Mandatory Complete all steps from User manual before Starting with Labs

Experiment No: 1A
Introduction to IoT, Transducers, Sensors and Actuators & IoT Protocols.

<u>Aim</u>: To understand basic concepts of IoT such as sensors and actuators, IoT Protocols and Hardware Communication Protocols.

<u>Description</u>:

1.) Understanding Internet of Things:

Internet of Things (IoT) is a system of interconnected objects, usually called smart devices, through the Internet. The object can be a heart monitor, a remote or an automobile with built-in sensors. That is objects that have been assigned an IP address and have the capability to collect and transfer data over a network. The objects interact with the external environment with the help of embedded technology, which helps them in taking decisions. Since these devices can now represent themselves digitally.

In other words,

The globally ruling technology acting as a single key to shrinking this whole universe to a tiny globally connected village, whereas IoT comprises of just two words which precisely depicts its definition.

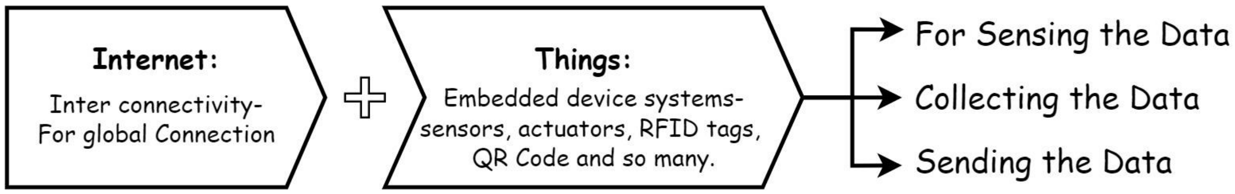

Internet:
Inter connectivity- For global Connection
Things:
Embedded device systems- sensors, actuators, RFID tags, QR Code and so many.
For Sensing the Data
Collecting the Data
Sending the Data

Figure 1: Introduction to IOT

➢ ***Main Components of IOT:***

- **Low-power embedded systems –**
  Less battery consumption, high performance are the inverse factors play a significant role during the design of electronic systems.
- **Cloud computing –**
  Data collected through IoT devices is massive and this data has to be stored on a reliable storage server. This is where cloud computing comes into play. The data is processed and learned, giving more room for us to discover where things like electrical faults/errors are within the system.
- **Availability of big data –**
  We know that IoT relies heavily on sensors, especially real-time. As these electronic devices spread throughout every field, their usage is going to trigger a massive flux of big data.

- **Networking connection –**
  In order to communicate, internet connectivity is a must where each physical object is represented by an IP address. However, there are only a limited number of addresses available according to the IP naming. Due to the growing number of devices, this naming system will not be feasible anymore. Therefore, researchers are looking for another alternative naming system to represent each physical object.

2.) Transducers Sensors and Actuators:

➢ ***Transducers:***

There are many variables which affect our everyday lives: the speed of a car, the velocity of the wind, and the temperature in a home. In most situations these variables are continuously monitored. It is these variables that are the feedback that is used to control the speed of a car, the operation of an air conditioner, heater levels, and oven temperatures. The elements that sense these variables and convert them to a usable output are transducers. For example, a transducer known as a thermocouple, is able to sense changes in temperature and produce output voltages representative of those changes.

A transducer is defined as a substance or a device that converts (or transfers) an input energy into a different output energy. Because of this broad definition, transducers come in many varieties converting many different types of energy.

Following are different types of transducers:

- Electrochemical Transducers:
  - pH probe – Converts chemical energy into an electrical energy
  - Molecular electric transducer – Converts motion in an electrolytic solution into electrical energy
  - Battery – Converts chemical energy directly into electrical energy
  - Fuel cell – Converts the energy from a reaction within a fuel cell to electrical energy
- Electroacoustic, Electromagnetic, Electrostatic Transducers:
  - Loudspeaker – Converts an electrical signal into sound
  - Microphone – Converts sound waves in air into an electrical signal
  - Hydrophone - Converts sound waves in water into an electrical signal.
  - Magnetic cartridge – Converts motion in a magnetic field into an electrical energy
  - Generator – Converts motion in a magnetic field into electrical energy
  - Electrometer – Converts static or energy from a vibrating reed into electricity
  - Van de Graaf generator – Converts static into high voltage
- Electromechanical transducers (Also called actuators):
  - Strain gauge – Converts the deformation (strain) of an object into electrical resistance
  - Galvanometer – Converts the electric current of a coil in a magnetic field into movement
  - Generators – Converts mechanical energy (motion) into electrical energy.

    - Motor – Converts electrical energy into mechanical energy
- Photoelectric Transducers:
    - Cathode ray tube (CRT) –Converts electrical signals into light energy for a visual output
    - Light bulb –Converts electrical energy into visible light and heat
    - Laser diode – Converts electrical energy into light energy
    - Photodiode - Converts light energy into electrical energy
- Thermoelectric Transducers:
    - Thermocouple – Converts heat energy into electrical energy
    - Temperature sensitive resistor (Thermistor) – a variable resistor affected by temperature changes (heat energy to electrical energy)
- Other types of transducers:
    - Geiger-Müller tube – Converts radioactive energy into electrical energy
    - Quartz Crystal – Converts mechanical stress into electricity (electrical energy)

➢ ***Sensors:***

Sensors detect the presence of energy, changes in or the transfer of energy. Sensors detect by receiving a signal from a device such as a transducer, then responding to that signal by converting it into an output that can easily be read and understood. Typically, sensors convert a recognized signal into an electrical – analog or digital – output that is readable. In other words, a transducer converts one form of energy into another while the sensor that the transducer is part of converts the output of the transducer to a readable format.

Transducers convert one form of energy to another, but they do not quantify the conversions. The light bulb converts electrical energy into light and heat; however, it does not quantify how much light or heat. A battery converts chemical energy into electrical energy but it does not quantify exactly how much electrical energy is being converted. If the purpose of a device is to quantify an energy level, it is a sensor.

Following are different types of sensors which are classified by the type of energy they detect:

- Thermal Sensors:
    - Thermometer – measures absolute temperature
    - Thermocouple gauge– measures temperature by its effect on two dissimilar metals
    - Calorimeter – measures the heat of chemical reactions or physical changes and heat capacity
- Mechanical Sensors:
    - Pressure sensor – measures pressure
    - Barometer – measures atmospheric pressure
    - Altimeter – measures the altitude of an object above a fixed level
    - Liquid flow sensor – measures liquid flow rate
    - Gas flow sensor – measures velocity, direction, and/or flow rate of a gas
    - Accelerometer – measures acceleration
- Electrical Sensors:

    - Ohmmeter – measures resistance
    - Voltmeter – measures voltage
    - Galvanometer – measures current
    - Watt-hour meter – measures the amount of electrical energy supplied to and used by a residence or business
- Chemical Sensors:
    - Oxygen sensor – measures the percentage of oxygen in a gas or liquid being analyzed
    - Carbon dioxide detector – detects the presence of CO2
- Optical Sensors:
    - Light sensors (photodetectors) – detects light and electromagnetic energy
    - Photocells (photoresistor) – a variable resistor affected by intensity changes in ambient light.
    - Infra-red sensor – detects infra-red radiation
- Acoustic Sensors:
    - Seismometers – measures seismic waves
    - Acoustic wave sensors – measures the wave velocity in the air or an environment to detect the chemical species present
- Other Sensor types:
    - Motion – detects motion
    - Speedometer – measures speed
    - Geiger counter – detects atomic radiation
    - Biological – monitors human cells

➢ ***Actuators:***

An actuator is something that actuates or moves something. More specifically, an actuator is a device that coverts energy into motion or mechanical energy. Therefore, an actuator is a specific type of a transducer.

- Thermal Actuators: One type of thermal actuator is a bimetallic strip. This device directly converts thermal energy into motion. This is accomplished by utilizing an effect called thermal expansion
- Electric Actuators: An electric motor is a type of an electric actuator
- Mechanical Actuators: Mechanical actuators convert a mechanical input (usually rotary) into linear motion. A common example of a mechanical actuator is a screw jack.

3.) IoT protocols:

IoT communication protocols are modes of communication that protect and ensure optimum security to the data being exchanged between connected devices.

The IoT devices are typically connected to the Internet via an IP (Internet Protocol) network. However, devices such as Bluetooth and RFID allow IoT devices to connect locally. In these cases, there’s a difference in power, range, and memory used. Connection through IP networks are comparatively complex, requires increased memory and power from the IoT devices while the range is not a problem. On the other hand, non-IP networks demand comparatively less power and memory but have a range limitation.

As far as the IoT communication protocols or technologies are concerned, a mix of both IP and non-IP networks can be considered depending on usage. IoT protocols and standards can be broadly classified into two separate categories:

➢ ***IoT Network Protocols:***

IoT network protocols are used to connect devices over the network. These are the set of communication protocols typically used over the Internet. Using IoT network protocols, end-to-end data communication within the scope of the network is allowed. Following are the various IoT Network protocols:

- HTTP (HyperText Transfer Protocol):
  HyperText Transfer Protocol is the best example of IoT network protocol. This protocol has formed the foundation of data communication over the web. It is the most common protocol that is used for IoT devices when there is a lot of data to be published. However, the HTTP protocol is not preferred because of its cost, battery-life, energy saving, and more constraints.

  Additive manufacturing/3D printing is one of the use cases of the HTTP protocol. It enables computers to connect 3D printers in the network and print three-dimensional objects and pre-determined process prototypes.

- LoRaWan (Long Range Wide Area Network):
  It is a long-range low power protocol that provides signal detection below the noise level. LoRaWan connects battery operated things wirelessly to the Internet in either private or global networks. This communication protocol is mainly used by smart cities, where there are millions of devices that function with less power and memory.

  Smart street lighting is the practical use case of LoRaWan IoT protocol. The street lights can be connected to a LoRa gateway using this protocol. The gateway, in turn, connects to the cloud application that controls the intensity of light bulbs automatically based on the ambient lighting, which helps in reducing the power consumption during day-times.

- Bluetooth:
  Bluetooth is one of the most widely used protocols for short-range communication. It is a standard IoT protocol for wireless data transmission. This communication protocol is secure and perfect for short-range, low-power, low-cost, and wireless transmission between electronic devices. BLE (Bluetooth Low Energy) is a low-

energy version of Bluetooth protocol that reduces the power consumption and plays an important role in connecting IoT devices.

Bluetooth protocol is mostly used in smart wearables, smartphones, and other mobile devices, where small fragments of data can be exchanged without high and memory. Offering ease of usage, Bluetooth tops the list of IoT device connectivity protocols.

- ZigBee:
  ZigBee is an IoT protocol that allows smart objects to work together. It is commonly used in home automation. More famous for industrial settings, ZigBee is used with apps that support low-rate data transfer between short distances.

  Street lighting and electric meters in urban areas, which provides low power consumption, use the ZigBee communication protocol. It is also used with security systems and in smart homes.

➢ ***IoT Data Protocols:***
IoT data protocols are used to connect low power IoT devices. These protocols provide point-to-point communication with the hardware at the user side without any Internet connection. Connectivity in IoT data protocols is through a wired or a cellular network. Some of the IoT data protocols are:

- Message Queue Telemetry Transport (MQTT):
  One of the most preferred protocols for IoT devices, MQTT collects data from various electronic devices and supports remote device monitoring. It is a subscribe/publish protocol that runs over Transmission Control Protocol (TCP), which means it supports event-driven message exchange through wireless networks.

  MQTT is mainly used in devices which are economical and requires less power and memory. For instance, fire detectors, car sensors, smart watches, and apps for text-based messaging.

- Constrained Application Protocol (CoAP):
  CoAP is an internet-utility protocol for restricted gadgets. Using this protocol, the client can send a request to the server and the server can send back the response to the client in HTTP. For light-weight implementation, it makes use of UDP (User Datagram Protocol) and reduces space usage. The protocol uses binary data format EXL (Efficient XML Interchanges).

  CoAP protocol is used mainly in automation, mobiles, and microcontrollers. The protocol sends a request to the application endpoints such as appliances at homes and sends back the response of services and resources in the application.

- Advanced Message Queuing Protocol (AMQP):
  AMQP is a software layer protocol for message-oriented middleware environment that provides routing and queuing. It is used for reliable point-to-point connection and supports the seamless and secure exchange of data between the connected devices and the cloud. AMQP consists of three separate components namely Exchange, Message Queue, and Binding. All these three components ensure a secure and successful exchange and storage of messages. It also helps in establishing the relationship of one message with the other.

  AMQP protocol is mainly used in the banking industry. Whenever a message is sent by a server, the protocol tracks the message until each message is delivered to the intended users/destinations without failure.

- Machine-to-Machine (M2M) Communication Protocol:
  It is an open industry protocol built to provide remote application management of IoT devices. M2M communication protocols are cost-effective and use public networks. It creates an environment where two machines communicate and exchange data. This protocol supports the self-monitoring of machines and allows the systems to adapt according to the changing environment.

  M2M communication protocols are used for smart homes, automated vehicle authentication, vending machines, and ATM machines.

- Extensible Messaging and Presence Protocol (XMPP):
  The XMPP is uniquely designed. It uses a push mechanism to exchange messages in real-time. XMPP is flexible and can integrate with the changes seamlessly. Developed using open XML (Extensible Markup Language), XMPP works as a presence indicator showing the availability status of the servers or devices transmitting or receiving messages.

  Other than the instant messaging apps such as Google Talk and WhatsApp, XMPP is also used in online gaming, news websites, and Voice over Internet Protocol (VoIP).

  ✓ IoT Protocols Offers a Secured Environment for Exchange of Data
  ✓ As per an article published by Forbes, approximately “32,000 smart homes and businesses are at risk of leaking data.” Therefore, it becomes important to explore the potentials of IoT protocols and standards, which creates a secure environment. Using these protocols, local gateways and other connected devices can communicate and exchange data with the cloud.

Conclusion:

Experiment No: 1B

Getting Started with Infi IoT Cloud platform

<u>Aim</u>: To Sign-up/Sign-in and explore Infi IoT Cloud

<u>Description</u>:

What are IoT Cloud Platforms?

