

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JnanaSangama, Belagavi – 590014.



MINI PROJECT REPORT

TEMPERATURE CONTROLLED SELF-REGULATING FAN

POST GRADUATING IN MASTER OF COMPUTER APPLICATION

For the academic year 2021-2022

SUBMITTED BY:

NAME: POOJA G A

USN : 1CR20MC058

NAME: MONISHA N

USN : 1CR20MC048

Under the guidance of:

Mrs. Divya

CMR INSTITUTE OF TECHNOLOGY, BENGALURU-560037



DEPARTMENT OF MASTER OF COMPUTER APPLICATION CERTIFICATE

This is to certify that the Mini Project Report entitled “TEMPERATURE CONTROLLED SELF REGULATING FAN” is a bonfire Mini Project work carried out by Pooja G A(1CR20MC058) and Monisha N(1CR20MC048), in partial fulfilment of ‘3th’ semester for the Degree of Master of Computer Application of Visvesvaraya Technological University, Belagavi, during the academic year 2021-22. It is certified that all corrections/suggestions indicated for Internal Assessments have been incorporated with the degree mentioned.

Mini Project Guide

Prof.Divya
Teaching Assistant
Dept. of MCA,

Head of Department

Prof.Gomati
Head of Department
Dept. of MCA,

External Viva

Name of the Examiners

1. _____
2. _____

Signature with Date

1. _____
2. _____

**CMR INSTITUTE OF TECHNOLOGY
Bengaluru -560037**

DECLARATION

I, Pooja G A(1CR20MC058) Monisha N (1CR20MC048), student of third semester MCA, Department of Master of Computer Application, CMR Institute of Technology, Bengaluru, declare, that the Mini Project Work entitled “TEMPERATURE CONTROLLED SELF REGULATING FAN” has been carried out by and submitted in partial fulfilment of the requirement of III semester Oct 2021-feb 2022. The matter embodied in this report has been submitted to any university or institute for the award of any other

degree or diploma.



Place: Bengaluru

Pooja G A and Monisha N

Date:

(1CR20MC058) AND (1CR20MC048)

ACKNOWLEDGEMENT

At the various stages in making the mini project, a number of people have given me invaluable comment on the manuscript. We take this opportunity to express my deepest gratitude and appreciation to all those who helped me directly or indirectly towards the successful completion of this project.

We would like to thank **Principal Dr. Sanjay Jain, CMR Institute of Technology** for his support throughout this project.

I express my whole hearted gratitude to **Prof. Gomati**, who is our respectable **Head of Dept. of MCA**. I wish to acknowledge for her valuable help and encouragement.

In this regard we owe a heartfelt gratitude to my guide **Prof.Divya Teaching Assistant of Department of MCA**, for her timely advice on the mini project and regular assistance throughout the project work. We would also like to thank the staff members of Department of MCA for their corporation.

ABSTRACT

This Project is automatic fan temperature controller that controls the temperature of a system by electric fan according to our requirement. Use of embedded technology makes this closed loop feedback control system efficient and reliable. Microcontroller allows dynamic and faster control. The sensed temperature and fan speed level values are simultaneously dependent on each other. It is very compact using few components and can be implemented for several applications including air-conditioners, water heaters, snow melter, ovens, heat-exchangers, mixers, furnaces, incubators, thermal baths and veterinary operating Tables. ARDUINO MKR WIFI 1010 micro controller is the heart of the circuit as it controls all the functions. The temperature sensor LM35 senses the Temperature and converts it into an electrical signal, which is applied to the microcontroller. The micro controller drives Transistor to control the fan speed and temperature of system.

Table of Content:

Chapter name	Page no
1.INTRODUCTION	8 - 9
2.PROPOSED SYSTEM	9 - 9
3.DESRIPTION	10 - 17
4.BLOCK DIAGRAM	19 - 20
5.DESIGN AND CODE	23 - 28
6.SCREEN SHOTS	29 – 31
7.CONCLUSION	34 - 34

List of Figures

Fig.1 Arduino MKR WIFI 1010

Fig.2 Bread board

Fig.3 DHT22 Temperature Humidity Sensor

Fig.4 2N3904 NPN Transistor

Fig.5 CPU Fan

Fig.6 1kohm resistor

Fig.7 Connection of things

INTRODUCTION

With the advancement in technology, intelligent systems are introduced every day. Everything is getting more sophisticated and intelligible. There is an increase in the demand of cutting edge technology and smart electronic systems. Microcontrollers play a very important role in the development of the smart systems as brain is given to the system. Microcontrollers have become the heart of the new technologies that are being introduced daily. A microcontroller is mainly a single chip microprocessor suited for control and automation of machines and processes. Today, microcontrollers are used in many disciplines of life for carrying out automated tasks in a more accurate manner. Almost every modern day device including air conditioners, power tools, toys, office machines employ microcontrollers for their operation. Microcontroller essentially consists of timers and counters, interrupts, memory, input/output ports, analog to digital converters on a single chip. With this single chip integrated circuit design of the microcontroller the size of control board is reduced and power consumption is low. This project presents

the design and simulation of the fan speed control system using PWM technique based on the room temperature. A temperature sensor has been used to measure the temperature of the room and the speed of the fan is varied according to the room temperature using PWM technique.

