SMART AGRICULTURE MONITORING SYSTEM

2015

System Proposed By:

Shreya Sinha (ABV-IIITM, Gwalior)
Ishita Roy (NMIT, Bangalore)
Srishti Gupta (IGDTUW, Delhi)
Shaurya Gupta (NSIT, Delhi)
Inderjeet Kaur (AKGEC, Ghaziabad)

SMART AGRICULTURE MONITORING SYSTEM

A SUMMER TRAINING PROJECT REPORT

Submitted by

Shreya Sinha (ABV-IIITM, Gwalior)
Ishita Roy (NMIT, Bangalore)
Srishti Gupta (IGDTUW, Delhi)
Shaurya Gupta (NSIT, Delhi)
Inderjeet Kaur (AKGEC, Ghaziabad)

Under the guidance of

Dr. SRN Reddy,

Associate Professor, IGDTUW

in partial fulfilment for the award of the certificate in

"Build Your Smart Device"

Organized by CSE Dept., **Indira Gandhi Delhi Technical University for Women**
in collaboration with

Microsoft

(From 15-06-2015 to 24-07-2015)

Acknowledgement

This project consumed huge amount of work, research and dedication. Still, implementation would not have been possible if we did not have a support of many individuals and organizations. Therefore we would like to extend our sincere gratitude to all of them.

First of all we are thankful to Indira Gandhi Delhi Technical University for Women (IGDTUW), Delhi for its technological and logistical support and for providing necessary resources concerning projects implementation.

We are also grateful to Dr. S.R.N Reddy for provision of expertise, and technical support in the implementation. Without his superior knowledge and experience, the Project would lack in quality of outcomes, and thus his support has been essential.

We would like to express our sincere thanks towards the project staff that devoted their time and knowledge in the implementation of this project.

Nevertheless, we express our gratitude toward our families and colleagues for their kind co-operation and encouragement which help us in completion of this project.

Regards,

Developers of the Smart Device

Certificate

Certified that the Project Work entitled

"Smart Agriculture Monitoring System"

submitted by

Shreya Sinha (ABV-IIITM, Gwalior)
Ishita Roy (NMIT, Bangalore)
Shaurya Gupta (NSIT, Delhi)
Srishti Gupta (IGDTUW, Delhi)
Inderjeet Kaur (AKGEC, Ghaziabad)

for the fulfilment of **6th Summer Training Program** titled **"Build Your Smart Device"** offered by **Indira Gandhi Delhi Technical University for Women (IGDTUW), Delhi** is an original work carried out by the student under my supervision in the laboratory.

Date:

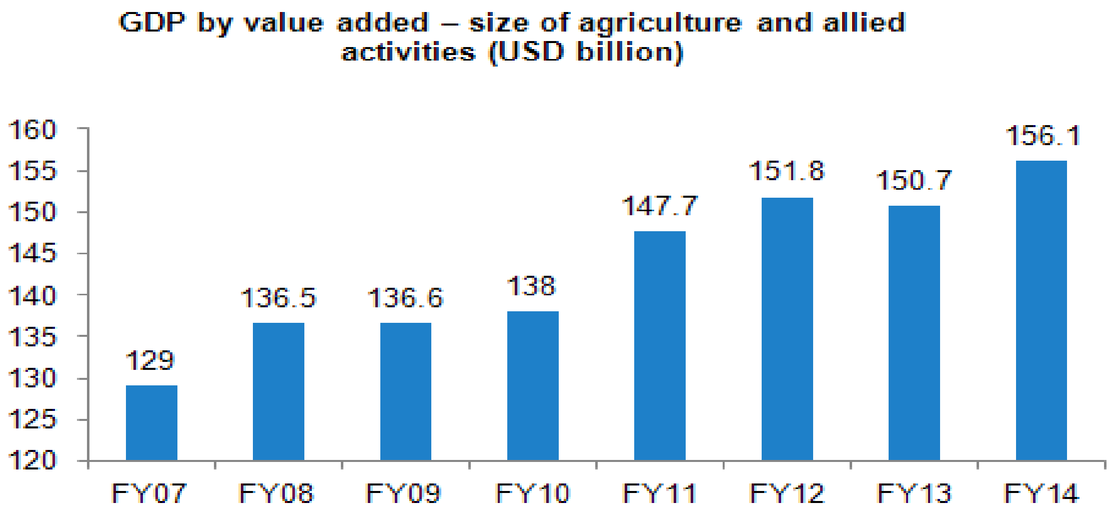
(Signature of the Supervisor)

Index

Introduction

Despite the focus on industrialisation, agriculture remains a dominant sector of the Indian economy both in terms of contribution to gross domestic product (GDP) as well as a source of employment to millions across the country.

Agriculture plays a vital role in the Indian economy. Over 70 per cent of the rural households depend on agriculture as their principal means of livelihood. Agriculture, along with fisheries and forestry, accounts for one-third of the nation's GDP and is its single largest contributor.



Source: Ministry of Agriculture, Print Release, Aranca Research Notes: GDP – Gross Domestic Product, CSO – Central Statistical Organisation CAGR: 2.8%

Highlights of Indian Agriculture sector

• GDP of agriculture and allied sectors in India was recorded at US$ 156.1 billion in FY14

• According to the advanced estimates of Central Statistical Organisation, agriculture and allied sector recorded a growth of 3.6 per cent in FY14

• From FY 07 –14, agriculture and its allied services grew at a CAGR of 2.8 per cent

• Agriculture is the primary source of livelihood for about 58 per cent of India's population

Although, Green Revolution has brought a positive impact in India, but on the other hand it has also resulted in negative impact. One of the biggest impacts is soil exhaustion which means depletion of nutrients in the soil due to farming of same crops again and again. Soil exhaustion generally takes place in rain forest areas.

Need for Monitoring Agriculture

When practiced without care, farming presents the greatest threat to species and ecosystems. Inappropriate agricultural practices and land use can also have an adverse impact on natural resources, like pollution of soil, water and air.

Agriculturists are confronting new difficulties and opportunities consistently to produce more food crops due to increase in global population about 9 billion by 2050. In farming ecological variables, for example, temperature, mugginess, soil dampness are fundamental requirements which have an impact on the development rate, profit of produce, sugar substance of natural product, acridity and so forth. This work deals with the above-mentioned natural variables effectively, so as to enhance crop production and reduce burden on the farmer.

Due to less availability of labour, the percentage of agriculture has been reduced recently. Hence, many cultivated lands are converted into buildings by industry or organization. Various researchers are currently working in computerized agriculture for simplified control over the farm trying to reduce the farmer burden and technically minimize human intervention in the agricultural processes by replacing them with computer programs.

We need a system decide whether wheat shall be grown this year or paddy. The system shall decide how much area will be given for which crop. The system shall decide how much of the crop shall be allowed to come to the market and how much of that shall be destroyed to prevent distress sales.

Should rain-deficient and groundwater deficient Rajasthan grow a water intensive crop like sugar cane and put tremendous stress on the scarce water resources of the state?

The system shall also decide whether a particular agricultural area should be kept fallow or allowed to be cultivated.

History records the benefits of irrigation, as well as the dismal failures caused by irrigation mismanagement. Today we are faced with dwindling water supplies, loss of water resources from pollution and permanent damage to land resources, in many cases due to mismanaging our most precious resources.

Thus, a system is required to so that agricultural practices and the environment both are sustained.

The Problem Statement

Although India is the second largest irrigated country of the world after China, only one-third of the cropped area is under irrigation. Irrigation is the most important agricultural input in a tropical monsoon country like India where rainfall is uncertain, unreliable and erratic India cannot achieve sustained progress in agriculture unless and until more than half of the cropped area is brought under assured irrigation.

However, care must be taken to safeguard against ill effects of over irrigation especially in areas irrigated by canals. Large tracts in Punjab and Haryana have been rendered useless due to faulty irrigation. In the Indira Gandhi Canal command area also intensive irrigation has led to sharp rise in sub-soil water level, leading to water-logging, soil salinity and alkalinity.

Proposed System

1) Priority to enhance yield in the agriculture sector.
2) Technological advancements necessary.
3) Our project aims at doing just this.
    - Monitoring helps in early detection.
    - Helps in identifying the deficiency of particular substance.
    - Replenishment enhances the soil quality and hence the crop production.
    - Keeping environmental degradation in check

Research papers published on similar systems

- www.scirp.org/Journal/PaperDownload.aspx?paperID=18950
- https://www.mdpi.com/1424-8220/14/10/19639/pdf
- www.s2is.org/ICST-2014/papers/1569977701.pdf
- www.ijens.org/vol_12_i_02/124402-5757-ijet-ijens.pdf

Limitations of the current systems

1. No commercial product already present
2. Only research papers published
3. Several basic prototypes available
4. No product has all the sensors incorporated

Benefits of the Proposed System

1) The proposed system will be a boon for farmers as it will:

   - Provide them with a handy kit to monitor the status of the soil thereby enabling them to add the required components only when required.