To bring physical objects online and make them communicate, cooperate and act intelligently without human intervention, the Internet of Things relies on IoT platforms to enable provisioning, management, and automation of smart objects within a given IoT infrastructure. Generally speaking, each IoT environment is a mashup of technologies by various vendors that form a complex and inherently diverse ecosystem which, without a common base for their integration, would stay fragmented, 'dumb', and, ultimately, unable to function. Therefore, it can be said that an IoT platform provides a 'meeting point' for all the connected devices and serves to collect and handle the data they deliver over the network.

At the other end of the Internet of Things cloud solution, there is cloud computing. Breathing new life into IT services, cloud computing is the latest technology buzz that has moved consumer and business applications to the web, thus enabling enterprises to optimise their IT performance and reduce costs that would be otherwise bloated by the need of creating and maintaining on-site IT architecture for storing data and running applications. Cloud-based solutions are not only more cost-effective in the long run; they also provide better security, corporate data mobility, increased co-worker collaboration, more advanced disaster recovery solutions, to name only a few benefits. What is more, cloud computing offers more flexibility which helps to shift the company's focus from IT hosting-related issues towards aspects directly affecting its business bottom-line.

Infi IoT Cloud Platform

Infi Iot is an Advance IoT development platform for connecting devices to the cloud and to remotely monitor and manage these devices.

Infi Iot provides features such as dashboards, automated data integration, Upload your past data, device alerts, notifications and the most amazing Analytics.

Get started with your project ideas connect your devices, control from platform, analyse your data and make future predictions with our unique IoT cloud platform

Specifications of IoT Cloud Platform:

- **Future prediction** – use historical and transactional data to predict future machine-related metrics, and to identify potential risks to your machines. Use predictions to identify risks, carry out proactive maintenance of machines, and avoid product delivery delays. Predictions help to create and meet production plans for Industries. Predictions help warn you of impending machine failure in advance. Preventive maintenance can help save the costs associated with machine breakdown or unavailability.
- **Multi-Threaded Interface** – is the ability of an operating system process to manage its use by more than one user at a time and to even manage multiple requests by the same user without having to have multiple copies of the programming running in the computer.
- **AES Secure Payloads –** AES is a symmetric block cipher with a block length of 128 bits. It allows using three different key lengths: 128, 192 or 256 bits. The encryption/ decryption algorithm consists of several rounds of processing; the number of rounds depends on the key length: 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Except for the last round, all other rounds are identical. The rounds in-order are one single-byte based substitution, row-wise permutation, column-wise mixing, addition of the round key.
- **IoT Protocols –** IoT data protocols are used to connect low power IoT devices. These protocols provide point-to-point communication with the hardware at the user side without any Internet connection. Connectivity in IoT data protocols is through a wired or a cellular network. *Refer Experiment 1A for more information.*
- **User Authentication –** User authentication is a process that allows a device to verify the identity of someone who connects to a network resource. Authentication is very important when you use dynamic IP addressing (DHCP) for computers on the trusted or optional network. It is also important if you must identify your users before you let them connect to resources on the external network.
- **No setup** – Setup are the sequence of process must be carried to setup a Cloud service and get started sending data to Cloud, it may include signing up/in, creating projects, defining hardware, Creating variables and widgets. Infi IoT don’t require any
- **Export/Import csv** – This Feature allows user to save logged data onto local systems by exporting as csv file, this helps user to use the data and perform analytics. Similarly, if user already have logged data, it can be uploaded using the import option on the cloud. This helps in utilizing past data for analytics and in making future predictions.

So How does Infi IoT Cloud Work?

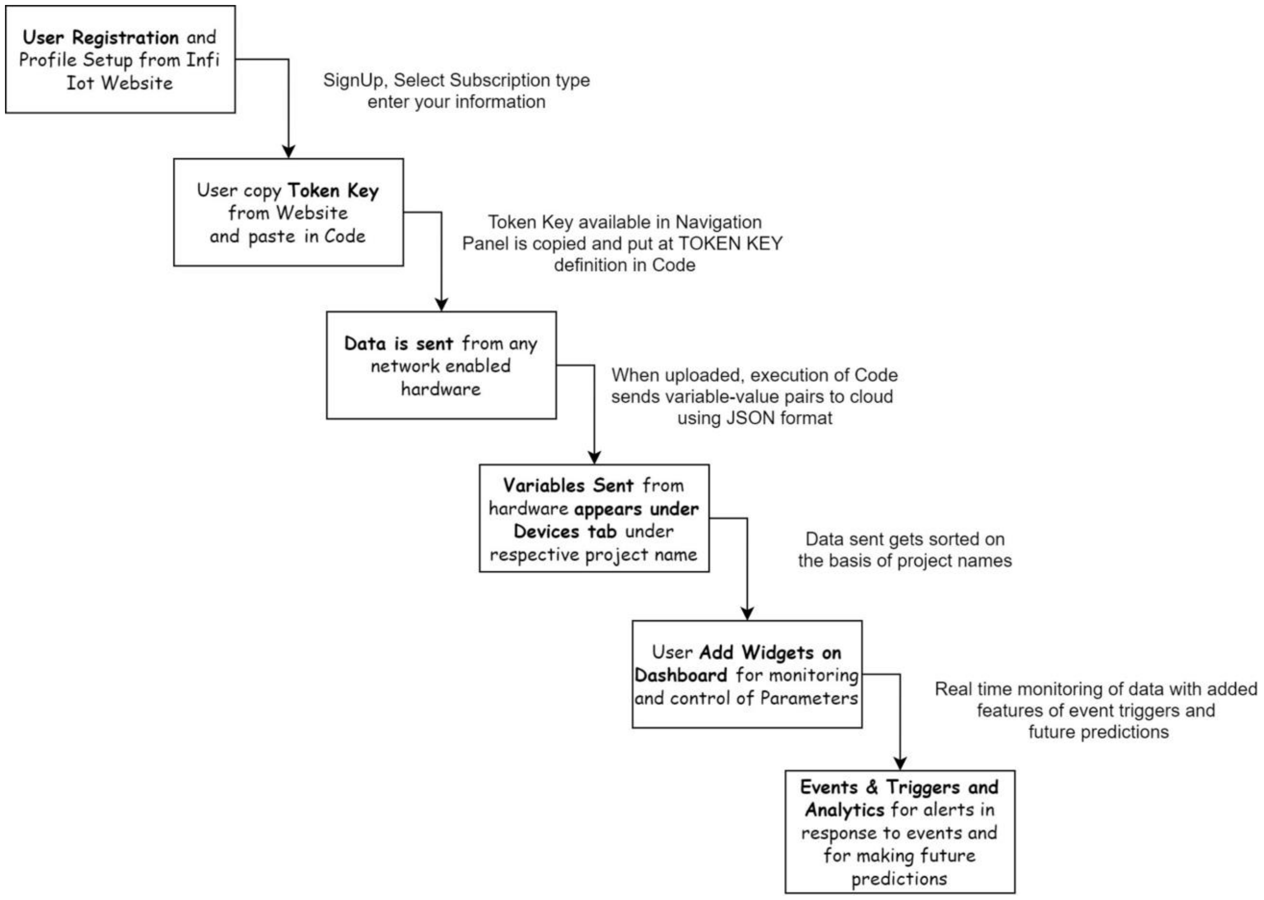

User Registration and Profile Setup from Infi Iot Website
SignUp, Select Subscription type enter your information
User copy Token Key from Website and paste in Code
Token Key available in Navigation Panel is copied and put at TOKEN KEY definition in Code
Data is sent from any network enabled hardware
When uploaded, execution of Code sends variable-value pairs to cloud using JSON format
Variables Sent from hardware appears under Devices tab under respective project name
Data sent gets sorted on the basis of project names
User Add Widgets on Dashboard for monitoring and control of Parameters
Real time monitoring of data with added features of event triggers and future predictions
Events & Triggers and Analytics for alerts in response to events and for making future predictions

Figure 2: Design Flow Infi IoT

<u>Procedure & Explanations</u>:

A. Signing-Up for IoT Cloud Platform

Visit http://www.infiiot.com and sign up for an account and log in to get started with cloud.

Please refer this part from the Infi Sense user Manual (under Infi IoT Cloud Section)

B. Profile Setup IoT Cloud Platform

Step 1: After Completing Procedure A (Signing up for IoT cloud), we **log in** to account.

Step 2: After logging in we are welcome with this page showing the **default dashboard** containing of different widgets for presentation.

Step 3: We first head to **Profile page** and enter details and complete the profile.(*This step is not mandatory*)

Step 4: In the Profile Page on top we can see a Token key, lets copy this key and keep it secret. This key will be used in code for the cloud to authenticate that incoming data is from you!

Note: Never share secret Token key with anyone, your data may go to wrong hands if shared.

C. Exploring Device Section on Infi IoT cloud:

Step 1: Click on **Devices** option on the **Navigation panel**

Device panel shows a list of all your IoT projects, by default it will contain a dummy project with some variables

On Infi IoT Cloud there is no need to create a project and variable on our own, when data (containing token key, project names and variable names) is sent to cloud, it automatically creates project/variables which are not already present

Step 2: Select any project by clicking on **view option** or Create a project by clicking on the round **+ icon** on top-right (*Creating a project is optional*)

Once inside **Project Section**, we can see all the variables related to the project. Variables are of two types: Raw Variables & Synthetic Variables.

Raw Variables: These variables contain values as sent from sensors. E.g. tempC: a variable containing raw temperature readings sent from sensor which are in °C

Synthetic Variables: These variables are related to other raw or static variables through arithmetic equations. E.g. tempF: synthetic variable with arithmetic equation $\left(TempC * \left(\frac{9}{5}\right)\right) + 32$ written during setup. This variable will perform the operation over the raw variable TempC and update values for tempF variable which will be in Fahrenheits

Variables also are auto-created when data is sent from hardware

Step 3: All **variables** can be either viewed as **table** or as **charts** using the two buttons on the variable labels

Table view gives a spreadsheet plot of all the data corresponding to server time, we can click on the dropdown on the top-right corner to sort data according to durations.

Char View gives a visual presentation of the data, again we can sort, pan, capture screenshot, import csv or sort chart using the options on the top-right corner

D. Exploring Dashboard Section on Infi IoT cloud

Step 1: Click on **Dashboard** option on the **Navigation panel**

By default, we see the default dashboard with different widgets

Widgets help in better analytics and visualization of data by using various plotting techniques such as Charts, Tables, Labels, Pie Charts, Controls and Indicators.

Charts:

Line Charts: displays information as a series of data points called 'markers' connected by straight line segments.

Double Axis Charts: This chart type uses two axes to easily illustrate the relationships between two variables with different magnitudes and scales of measurement. The relationship between two variables is referred to as correlation.

Scatter Plot: uses dots to represent values for two different numeric variables. The position of each dot on the horizontal and vertical axis indicates values for an individual data point. They are used to observe relationships between variables.

Bar Chart: presents categorical data with rectangular bars with heights or lengths proportional to the values that they represent.

Indicators:

On/Off Indicator: used to indicate value of a variable using large colored indicator, with one color for on status and other color for off status,

Gauge: uses gauge for indicating values variable between fixed max and min ranges

Thermometer: used to visualize temperature using classic thermometer visuals

Controls:

Switch: uses classic switch to update values for variable it is assigned, it is used to control actuators from the cloud side. Switches are used mostly to update variables with boolean values, e.g. for controlling lights.

Slider: uses a sliding axis with a control notch, the values can be set within any range, which updates the variable. Slider are used when the control variable can have more than values, e.g. for controlling fan.

Pie Chart: circular statistical graphic, which is divided into slices to illustrate numerical proportion. In a pie chart, the arc length of each slice, is proportional to the quantity it represents.

Labels: Text representation of variable for monitoring specific variable values with respect to time, the displayed value can be

Tables: a data structure that organizes information into rows and columns. It can be used to both store and display data in a structured format.

Step 2: We can also **create Dashboard** for next project by clicking on **add Dashboard option** on top right corner

In this newly created Dashboard, click on **Add Widget** option to add any widgets as explained in Step 1.

Once Created Widgets can be seen on Dashboard.

E. Predictions on Infi IoT Cloud

Step 1: Click on **Predictions** option on the **Navigation panel**

Prediction Panel Consists of three sections:

Load Data: First we need to load past data for future prediction, there are two options for loading data:

Import .csv: to load data from .csv file

Choose from Cloud**:** to work on data from Cloud account.

Train Model: Selection of algorithms and hyperparameter tuning for model training in this section

See Predictions: After Successful Model training the predicted values are shown under this section. Values are predicted from their trends.

Step 2: Select on **use data from cloud radio button** and select the variable to predict the values.

Verify that the data plotted matches with the selected variable.

Once done uploading data, we move to model training

Step 3: In this Section we select the algorithm from dropdown menu and tune hyperparameters (optional) and click Train Model button below.

While model is getting trained, we can see the progress on top bar and epochs on the console log

Step 4: We now move to **See Prediction plot** on third section, here we can see the predicted values according to the parameters provided.

Conclusion:

Experiment No: 1C

Interfacing LCD, LED, Button

<u>Aim</u>: To interface display, indication and control modules.

<u>Description</u>:

LCD Module (16*2)

Liquid Crystal Display(LCDs) provide a cost effective way to put a text output unit for a microcontroller. As we have seen in the previous tutorial, LEDs or 7 Segments do no have the flexibility to display informative messages. This display has 2 lines and can display 16 characters on each line. Nonetheless, when it is interfaced with the micro controller, we can scroll the messages with software to display information which is more than 16 characters in length.

Most of the 16x2 LCDs use a Hitachi HD44780 or a compatible controller.

