VISVESVARAYA TECHNOLOGICAL UNIVERSITY
Jnana Sangama, Machhe, Belgaum – 590 018

A Project Report on

Small Scale IoT Enabled Automated Greenhouse

Submitted in partial fulfillment for the award of the degree of

Bachelor of Engineering
In
Electronics and Communication Engineering

Submitted by

Kiran C Ranebennur (1PI14EC026)

Anish Arun Nesarkar (1PI14EC005)

Muheen Basha F (1PI14EC038)

Under the Guidance of

Dr. Venkatesh Vadde
Department of ECE
PESIT, Bengaluru – 85

Department of Electronics and Communication Engineering
PES Institute of Technology,
100 Feet Ring Road, BSK III Stage, Bengaluru - 560085

2017-2018

PES INSTITUTE OF TECHNOLOGY

100 Feet Ring Road, BSK III Stage, Bangalore - 560085

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the project work titled **Small Scale IoT Enabled Automated Greenhouse** is a bona fide work carried out by **Anish Arun Nesarkar, Kiran C Ranebennur and Muheen Basha F** bearing respective University seat numbers **1PI14EC005, 1PI14EC026, 1PI14EC038**, in partial fulfillment for the award of Bachelor of Engineering in Electronics and Communication Engineering of the Visvesvaraya Technological University, Belgaum during the year 2017-2018. It is certified that all the corrections/ suggestions indicated for internal Assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed by the university, for the award of the said degree.

Guide: Venkatesh 8.5.2018

Dr.Venkatesh Vadde
Dept. of ECE,
PESIT, Bengaluru-85

Head of Department: 8/5/18

Prof. Anuradha M
Dept. of ECE,
PESIT, Bengaluru-85

Prof. Anuradha M
HOD - ECE Department, PESIT
Bangalore - 560 085.

Principal:

Dr. K.S. Sridhar
PESIT
Bengaluru-85

8.5.18

Principal
PES Institute of Technology
100 Feet Ring Road, BSK III Stage
Bangalore-560085

External Viva

Name of Examiners	Signature with Date
1. Dr. Srinath VR	10/5/18
2. Dr. J. Manikandan	10/5/18
3.	

DECLARATION

We, Anish Arun Nesarkar, Kiran C Ranebennur, Muheen Basha F, hereby declare that the dissertation titled, **Small Scale IoT Enabled Automated Greenhouse**, is an original work done by us under the guidance of Dr. Venkatesh Vadde, Department of Electronics and Communication Engineering, PES Institute of Technology and is being submitted in partial fulfillment for the award of **Bachelor of Engineering** in **Electronics and Communication Engineering from Visvesvaraya Technological University, Belgaum** during the academic year 2017-2018. The matter contained in this report has not been submitted to any other university or institution for the award of any degree or diploma.

PLACE: BANGALORE
DATE: 08/05/2018

NAME AND SIGNATURE OF THE CANDIDATE

ANISH ARUN NESARKAR (1PI14EC005)

KIRAN C RANEBENNUR (1PI14EC026)

MUHEEN BASHA F (1PI14EC038)

Abstract

A greenhouse is a structure that provides an environment to grow plants all year round, even on extremely hot , cold and cloudy days. However, extreme environmental factors inside the greenhouse such as high temperatures and low humidity can negatively impact the plants. Consequently, controlling this environment is essential in order for the plants to grow strong and healthy.

The main goal of this project is to design and build an small scale automated greenhouse that can maintain and control the microclimatic environment, by acting upon live sensor readings and is enabled to provide the status of the system to the owner.

The project was split mainly into two parts: The first part consisted of designing and constructing a greenhouse. The second part was to automate the greenhouse with the help of microcontrollers and sensors. The microcontroller/processor(s) used to automate the greenhouse is a Raspberry Pi and an Arduino Mega in a Master-Slave configuration. The sensors sense the climatic parameters like Air temperature, Relative Air humidity and Soil moisture inside the structure. Appropriately the actuators like a set of inlet and exhaust fans and water pumps are triggered in order to maintain the parameters as close to the preset conditions. The cooling fans are used to reduce the temperature and the pumps are used to water the soil beds and hence increase soil moisture.

The greenhouse design is made modular by using adjustable windows and Shade nets for passive ventilation to curtail the extremely high temperature at certain peak hours during summer. Thus the temperature could be controlled to be maintained in the desired range. The data logs collected by the System are then pushed to the user in a timely manner via an email or a custom UI interface.

Acknowledgement

Education along with the process of gaining knowledge and stronghold of subject is a continuous and ongoing process. It is an appropriate blend of mindset, gained skills, experience and knowledge gained from various resources. This project would not have been possible without the support of many people.

Our most sincere and grateful acknowledgement to P.E.S Institute of Technology for providing us with the opportunity to pursue our degree and thus helping us in Shaping our career.

We convey our deep gratitude to our guide, **Dr. Venkatesh Vadde**, Professor, Department of Electronics and Communication, PES Institute of Technology for his elevating inspiration, kind supervision and motivation towards completion of the project. We also, acknowledge constructive comments and uplifting spirit by our panel members during the project presentation.

We thank profusely the PES facility for providing a cooperation and resource to complete this endeavor successfully.

We express our profound thanks to **Dr. K.N.B. Murthy**, Vice-Chancellor, PES Institute of Technology for providing us with an open plot in the college premises to completely execute our project.

We take immense pleasure in thanking, **Prof. Anuradha M**, Head of the Department, Dept. of ECE, PESIT for her constant support and guidance.

We express our warm thanks to **Dr. Srivatsa**, Professor, Dept. of EEE, PESIT for his constant friendly advice during the project work.

We would also like to thank all the teaching and non-teaching staff and the management of PES Institute of Technology for providing us timely help and lab support throughout the course of the project.

Table of Contents

Chapters

List of Figures

List of tables

Chapter 1

Introduction

In this chapter we give an introduction to the concept of a greenhouse and list out the different types of greenhouses currently being used. The next section explains about the existing automated greenhouse systems. Following this section, Literature review for the existing greenhouses and the materials used for building an automated greenhouse is explained. After having a detailed literature review for our project, the motivation to build a greenhouse is explained in the next section. A brief outline of the report is explained in the last section.

1.1 Intro to Greenhouses:

A **greenhouse** is a structure which is typically covered with transparent/translucent materials that protects the plants from external climate conditions, which creates an optimal growth environment, and also offers great solution for sustainable and efficient year around cultivation. It is a structure with roof and walls which are made of transparent material, such as glass or polythene sheets(polyhouses) in which plants requiring a regulated climatic conditions are usually grown. Nowadays a greenhouse operates as a system, which are also referred to as controlled environment agriculture (CEA), controlled environment plant production system (CEPPS), or phytomation system.

1.2 Automated Greenhouses

A greenhouse needs to be manually attended for its proper operation as it can be extremely hot at noon and later starts cooling in the evening eventually losing all the heat. This requires human intervention for irrigation cycle and ventilation to be maintained at the required level at a particular time of the day.

These processes can be automated and a fair degree of control over various climatic parameters can be achieved by using passive and active elements in a closed system. These parameters form inputs to a set of control systems to activate their respective actuators.

Different systems provide control over different parameters which may or may not be dependent on each other.(Ex: Temp and humidity slightly dependent).An automated system eliminates human error and has very low error tolerance. These details can be then sent to the user/owner with necessary notifications and updates about the conditions inside the greenhouse and certain actions that are to be done by the user..

1.3 Literature Review:

Greenhouses are constructed in different shapes and are designed based on the geographical location, climatic zones, sun pattern ,wind direction, elevation and the surrounding environment.

Greenhouses are usually implied in a large business scale for commercial farming and there is no such open and easy platform for data analytics and other open source softwares to be used in such systems for a small scale greenhouse.

Greenhouses are particularly built on a large scale(generally in acres) with a lot of initial capital investment. As India is prone to fragmentation of agricultural land the current technology is not applicable here. So there is a need for a small scale version of a greenhouse setup to analyse the climatic control effects and arrive at results and extrapolate it to any scale applicable.

The highly pressurized and complex irrigation systems currently used in the commercial markets are not feasible and a simple drip irrigation system is sufficient for small scale structures.

Very faint literature is available on Small scale designs for DIY types of greenhouses that are easily adaptable in INDIA. Ex: Rooftop greenhouse, mushroom farming setup etc.

1.4 Motivation

The purpose of this project is to establish an automated greenhouse platform for sensor based horticulture and analytics. Such greenhouses can help in growing food under all kinds of climatic conditions. A greenhouse makes it possible to grow unseasonal, specific crops all year round. Additionally, automation removes human error and uncertainty, providing timely updates and efficiency in performance for growing crops. Such a setup can help people take control of growing their own healthy food with much less effort. With the help of the results we expect from the project, these concepts can also be extrapolated to a larger scale for controlled commercial farming. Some of the main problems that are to be addressed by this project are given below.

1. Seasonal crops cannot be cultivated throughout the year.
2. Uncertainty in weather that affects the yield of crops.
3. Manually operated Greenhouses require constant care for their operation which leads to increase in human labor cost and also prone to human error.
4. Variation in prices of the crops due to imbalance in supply and demand.
5. Extensive use of chemical pesticides which is the main cause to health related issues.

1.5 Goals & deliverables of Project

1. To design, fabricate and integrate a greenhouse structure installed with suitable sensors and actuators enabling climate control.

2. To successfully interface all the sensors and actuators to the microcontrollers, along with development of suitable software to collect valid data logs.

3. To develop a closed loop climate control algorithms to control the greenhouse climate and irrigate soil-beds using drip irrigation, and demonstrate the results with data logs.

4. To develop a WiFi/GSM based IoT communication mechanism for wireless updates on health and growth parameters of plants.

1.6 Outline of this Report

This report is intended to deliver a complete guide on how to build a low cost small scale greenhouse. It talks about the processes that went into building a low cost automated IOT enabled greenhouse. The report deals with all the different aspects of designing and constructing a greenhouse, selection of materials , control system algorithms , Raspberry Pi configurations in detail. It presents a detailed documentation of a scaled prototype of a climate monitoring system with its hardware setup of Microprocessor/microcontrollers , sensors and actuators and also with its system software which drives the hardware with an IOT component embedded into the system.

The report also presents various results we achieved in our experiments for various defined scenarios of climatic control parameter variations in different combinations and appropriate conclusions are drawn. Different case studies for a few specific crops are discussed and their economic feasibility with a basic estimated business plan is put forth.

This report is intended to serve as a well-documented guide for a small greenhouse construction setup and serves as a template that can be definitely scaled up.

Chapter 2:

Greenhouse Design & Construction

This chapter gives a detailed information on the design and construction of the greenhouse structure used for the project. The details include the materials used and a step by step procedure for constructing a small scale greenhouse. It also sheds light on the problems encountered and the modifications that had to be performed to solve these problems. This chapter also explains about the positioning of sensors and actuators and the reasons behind their placements in the greenhouse. Following this section, climate control requirements are mentioned for the designed greenhouse.

2.1 Greenhouse design: Plans

The designs of a greenhouse depends on various factors land availability, and the size of the greenhouse. In this project we have designed plans for a given trapezoidal shaped plot with longer side of 11ft ,shorter side of 5ft and a length of 20ft.
We have used a free online 3D modelling tool called SketchUp. This software provides a great virtual environment, a friendly UI and just enough tools to design and visualize the greenhouse structure.

The plot is facing in the north direction with a rainwater canal in the west at about a distance of 20 ft, a small store room in the south and the other 2 directions are open air spaces.
Two different designs were developed
1) Trapezoidal shaped structure 2) Rectangular shaped structure designs.

The finalized design of greenhouse is of a rectangular(cuboidal) shape with a slightly curled sloping roof for easy passage of rain water to the side. The higher side wall is of height 8 ft and the lower wall height is 6ft.

Fig. 2.1 Site before Greenhouse construction

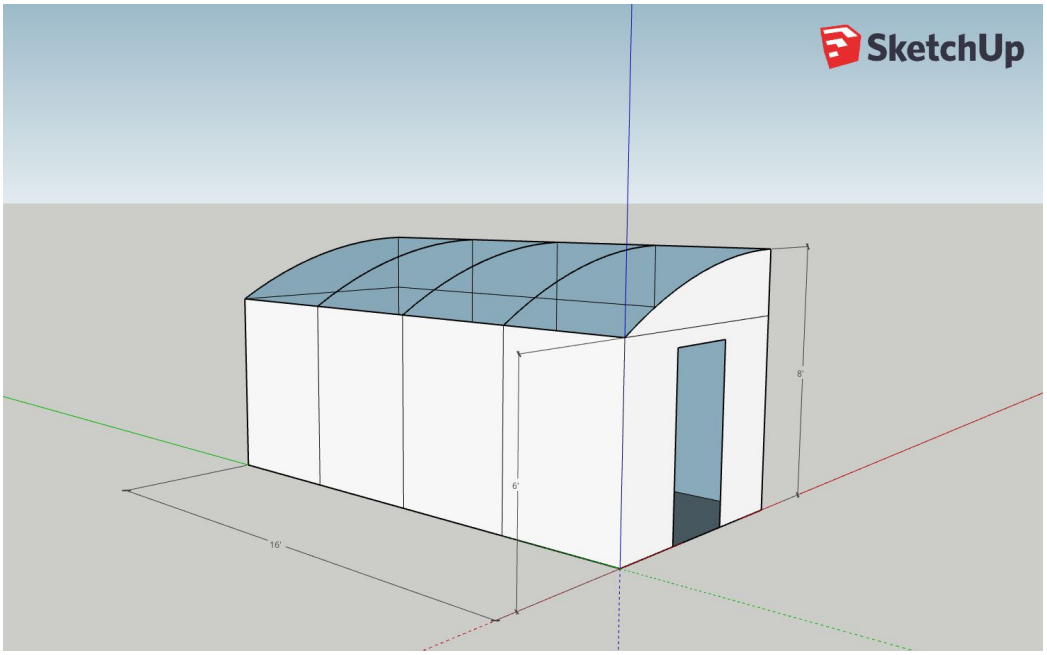


Fig. 2.2 SketchUp Design of Greenhouse

2.2 Greenhouse design: Materials used

1	PVC pipes(2',4',6')
2	Poly Sheet
3	Metal Frame
4	PVC Door
5	Shade Nets
6	Silicon Glue
7	Tapes
8	Concrete
9	Exhaust fans
10	Water Storage Tank
11	Soil beds

Table 2.1 Greenhouse materials used

2.3 Greenhouse Construction:

Construction is the process which extended more than 50% of the projects duration which will be explained in different stages. As mentioned above the design of the greenhouse was finalized.

The materials was procured by inquiring different vendors (online as well as local) to start the construction as soon as possible.

2.3.1 Stage 1: Preconstruction

Before the construction actually begins the following basic requirements had to be completed.

1) Water connection

Fig 2.3 : Water connection

2) Electricity connection

Fig 2.4 : Electricity connection from nearby Power grid

3) Levelling of ground and digging of appropriate pits for the foundation.

Fig 2.5 : Levelling of ground and digging appropriate holes for foundation

2.3.2 Stage 2: Foundation

For the foundation M15 grade concrete (in ratio cement: sand: jelly: water :: 1 : 4 : 4 : for semisolid) was used. An inverted bucket, with its bottom sliced, was used as a mould for the concrete pillar base to set. Followed by 3 days of curing, 6 inch PVC pipes of height 2 feet and 6 inches were used as the foundation for the pillars (They were placed 1 feet 6 inches below ground level and 1 foot above ground level).

Fig 2.6 : Foundation using PVC pipes

Fig 2.7 : Foundation horizontal view

2.3.3 Stage 3: Pillars

The pillars of 4 inch PVC pipes were installed vertically from ground level upto 7 feet on the Left side structure and 9 feet above the ground on the Right side structure of greenhouse. The cement is filled up upto 1 foot above the ground inside the 4 inch pipe. The ground is levelled for the further process. This is followed by another 3 days for curing of cement.

Fig 2.8 Erecting PVC pillars

2.3.4 Stage 4: Roof and Horizontal beams

Once the vertical pillars are set, the beams of 2 inch diameter PVC pipes were used for horizontal support and roof as a supporting structures for the poly-sheet. Using a drilling machine with cylindrical wood boring bit, the holes were carefully marked and cut for the vertical beams. Same technique was used for beams to be installed across the horizontal pipes.

Fig 2.9 : Installation of horizontal beams

2.3.5 Stage 5: Metal frame with Access door

A Mild Steel frame was fabricated from a nearby fabricator for the given design and it was installed to the front face of the greenhouse structure. A door made of UPVC was designed and fitted to access the greenhouse and secure it.

Fig 2.10 : PVC door with Metal frame

2.3.6 Stage 6: Installing of Poly-Sheet

A 70% transparent Polyethylene Sheet is installed by appropriate measurements and an offset of 1.5ft is kept at the bottom to package the sheet under the soil beds. The complete structure is covered with 3 pieces of polysheet, 1st covering the back face along with the right face ,the 2nd covering the slope of the roof stretching to the left face till the ground, and last is a small piece covering the front face cut out for the door portion. All these pieces are

carefully measured cut fitted and stick to the PVC pipe structure with silicon-glue. They are kept in place with duck-tape, acrylic-tape and masking tape until the glue is set.

The polysheet is cut with circles of smaller diameter than the vertical pillars stretched and inserted where the pipe is sticking outside at the top of vertical columns. It is padded with an extra layer of sheet for safety from damage of stretching, sealed with silicon-glue for water not to leak in. All the polysheet is sufficiently stretched to make it tight. A wire is tied in zigzag fashion inside between pillars for the sheet not to sag at roof.

The 70% transparent Polyethylene Sheet is a material specifically used as greenhouse material. It reflects UV radiation and traps infrared radiation the keeping the inside temperature much more than outside (very useful in cold places).

Fig 2.11 : Greenhouse structure covered with poly sheet

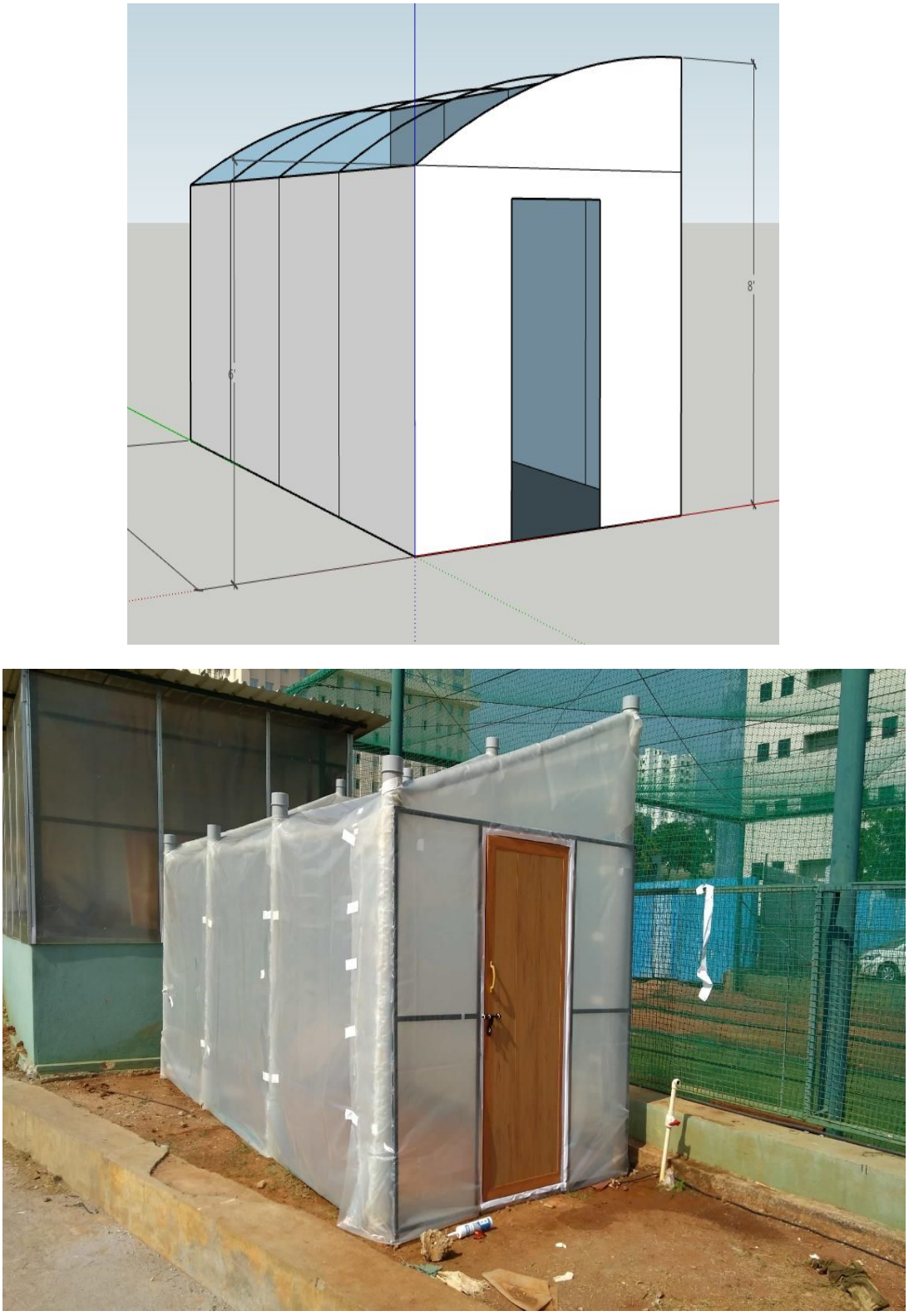


Fig 2.12 : Isometric view of Greenhouse with poly sheet

Specific placements of ventilation panels are critical and those bits cut out and covered up with a greenhouse grade Shade nets of 50% penetration for the windows.

