from the ATmega 328p. As the temperature goes beyond than 60^{o}C, the bulbs have automatically turned off.

A lot of research has been done in this area , any approach has some features and limitations, [7] successes in monitoring the temperature with high speed and efficiency but the settings of temperature is made manually by varying a potentiometer, also the system need some security features and it's only suitable for short distance due to wireless connection (not more than 20m indoor and 40m outdoor).

In [6] approach they also succeeded in monitoring temperature but the settings of temperature is made on the software code and control in distance was not under their considerations. In like manner [8] accomplished a full control of a temperature, but in short distance. Furthermore [9] concentrates on full control of the temperature using a PID controller, but also there is no control in distance.

Finally in [10] the approach focused on monitoring the temperature more than control it but there is no remotely temperature control . All of these advantages and disadvantages will be considered in this system.

Chapter Three

SYSTEM DESIGN

Chapter Three

System Design

3.1 System Block Diagram

The block diagram of the system is showed in Figure 3-1 bellow. The user sends the desired temperature in Celsius to the server computer through Internet. The server receives the desired temperature then it sends the temperature to the Arduino using serial communication.

The Arduino measures the actual temperature using temperature sensor LM35, then it compares the actual temperature with the desired temperature. According to that, the Arduino decides to open the heater or the cooler.

After that, the Arduino sends feedback to server contains the desired temperature versus the actual temperature.

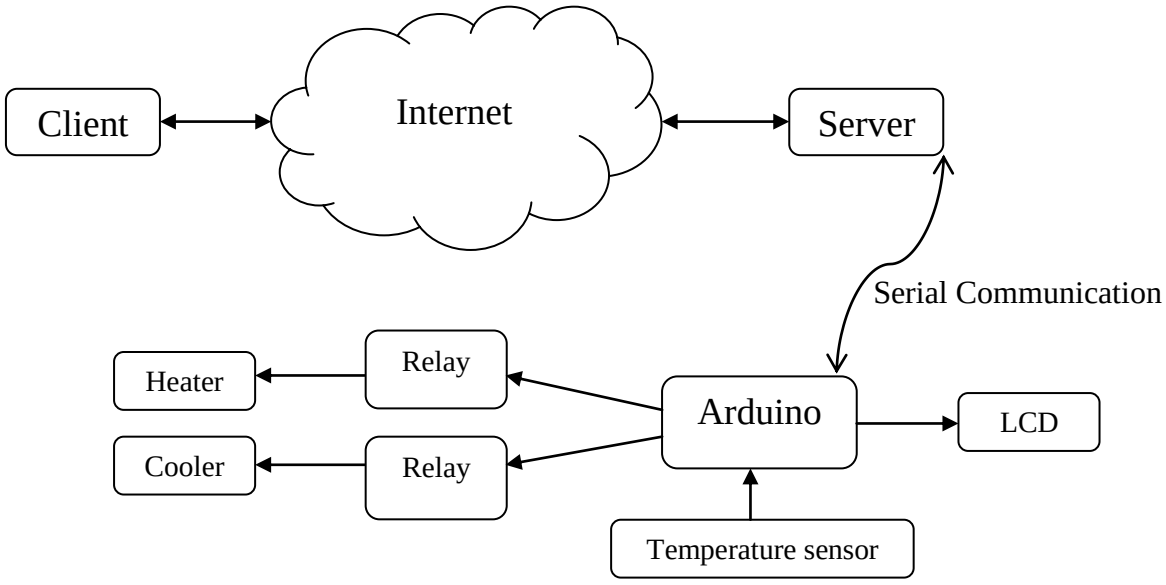


Figure 3-1: System Block Diagram

3.2 System Description

The system works in consonance with an algorithm that waits for the desired temperature to be entered by the system's user, then the Arduino handles the desired temperature and translates it into an action according to the actual temperature. First of all, the connection between the user (client) and the server must be established. Then, user starts to send the desired temperature degree, after that the server sends the received data to the Arduino via serial port. Finally the Arduino generates the controlling signal to operate the heater or the cooler and displays the temperature in an LCD. Figure 3-2 describes the system algorithm.

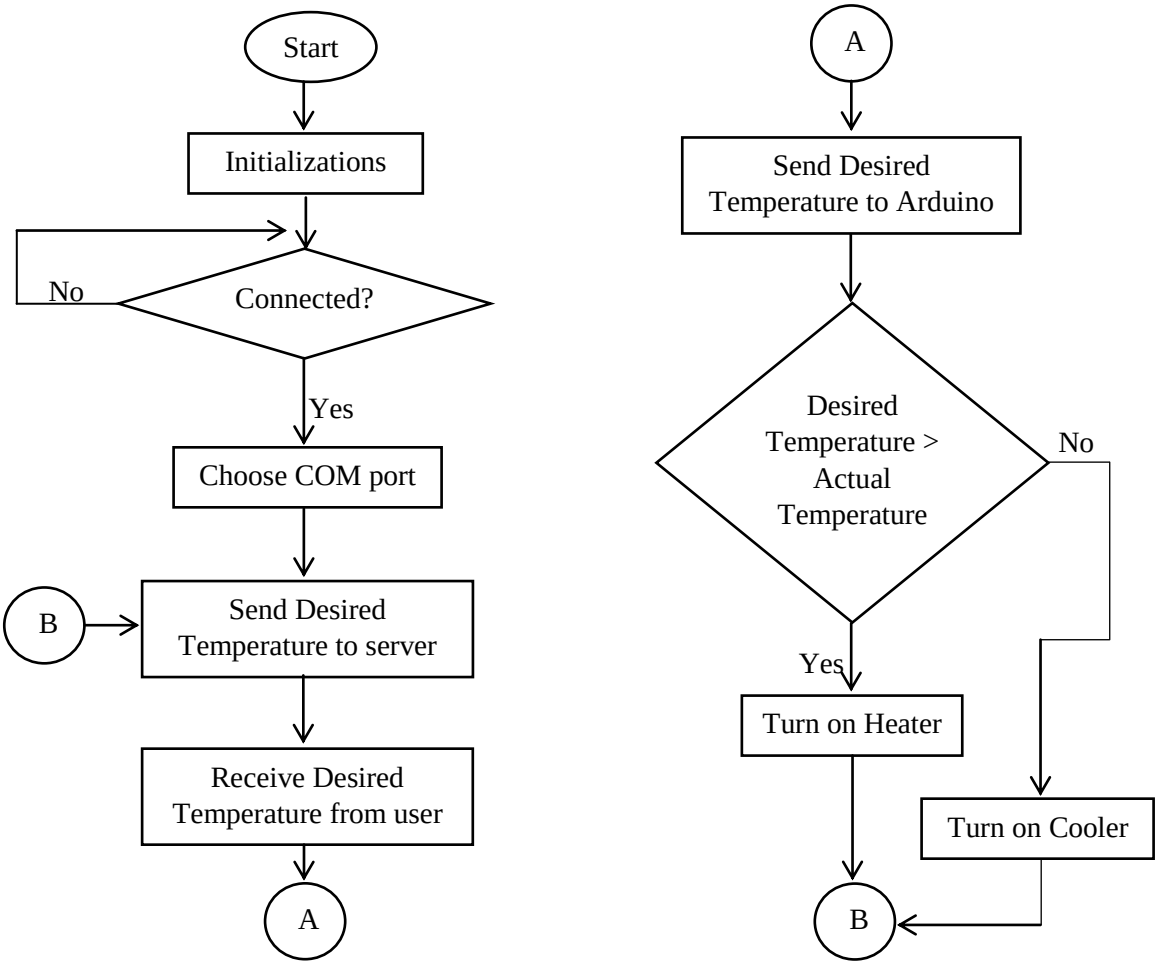


Figure 3-2: System Algorithm Flowchart

3.3 Client/Server model

The client and the server applications had been developed using visual studio 2015. Both client and server are secured with an encrypted username and password, after authentication, the client encrypts the data which entered by the user before transfers it through Internet by using Winsock tool. On the server side, the server receives the data then decrypts it before sends the data serially using COM port to the Arduino.

The application is protected by password only known by the operator as part of the authentication. Figure 3-3 shows the login form, both client and server has the same login form. The operator enters username and password, after that the system encrypts username and password and finally compares them with a saved username and password.