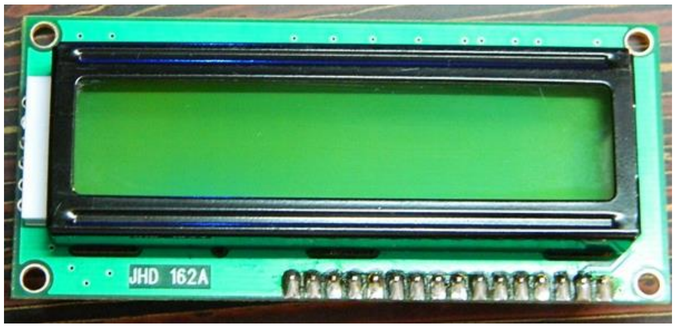


Figure 4: LCD Front View

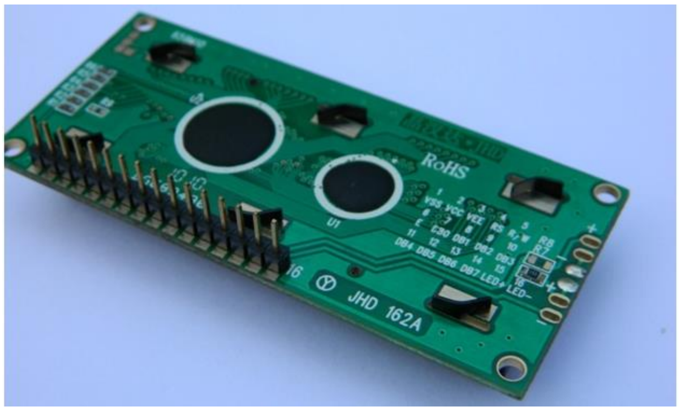


Figure 3: LCD Back View

➢ ***Block Diagram:***

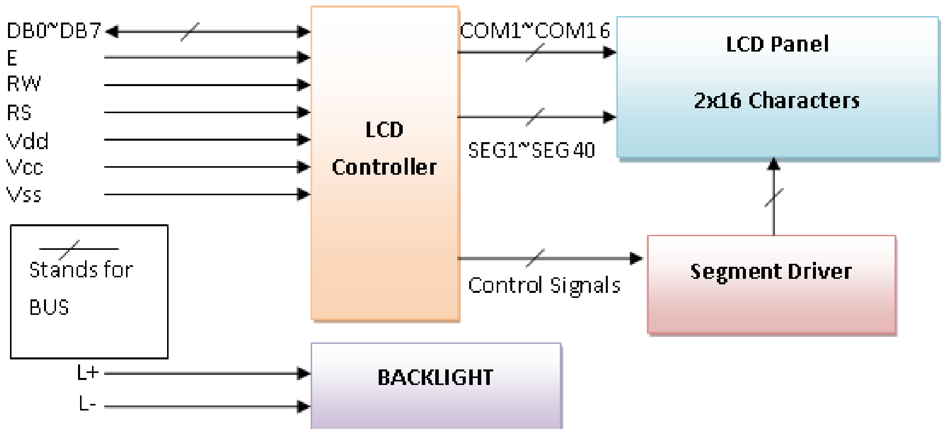


Figure 5: Block Diagram of LCD

➢ ***Pin Configuration:***

Pin Number	Function
D1	Ground
D2	Vcc
D3	Contrast adjustment (VO)
D4	Register Select (RS). RS=0: Command, RS=1: Data
D5	Read/Write (R/W). R/W=0: Write, R/W=1: Read
D6	Clock (Enable). Falling edge triggered
D7	Bit 0 (Not used in 4-bit operation)
D8	Bit 1 (Not used in 4-bit operation)
D9	Bit 2 (Not used in 4-bit operation)
D10	Bit 3 (Not used in 4-bit operation)
D11	Bit 4
D12	Bit 5
D13	Bit 6
D14	Bit 7
D15	Back-light Anode (+)
D16	Back-Light Cathode (-)

➢ ***Interface and wirings:***

LCD can be interfaced with the microcontroller in two modes, 8 bit and 4 bit.

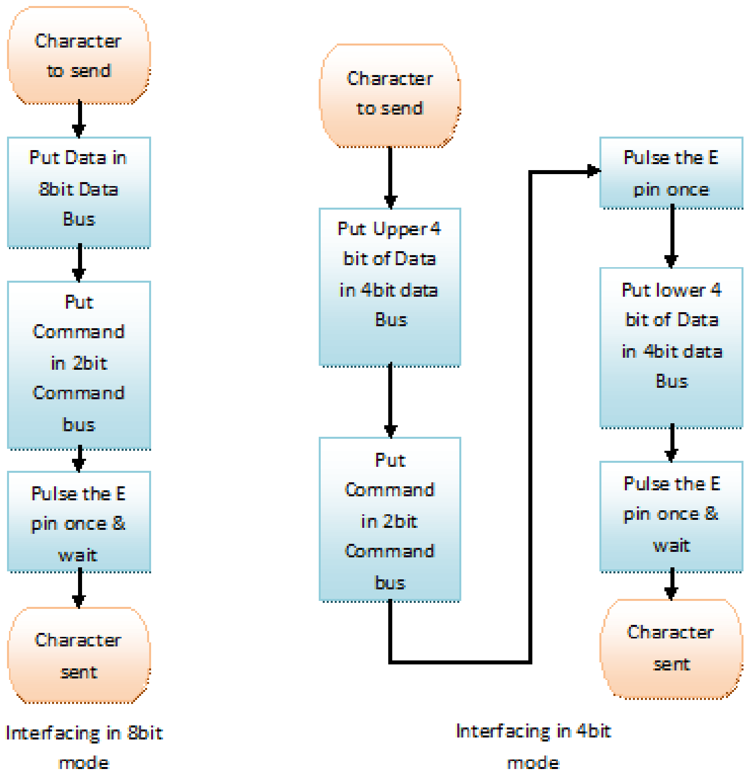

Character to send
Put Data in 8bit Data Bus
Put Command in 2bit Command bus
Pulse the E pin once & wait
Character sent
Interfacing in 8bit mode
Character to send
Put Upper 4 bit of Data in 4bit data Bus
Put Command in 2bit Command bus
Pulse the E pin once
Put lower 4 bit of Data in 4bit data Bus
Pulse the E pin once & wait
Character sent
Interfacing in 4bit mode

Figure 6: Flow Chart of interfacing LCD Display

The important pins sorted according to their tasks are as follows:

- **Data Lines:** In this mode, all of the 8 data lines DB0 to DB7 are connected from the microcontroller to a LCD module as shown the schematic.
- **Control Lines:'** The RS, RW and E are control lines, as discussed earlier.
- **Power & contrast**: Apart from that the LCD should be powered with 5V between **PIN 2(VCC)** and **PIN 1(Gnd)**. **PIN 3** is the contrast pin and is output of center terminal of potentiometer (voltage divider) which varies voltage between 0 to 5v to vary the contrast.
- **Back-light:** The PIN 15 and 16 are used as backlight. The led backlight can be powered through a simple current limiting resistor as we do with normal led.

Most IoT applications use **LCD with I2C module,** let's see why…

➢ *Why use LCD with I2C Module:*

To use LCD directly with a microcontroller, you would need at least 6 pins – RS, EN, D7, D6, D5, and D4 (with R/W pin permanently grounded for write operation). When you are engaged in a complex project bundled with large number of modules, you may face pin shortage for your microcontroller. In such situations, I2C LCD comes handy making use of only 2 pins (SDA & SCL) to talk to MCU, thereby saving at least 4 pins. If you already have I2C devices, this LCD module actually uses no more pins at all.

Before moving further, let's see how an LCD can be controlled by I2C. In the image shown above, you can see a backpack module fitted to the back of the LCD. He does the work using an I/O expander PCF8574, that communicates with the micro-controller by I2C.

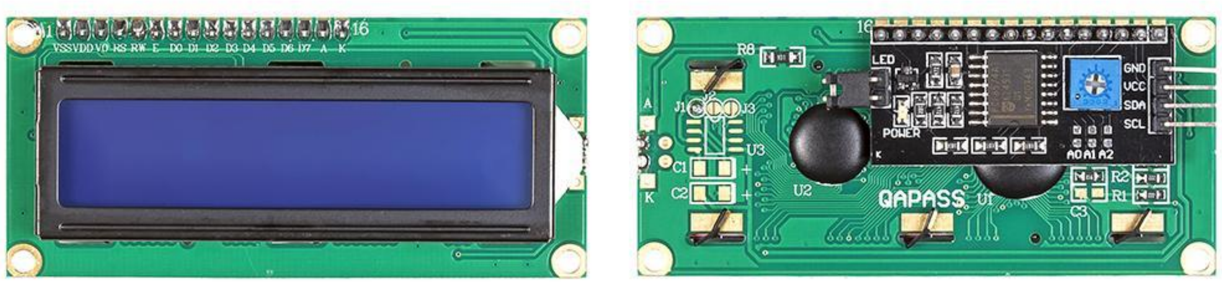


Figure 7: Lcd with I2C Module

➢ *Remote 8-bit I/O expander PCF8574:*

The PCF8574 provides general purpose remote I/O expansion for most microcontroller families via the two-line bidirectional bus (I2C-bus). The device consists of an 8-bit quasi-bidirectional port and an I2C-bus interface.

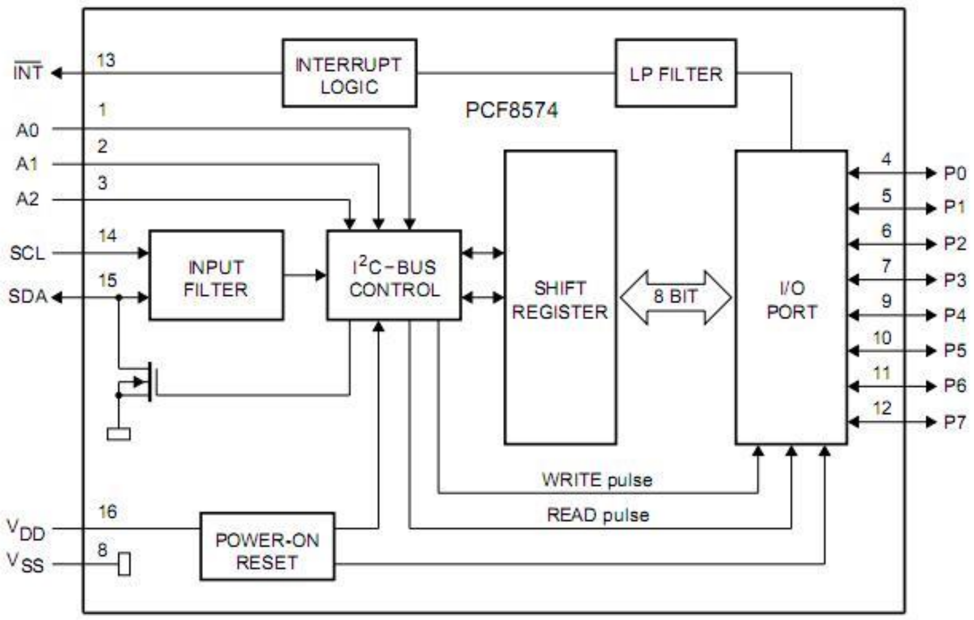


Figure 8: Remote 8-bit I/O Expander Module

A quasi-bidirectional I/O can be used as an input or output without the use of a control signal for data direction. The PCF8574 has a low current consumption and includes latched outputs with high current drive capability. It also possesses an interrupt line (INT) which can be connected to the interrupt logic of the microcontroller. By sending an interrupt signal on this line, the remote I/O can inform the microcontroller if there is incoming data on its ports without having to communicate via the I2C-bus. This means that the PCF8574 can remain a simple slave device.

- ***I2C LCD Backpack:***

You can use this with LCD modules that have a HD44780-compatible interface with various screen sizes. The key is that your LCD must have the interface pads in a single row of sixteen, so it matches the pins on the backpack.

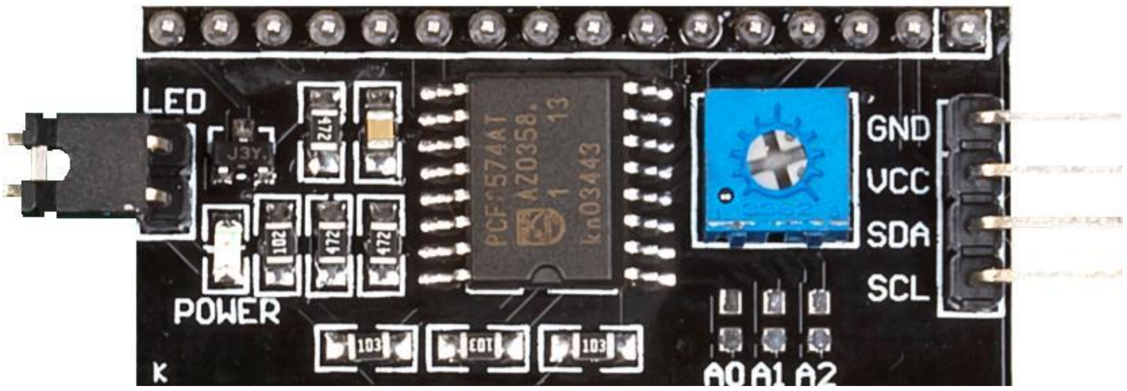


Figure 9: I2C LCD Backpack

As shown in the above image, the backpack has 4 pins namely GND, Vcc, SDA and SCL.
Besides, the backpack has the following key features

- LED jumper should be in position to turn on the back-light.

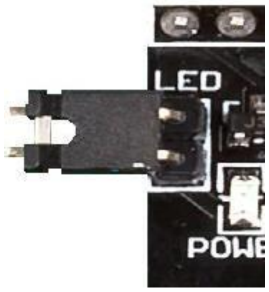


Figure 10: LED Jumper on I2C Module

- The blue potentiometer on the backpack bord adjusts contrast. Adjust it to a position where the characters are bright and the background does not have boxes behind the characters.