   - Reducing their visits to the laboratory to get the soil samples analyzed.

2) The system will also help in effective environmental monitoring by ensuring that excess of any component is not being added to soil unnecessarily.

3) In the proposed system, the drawbacks are overcome by using a small credit card sized computer called Raspberry Pi. Our device can be placed directly in the field and it is connected to all the necessary tools and also it sends the data to a database. By this proposed system we can reduce the cost, less maintenance and water can be used very efficiently and effectively.

The Smart Agriculture Model

The Smart Agriculture models allow monitoring multiple environmental parameters involving a wide range of applications. It has been provided with sensors for air and soil temperature and humidity. The main applications for this model are precision agriculture, irrigation systems, greenhouses, weather stations, etc.

The following are its advantages:

•Produce better yield and quality

•Optimize production

•Reduce water waste

•Manage energy efficiently

•Maximize nutrient use

•Mitigate excessive runoff and deep percolation loss

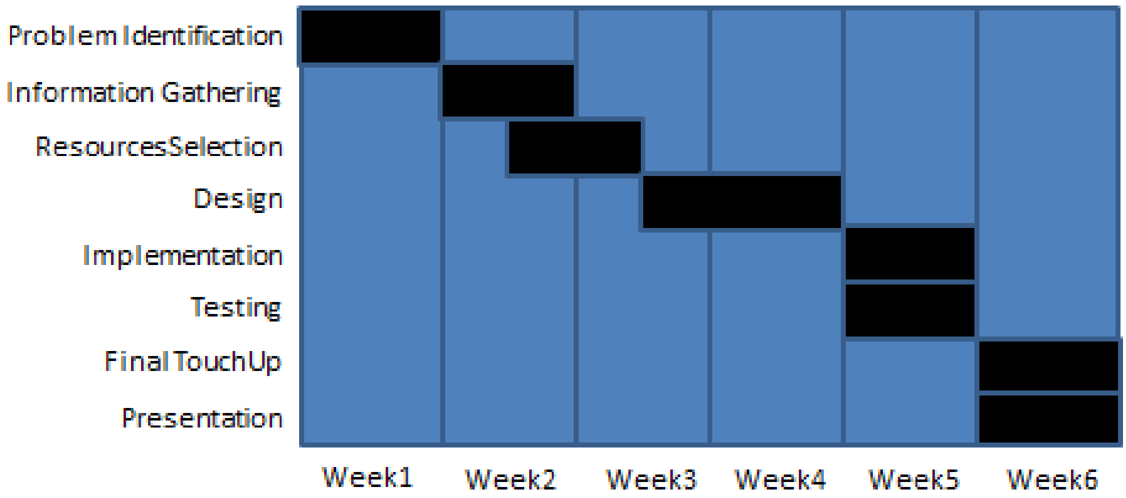
•Help protect water quality

In short, more money in the bank at harvest.

Project Plan

The following Gantt chart represents the entire project plan for developing the Smart Agriculture Monitoring System.

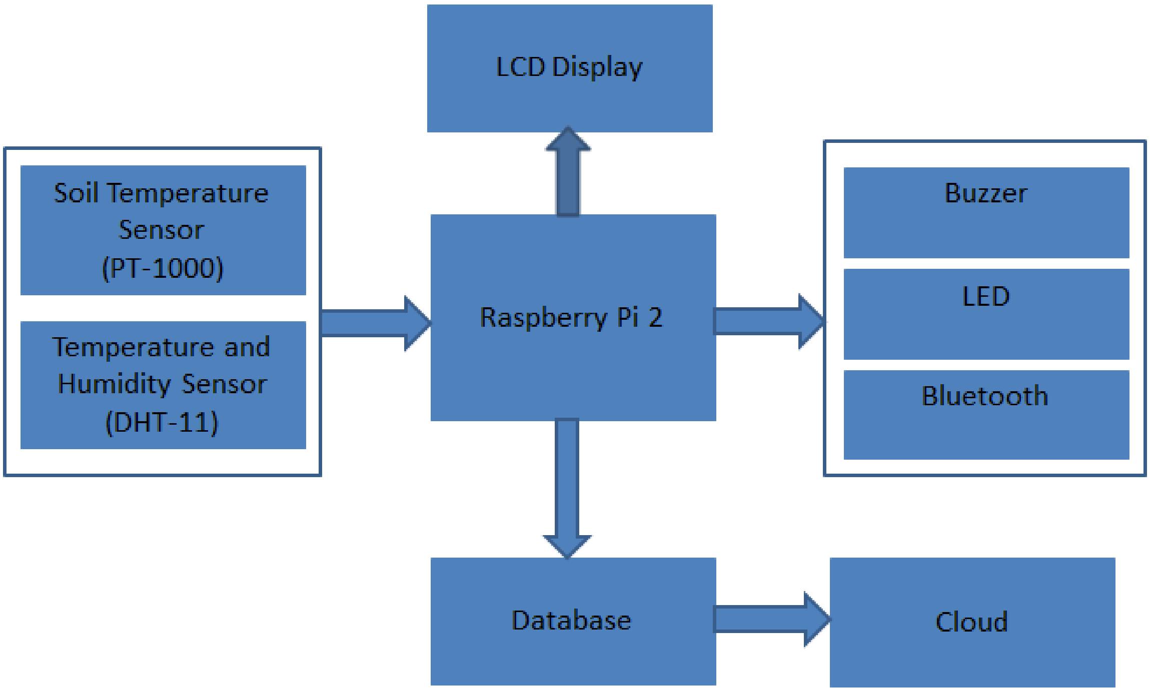
The entire duration of six weeks of product development have split up as per the given diagram.



Overview of the Proposed System

Input	Processor	Transmission	Output
1) Soil Temperature sensor 2) Temperature and Humidity Sensor	Raspberry Pi 2	1) Wired: Cables 2) Wireless: Bluetooth	1) Display 2) Sound 3) Sensor Readings 4) Cloud (through Database) 5) LED

Basic Sensor Interfacing Diagram

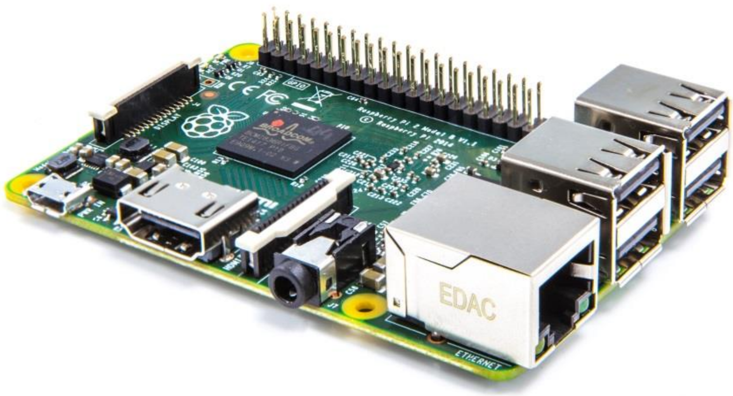


List of components used

- Raspberry Pi 2
- PT-1000 Soil Temperature Sensor
- DHT-11 Temperature and Humidity Sensor
- Buzzer
- Buttons
- LCD Screen
- ADC
- Potentiometer
- Resistor
- Switch Button
- Bluetooth
- LED

Interfacing of Major Components

1) Raspberry Pi 2



FEATURES OF RASPBERRY PI2 (BCM2836 SoC)

A. Chip : Broadcom BCM2836 SoC

B. Core architecture: Quad-core ARM Cortex-A7

C. CPU : 900 MHz

D. GPU: Dual Core VideoCore IV® Multimedia Co-Processor, Provides Open GL ES 2.0, hardware-accelerated OpenVG, and 1080p30, H.264 high-profile decode, Capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs with texture filtering and DMA infrastructure

E. Memory: 1GB LPDDR2

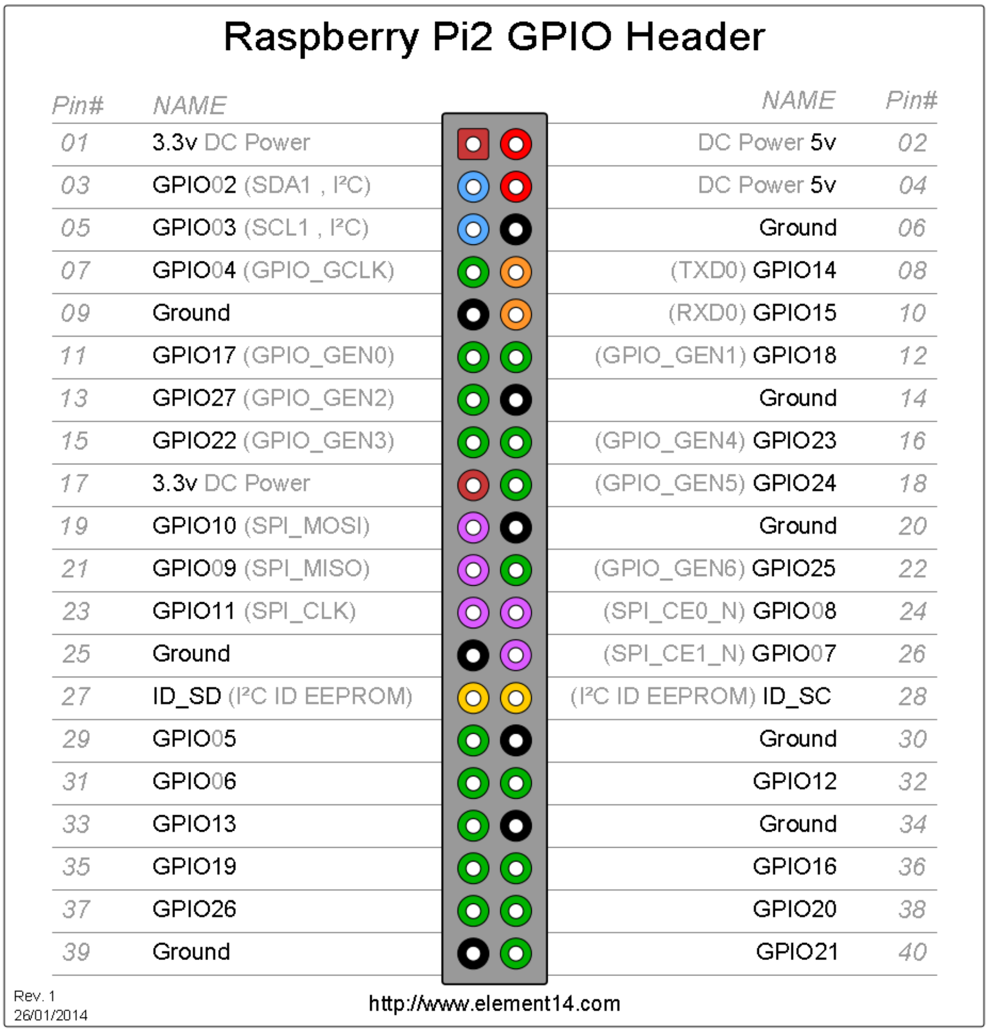
Raspberry Pi is a little PC, at a reasonable cost. This makes it ideal for agribusiness where a little gadget can undoubtedly be set for a situation and mounted inside an electrical box. The Raspberry Pi board contains a processor and design chip, program memory (RAM –

Random-Access Memory) and different interfaces and connectors for outer gadgets. It is low-cost, effective and it consumes less power. Raspberry Pi works as a standard PC.

Central to Raspberry Pi 2 is a new Broadcom chip, BCM2836, for which the Foundation is lead customer.

User interface response is snappier, applications load faster, and tools run faster. It is mostly quantitative rather than qualitative, except with video, where quantitative becomes qualitative.

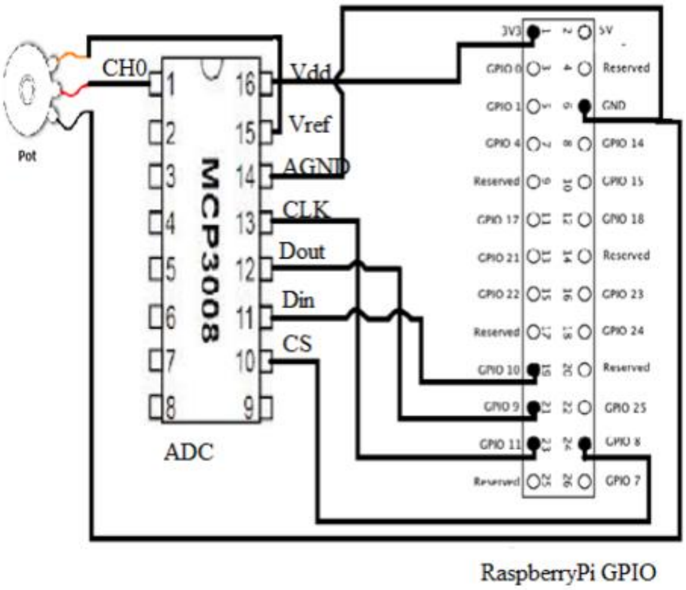
Pin Diagram



2) MCP3208 ADC WITH RASPBERRY PI 2

The MCP3208 12-bit Analog-to-Digital Converter (ADC) combines high performance and low power consumption in a small package, making it ideal for embedded control applications. The MCP3208 features a successive approximation register (SAR) architecture and an industry-standard SPI™ serial interface, allowing 12-bit ADC capability to be added to any PICmicro® microcontroller. The MCP3208 features 100k samples/second, 8 input channels, low power consumption (5nA typical standby, 400 µA max.active), and is available in 16-pin PDIP and SOIC packages. Applications for the MCP3208 include data acquisition, instrumentation and measurement, multi-channel data loggers, industrial PCs, motor control, robotics, industrial automation, smart sensors, portable instrumentation and home medical appliances.

Interfacing Diagram



Requirements and Source Code

The Raspberry Pi GPIO connector has only digital inputs. If we wish to measure a voltage, we need to use a separate analog-to-digital converter (ADC). Use the MCP3008 eight-channel ADC chip. This chip actually has eight analog inputs, so we can connect up to eight sensors to one of these and interface to the chip using the Raspberry Pi SPI interface.

To make this, we need:

- Breadboard and jumper wires
- MCP3208 eight-channel ADC IC
- 10kΩ trimpot
- Ethernet Cable

1. Download image from link below for Raspbian i.e. Debian OS for Raspberry Pi.

    http://www.raspberrypi.org/downloads

2. Write this image into SD Card using Win32 disk imager. Download disk imager from link below.

    http://sourceforge.net/projects/win32diskimager/

3. Download remote SSH client Putty from link below.

    http://putty.en.softonic.com/

4. For desktop view on remote client download Xming from link below.

    http://sourceforge.net/projects/xming/

5. For detecting ip addresses of clients present in a network download Netscanner from link below.

    http://download.cnet.com/Net-Scan/3000-2112_4-10580446.html

6. Put SD Card into Raspberry Pi"s SD card slot, connect Ethernet cable and power on raspberrypi board.
7. Connect your PC with Ethernet cable (from same modem), and start net scanner.
8. Netscanner will scan your clients present into that particular modem and give ip address of raspberrypi board.
9. Run Xming

10. Run Putty with host address of raspberrypi as given by netscanner, with SSH->X11 enable.

11. Default login id for raspberrypi is pi and password raspberry. To view desktop on remote PC write on command line

    $ startlxde

12.Raspberry default desktop will be open at your PC then.

13. Open LX terminal on raspberrypi‟s desktop.

14. Load the SPI device into raspberrypi using command prompt, type command

    $ modprobe spidev

15. Check 'spidev' installation using command

    $ lsmod

16. Install the 'git' package for enabling installation of other packages from Github using command

    $ sudo apt-get install git-core

17. To compile 'py-spidev package, install python-dev package using command

    $ sudo apt-get install python-dev

18. Download and compile py-spidev' package, using commands

    mkdir py-spidev

    cd py-spidev

    wget https://raw.github.com/doceme/py-spidev/master/setup.py

    wget https://raw.github.com/doceme/py-pidev/master/spidev_module.c

    sudo python setup.py install

19. Interface ADC MCP3008 as per circuit diagram.

20. Write python source code on leafpad editor. Save using .py extension.

21. Run desired .py extension file on LX terminal using command

    $ sudo python filename.py

22. Values of voltages as input to ADC can be seen at LX terminal.

```python
import spidev, time


spi = spidev.SpiDev()

spi.open(0, 0)


def analog_read(channel):

        r = spi.xfer2([1, (8 + channel) << 4, 0])

        adc_out = ((r[1]&3) << 8) + r[2]

        return adc_out


while True:

    reading = analog_read(0)

    voltage = reading * 3.3 / 1024

    print("Reading=%d /t Voltage=%f" % (reading, voltage))

    time.sleep(1)
```

3) SOIL TEMPERATURE SENSOR (PT-1000)

Specifications

Measurement range: -50 ~ 300ºC

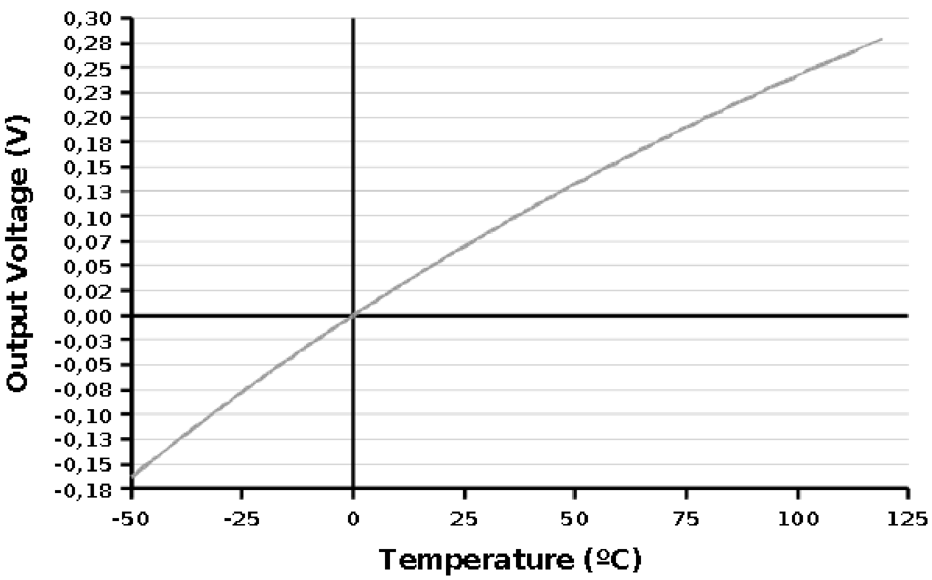
Accuracy: DIN EN 60751

Resistance (0ºC): 1000Ω

Diameter: 6mm

Length: 40mm

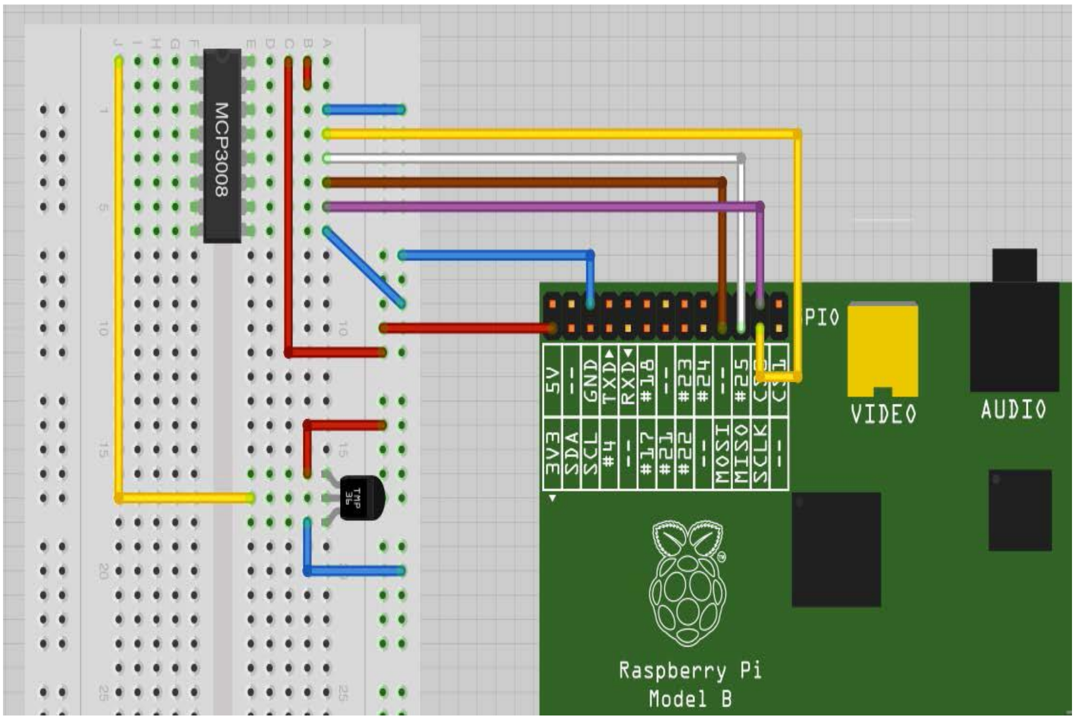
Cable: 2m



Measurement Process

The resistance of the PT1000 sensor varies between approximately 920Ω and 1200Ω in the range considered useful in agriculture applications (-20 ~ 50ºC approximately), which results in too low variations of voltage at significant changes of temperature for the resolution of the Wasp mote's analog-to-digital converter. The function of the library read Value returns the temperature value in Celsius degree (ºC). The power supplies required by the sensor, both 3.3V and 5V, are controlled through a digital switch that can be opened or closed via software activating the digital pin DIGITAL5. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board.

Interfacing Diagram



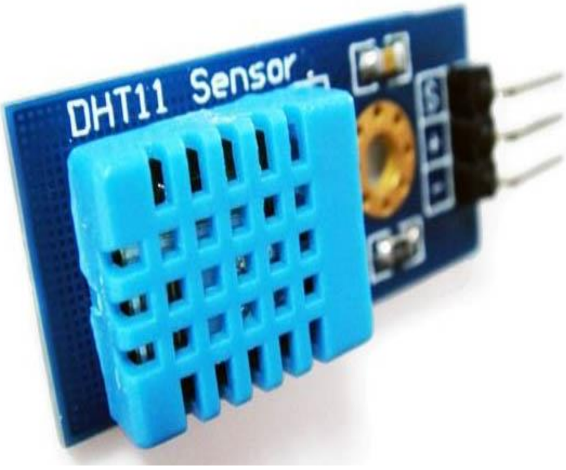
Requirements and Source Code

To make this, we need:

- MCP3208 ADC
- PT-1000 Soil Temperature Sensor
- Resistor

```
import spidev, time


spi = spidev.SpiDev()

spi.open(0, 0)


def analog_read(channel):

        r = spi.xfer2([1, (8 + channel) << 4, 0])

        adc_out = ((r[1]&3) << 8) + r[2]

        return adc_out


while True:

        reading = analog_read(0)

        voltage = reading * 3.3 / 1024

        temp_c = voltage * 100 - 50

        temp_f = temp_c * 9.0 / 5.0 + 32

        print("Temp C=%f /t /tTemp f=%f" % (temp_c, temp_f))

        time.sleep(1)
```

4) TEMPERATURE AND HUMIDITY SENSOR (DHT-11)



Digital temperature and humidity sensor is a composite Sensor contains a calibrated digital signal output of the temperature and humidity. Application of dedicated digital modules collection technology and the temperature and humidity sensing technology, to ensure that the product has high reliability and excellent long-term stability.

The DHT11 is a relatively cheap sensor for measuring temperature and humidity. The DHT11 has three lines: GND, +5V and a single data line. By means of a handshake, the values are clocked out over the single digital line.

Relative humidity is a measure of the amount of water vapour contained within the air which is usually expressed as percentage humidity. Humidity is a very important environmental element that must be controlled for healthy plants. It controls the rate of transpiration and how the nutrients are received by the plant.

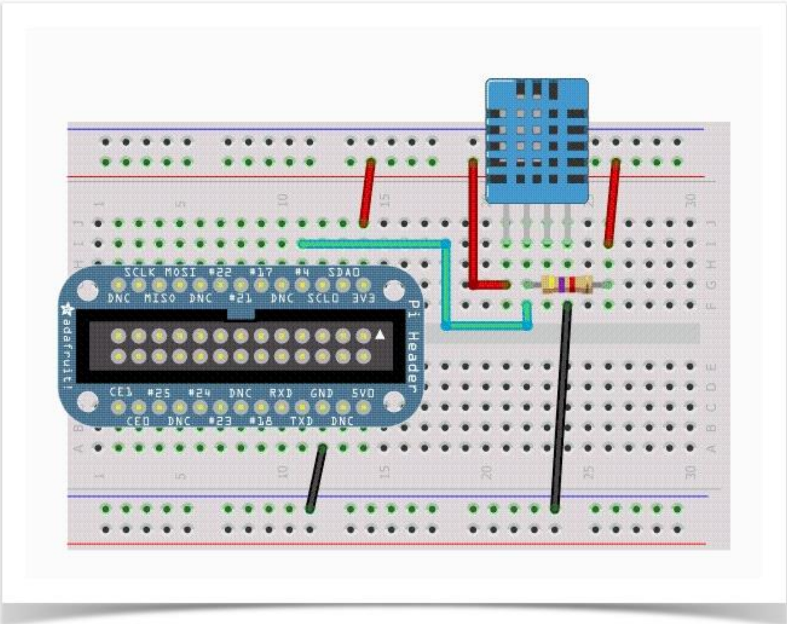
The humidity levels works as a pressure cap on the plant, keeping the moisture in the plant, allowing it to have proper transpiration rates of the fluids. When humidity levels drop too low, the plants transpire at a rate much quicker than that of nutrient uptake. The nutrients or minerals do not transpire through the plant, only the water does. So this leaves behind a concentrated level of nutrients in the plant that will actually cause a nutrient overload, which is not desirable for plants.

Conversely, when humidity levels get too high, moisture is building up on the plants and walls, forming whole colonies of moulds, fungi, and mildews. Similarly the biochemical functions in plants that are required for growth and survival are temperature dependent.

An optimal temperature range within which a particular plant species will be carrying out photosynthesis at its maximum rate (given that sufficient CO2, water and light are also present) outside this range, photosynthesis and other plant processes begin to slow down, to the point where they stop and growth ceases.

DHT11 sensor was chosen to monitor ambient temperature and humidity. This sensor proved to be reliable and stable With its small size, low power consumption, and ability to function in all kinds of harsh application occasions.

Interfacing Diagram



Requirements and Source Code

To make this, we need:

• DHT-11 Temperature and Humidity Sensor

• Jumper Wires

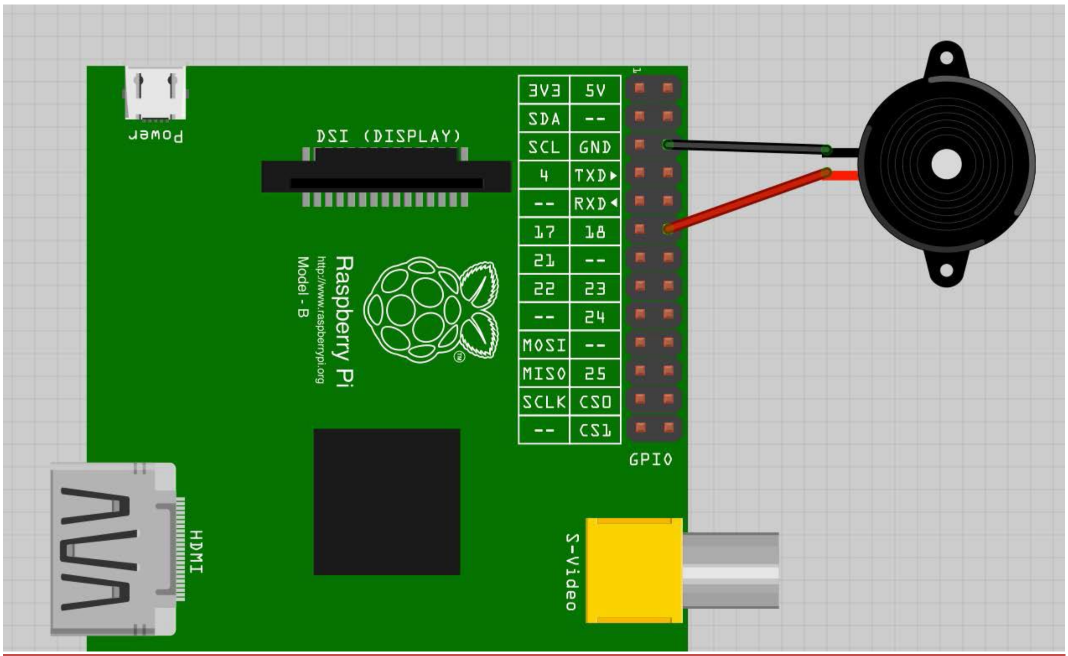
```
import Adafruit_DHT
import time

sensor = Adafruit_DHT.DHT11
pin = 20

humidity, temperature = Adafruit_DHT.read_retry(sensor, pin)
if humidity is not None and temperature is not None:
	print 'Temp={0:0.1f}*C  Humidity={1:0.1f}%'.format(temperature, humidity)
else:
	print ‘Failed to get reading. Try Again!’
```

5) Buzzer with Raspberry Pi 2

We use a piezo-electric buzzer connected to a GPIO pin of the Raspberry Pi 2 board.

The function of the buzzer is to serve as an alert if the sensor readings cross the set threshold. The produces a ringing condition whenever critical conditions are reached.

Interfacing Diagram



Requirements and Source Code

To make this, we need:

• Buzzer

• Jumper Wires

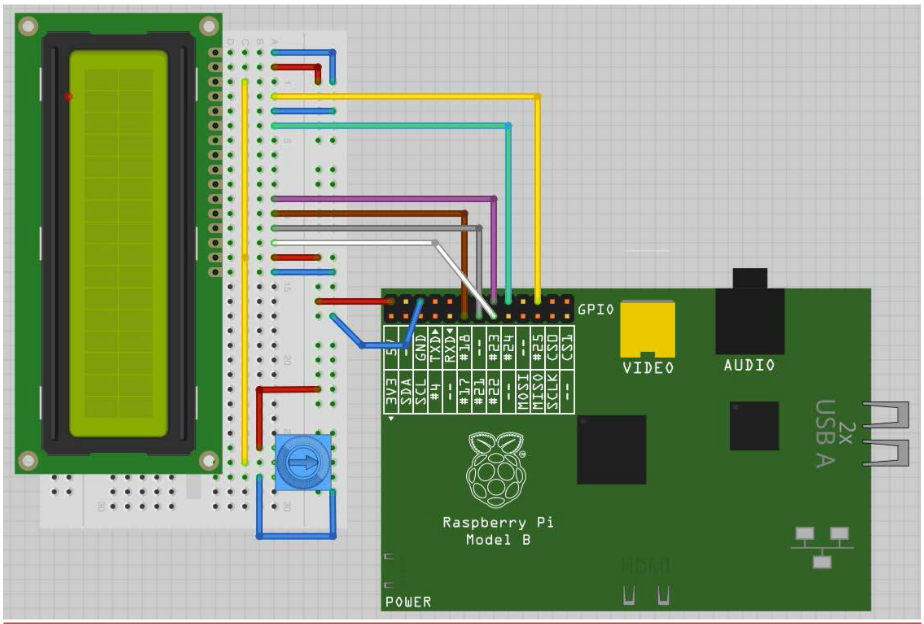
```
import RPi.GPIO as GPIO
import time

buzzer_pin = 18
GPIO.setmode(GPIO.BCM)
GPIO.setup(buzzer_pin, GPIO.OUT)

def buzz(pitch, duration):
	period = 1.0 / pitch
	delay = period / 2
	cycles = int(duration * pitch)
	for i in range(cycles):
		GPIO.output(buzzer_pin, True)
		time.sleep(delay)
		GPIO.output(buzzer_pin, False)
		time.sleep(delay)
while True:
	pitch_s = raw_input("Enter Pitch (200 to 2000): ")
	pitch = float(pitch_s)
	duration_s = raw_input("Enter Duration (seconds): ")

	duration = float(duration_s)
	buzz(pitch, duration)
```

6) LCD WITH RASPBERRY PI 2

Used an HD44780-compatible LCD module and wire it to the GPIO connector. These modules are available in a number of sizes and are specified by the number of columns and rows of letters they can display.

We use the LCD Screen to display data from the various sensors. Both, PT-1000 and DHT-11 send their reading to the 16X2 LCD screen.

Interfacing Diagram



Requirements and Source Code

To make this we need:

- Breadboard and jumper wires
- 16×2 HD44780-compatible LCD module
- Row of 16 header pins
- 10kΩ trimpot

The Adafruit Raspberry Pi example code, available from GitHub, includes a library for controlling LCD displays that use the HD44780 driver.

The Adafruit library is not installed as a *proper* library, so to use it, we first need to download the folder structure.

```
$ sudo apt-get install git
```

Now, we can download the folder structure from GitHub:

```
$ git clone https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code.git
```

Change directory into the Adafruit code using:

```
$ cd Adafruit-Raspberry-Pi-Python-Code
$ cd Adafruit_CharLCD
```

There is a test program here that displays the time and IP address of the Raspberry Pi.

But first, if we are using a newer Raspberry Pi model B, revision 2, we will need to edit the file *Adafruit_CharLCD.py*:

```
$ nano Adafruit_CharLCD.py
```

Search for the line:

```
def __init__(self, pin_rs=25, pin_e=24, pins_db=[23, 17, 21, 22], GPIO = None):
```

Replace the number 21 with 27 so that the line looks as follows, and then save and exit the file:

```
def __init__(self, pin_rs=25, pin_e=24, pins_db=[23, 17, 27, 22], GPIO = None):
```

We now run the example program with the command:

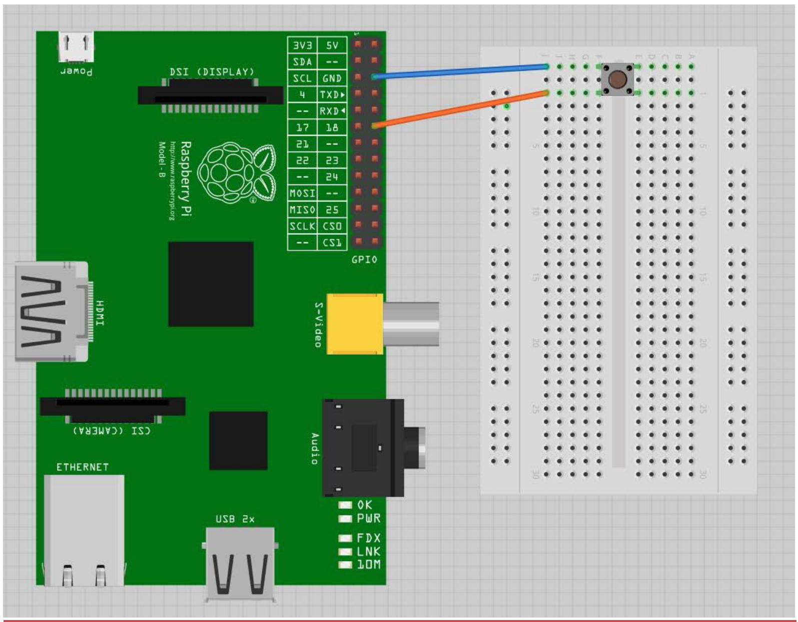
$ sudo python Adafruit_CharLCD_IPclock_example.py

```
from Adafruit_CharLCD import Adafruit_CharLCD
from time import sleep

lcd = Adafruit_CharLCD()
lcd.begin(16,2)

i = 0

while Tue:
	lcd.clear()
	lcd.message('Counting: ' + str(i))
	sleep(1)
	i = i + 1
```

7) Button with Raspberry Pi 2

Two tactile push switches are being used to control the functioning of both the sensors. Each switch is dedicated to a single sensor.

Interfacing Diagram



Requirements and Source Code

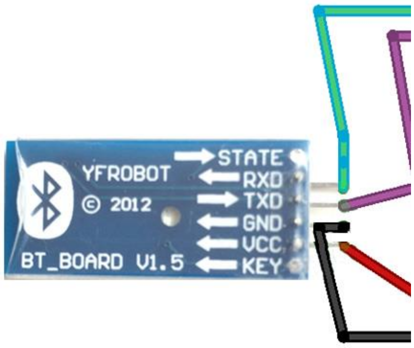
To make this we need:

• Tactile Push Switch (Button)

• Jumper Wires

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.IN, pull_up_down=GPIO.PUD_UP)

while True:
        input_state = GPIO.input(18)
        if input_state == False:
                print('Button Pressed')
        time.sleep(0.2)
```

8) BLUETOOTH WITH RASPBERRY PI 2

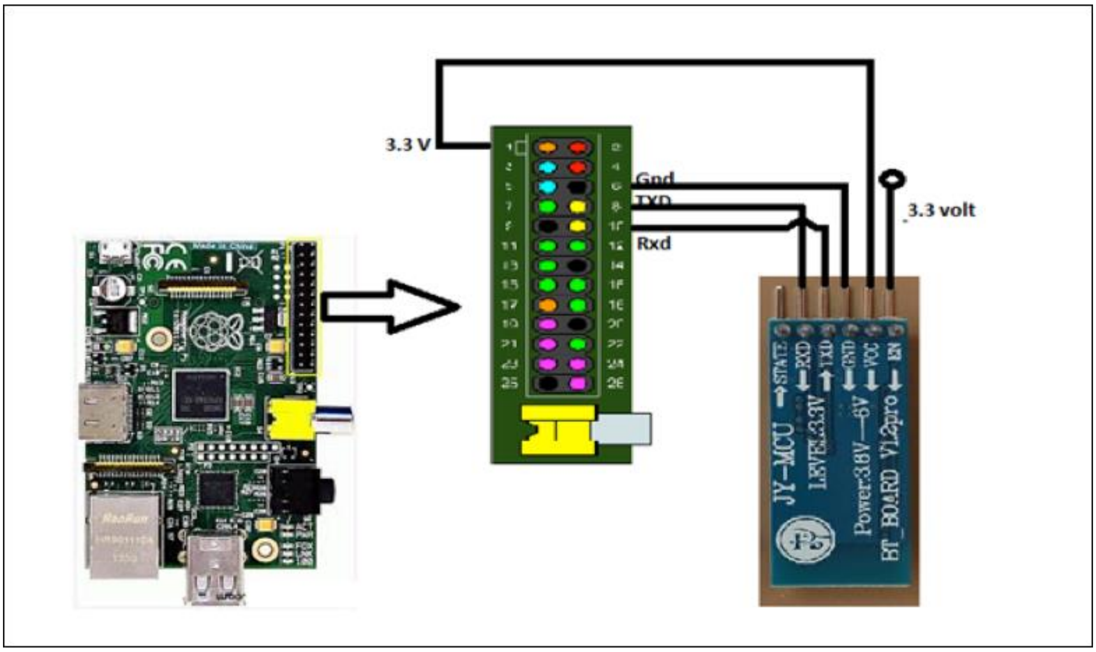


HC-05 Module is required HC-05 module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup. The Bluetooth module works as a Serial (RX/TX) pipe i.e. utilizes Serial Port Profile. Any serial stream from 9600 to 115200bps can be transferred seamlessly to the target.

Interfacing of HC-05 Bluetooth module to Raspberry Pi and establishing a communication between an android based mobile phone and Raspberry Pi.

The Bluetooth will be used to transfer the sensor data to the mobile phone for being available locally and being transferred to other smart devices for research usage.

Interfacing Diagram



Requirements and Source Code

To make this we need:

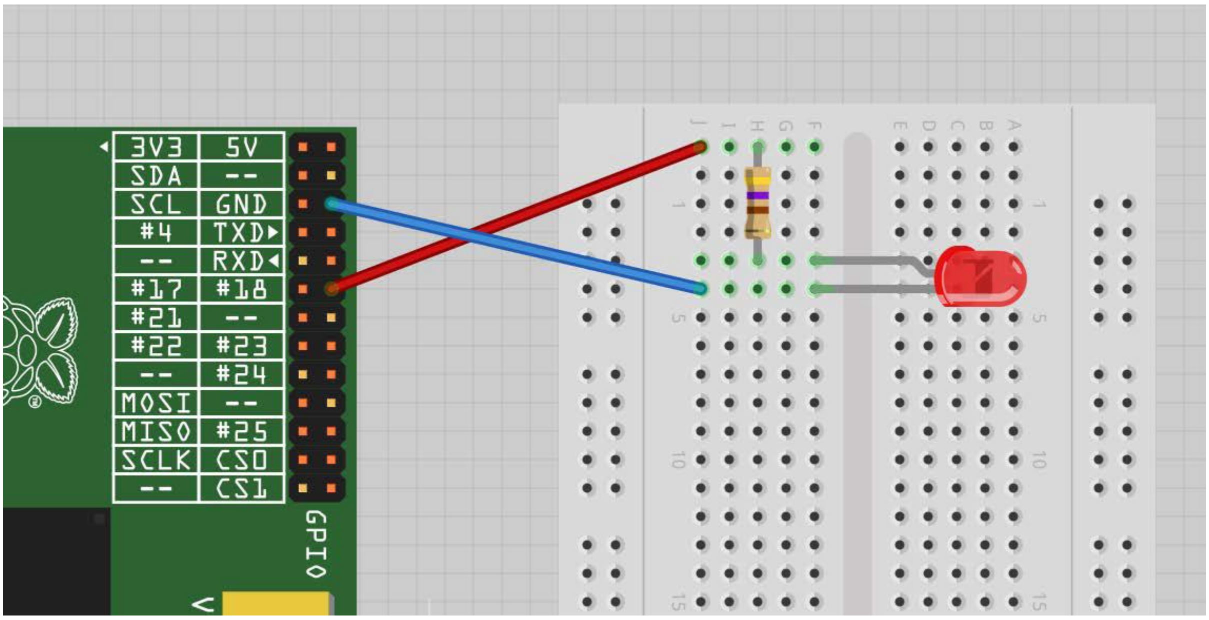
- HC-05 Bluetooth Module
- Bluetooth SPP Manager Android App

```
import serial                       # to access serial port
import time                         # to access time functionality

port=serial.Serial("/dev/ttyAMA0",9600, timeout=3.0)        # opening serial port ttyAMA0

print "\nBluetooth Communication Between RPI & ANDROID mobile Device\n"

while True:
        time.sleep(5)               # to give delay of 5 ms
        rcv=port.read(50)           # read if text available at serial port up to length of 50
        print "Received String from Android: " + rcv time.sleep(2)
        keyin = raw_input("Enter a string to send:")        # asking for input from keyboard
        port.write(keyin)
        time.sleep(5)
```

9) LED WITH RASPBERRY PI 2

An LED is used to serve as an indicator to whether the device is working or not. The LED turns ON as soon the device is switched ON and it goes OFF as soon as the device is switched off.

Interfacing Diagram



Requirements and Source Code

To make this we need:

- Breadboard and jumper wires
- 1kΩ resistor
- LED

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)

while (True):
        GPIO.output(18, True)
        time.sleep(0.5)
        GPIO.output(18, False)
        time.sleep(0.5)
```

Database Connectivity of the Sensor Readings

The usage of our Smart Agriculture Monitoring System is not just limited to that by farmers for keeping a track of their field contents but can also be extended for research purposes.

For this reason we have provided a database in the device which will record sensor data each time it is used. The readings accumulated over a period of time can be properly mined using appropriate data-mining algorithms and can be used by researchers to forecast the upcoming soil trends based on the past results.

This will be an efficient tool in figuring out what crop will be best suited for a particular soil in the times to come so that the farmers can plan their resources and cultivation practices accordingly.

The database readings can also be uploaded on the cloud for usage and analysis by various Agricultural Institutions.

Requirements and Source Code

Step1: First of all updates the packages using command:

```
$ sudo get-apt update
```

Step2: Download the database (for example MySQL). Open the LX-terminal and type the following command:

```
$ sudo apt-get install mysql-server python-mysqldb
```

This command will download the MySQL database in raspberry pi. During downloading , a password will have to enter, this is the root password of the database

Step3: Open the database by entering following command in LX-terminal:

```
$ mysql -u root –p
```

Enter password:

Mysql>CREATE DATABASE abcd;

USE abcd;

Now for create an account, enter the following commands:

Mysql >CREATE USER “pardeep"@"localhost" IDENTIFIED BY “password";

Mysql>GRANT ALL PRIVILEGES ON abcd.*TO “pardeep"@"localhost";

Mysql>FLUSH PRIVILEGES;

Mysql>quit

Step4: Now database is ready and we need to create a table as well. Open the mysql database using command:

$ mysql –u pardeep –p

Password: Use abcd;

Now create a table.

For example we are here creating a table which stores the potentiometer data with current date and time.

Mysql>CREATE TABLE igdtu(potdata FLOAT,time TIME,date DATE);

If you don’t get any error, table has created.

You can check it using mysql command:

Mysql>show tables;

Step5: Open the leafpad and write the python program as given in “Program" portion. Save it as name followed by “.py”. For example abc.py

Step6: Open the LX-terminal and run the file abc.py using command:

$ sudo python abc.py

Running of this file read the data from potentiometer and save it into a local database.

Now you can check your stored data into database.

System Description and Working

The Smart Agriculture Monitoring System consists of an analog soil temperature sensor which measures the temperature of the soil it has been inserted into.

Assuming the field for the growth of a particular crop we have fixed the suitable soil temperature to be 27 ^{O}C and thus any temperature greater than this makes the buzzer beep. The present temperature can also be read by the farmer from the LCD screen.

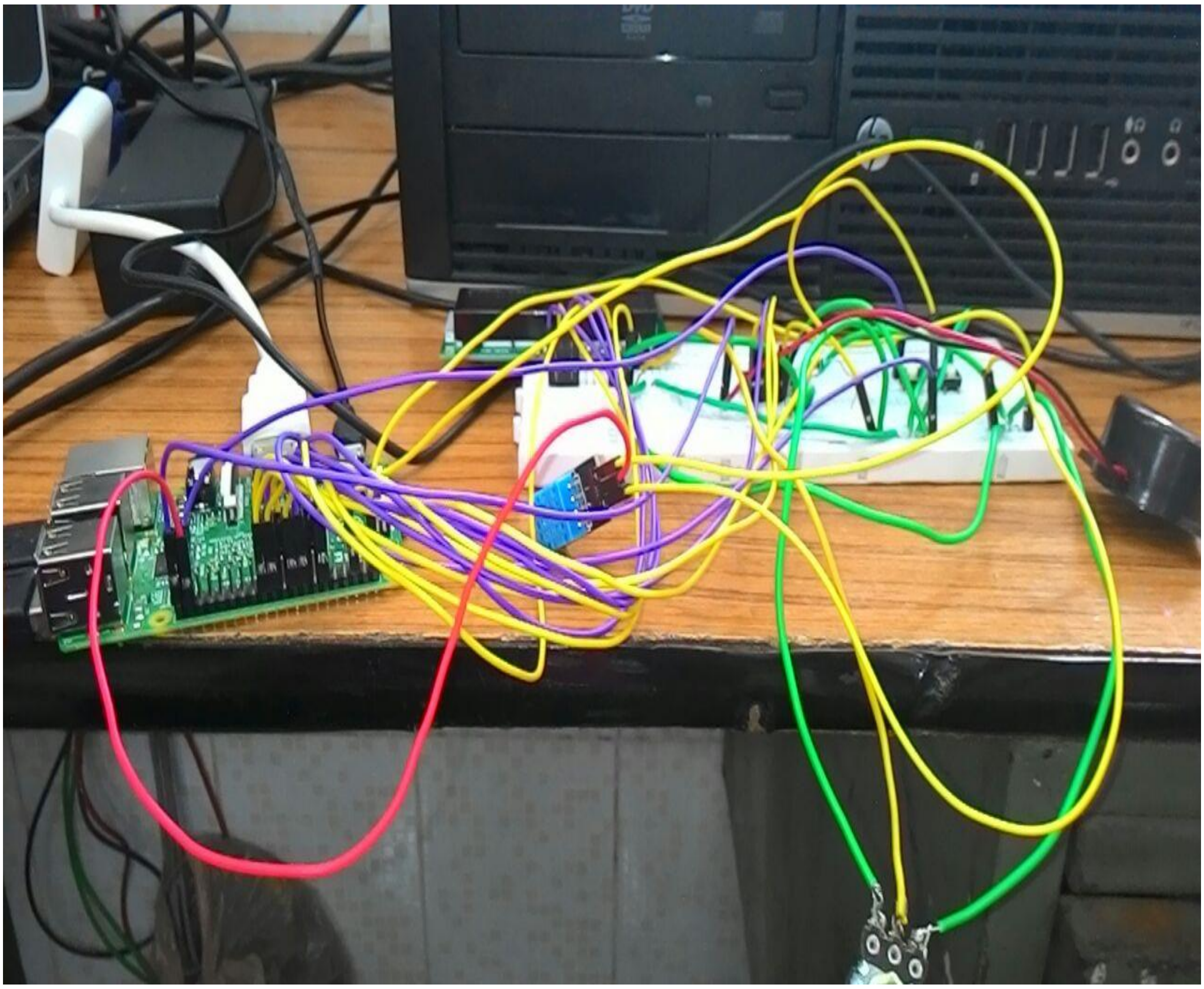
We also have a DHT-11 sensor which records the surrounding temperature and humidity which again can be read off the screen. The farmer is provided with 2 buttons which govern the working of the two sensors present.

Depending upon his/her requirement the farmer can press the button which sets the specific sensor in motion.

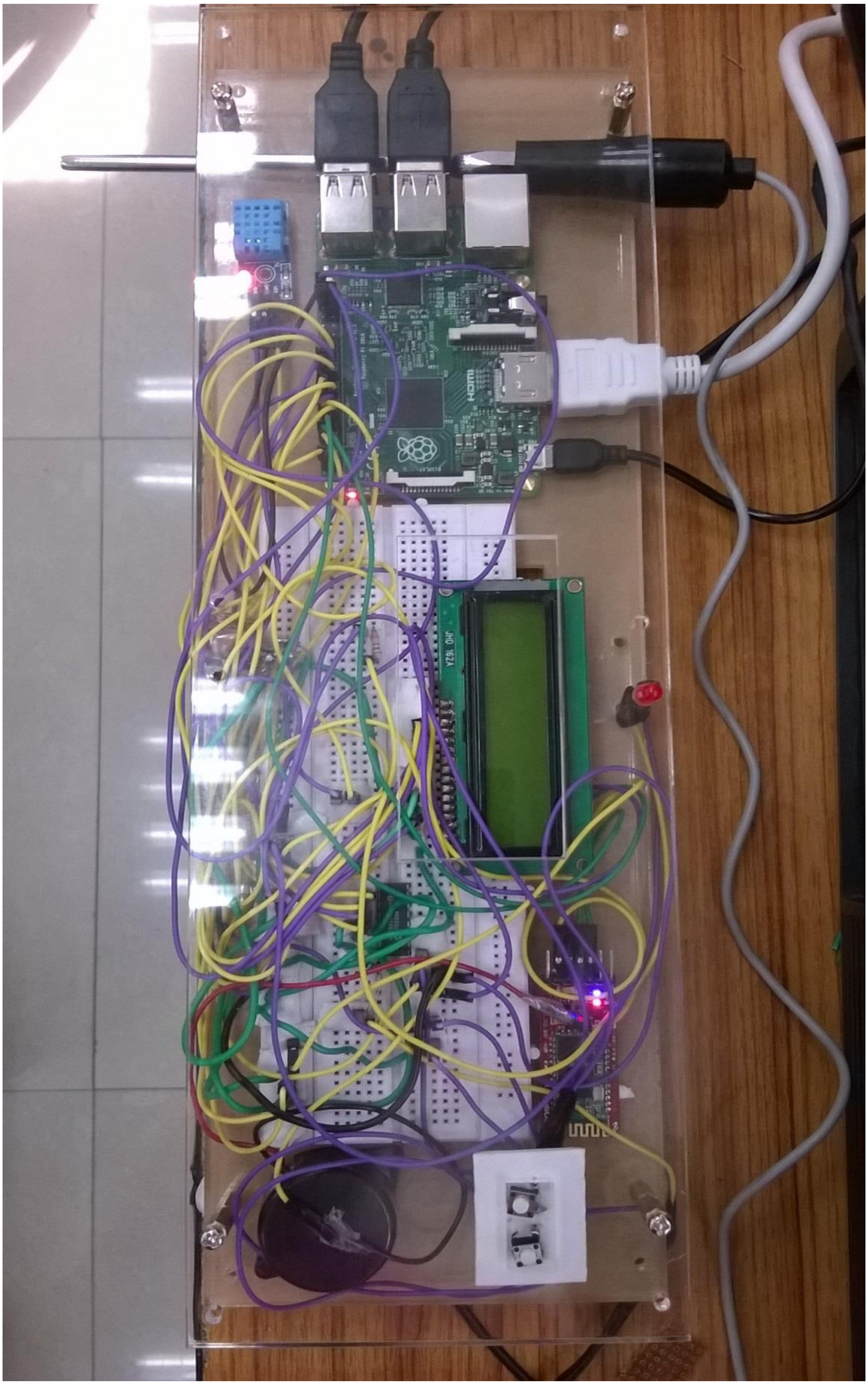
Real Time Interfacing Clips

Intermediate Stage of Product Development

(Naked Circuit without the Outer Packaging)



Complete Product (along with acrylic casing)



Complete Source Code of the Device

```
import spidev
import time
import MySQLdb
import RPi.GPIO as GPIO
import Adafruit_DHT
from Adafruit_CharLCD  import Adafruit_CharLCD
import serial

port=serial.Serial("/dev/ttyAMA0" , 9600, timeout=3.0)

GPIO.setmode(GPIO.BCM)
GPIO.setup(19, GPIO.OUT)

lcd=Adafruit_CharLCD()
lcd.begin(16,2)

sensor = Adafruit_DHT.DHT11
pin = 20

db=MySQLdb.connect("localhost","user","","final")
curs=db.cursor()

buzzer_pin= 21

GPIO.setmode(GPIO.BCM)
GPIO.setup(buzzer_pin, GPIO.OUT)

GPIO.setup(18, GPIO.IN, pull_up_down=GPIO.PUD_UP)
```

```
GPIO.setup(26, GPIO.IN, pull_up_down=GPIO.PUD_UP)

spi=spidev.SpiDev()
spi.open(0,0)

def buzz(pitch, duration):
        GPIO.output(buzzer_pin, True)
        time.sleep(duration)
        GPIO.output(buzzer_pin, False)

def analog_read(channel):
        r=spi.xfer2([1, (8 + channel) << 4,0])
        adc_out=((r[1]&3) << 8) + r[2]
        return adc_out

while True:
        input_state=GPIO.input(18)
        if input_state == False:
                GPIO.output(19, True)
                reading=analog_read(0)
                voltage=reading * 3.3 / 1024
                temp_c=voltage * 100 + 26
                temp_f=temp_c * 9.0 / 5.0 + 32
                print("Temp C=%f\t\tTemp f=%f" % (temp_c, temp_f))
                lcd.message("Temp C=%f" % (temp_c))
                time.sleep(1.5)
                lcd.clear()
                lcd.message("Temp f=%f" % (temp_f))
                time.sleep(1)
                lcd.clear()
                curs.execute( "INSERT INTO pt1000(temp_c, temp_f) VALUES('%s','%s')",
                                (temp_c, temp_f))
```

```
        port.write(“ temp_c \n”)
        port.write(str(temp_c))
        port.write(“ temp_f \n”)
        port.write(str(temp_f))

        if temp_c>27:
            buzz(500, 2)
        time.sleep(0.5)
        db.commit()
        GPIO.output(19, False)
    input_state2=GPIO.input(26)
    if input_state2 == False:
        GPIO.output (19, True)
        humidity, temperature = Adafruit_DHT.read_retry(sensor, pin)
        if humidity is not None and temperature is not None:
            print 'Temp={0:0.1f}*C  Humidity={1:0.1f}%'.format(temperature,
                humidity)
            lcd.message('Temp={0:0.1f}*C'.format(temperature))
            time.sleep(1.5)
            lcd.clear()
            lcd.message('Humidity={0:0.1f}%'.format(humidity))
            time.sleep(1)
            lcd.clear()
            curs.execute("INSERT INTO dht11(temp,hum)
                        Values(%s,%s)" % (temperature, humidity))

            db.commit()
        if temperature>27 or humidity<30:
            buzz(500, 2)
        time.sleep(0.5)
        GPIO.output(19, False)
```

Future Work to be done

We have successfully developed a monitoring system that allows the farmer to get to know the temperature of his soil and the temperature and humidity of his surroundings within seconds. The system consists of a LCD screen, an analog soil temperature sensor, a DHT-11 sensor and a buzzer. We can extend this project and include various other sensors such as soil moisture sensor, pH sensor, gas sensor and light intensity sensors which will give a complete picture of the soil to the farmer thus enabling him to grow the right kind of crop or make the surroundings suitable for his crop to ensure maximum yield and thus maximum profit.

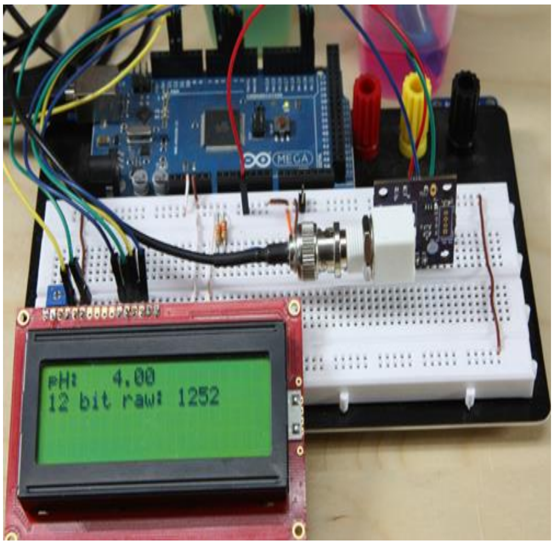
Immediate Future Aims:

1) Incorporating Soil Moisture Sensor

Measuring soil moisture is important in agriculture to help farmers manage their irrigation systems more efficiently. Not only are farmers able to generally use less water to grow a crop, they are able to increase yields and the quality of the crop by better management of soil moisture during critical plant growth stages.

Besides agriculture, there are many other disciplines using soil moisture sensors. Golf courses are now using sensors to increase the efficiencies of their irrigation systems to prevent over watering and leaching of fertilizers and other chemicals offsite.

2) Incorporating Soil pH Sensor



Soil pH sensor measures the pH level of the soil so that it can be kept in the ideal range as required by the particular plant.

3) Making the Data from Database Globally Available over Cloud

The data from the database needs to be made available online so that it can used by various research organisations and agricultural institutes for proper processing of the data obtained and for drawing useful conclusions.

References

1) http://www.ext.colostate.edu/drought/soilmoist.html
2) http://www.decagon.com/products/soils/volumetric-water-content-sensors/5tm-soil-moisture-and-temperature-sensor/
3) http://www.iuss.org/19th%20WCSS/Symposium/pdf/2153.pdf
4) http://www.stevenswater.com/soil_moisture_sensors/
5) http://www.rjpbcs.com/pdf/2015_6(3)/[215].pdf
6) Raspberry Pi Cook Book- O Reilly
7) http://www.mobileeducationkit.net/labmanuals/LAB-Manual-RaspberryPi.pdf