Shade nets are used here to restrict the temperature from rising too rapidly during the peak hours of the day during the summer season.

The ventilation is made modular so that during seasons other than summer,the window flaps can be rolled down and covered up and the shade nets too can be rolled up to allow more light and heat into the greenhouse during winter.

Fig 2.13 : Greenhouse with shade nets

2.4 Sensors and Actuators Placement

The sensors used for the project were temperature sensor (to record the temperature inside the greenhouse) , humidity sensor (to record the humidity inside the greenhouse), ultrasonic sensor (to measure the water level of the storage tank) and soil moisture sensor (to measure the moisture content in the soil). Two exhaust fans (inlet and outlet) were used as actuators to cool down the greenhouse.

The 4 temperature/humidity sensors were suspended from above at different positions in the greenhouse to get the accurate temperature and humidity inside the greenhouse. The one soil moisture sensor were placed in each individual soil bed. The ultrasonic sensor was placed above the water tank to get the water level correctly. The actuator fans were placed at two extreme corners of the greenhouse, both on the higher side wall of the greenhouse.

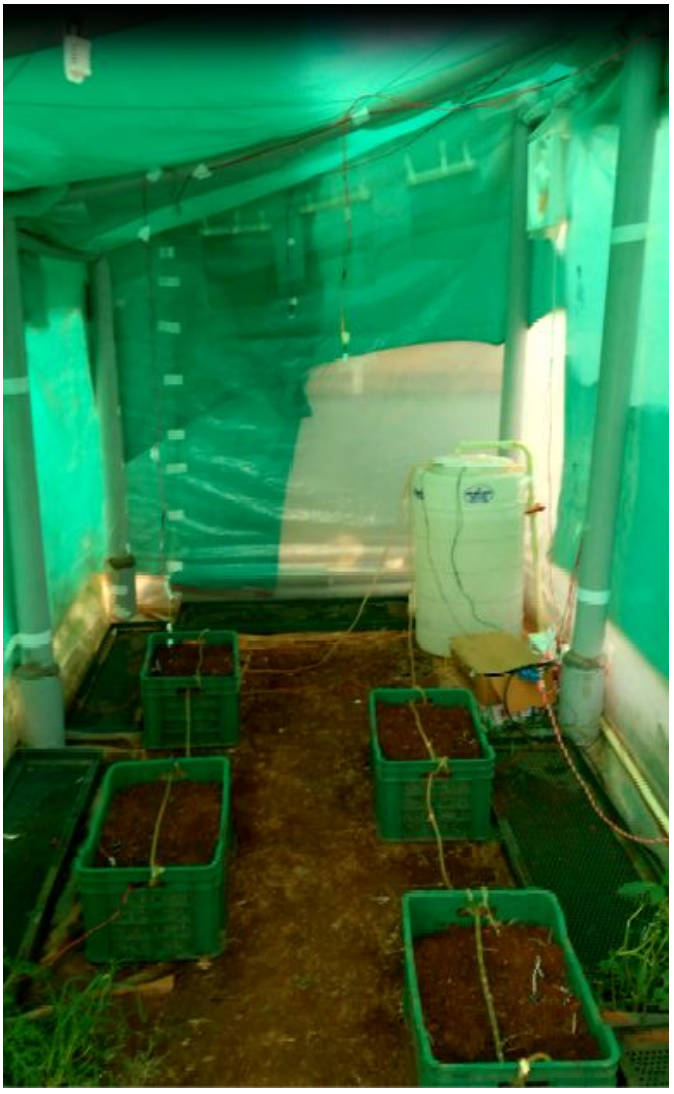


Fig 2.14 : Sensors and Actuators placement

2.5 Climate Control Requirements

Specific placements of ventilation panels are critical and those bits were cut out and covered up with a greenhouse grade Shade nets of 50% penetration for the windows. Shade nets are used here to restrict direct sunlight getting into the greenhouse and thus prevent the temperature from rising too rapidly at the peak hours of the day during the summer season. The ventilation is made modular so that during seasons other than summer, Ex: during Winter the window flaps can be rolled down and covered up and the shade nets too can be rolled up to allow more light and heat into the greenhouse.

Chapter 3

HW Integration for Greenhouse

This chapter deals with the working of the overall hardware architecture and different types of sensors and actuators used in this project. It talks about the specifications, the working principles and circuit schematics of each component and also about the interfaces and calibration techniques utilized for integrating it as a complete unit. The chapter also talks about the Drip Irrigation setup implemented in the greenhouse.

3.1 Hardware Architecture:

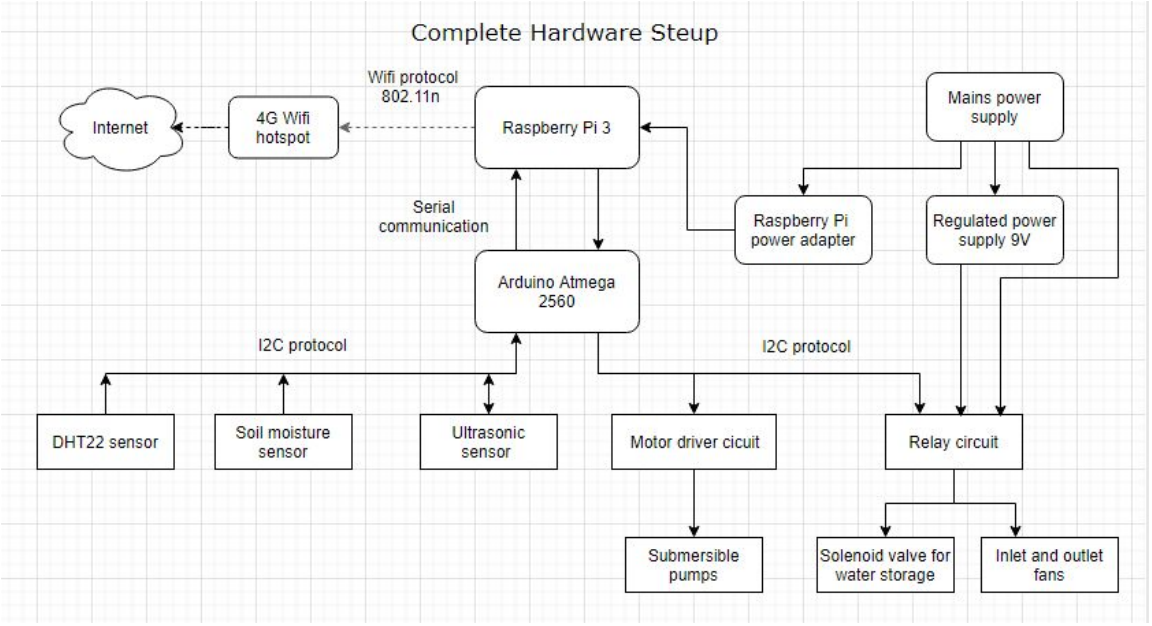


Fig: 3.1 System Hardware Architecture block diagram

Electronic Components used:

1	Raspberry Pi
2	Arduino mega
3	Temperature sensor
4	Humidity Sensor
5	Soil Moisture Sensor
6	Ultrasonic Sensor
7	Motor driver
8	Relay
9	Water pump
10	Solenoid Valve
11	Connecting Wires
12	Zero board
13	USB Cable

Table 3.1 : Electronic components used

The main components of the Hardware architecture of the greenhouse are the Raspberry Pi 3 Model B and an Arduino AtMega 2560 setup in a Master-Slave configuration which are communicating with each other through the serial USB ports on each device.

The Raspberry Pi is used as it provides an open platform for additional functionalities like onboard processing and data handling, a scope for image processing and a neatly integrated Wifi setup for easy internet connectivity for the system.
The Arduino provides an extra layer of abstraction to the setup as it manages the actual hardware sensing and actuation control of the system from the Raspberry Pi. Another main advantage of using an Arduino is due to its onboard 10-bit Analog to Digital Converter (ADC) which is not available in the Raspberry Pi.

A set of 4 DHT sensors, 4 soil moisture sensors(one for each soil bed), a relay breakout board, two motor driver circuit boards and an ultrasonic sensor are all connected to the Arduino using the I2C protocol with the help of connecting wires.

The DHT sensors provide a digital output on their data pin and hence are connected to the digital input pins of the Arduino. The Soil moisture sensors provide both Analog and Digital output. Here we use the analog output for more precision sensing and hence the sensors are connected to the Analog input pins of Arduino. The Relay board which consists of 4 relays is connected with a digital pin for each actuator as it accepts a digital input for actuation and is powered by a 5V supply common from the Arduino.

The Inlet and Outlet Fans are connected in parallel to the 1st relay and the solenoid valve is connected to the 3rd relay.The Fans are connected to the relay between the common and the Normally Open(NO) terminal .This means that the fans are switched on when the relay is ON and vice-versa. The solenoid valve is connected between the common and Normally Closed(NC) terminal. This means that the valve is switched ON ie. the valve is closed when the relay is OFF.

When the RPi is powered ON it initiates the Arduino to start the setup cycle and then acquires the parameter values through the serial port from the Rpi . The RPi sends the constant preset parameters to the Arduino as constants reference values for actuation. The Arduino then starts its sensing cycle and compares these values with reference values and performs necessary actuation to arrive at the desired conditions. The Arduino then acquires the parameter values from the different sensors and sends this data on the serial port to the RPi which stores it in to a .csv format as its database. The Arduino then regularly checks for the level of water in the storage tank inside the greenhouse by triggering the ultrasonic sensor that is placed in the tank which senses the distance between the water surface and the sensor itself and appropriately closes and opens the solenoid valve based on the sensor values. The actuators used are namely Inlet and Outlet fans for temperature control and submersible water pumps for both soil moisture and humidity control through drip irrigation and sprinkler system respectively.

The RPi is configured to connect automatically to a 4G enabled hotspot to send timely updates about the measured parameters and the logged parameter files by email to the user/owner.

3.2 Sensor selection:

3.2.1 Digital Humidity and Temperature (DHT) sensor:

DHT22 sensor uses exclusive digital signal collecting technique and humidity sensing technology and can output calibrated digital signal. Small size & low power consumption makes it suited for all kinds of harsh applications.

3.2.1.1 Sensor specifications:

- Power supply: 3.3V – 6V DC
- Output signal: single-bus
- Sensing period: ~2s
- measuring range: humidity 0-100% RH / temperature -40°C – 125°C
- accuracy: humidity ±3% / temperature ±0.5°C

3.2.1.2 Sensor Working Principle:

The DHT sensor consists of a NTC temperature sensor (like a thermistor), a humidity sensing component, and an IC on the back side of the sensor.

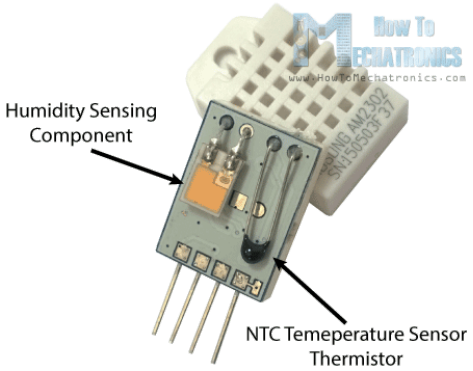


Fig. 3.2 DHT22 Temperature and Humidity Sensor

For measuring temperature the sensors use a NTC temperature sensor or a thermistor.A thermistor is a variable resistor that changes its resistance with change of the temperature.The term “NTC” represent “Negative Temperature Coefficient”, which means that the resistance decreases with increase in temperature.These sensors are made by using semiconductive materials such as ceramics or polymers in order to provide larger changes in the resistance with just small changes in temperature.

On the other hand for humidity measurement, the humidity sensing component has two electrodes with moisture holding substrate between the two electrodes. As the humidity in the air changes, the conductivity of the substrate varies or the resistance between the electrodes changes. This variation in resistance is measured and processed by the IC on board the sensor which makes it ready to be read by a microcontroller.

3.2.1.3 Circuit Schematics:

The DHT22 sensor has four pins, VCC, data pin, GND and a not connected pin which has no usage. A pull-up resistor of value 10K to 15K Ohms is required to keep the data line high and in order to enable the communication between the sensor and the Arduino Board. Here we have used the sensors that come with a breakout boards with built-in pull-up resistor and having just 3 pins.

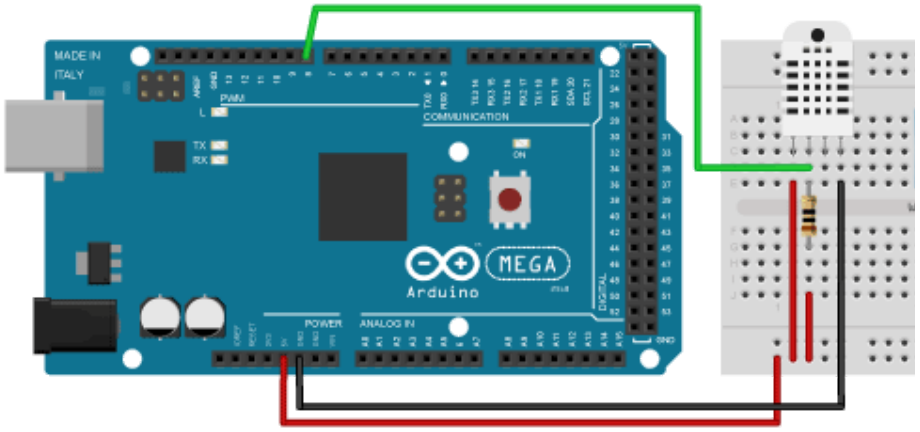
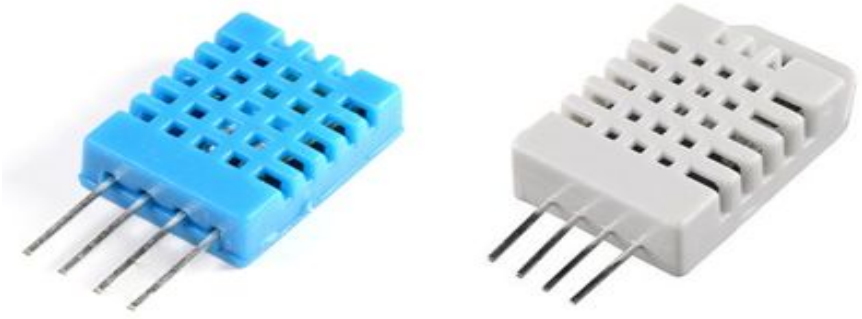


Fig. 3.3 Arduino and DHT22 Setup

The DHT11 and DHT22 sensors were compared and the comparison of their specifications of the two sensors is depicted in a table below.



DHT11 DHT22

DHT11		DHT22
0 - 50°C / ± 2°C	*Temperature Range*	-40 - 125 °C / ± 0.5 °C
20 - 80% / ± 5%	*Humidity Range*	0 - 100 % / ± 2-5%
1Hz (one reading every second)	*Sampling Rate*	0.5 Hz (one reading every two seconds)
15.5mm x 12mm x 5.5mm	*Body Size*	15.1mm x 25mm x 7.7mm
3 - 5V	*Operating Voltage*	3 - 5V
2.5mA	*Max Current During Measuring*	2.5mA

Table 3.2 DHT11 versus DHT22

DHT22 being the advanced one was selected for the project as it greater measuring range and a better accuracy than the other sensor. The operating voltage of both sensors is from 3 to 5 volts, while the max current used when measuring is same at 2.5mA.

3.2.2 Soil moisture sensor:

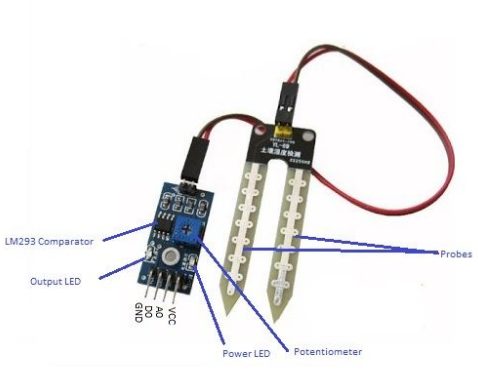


Fig. 3.4 Soil Moisture Sensor

The **Moisture sensor** is a sensor used to measure the water content(moisture) in the soil. This allows us to measure the moisture in a volumetric system.i.e volume of water for total volume of the soil bed. When the water content in the soil is low the sensor sends a high signal and when the soil is wet the low signal is sent.

3.2.2.1 Sensor Specifications:

- Working Voltage:**5V**
- Working Current:**<20mA**
- Interface type:**Analog**
- Working Temperature:**10°C~40°C**

3.2.2.2 Sensor Working Principle:

The Soil Moisture Sensor depends on the capacitance to measure the dielectric permittivity of the surrounding medium. In soil, dielectric permittivity is a function of the water content between the sensor. The sensor creates a voltage proportional to the dielectric permittivity, and therefore the water content of the soil affects its value. The sensor averages the water content over the sensors entire length. There is a 3 cm zone of influence with respect to the flat surface of the sensor, but it has very little sensitivity at the extreme edges.The Soil Moisture Sensor is used to measure the loss of moisture over time due to various factors like evaporation and plant uptake,evaluate optimum soil moisture contents for different species of plants and monitor soil moisture content to control irrigation in the greenhouse.