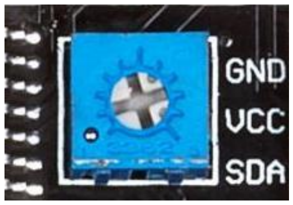


Figure 11: Potentiometer for adjusting Contrast

➢ *Interfacing with NodeMCU:*

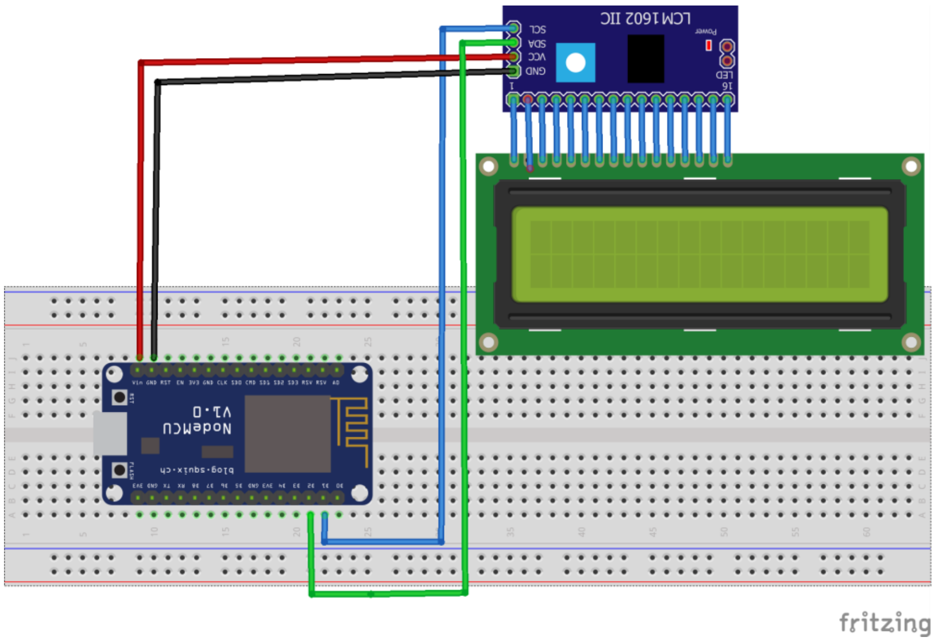


Figure 12: NodeMCU interface with I2C LCD Breadboard

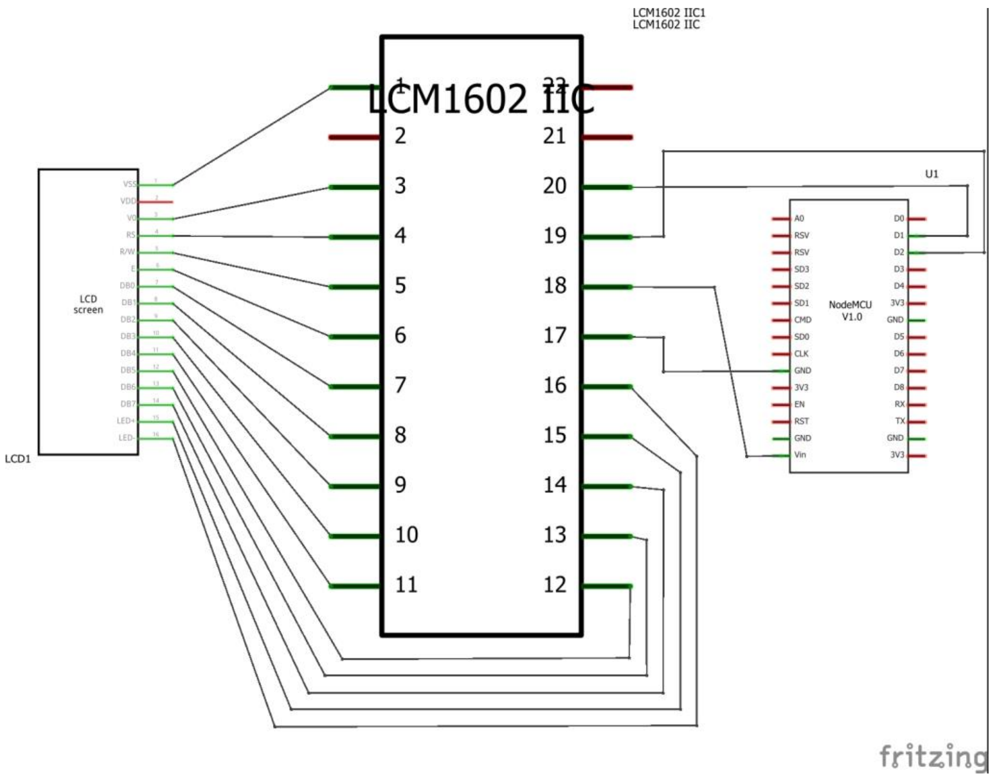


Figure 13: NodeMCU interface with I2C LCD Schematics

- ***I2C LCD Code & Explanation:***

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
```

The sketch starts by including LiquidCrystal_I2C library which is included in infisense library and wire library which comes with Arduino Ide.

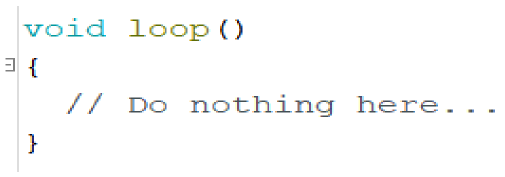
```
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

Set the LCD address to 0x27 for 16 characters and 2-line display

```
void setup()
{
  lcd.begin();
  lcd.clear();
  lcd.backlight();
  lcd.print("Hello, world!");
}
```

In the ‘setup’ function: we will use two functions: The first function is begin (). This is used to specify the dimensions of the display i.e. how many columns and rows the display has.

The second function is clear (). This clears the LCD screen and moves the cursor to the upper left corner.
The backlight () function turns on the backlight of LCD (we can try noBacklight () function for disabling backlight)
we use the print () function to display the message that we see on the first line of the screen.



```
void loop()
{
  // Do nothing here...
}
```

We can leave loop empty for testing LCD module

Button & LED

Button and Led are the most basic components of all circuits, they help to interact with user and communicate message. Push buttons help taking feedback and controls from user and Led helps in communication status of ingoing tasks to user. Let's understand how these works and how to use them:

- ***LED (Light Emitting Diodes):***

They are the most visible type of diode, that emit a fairly narrow bandwidth of either visible light at different colored wavelengths, invisible infra-red light for remote controls or laser type light when a forward current is passed through them.

The LED is basically just a specialized type of diode as they have very similar electrical characteristics to a PN junction diode. This means that an LED will pass current in its forward direction but block the flow of current in the reverse direction.

LED are made from a very thin layer of fairly heavily doped semiconductor material and depending on the semiconductor material used and the amount of doping, when forward biased an LED will emit a colored light at a particular spectral wavelength.

When the diode is forward biased, electrons from the semiconductor's conduction band recombine with holes from the valence band releasing sufficient energy to produce photons which emit a monochromatic (single color) of light. Because of this thin layer a reasonable number of these photons can leave the junction and radiate away producing a colored light output.

Then we can say that when operated in a forward biased direction LED are semiconductor devices that convert electrical energy into light energy.

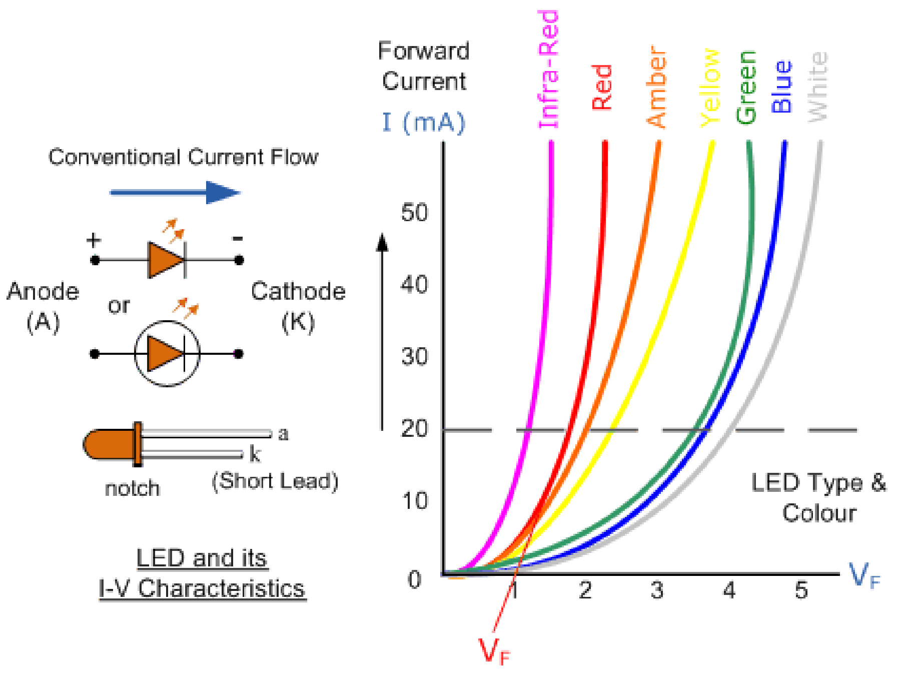


Figure 14: LED and its I-V Characteristics

Figure 14 shows Schematic Symbol of led and its pinouts, longer pin (a) is called as Anode and smaller pin(k) is called Cathode. The graph shoes Current and Voltage Characteristics for different wavelengths (colors).

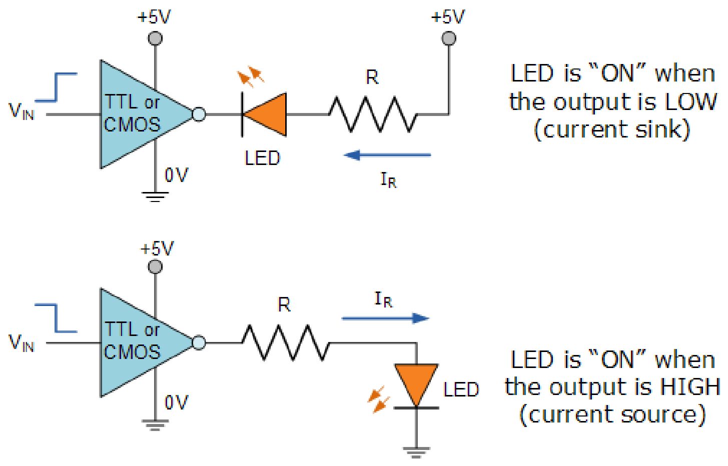


Figure 15: Led Drive

Now that we know what is an LED, we need some way of controlling it by switching it "ON" and "OFF". The output stages of both TTL and CMOS logic gates can both source and sink useful amounts of current therefore can be used to drive an LED. Normal integrated circuits (ICs) have an output drive current of up to 50mA in the sink mode configuration, but have an internally limited output current of about 30mA in the source mode configuration.

Either way the LED current must be limited to a safe value using a series resistor as we have already seen. Below are some examples of driving light emitting diodes using inverting ICs but the idea is the same for any type of integrated circuit output whether combinational or sequential.

- ***Button:***

A Button is a simple switch mechanism to control some aspect of a machine or a process. Buttons are typically made out of hard material, usually plastic or metal. The surface is usually flat or shaped to accommodate the human finger or hand, so as to be easily depressed or pushed. Buttons are most often biased switches, although many un-biased buttons (due to their physical nature) still require a spring to return to their un-pushed state.

A discrete signal (digital signal) supplied to micro-Controller NodeMCU is known as digital input. This signal can be generated manually using a push button switch. Push button switch provides connectivity between its terminals when pressed. When the button is released terminals get disconnected.

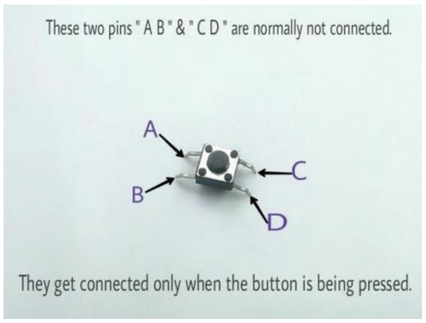

These two pins " A B " & " C D " are normally not connected.
A
B
C
D
They get connected only when the button is being pressed.

Figure 16: Push Button Pinout

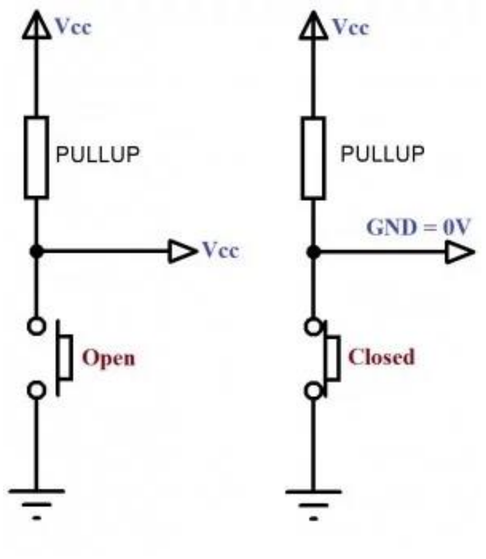


Figure 17: Button Interface with Pullup resistor

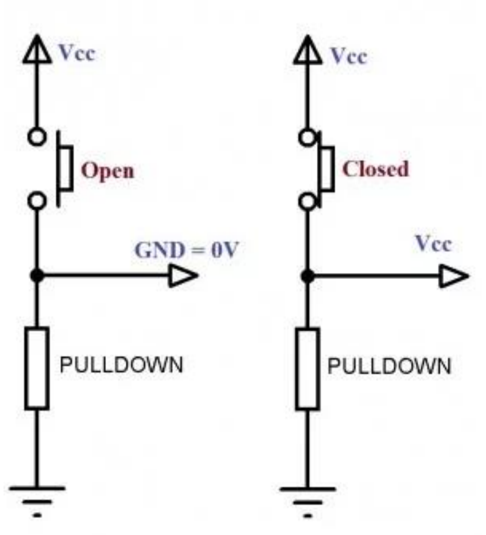


Figure 18: Button Interface with Pullup resistor

- ***Button Debouncing Problem:***

  Switch Bounce happens when you close a mechanical switch. When you close a switch, it tends to literally bounce upon the metal contact which connects the circuit. Usually switches take a few microseconds to a few milliseconds to completely close. What this means in terms of digital logic is that as the switch physically bounces your logic can switch back and forth low-to-high-to-low-etc... until your switch settles down.

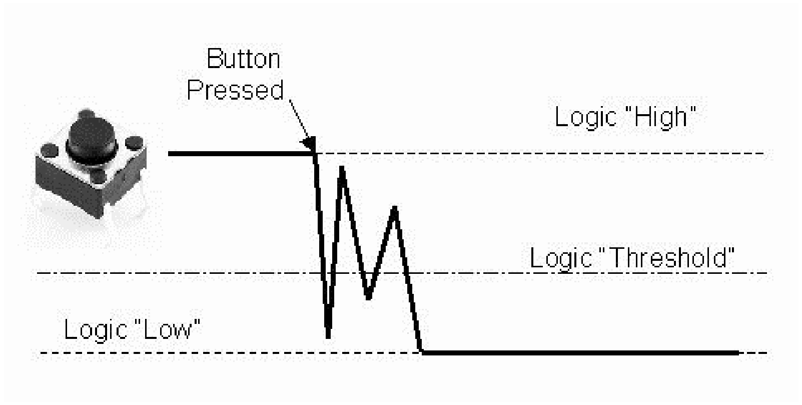


Figure 17: Button Bounce Concept

Button bounce can be eliminated using RC delay in Hardware or by using millis() in software. EasyButton library handles button bounce very easily.