PROPOSED SYSTEM

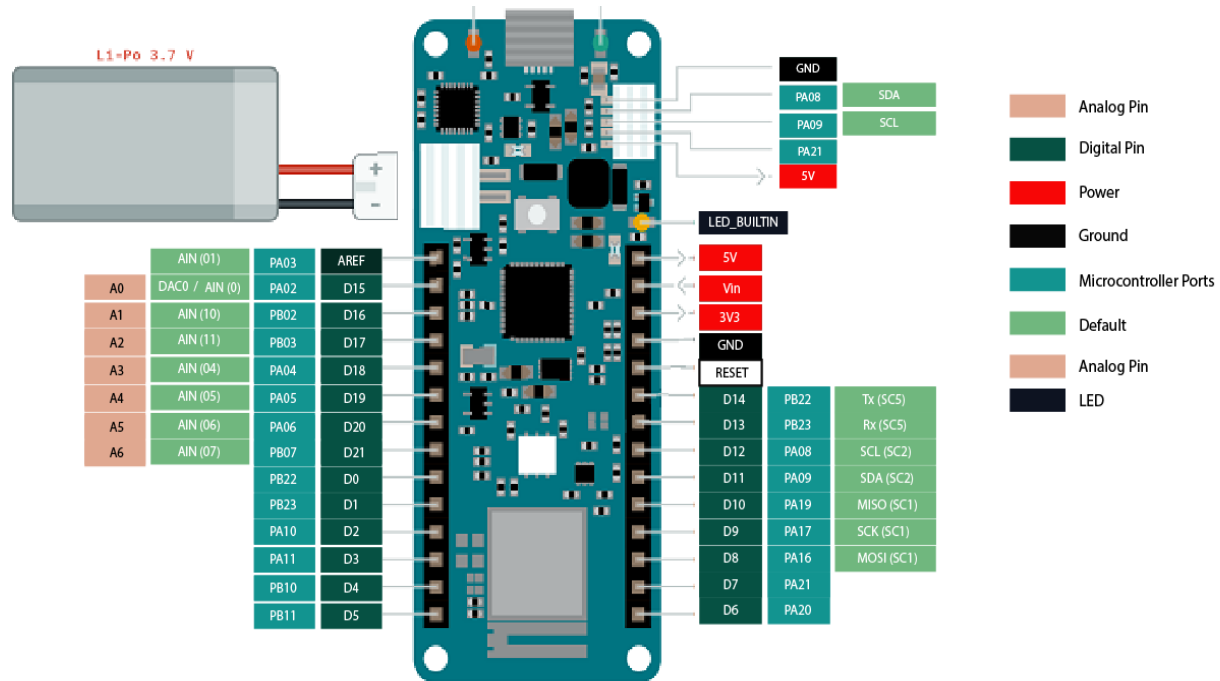
In the proposed system, microcontroller plays a vital role in the smart systems development. Microcontrollers have become an essential part in the present technologies that are being presented daily. This article discusses temperature based fan speed control and monitoring system using an Arduino MKR WIFI 1010 system. This system is used to control the cooling system automatically based on the room temperature. The system uses an Arduino board to implement a control system. Since this system is proposed to control the cooling system and it is very important to know Arduino controlled system well.

DESCRIPTION

The temperature-based fan speed control system can be done by using an electronic circuit using an Arduino MKR WIFI 1010 board. Now Arduino board is very progressive among all electronic circuits, thus we employed Arduino board for fan speed control. The proposed system is designed to detect the temperature of the room and send that information to the Arduino board. Then the Arduino board executes the contrast of current temperature and set temperature based on the inbuilt program of the Arduino.

The outcome obtained from the operation is given through the output port of an Arduino MKR WIFI 1010 board connecting fan.

ARDUINO MKR WIFI 1010



Get the MKR WIFI 1010. It connects easily to other Arduino products and is configurable using Arduino software — and you don't need to be a network expert. This is the newest version of the MKR 1000 WIFI, but with an ESP32 module on board made by U-BLOX.

The MKR WIFI 1010 is a significant improvement on the MKR 1000 WIFI. It's equipped with an ESP32 module made by U-BLOX. This board aims to speed up and simplify the

prototyping of WiFi based IoT applications thanks to the flexibility of the ESP32 module and its low power consumption.

The board is composed of three main blocks:

- SAMD21 Cortex-M0+ 32bit Low Power ARM MCU;
- U-BLOX NINA-W10 Series Low Power 2.4GHz IEEE® 802.11 b/g/n Wi-Fi; and
- ECC508 Crypto Authentication.

The MKR WIFI 1010 includes 32-bit computational power, the usual rich set of I/O interfaces, and low power Wi-Fi with a Cryptochip for secure communication using SHA-256 encryption. Plus, it offers ease of use Arduino Software (IDE) for code development and programming. All of these features make this board the preferred choice for the emerging IoT battery-powered projects in a compact form.

Its USB port can be used to supply power (5V) to the board. It has a Li-Po charging circuit that allows the Arduino MKR WIFI 1010 to run on battery power or an external 5 volt source, charging the Li-Po battery while running on external power. Switching from one source to the other is done automatically.

Warning: Unlike most Arduino boards, the MKR WIFI 1010 runs at 3.3V. The maximum voltage that the I/O pins can tolerate is 3.3V. Applying voltages higher than 3.3V to any I/O pin could damage the board. While output to 5V digital devices is possible, bidirectional communication with 5V devices needs proper level shifting.