3.2.2.3 Circuit schematic:

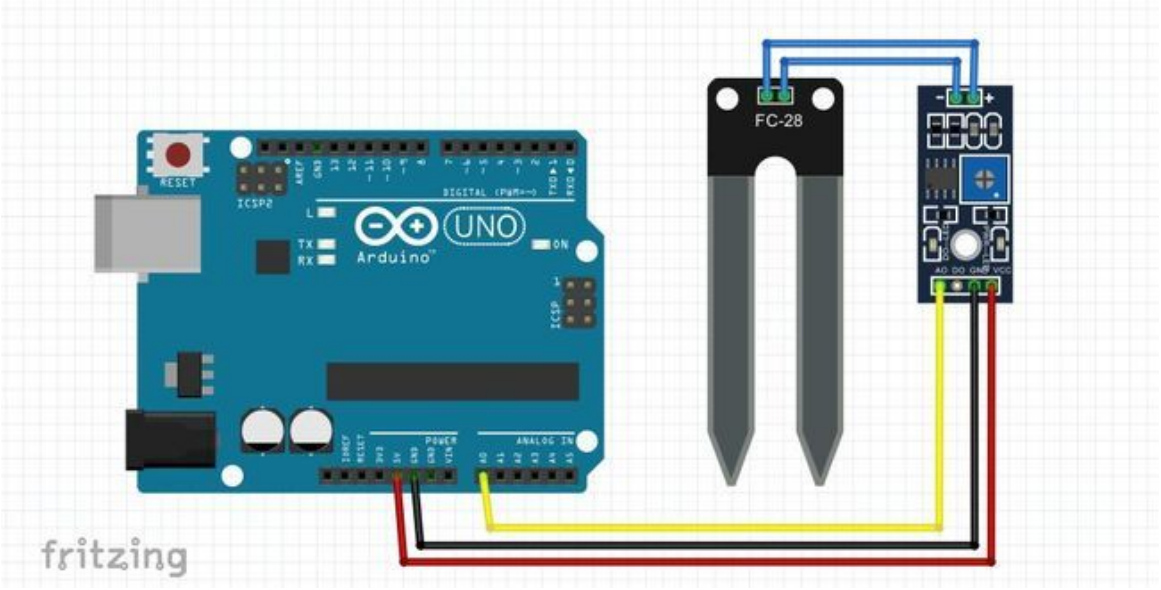


Fig 3.5 : Arduino and Soil moisture Setup

The moisture sensor is usually connected to a driving circuit which has a potentiometer and this circuit has 4 pins namely VCC, GND, Analog out and a Digital out.

The potentiometer is provided to calibrate the sensor output values for different soil types and preset moisture levels to be achieved.

3.2.3 Ultrasonic sensor:

Fig 3.6 : Ultrasonic sensor

An Ultrasonic sensor is a device that is used to measure the distance to an object by using sound waves. It measures distance by sending out a sound wave at a specific frequency and listening for that sound wave that bounce back from an obstacle near it. The sensor we are using is the generic HC-SR04.

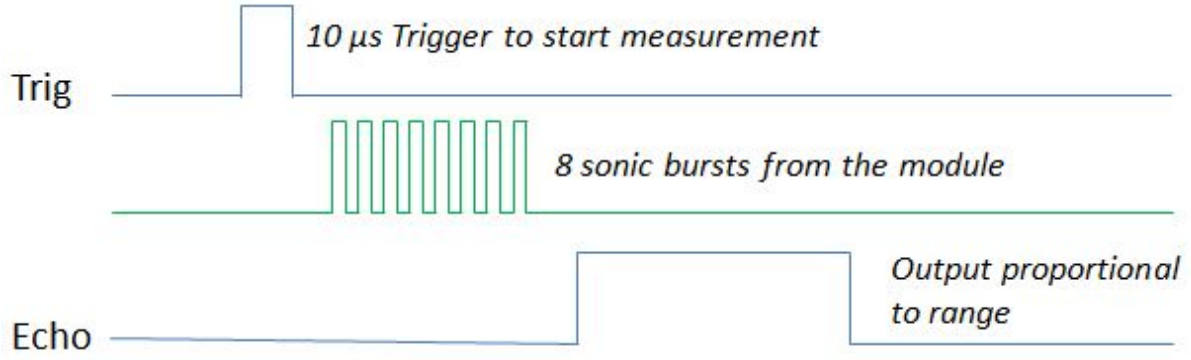
3.2.3.1 Sensor specifications:

- Operating voltage: 3 - 5V
- Practical Measuring Distance: 2 cm to 80cm
- Accuracy: 0.5 cm
- Operating Frequency: 40Hz
- Wave frequency: 40kHz
- Operating Current: <15mA

3.2.3.2 Sensor working principle:

The Ultrasonic sensor is a transmitter receiver pair.The transmitter transmits an ultrasonic wave of frequency 40 kHz and this wave travels in air and when it gets obstructed by any material or surface it gets reflected back toward the sensor this reflected wave is observed by the receiver module.

To generate the ultrasound waves the Trigger Pin is set on a High State for 10 µs. the sensor will send out an 8 cycle sonic burst which will travel at the speed sound and it will be received in the Echo pin. The Echo pin will output the time in microseconds the sound wave traveled in the process.



The Echo pin value will be double number required to reach the object because the sound wave needs to travel forward and bounce back the sensor. In order to calculate the distance in cm we need to multiply the received travel time value from the echo pin by 0.034 and divide it by 2.

3.2.3.3 Circuit schematic:

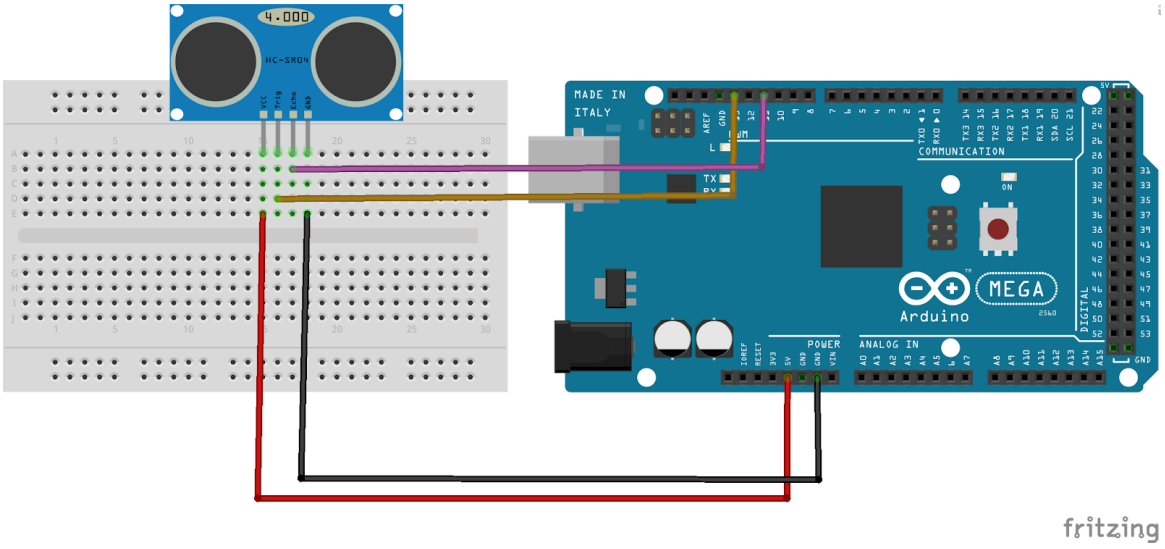


Fig 3.7 : Arduino and Ultrasonic sensor Setup

The Ultrasonic sensor is a 4 pin module, namely Vcc, Trigger, Echo and Ground respectively. The sensor is powered by a 5V power supply and a digital pin for the Arduino is connected to the Trigger pin. A short pulse (typically 10us) is applied to the Trig pin which sends very short pulses to ultrasonic waves and these waves get reflected when the fall on an obstacle and hence reflect back on the Echo pin. The Echo pin outputs the time in milliseconds the time taken for the wave to travel to and fro. The distance in cm is calculated by multiplying the velocity of sound in air in cm and the value of the Echo sensor divided by half.

3.3 Actuator Selection

3.3.1 Relay circuit board:

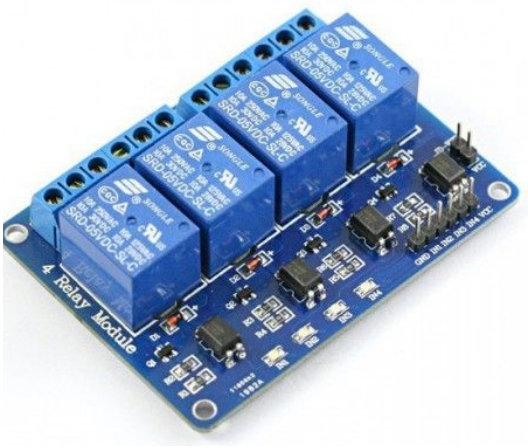


Fig 3.8 : Relay circuit board

Relays are mechanical contact switches which are operated with help of electric signals. The switching mechanism is carried out with the help of the electromagnet. They consist of an electromagnet and also a set of contacts depending the type of relay. The main operation of a relay comes in where only a low-power signal in to be used to control a circuit. The high end applications of relays require high power to be driven by electric motors and water pumps . In this project a 4-channel relay board is used as it is more compact and easy to handle.

3.3.1.1 Device specifications:

- Trigger Voltage (Voltage across coil) : 5V DC
- Trigger Current (Nominal current) : 70mA
- Maximum AC load current: 10A @ 250V/125V AC
- Maximum DC load current: 10A @ 30/28V DC
- Compact 5-pin configuration with plastic moulding
- Operating time: 10 msec Release time: 5 msec
- Maximum switching: 300 operating/minute (mechanically)

3.3.1.2 Device working:

Relay is an electromagnetic switch with a wire coil, surrounded by an iron core. A path of very low reluctance for the magnetic flux is provided for the movable armature and also to the switch point contacts in the relay. The movable armature is connected to the yoke that is mechanically connected to the switch point contacts. These parts are safely held with the help of a spring. The spring is used so as to produce an air gap in the circuit when the relay becomes de-energized

The working of a relay can be better understood by explaining the following diagram given below.

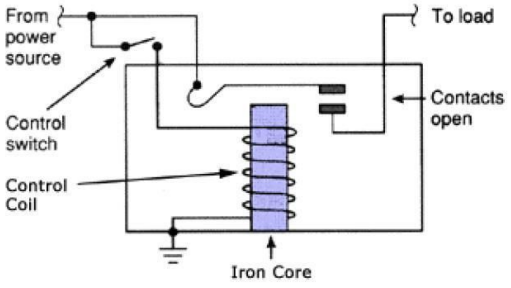


Fig. 3.9 Relay schematic diagram

The diagram shows an inner section diagram of a relay. An iron core is surrounded by a control coil. The power source is given to the electromagnet through a control switch and through contacts to the load. When current starts flowing through the control coil of the relay, the electromagnet starts energizing and hence intensifies the magnetic field strength. Thus the upper contact arm experiences a force of attraction to the lower fixed arm and thus closes the contacts causing a short circuit for the power to the load. On the contrary, if the relay was already de-energized when the contacts were closed, then the contact move oppositely and create an open circuit.

As soon as the current in the coil is off, the movable armature will be returned by a force back to its initial position. This force will be almost equal to half the strength of the magnetic force. This force is mainly provided by two factors. They are the spring and also gravity.

Relays are mainly made for two basic operations.. For low voltage applications, more preference will be given to reduce the undesirable noise component of the whole switching circuit. For high voltage applications, they are mainly designed to reduce a phenomenon called arcing which arises due to high voltage potentials at the terminals.

Pole and Throw

Relays have a working profile exactly same as of a switch.. A relay is used to switch one or more contacts (poles). Each pole has contacts that can be thrown in three ways mainly.

- **Normally Closed Contact (NC)** – NC contact is also known as break contact. When the relay is activated, the circuit disconnects. It circuit is activated when the relay is inactive.
- **Normally Open Contact (NO)** – NO contact is also called a make contact. It closes the circuit when the relay is activated.This is opposite to the NC contact. When the relay is activated, the circuit connects.
- **Change-over (CO) / Double-throw (DT) Contacts** – This type of contacts are used to control two types of circuits. They are used to control a NO contact and also a NC contact with a common terminal. According to their type they are called by the names **break before make** and **make before break** contacts.

3.3.2 Submersible Motor Pump:

Fig. 3.10 Submersible Motor

The submersible pump has a DC motor embedded in a tightly fit plastic mould which has a small chamber with an inlet and outlet port. This chamber consists of a fan like structure which is connected to the DC motor.

3.3.2.1 Actuator Specification:

- DC Voltage: 3 - 9V
- Maximum lift: 40-110cm / 15.75″-43.4″
- Flow rate: 80-120L/H
- Inside diameter of water outlet: 5mm / 0.2″
- Material: engineering plastic
- Rated Current: 60mA – 90 mA @ Rated Voltage
- Operating Temp Range: -20 °C ~ +60 °C
- Rated Speed: 9,000 RPM / 150Hz @ Rated Voltage

3.3.2.2 Actuator working:

When the pump is powered on the DC motor starts rotating and in turn rotates the fan connected to its shaft. As the chamber is leak proof , suction is experienced at the inlet port. This pulls the water inside the chamber and then the motor pushes out the water due to the fast rotations of the fan through outlet port. The flow rate of the pump is found to be sufficient for the Drip irrigation setup for the scale of this project.

3.3.3 Inlet and Outlet Fans :

Fig 3.11 : Exhaust fan

The type of Fan used in the project is an Exhaust fan. These fans pull or push air through them based on their orientation. Two fans are used in the project. One as Inlet and other as the Outlet. These come in different designs and specifications.

3.3.3.1 Actuator specification:

- Sweep/Fan Size (in mm):200
- Operating Voltage : 220V
- Motor Speed (RPM):1350
- Power Consumption (in Watts):25
- Air Displacement (CMM):500
- Back Shutter:Yes

3.3.3.2 Actuator working:

The Inlet and Outlet fans are connected in parallel so as to operate them as a single unit. The relay circuit is installed in series to these to act as a switch. When the relay turns ON the AC motor in the fan rotates and hence generates Air flow through itself.

3.3.4 Motor driver circuit:

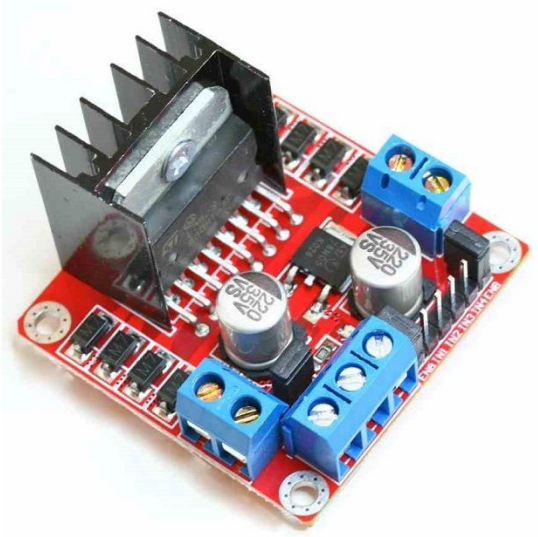


Fig 3.12: Motor driver circuit

A Motor driver circuit comprises of an integrated circuit chip typically used to control motors used in a system. Motor driver act as an interface between microprocessors and the motors in the system. L293D, L293NE are the most commonly used motor driver IC's. L293D consist of two H-bridge. These ICs are designed to control 2 DC motors simultaneously. H-bridge is the simplest circuit for controlling a low current rated motors.

3.3.4.1 Need for Motor Driver Circuit:

Most microprocessors operate at low voltages and require a small amount of current to operate, but the motors require relatively higher voltages and current . Thus this current demand cannot be supplied to the motors from the microprocessor as it may result in damaging the microprocessor and the motor itself. This is the primary reason to use the motor driver circuit. The motor driver translates the low voltage and current signal in the microcontrollers proportionally in the higher operating range.

3.3.4.2 Device specification:

- Operating Voltage :5V to 12V.
- Motor controller: L293D, Drives 2 DC motors or 1 stepper motor.
- Max current: 600mA per output channel.
- Peak Output Current: 1.2 Amp.

3.3.4.3 Device Working:

The Motor driver circuit receives either digital or analog(PWM) signals from the Arduino and transmits the relative signal to the motors. It has two input pins for each motor. The input to these pins decides the direction and the speed of the motor.

Consider IN1 and IN2 are the pins for Motor 1. If a HIGH digital signal is supplied at IN1 and IN2 is grounded, the motor turns in one direction. The direction can be reversed by reversing the supply signals.

The Pulse Width Modulated (PWM) signals are used to control the speed of the motor. The duty cycle of these signals determine the speed control over the motors.

3.4 Interfacing & Calibration:

3.4.1 Microprocessor/Microcontroller interfacing:

The Raspberry Pi 3 and Arduino are interfaced by an USB to Arduino port connector cable the data transfer between them is achieved by using the serial communication protocol. This topic on communication protocol is dealt in detail in the software architecture section. All the sensors and actuators are interfaced to the Arduino by using the I2C protocol.

3.4.2 Sensor Interfacing and Calibration:

3.4.2.1 DHT sensor:

The Digital humidity and Temperature (DHT) sensors provide a digital output pin for communication. This pin is connected to the digital pin of the arduino for sensing the corresponding temperature and humidity parameters. The DHT22 sensors have their own specified single wire protocol used for transferring the data. This protocol requires precise timing and the timing diagrams for getting the data from the sensors can be found from the datasheets of the sensor. However, because we use the DHT library which takes care of the timing constraints, the interfacing is simplified.

3.4.2.2 Soil moisture sensor:

The Soil moisture sensor with its driving circuit provides both a digital and an analog output. In this scenario we are using the analog value for better accuracy and reliability. This analog value is sensed by connecting it to the Analog pins of the Arduino. The Analog input of the Arduino has an onboard 10 bit ADC that gives readings ranging from 0 to 1023.

The sensor was initially tested in water. The readings achieved were between 200 and 1023 where 1023 being the most dry the sensor can be and 200 being the most wet. This ensures that the sensor is functioning properly as these sensors are prone corrosion and improper etching. In order to use the soil moisture effectively it is to be calibrated in the soil. The soil to be used is tested in its driest and the most wet state by placing the sensor in a sample soil bed. The same tests were done three times and a calibration table was developed for the same.The optimum range for the soil moisture was found to be 250 to 1000 units of ADC value. In order to adjust the calibration the potentiometer on the driving circuit is used.

Levels	Analog values
0	879
1	782
2	640
3	552
4	502
5	453
6	376
7	296

Table 3.3 Soil moisture calibration table

3.4.2.3 Ultrasonic sensor:

The Ultrasonic sensor has four pins namely VCC, Trigger, Echo and GND. The Trigger and Echo pins are connected to the digital pins of the Arduino. The trigger pin accepts a digital pulse signal and the Echo pin provides a digital output value in units of time (microseconds). This value is then sensed by the Arduino and the distance value between the water surface and the sensor is calculated in the microcontroller.

3.4.3 Actuators Interfacing:

3.4.3.1 Relay and Fans Interfacing:

The relay circuit used is powered by 5V power supply and the control signals are sent through one digital pin of the arduino per relay. The switching side of the relay is connected in series to the circuit that is to be controlled. When the signal is ON the relay switches on and based on the connected pin in the relay (NO or NC) and the circuit is switched ON/OFF accordingly. The fans are powered by a 220V mains supply and a series connection to these

relays is made in the NO (normally open) configuration, which means that the fans switch on when the relay switches ON.

The solenoid of the water storage system is connected to the NC (normally connected) configuration, the valve is ON(closed) when the relay is OFF as water is to be let open only for a stipulate amount of time based on the sensor.

3.4.3.2 Submersible pumps Interfacing:

The submersible pumps connected through the motor driver are operated at different speeds by using a PWM signal hence adding a deeper degree of control over the irrigation process of the system. The motor driver is powered by a regulated power supply set at 9V and that in turn is the power source to these pumps.

A motor driver circuit can operate two motors and has two input signal pins and an enable pin for each motor. The digital pins of the Arduino are used as signals pins and an enable pin needs to be HIGH for the motor to get activated. The water flow is controlled i.e.the speed of the pump by using a PWM signal as input to the motor driver.

3.5 Drip Irrigation Setup

Drip Irrigation is a mechanism to water the plants in most efficient way. It was implemented in the project to water the four soil beds. Four water pumps is immersed in an 150 litre water tank and four pipes were connected to the soil beds from the water pumps. The setup is as shown in the figure below. Fig 3.13 shows the proposed irrigation setup and Fig 3.14 shows the setup that was implemented in our project.

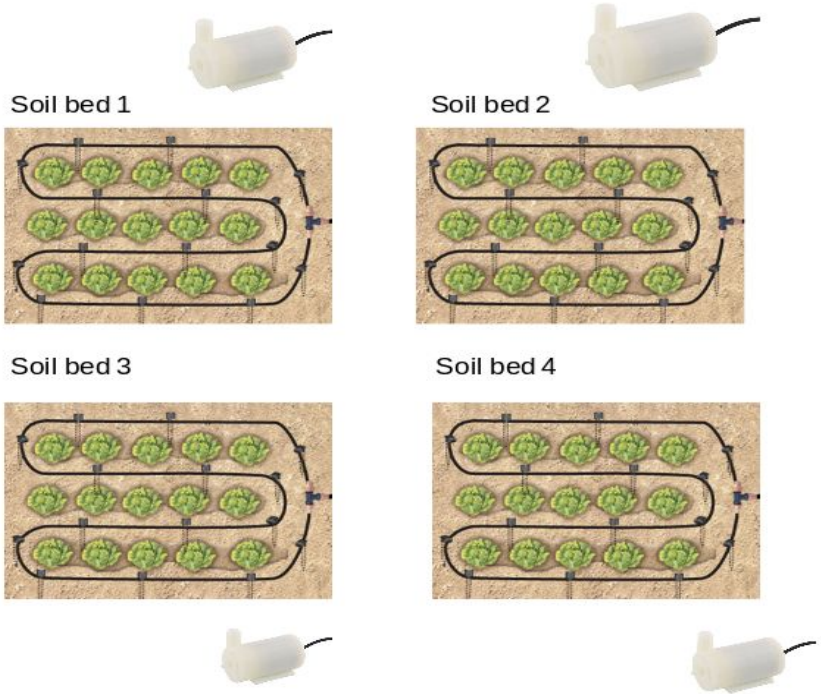


Fig 3.13 : Proposed Drip Irrigation setup

Fig 3.14 : Implemented Drip Irrigation setup

Fig 3.14 shows the placement of water storage tanks and soil beds. As seen from the above figure 3.14, the drip irrigation is carried out by getting small holes for the water pipes so that water comes out in the form of drips through these pipes. The thrust of water drips could be controlled by varying the motor speeds using PWM.

Chapter 4

Software Design for Greenhouse Automation

Microprocessor Raspberry Pi and microcontroller Arduino are used in Master Slave Configuration to carry out the sensing and actuation. Raspbian Operating System was loaded onto a 32gb memory card and installed in Raspberry Pi. The programs to be executed were written in python 2.7 version which was inbuilt in the raspbian OS. Arduino IDE was installed in the raspbian and C programming was used to program the Arduino. A flow chart for each working component is given for each section. The last section explains about the server design and the end user output along with climate control algorithm.

4.1 System Software

A detailed explanation is provided for setting up the system software in Raspberry Pi and the Arduino IDE.

4.1.1 Software Design for Raspberry Pi

The Raspberry Pi supports many operating systems, includes a derivative of Debian Linux titled - Raspbian, a port of Arch Linux OS, and Pidora which is port of Fedora -Linux OS. Raspbian is OS recommended by Raspberry -Pi Incorporation for new users due to the included software packages and user-friendly interface.

4.1.1.1 Installing Raspbian Operating System

Here we will setup the Raspberry pi with the operating system called Raspbian. In this project we use Raspbian Operating system.Using NOOBS software image file operating

system is going to install and it is the easiest way to do. If we purchase the SD card from RPi dealers,NOOBS files will be preloaded in it.If not, then we need to download the software and install onto a SD card.Now lets see how NOOBS file can loaded into SD card. Use a 4GB micro SD card for Raspberry Pi. Use NOOBS image from Raspberry Pi website and copy to SD card.

Select standard NOOBS image ,and download it from torrent.The image is more than 1GB size and download is quicker if we select torrent download. After downloading NOOBS version 1.4.1 which is in the form of zip file.

4.1.1.2 Setting up Raspberry Pi

By connecting Raspberry Pi to monitor or TV and keyboard we can do initial configuration. Pi connected to TV via HDMI cable and to monitor via VGA connector (use HDMI to VGA converter).

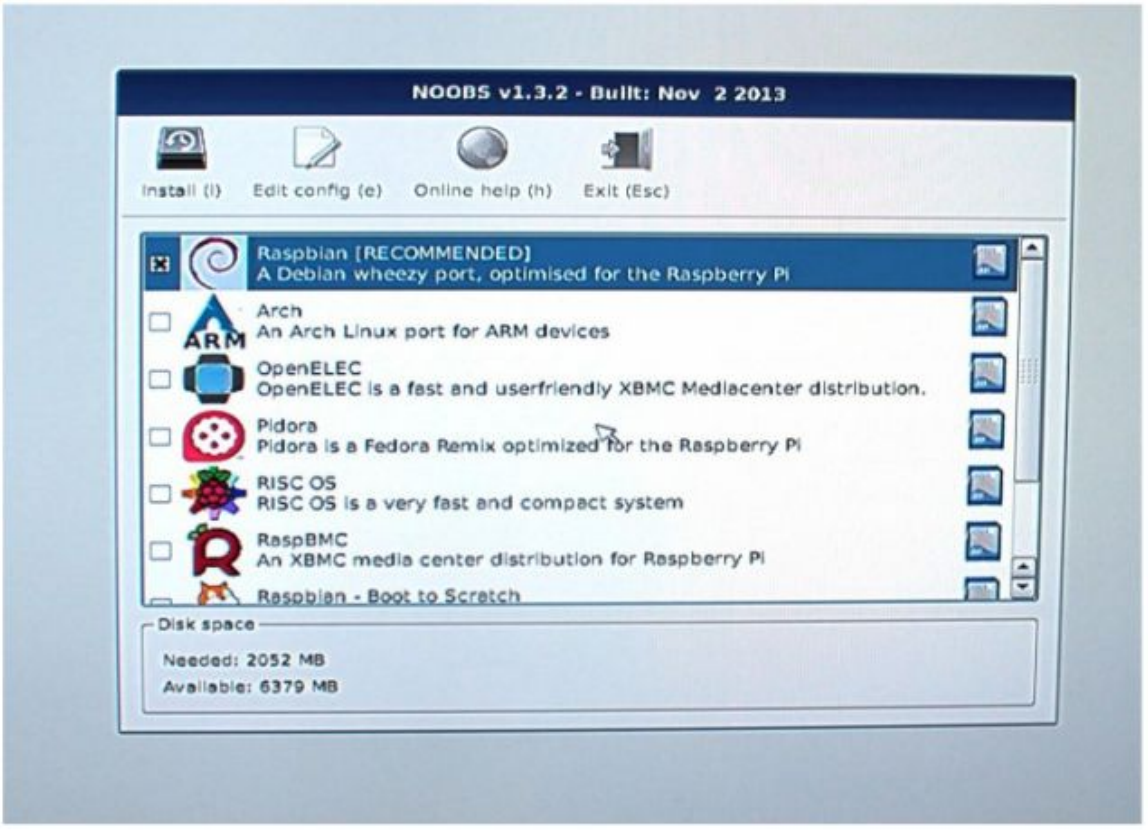


Fig 4.1 Raspbian OS Screenshot

This is the easiest way to startup with the Pi. Slot is provided in Pi board to insert SD card to store operating system and other files and connect keyboard and mouse to USB port.

Next power the Pi board by connecting micro-USB charger into power socket and start screen of Operating system will appear on monitor.Now install Raspbian operating system. After installing, Pi restarts and it opens with Linux operating system.

After completion of installation when the system boots for first time, it displays the configuration tool on screen.This tool helps to configure the features of operating system. Few features are listed below

Change User Password – Recommended to keep your system secure. The default username is *pi* and the password is *raspberry*.
Enable Boot to Desktop – This is not required for a robot. It provides a graphical interface similar to that used by Windows
Enable Camera – This is required if you want to use the Raspberry Pi camera module
Advanced Options – You can reduce the memory allocation to 16MB and provide a unique
hostname for your Raspberry Pi. It is normally a good idea to check for an updated version as well,but that is only possible when connected to the Internet.

Once completing the configuration, select finish and reboot the Pi. If any changes had to be done in future, we can perform it by typing below command

sudo raspi-config

Many messages scroll over monitor when operating system has been installed which is just an common process.Many PCs will do the same but these will be hidden so that user has impression of easy to understand.

The $ sign is known as order provoke and demonstrates that the interface to the PC is arranged for information. The substance at the left shows the username you are marked in as, the hostname of the PC and the present list that you are in. The # demonstrates that you are

running with super-client benefits. Once completing the configuration, select finish and reboot the Pi.

4.1.1.3 Configure Static IP address

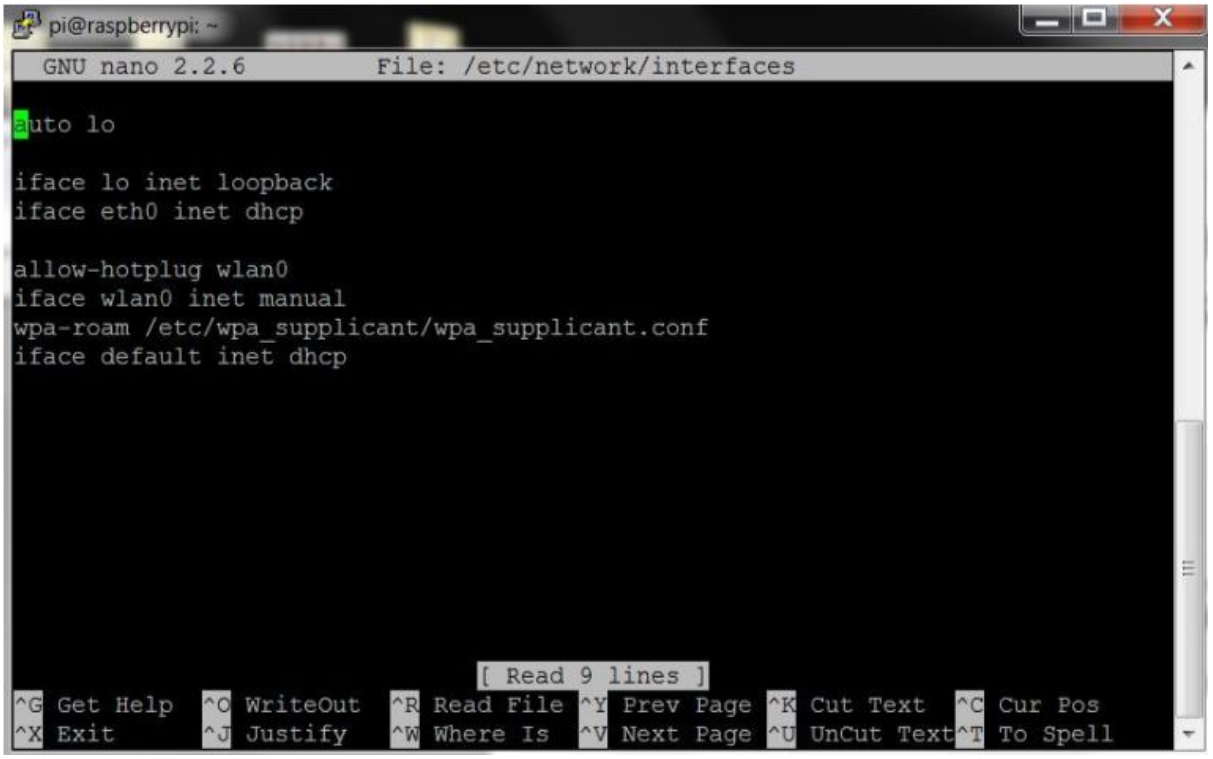
Raspberry pi usually takes IP automatically from router that is dynamic IP and it is denoted by DHCP when connected to a network.This IP address changes automatically once Pi is disconnected from network.Here in our LAB network is static,so we need to change the dynamic to static. Below steps are mentioned regarding configuration changes.

Step1

Editing Network Configurations

Type a command

sudo nano /etc/network/interfaces



check configuration of Pi whether its in static or dynamic.Configure as below if its static

iface eth0 inet static

then enter static IP configurations

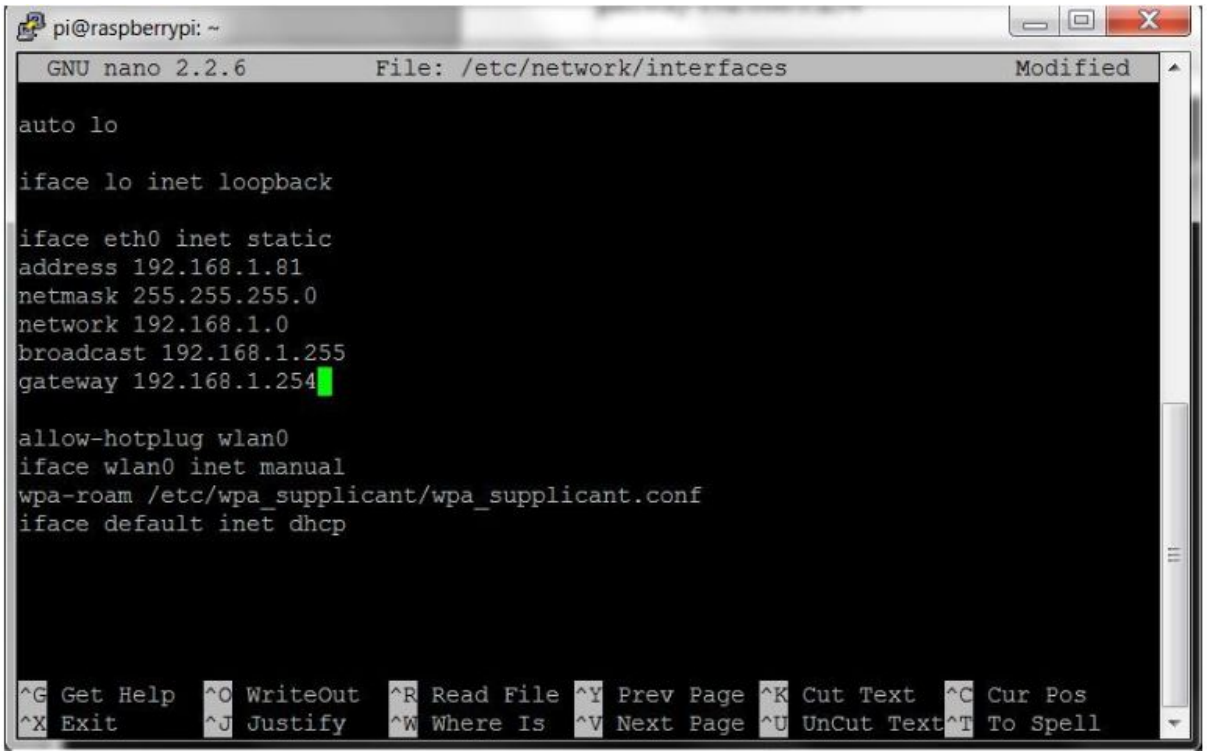
netmask 255.255.255.0

address 192.168.1.81

gateway 192.168.1.254

network 192.168.1.0

broadcast 192.168.1.255



It should look like above display.When it is done, press CTRL+X to exit and YES to save changes.

Step2:

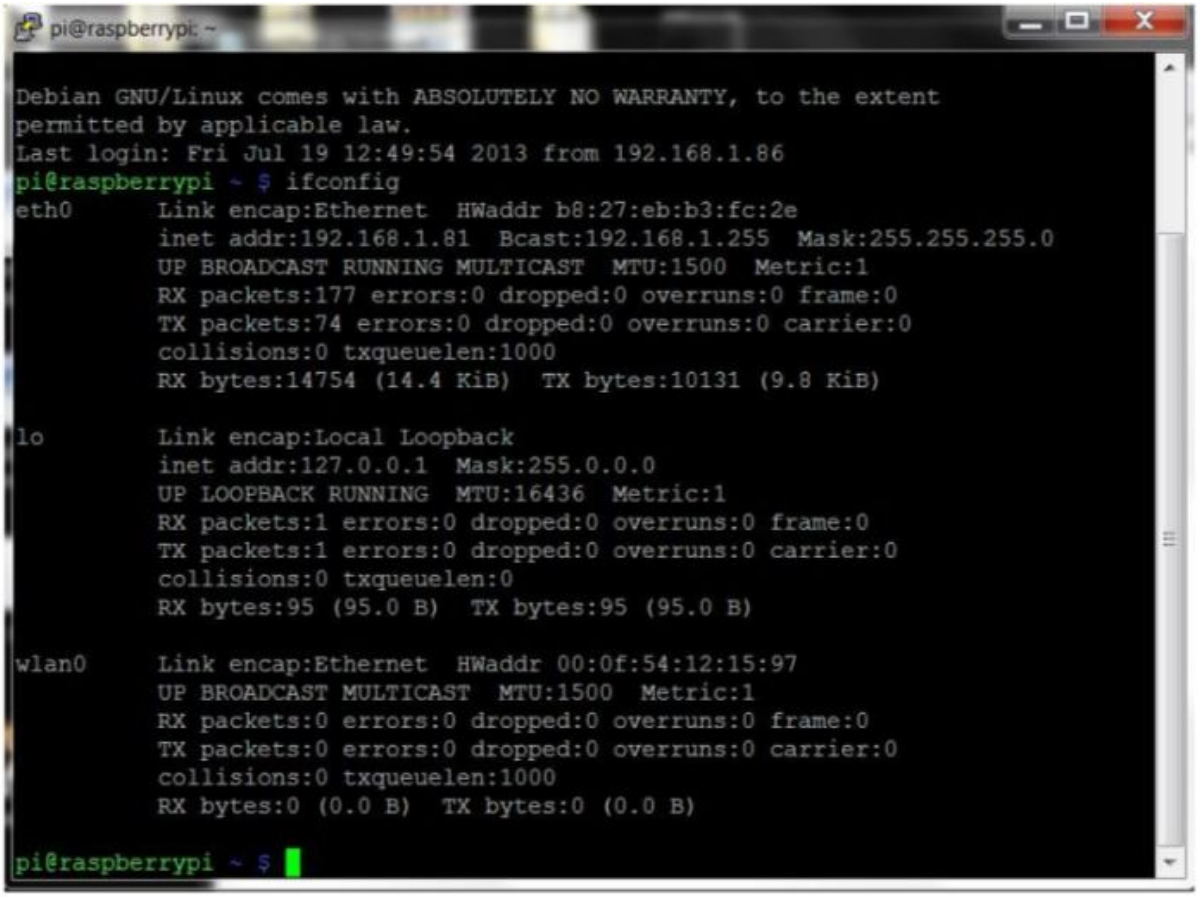
Recheck Static Configurations

Type below command to reboot

$sudo reboot

To recheck, type command

ifconfig



Now Static IP address is configured and ready to use the network.

4.1.1.4 Configure the DHCP server

IP address is static now on Raspberry Pi board.If we are using this Pi at dynamic IP address that is IP address is allocated when connected to network , for this we need reconfigure it to dynamic IP address.This is accomplished using DHCP protocol(dynamic host configuration protocol).

Type a command :

$sudo nano /etc/network/interfaces

Change the following:

iface eth0 inet static

to iface eth0 inet dynamic

Now its configured to dynamic address.

4.1.1.5 Putty Software

Putty software helps to build a communication link between Laptop and Raspberry Pi board. It's popular SSH mode helps to establish a secure connection and this software doesn't require any installation. It is for programmers and network administrators, as it is simple to use.The software give access only to the terminal of the Rpi.

4.1.2 Arduino Mega

Arduino uses the software Arduino IDE. It is a cross-platform application in C, and derived from IDE for Processing programming language. It is designed to understand programming to newcomers unfamiliar with software development technology. It also includes a code editor that help in syntax highlighting, brace matching, and automatic indentation. It is capable of compiling and uploading .ino programs to the arduino board with a single click.

Command-line interface can be avoided to run programs and edit makefiles. Although building on command-line is also possible with Ino. The Arduino IDE comes with a C/C++ library called 'Wiring', which is easier to work with many common input/output operations. Arduino programs are written in embedded C, but users need to define two functions to run a program:

setup() – a function which runs once at start of program and also can initialize settings loop() – a function which runs repeatedly until the board is powered off

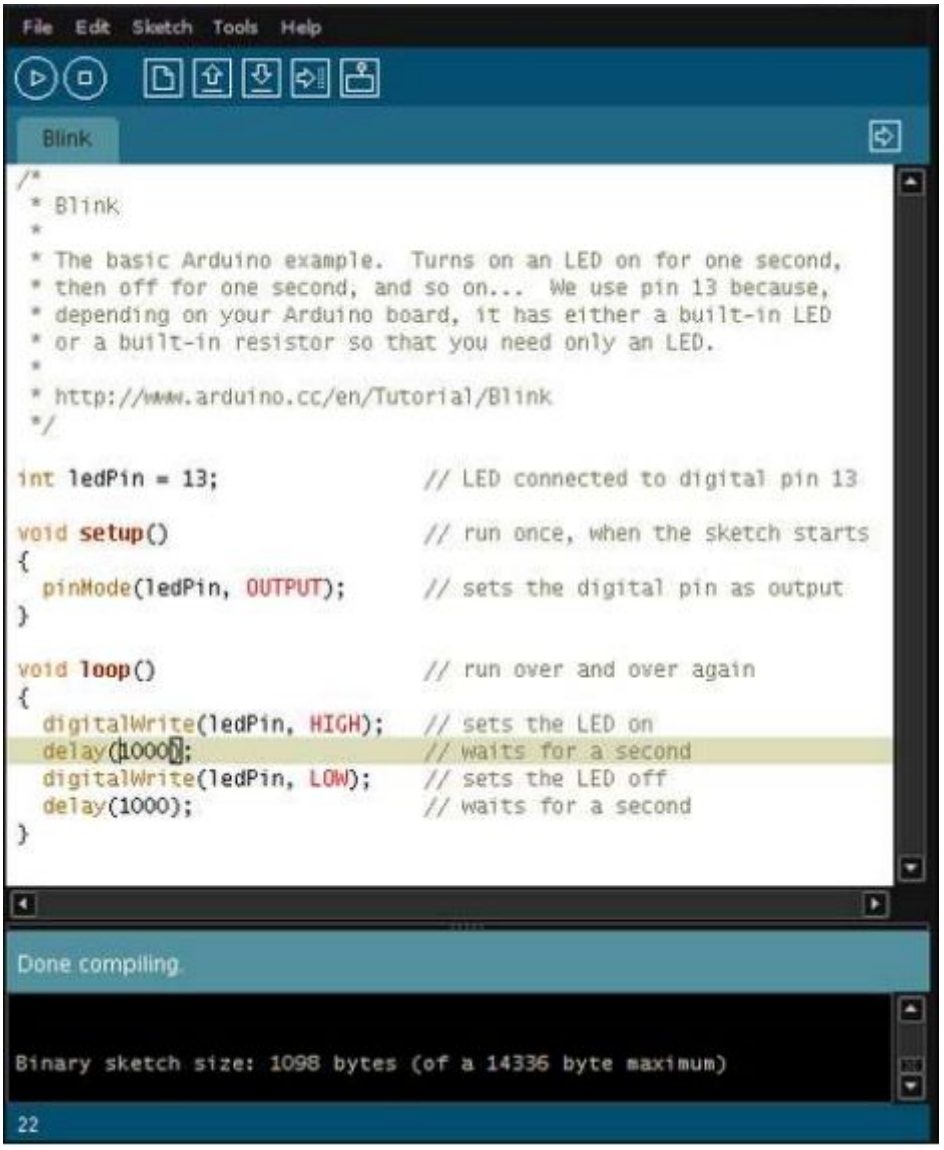


Fig 4.2 screenshot of Arduino IDE

A typical program for Arduino is to blink LED on and off. The program would look like below:

```
#define LED_PIN 13

void setup () {
    pinMode (LED_PIN, OUTPUT);      // enable pin 13 for digital output
}

void loop () {
    digitalWrite (LED_PIN, HIGH);  // turn on the LED
    delay (1000);                  // wait one second (1000
milliseconds)
    digitalWrite (LED_PIN, LOW);   // turn off the LED
    delay (1000);                  // wait one second
}
```

For the above given code to work, the positive of LED is connected to pin 13 and the negative is connected to ground. The Arduino IDE uses GNU toolchain and AVR Libc to compile programs. It uses avrdude for uploading programs to the board.

Arduino programs are divided in three parts: structure, values and functions. Following are some data types in Arduino IDE are

•void
•boolean
•String - object
•char
•double
•byte - 8 bit data
•long – 32 bit data.
•float
•string - char array
•array
•int - 16-bit data

Arithmetic operators:

Arithmetic operators usually include addition,multiplication,subtraction and division.For fractions,we can use float variables,if we can handle large size and slow computation speeds in Arduino. e.g. ,

z = z + 3;
y = y – 7;
k = l * 6;
q = q / 5;

Comparison operators:

Comparisons of one constant or variable with another are used in if statements to test if specified condition is true.

e.g. ,

i != j // i is not equal to j

i <= j // i is less than or equal to j

i < j // i is less than j

i > j // i is greater than j

i == j // i is equal to j

i >= j // i is greater than or equal to j

Logical operators :

Logical operators help in logically combining two expressions and return a TRUE or FALSE depending on operator.

There are three logical operators, AND, NOT and OR. for e.g. ,

Logical OR: if (i > 0 || j > 0) // true if either expression is true

Logical AND: if (i > 0 && i < 5) // true only if both expressions are true

Logical NOT: if (!i > 0) // true only if expression

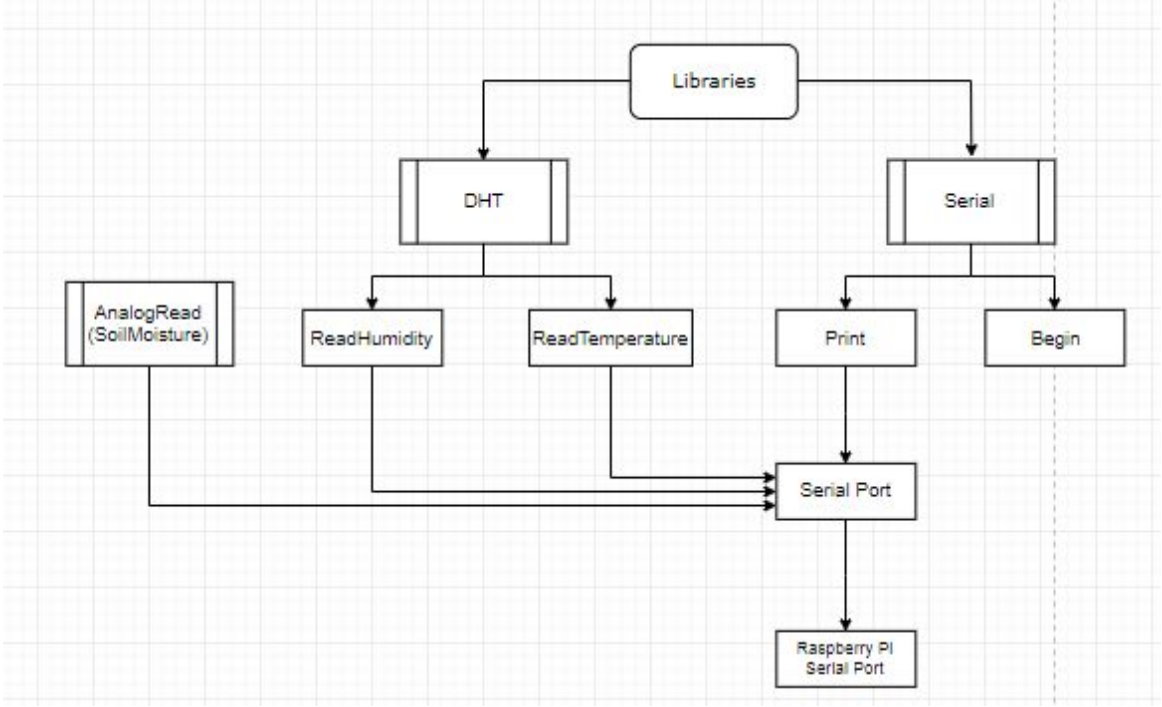
4.2 Sensor data-logger SW

The sensor values from four temperature sensor, humidity sensor and soil moisture sensor were transferred via serial communication from Arduino Mega to Raspberry Pi through a serial port. These values were logged into .CSV file in the Raspberry Pi and stored in home directory of Raspberry Pi. A software design was made to carry out data logging accordingly along with the Time stamp to indicate the date and time at which the data was logged. Following in sections 4.2.1 and 4.2.2, software design for Arduino mega and Raspberry Pi are described along with flow chart respectively.

4.2.1 Arduino data logger SW design

The values from four different sensors of temperature, humidity and soil moisture were recorded in Arduino on serial monitor. DHT22 library was used to operate dht22 sensor which is basically temperature and humidity sensor. Four temperature and humidity sensors placed at four different positions of the greenhouse were connected to four digital pins on Arduino Mega. readTemperature() and readHumidity() built-in functions in dht22 library were used to read data from the sensors. The four soil moistures sensors placed in four different soil beds were connected to analog pins of Arduino Mega. A built-in function AnalogRead() was used to read soil moisture sensor values.

All these 12 values i.e four temperature, four humidity and four soil moisture were transferred to serial port using built-in serial library of Arduino through printing them on serial monitor using Serial.print(). Below figure 4.3 shows the flowchart of the Arduino data logger SW.



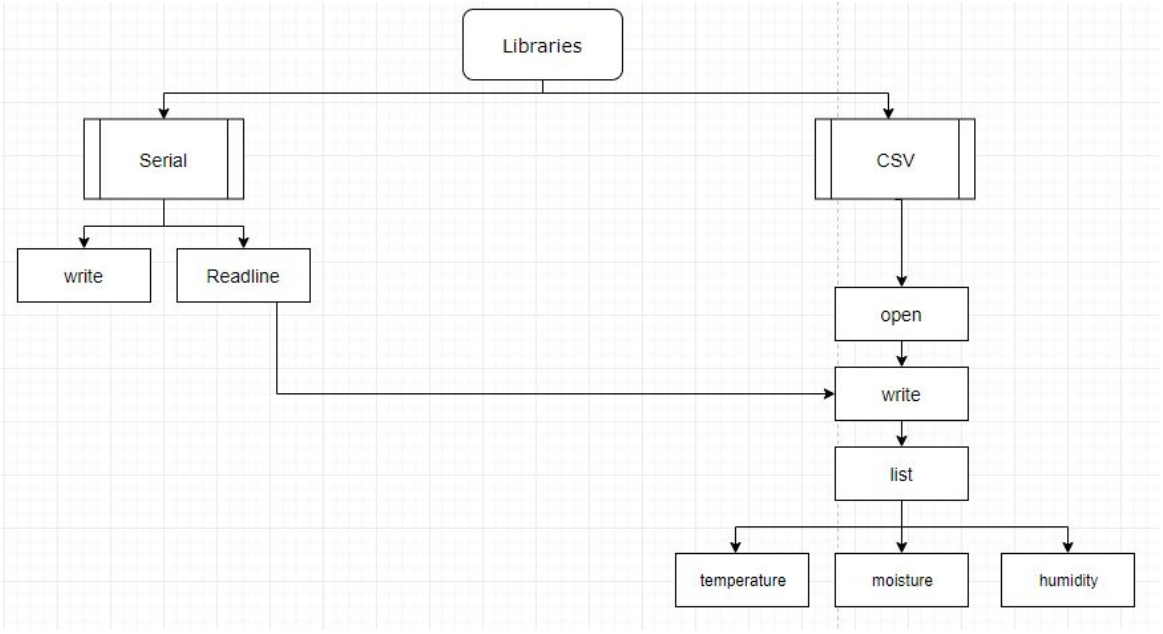
4.3 Arduino data-logger flowchart

4.2.2 Raspberry Pi data-logger SW design

The sensor values received from Arduino were read on the serial port of raspberry pi. A built-in serial library was used to carry out the functions on serial port. A ReadLine() function was used to read the data on serial port of raspberry pi and log into into a CSV file in a list format of rows and columns by importing CSV library.

A column was created in CSV file to represent timestamp of the received data. The next four columns represented four temperature values. Similarly another four columns for humidity and soil moisture respectively.

The data was logged every ten minutes throughout the day for 24 hours and sent to the user via email notification which will be explained later in this section. Below figure 4.4 shows flowchart of raspberry pi data-logging SW design.



4.4 Raspberry Pi flowchart for data-logging

4.3 Actuator and Irrigation SW for automation

The actuation for cooling down the greenhouse was carried out by inlet and outlet exhaust fans. After the sensor data was logged in the raspberry pi, it was necessary to control the greenhouse climate based on these parameters. Based on these parameters, an algorithm was designed to instruct the arduino to carry out actuation or irrigation depending upon the situation by sending a command from raspberry pi.

4.3.1 Raspberry Pi actuation and Irrigation SW for automation

The CSV containing data logs stored in the home directory of raspberry pi was accessed. The built-in Sys library was imported to carry out arithmetic operations on the sensor values. An average of Temperature and humidity from four different sensors was calculated to determine the current average temperature and humidity level of the greenhouse respectively. Consequently, the soil moisture sensors values that varied from 0 to 1023 were scaled to percentage form 0 to 100 percent. An ultrasonic sensor was used to determine the water level of the storage tank.

Based on the Greenhouse parameter readings, a command was transmitted to the arduino whether to actuate the fans or irrigate the soil bed using motor pumps or fill up the storage tank using solenoid via serial port using the write() function from serial library of raspberry pi. Below figure 4.5 shows the flowchart of Raspberry pi actuation and irrigation.

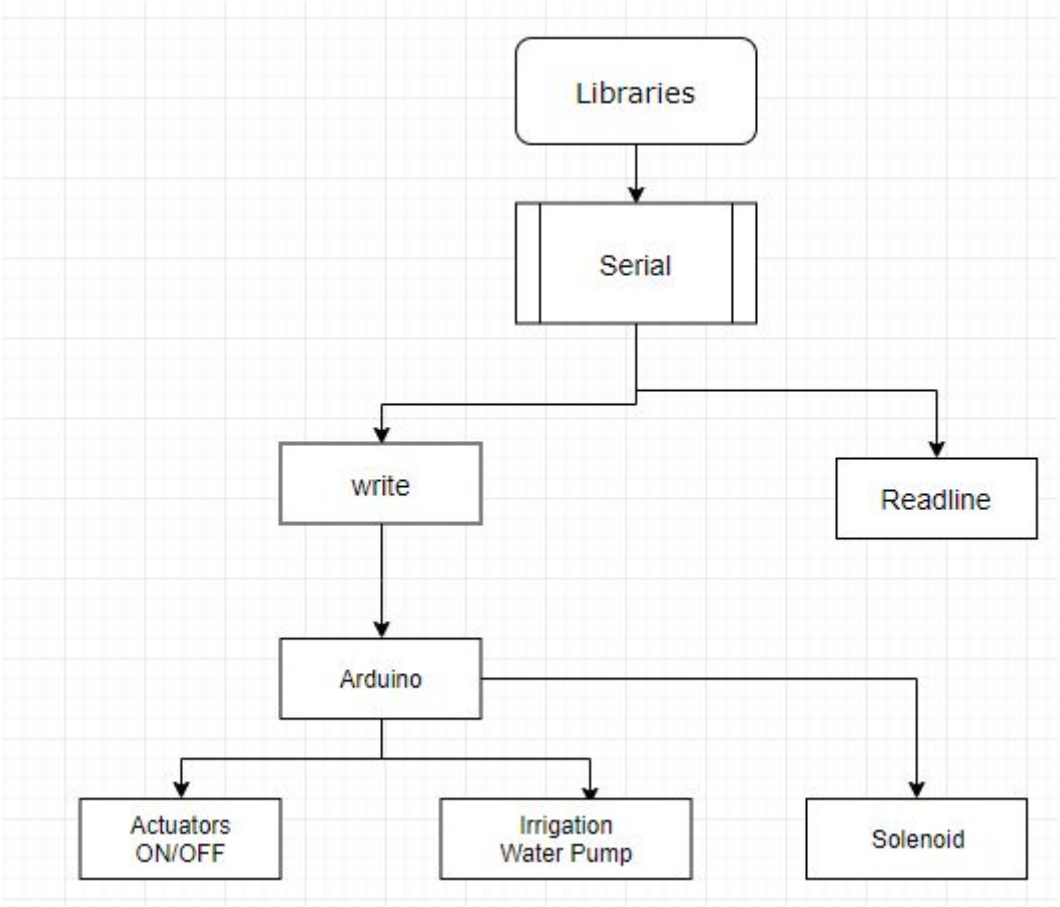
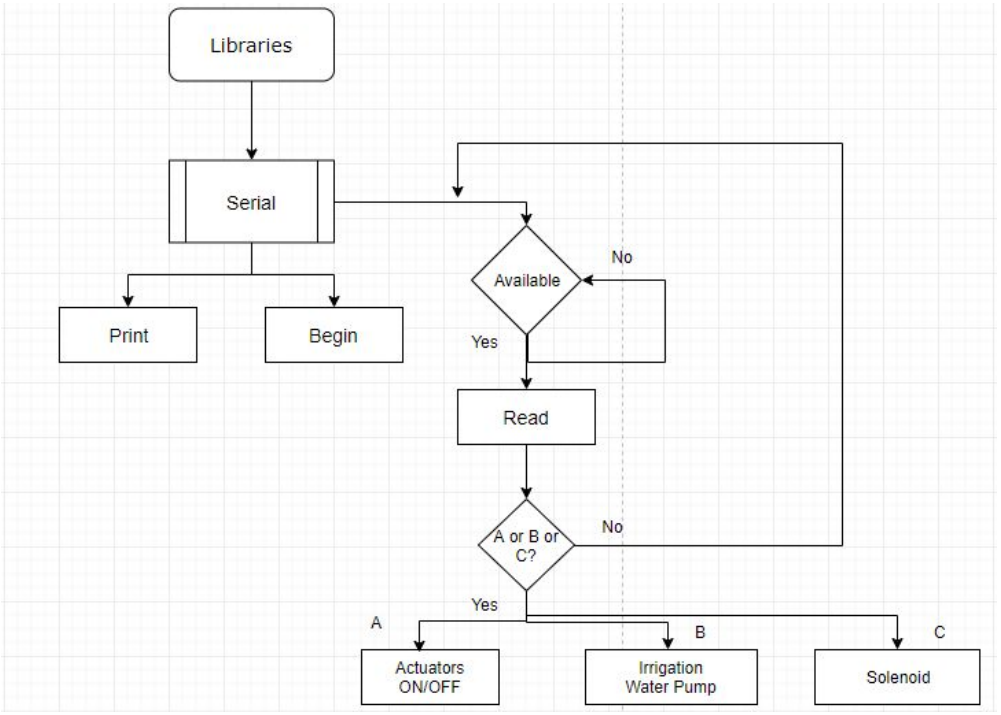


Fig 4.5 Raspberry Pi actuation and Irrigation flowchart

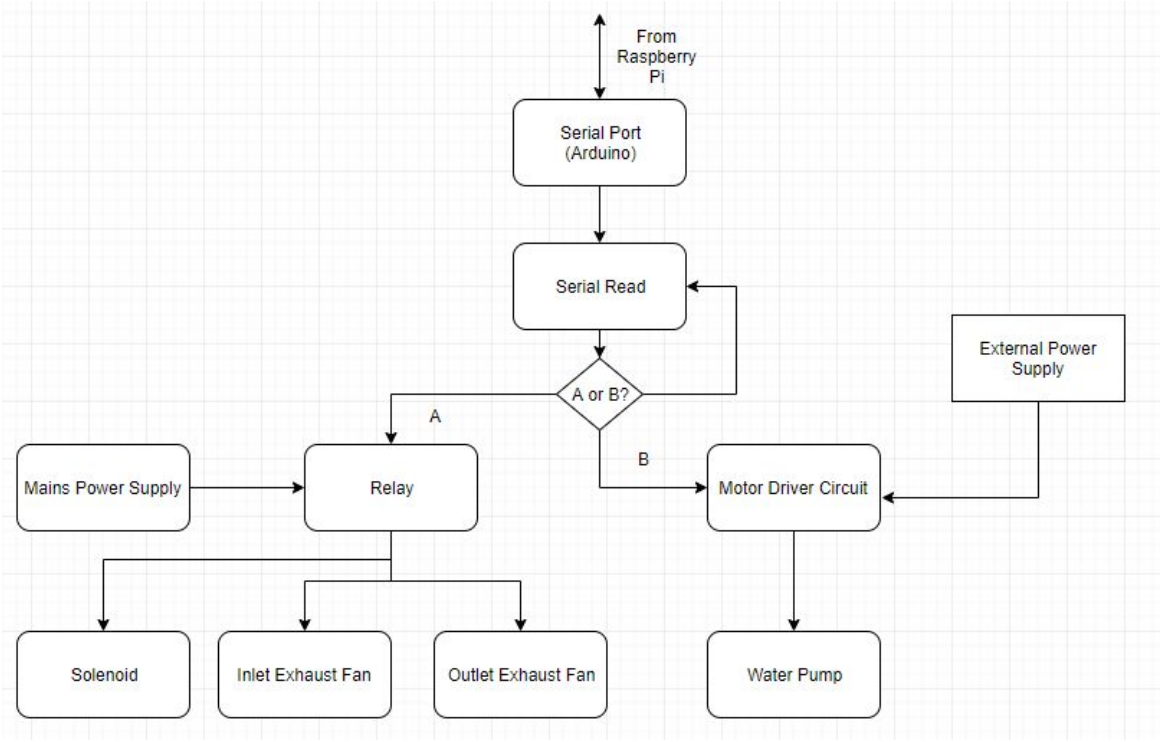
4.3.2 Arduino actuation and Irrigation SW automation

The arduino was programmed to listen on its serial port the incoming command from raspberry pi. When a data was available on arduino serial port, it was read and accordingly actuation, irrigation and filling up storage tank tasks were carried out.



4.6 Arduino actuation and Irrigation flowchart

The actuation of exhaust fans and solenoid was carried out using relay. The relay was connected to the digital pins of arduino. Based on the command received from raspberry pi, the relay either switches on Solenoid to fill up the water storage tank or switch on the exhaust fans or both accordingly. On the other hand, irrigation was carried out using submersive water pumps. Since arduino alone cannot drive the motors directly because of current supply constraint, the motors were connected to motor driver circuit which was in turn connected to regulated power supply of 12 volts. The motor driver was connected to pwm pins of arduino.

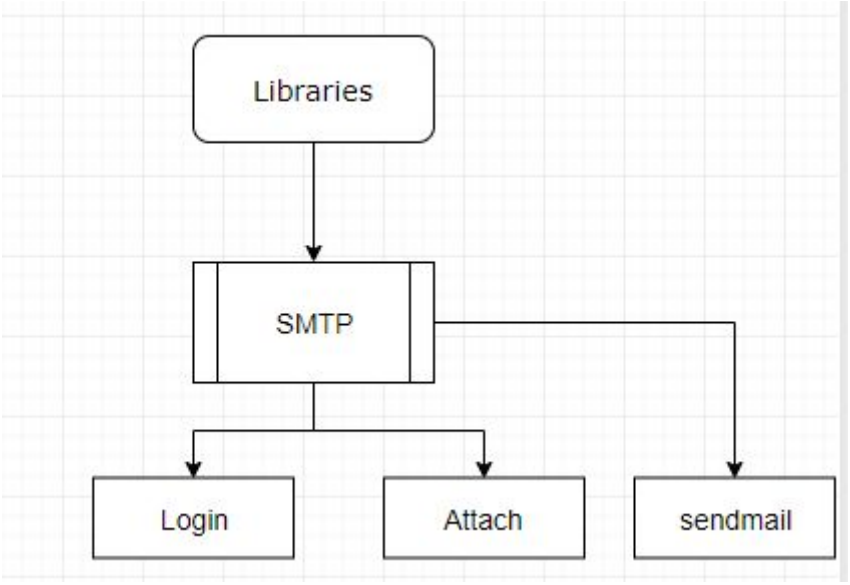


4.7 Arduino actuation and irrigation

4.4 Server SW design

The raspberry pi acted as a server for sending email notifications to the user. An Wifi hotspot was used to connect Raspberry Pi to the internet. An in-built SMTP (Simple Mail Transfer Protocol) was imported to send emails. An individual gmail ID was created for the project purpose called ‘automatedgreenhousepes’. Login() function was used to login to the server gmail address. The user’s email address was provided in the Login() function. An Attach() function was used to attach .CSV file of the logged data as an attachment for user’s reference. The sendmail() function sent the mail to the specified user.

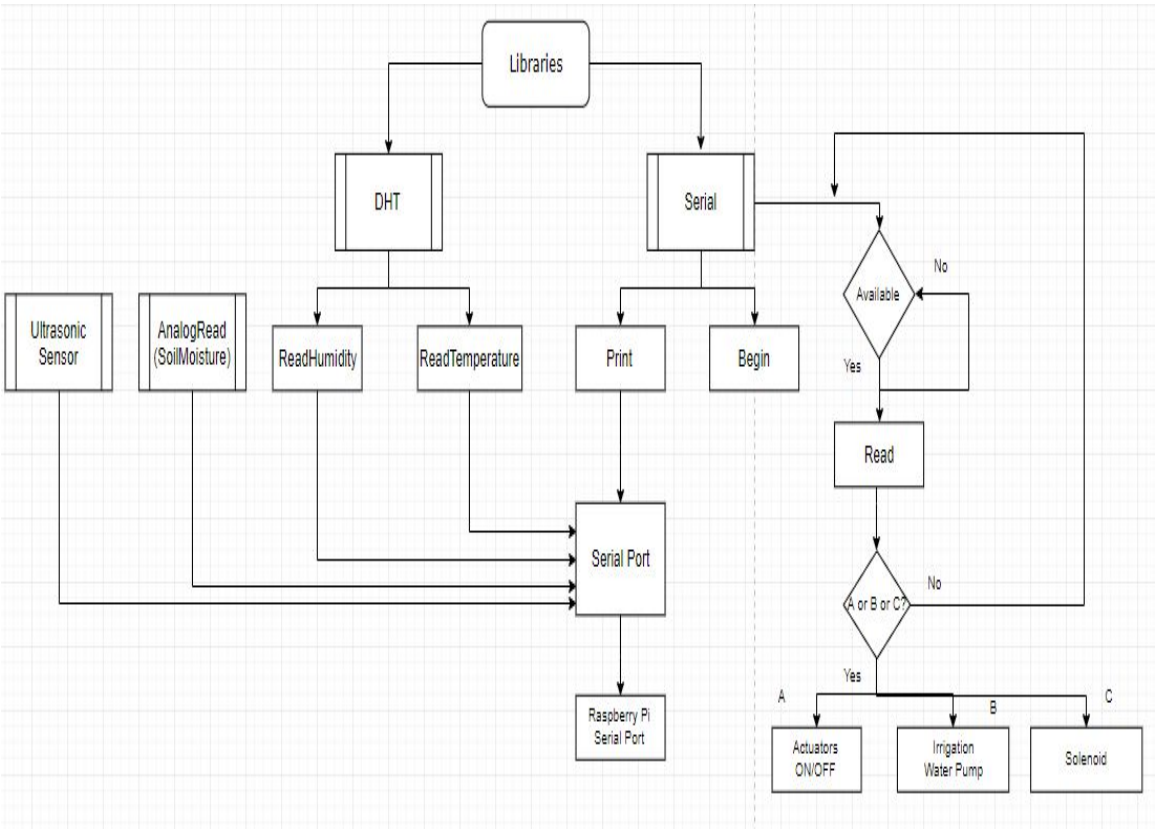
The email provided the user with information on current status of the greenhouse like the temperature, humidity, water level of storage tank and moisture level of the soil beds along with an attached data log file of 24 hours. Below figure 4.4.1 is the flowchart for Raspberry Pi server design.



4.8 Raspberry pi server design flowchart

4.5 Climate Control Algorithm

The Raspberry Pi and Arduino formed a single unit to execute climate control algorithm. The continuous uninterrupted communication between Raspberry Pi and Arduino Mega helped in control the greenhouse parameters efficiently. Below figure 4.5.1 and 4.5.2 shows the complete flow chart for Arduino Mega and Raspberry Pi software design respectively.



4.9 Arduino control automation Flow chart

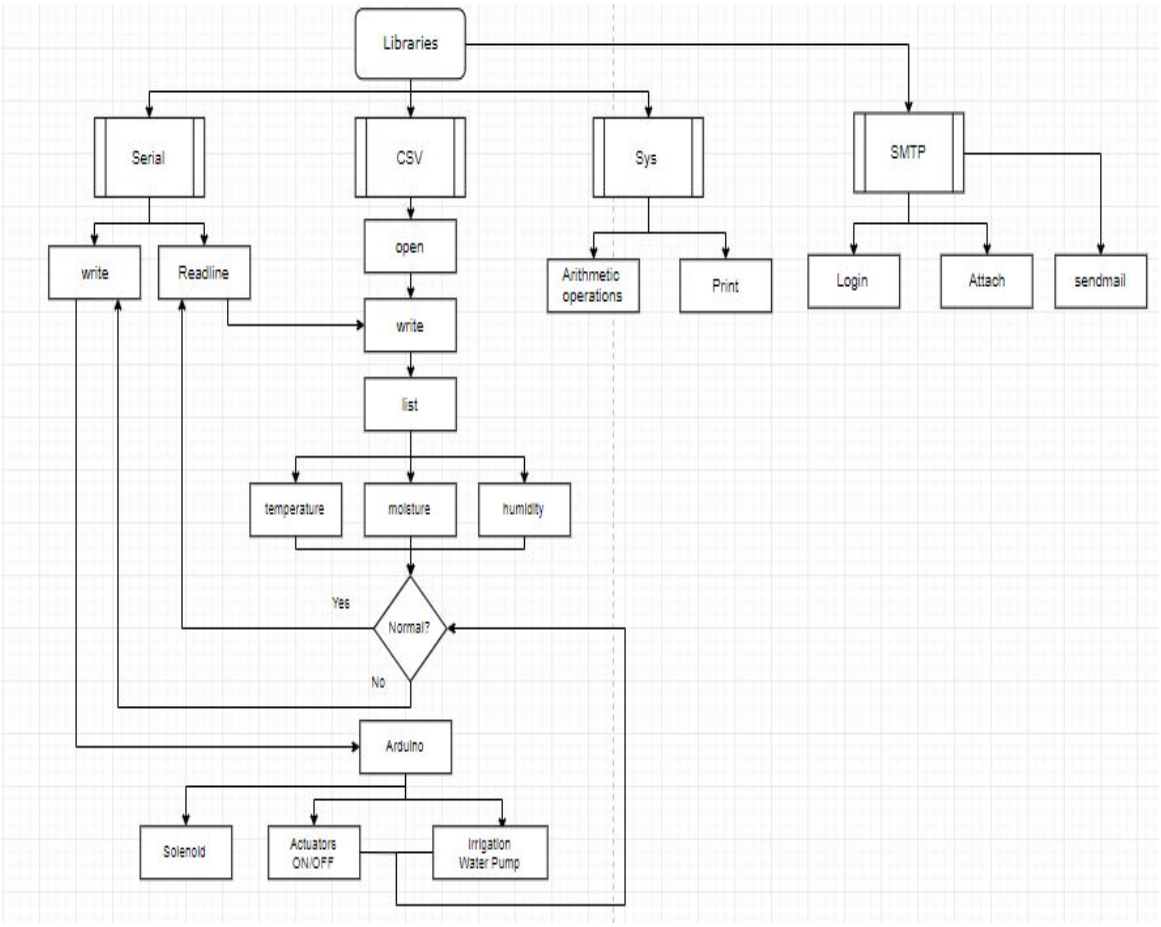


Fig 4.10 Raspberry Pi control automation Flow chart

The Climate control algorithm for the automated greenhouse is as follows :

1. Switching on the main power supply. Usually takes 1 minute to boot up raspberry Pi.
2. Transmitting the sensor readings from temperature, humidity, ultrasonic and soil moisture sensor from Arduino to Raspberry via serial communication.
3. Logging the received data every 10 minutes from arduino into .CSV file in raspberry pi in the form of list along with timestamp.
4. Calculating the average temperature and humidity from four sensors and representing soil moisture sensor values in the percentage form.
5. Comparing the current parameters values to the required parameter values and carrying out different tasks as below:
    a. If current temperature is more than the threshold temperature, send command to arduino to switch on fans otherwise switch off the fans.
    b. If the temperature is very high than the threshold, then

c. If current humidity is less than the required humidity, send command to arduino to switch on water pump to water the ground to increase humidity otherwise switch the motor off.

d. If the water level in the storage tank is less than the threshold, send command to arduino to switch on the solenoid to release water in the tank otherwise switch it off.

e. If the moisture content in any of the soil bed is less than the threshold, send command to the arduino to irrigate that particular soil bed using water pump otherwise switch it off.

6. Continue the steps from 2 to 5 for 24 hours.
7. Send an email notification to the user on status of the greenhouse indicating temperature, moisture, water level of storage tank and moisture levels of soil bed along with data log as an attachment.

Below figures 4.11 and 4.12 shows the sample raw data CSV file and screenshot of the email format that user receives respectively.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Time Stamp	temp1	temp2	temp3	temp4	humid1	humid2	humid3	humid4	soil1	soil2	soil3	soil4
2	Thu Apr 12 16:30:40 2018	30	29.6	29.3	29.4	43.6	45.6	43.6	45.4	130	144	125	122
3	Thu Apr 12 16:30:47 2018	30	29.6	29.4	29.4	43.6	45.5	43.6	45.3	127	143	124	122
4	Thu Apr 12 16:30:53 2018	30	29.6	29.4	29.4	43.6	45.4	43.5	45.2	178	185	167	181
5	Thu Apr 12 16:30:59 2018	30	29.6	29.4	29.4	43.5	45.3	43.4	45.1	183	194	181	182
6	Thu Apr 12 16:31:05 2018	29.9	29.5	29.4	29.4	43.4	45.2	43.3	45.1	169	182	165	165
7	Thu Apr 12 16:31:11 2018	29.9	29.5	29.4	29.4	43.4	45.1	43.3	45	141	154	130	132
8	Thu Apr 12 16:31:18 2018	29.9	29.5	29.4	29.4	43.3	45.1	43.2	44.9	137	152	130	132
9	Thu Apr 12 16:31:24 2018	29.9	29.5	29.4	29.4	43.3	45	43.1	44.9	161	176	161	161
10	Thu Apr 12 16:31:30 2018	29.9	29.5	29.4	29.4	43.3	45	43.1	44.8	173	188	176	173
11	Thu Apr 12 16:31:36 2018	29.8	29.5	29.4	29.4	43.2	45	43	44.8	163	177	161	158
12	Thu Apr 12 16:31:43 2018	29.8	29.5	29.4	29.4	43.2	44.9	43	44.7	152	166	144	142
13	Thu Apr 12 16:31:49 2018	29.8	29.5	29.4	29.4	43.2	44.9	43	44.7	131	145	121	122
14	Thu Apr 12 16:31:55 2018	29.8	29.5	29.4	29.4	43.2	44.9	42.9	44.7	129	144	120	121
15	Thu Apr 12 16:32:01 2018	29.8	29.5	29.4	29.4	43.3	44.8	42.9	44.7	150	165	148	148
16	Thu Apr 12 16:32:07 2018	29.8	29.5	29.4	29.4	43.3	44.8	42.8	44.7	177	191	179	177
17	Thu Apr 12 16:32:14 2018	29.7	29.5	29.4	29.4	43.2	44.8	42.8	44.6	161	175	155	153
18	Thu Apr 12 16:32:20 2018	29.7	29.5	29.4	29.4	43.3	44.8	42.8	44.6	130	145	121	123
19	Thu Apr 12 16:32:26 2018	29.7	29.5	29.4	29.4	43.3	44.8	42.8	44.6	176	190	177	176
20	Thu Apr 12 16:32:32 2018	29.7	29.5	29.4	29.4	43.3	44.8	42.8	44.6	143	156	132	132
21	Thu Apr 12 16:32:38 2018	29.7	29.5	29.4	29.4	43.3	44.8	42.8	44.7	155	171	154	154
22	Thu Apr 12 16:32:45 2018	29.7	29.5	29.4	29.4	43.3	44.8	42.8	44.6	165	178	161	160
23	Thu Apr 12 16:32:51 2018	29.6	29.5	29.4	29.4	43.3	44.8	42.8	44.6	133	148	125	127
24	Thu Apr 12 16:32:57 2018	29.6	29.5	29.4	29.4	43.3	44.8	42.8	44.6	177	192	179	177

Fig 4.11 Raw Data Sample

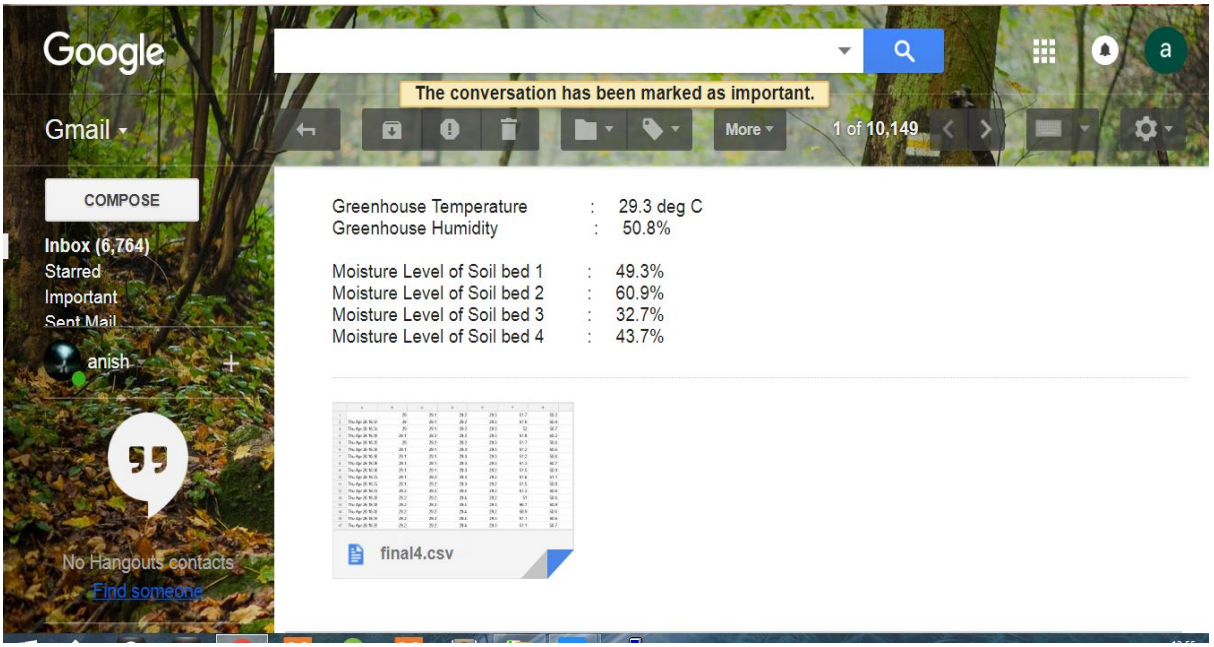


Fig 4.12 Email notification screenshot

Chapter 5

Data Acquisition and Analysis

The acquisition and analysis of data here refers to the temperature logs and humidity logs recorded by the raspberry pi throughout the day for a couple of days. The purpose of carrying out this procedure was to study the variations of temperature and humidity in the greenhouse with the changing climatic conditions outside the greenhouse. Since the project was carried out in summer season, there were tremendous amount of variations in temperature and humidity. It was required to resort to alternate solutions to bring down the temperature. So, shade nets were installed over the polysheet greenhouse to cool down the greenhouse. The shade nets were installed in a modular way so that they could be taken down in winter season to help trap the heat.

The following sections provides a detailed analysis of temperature and humidity variations of greenhouse taking into consideration the effects of exhaust fans, passive ventilation using shade nets and sprinkler irrigation.

5.1 Temperature data logs: open-loop

The open-loop temperature data logs refers to the behaviour of temperature changes in the greenhouse without any actuation or ventilation. Below figure 5.1 shows the temperature versus time graph for the greenhouse throughout the day.

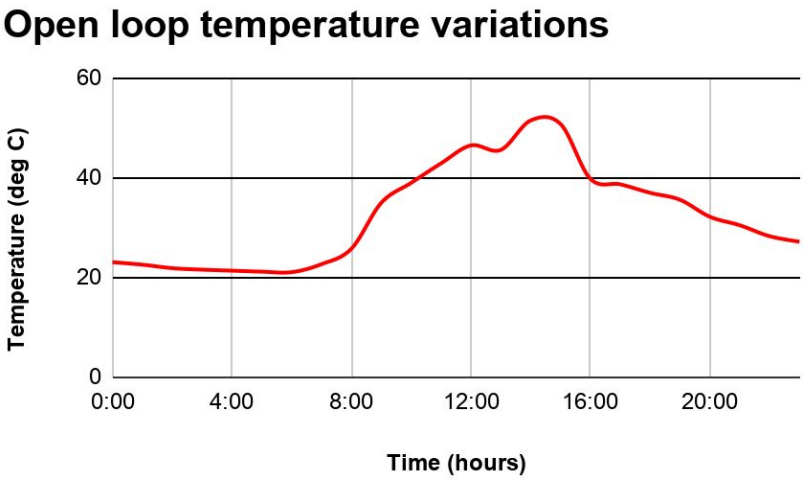


Fig 5.1 : Open loop temperature variation

As we can see from the above temperature graph, the peak temperature is 51.4 deg C at around 2pm in the afternoon which is not sustainable to grow plants. Therefore, there was a need to bring in active and passive ventilation using exhaust fans and shade nets respectively to bring down the greenhouse temperature.

5.2 Temperature data logs: closed-loop

The close-loop temperature data logs refers to the behaviour of temperature changes in the greenhouse under the effect of exhaust fans only, passive ventilation through shade nets only and both exhaust fans and shade nets. The following 3 sub-sections provides the temperature variations for different setups.

5.2.1 Active ventilation temperature data log

The active ventilation of greenhouse is carried out using two exhaust fans i.e inlet and outlet exhaust fans. The fans are connected to a relay which is controlled by arduino. Both the fans are switched on simultaneously when the temperature rises above a threshold value and switched off when the temperature falls below the threshold value. Below figure 5.2 shows the graph of temperature versus time during active ventilation of the greenhouse.

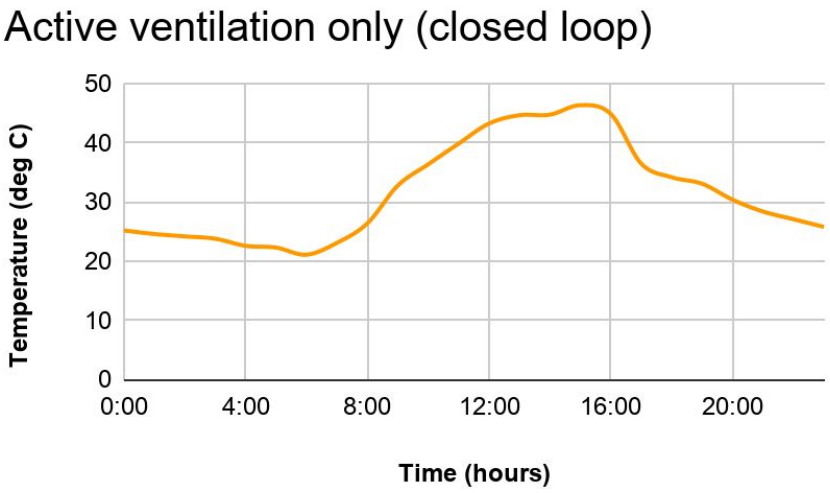


Fig 5.2 Closed loop Active ventilation only

As we can see from the Fig 5.2, In closed loop active ventilation using inlet and outlet exhaust fans, the peak greenhouse temperature was brought down from 51.4 deg C to 46.4 deg C. This temperature too does not seem to be favourable and hence there had to be passive ventilation too. The next section shows the temperature variations using passive ventilation only.

5.2.2 Passive ventilation temperature data log

The passive ventilation of the greenhouse is carried out using shade nets. When recording the temperature during passive ventilation, the exhaust fans were switched off to understand the effect of using only passive ventilation for the greenhouse. The passive ventilation for greenhouse was done by covering the entire poly-sheet greenhouse with shade nets and providing modular windows on a certain height from ground to the greenhouse on poly-sheets. Below figure 5.3 shows the temperature versus time graph for passive ventilation of the greenhouse.

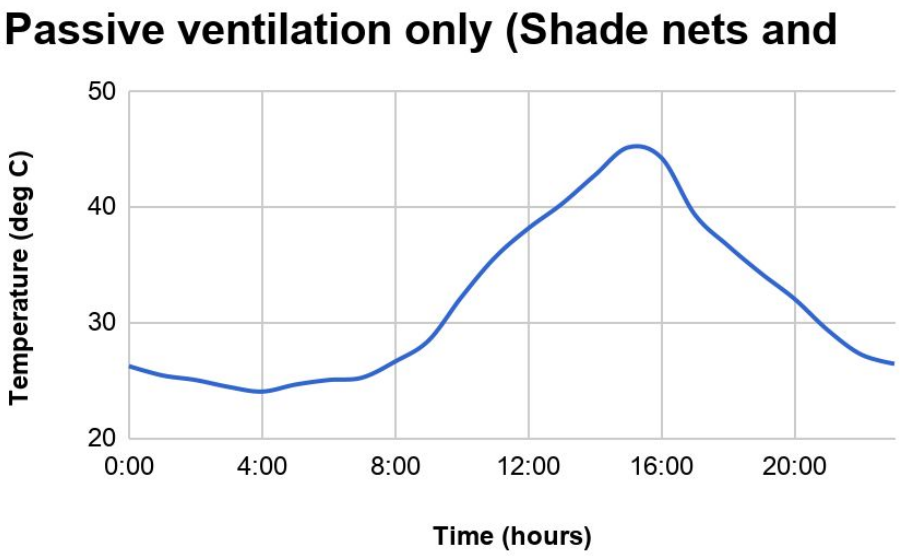


Fig 5.3 Passive ventilation using shade nets

As we can see from the fig 5.3, In open loop passive ventilation using shade nets, the peak greenhouse temperature was brought down from 51.4 deg C to 45.1 deg C. This too seemed to be unfavourable for growing crops inside greenhouse, so we had to resort to both active and passive ventilation and having modular windows for the greenhouse for movement of hot air inside the greenhouse. Following subsection shows the effect of both active and passive ventilation for greenhouse.

5.2.3 Active and passive ventilation temperature data log

After observing the behaviour of temperature variations for active and passive ventilation individually, a third experiment was conducted to include both active and passive ventilation to gain maximum result. The active ventilation was carried out by exhaust fans and passive ventilation was carried out by the modular windows along with shade nets. There was drastic decrease in the temperature which can be shown in the temperature versus time graph in the figure 5.4 below.

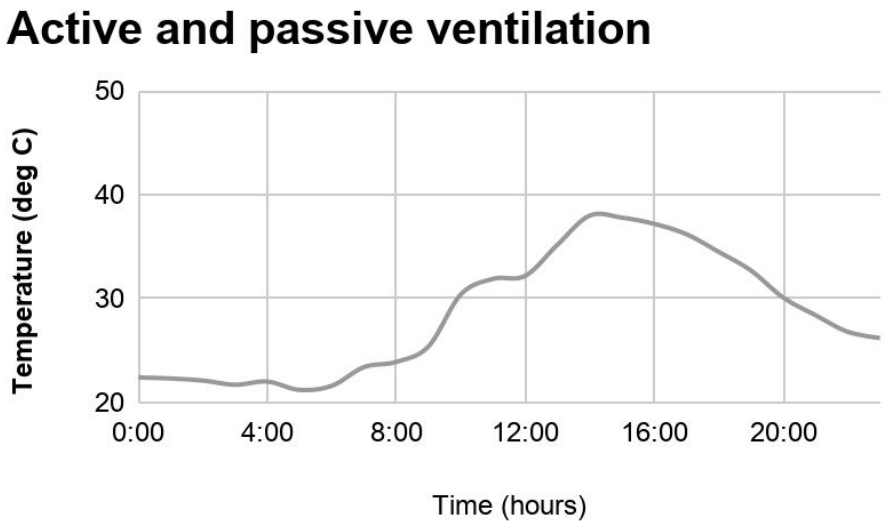


Fig 5.4 Active and Passive ventilation

As we can see from the fig 5.4, when using both closed loop active and passive ventilation using exhaust fans and shade nets respectively, the peak greenhouse temperature was certainly brought down from 51.4 deg C to 38 deg C. This was a drastic change in the temperature variations. Therefore, having a closed loop active and passive ventilation worked out to be the best solution for our high temperature problem. Below graph gives a better understanding for comparing temperature variations for open loop and closed loop system.

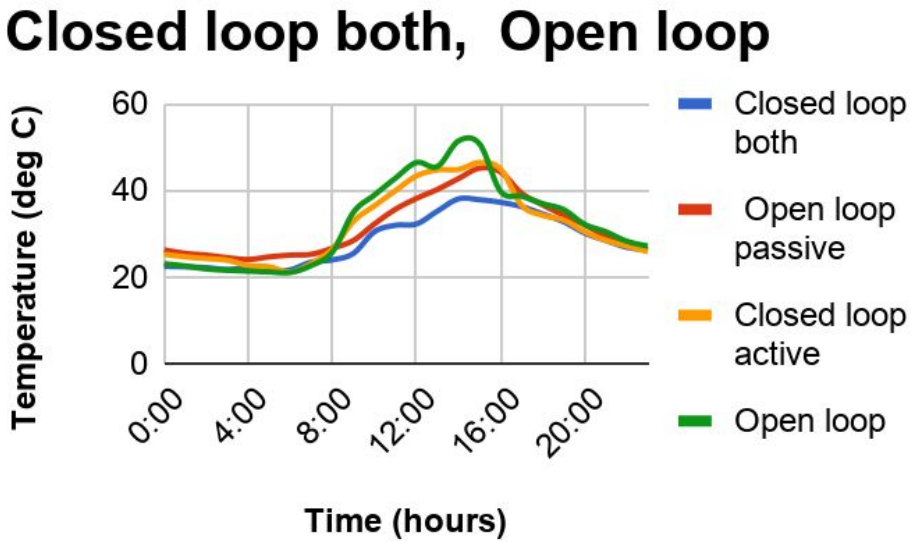


Fig 5.5 Open loop and Closed loop temperature variations

Also, we were able to decrease the temperature by another 3 deg C by having humidity control over the greenhouse which will explained in the following section.

5.3 Humidity data logs: open-loop

Apart from having a control over the temperature of the greenhouse, there was a need to understand the humidity variations inside the greenhouse. The open-loop humidity data logs refers to the behaviour of humidity changes in the greenhouse without any sprinkler mechanism which usually helps in increasing the humidity and will be discussed in next section under closed-loop humidity data logs. Below figure 5.6 shows the humidity versus time graph for the greenhouse throughout the day.

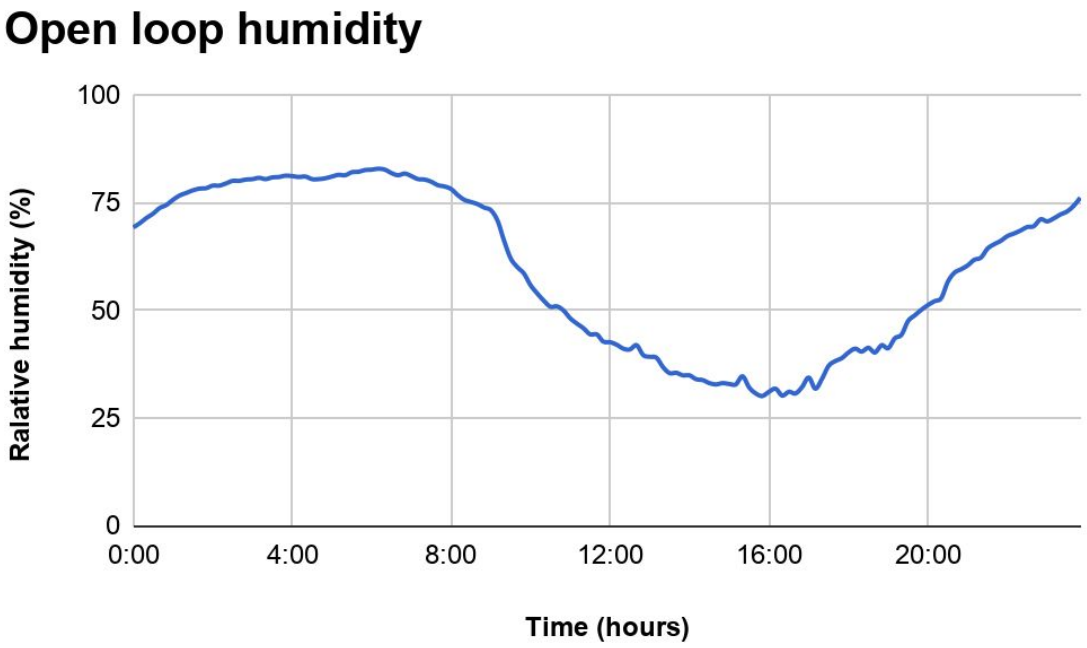


Fig. 5.6 Open loop humidity variations

As we can see from the above humidity graph, the least relative humidity is 30.1% at around 4pm in the afternoon which is not sustainable to grow plants. Therefore, there was a need to increase the humidity by having a sprinkler mechanism using water pipes to increase the relative humidity. Therefore water was sprinkled over gunny bags which usually evaporate water quickly to increase humidity. The sprinkler system was controlled using a

water pump having PWM. The following section shows humidity variation under closed loop condition having sprinkler system.

5.4 Humidity Data log : Closed loop

The close-loop humidity data logs refers to the behaviour of humidity changes in the greenhouse under the effect of sprinkler system. As discussed in the previous open loop humidity section, the sprinkler system helped in increasing the humidity drastically as shown in the below graph.

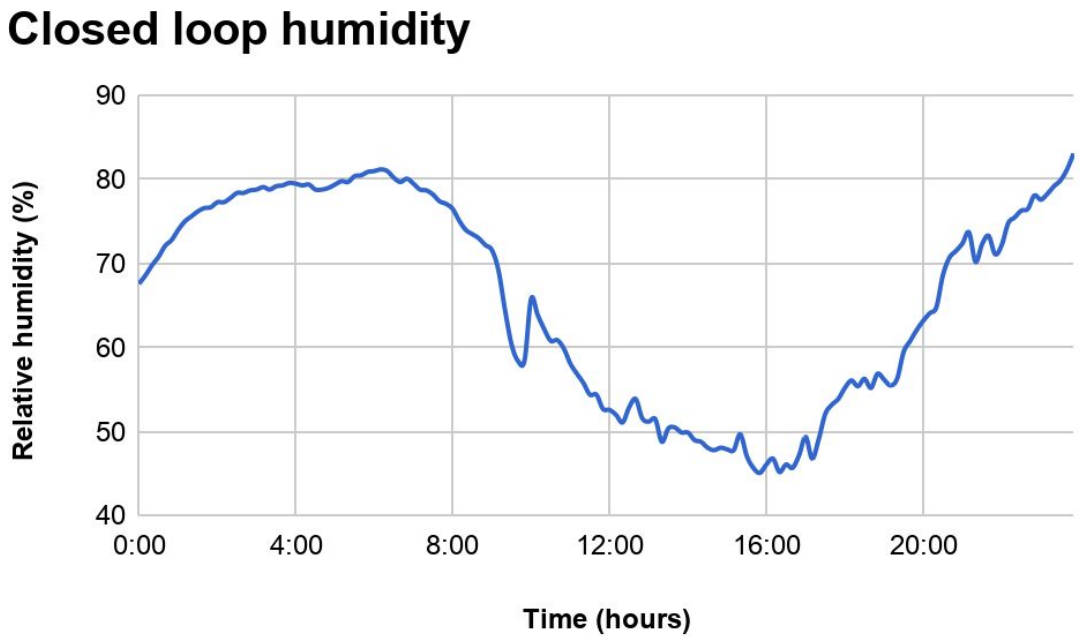


Fig 5.7 Closed loop humidity variations

As we can see from the above Fig 5.7, the least humidity of 30.1% in open loop system was increased to 45.1% in closed loop system. Thus, humidity control for the greenhouse was achieved by having a sprinkler system. Below figure 5.8 shows the comparison of open loop humidity to closed loop humidity to give a better understanding of humidity control.

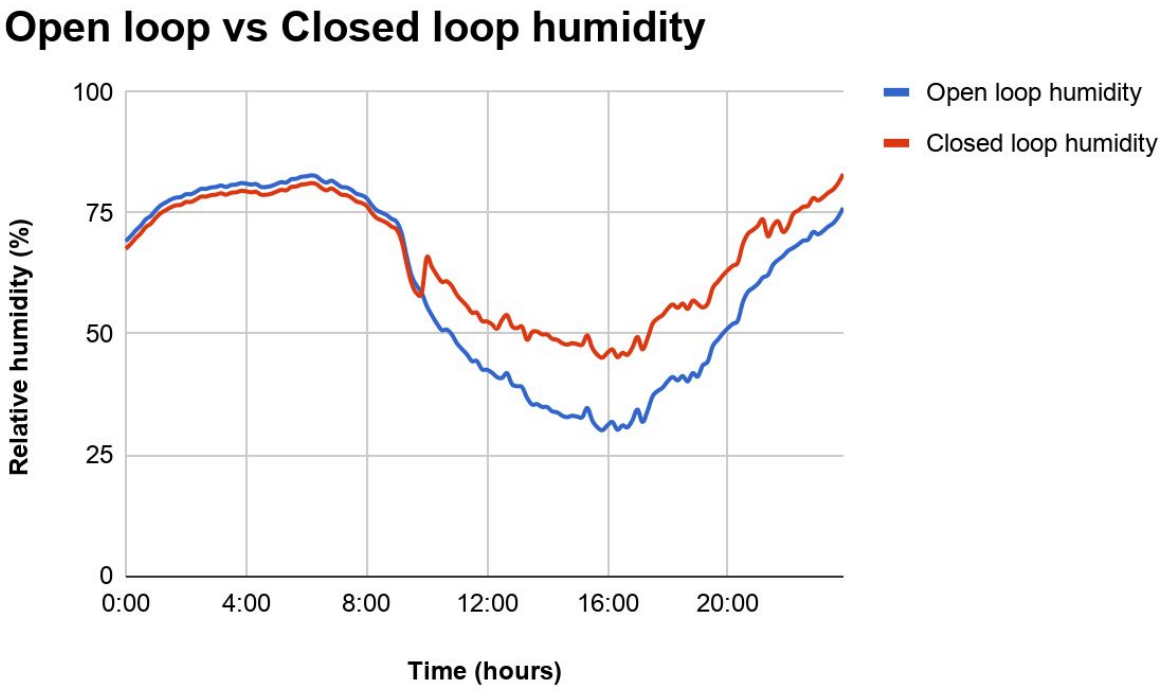


Fig. 5.8 Open loop v/s Closed loop humidity

This closed loop mechanism of humidity control also helped in reducing the greenhouse temperature indirectly. The effect of humidity control on temperature of the greenhouse will be explained in next section.

5.5 Effect of Humidity control on Temperature

We know that temperature and humidity are inversely proportional to each other. So, if the humidity is increased, the temperature certainly falls down. This concept was tested for the greenhouse by increasing the relative humidity. After getting successful results from the closed loop humidity control, the temperature in closed loop control was recorded again. Below figure 5.9 shows the effect of humidity control over temperature of the greenhouse.

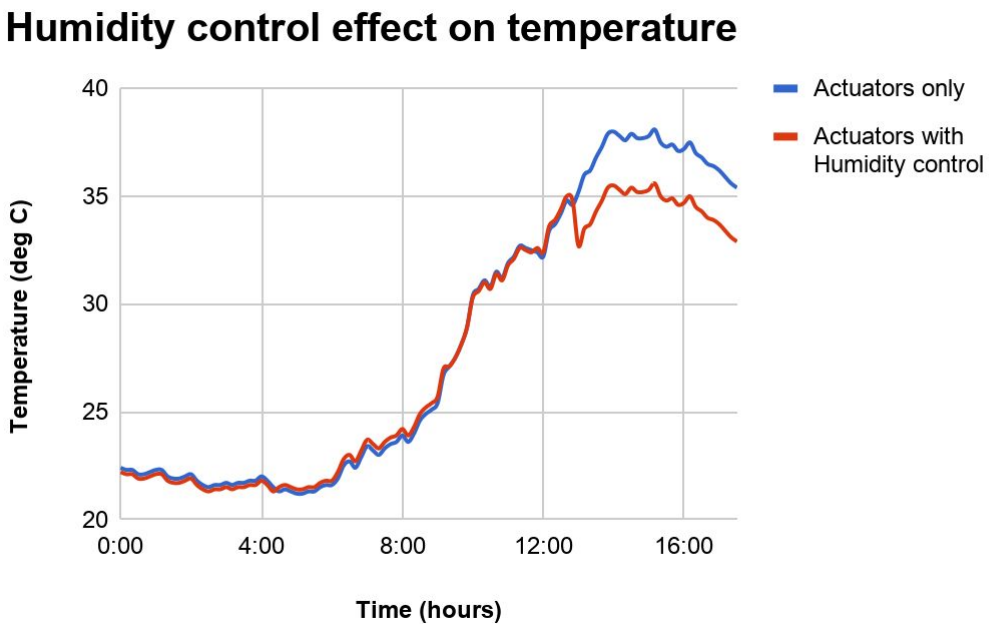


Fig 5.9 Humidity control effect on temperature

As seen from the above figure 5.9, there was a change of around 3 deg C in temperature under the effect of humidity control. Thus, the greenhouse temperature could be brought down to as low as 32 deg C using actuators, shade nets and with the help of humidity control. This is limited only to summer season as the outside temperature is usually high. In winter season, shade nets and humidity control could be avoided as the temperature is usually low.

Chapter 6

Summary of Results and Conclusions

6.1 Summary of Results

1. Completion of the Greenhouse construction as per the designed 3D model by utilizing suitable materials. This structure is designed and and implemented to serve as a platform to perform different experiments pertaining to automation, biotechnology, climate control system and various other fields.
2. Successful interfacing and integration of the complete Hardware architecture consisting of 9 sensors (4 DHT, 4 soil moisture and 1 ultrasonic) and 8 actuators (2 fans, 4 irrigation motors , 1 humidity control motor and 1 solenoid valve).
3. Developed a System software architecture which drives the hardware system components. Successfully Tested and verified the working of the complete Automated system with both SW and HW components. A database system is developed to store all the different parameters from the sensors.
4. An IOT component is developed for the user/owner of the greenhouse to be updated with the current climatic conditions inside the greenhouse with an addition of data logs in the form of attachment to an email from the System to the user's email.The data logs obtained can be used to perform data analytics.

6.2 Conclusion

The main objective of this project was to establish an automated microclimate controlled greenhouse for modular and reliable agriculture. The motivation was to determine

the feasibility of integrating this automated greenhouse which is IOT enabled and to analyse its scalability in the real world scenario.

In this project we have designed a greenhouse structure considering various constraints based on the given plot of land. The materials used in building the structure are easily available, relatively easy to install and cost-effective. As INDIA lies in the tropical zone of temperature, we faced extreme temperatures in the summer due to the heat retention property of the polythene sheet. Hence a modularized method to passively ventilate and reduce the heat entering into the greenhouse using shade nets was implemented.

We have used a set of 4 sensors for each parameter like temperature and humidity to maintain uniformity in data acquisition. All of the 4 soil beds used have an individual soil moisture sensor each to provide independent irrigation cycle. Data from each sensor is recorded in the Raspberry Pi which is later used to validate the working of the control algorithms implemented and also to perform data analytics. The user is then updated regularly about the the current conditions inside the greenhouse via an email. This email also contains the data logs as attachment recorded for 24 hours.

6.3 Future Work

As mentioned above in the results, this project serves as a platform to develop and implement various enhancements to the current project. Some of them can be follow on project with the current infrastructure.

Different control algorithms can be tested out to achieve various level of control over the climate control parameters. An Image processing component to the project can also be developed efficiently due to the use of Raspberry Pi which has amazing onboard data processing capability and an easy camera interface. A sophisticated camera maneuvering system can be implemented (like a Spider cam) for accurate and enhanced image acquisition . Due to the modularity in the construction design of the greenhouse, automation of the ventilation windows and shade nets can be implemented.

Different kinds of growing conditions can be arrived at and the response of the plants to these varied conditions can be studied in detail for different crops at the same time. As each soil bed is individually monitored, plants with different irrigation cycles can be grown under a single system.

REFERENCES:

1. Thangavel Bhuvaneswari, Joshua Tan Hong Yao,"Automated Greenhouse",2014 IEEE International Symposium on Robotics and Manufacturing Automation.

2. Liu, K." Energy Efficiency and Environmental benefits of rooftop gardens",Construction Canada, v. 44 ,no. 2, p. 17, 20-23, March 2002.

3. Ong,B,"Green plot ratio: an ecological measure for architecture and urban planning",Landscape and Urban Planning, 63(4) pp 197- 211, 2003.

4. Automated Greenhouse Monitoring system(temperature and Humidity)Available online:http://sci.uonbi.ac.ke/sites/default/files/cbps/sci/sci/KANG%E2%80%99ALIA.pdf

5. https://howtomechatronics.com/arduino-projects/
6. https://www.raspberrypi.org/

APPENDICES

Appendix A

Raspberry Pi Code

```
#####Libraries

import serial
import csv
from csv import reader
import sys
from serial import SerialException
import time
import datetime
from time import ctime
import smtplib
import mimetypes
from email.mime.multipart import MIMEMultipart
from email import encoders
from email.message import Message
from email.mime.base import MIMEBase
from email.mime.text import MIMEText
####################


#####Serial Communication
try:
        ser = serial.Serial('/dev/ttyACM0',9600,timeout=3000, xonxoff = False, rtscts = False,
dsrdtr = False)
        ser.flushInput()
        ser.flushOutput()
except serial.SerialException:
        try:
        ser = serial.Serial('/dev/ttyACM1',9600,timeout=3000, xonxoff = False, rtscts = False,
dsrdtr = False)
        ser.flushInput()
        ser.flushOutput()
        except serial.SerialException:
        try:
        ser = serial.Serial('/dev/ttyACM2',9600,timeout=3000, xonxoff = False, rtscts = False,
dsrdtr = False)
        ser.flushInput()
        ser.flushOutput()
        except serial.SerialException:
        try:
```

```
                ser = serial.Serial('/dev/ttyACM3',9600,timeout=3000, xonxoff = False, rtscts
= False, dsrdtr = False)
                ser.flushInput()
                ser.flushOutput()
        except serial.SerialException:
                pass
#######################

##########Varialbles

x = 30.0
y = 600
t = 600.0
s1 = 0 ##temperory variable
s2 = 0
s3 = 0
s4 = 0
numofsensors = 4
z = 10.0
##########

#########Creating csv file

dn = "final4.csv"
csv= open(dn,"w")
na = "A,B"
nm = "\"" +na+ "\""
####################

##########sending email function

def sendemail(body):
        emailfrom = "automatedgreenhousePes@gmail.com"
        emailto = "muheenbasha2011@gmail.com"
        fileToSend = "final4.csv"
        username = "automatedgreenhousePes"
        password = "ag9876543210"

        msg = MIMEMultipart()
        msg["From"] = emailfrom
        msg["To"] = emailto
        msg["Subject"] = "Automated greenhouse report"
        msg.preamble = "....."

        body = MIMEText(body)
        msg.attach(body)

        ctype, encoding = mimetypes.guess_type(fileToSend)
        if ctype is None or encoding is not None:
```

```
        ctype = "application/octet-stream"

        maintype, subtype = ctype.split("/", 1)


        fp = open(fileToSend, "rb")
        attachment = MIMEBase(maintype, subtype)
    attachment.set_payload(fp.read())
        fp.close()
    encoders.encode_base64(attachment)
    attachment.add_header("Content-Disposition", "attachment", filename=fileToSend)

        msg.attach(attachment)
        server = smtplib.SMTP("smtp.gmail.com:587")
        server.starttls()
        server.login(username,password)
        server.sendmail(emailfrom, emailto, msg.as_string())
        server.quit()
####################

############Average humidity

def humidity():

        humid = 0
        for i in range(len(data)):
        humid += float(data[i][0])
        if(len(data) > 0):
        print 'avg humidity',round((humid/len(data)),1),"%"
        return
###############

###########Average Temperature

def temperature():
        temp = 0
        for i in range(len(data)):
        temp += float(data[i][1])
        print 'temperature',round(float(data[i][1]),1),"deg C"
        if(len(data) > 0):
        print 'avg temperature',round((temp/len(data)),1),"deg C"
        return
##################




###########Average Moisture
```

```
def moisture():

        moist = 0
        for i in range(len(data)):
        moist = float(data[i][2])
        if(len(data) > 0):
        print 'moisture',moist
        return
###################

########Sending Command to Arduino to receive Data

def send_and_receive( theinput ):
  ser.flushInput()
  ser.flushOutput()
  ser.write( theinput )
  return ser.readline()
  time.sleep(0.1)
##############

#################Data log function

def log_data(t):
        time.sleep(t)
        print 'Starting data log'
        data_raw = send_and_receive('1')
        print 'Data logged'
        print '\n'
        csv.write(data_raw)
        csv.flush()
        with open('final1.csv','r') as f:
        data = list(reader(f))
        csv.write(ctime())
###################

#########Check temperature function

def CheckTemperature():

        with open('final4.csv','r') as f:
        data = list(reader(f))
        if(((float(data[len(data)-1][1]) + float(data[len(data)-1][2]) +
float(data[len(data)-1][3]) + float(data[len(data)-1][4]))/numofsensors) > x):
        ser.flushInput()
        ser.flushOutput()
        ser.write('2')
        print 'The Temperature is',round(((float(data[len(data)-1][1]) +
float(data[len(data)-1][2]) + float(data[len(data)-1][3]) +
float(data[len(data)-1][4]))/numofsensors),1),"deg C :","Switching On Fan"
        print "\n"
```

```
        if(((float(data[len(data)-1][1]) + float(data[len(data)-1][2]) +
float(data[len(data)-1][3]) + float(data[len(data)-1][4]))/numofsensors) < x):
        ser.flushInput()
        ser.flushOutput()
        ser.write('3')
        print 'The Temperature is',round(((float(data[len(data)-1][1]) +
float(data[len(data)-1][2]) + float(data[len(data)-1][3]) +
float(data[len(data)-1][4]))/numofsensors),1),"deg C :","Switching Off Fan"
        print '\n'
################

############Check moisture

def CheckMoisture():

        with open('final4.csv','r') as f:
        data = list(reader(f))
        if((float(data[len(data)-1][9])) > y):
        ser.flushInput()
        ser.flushOutput()
        ser.write('4')
        s1 = round((100-((float(data[len(data)-1][9]) - 400)/900)*100),1)
        print 'Moisture level of soil bed 1 is',s1,"% :","Starting Drip Irrigation"

        if((float(data[len(data)-1][9])) < y):
        ser.flushInput()
        ser.flushOutput()
        ser.write('0')
        s1 = round((100-((float(data[len(data)-1][9]) - 400)/900)*100),1)

        print 'Moisture level of soil bed 1 is',round(s1,1),"% :"

        if((float(data[len(data)-1][10])) > y):
        ser.flushInput()
        ser.flushOutput()
        ser.write('5')
          s2 = (100-((float(data[len(data)-1][10]) - 400)/900)*100)
        print 'Moisture level of soil bed 2 is',round(s2,1),"% :","Starting Drip Irrigation"

        if((float(data[len(data)-1][10])) < y):
        ser.flushInput()
        ser.flushOutput()
         ser.write('0')
        s2 = (100-((float(data[len(data)-1][10]) - 400)/900)*100)
        print 'Moisture level of soil bed 2 is',round(s2,1),"% :"

        if((float(data[len(data)-1][11])) > y):
        ser.flushInput()
```

```
            ser.flushOutput()
        ser.write('6')
            s3 = (100-((float(data[len(data)-1][11]) - 400)/900)*100)
            print 'Moisture level of soil bed 3 is',round(s3,1),"% :","Starting Drip Irrigation"

            if((float(data[len(data)-1][11])) < y):
            ser.flushInput()
             ser.flushOutput()
            ser.write('0')
            s3 = (100-((float(data[len(data)-1][11]) - 400)/900)*100)
            print 'Moisture level of soil bed 3 is',round(s3,1),"% :"

            if((float(data[len(data)-1][12])) > y):
            ser.flushInput()
                ser.flushOutput()
            ser.write('7')
            s4 = (100-((float(data[len(data)-1][12]) - 400)/900)*100)
            print 'Moisture level of soil bed 4 is',round(s4,1),"% :","Starting Drip Irrigation"

            if((float(data[len(data)-1][12])) < y):
             ser.flushInput()
            ser.flushOutput()
            ser.write('0')
            s4 = (100-((float(data[len(data)-1][12]) - 400)/900)*100)
            print 'Moisture level of soil bed 4 is',round(s4,1),"% :"
################




#########Checking water level

def CheckWaterLevel():

            if((float(data[len(data)-1][13])) > z):
            ser.flushInput()
            ser.flushOutput()
            ser.write('8')
            if((float(data[len(data)-1][13])) < z):
            ser.flushInput()
            ser.flushOutput()
            ser.write('0')


############Sending Mail

def MailData():

            with open('final4.csv','r') as f:
```

```
        data = list(reader(f))
        if((len(data) % 20) == 0):
        print len(data)
        temp = 0
        for i in range(len(data)):
        temp += float(data[i][2])

        body = (("Greenhouse Temperature            :
")+str(round(((float(data[len(data)-1][1]) + float(data[len(data)-1][2]) +
float(data[len(data)-1][3]) + float(data[len(data)-1][4]))/numofsensors),1))+(" deg
C")+("\n")+
          ("Greenhouse Humidity                :  ")+str(round(((float(data[len(data)-1][5]) +
float(data[len(data)-1][6]) + float(data[len(data)-1][7]) +
float(data[len(data)-1][8]))/numofsensors),1))+("%")+("\n")+("\n")+
                ("Moisture Level of Soil bed 1          :
")+str(round((100-((float(data[len(data)-1][9]) - 400)/900)*100),1))+("%")+("\n")+
                ("Moisture Level of Soil bed 2          :
")+str(round((100-((float(data[len(data)-1][10]) - 400)/900)*100),1))+("%")+("\n")+
                ("Moisture Level of Soil bed 3          :
")+str(round((100-((float(data[len(data)-1][11]) - 400)/900)*100),1))+("%")+("\n")+
                ("Moisture Level of Soil bed 4          :
")+str(round((100-((float(data[len(data)-1][12]) - 400)/900)*100),1))+("%"))

        sendemail(body)
        print "Status Report Mail Sent"
        print '\n'
#######Main Loop

while 1:

        log_data(t)
        CheckTemperature()
        with open('final4.csv','r') as f:
        data = list(reader(f))
        print 'The Average Humidity is',round(((float(data[len(data)-1][5]) +
float(data[len(data)-1][6]) + float(data[len(data)-1][7]) +
float(data[len(data)-1][8]))/numofsensors),1),"%"
        print '\n'
        CheckMoisture()
        CheckWaterLevel()
        MailData()
```

Appendix B

Arduino Code

```
#include <SoftwareSerial.h>
#include "DHT.h"
#define trigPin 13
#define echoPin 12
DHT dht1(22, DHT22);
DHT dht2(28,DHT22);
DHT dht3(34,DHT22);
DHT dht4(40,DHT22);
char ch;

void setup()
{
  float t;
  Serial.begin(9600);
  pinMode(2,OUTPUT); //relay 1
  pinMode(4,OUTPUT); //relay 2
  pinMode(8,OUTPUT); //motor 1
  pinMode(10,OUTPUT); //motor 2
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  dht1.begin();
  dht2.begin();
  dht3.begin();
  dht4.begin();
}

void loop()
{
  delay(1000);
  long duration, d;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  d = (duration/2) / 29.1;
  int s1 = analogRead(A2);
  int s2 = analogRead(A6);
  int s3 = analogRead(A10);
  int s4 = analogRead(A14);
```

```
//temp 1
float h1 = dht1.readHumidity();
float t1 = dht1.readTemperature();
float f = dht1.readTemperature(true);

//temp 2
float h2 = dht2.readHumidity();
float t2 = dht2.readTemperature();

//temp 3
float h3 = dht3.readHumidity();
float t3 = dht3.readTemperature();

//temp 4
float h4 = dht4.readHumidity();
float t4 = dht4.readTemperature();

if (isnan(h1) || isnan(t1) || isnan(f))
{
 Serial.println("-1");
 return;
}
 if (isnan(h2) || isnan(t2) || isnan(f))
{
 Serial.println("-2");
 return;
}
 if (isnan(h3) || isnan(t3) || isnan(f))
{
 Serial.println("-3");
 return;
}
 if (isnan(h4) || isnan(t4) || isnan(f))
{
 Serial.println("-4");
 return;
}
///////////////


if (Serial.available())
{
 ch = Serial.read();
 switch(ch)
 {
   case '1':
   {
     Serial.print("");
```

```
    Serial.print("");
    Serial.print(",");
    Serial.print(t1);
    Serial.print(",");
    Serial.print(t2);
    Serial.print(",");
    Serial.print(t3);
    Serial.print(",");
    Serial.print(t4);
    Serial.print(",");

    Serial.print(h1);
    Serial.print(",");
    Serial.print(h2);
    Serial.print(",");
    Serial.print(h3);
    Serial.print(",");
    Serial.print(h4);
    Serial.print(",");

    Serial.print(s1);
    Serial.print(",");
    Serial.print(s2);
    Serial.print(",");
    Serial.print(s3);
    Serial.print(",");
    Serial.print(s4);

    Serial.print(",");
    Serial.print(d);

    Serial.println("");

    break;
   }

  case '2':
   {

    digitalWrite(2,HIGH);
    break;
   }

   case '3':
   {
    digitalWrite(2,LOW);
    break;
   }
```

```
//soil 1
case '4':
{

  if(s1 > 600)
   {
    analogWrite(8,250);
    delay(60000);
    s2 = analogRead(A2);
    analogWrite(8,0);
   }

   break;
}

//soil 2
case '5':
{

  if(s2 > 600)
   {
    analogWrite(8,250);
    delay(60000);
    s2 = analogRead(A6);
    analogWrite(8,0);
   }

   break;
}


//soil 3
case '6':
{

  if(s3 > 600)
   {
    analogWrite(9,250);
    delay(60000);
    s2 = analogRead(A10);
    analogWrite(9,0);
   }

   break;
}

//soil 4
```

```
   case '7':
   {

     if(s4 > 600)
      {
       analogWrite(9,250);
       delay(60000);
       s2 = analogRead(A14);
       analogWrite(9,0);
      }

      break;
   }

   //water level
    case '8':
     {

      digitalWrite(4,HIGH);
      break;
     }

     case '9':
     {
      digitalWrite(4,LOW);
      break;
     }

     default:
     {
      digitalWrite(2,LOW);
      digitalWrite(4,LOW);
      analogWrite(8,0);
      analogWrite(10,0);
     }
   }
   Serial.flush();
  }

}
```

Word Count: 11517

Plagiarism Percentage 14%

Matches

1 **World Wide Web Match**
View Link

2 **World Wide Web Match**
View Link

3 **World Wide Web Match**
View Link

4 **World Wide Web Match**
View Link

5 **World Wide Web Match**
View Link

6 **World Wide Web Match**
View Link

7 **World Wide Web Match**
View Link

8 **World Wide Web Match**
View Link

9 **World Wide Web Match**
View Link

10 **World Wide Web Match**
View Link

11 **World Wide Web Match**