- ***Interfacing:***

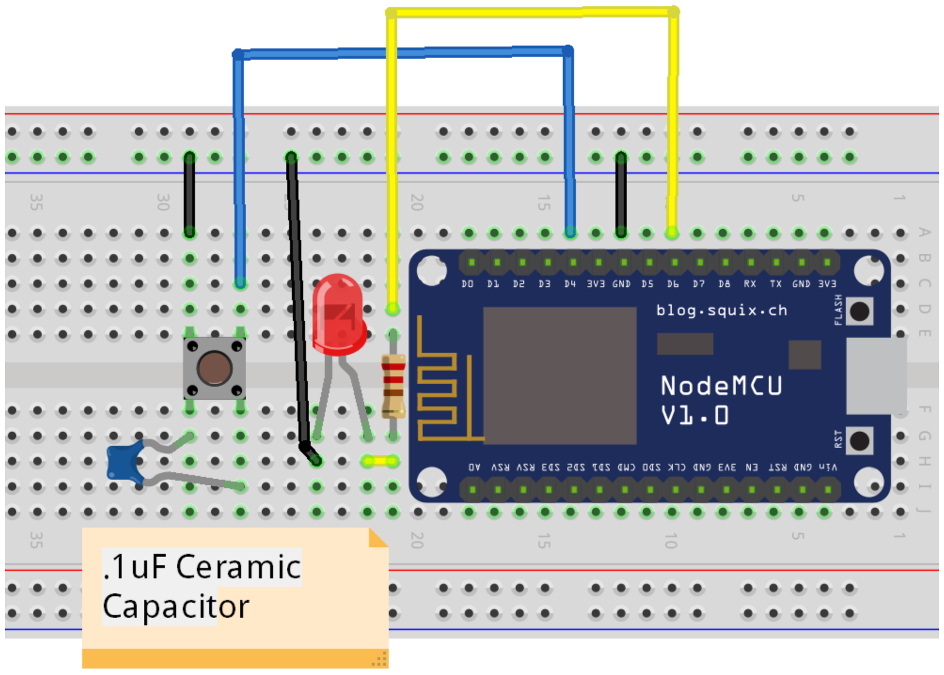
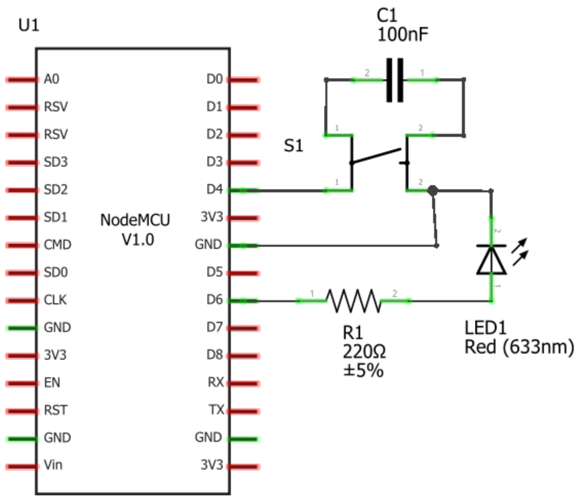


Figure 18: Interfacing LED and Button

- ***Interfacing Schematics:***



- ***Button & LCD Code & Explanation:***

```
#include <EasyButton.h>
```

EasyButton is a small Arduino library for debouncing momentary contact switches like tactile buttons. It uses events and callbacks to trigger actions when a button is pressed once or held for a given duration. It also provides a sequence counter to be able to rise an event when a given pattern of presses has been matched. First, we include the library

```
#define BUTTON_PIN 2  //i.e. D4
#define LED_PIN 12 //i.e. D6
```

Declare the Arduino pin where the button and LED is connected to.

```
EasyButton button(BUTTON_PIN);
```

Create instance of the button.

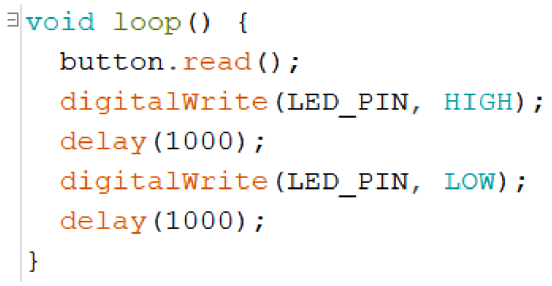
```
void onPressed() {
  Serial.println("Button has been pressed!");
}
```

onPressed () is a Callback function to be called when the button is pressed.

```
void setup() {
  Serial.begin(96000);
  button.begin();
  button.onPressed(onPressed);
  pinMode (LED_PIN, OUTPUT);
}
```

Initialize Serial for debugging purposes.

Initialize the button.
Add the callback function to be called when the button is pressed.
Define pinmode as outputs for LED pin



```
void loop() {
  button.read();
  digitalWrite(LED_PIN, HIGH);
  delay(1000);
  digitalWrite(LED_PIN, LOW);
  delay(1000);
}
```

Continuously read the status of the button.
Triggering LED on and off.
When Button is pressed "*Button has been pressed*" message is displayed on Serial Monitor.

Procedure & Explanations:

Conclusion:

Experiment No: 2A

Capacitive Soil Moisture Sensor and Soil Temperature Sensors

Aim: Understanding and Interfacing Capacitive Soil Moisture Sensor and Soil Temperature Sensors

Description:

Capacitive Soil Moisture Sensor:

Soil moisture sensors measure the volumetric water content in soil. Since the direct gravimetric measurement of free-soil moisture requires removing, drying, and weighing of a sample, soil moisture sensors measure the volumetric water content indirectly by using some other property of the soil, such as electrical resistance, dielectric constant, or interaction with neutrons, as a proxy for the moisture content.

Capacitive Soil moisture sensor measures soil moisture levels by capacitive sensing rather than resistive sensing like other sensors on the market. These are made of corrosion resistant material which gives it an excellent service life. Insert it in to the soil around your plants and impress your friends with real-time soil moisture data! This module includes an on-board voltage regulator which gives it an operating voltage range of 3.3 ~ 5.5V. These are perfect for low-voltage MCUs, both 3.3V and 5V. For compatibility with a Raspberry Pi it needs an ADC converter.

Pin No	Function
1	Ground
2	Vcc
3	Analog Output

Table 1: Capacitive Soil Sensor Pinout

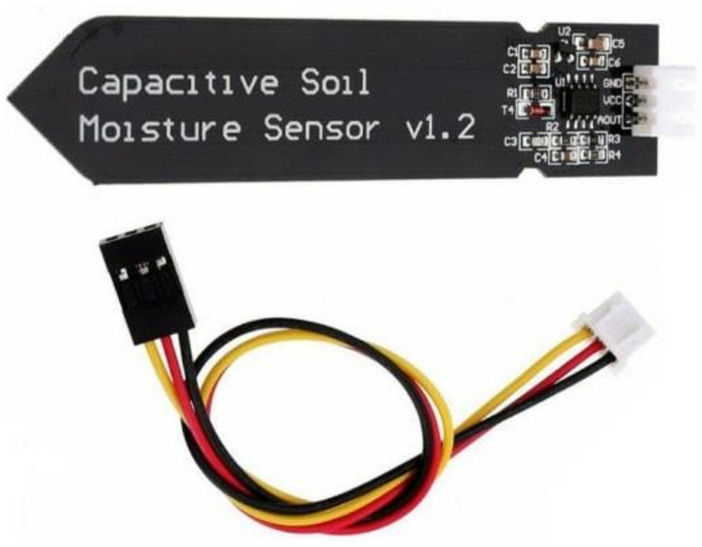


Figure 19:Capacitive Soil Sensor

➢ ***Specifications:***

- Supports 3-pin sensor interface
- Analog output
- Operating Voltage: 3.3 ~ 5.5 VDC
- Output Voltage: 0 ~ 3.0VDC
- Operating Current: 5mA
- Interface: PH2.0-3P
- Dimensions: 3.86 x 0.905 inches (L x W)
- Weight: 15g

➢ ***Working Principal:***

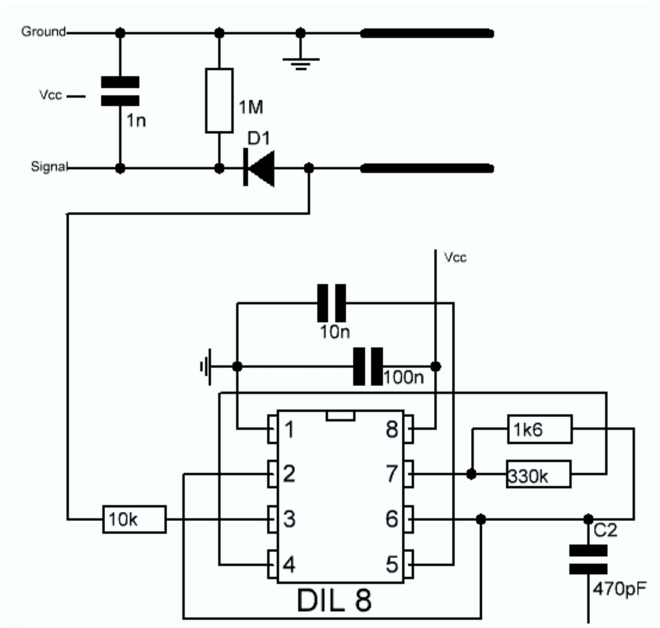


Figure 20: Schematics of Capacitive Soil Sensor

There is a fixed frequency oscillator that is built with a 555 Timer IC. The square wave generated is then fed to the sensor like a capacitor.

The capacitor consists of three pieces: A positive plate, a negative plate and the space in-between the plates, known as the dielectric.

A capacitive moisture sensor works by measuring the changes in capacitance caused by the changes in the dielectric. It does not measure moisture directly (pure water does not conduct electricity well), instead it measures the ions that are dissolved in the moisture These ions and their concentration can be affected by a number of factors, for example adding fertilizer for instance will decrease the resistance of the soil. Capacitive measuring basically measures the dielectric that is formed by the soil and the water is the most important factor that affects the dielectric.

The capacitance of the sensor is measured by means of a 555 based circuit that produces a voltage proportional to the capacitor inserted in the soil. We then measure this voltage by use of an Analog to Digital Converter which produces a number that we can then interpret as soil moisture.

Capacitive measuring has some advantages, It not only avoids corrosion of the probe but also gives a better reading of the moisture content of the soil as opposed to using a resistive soil moisture sensor.

➢ ***Significance of Soil Moisture:***

- The yield of crop is more often determined by the amount of water available rather than the deficiency of other food nutrients
- Soil water acts as nutrient itself and help regulate soil temperature.
- Soil water helps in chemical and biological activities of soil
- Soil Water is also essential for photosynthesis

Soil Temperature Sensor (DS18B20):

Temperature detection is the foundation for all advanced forms of temperature control and compensation. There are four commonly used temperature sensor types:

Negative Temperature Coefficient (NTC) thermistor

A thermistor is a thermally sensitive resistor that exhibits a large, predictable, and precise change in resistance correlated to variations in temperature. An NTC thermistor provides a very high resistance at low temperatures. As temperature increases, the resistance drops quickly. The effective operating range is -50 to 250 °C for glass encapsulated thermistors or 150°C for standard.

Resistance Temperature Detector (RTD)

An RTD, also known as a resistance thermometer, measures temperature by correlating the resistance of the RTD element with temperature. An RTD consists of a film or, for greater accuracy, a wire wrapped around a ceramic or glass core. Platinum RTDs offer a fairly linear output that is highly accurate (0.1 to 1 °C) across -200 to 600 °C. While providing the greatest accuracy, RTDs also tend to be the most expensive of temperature sensors.

Thermocouple

This temperature sensor type consists of two wires of different metals connected at two points. The varying voltage between these two points reflects proportional changes in temperature. Thermocouples are nonlinear, they operate across the widest temperature range, from -200 °C to 1750 °C.

Semiconductor-based sensors

A semiconductor-based temperature sensor is placed on integrated circuits (ICs). These sensors are effectively two identical diodes with temperature-sensitive voltage vs current characteristics that can be used to monitor changes in temperature. They offer a linear response but have the lowest accuracy of the basic sensor types at 1 °C to 5 °C. They also have the slowest responsiveness (5 s to 60 s) across the narrowest temperature range (-70 °C to 150 °C).

The DS18B20 is a 1-wire programmable Temperature sensor from maxim integrated. It is widely used to measure temperature in hard environments like in chemical solutions, mines or soil etc. The constriction of the sensor is rugged and also can be purchased with a waterproof option making the mounting process easy. It can measure a wide range of temperature from -55°C to +125° with a decent accuracy of ±5°C. Each sensor has a unique address and requires only one pin of the MCU to transfer data so it a very good choice for measuring temperature at multiple points without compromising much of your digital pins on the microcontroller.

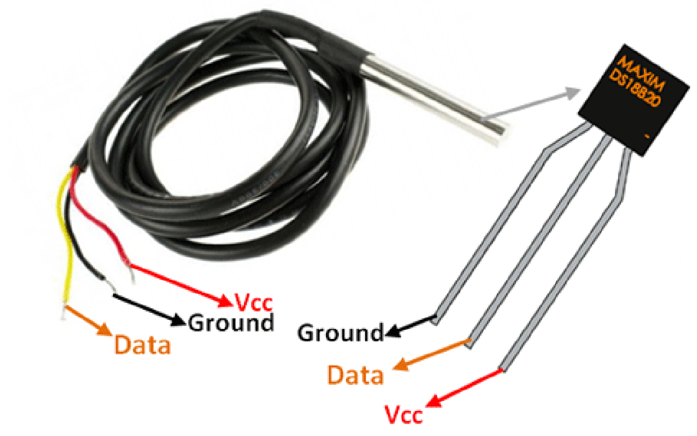


Figure 21: Soil Temperature Sensor

➢ ***DS18B20 Pinout:***

Pin No	**Pin Name**	**Description**
1	Ground	Connect to the ground of the circuit
2	VCC	Powers the Sensor, can be 3.3V or 5V
3	Output	This pin gives output the temperature value which can be read using 1-wire method

➢ ***Specifications:***

- Programmable Digital Temperature Sensor
- Communicates using 1-Wire method
- Operating voltage: 3V to 5V
- Temperature Range: -55°C to +125°C
- Accuracy: ±0.5°C
- Output Resolution: 9-bit to 12-bit (programmable)
- Unique 64-bit address enables multiplexing
- Conversion time: 750ms at 12-bit
- Programmable alarm options
- Available as To-92, SOP and even as a waterproof sensor

- ***DS18B20 Temperature Sensor Working:***

The sensor works with the method of 1-Wire communication. It requires only the data pin connected to the microcontroller with a pull up resistor and the other two pins are used for power as shown below.