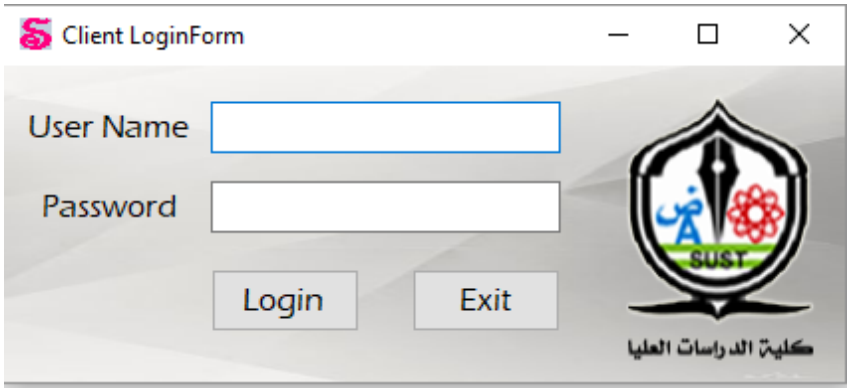


Figure 3-3: Login form

In the Server application, the server computer starts listening to a specific port number automatically as illustrated in figure 3-4, client port number can be changed just by typing the desired port number.

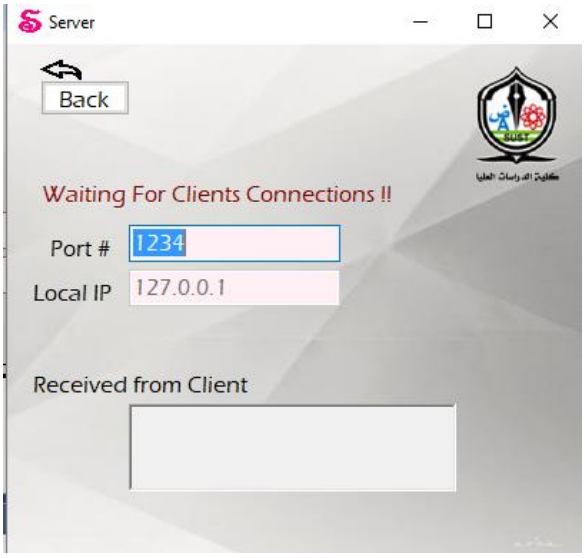


Figure 3-4: Server application

In the client side, IP address and port number of the server computer must be specified to create a connection by pressing connect button as shown in figure 3-5.

After connection successfully, then server sends its available COM ports to be chosen by the client, client computer adds the available COM ports to the combo box items as illustrated in figure 3-6.

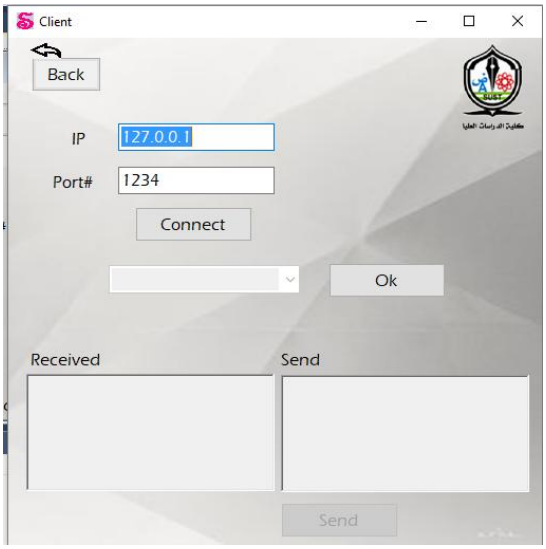


Figure 3-5: Client application

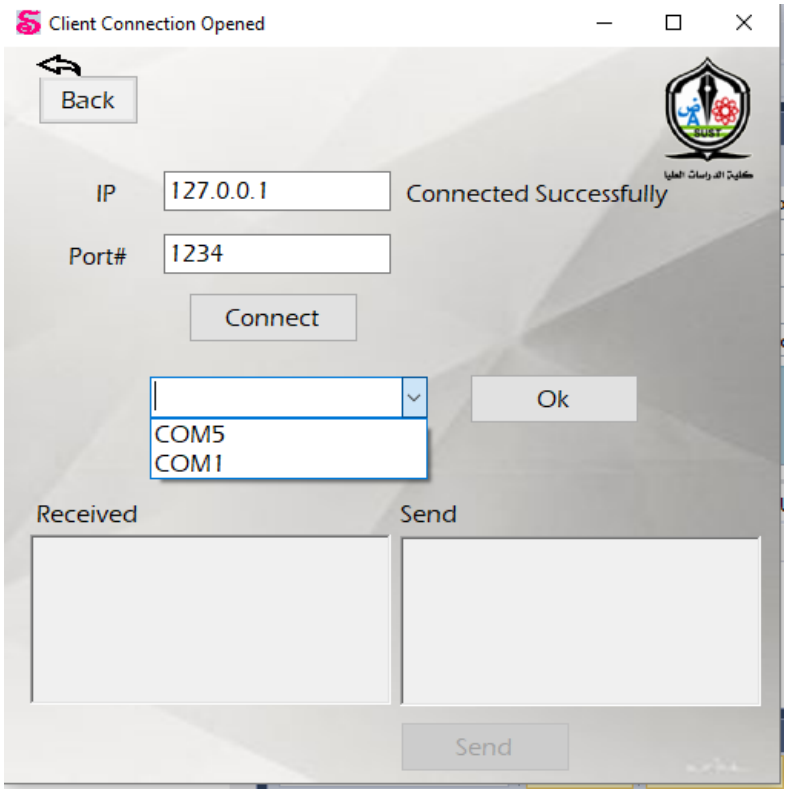


Figure 3-6: Adding available COM ports to the client

Receiving and transmitting text boxes changed from “disabled” to “enabled” after choosing a specific COM port then pressing OK button. After that the user can enter the desired temperature to the control circuit through server computer.

Chapter Four

SIMULATION, IMPLEMENTATION, RESULTS AND DISCUSSION

Chapter Four

Simulation, Implementation, Results and Discussion

4.1 Client/Server Procedures

The implementation has been developed by using visual basic programming language using visual studio 2015 software. The system composed of two GUIs for both client and server, the first one is the authentication GUI and the second one is the main GUI. When system starts, the server will listen to a specific port number and waits for client connection as illustrated in figure 4-1.

To set up the connection between client and server, the IP address of the server computer must be entered by the user as illustrated in figure 4-2.

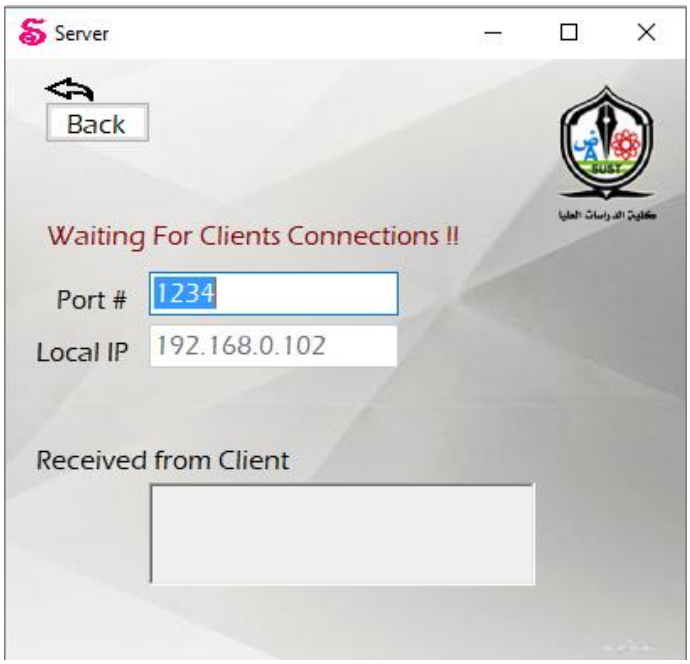


Figure 4-1: Server main GUI

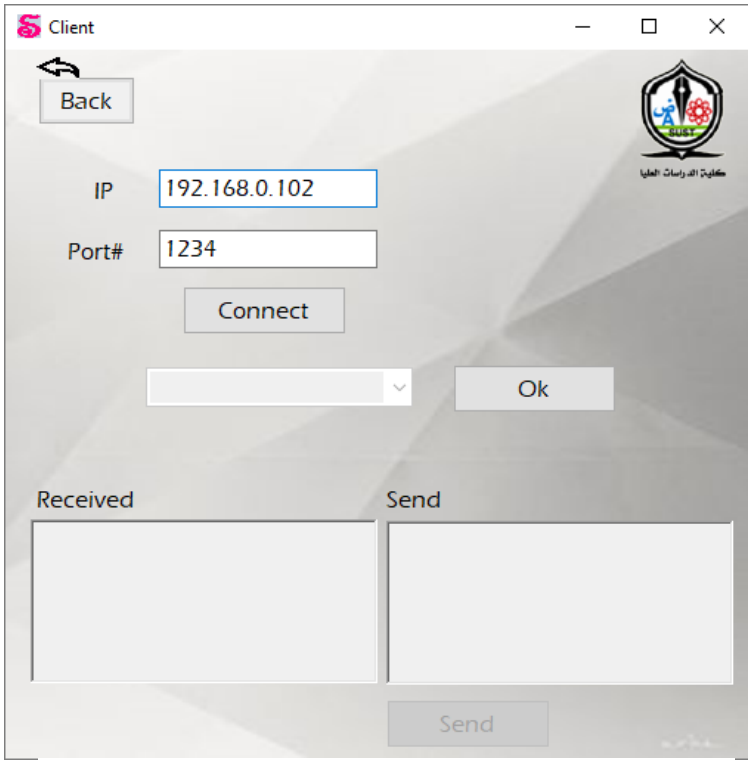


Figure 4-2: Client main GUI

After entering server IP address and pressing "connect" button, the client gives an indication if it's connected successfully. Also the server computer sends its available COM ports to the client computer. After that the client computer receives the available COM ports to the server. The user can control the desired COM port from the available COM ports as illustrated in figure 4-3.

Then the user can control the specific room temperature just by typing the wanted temperature in the text box before pressing "Send" button as shown in figure 4-4.

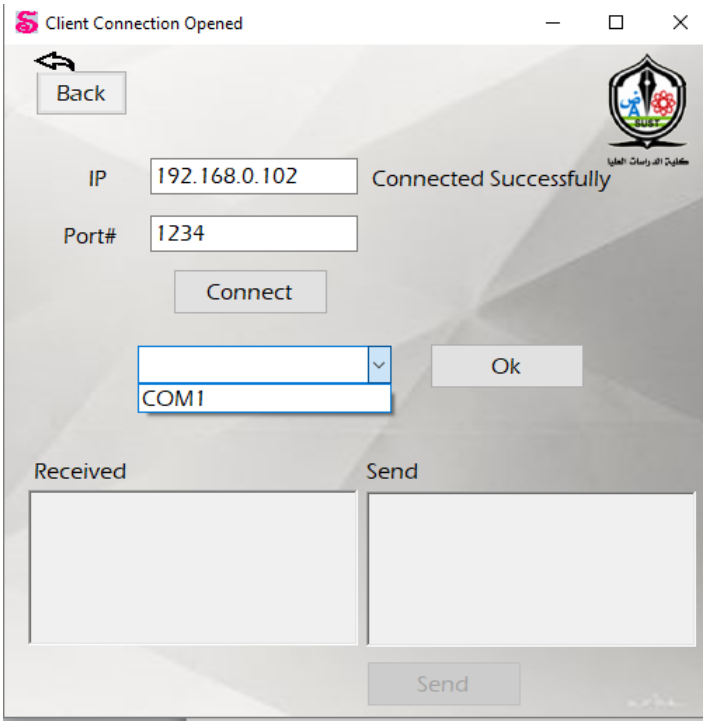


Figure 4-3: Client COM port selection

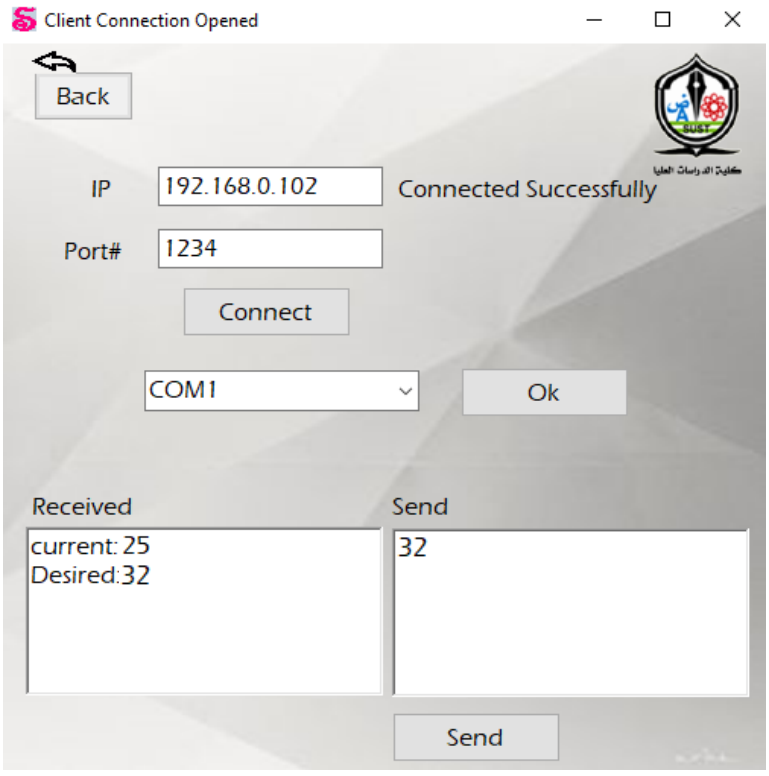


Figure 4-4: Client commands sending

4.2 Arduino control circuit simulation

As shown in Figure 4-4, the user sends the number 32, which is mean the desired temperature is 32°C. Figure 4-5, Figure 4-6 andFigure 4-7 show the simulation on proteus software. When the data is received from serial port through Internet, the arduino translates this data to actions, they are three actions in this system. The first one is heating action, it happened when the desired temperature is greater than the current temperature as illustrated in Figure 4-5 and Figure 4-4. The second action is cooling which is the opposite of heating, Figure 4-6 shows this action. Lastly, the third action is the defalut, by turining off both of heater and cooler, and it's happening when the desired temperature is equal to the current temperature as shown in Figure 4-7.

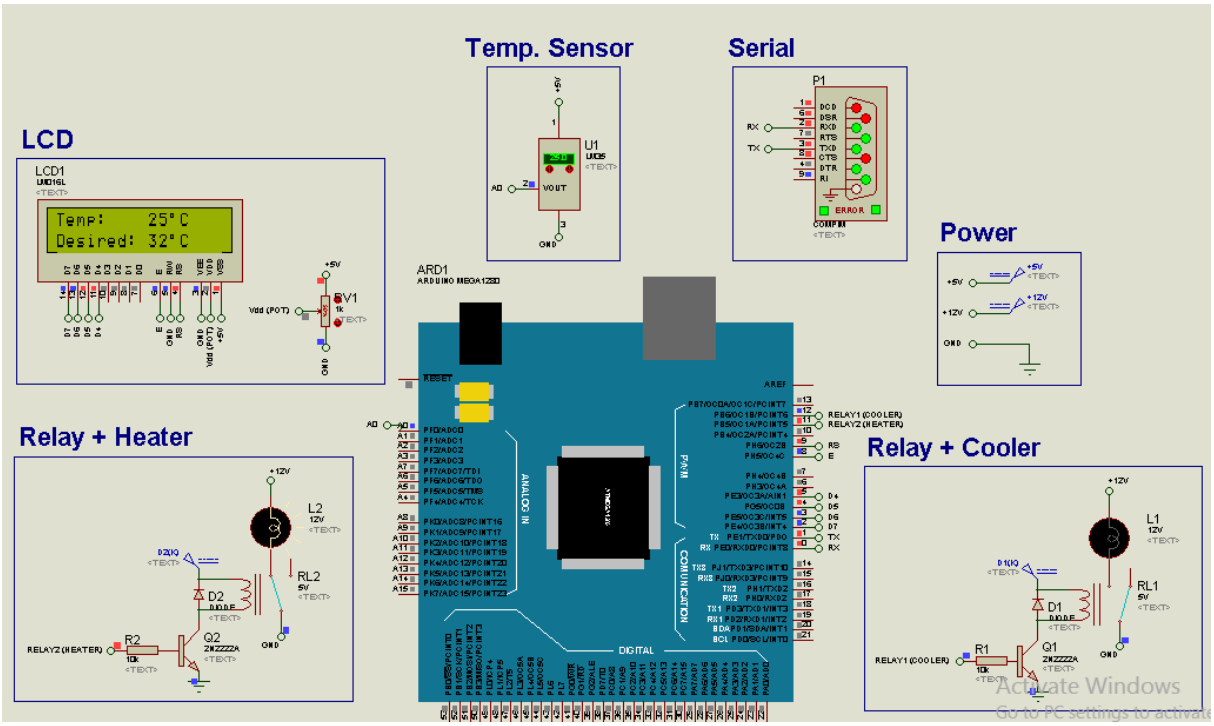


Figure 4-5: Heating operation

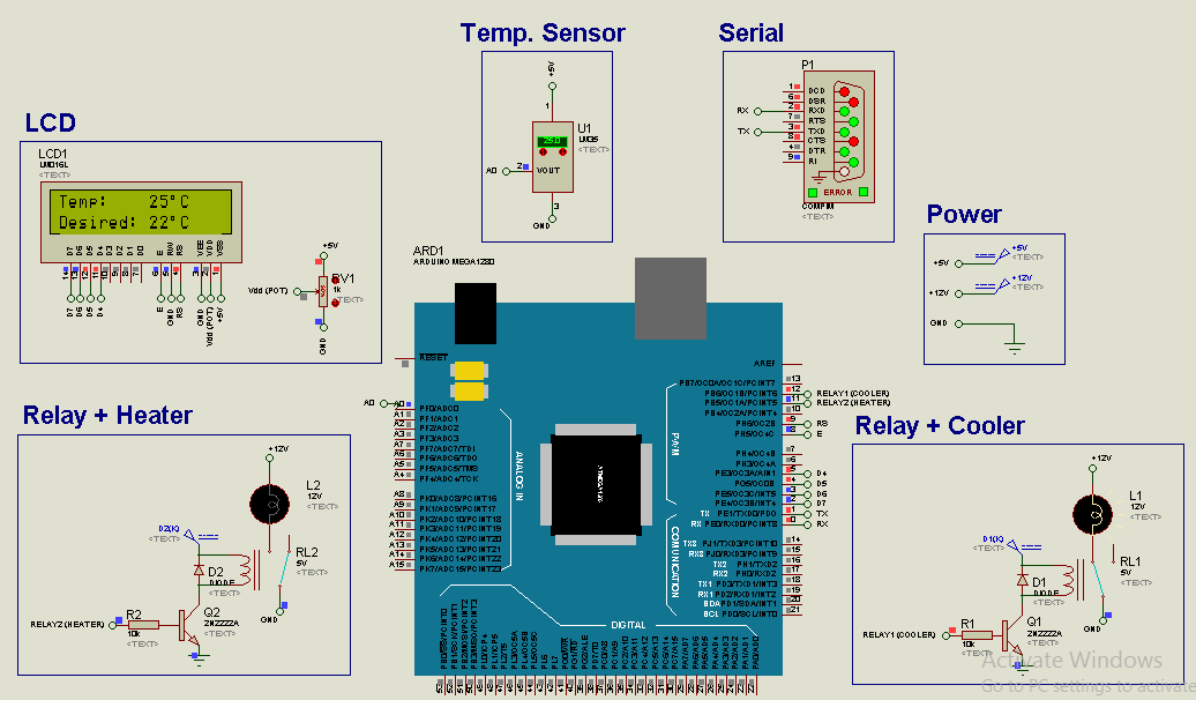


Figure 4-6: Cooling operation

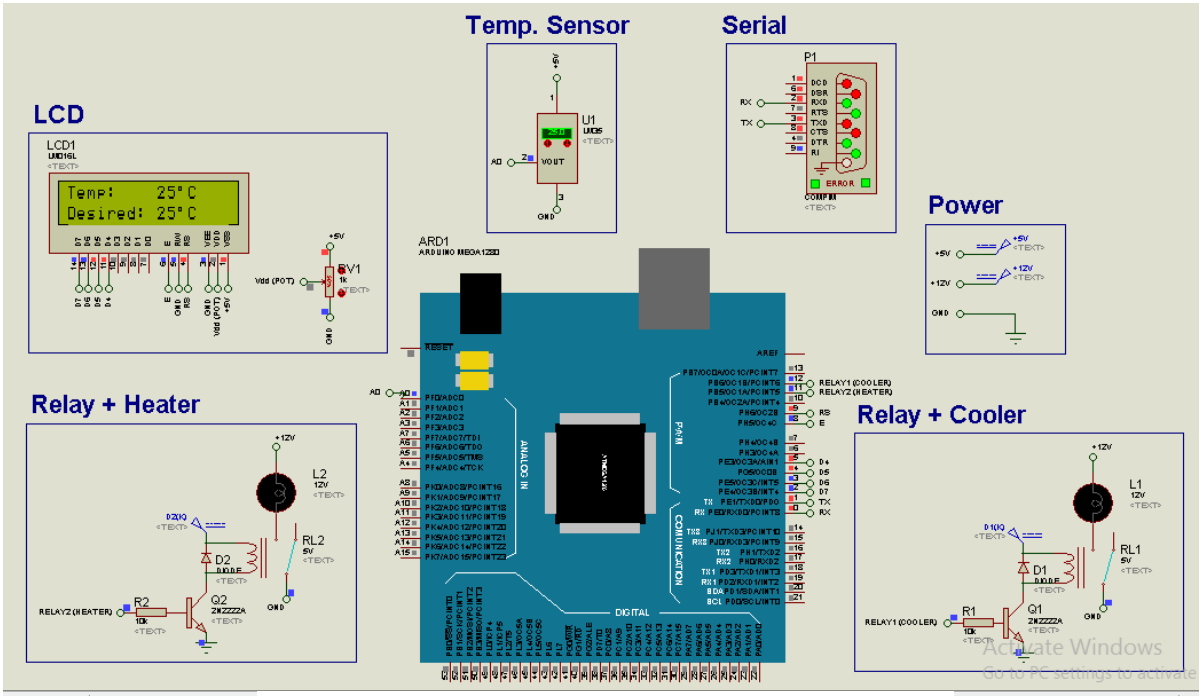


Figure 4-7: Default operation

4.3 Arduino control circuit implementation

In the hardware Implementation, instead of using heater and cooler, a green and red LED had being used. The green LED showed the operation of cooling and the red LED showed the operation of heating. After starting the system the server application showed a text “Waiting for client connection”, also it showed its IP address and port number. In the other side, the user in the client application entered the IP address of the server and the same port number as illustrated in Figure 4-8.

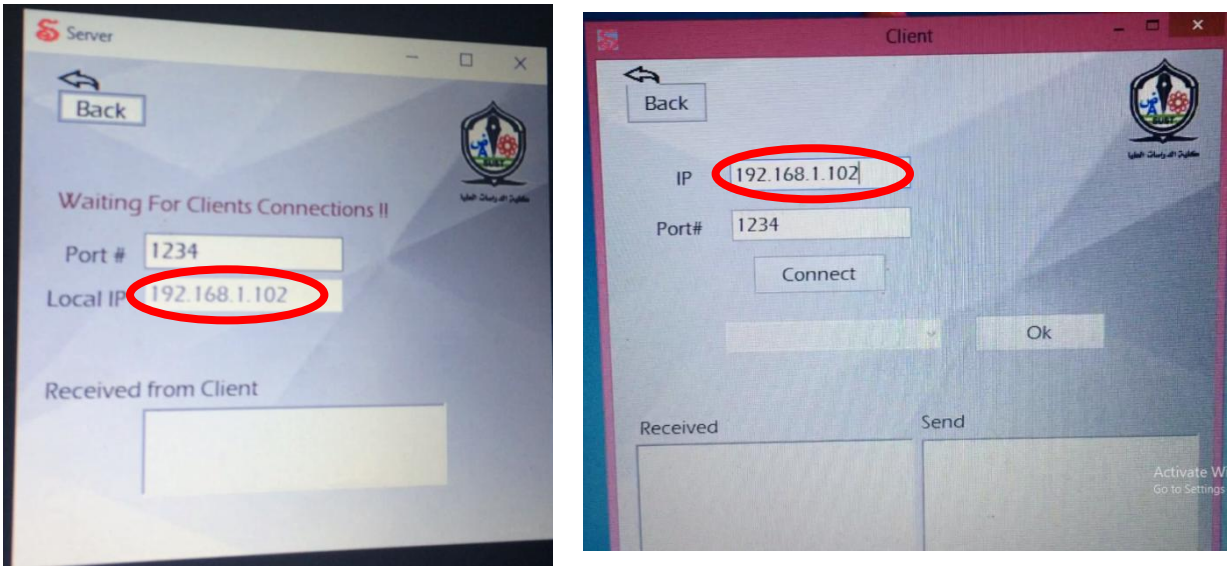


Figure 4-8: Server and Client setup

After that, the user pressed “connect” button and chose a specific COM port. Then the user pressed “OK” button and started to enter the desired temperature as illustrated in Figure 4-9, the desired temperature had been sent to the server, and the server in its turn received it and sent it to the control circuit as showed in Figure 4-10.