The Arduino MKR wifi 1010 is based on the SAMD21 microcontroller.

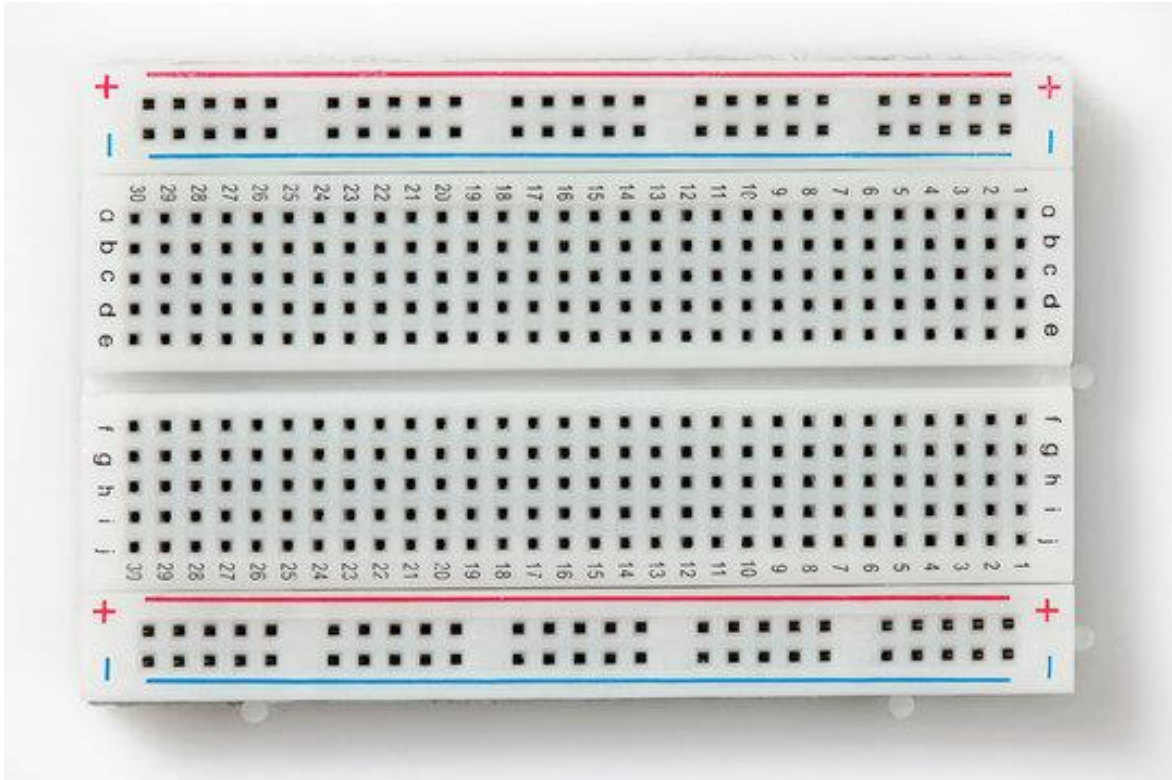
MICROCONTROLLER	SAMD21 Cortex®-M0+ 32bit low power ARM MCU (datasheet)
RADIO MODULE	U-blox NINA-W102 (datasheet)
BOARD POWER SUPPLY (USB/VIN)	5V
SECURE ELEMENT	ATECC508 (datasheet)
SUPPORTED BATTERY	Li-Po Single Cell, 3.7V, 1024mah Minimum
CIRCUIT OPERATING VOLTAGE	3.3V
DIGITAL I/O PINS	8
PWM PINS	13 (0 .. 8, 10, 12, 18 / A3, 19 / A4)
UART	1
SPI	1
I2C	1
ANALOG INPUT PINS	7 (ADC 8/10/12 bit)
ANALOG OUTPUT PINS	1 (DAC 10 bit)
EXTERNAL INTERRUPTS	10 (0, 1, 4, 5, 6, 7, 8,9, 16 / A1, 17 / A2)
DC CURRENT PER I/O PIN	7 ma
CPU FLASH MEMORY	256 KB (internal)
SRAM	32 KB

EEPROM	No
CLOCK SPEED	32.768 khz (RTC), 48 mhz
LED_BUILTIN	6
USB	Full-Speed USB Device and embedded Host
LENGTH	61.5 mm
WIDTH	25 mm
WEIGHT	32 gr.

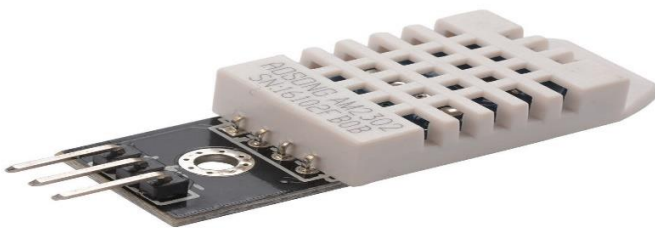
BREAD BOARD

Breadboards are one of the most fundamental pieces when learning how to build circuits.

In this tutorial, you will learn a little bit about what breadboards are, why they are called breadboards, and how to use one. Once you are done you should have a basic understanding of how breadboards work and be able to build a basic circuit on a breadboard.



DHT22 TEMPERATURE HUMIDITY SENSOR



The DHT22 is a basic, low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin (no analog input pins needed). It's fairly simple to use, but requires careful timing to grab data. The only real downside of this sensor is you can only

get new data from it once every 2 seconds, so when using our library, sensor readings can be up to 2 seconds old. Simply connect the first pin on the left to 3-5V power, the second pin to your data input pin and the rightmost pin to ground. Although it uses a single-wire to send data it is not Dallas One Wire compatible! If you want multiple sensors, each one must have its own data pin.

2N3904 NPN Transistor



2N3904 is a semiconductor device used to amplify or switch electronic signals and electrical power. It is composed of semiconductor material usually with at least three terminals for connection to an external circuit. A voltage or current applied to one pair of the transistor's

terminals controls the current through another pair of terminals. Because the controlled (output) power can be higher than the controlling (input) power, a transistor can amplify a signal. Today, some transistors are packaged individually, but many more are found embedded in integrated circuits.

Features:-

- Advanced process technology
- Low error voltage
- Fast switching speed
- Full-voltage operation
- High power and current handling capability

CPU FAN



A computer fan is any fan inside, or attached to, a computer case used for active cooling. Fans are used to draw cooler air into the case from the outside, expel warm air from inside and move air across a heat sink to cool a particular component. Both axial and sometimes centrifugal (blower/squirrel-cage) fans are used in computers. Computer fans commonly come in standard sizes, such as 120 mm (most common), 140 mm, 240 mm, and even 360 mm. Computer fans are powered and controlled using 3-pin or 4-pin fan connectors.

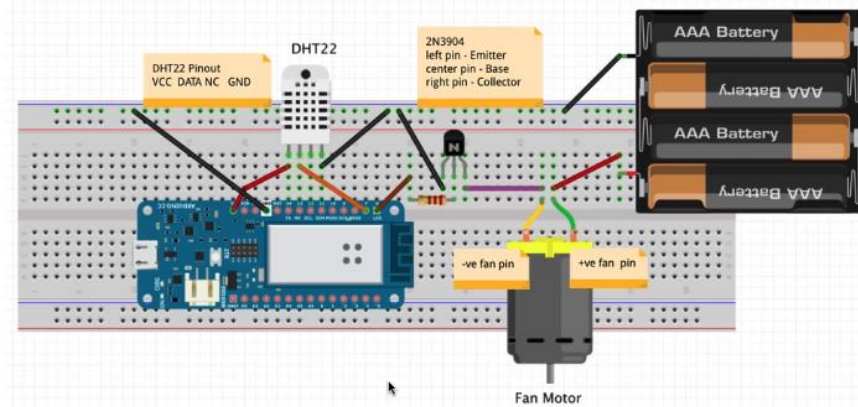
RESISTOR

A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages, bias active elements, and

terminate transmission lines, among other uses. High-power resistors that can dissipate many watts of electrical power as heat, may be used as part of motor controls, in power distribution systems, or as test loads for generators. Fixed resistors have resistances that only change slightly with temperature, time or operating voltage. Variable resistors can be used to adjust circuit elements (such as a volume control or a lamp dimmer), or as sensing devices for heat, light, humidity, force, or chemical activity.

WORKING OF PROJECT

Temperature Control Thing



MKR Pin	DHT22 Pin	Other
5V	VCC	
GND	GND	
7	Data	
6		resistor

Transistor Pins	
Emitter	GND
Base	resistor
Collector	-ve fan motor

Power Pack	
+ve	+ve fan motor
-ve	GND

Activate Wii
Go to Settings t

This project is a standalone automatic fan speed controller that controls the speed of an electric fan according to our requirement. Use of embedded technology makes this closed loop feedback control system efficient and reliable. Microcontroller allows dynamic and faster control.

The sensed temperature and fan speed level may be varied. It is very compact using few components and can be implemented for several applications including air conditioners, water heaters, snow melters, ovens, heat exchangers, mixers .

ARDUINO MKR WIFI 1010 micro controller is the heart of the circuit as it controls all the functions. The temperature sensor senses the temperature and converts it into an electrical signal, which is applied to the micro controller.

APPLICATIONS

1. Temperature based fan -speed controller is useful for cooling the processor in the laptops and personal computers “more efficiently”. Generally, fan in laptop comes with only two or three possible speeds. So, it results in more power consumption.
2. The fan designed in this project, has different values of speed according to temperature change. This can be also used in small scale industries for cooling the electrical equipment.

ADVANTAGES

1. This project can be used in Home.
2. This project can be used in Industry.
3. This will help in saving the energy.
4. To monitor the environments that is not comfortable, or possible, for humans to monitor, especially for extended periods of time.
5. Prevents waste of energy when it's not hot enough for a fan to be needed.

DISADVANTAGES

- 1.It can only be maintained by technical person. Thus, it becomes difficult to be maintained.
- 2.Due to temperature variation, after sometimes its efficiency may decrease.

FUTURE SCOPE

- 1.We can monitor more parameters like humidity, light and at the same time control them.
- 2.We can send this data to a remote location using mobile or internet.
- 3.We can draw graphs of variation in these parameters using computer.

SOFTWARE SPECIFICATION

Arduino is an open source electronics platform which uses simple I/O boards and a development environment to interact with the board. Arduino boards are capable of performing multiple functionalities based on the set of instructions sent to the microcontroller. The open-source Arduino software (Arduino IDE) is used to write code (C and C++) and upload it to the board. This is an analysis of the Arduino IDE architecture

performed by exploring the udemy courses. We have aimed at providing insight into the system from different viewpoints. First, a context view along with stakeholder analysis is performed, followed by development view, deployment view and finally technical debt.

OS:- , CPU with Pentium 4 or above , WINDOWS 10

AURDINO IDE:-AURDINO CLOUD

RAM:- 256MB

HARD DISK STORAGE SPACE-600MB

HARDWARE REQUIREMENTS

For the Arduino IDE in cloud or to install and run, the system is required to have 256 MB RAM on top of the requirements for the operating system, CPU with Pentium 4 or above.

SOURCE CODE

```
#include<DHT.h>
```

```
#include<DHT_U.h>
```

```
/*
```

Sketch generated by the Arduino IoT Cloud Thing "IOT cloud temperature controlled fan"

<https://create.arduino.cc/cloud/things/72c9880a-49d8-4daa-9916-3061debb65ac> Arduino

IoT Cloud Variables description

The following variables are automatically generated and updated when changes are made to the Thing float

```
max_temp;
```

```
float temperature;
```

```
bool fan_on;
```

```
bool override_fan_control;
```

Variables which are marked as READ/WRITE in the Cloud Thing will also have functions which are called when their values are changed from the Dashboard.

These functions are generated with the Thing and added at the end of this sketch.

```
*/
```

```
#include "thingProperties.h"
```

```
#define DHTPIN 7
```

```
#define DHTTYPE DHT22 DHT
```

```
dht(DHTPIN DHTTYPE);
```

```
#define fan 6
```

```
#define default_max_temp 20
```

```
void setup() {
```



```

// Initialize serial and wait for port to open:

Serial.begin(9600);

// This delay gives the chance to wait for a Serial Monitor without blocking if none is found delay(1500);


// Defined in thingProperties.h
initProperties();

// Connect to Arduino IoT Cloud
ArduinoCloud.begin(ArduinoIoTPreferredConnection);

/*

The following function allows you to obtain more information related
to the state of network and IoT Cloud connection and errors the higher
number the more granular information you'll get.

The default is 0 (only errors).

Maximum is 4

*/

setDebugMessageLevel(2);

ArduinoCloud.printDebugInfo();

//Initialize dht.begin();

pinMode(fan, OUTPUT); digitalWrite(fan,LOW);//turns
off fan if(max_temp==NULL){
max_temp=default_max_temp;

}
}

```

```
void loop() {  
  
  // Initialize serial and wait for port to open:  
  
  Serial.begin(9600);  
  
  // This delay gives the chance to wait for a Serial Monitor without blocking if none is found delay(1500);  
  
  
  
  // Defined in thingProperties.h  
  initProperties();  
  
  
  
  // Connect to Arduino IoT Cloud  
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);  
  
  
  
  /*  
  
  The following function allows you to obtain more information related  
  to the state of network and IoT Cloud connection and errors the higher  
  number the more granular information you'll get.  
  
  The default is 0 (only errors).  
  
  Maximum is 4  
  
  */  
  setDebugMessageLevel(2);  
  
  ArduinoCloud.printDebugInfo();  
  
  //Initialize dht.begin();  
  
  pinMode(fan, OUTPUT); digitalWrite(fan,LOW);//turns  
  off fan
```

```
if(max_temp==NULL){  
    max_temp=default_max_temp;  
  
}  
}
```

```
void loop() {  
    ArduinoCloud.update();  
  
    // Your code here  
  
    //read temperature data  
    temperature=dht.readTemperature();  
    Serial.print("Temp:"); Serial.print(temperature);  
    Serial.print("Celcius");  
  
    //Control fan or regulate temperature based on user input settings  
    update_system();  
  
    delay(5000); //delay for 5 seconds  
  
}
```

```
void regulate_temperature(){  
    if(temperature>max_temp){  
  
        //turn on fan  
        digitalWrite(fan,HIGH);  
        fan_on=true;  
  
    }
```

```
else

{

    digitalWrite(fan,LOW);
    fan_on=false;

}

}

void turn_fan_on_off(){
    if(fan_on){
        digitalWrite(fan,HIGH);

    }

    else ArduinoCloud.update();

    // Your code here

    //read temperature data
    temperature=dht.readTemperature();
    Serial.print("Temp:"); Serial.print(temperature);
    Serial.print("Celcius");

    //Control fan or regulate temperature based on user input settings
    update_system();

    delay(5000);//delay for 5 seconds

}
```

```

void regulate_temperature(){
    if(temperature>max_temp){

        //turn on fan
        digitalWrite(fan,HIGH);
        fan_on=true;

    }
    else

    {

        digitalWrite(fan,LOW);
        fan_on=false;

    }
}

```

```

void turn_fan_on_off(){
    if(fan_on){
        digitalWrite(fan,HIGH);

    }

    else

    /*

```

Since FanOn is READ_WRITE variable, onFanOnChange() is executed every time a new value is received from IoT Cloud.

```

*/

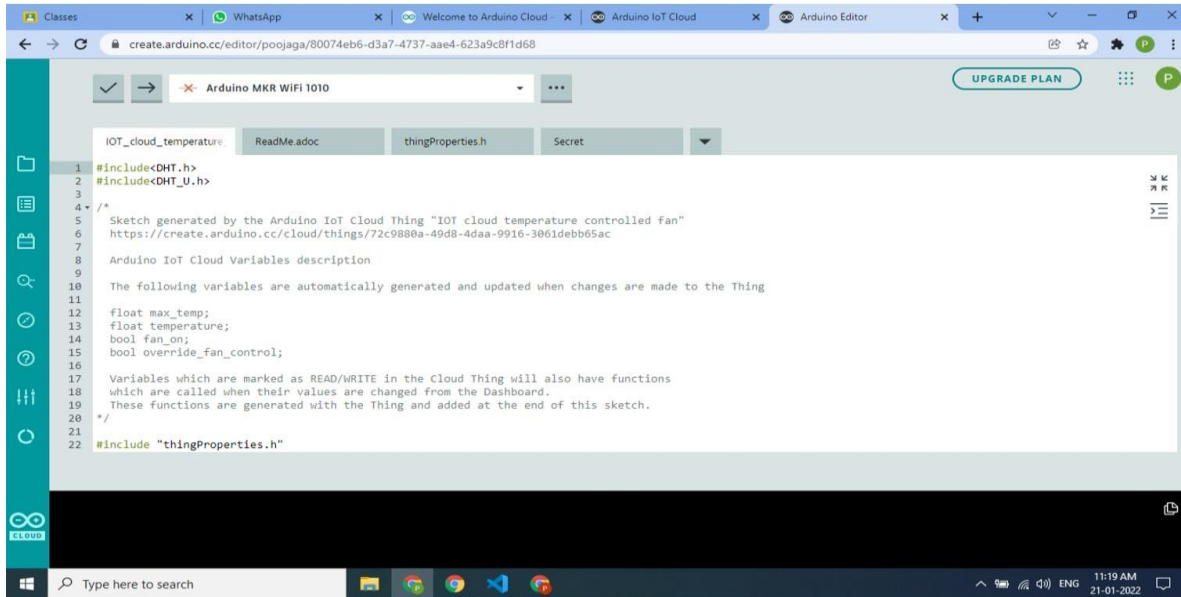
void onFanOnChange() {

    // Add your code here to act upon FanOn change
    update_system();

}

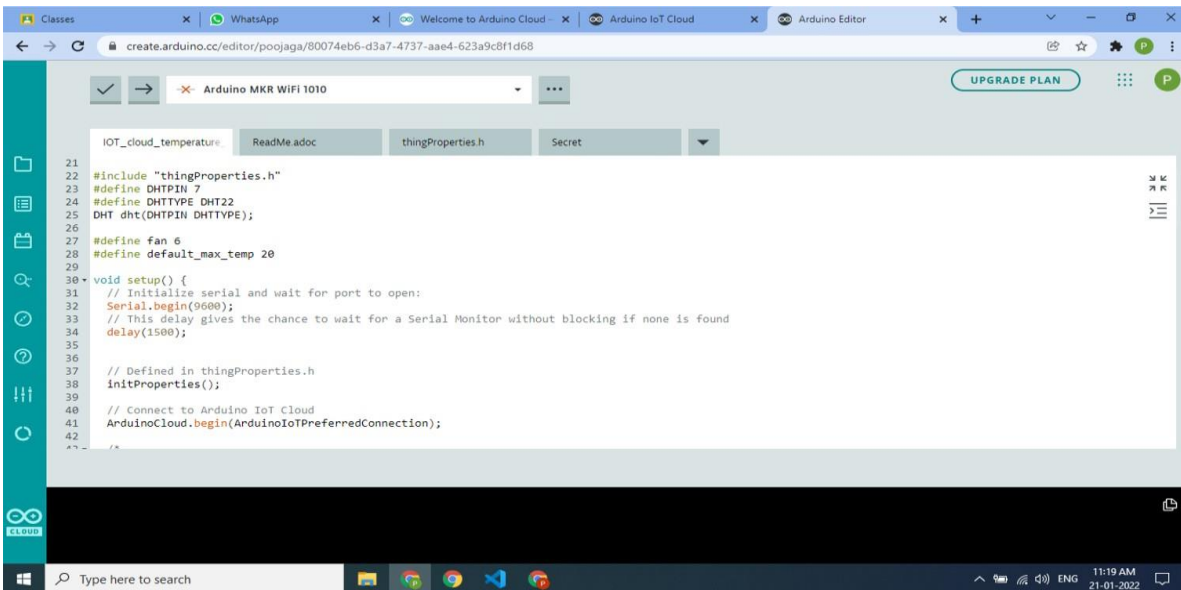
```

SCREEN SHOT



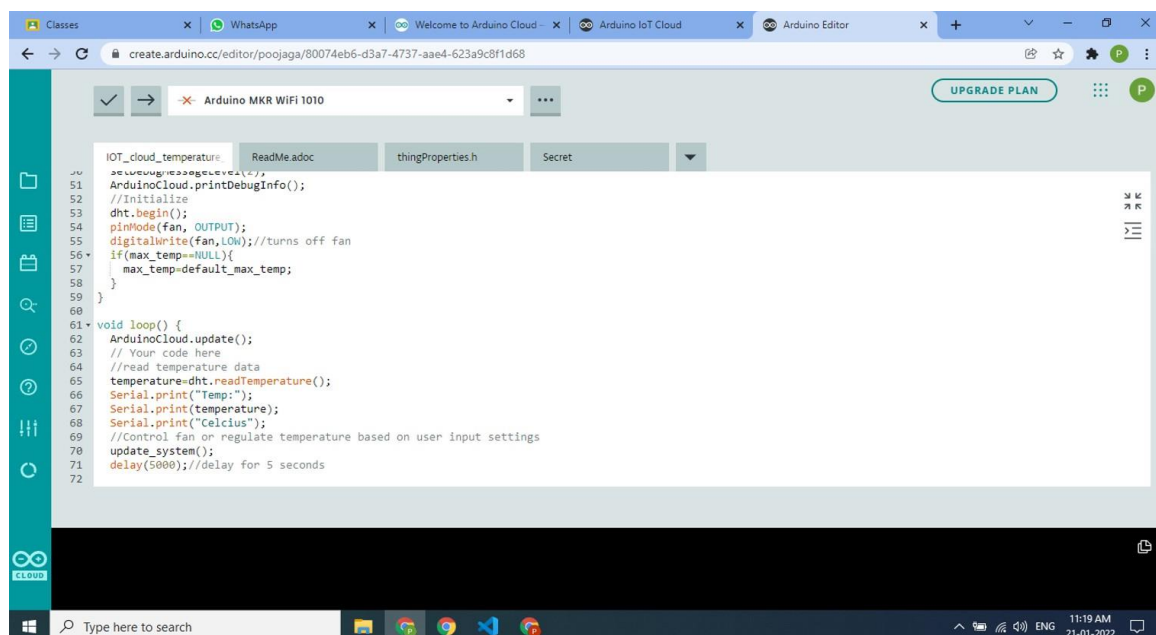
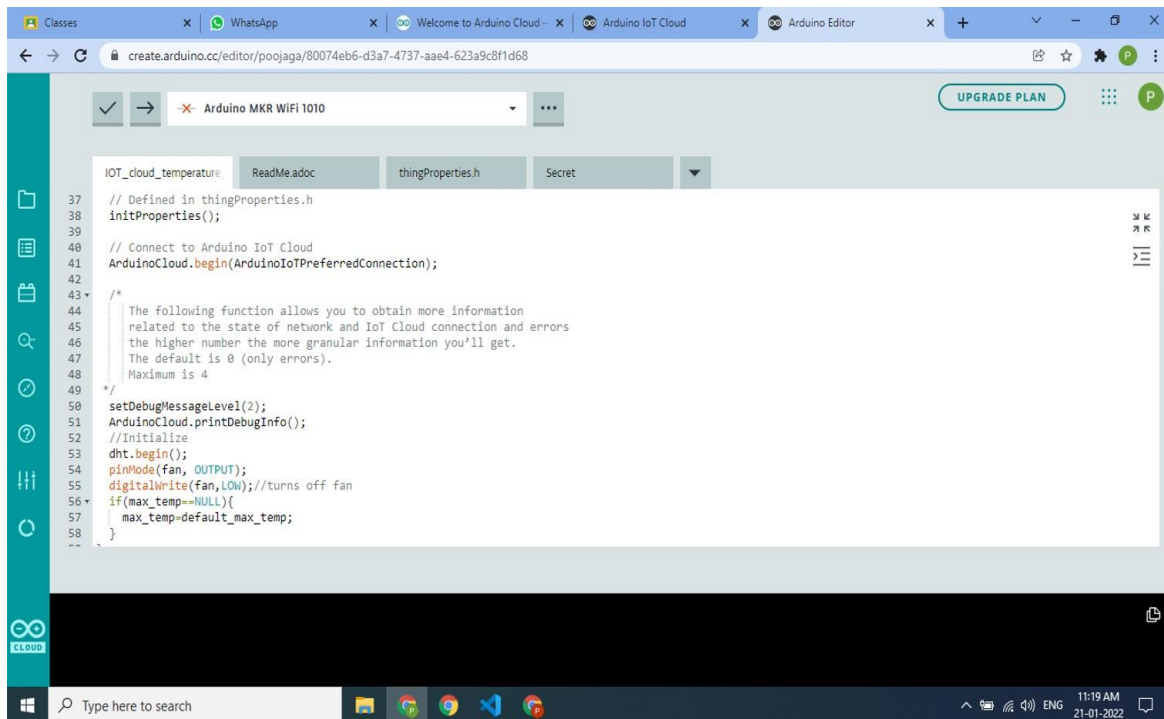
The screenshot shows the Arduino IDE interface with the following code:

```
1 #include<DHT.h>
2 #include<DHT_U.h>
3
4 /*
5  Sketch generated by the Arduino IoT Cloud Thing "IOT cloud temperature controlled fan"
6  https://create.arduino.cc/cloud/things/72c9880a-49d8-4daa-9916-3b61debb65ac
7
8  Arduino IoT Cloud Variables description
9
10 The following variables are automatically generated and updated when changes are made to the Thing
11
12 float max_temp;
13 float temperature;
14 bool fan_on;
15 bool override_fan_control;
16
17 Variables which are marked as READ/WRITE in the Cloud Thing will also have functions
18 which are called when their values are changed from the Dashboard.
19 These functions are generated with the Thing and added at the end of this sketch.
20 */
21 #include "thingProperties.h"
```



The screenshot shows the continuation of the code in the Arduino IDE with the following code:

```
21
22 #include "thingProperties.h"
23 #define DHTPIN 7
24 #define DHTTYPE DHT22
25 DHT dht(DHTPIN, DHTTYPE);
26
27 #define fan 6
28 #define default_max_temp 20
29
30 void setup() {
31   // Initialize serial and wait for port to open:
32   Serial.begin(9600);
33   // This delay gives the chance to wait for a Serial Monitor without blocking if none is found
34   delay(1500);
35
36   // Defined in thingProperties.h
37   initProperties();
38
39   // Connect to Arduino IoT Cloud
40   ArduinoCloud.begin(ArduinoIoTPreferredConnection);
41
42   //
```



The screenshot shows the Arduino IDE interface with the following code in the main editor:

```
74 }
75 void regulate_temperature(){
76   if(temperature>max_temp){
77     //turn on fan
78     digitalWrite(fan,HIGH);
79     fan_on=true;
80   }
81   else
82   {
83     digitalWrite(fan,LOW);
84     fan_on=false;
85   }
86 }
87 void turn_fan_on_off(){
88   if(fan_on){
89     digitalWrite(fan,HIGH);
90   }
91   else
92   {
93     digitalWrite(fan,LOW);
94   }
95 }
96 }
```

The IDE shows the 'IOT_cloud_temperature' file is selected, with tabs for 'ReadMe.adoc', 'thingProperties.h', and 'Secret'. The bottom status bar indicates the board is 'Arduino MKR WiFi 1010'.

The screenshot shows the Arduino IDE interface with the following code in the main editor:

```
113 regulate_temperature();
114 }
115
116 /*
117  Since OverrideFanControl is READ_WRITE variable, onOverrideFanControlChange() is
118  executed every time a new value is received from IoT Cloud.
119  */
120 void onOverrideFanControlChange() {
121   // Add your code here to act upon OverrideFanControl change
122   update_system();
123 }
124
125 /*
126  Since FanOn is READ_WRITE variable, onFanOnChange() is
127  executed every time a new value is received from IoT Cloud.
128  */
129 void onFanOnChange() {
130   // Add your code here to act upon FanOn change
131   update_system();
132 }
133
134 }
```

The IDE shows the 'IOT_cloud_temperature' file is selected, with tabs for 'ReadMe.adoc', 'thingProperties.h', and 'Secret'. The bottom status bar indicates the board is 'Arduino MKR WiFi 1010'.

TESTING

The purpose of testing phase is finding out the errors or effects by testing each system segments. These components may be functions, objects or the components are incorporated to form the whole system during system testing. At this step, testing should concentrate on the functional requirements of the system such that requirements are met and should not behave in an unexpected way. The inputs are data testes which are used to test the system and by using these inputs the output can be obtained when the system work according to its specification. Cohesive system behaviour can be examined. The system behaviour can be carefully examined with all possible combinations of conditions by choosing the test cases. In like manner, expected behaviour of the system under distinctive blends is given. Accordingly test cases are chosen which have inputs and the yields are on expected lines, inputs that are substantial and for which suitable message must be given and inputs that do not happen every now and again which can be viewed as unit cases.

Unit Testing: Unit testing focuses verification effort on the unit of software design. Important control paths can be tested by utilizing the unit test plans which are prepared during system development design phase to know the error that are present in module boundary. Each module interfaces were tested to ensure proper flow of information in to and out of the modules under consideration every error handling path were tested and all statements present in module are performed at least once by exercising every independent path. Every unit Is tested in order to verify if it will fall in any kind situation. Testing was held while programming is done and every unit has to be working in acceptable way during the end of testing phase, as regard to expected output from the module.

Unit testing for DHT22 sensor: Indicates UTC for DHT22 sensor with sample input of room temperature, humidity and resulting output as analog signals to Aurdino Maker WIFI 1010 and the result shows test performed stand successfully.

Unit testing for Arduino Maker WIFI 1010: Indicates UTC for Aurdino Maker WIFI 1010 with sample input of analog signals from sensor and resulting output monitor and the result shows test performed stand successfully.

Integration Testing

Around an interface the data could be lost: a module could have a contrary consequence on another sub function when they are aggregated the required measured function may not be produced global data structure issues are existing. Carrying out testing in order to revel the errors related to interfaces and at the same time building program structures symmetric is used that is nothing about integration testing. During the testing step every module are aggregated. Then the whole program was tested entirely.

Integration testing for high temperature detection

integration test case for detection of high temperature with sample input as room temperature detection and resulting output should be LED light blinking result shows test performed stand successful.

System Testing

After integration testing, the software is gathered entirely as a package where the error of interface have been brought out and correlated the final series of software tests, validation test begin. Validation test is accomplished when the

software functions in a manner that can be reasonably expected by the customer. The validation test is attained whenever the software serves in a way that can be sensibly required by the client. Here the system was tested against system requirements specification. System testing was actually a chain of different test whose main function is to fully utilize the computer system. Even though every test has an alternate aim all work to confirm that every system element is integrated in a right manner and function are allocated in right way.

CONCLUSION

We have developed a circuit that triggers the temperature system when maximum temperature is detected. The circuit mainly uses the **DTH 22 sensor** to detect temperature. The sensor has excellent sensitivity combined with the quick response time. This low signal is monitored by the microcontroller and sends the signal to the system to regulate fan by writing suitable code.

REFERENCES

<https://www.arduino.cc/en/Main/Software>

www.ti.com/product/LM35

www.learningaboutelectronics.com