The pull-up resistor is used to keep the line in high state when the bus is not in use. The temperature value measured by the sensor will be stored in a 2-byte register inside the sensor. This data can be read by the using the 1- wire method by sending in a sequence of data. There are two types of commands that are to be sent to read the values, one is a ROM command and the other is function command. The address value of each ROM memory along with the sequence is given in the datasheet below. You have to read through it to understand how to communicate with the sensor.

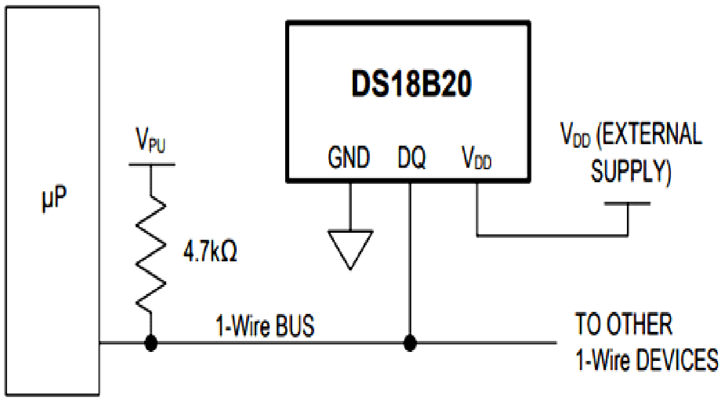


Figure 22: DS18B20 Connection Diagram

<u>Interfacing Diagram and Schematics</u>:

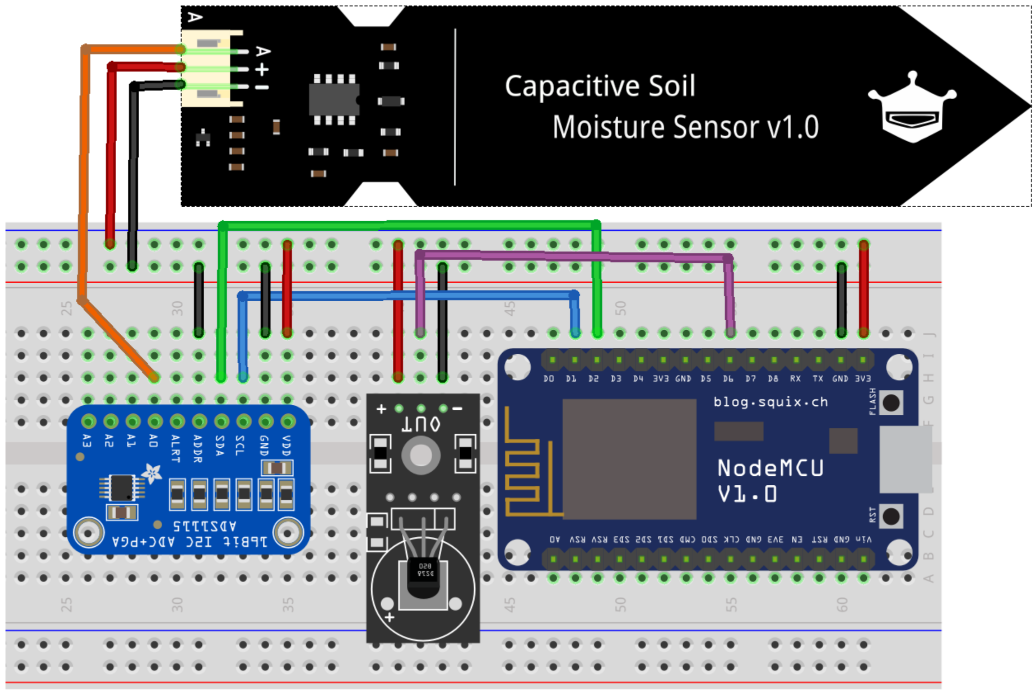

Capacitive Soil Moisture Sensor v1.0
A
+
-
A3 A2 A1 A0 ALRT ADDR SDA SCL GND VDD
ADS1115
16Bit I2C ADC+PGA
OUT
blog.squix.ch
NodeMCU V1.0
FLASH
RST

Figure 23: Interfacing Diagram DS18B20 and Soil Moisture with ADS115 16bit ADC

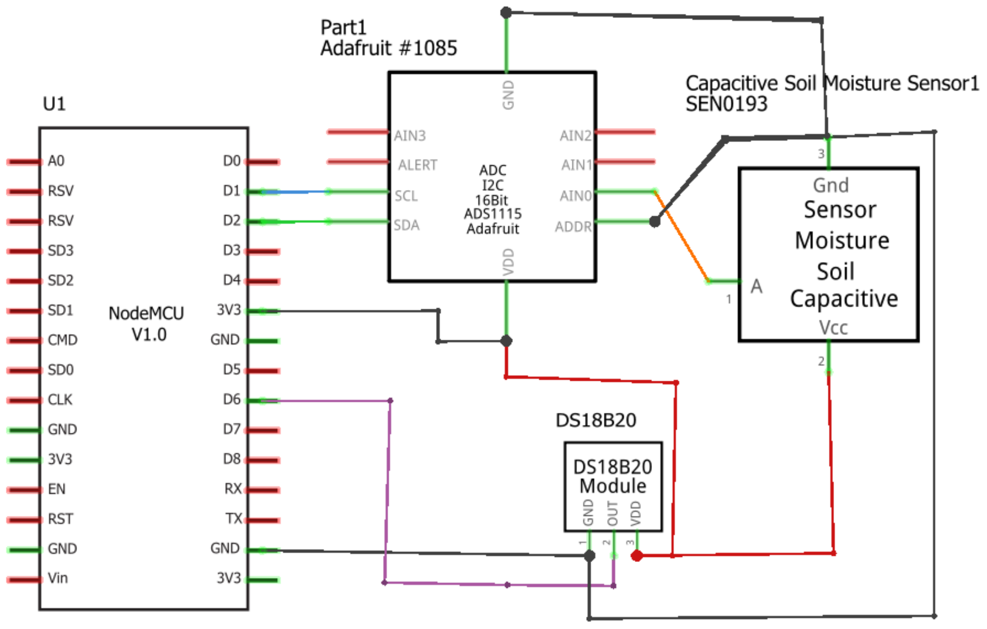


Figure 24: Interfacing Schematics for DS18B20 and Soil Moisture with ADS115 16bit ADC

<u>Code Explanation</u>:

```
#include <OneWire.h>
#include <DallasTemperature.h>
#include <Adafruit ADS1015.h>
```

Include all the header files to code

```
#define ONE_WIRE_BUS 12
```

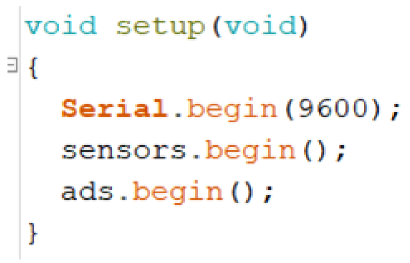
Define the pin to which temperature sensor is connected

```
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
Adafruit_ADS1115 ads(0x48);
```

Setup a oneWire instance to communicate with any OneWire device
Pass oneWire reference to DallasTemperature library
Setup ads115 instance to communicate with ADC

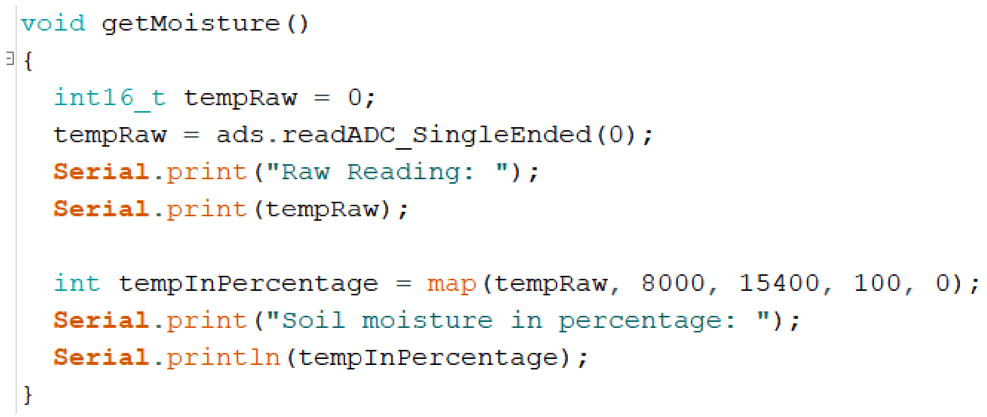
```
getMoisture();
getSoilTemperature();
```

Function declaration for functions to read moisture and temperature



```
void setup(void)
{
  Serial.begin(9600);
  sensors.begin();
  ads.begin();
}
```

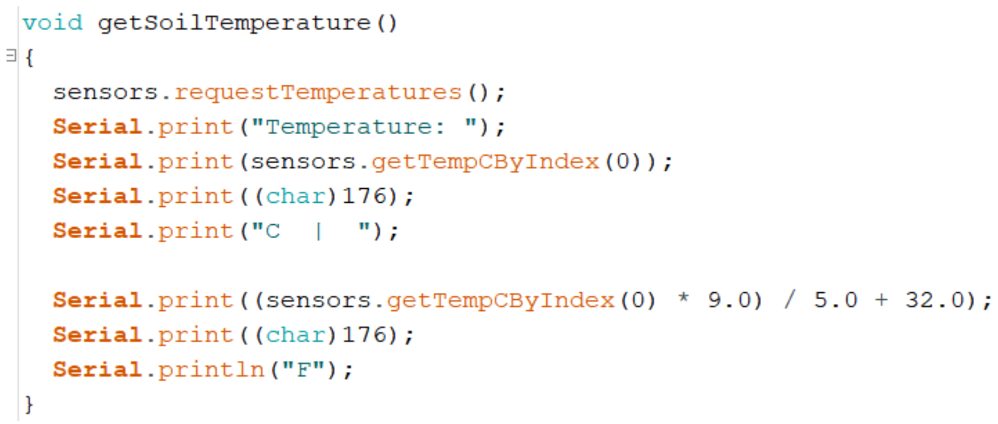
Set Baud rate for communication
Start up the library for DS18B20 and for ADC



```
void getMoisture()
{
  int16_t tempRaw = 0;
  tempRaw = ads.readADC_SingleEnded(0);
  Serial.print("Raw Reading: ");
  Serial.print(tempRaw);

  int tempInPercentage = map(tempRaw, 8000, 15400, 100, 0);
  Serial.print("Soil moisture in percentage: ");
  Serial.println(tempInPercentage);
}
```

Define variable for raw values
Measure raw data from adc by calling readADC_SingleEnded(CHANNEL_NUMBER) method and finally convert it to percentages: 0% for Dry soil and 100% shows wet soil.



```
void getSoilTemperature()
{
  sensors.requestTemperatures();
  Serial.print("Temperature: ");
  Serial.print(sensors.getTempCByIndex(0));
  Serial.print((char)176);
  Serial.print("C  |  ");

  Serial.print((sensors.getTempCByIndex(0) * 9.0) / 5.0 + 32.0);
  Serial.print((char)176);
  Serial.println("F");
}
```

Send the command to get temperatures
print the temperature in Celsius
print the temperature in Fahrenheit

Procedure:
Conclusion:

Experiment No:

Capacitive Soil Moisture Sensor and Soil Temperature Sensors with InfiIot Cloud

Aim: Understanding and Interfacing Capacitive Soil Moisture Sensor and Soil Temperature Sensors

Description:

LCD Module (16*2)

- ***Pin Configuration:***

Interfacing Diagram and Schematics:

Code Explanation:

Procedure:
Conclusion:

Conclusion:

Experiment No:

Interfacing Leaf Sensor and Rain Sensor

<u>Aim</u>: To interface Infi-Leaf leaf sensor module and rain Sensor module with micro-controller

<u>Description</u>:

Infi-Leaf Sensor

Leaf wetness refers to the presence of free water on plant leaves, and can result from several factors, including precipitation intercepted by the canopy and dew. Dew forms as a result of the condensation of water vapor on a canopy that has experienced radiative cooling to the atmosphere, causing their temperature to drop below the dew point temperature of the surrounding air space.

There are two possible sources for the water vapor that condenses on the canopy. The first and most commonly noted and modeled source of water vapor is from the atmosphere. This contribution to the overall dew amount is referred to as dew fall. The second source is evaporation from the soil surface. This process is referred to as distillation, or occasionally, dew rise. Another natural source of free water on leaves is guttation, which occurs in some, but not all crops. In guttation water is exuded from the interior of the leaf onto the surface.

- ***Importance of Leaf Wetness:***

The presence of water on the canopy, in the form of dew, intercepted precipitation or irrigation has an impact on the infection, germination and sporulation of many fungal diseases.

Information regarding the wetness duration is used as an input in disease warning systems, which aid growers in determining the appropriate time for the use of preventative measures, such as fungicide application.

The model accumulates daily severity ratings that depend on the duration of wetness and the average temperature during the time period. These severity values accumulate daily until the suggested threshold is reached, at which time a grower should apply fungicides.