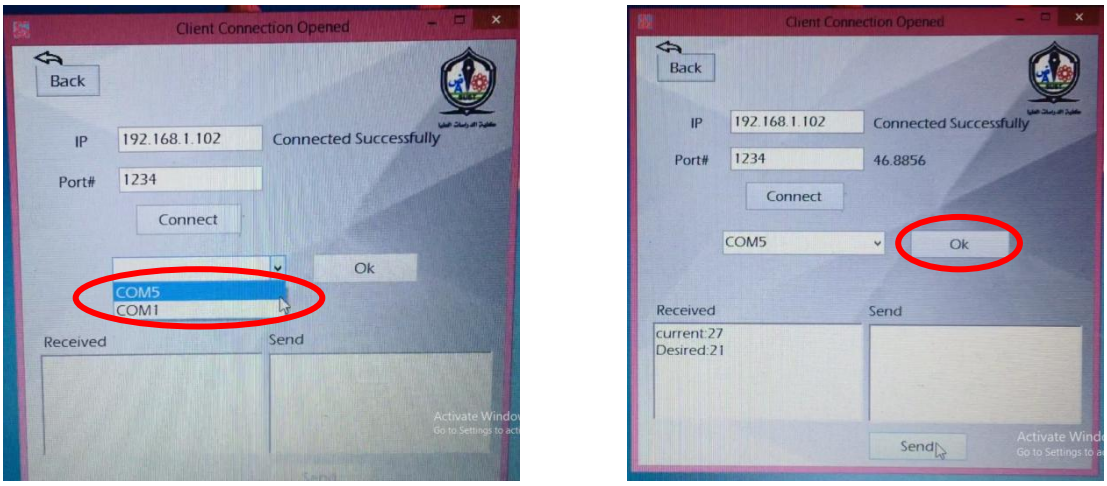


Figure 4-9: Opening COM port

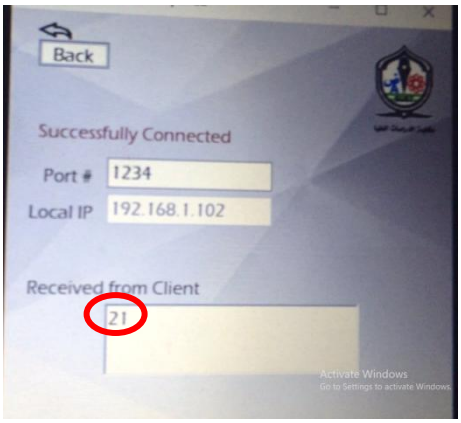


Figure 4-10: Receiving low temperature from the client

The control circuit decided to open the cooler (turn on greed LED)because the desired temperature is lower than the actual temperature. As illustrated in Figure 4-11.

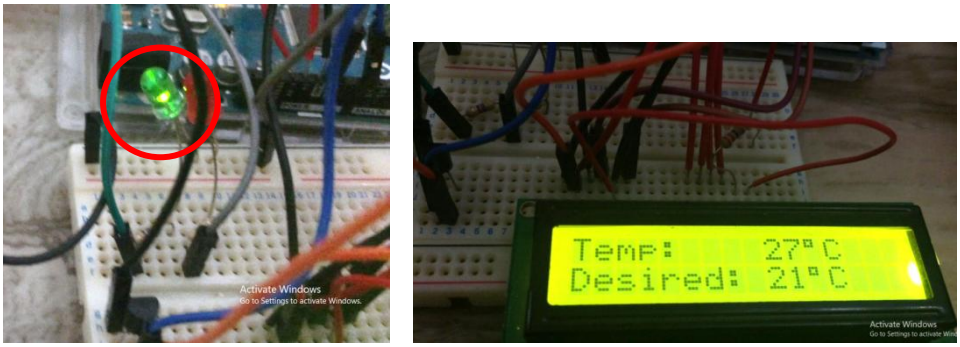


Figure 4-11: Hardware system response in cooling

Finally, the user entered a temperature which is higher than the actual temperature, in this case the user entered 33°C as shown in Figure 4-12.

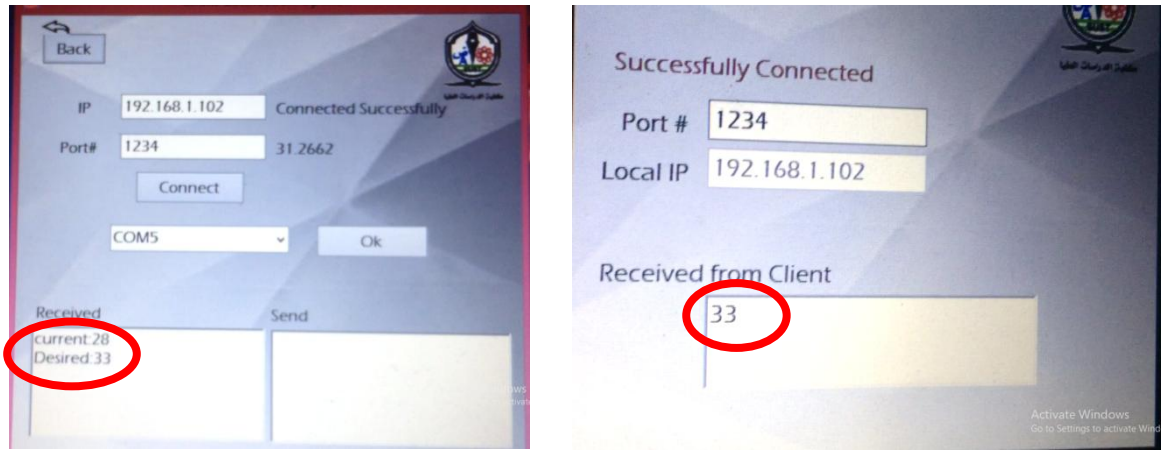


Figure 4-12: Sending and receiving low temperature

The control circuit received the high temperature and opened the heater (which is indicated by a red LED) as illustrated in Figure 4-13.

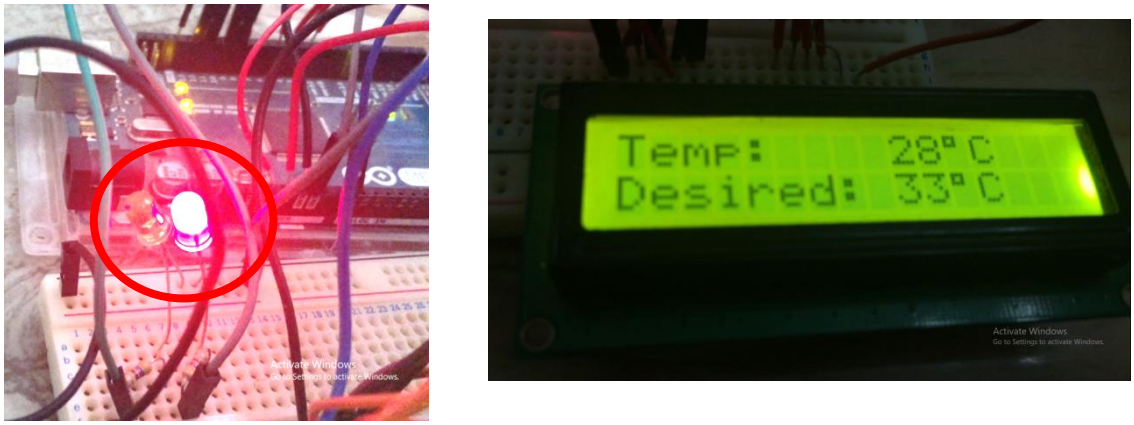


Figure 4-13: Sending and receiving high temperature

The control circuit by default turns off both of heater and cooler. In this case the desired temperature is equal to the actual temperature, both of LEDs are turned off as showed in Figure 4-14.

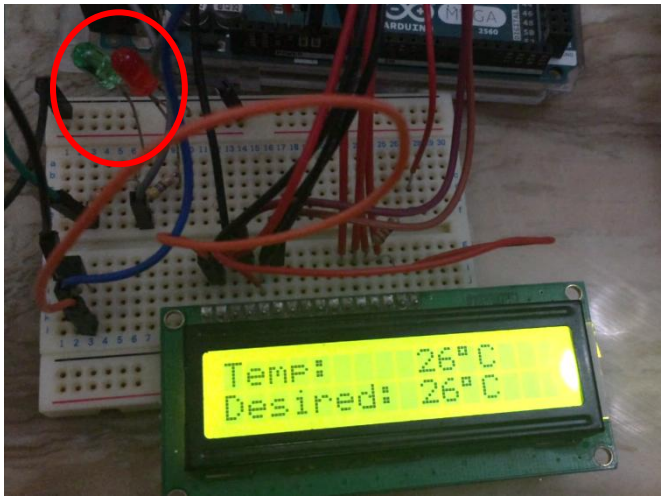


Figure 4-14: Equality of desired temperature and actual temperature

4.4 Response Time

The response time or delay defines how long it takes for the command to totally arrive at the destination**.** The following table shows the delay times depending on the way used in transferring the commands (Loopback**,** WLAN**,** Internet)**.**

Table 4-1: The Response Time

Method	**Response Time (millisecond)**
Loopback	1 – 18 msec
WLAN	16 – 99 msec
Internet	17 – 139 msec

Chapter Five

CONCLUSION AND RECOMMENDATIONS

Chapter Five

Conclusion and Recommendations

5.1 Conclusion

This thesis investigated the temperature control via Arduino through Internet. The system composed of GUI's that gives a self explanatory way to send commands to the desired object. The knowledge of the underlying infrastructure is not required to achieve the process of transporting the command.

An Asymmetric key encryption algorithm Rijndael has been handling using visual basic programming language, it's a 128-bit encryption algorithm which used for transferring an encrypted data between client and server to insure a security and protection. Also the authentication login form had been developed with hashed user name and password. The Implementation of the system had been developed using wired serial Interface due to high cost of wireless modules. The power consumption is not high and can be offered by using portable batteries. The overall time response from the initial results is affected by the Internet speed and connection type, but it's still possible to control the temperature with a high reliability.

5.2 Recommendations

In light of the discoveries and conclusion of the study, here are a few features to be considered in future work:

- Improve the system performance and flexibility by using mobile application or websites.
- Adding portable batteries can enhance the system power supply.
- Using wireless communication between hardware and software or embed the server computer within the control circuit.
- Enhance System security and protection.

REFFEERENCES

[1] E. HARSHAVARDHAN GOUD, "Real Time Based Temperature Control Using Arduino", *International Journal of Innovations in Engineering and Technology*, vol. 8, no. 2, 2017.

[2] "Arduino - Introduction", *Arduino.cc*, 2018. [Online]. Available: https://www.arduino.cc/en/Guide/Introduction. [Accessed: 18- Jul- 2018].

[3] "Arduino Uno Rev3", *Store.arduino.cc*, 2018. [Online]. Available: https://store.arduino.cc/usa/arduino-uno-rev3. [Accessed: 18- Jul- 2018].

[4] A. R3, "Arduino Mega 2560 R3 - DEV-11061 - SparkFun Electronics", *Sparkfun.com*, 2018. [Online]. Available: *https://www.sparkfun.com/products/11061. [Accessed: 18- Mar- 2018]. [5]*
[5] "Serial Communication - learn.sparkfun.com", *Learn.sparkfun.com*, 2018. [Online]. Available: https://learn.sparkfun.com/tutorials/serial-communication/rules-of-serial. [Accessed: 26- Feb- 2018].

[6] E. HARSHAVARDHAN, A. HARSHIKA, G. AKHIL, D. CHARISHMA, K. BHUPATHI and I. KUMARA, "Real Time Based Temperature Control Using Arduino", International Journal of Innovations in Engineering and Technology, vol. 8, no. 2, 2017.

[7] SHWETA MANGHNAN1, CHITTAMPALLY RAJITHA and KASARAM PRIYANKA, "Arduino Based Wireless System For

Temperature And Humidity Monitoring", *International Conference On Emerging Trends In Engineering, Science And Management.* 2017

[8] VIJAY S., MADAN B., AVINASH D. and CHANDRAKANT L, "Remote Temperature Monitoring System Using ARM, Arduino and ZigBee", *International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 5* 2016.

[9] R. NAIR and K. MOHAN, "Control of Temperature Using PID Controller", *International Journal of Science and Research* (*IJSR*), vol. 5, no. 5, pp. 1203-1206, 2015.

[10] RINA ABDULLAH, ZAIRI ISMAEL RIZMAN, NIK NUR SHAADAH NIK DZULKEFLI, SYILAIZAWANA ISMAIL, ROSMAWATI SHAFIE and MOHAMAD HUZAIMY JUSOH, "Design An Automatic Temperature Control System For Smart TudungSaji Using Arduino Microcontroller", *ARPN Journal of Engineering and Applied Sciences.* 2016.

APPENDIX A: Visual Basic code for client

A.1 Authentication

```
Imports System.IO
Imports System.Security.Cryptography
Public Class LoginForm
Dim  UserName As String, Password As String

Private Sub LoginButton_Click(sender As Object, e As EventArgs)
      Handles LoginButton.Click
      UserName = UserNameTextBox.Text
      Password = PasswordTextBox.Text

      If my_encryption(UserName) = "Qf5zLVzUOuZkpBRiVCAFVg=="And
my_encryption(Password) = "SYEi1d3HaKSrAjikurIiyg=="Then
            UserNameTextBox.Text = Nothing
            PasswordTextBox.Text = Nothing
            Me.Hide()
            Form1.Show()
      Else
            MsgBox("wrong username or password")
            UserNameTextBox.Text = Nothing
            PasswordTextBox.Text = Nothing

      End If
End Sub

Private Sub ExitButton_Click(sender As Object, e As EventArgs) Handles
      ExitButton.Click
      Application.Exit()
End Sub


Function my_encryption(ByRef text As String)
      Dim  crypto As New  RijndaelManaged()
      Dim  key As Byte() = {1, 23, 32, 45, 12, 34, 56, 32, 12,    34,
21, 34, 32, 123, 43, 21}
      Dim  iv As Byte() = {12, 34, 34, 34, 21, 56, 53, 21, 35,    65,
43, 24, 23, 56, 7, 45}
      Dim  tex() As Byte = System.Text.Encoding.Unicode.GetBytes(text)
      Dim  str As New MemoryStream
      Dim  encr As New CryptoStream(str, crypto.CreateEncryptor(key,
iv), CryptoStreamMode.Write)
      encr.Write(tex, 0, tex.Length)
      encr.FlushFinalBlock()
      text = Convert.ToBase64String(str.ToArray)
```

```
      Return text
EndFunction
EndClass
```

A.2 Main Program

```
Imports System.Security.Cryptography
Imports System.IO
Imports System.Text
Imports System.Media

Public Class Form1
Dim  rx_data As String = ""
Dim  online_ports As String
Dim  port As String
Dim  send_data As String
Dim  new_ports As String
Dim  desired AsS tring
Dim  rx As String
Dim  i As Integer
Dim  j As Integer = 0
Dim  k As Integer = 1
Dim  line As Integer = 1
Dim  stopp As DateTime
Dim  start As DateTime
Dim  elapsedTime AsTimeSpan
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Private Sub ConnectButton_Click(sender As Object, e As EventArgs)
Handles ConnectButton.Click
      Winsock.Close()
      Winsock.Connect(IPTextBox.Text, PortNumTextBox.Text)
      Me.Text = "Client Connection Opened"
      Label5.Show()
      Label5.Text = "Connected Successfully"
End Sub
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Private Sub SendButton_Click(sender As Object, e As EventArgs) Handles
SendButton.Click
      Label6.Show()
      start = Now
      desired = TxRichTextBox.Text
      send_data = TxRichTextBox.Text
      TxRichTextBox.Text = ""
      Winsock.SendData(my_encryption(send_data))
End Sub
```

```
Private Sub Winsock_DataArrival(sender As Object, e As
AxMSWinsockLib.DMSWinsockControlEvents_DataArrivalEvent) Handles
Winsock.DataArrival
      On Error Resume Next
      stopp = Now
      elapsedTime = stopp.Subtract(start)
      Label6.Text = elapsedTime.TotalMilliseconds.ToString()
      Winsock.GetData(rx_data)
      my_decryption(rx_data)
      If (rx_data.IndexOf("COM")) <> -1 Then
            i = (rx_data.IndexOf("COM")) - 1
            online_ports = Mid(rx_data, (i + 2), rx_data.Length)
            j = 0
            k = 0
            new_ports = online_ports.Replace(vbCrLf, "*")
                  For j = 0 To (new_ports.Length - 1)
                        If (new_ports.Chars(j) <>"*") Then
                              port = port + new_ports.Chars(j)
                        Else
                              ComboBox1.Items.Add(port)
                              port = ""
                        End If
                        ComboBox1.Enabled = True
                  Next
      Else
            rx = ""
            rx =
rx_data.Chars(Convert.ToInt32((rx_data.IndexOf("current:")) + 8)) +
rx_data.Chars(Convert.ToInt32((rx_data.IndexOf("current:")) + 9))
RxRichTextBox.Text = "current:" + rx&Environment.NewLine&"Desired:" +
desired
      End If

End Sub
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Private Sub BackButton_Click(sender As Object, e As EventArgs) Handles
BackButton.Click
      Me.Hide()
      LoginForm.Show()
End Sub
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Private Sub Form1_Load(sender As Object, e As EventArgs)
HandlesMyBase.Load
      ComboBox1.Enabled = False
      RxRichTextBox.Enabled = False
      TxRichTextBox.Enabled = False
      SendButton.Enabled = False
      Label6.Hide()
      Label5.Hide()
```

```
End Sub
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Private Sub OkButton_Click(sender As Object, e As EventArgs) Handles
OkButton.Click
      send_data = ComboBox1.Text
      RxRichTextBox.Enabled = True
      TxRichTextBox.Enabled = True
      SendButton.Enabled = True
      Winsock.SendData(my_encryption(send_data))
End Sub
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Function my_encryption(ByRef text As String)
      Dim  crypto As New  RijndaelManaged()
      Dim  key As Byte() = {1, 23, 32, 45, 12, 34, 56, 32, 12, 34, 21,
34, 32, 123, 43, 21}
      Dim  iv As Byte() = {12, 34, 34, 34, 21, 56, 53, 21, 35, 65, 43,
24, 23, 56, 7, 45}
      Dim  tex() As Byte = System.Text.Encoding.Unicode.GetBytes(text)
      Dim  str As New  MemoryStream
      Dim  encr As New  CryptoStream(str, crypto.CreateEncryptor(key,
iv), CryptoStreamMode.Write)
      encr.Write(tex, 0, tex.Length)
      encr.FlushFinalBlock()
      text = Convert.ToBase64String(str.ToArray)
      Return text
End Function
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Function my_decryption(ByRef text As String)
      Dim  crypto As New  RijndaelManaged()
      Dim  key As Byte() = {1, 23, 32, 45, 12, 34, 56, 32, 12, 34, 21,
34, 32, 123, 43, 21}
      Dim  iv As Byte() = {12, 34, 34, 34, 21, 56, 53, 21, 35, 65, 43,
24, 23, 56, 7, 45}
      Dim  encry() As Byte = Convert.FromBase64String(text)
      Dim  stream As New  MemoryStream
      Dim  dec As New  CryptoStream(stream,
crypto.CreateDecryptor(key, iv), CryptoStreamMode.Write)
      dec.Write(encry, 0, encry.Length)
      dec.FlushFinalBlock()
      text = System.Text.Encoding.Unicode.GetString(stream.ToArray)
      Return text
End Function
End Class
```

APPENDIX B: Visual Basic code for server

B.1 Authentication

```
Imports System.IO
Imports System.Security.Cryptography
Public Class LoginForm
Dim  UserName As String, Password As String
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Private Sub LoginButton_Click(sender As Object, e As EventArgs)
      Handles LoginButton.Click
      UserName = UserNameTextBox.Text
      Password = PasswordTextBox.Text
      If my_encryption(UserName) = "Qf5zLVzUOuZkpBRiVCAFVg==" And
my_encryption(Password) = "SYEi1d3HaKSrAjikurIiyg==" Then
            UserNameTextBox.Text = Nothing
            PasswordTextBox.Text = Nothing
            Me.Hide()
            Form1.Show()
      Else
            MsgBox("wrong username or password")
            UserNameTextBox.Text = Nothing
            PasswordTextBox.Text = Nothing
      End If
End Sub
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Private Sub ExitButton_Click(sender As Object, e As EventArgs) Handles
ExitButton.Click
      Application.Exit()
End Sub
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Function my_encryption(ByRef text As String)
      Dim  crypto As New  RijndaelManaged()
      Dim  key As Byte() = {1, 23, 32, 45, 12, 34, 56, 32, 12, 34, 21,
34, 32, 123, 43, 21}
      Dim  iv As Byte() = {12, 34, 34, 34, 21, 56, 53, 21, 35, 65, 43,
24, 23, 56, 7, 45}
      Dim  tex() As Byte = System.Text.Encoding.Unicode.GetBytes(text)
      Dim  str As New  MemoryStream
      Dim  encr As New  CryptoStream(str, crypto.CreateEncryptor(key,
iv), CryptoStreamMode.Write)
      encr.Write(tex, 0, tex.Length)
      encr.FlushFinalBlock()
      text = Convert.ToBase64String(str.ToArray)
      Return text
End Function
End Class
```

B.2 Main program

```
Imports System.IO
Imports System.IO.Ports
Imports System.Text
Imports System.Security.Cryptography

PublicClassForm1

Dim  rx_data As String
Dim  send_data As String
Dim  rec_from_serial As String
Dim  send_to_serial As String
Dim  preamb As String
Dim  ports_name As String()
Dim  port As String
Dim  num_of_ports As Integer = 0
Dim  stopp As DateTime
Dim  start As DateTime
Dim  elapsedTime As TimeSpan
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Private Sub Winsock_ConnectionRequest(sender As Object, e As
AxMSWinsockLib.DMSWinsockControlEvents_ConnectionRequestEvent) Handles
Winsock.ConnectionRequest
      Dim  online_com As String = ""
      Winsock.Close()
      Winsock.Accept(e.requestID)
      Me.Text = "Server Connection Opened"
      Label6.Text = "Successfully Connected"
      RxRichTextBox.Enabled = True
      num_of_ports = 0
      For Each port In ports_name
            online_com = port &Environment.NewLine&online_com
            num_of_ports = num_of_ports + 1
      Next port
      online_com = num_of_ports&Environment.NewLine&online_com
      Winsock.SendData(my_encryption(online_com))
End Sub
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Private Sub Winsock_DataArrival(sender As Object, e As
AxMSWinsockLib.DMSWinsockControlEvents_DataArrivalEvent) Handles
Winsock.DataArrival
      Winsock.GetData(RxRichTextBox.Text)
      rx_data = RxRichTextBox.Text
      RxRichTextBox.Text = Nothing
      my_decryption(rx_data)
      If (rx_data.IndexOf("COM")) <> -1 Then
            SerialPort.PortName = rx_data
```

```
            SerialPort.BaudRate = 9600
            SerialPort.Open()
      End If
      If (SerialPort.IsOpen) Then
            SerialPort.WriteLine(rx_data)
            start = Now
            rec_from_serial = (SerialPort.ReadExisting)
            SerialPort.Close()
      Else
            SerialPort.Close()
            SerialPort.Open()
            If (SerialPort.IsOpen) Then
                  SerialPort.WriteLine(rx_data)
                  start = Now
                  rec_from_serial = (SerialPort.ReadExisting)
                  SerialPort.Close()
                  RxRichTextBox.Text = rx_data
            End If
      End If
      Winsock.SendData(my_encryption(rec_from_serial))
End Sub
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Private Sub BackButton_Click(sender As Object, e As EventArgs) Handles
BackButton.Click
      Me.Hide()
      LoginForm.Show()
End Sub
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Private Sub SerialPort_DataReceived(sender As Object, e As
SerialDataReceivedEventArgs) Handles SerialPort.DataReceived
      System.Threading.Thread.Sleep(500)
      Try
            If SerialPort.IsOpenThen
                  SerialPort.Close()
            End If
      Finally
            If SerialPort.IsOpen Then
                  SerialPort.Close()
            End If
      End Try
      If SerialPort.IsOpen Then
            rec_from_serial = (SerialPort.ReadExisting)
            SerialPort.Close()
      Else
            If SerialPort.IsOpen Then
                  rec_from_serial = (SerialPort.ReadExisting)
            SerialPort.Close()
            End If
      End If
```

```
Winsock.SendData(my_encryption(rec_from_serial))
End Sub
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Private Sub  Form1_Load(sender As Object, e As EventArgs)
HandlesMyBase.Load
      ports_name = SerialPort.GetPortNames()
      RxRichTextBox.Enabled = False
      Winsock.Close()
      LocalIPTextBox.Text = Winsock.LocalIP
      Winsock.LocalPort = PortNumTextBox.Text
      Winsock.Listen()
      Label6.Text = "Waiting For Clients Connections !!"
End Sub
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Function my_encryption(ByRef text As String)
      Dim  crypto As New  RijndaelManaged()
      Dim  key As Byte() = {1, 23, 32, 45, 12, 34, 56, 32, 12, 34, 21,
34, 32, 123, 43, 21}
      Dim  iv As Byte() = {12, 34, 34, 34, 21, 56, 53, 21, 35, 65, 43,
24, 23, 56, 7, 45}
      Dim tex() As Byte = System.Text.Encoding.Unicode.GetBytes(text)
      Dim str As New MemoryStream
      Dim encr As New CryptoStream(str, crypto.CreateEncryptor(key,
iv), CryptoStreamMode.Write)
      encr.Write(tex, 0, tex.Length)
      encr.FlushFinalBlock()
      text = Convert.ToBase64String(str.ToArray)
      Return text
End Function
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Function my_decryption(ByRef text As String)
      Dim  crypto As New RijndaelManaged()
      Dim  key As Byte() = {1, 23, 32, 45, 12, 34, 56, 32, 12, 34, 21,
34, 32, 123, 43, 21}
      Dim  iv As Byte() = {12, 34, 34, 34, 21, 56, 53, 21, 35, 65, 43,
24, 23, 56, 7, 45}
      Dim encry() As Byte = Convert.FromBase64String(text)
      Dim  stream As New MemoryStream
      Dim dec As New CryptoStream(stream, crypto.CreateDecryptor(key,
iv), CryptoStreamMode.Write)
      dec.Write(encry, 0, encry.Length)
      dec.FlushFinalBlock()
      text = System.Text.Encoding.Unicode.GetString(stream.ToArray)
      Return text
End Function
End Class
```

APPENDIX C: Arduino code

```
#include <LiquidCrystal.h>

int i = 0;

byte reading, heater = 11, cooler = 12, TempArray[10], Temp;

String data;

String prev_data = "";

int d1 = 0;

constint rs = 9, en = 8, d4 = 5, d5 = 4, d6 = 3, d7 = 2;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

byte customChar[] =
{B11100,B10100,B11100,B00000,B00000,B00000,B00000,B00000}; //for the
sign of degree

void setup()

{

Serial.begin(9600);

pinMode(A0, INPUT); // Tell Arduino to make its Analog A0 pin as Input
reading pin

pinMode(heater, OUTPUT);

pinMode(cooler, OUTPUT);

lcd.begin(16, 2);

lcd.setCursor(0, 0);

lcd.print("Temp:");

lcd.setCursor(0, 1);

lcd.print("Desired:");

lcd.createChar(0, customChar);

}

void loop()

{

  Temp = 0;
```

```
for (i = 0; i < 10; i++)
  {
      TempArray[i] = (5.0 * analogRead(A0) * 100.0) / 1024;
      Temp = Temp + TempArray[i];
  }
reading = Temp / 9;
Serial.print("current:");
Serial.println(reading);
if (Serial.available() > 0)
    {
      data = Serial.readString();
      d1 = data[1] - 48 + ((data[0] - 48) * 10);
    }
lcd.setCursor(9, 0);
lcd.print(reading);
lcd.print("    ");
lcd.setCursor(11, 0);
lcd.write(byte(0));
lcd.print("C");
lcd.setCursor(9, 1);
lcd.print(d1);
lcd.print("    ");
lcd.setCursor(11, 1);
lcd.write(byte(0));
lcd.print("C");
if(d1 >= 0 && d1 <= 40)
    {
if ( (reading == d1) || ((reading + 1) == d1) || ((reading - 1) ==
d1))
```

```
      {
digitalWrite(heater, LOW);
digitalWrite(cooler, LOW);
delay(1000);
Serial.print("current:");
Serial.println(reading);
      }
else if ( (reading > d1))
      {
delay(1000);
digitalWrite(heater, LOW);
digitalWrite(cooler, HIGH);
Serial.print("current:");
Serial.println(reading);
      }
else
      {
delay(1000);
digitalWrite(heater, HIGH);
digitalWrite(cooler, LOW);
Serial.print("current:");
Serial.println(reading);
      }
    }
 }
```

APPENDIX D: LM35 Datasheet

National Semiconductor

November 2000

LM35
Precision Centigrade Temperature Sensors

General Description

The LM35 series are precision integrated-circuit temperature sensors, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature. The LM35 thus has an advantage over linear temperature sensors calibrated in ° Kelvin, as the user is not required to subtract a large constant voltage from its output to obtain convenient Centigrade scaling. The LM35 does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55 to +150°C temperature range. Low cost is assured by trimming and calibration at the wafer level. The LM35's low output impedance, linear output, and precise inherent calibration make interfacing to readout or control circuitry especially easy. It can be used with single power supplies, or with plus and minus supplies. As it draws only 60 µA from its supply, it has very low self-heating, less than 0.1°C in still air. The LM35 is rated to operate over a −55° to +150°C temperature range, while the LM35C is rated for a −40° to +110°C range (−10° with improved accuracy). The LM35 series is available packaged in hermetic TO-46 transistor packages, while the LM35C, LM35CA, and LM35D are also available in the plastic TO-92 transistor package. The LM35D is also available in an 8-lead surface mount small outline package and a plastic TO-220 package.

Features

- Calibrated directly in ° Celsius (Centigrade)
- Linear + 10.0 mV/°C scale factor
- 0.5°C accuracy guaranteeable (at +25°C)
- Rated for full −55° to +150°C range
- Suitable for remote applications
- Low cost due to wafer-level trimming
- Operates from 4 to 30 volts
- Less than 60 µA current drain
- Low self-heating, 0.08°C in still air
- Nonlinearity only ±¼°C typical
- Low impedance output, 0.1 Ω for 1 mA load

Typical Applications

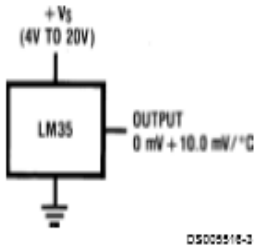
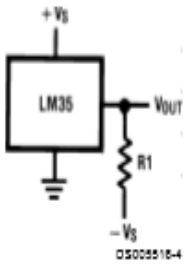


FIGURE 1. Basic Centigrade Temperature Sensor (+2°C to +150°C)



Choose $R_1 = -V_S/50$ µA
V_{OUT} = +1,500 mV at +150°C
= +250 mV at +25°C
= −550 mV at −55°C

FIGURE 2. Full-Range Centigrade Temperature Sensor