Abstract

The embedded system has a close and indivisible relationship with environmental science aesthetics. This paper deals with a simple, easy to install, microcontroller-based circuit to monitor and record the value of temperature, water level, and sunlight of the natural environment that are continuously modified and controlled in order to optimize them to achieve maximum plant growth and yield. The controller communicates with the various sensor modules in real-time in order to control the light, aeration, and drainage process efficiently inside a greenhouse by actuating a cooler, dripper, and lights respectively according to the necessary condition of the crops. The environment is an important resource that guides the environmental development of the embedded system, and an important guarantee that makes the embedded system develop and innovate along a healthy, scientific environmental road. This makes the proposed system to be an economical, portable, and a low maintenance solution for greenhouse applications, especially in rural areas and for small scale agriculturists.

1. Introduction

The greenhouse effect on Earth. Some incoming sunlight is reflected by the Earth's atmosphere and surface, but most are absorbed by the surface, which is warmed. Infrared (IR) radiation is then emitted from the surface. Some IR radiation escapes to space, but some are absorbed by the atmosphere's greenhouse gases (especially water vapor, carbon dioxide, and methane) and reradiated in all directions, some to space and some back toward the surface, where it further warms the surface and the lower atmosphere.
Human activities are changing Earth's natural greenhouse effect. Burning fossil fuels like coal and oil put more carbon dioxide into our atmosphere.

NASA has observed increases in the amount of carbon dioxide and some other greenhouse gases in our atmosphere. Too much of these greenhouse gases can cause Earth's atmosphere to trap more and more heat. This causes Earth to warm up.

1.1 Description

A greenhouse is a structure that is built of walls and a transparent roof and is designed to maintain regulated climatic conditions. These structures are used for the cultivation of plants, fruits, and vegetables which require a particular level of sunlight, temperature, humidity, and soil moisture. IoT and Arduino based Greenhouse Environment Monitoring and Controlling Project is designed to maintain these conditions in the greenhouse.

1.1 Background

Just like a glass greenhouse, Earth's greenhouse is also full of plants! Plants can help to balance the greenhouse effect on Earth. All plants from giant trees to tiny phytoplankton in the ocean take in carbon dioxide and give off oxygen.

The ocean also absorbs a lot of excess carbon dioxide in the air. Unfortunately, the increased carbon dioxide in the ocean changes the water, making it more acidic. This is called ocean acidification.

More acidic water can be harmful to many ocean creatures, such as certain shellfish and coral. Warming oceans from too many greenhouse gases in the atmosphere can also be harmful to these organisms. Warmer waters are the main cause of coral bleaching.

1.2Motivation

- This system helps in monitoring and controlling the climatic conditions that are favorable for the cultivation of a particular plant. By using this system, crop growth can be improved along with maximized yield, irrespective of the weather conditions.

- This project can be further enhanced to monitor and control the pesticide level.

2. Literature Survey

The greenhouse is widely used to shield crops from excess cold or heat, raining, and unwanted pests. A greenhouse makes it possible to grow certain types of crops year-round and fruits, vegetables, and flowers are what a greenhouse most commonly grows. Good design and advanced technology applied in the greenhouse industries help farmers to protect their crops suffered from an attack by pests including viruliferous whiteflies. Several types of greenhouse designs that match the climate requirement in the existing greenhouse commercial design which is gable, gambrel, skillion, raised arch, sawtooth, and others. A greenhouse is essentially meant to permit at least a partial control of microclimate within it. The control of the greenhouse environment means the control of extreme heat and humidity, soil moisture, nutrients, and significant inputs of heat and light may be required. The maximum greenhouse temperatures should not exceed 30-35°C. High-temperature exposure in the greenhouse during sunny days may harm the crops inside that cause crop stress and burn effects. Therefore, cooling technologies play an important role in the greenhouse industry. Same as the diseases that spread out through the wind affecting crop growth, quality, and productivity. The heat trapped inside the greenhouse influences the increase of temperature and it can be minimized by the greenhouse design and technology applied.

Greenhouse farmers cannot precisely detect the level of humidity inside the greenhouse. They only know the condition inside the greenhouse manually and feel it by themselves. Ultimately, experiences play a bigger part in their daily operations. If the condition is too dry, they will give water to the plants or soil, but if it is too humid, they will open the rooftop of the greenhouse, especially in the daylight. In designing such a system, there is a limitation to problems, to see how far this system can do its tasks. This limitation according to the situation where this system will be used later. The limitation is the system can detect the humidity of air in the building. When the humidity sensor reaches a certain threshold, the humidity sensor will send a signal to the microcontroller which will then process the signal, to be sent into the connected computer. Computer functions as a server that connects to the android platform. There are 3 kinds of activities that are designed in the system. First, monitor the humidity level in the greenhouse. Secondly, if the greenhouse is too dry, the water sprayer can be activated, increasing the humidity level. It also can deactivate the water sprayer. Third, if the greenhouse is too humid, the rooftop can be opened to lower the humidity level. The third function can be used to open or close the rooftop based on the needs.

3. Proposed Work

3.1 Problem Statement

To design and implement an IOT based Greenhouse Monitoring System. The detection system also includes an alarm and also an LCD panel to alert the users. The system is implemented as an Arduino based gas detector alarm, temperature detector, light detector, and water level detector.

3.2 Features

- Alerting Alarm System
- Keep an eye on every KPI of a greenhouse.
- Get real-time updates anywhere anytime and take actions fast.
- Go for plant-centric lighting for maximum production.
- Protect plants from extreme weather fluctuations.
- Reliability, usability, portable

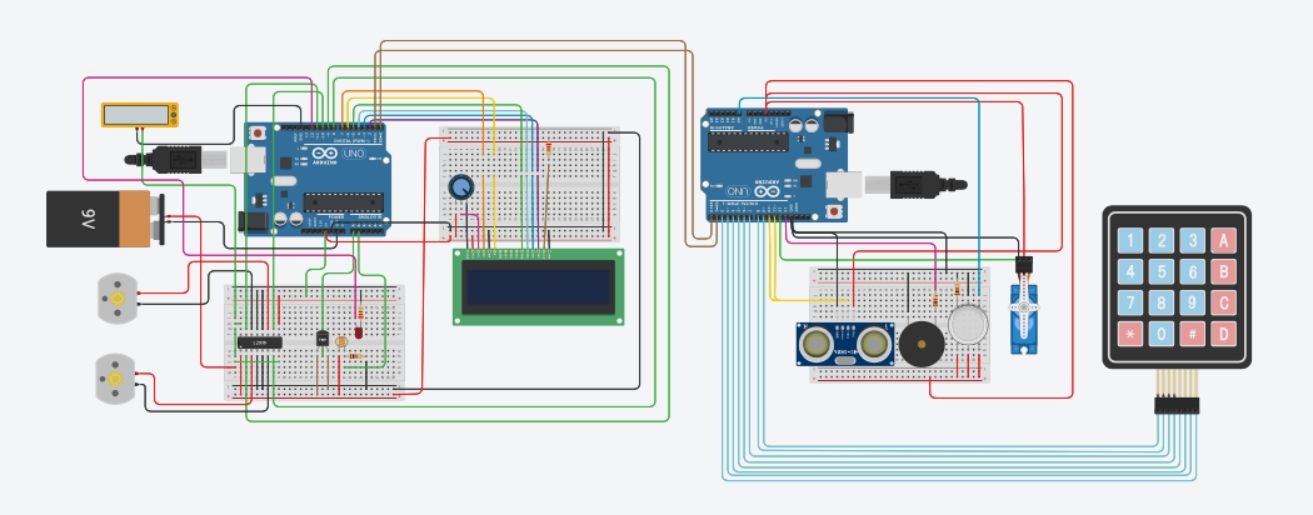
3.3 Objectives

- The main objective of this project is to automatically control the system in the greenhouse using a temperature sensor, humidity sensor, light sensor, and gas sensor.
- An additional consideration is the toxicity hazard created by a leak of gas with both toxic and combustible properties.
- The objective of the proposed Greenhouse Monitoring System is to provide a solution by designing an automatic system that can detect the leakage of gas, temperature of the atmosphere, light in the surrounding and the water level.

3.4 Scope

- It can be used in commercial greenhouses.
- This project defines the minimum mandatory requirements considering the overall environment of a greenhouse.
- Can be installed and put to use in every greenhouse.
- Alert people who are nearby thus ensuring that the user needs not always be present on the site.

3.5 System Architecture:



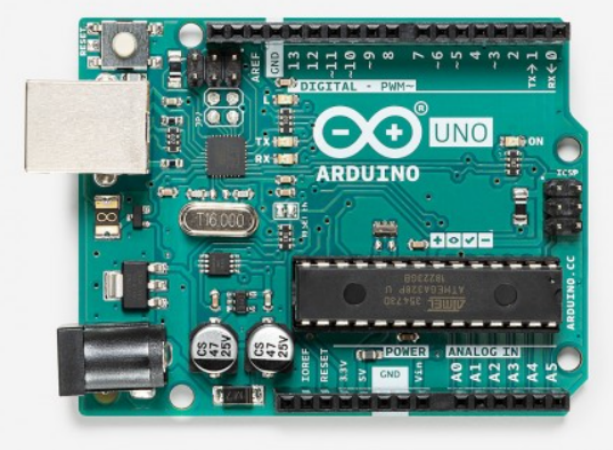
3.6 Project Design

3.6.1 Components Required:

- Arduino UNO R3
- Gas Sensor Module
- 220Ω Resistor
- 1 kΩ Resistor
- 250 kΩ Potentiometer
- Photoresistor
- Temperature Sensor
- DC Motor
- H-bridge Motor Driver
- Voltage Multimeter
- Ultrasonic Distance Sensor
- Micro Servo
- Red LED
- Keypad 4x4
- Piezo Buzzer
- 16X2 LCD
- Breadboard
- 9 Volt Battery
- Connecting wires

3.6.2 Components Description:

1. **Arduino UNO R3:** Arduino Uno is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), 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 an AC-to-DC adapter or battery to get started.



Technical Specifications:

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)

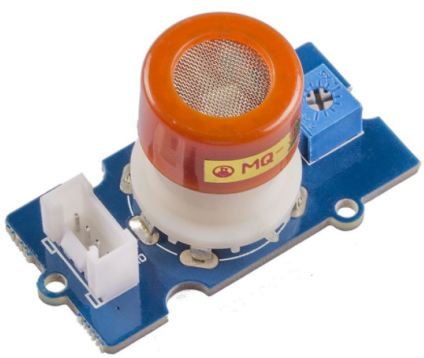
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

2. Smoke Sensor Module (MQ3) :

This module contains an MQ3 sensor that actually detects LPG gas, a comparator (LM393) for comparing MQ3 output voltage with a reference voltage. It gives a HIGH output when smoke is sensed. A potentiometer is also used for controlling the sensitivity of gas sensing. This module is very easy to interface with microcontrollers and Arduino and easily available in the market by the name "Smoke Sensor Module". We can also build it by using LM358 or LM393 and MQ3.

Features

- Sensor Type - Semiconductor
- Easy SIP header interface
- Compatible with most of the microcontrollers
- Low-power standby mode
- Requires heater voltage
- Good sensitivity to alcohol gas
- Fast response and High sensitivity
- Long life and low cost
- Requires simple Drive circuit



MQ-3 Sensor Module

Pin Name	Description
VCC	This pin powers the module, typically the operating voltage is +5V
GND	Used to connect the module to system ground
Digital Out (DO)	You can also use this sensor to get digital output from this pin, by setting a threshold value using the potentiometer
Analog Out (AO)	This pin outputs 0-5V analog voltage based on the intensity of the gas

MQ 3 Sensor Module

3. Piezo Buzzer

Piezo buzzers are simple devices that can generate basic beeps and tones. They work by using a piezo crystal, a special material that changes shape when a voltage is applied to it. In simplest terms, a piezo buzzer is a type of electronic device that's used to produce a tone, alarm, or sound. It's lightweight with simple construction, and it's typically a low-cost product.

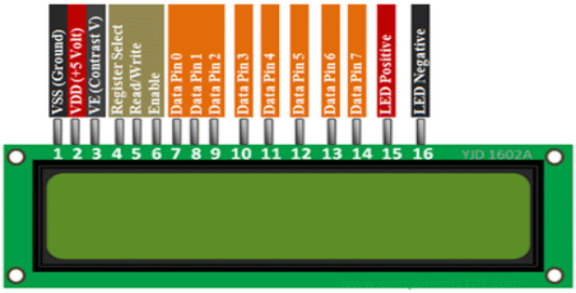
Features

- Rated Voltage: 6V DC.
- Operating Voltage: 4-8V DC.
- Rated current: <30mA.
- Sound Type: Continuous Beep.
- Resonant Frequency: ~2300 Hz.
- Small and neat sealed package. Breadboard and Perf board friendly

4. LCD PANEL

Features

- 16×2 LCD display Operating Voltage is 4.7V to 5.3V
- Current consumption is 1mA without a backlight Alphanumeric LCD display module, meaning it can display the alphabets and numbers.
- Consists of two rows and each row can print 16 characters.
- Each character is built by a 5×8 pixel box Can work on both 8-bit and 4-bit mode.
- It can also display any custom generated characters Available in Green and Blue Backlight.

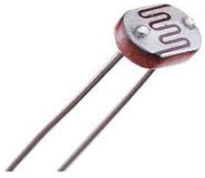


5. Photoresistor

A photoresistor (acronymed LDR for Light Decreasing Resistance, or light-dependent resistor, or photo-conductive cell) is a passive component that decreases resistance with respect to receiving luminosity (light) on the component's sensitive surface.

Features

Photoresistor LDR's are light-dependent devices whose resistance is decreased when light falls on them and that is increased in the dark. When a light dependent resistor is kept in dark, its resistance is very high. This resistance is called dark resistance.



6. Temperature Sensor

A temperature sensor is an electronic device that measures the temperature of its environment and converts the input data into electronic data to record monitor, or signal temperature changes. There are many different types of temperature sensors

Features

- TMP36 is specified from −40°C to +125°C.
- Provides a 750 mV output at 25°C.
- Operates to 125°C from a single 2.7 V supply.
- The TMP36 is functionally compatible with the LM50.
- TMP36 has an output scale factor of 10 mV/°C.

7. DC Motor

A DC motor is any of a class of rotary electrical motors that converts direct current electrical energy into mechanical energy. The most common types rely on the forces produced by magnetic fields. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the direction of current in a part of the motor.

Features

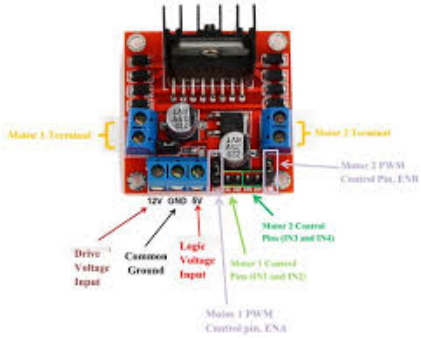
- Runs on DC power or AC line voltage with a rectifier.
- Operating speeds of 1,000 to 5,000 rpm.
- 60-75% efficiency rate.
- High starting torque.
- Low no-load speeds.

8. H-Bridge Motor Driver

An H-bridge is an electronic circuit that switches the polarity of a voltage applied to a load. These circuits are often used in robotics and other applications to allow DC motors to run forwards or backward.

Features

- Popular L298N Dual H-Bridge Motor Driver chip.
- Drives motors from 5-35V at up to 2A per channel.
- Provides 4 LEDs that reflect the state of the control logic.
- Independent direction, speed, and braking for each motor.
- Screw terminals for easy connections to motors and power.



9. Voltage Multimeter

A multimeter or a multitester, also known as a VOM (volt-ohm-milliammeter), is an electronic measuring instrument that combines several measurement functions in one unit. A typical multimeter can measure voltage, current, and resistance. Analog multimeters use a microammeter with a moving pointer to display readings.

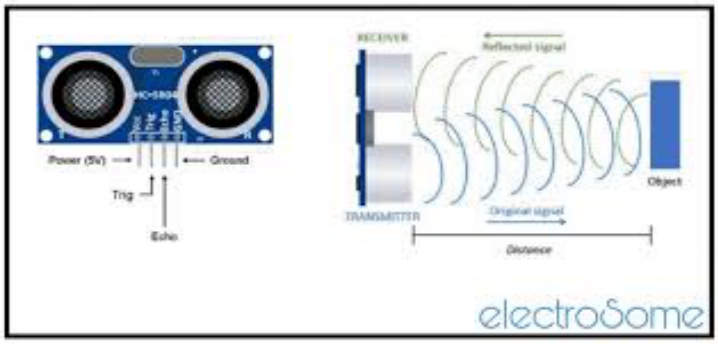
Features

- The ability to select a range of voltages.
- The ability to measure AC and DC voltages.
- The ability to measure ohms.

10. Ultrasonic Distance Sensor

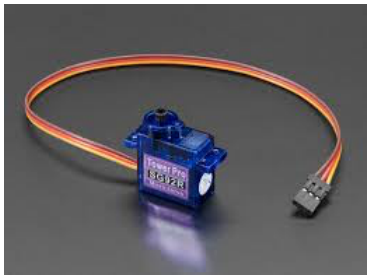
Ultrasonic sensors work by emitting sound waves at a frequency too high for humans to hear. They then wait for the sound to be reflected back, calculating distance based on the time required. This is similar to how radar measures the time it takes a radio wave to return after hitting an object.

Features



- Supply current: 15mA.
- Supply voltage: 5V (DC).
- Modulation frequency: 40Hz.
- Output: 0 – 5V (Output high when obstacle detected in range).
- Beam Angle: Max 15 degrees.
- Distance: 2cm – 400cm.
- Accuracy: 0.3cm.
- Communication: Positive TTL pulse.
-

11. Micro Servo



The tiny little servo can rotate approximately 180 degrees (90 in each direction) and works just like the standard kinds you're used to but smaller. You can use

any servo code, hardware, or library to control these servos.

Features

- Weight: 9g / 0.32oz
- Voltage: 4.8V nominal (3V to 6V DC)
- Torque: 2.5 kg-cm

12.Keypad 4x4

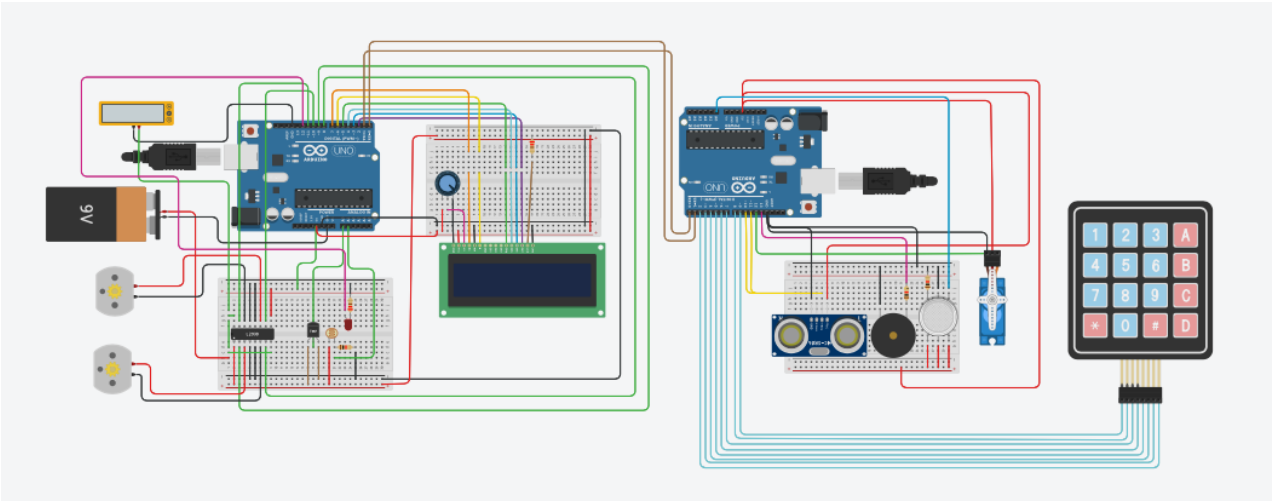
Keypad 4x4 is used for loading numerics into the microcontroller. It consists of 16 buttons arranged in a form of an array containing four lines and four columns. It is connected to the development system by a regular IDC 10 female connector plugged in some development system's port.

Features

- Maximum Voltage across EACH SEGMENT or BUTTON: 24V.
- Maximum Current through EACH SEGMENT or BUTTON: 30mA.
- Maximum operating temperature: 0°C to + 50°C.
- Ultra-thin design.
- Adhesive backing.
- Easy interface.
- Long life.

All other components are the standard components used in Electronics and IoT.

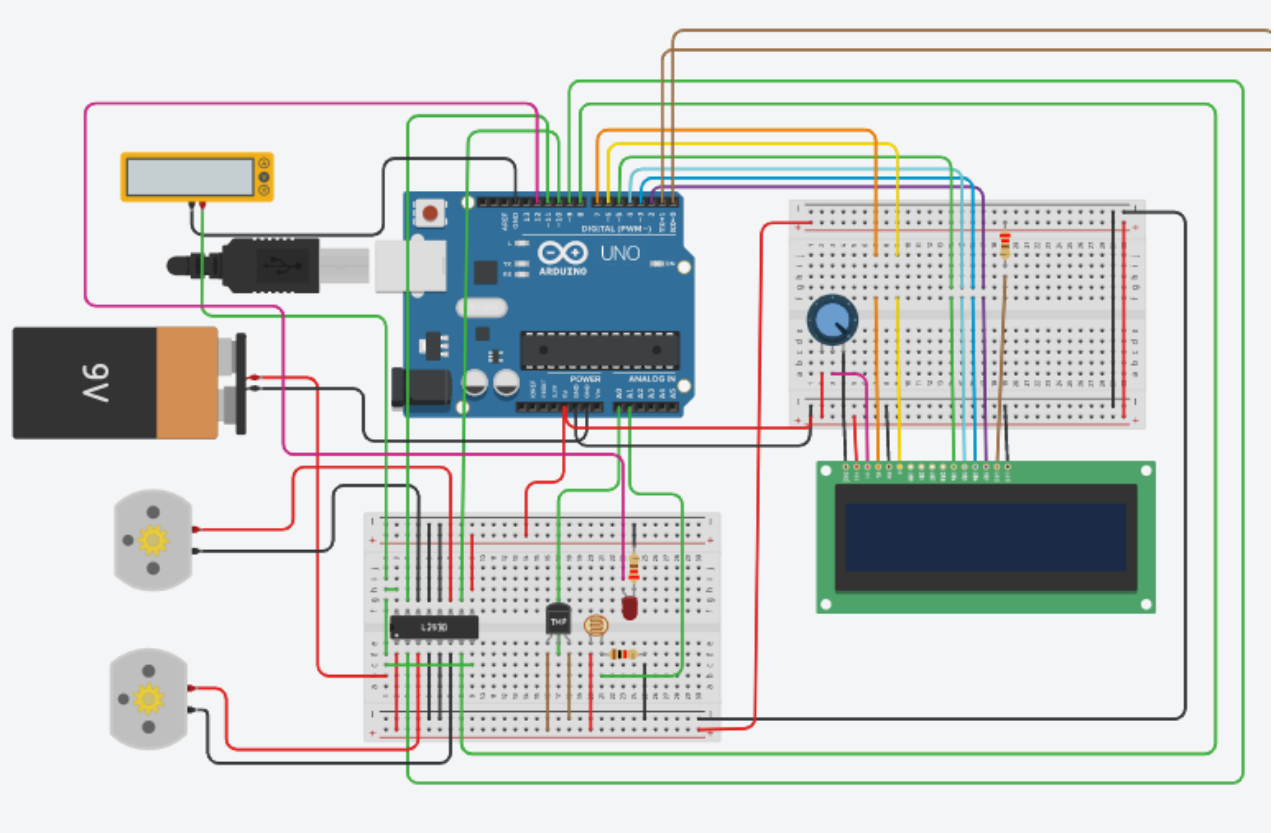
3.7Circuit Schematic



3.7.1 Working

Working of Arduino 1

Components connected to the Arduino 1 are a 9V battery for the 2x DC Motors, IC L293D, Temperature sensor, Photo-resistor, LED, Multimeter, LCD Display, and Potentiometer. This Arduino handles Temperature, Light and displays the overall status of the laboratory using the LCD with the help of a seamless connection between the Arduino 2. When working the Temperature Sensor uses the data and decoded it to get the current room temperature; if the temperature < 20 the room is “TOO COLD” then the Motor 1 (Heating Component) get triggered and if the temperature > 27 which implies that the room is “TOO HOT” then Motor 2 (Cooling Component) get triggered. If the temperature is between [20, 27] the room is at the “PERFECT” condition. The light is a necessity to any greenhouse lab, Photo-resistor is used to detect the current light in the room; if light intensity < 100 denotes “TOO DARK” and the LED bulb glows inside the room to be lit and if the light intensity > 180 represents “TOO BRIGHT” and turn off the LED bulb, “PERFECT” condition of light is when density is between [100, 180]. A Multimeter is connected to check the current-voltage usage of the system and a Potentiometer is used to adjust the LCD display brightness according to the user.



Code for Arduino 1

Text

1 (Arduino Uno R3)

```
// This arduino in general is responsible for air quality assessm
#include <LiquidCrystal.h>  // This is to properly utilize LCD Di

// These are used to connect to LCD screen
int regSel = 7; // This will select the register for LCD signalli
int enable = 6; // This will enable the LCD in general
int sig1 = 5; // Signal 1
int sig2 = 4; // Signal 2
int sig3 = 3; // Signal 3
int sig4 = 2; // Signal 4

// These are used to run our heating and cooling elements
int motor1 = 8; // Control clockiwise heater
int motor2 = 9; // Control anticlockwise heater
int motor3 = 10; // Control clockwise cooler
int motor4 = 11; // Control anticlockwise cooler

// This will control the artificial light
int light = 12; // led pin

// These are actual sensors used to detect
char temp = A0; // temperature sensor
char lumos = A1; // photoresistor

// Static declarations for our own use
float temperature; // will hold temp in celcius
float intensity; // will hold light intensity in lu
int flag = 0; // will determine welcome message

// Initialize a object of LCD
LiquidCrystal lcd(regSel, enable, sig1, sig2, sig3, sig4);

// Function declarations
float readTemperature(char); // determines temperature
float readLuminousity(char); // determines lighting
void authenticate(); // whether to allow functions

void setup() {
    // Pins  to which we will write data
    pinMode(motor1, OUTPUT);
    pinMode(motor2, OUTPUT);
    pinMode(motor3, OUTPUT);
    pinMode(motor4, OUTPUT);
    pinMode(light, OUTPUT);

    // Pins from which we will read stuff
    pinMode(temp, INPUT);
    pinMode(lumos, INPUT);

    // Serial monitor to debug stuff
    Serial.begin(1156200); // baud rate of 1156200 for comm
    // LCD to actuall log stuff;
    Serial.flush();
    lcd.begin(16, 2); // 16*2 grid of lcd
}

void loop() {
    lcd.clear();
    temperature = readTemperature(temp); // reads temp
    lcd.setCursor(3, 0); // decides lcd position
    lcd.print("Temp: "); // prints supplied string
    lcd.setCursor(9, 0);
    lcd.print(temperature);
    if (temperature < 20) { // For a tropical greenhouse below 20
        lcd.setCursor(4, 1);
        lcd.print("TOO COLD");
        digitalWrite(8, HIGH); // enables heater
        digitalWrite(10, LOW); // disables cooler
```

```
    } else if (temperature > 27) { // Above 27 is hot for a tempe
        lcd.setCursor(4, 1);
        lcd.print("TOO HOT");
        digitalWrite(8, LOW); // disables heater
        digitalWrite(10, HIGH); // enables cooler
    } else {
        lcd.setCursor(4, 1);
        lcd.print("PERFECT");
        digitalWrite(8, LOW); // disables both
        digitalWrite(10, LOW);
    }
    digitalWrite(9, LOW); // prevents anticlockwise motion
    digitalWrite(11, LOW);
    delay(150);
    lcd.clear();
    intensity = readLuminousity(lumos); // reads light
    lcd.setCursor(1, 0);
    lcd.print("Intensity: ");
    lcd.setCursor(12, 0);
    lcd.print(intensity);
    if (intensity < 100) { // If it is too dark outside
        lcd.setCursor(4, 1);
        lcd.print("TOO DIMM");
        digitalWrite(light, HIGH); // enable lighting
    } else  if (intensity > 180) { // It is morning now
        lcd.setCursor(4, 1);
        lcd.print("TOO BRIGHT");
    } else {
        lcd.setCursor(4, 1);
        lcd.print("PERFECT");
    }
    delay(150);
    digitalWrite(light, LOW); // Light is kept at blinking state
}
```

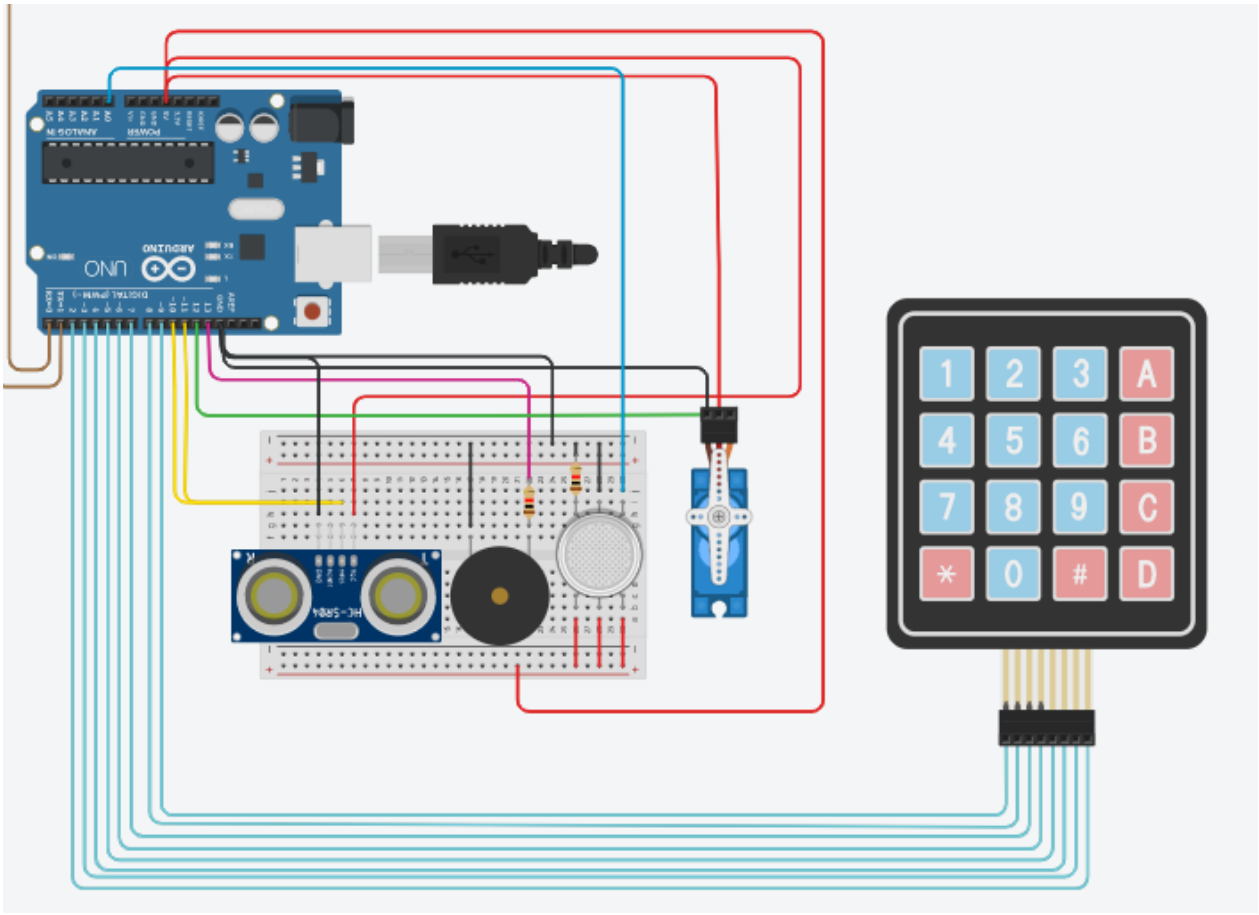
```

// This function reads temperature and converts to celcius
float readTemperature(char pin) {
    float volts = analogRead(pin); // this value is in terms of m
    float value = map(((volts - 20) * 3.04), 0, 1023, -40, 125);
    return value;
}

// This function reads light and converts to luminousity
float readLuminousity(char pin) {
    float lum = analogRead(pin); //  this value is likely b/w 0 -
    float value = map(lum, 0, 700, 0, 255); // this will be more
    return value;
}

/**************
* This function colud be used to unlock device post auth
* however serial comm is disabled
***************/
// This function will toggle flag
void authenticate() {
  lcd.setCursor(1,0);
  if (char(Serial.read()) == 'w')
  lcd.print('y');
  else
  lcd.print('n');

}

```

Serial Monitor

Working of Arduino 2

Components connected to the Arduino 2 are Ultrasonic Distance Sensor, Piezoelectric Buzzer, Gas Sensor, Servo Motor, Keypad 4x4. This unit encompasses controlling the system with Authentication via the Keypad with a secure access pin. This unit controls the water/humidity level inside the Lab and constantly monitors the Gas Level inside the Lab. If the Water Level > 200 then the water is pumped via the Servo, after pumping of excess water the pump returns to the original state. There is a Gas Sensor attached to monitor the gas density inside Lab, it starts a slow Buzzer if the gas density value > 100, to indicate the

current gas level repeatedly. Keypad Authentication is set for the access to the rightful owner using the access PIN.



Code for Arduino 2

Text | 2 (Arduino Uno R3)

```
// This arduino is mainly responsible for handling water levels a
#include <Servo.h>  // This will allow us to use servo motor for
#include <Keypad.h>// This will allow keypad based auth into devi

// These are general keypad configuration
byte rows = 4;
byte cols = 4; // 4*4 keypad setup
char keys[4][4] = { // These represent each key in keypad as 2D a
    {'1', '2', '3', 'A'},
    {'4', '5', '6', 'B'},
    {'7', '8', '9', 'C'},
    {'*', '0', '#', 'D'}
};
byte rowPins[4] = {9, 8, 7, 6}; // represents pins for map to row
byte colPins[4] = {5, 4, 3, 2}; // represents pins for map to col

// This represents an actual instance of keypad
Keypad kp = Keypad(makeKeymap(keys), rowPins, colPins, rows, cols

// This represents an instance of servo motor controllor
Servo svr;

// These represent various pins to be used by arduino
char gas = A0; // used to read gas sensor value
int echo = 10; // used to read distance sensor reading
int trig = 11; // used to send pulse
int serv = 12; // controls servo angle
int buzz = 13; // sends buzziness

// Static declarations
String password= "0000"; // contains the master key
String pressed=""; // contains the key entered

float distance; // stores distance reading from ultrasonic sensor
```

```
float gaser; // stores gassed value from gas sensor

int flag = 0; // trigger for authentication

// Function declarations
float measureDistance(int, int); // measures distance
float measureSmoke(int); // measures smoke
void checkKeyPad(); // makes auth mandatory
void setup() {
    Serial.begin(1156200); // serial monitor for communication  p
    Serial.flush();

    pinMode(trig, OUTPUT); // sends pulse
    pinMode(echo, INPUT); // reads echo signal
    pinMode(gas, INPUT); // reads gas value
    pinMode(buzz, OUTPUT); // sets piezo value

    svr.attach(serv); // Attaches instance of servo controller to
}

void loop()
{
    distance = 0.01723 * measureDistance(trig, echo); // 340 * 10
    Serial.print("Distance: ");
    Serial.print(distance);
    Serial.println();
    if (distance > 200) { // we only need to pump water if the di
        for (int i = 0; i <= 180; i++) {
            svr.write(i);
            delay(5); //delay of 10 ms to reach the position
        }
      delay(50);
        for (int i = 180; i >= 0; i--) {
            svr.write(i); // pump must return to its original pos
            delay(5);
        }
    } else {
        svr.write(0); // so that the pump returns to its original
    }

    gaser = measureSmoke(gas); // read gas reading
    Serial.print("Gas: ");
    Serial.print(gaser);
    Serial.println();
    if (gaser > 100) {
        for (int i = 200; i <= 600; i += 50) {
            tone(buzz, i, 10); // will allow toned buzzing
            delay(5);
        }
    } else {
        digitalWrite(buzz, 0); // stop buzz
    }
    svr.write(0);
    delay(20);
}

// Finds out distance travelled in cms
float measureDistance(int triggerPin, int echoPin) {
    digitalWrite(triggerPin, LOW);
    delayMicroseconds(2);
    // Sets the trigger pin to HIGH state for 10 microseconds
    digitalWrite(triggerPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(triggerPin, LOW);
    // Reads the echo pin, and returns the sound wave travel time
    return pulseIn(echoPin, HIGH);
}
```

```
// Finds out the quantity of smoke in air
float measureSmoke(int pin) {
    return analogRead(pin);
}

/*************
* This function served as authentication, however is
* non-functional due to no serial communication
**************/

// Listens for key press event
void checkKeyPad() {
    char key = kp.getKey();
    if (int(key) != 0) {
        if (key == 'C') { // This will allow us to clear our inpu
            pressed = "";
            delay(100); // To clear any cache
        } else if (key == '*') { // This will allow us to submit
          if (pressed == password) {
                flag = 1; // will exit check loop
          }
          Serial.write(char('w'));
        } else {
            pressed.concat(key);  // store entered stuff
            Serial.write(char(key));
        }
    }
}
```

Serial Monitor

3.8 System Architecture

- The gas sensor (B1 pin) is connected to the A0 pin of the second Arduino.
- The +ve of the piezo buzzer is connected to D13 of the second Arduino.
- The enable pin of the LCD is connected to D6 of the first Arduino.
- The register select pin of the LCD is connected to D7 of the first Arduino.
- The read /write pin of the LCD is grounded.
- The contrast pin of the LCD is connected to the wiper pin of the potentiometer
- The trigger pins of the ultrasonic distance sensor are connected to D10 and D11 of the first Arduino.
- The signal pin of the micro servo is connected to D12 of the first Arduino.
- All the pins of the keypad are connected to the first Arduino through D2 to D9.
- The output pin of the temperature sensor is connected to the A0 pin of the first Arduino.
- The anode of the LED is connected to D12 of the first Arduino.
- The second terminal of the photoresistor is connected to the A1 pin of the first Arduino.

- The D0 of the first Arduino is connected to the D1of the second Arduino and vice versa.
- The DB4, DB5, DB6, and DB7 of LCD are connected to the D5, D4, D3, and D2 of the first Arduino respectively.
- The input pins 1,2,3 and 4 of the H-Bridge Motor Driver are connected to the D9, D8, D10, and D11 of the first Arduino respectively.

4 Conclusion and Future Work

A step-by-step approach in designing the microcontroller based system for measurement and

control of the four essential parameters for plant growth, i.e. temperature, water level, and light intensity, has been followed. The continuously decreasing costs of hardware and software, the wider acceptance of electronic systems in agriculture, and an emerging agricultural control system industry in several areas of agricultural production, will result in reliable control systems that will address several aspects of quality and quantity of production. Further improvements will be made as less expensive and more reliable sensors are developed for use in agricultural production. This study has successfully developed a prototype of an automated greenhouse that consists of a new design of greenhouse, greenhouse cooling, and monitoring system. G.I pipe is the recommended material that meets the greenhouse requirement include, including the durability, strength, anti-rust, material and fitting availability, outdoor purpose and the cost of the material. The control system has automatically controlled the cooling and water features such as the misting unit, roof cooling, watering, ventilation, and ceiling fan. The control system actuation or execution is based on the condition of the greenhouse that depends on the temperature inside the greenhouse. The monitoring system will preview the information from the sensors

5 References

[1] Arduino., "what is Arduino ?", Arduino Guide Introduction. Available [Accessed : 14 Oktober 2012]: http://arduino. cc/en/Guide/Introduction
[2] Instructables., "Visualize Humidity with the SHT 11 Sensor". Available [Accessed: 20 December 2012]: http://www. instructables. com /id/VISUALIZE-humiditywith-The-SHT11sensor/step2/Wire-The-sensor
[3] Instructables.,"Visualize Humidity with the SHT 11 Sensor". Available [Accessed: 20 December 2012]: http://www. instructables. com/id/VISUALIZE-humiditywith-The-SHT11sensor/#step1
[4] Parallax.,"Sensirion SHT11 Sensor Module". Available [Accessed: 22 December 2012]: http://www. parallax. com/Portals/0/Downloads/docs/prod/acc/ SensirionDocs. pdf
[5] Sensirion The Sensor Company., "Datasheet SHT 1x (SHT 10, SHT 11, SHT 15)". Available [Accessed: 23 December 2012]: http://www. sensirion. com. cn/down/downimg/DatasheetSHT1x%20V5. pdf
[6] IEEE Standards, "Get IEEE 802® : Local and Metropolitan Area Network Standards". Availabe [Accessed: 22 Desember 2012]: http://standards. ieee. org/ getieee802/download/802. 11g-2003. pdf
[7] Mittal, M., Tripathi, G., "Green House Monitor and Control Using Wireless System Network", VSRDIJEECE, Vol. 2 (6), 2012, 337-345, 2012.
[8] Ai, Q., Chen, C., "Green House Environment Monitor Technology Implementation Based on Android Mobile Platform", IEEE Conference Publications. Page(s): 5584-5587, 2011.