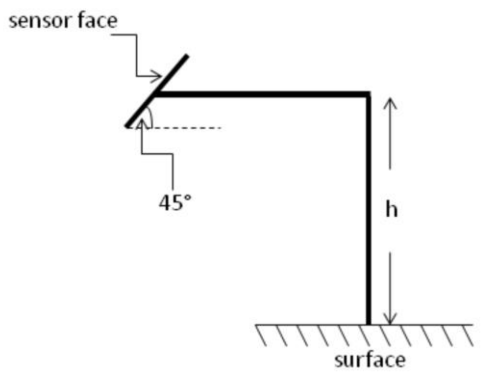


Figure 25: Placement orientation for leaf sensor

➢ *Technical Specifications and features of Leaf Wetness:*

- Operating Voltage: 3.3-5.5 VDC
- Output Voltage: 0-3 VDC (Analog)
- Operating Current: 5mA
- Interface: PH2.0-3P
- Mimics an actual Leaf
- Leaf Wetness Sensing using capacitive grid sensor
- Unmatched accuracy in an easy to use sensor
- Can sense milligram levels of water condensing on leaf surface

➢ *Components and Working of Leaf Wetness:*

Infi-Leaf measures leaf surface wetness by measuring the dielectric constant of the sensor's upper surface. It has a very low power requirement, which allows for many measurements over a long period of time (such as a growing season) with minimal battery usage. Infi-Leaf sensor also has very high resolution, which allows for detection of very small amounts of water (or ice) on the sensor surface. Water on the sensor surface does not need to bridge electrical traces to be detected, as is common with resistance-based surface wetness sensors. This means that the Infi-Leaf does not need to be painted before use, which eliminates the need for individual sensor calibration.

Pin No	**Function**
1	Vcc
2	Ground
3	Analog Output

Table 3: Leaf Sensor Pinout

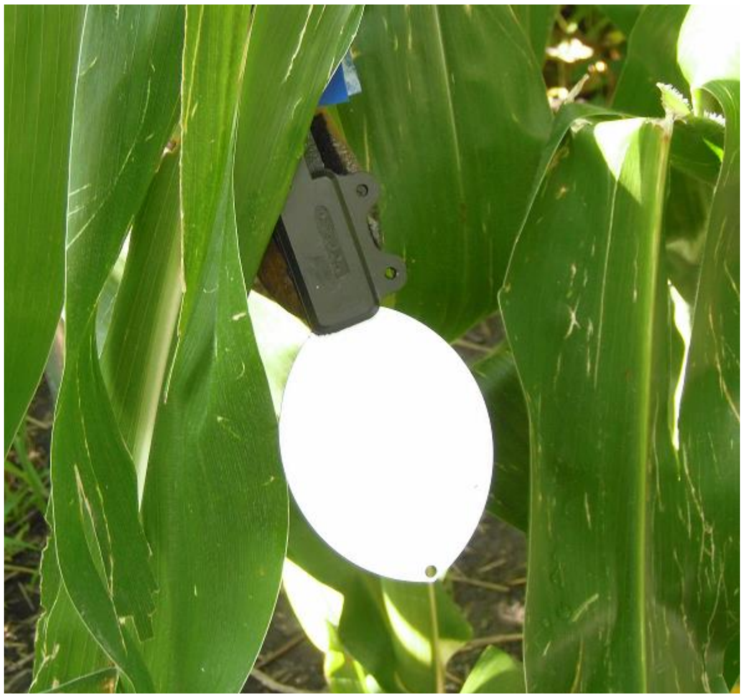


Figure 26: Leaf Sensor

Rain Sensor

Nowadays, conserving water as well as its proper usage is essential in everyone’s life. Here is a sensor namely rain sensor which is used to detect the rain and generate an alarm. So, we can conserve water to use it later for different purposes. There are several methods available for conserving water like harvesting, etc. Using this method we can increase the level of underground water. These sensors are mainly used in the field like automation, irrigation, automobiles, communication, etc. This article discusses a simple as well as reliable sensor module which can be available at low cost in the market.

A rain sensor is one kind of switching device which is used to detect the rainfall. It works like a switch and the working principle of this sensor is, whenever there is rain, the switch will be normally closed.

The rain sensor module/board is shown below. Basically, this board includes nickel coated lines and it works on the resistance principle. This sensor module permits to gauge moisture through analog output pins & it gives a digital output while moisture threshold surpasses.

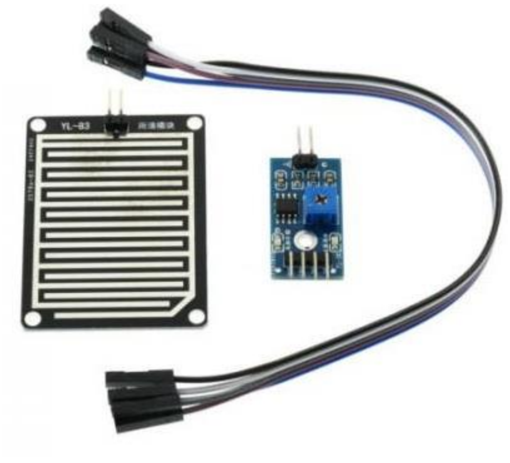


Figure 28: Soil Sensor

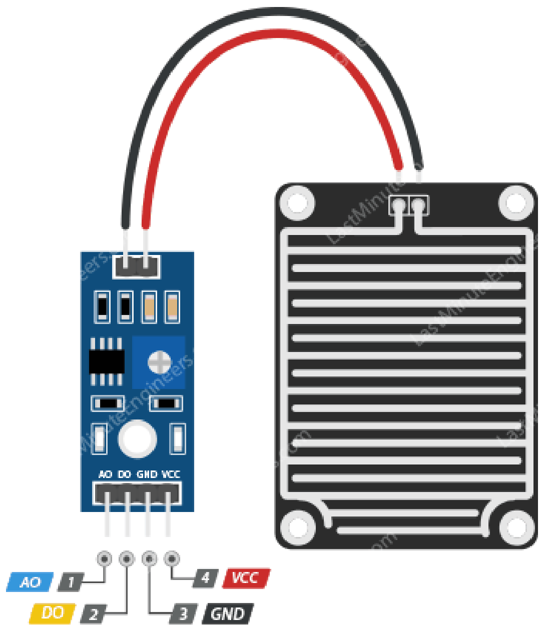


Figure 27: Rain Sensor pinout and connection

- ***Why measure Rain?***
  - Precipitation, especially rain, has a dramatic effect on agriculture. All plants need at least some water to survive, therefore rain (being the most effective means of watering) is important to agriculture.
  - Drought can kill crops and increase erosion, while overly wet weather can cause harmful fungus growth.
  - Well irrigated plants suffer less from pests and diseases due to strong immune systems.
  - Heat stress is reduced and plants produce higher yield and better-quality produce.
  - Plants must have water in order to carry out photosynthesis

➢ ***Specifications and Features of Rain Sensor:***
- o Input Voltage: 3.3-5VDC
- o Output type: Analog Voltage
- o Operating Current: 10mA
- o Anti-Oxidation and Anti-Conductivity
- o Rain sensor board resistance when Wet-100KΩ and when Dry- 2MΩ

➢ ***Working Principle of Rain Sensor:***

The sensing pad with series of exposed copper traces, together acts as a variable resistor (just like a potentiometer) whose resistance varies according to the amount of water on its surface.

This resistance is inversely proportional to the amount of water:

- The more water on the surface means better conductivity and will result in a lower resistance.
- The less water on the surface means poor conductivity and will result in a higher resistance.

The sensor produces an output voltage according to the resistance, which by measuring we can determine whether it's raining or not.

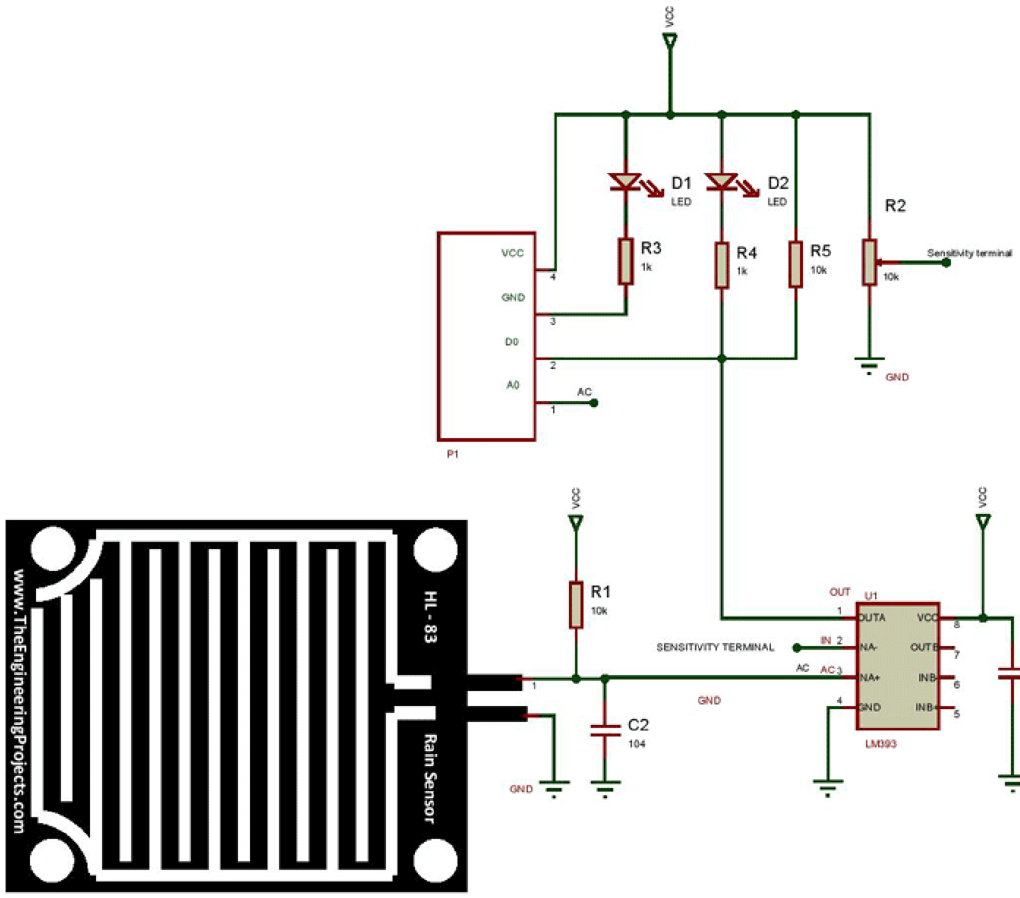


Figure 29: Rain Sensor Schematics

The sensor contains a sensing pad with series of exposed copper traces that is placed out in the open, possibly over the roof or where it can be affected by rainfall. Usually these traces are not connected but are bridged by water. The sensor also contains an electronic module that connects the sensing pad to the Arduino.

The module produces an output voltage according to the resistance of the sensing pad and is made available at an Analog Output (AO) pin. The same signal is fed to a LM393 High Precision Comparator to digitize it and is made available at an Digital Output (DO) pin. The module has a built-in potentiometer for sensitivity adjustment of the digital output (DO). You can set a threshold by using a potentiometer; So that when the amount of water exceeds the threshold value, the module will output LOW otherwise HIGH.

Rotate the knob clockwise to increase sensitivity and counterclockwise to decrease it. Apart from this, the module has two LEDs. The Power LED will light up when the module is powered. The Status LED will light up when the digital output goes LOW.

Interfacing Diagram and Schematics:

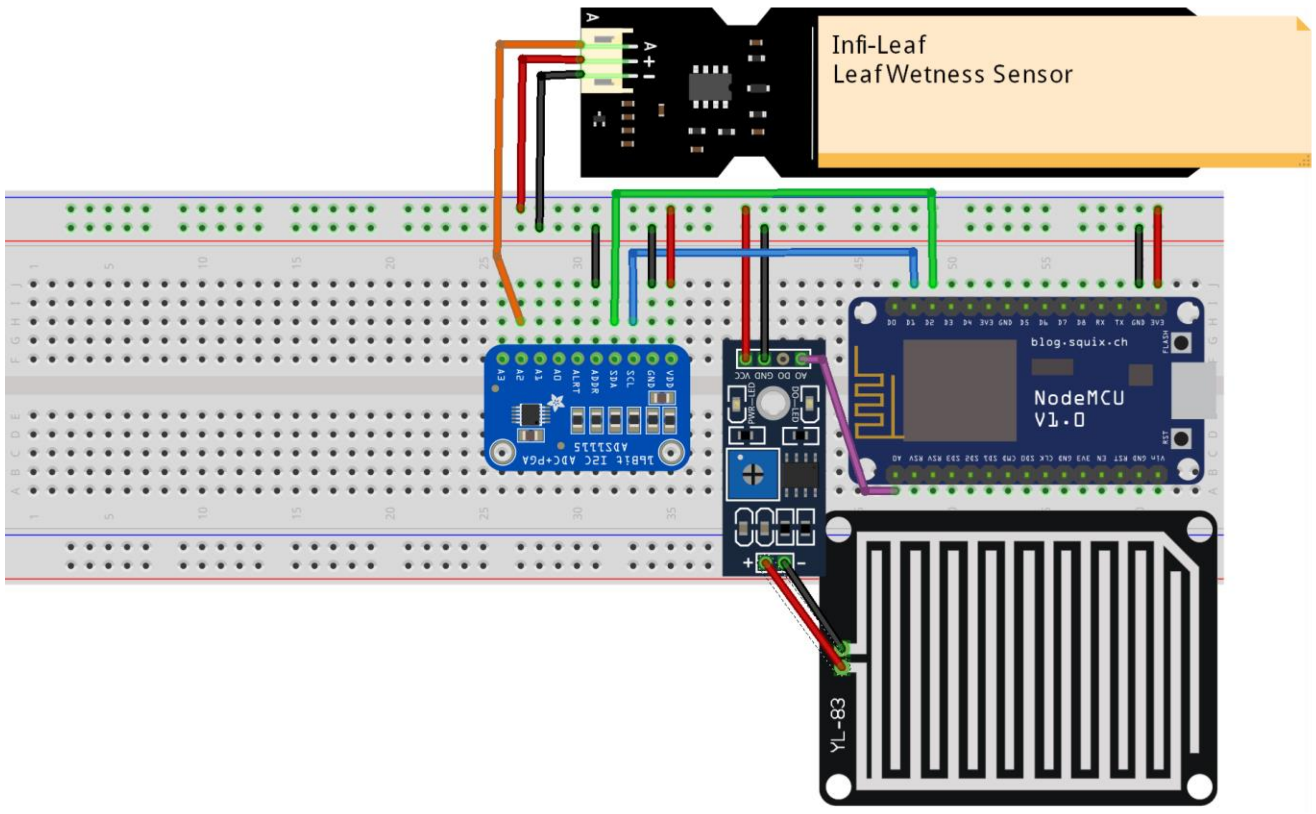


Figure 30: Leaf Sensor and Rain Sensor Interfacing

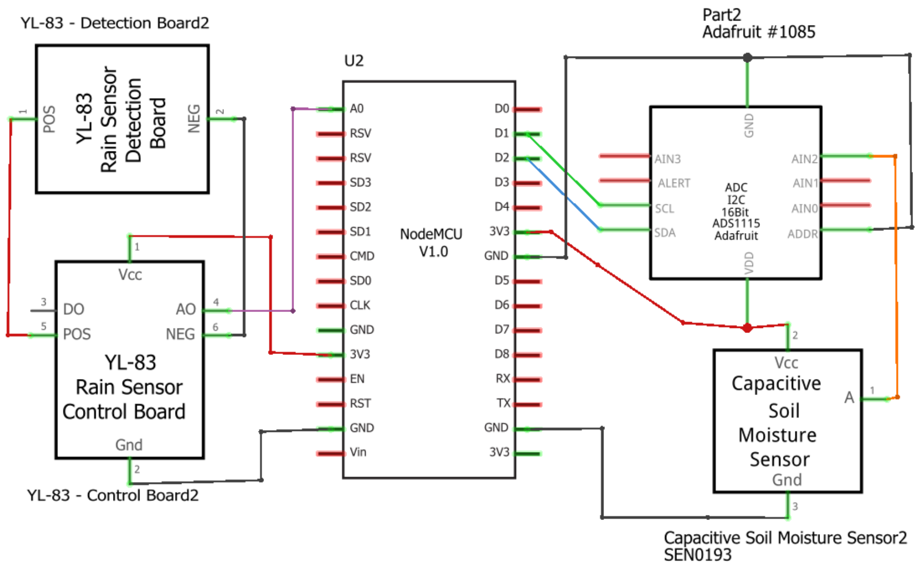


Figure 31: Leaf Sensor and rain Sensor Schematic Diagram

<u>Code Explanation</u>:

<u>Procedure</u>:
<u>Conclusion</u>:

<u>Conclusion</u>:

Experiment No:

Interfacing UV and Light Sensor

Aim: Interfacing UV Sensor (ML8511) and Light Sensor (BH1750) with micro-controller

Description:

UV Sensor (ML8511)

The ML8511 UV sensor is easy to use the ultraviolet light sensor. The MP8511 UV (ultraviolet) Sensor works by outputting an analogue signal in relation to the amount of UV light that's detected. This breakout can be very handy in creating devices that warn the user of sunburn or detect the UV index as it relates to weather conditions.
This sensor detects 280-390nm light most effectively. This is categorized as part of the UVB (burning rays) spectrum and most of the UVA (tanning rays) spectrum. It outputs an analogue voltage that is linearly related to the measured UV intensity (mW/cm2). If your microcontroller can do an analogue to digital signal conversion then you can detect the level of UV!

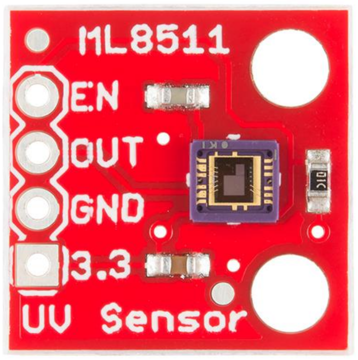


Figure 32: UV Sensor

➢ ***Why measure UV?***

- UV light has been shown to drive increases in the plant production of active substances in medicinal plants, including antioxidant benefits of numerous plants or THC levels in cannabis.
- UV light can also help maintain a healthy growing environment by reducing mold, mildew, and certain plant pests - all of which need alternatives to chemicals due to increasing fungicidal resistance.
- As the price of UV LEDs continues to decline, the ability to cost effectively incorporate targeted UV into the growing process with the right wavelengths, the right dosage, and at the right time in the life cycle of specific plant species will improve.

- ***Specifications of UV Sensor:***
  - Operating Voltage: 2.7-3.3 VDC
  - Operating Current: 300µA (active), 0.1µA (Standby)
  - Output type: Analog
  - Photodiode sensitive to UV-A and UV-B
  - Embedded operational Amplifier

- ***Principle of working:***

The sensor ML8511 has a UV Photodiode sensitive to UV-A and UV-B. Then it has internal Embedded operational amplifier which will convert photocurrent to voltage output depending on the UV light intensity. Through the voltage output it is easy to interface with external micro controllers and ADC.

The sensor requires 3.3V with low supply current 300µA, and gives output in Analog signal variation. This breakout board can be easily interfaced with all kind of micro controllers (which having ADC) and Arduino boards.

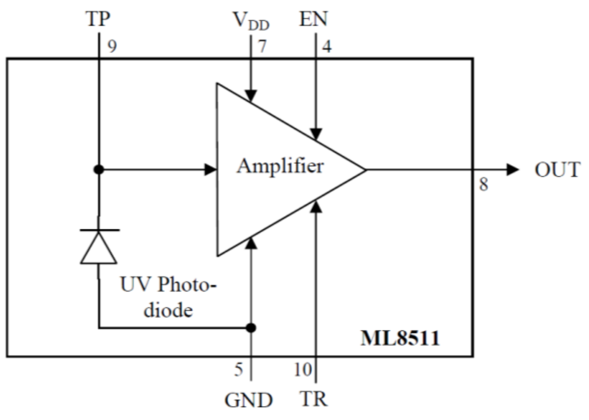


Figure 32: ML8511 Block Diagram

Pin	Function
1	Vcc (+3.3V)
2	Ground
3	Output
4	Enable

Table 2: ML8511 Pinout

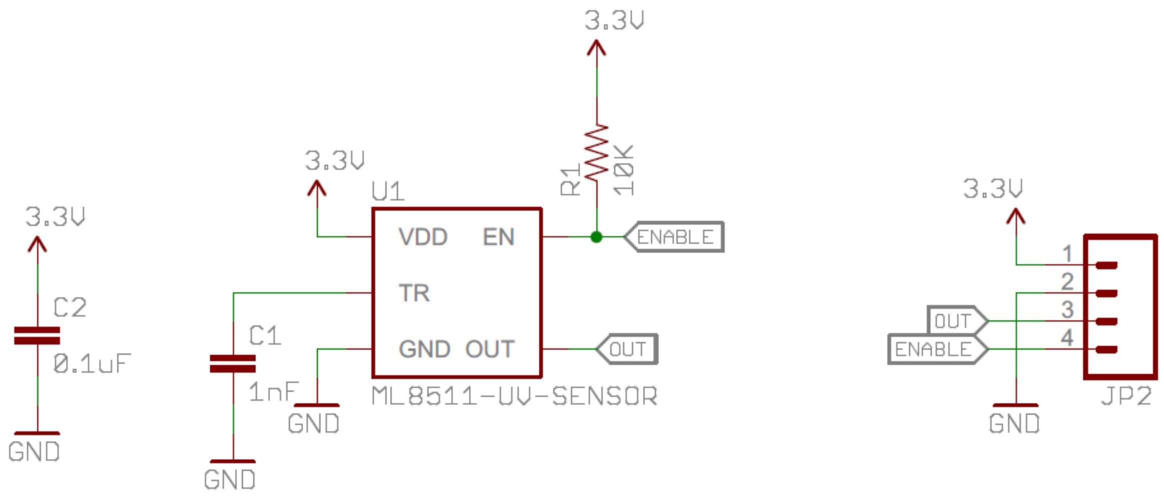


Figure 33: ML8511 Sensor Schematics

Light Sensor (BH1750)

This is a BH1750 light intensity sensor breakout board with a 16-bit AD converter built-in which can directly output a digital signal, there is no need for complicated calculations. This is a more accurate and easier to use version of the simple LDR which only outputs a voltage that needs to be calculated in order to obtain meaningful data. With the BH1750 Light Sensor intensity can be directly measured by the luxmeter, without needing to make calculations. The data which is output by this sensor is directly output in Lux (Lx). When objects which are lighted in homogeneous get the 1 lx luminous flux in one square meter, their light intensity is 1lx. Sometimes to take good advantage of the illuminant, you can add a reflector to the illuminant. So that there will be more luminous flux in some directions and it can increase the illumination of the target surface.

The BH1750 is a light intensity sensor that can be used to adjust the brightness of display in mobiles and LCD displays. It can also be used to turn the headlights of cars on/off based on the outdoor lighting. The sensor uses I2C communication protocol so that makes it super easy to use with microcontrollers. The SCL and SDA pins are for I2C. There is no calculation needed to measure the LUX value because the sensor directly gives the lux value. Actually, it measures the intensity according to the amount of light hitting on it. It operates on voltage range of 2.4V-3.6V and consumes really small current of 0.12mA. The results of the sensor does not depends upon the light source used and the influence of IR radiation is very less. There are very less chances of any error because the variation in measurement is as low as +/-20%.

Figure 35: BH1750 (Light) Sensor

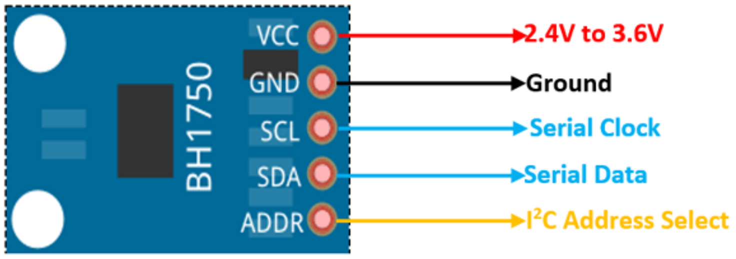


Figure 34: BH1750 (Light) Sensor and pinout

➢ ***Why measure Light (LUX)?***

- Process of photosynthesis uses light energy which are used to combine CO2 with water to make sugars and O2. Higher the light intensity, more rapid this process
- In places high light intensities, leaves are smaller and the internodal distance (distance between where leaves and branches emerge) will be shortened.

➢ ***Specifications of Light Sensor (BH1750):***

- Supply Voltage: 2.4-3.6VDC
- Operating Current: 0.12mA
- Interface: I2C
- No need of calculations-Direct Digital values is given as output
- On-board ADC to convert analog light intensity to digital LUX values
- Consists of 50Hz/60Hz noise rejection function

➢ ***Working principle of Light Sensor (BH1750):***

Ambient light sensors contain a photodiode which can sense light and convert it into electricity. Light is measured depending upon its intensity. From the block diagram, PD is the photodiode which is used to sense the light. Its response is approximate to the human eye response.

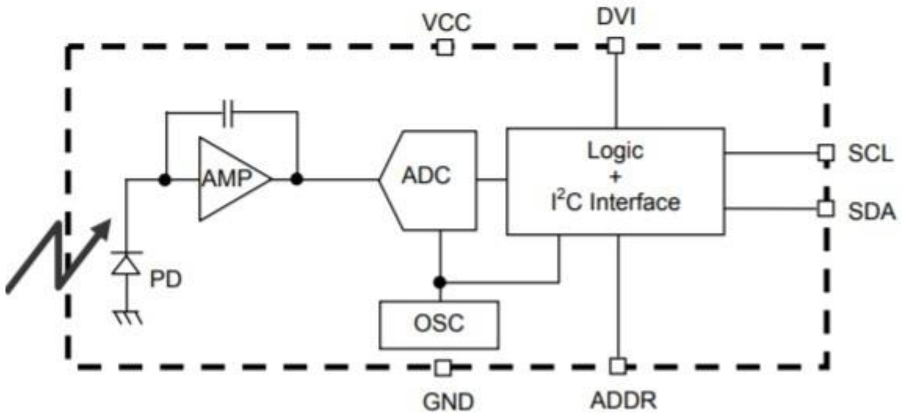


Figure 36: Block Diagram Light Sensor

In BH1750 sensor an Opamp – AMP is integrated which converts the current from the photodiode into voltage. BH1750 uses an ADC to convert the analog values provided by AMP into digital values. The logic+I2C block shown in the block diagram is the unit where illuminance values are converted to LUX and I2C communication process takes place. OSC is the internal clock oscillator of 320kHz, used as a clock for internal logic.

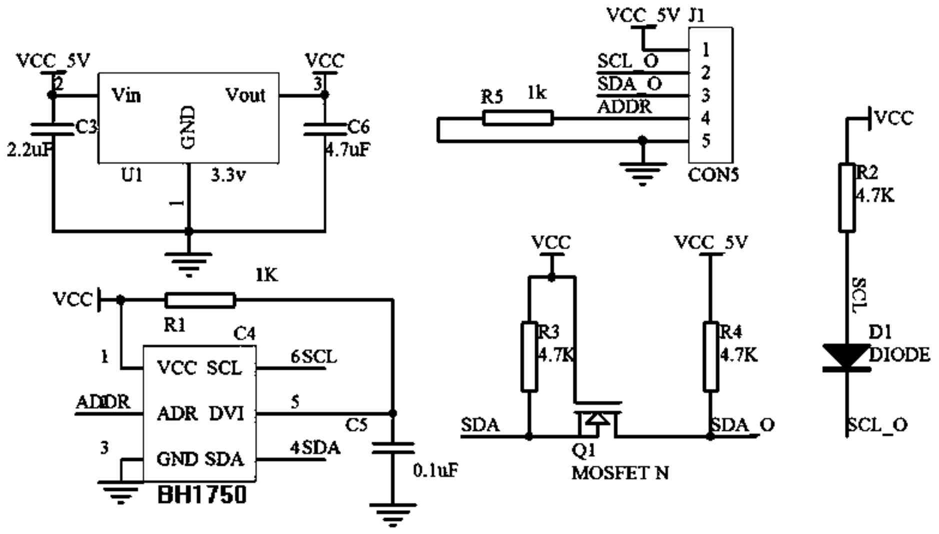


Figure 37: Light Sensor (BH1750) Schematics Diagram

Interfacing Diagram and Schematics:

Code Explanation:

Procedure:
Conclusion:

Conclusion: