A

MAJOR PROJECT REPORT

on

IoT Based Temperature and Humidity Monitoring System

Submitted in partial fulfilment of the requirements of the degree of

BACHELOR OF TECHNOLOGY

Under Guidance of

Mr. Yogendra Singh Solanki
Assistant Professor
Electronics & Communication Engineering
TINJRIT, Udaipur

Submitted by

1. Kushal Soni (Roll No.17ETCEC011)
2. Nitish Malviya (Roll No.17ETCEC013)
3. Sonal Kumari (Roll No.17ETCEC018)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR

OCTOBER - 2021

A

MAJOR PROJECT REPORT

on

IoT Based Temperature and Humidity Monitoring System

Submitted in partial fulfilment of the requirements of the degree of

BACHELOR OF TECHNOLOGY

Under Guidance of

Mr. Yogendra Singh Solanki
Electronics & Communication Engineering
Assistant Professor
TINJRIT, Udaipur

Submitted by

1. Kushal Soni (Roll No. 17ETCEC011)
2. Nitish Malviya (Roll No. 17ETCEC013)
3. Sonal Kumari (Roll No. 17ETCEC018)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR

OCTOBER - 2021

Department of Electronics and Communication Engineering
Techno India NJR Institute of Technology, Udaipur

Certificate

This is to certify that this Major Project report entitled "IoT Based Temperature and Humidity Monitoring" by Kushal Soni, Nitish Malviya and Sonal Kumari have completed the work under my supervision and guidance, hence approved for submission in partial fulfilment for the award of degree of Bachelor of Technology in Electronics and Communication to the Department of Electronics and Communication Engineering, Techno India NJR Institute of Technology, Udaipur during academic session 2017-2021.

Mr. Yogendra Singh Solanki
Assistant Professor
Dept. of E.C.E, TINJRIT, Udaipur
Date:

Mr. Pradeep Chhawchariya
Head of Department
Dept. of E.C.E TINJRIT, Udaipur
Date:

Department of Electronics and Communication Engineering
Techno India NJR Institute of Technology, Udaipur

Examiner certificate

This is to certify that the following students

Kushal Soni

Nitish Malviya

Sonal Kumari

of final year B.Tech. (Electronics & Communication Engineering), were examined for the project work entitled

"IoT Based Temperature and Humidity Monitoring"

during the academic year 2017 – 2021 at Techno India NJR Institute of Technology, Udaipur

Remarks:

Date:

Signature
(**Internal Examiner**)

Signature
(**External Examiner**)

PREFACE

IoT describes a system where items in the real world, and sensors within or attached to these items, are connected to the Internet via wireless and wired Internet connections. Machine-to-machine communications and intelligence drawn from the devices and therefore the network can enable businesses to alter bound basic tasks while not looking at central or cloud-based applications and services. Technology has changed our lives day by day. The system that was done manually is developed into an automated system that can save time and energy. Temperature and humidity values are important for many types of users or systems such as farmers, offices, cars, humidors, museums, industrial spaces, including a greenhouse. Temperature and humidity values are also useful or weather prediction. In this paper, IoT based temperature and humidity monitoring and control system is proposed. In this system, temperature and humidity values of the environment will be monitored, stored and displayed on the web via the Internet.

ACKNOWLEDGMENT

We take this opportunity to record our sincere thanks to all who helped us to successfully complete this work. Firstly, we are grateful to our supervisor Mr. Yogendra Singh Solanki for his invaluable guidance and constant encouragement, support and most importantly for giving us the opportunity to carry out this work.

We would like to express our deepest sense of gratitude and humble regards to our **Head of Department Mr. Pradeep Chhawcharia** for giving invariable encouragement in our endeavours and providing necessary facility for the same. Also, a sincere thanks to all faculty members of ECE, TINJRIT for their help in the project directly or indirectly.

Finally, we would like to thank my friends for their support and discussions that have proved very valuable for us. We are indebted to our parents for providing constant support, love and encouragement. We thank them for the sacrifices they made so that we could grow up in a learning environment. They have always stood by us in everything we have done, providing constant support, encouragement and love.

Kushal Soni, 17ETCEC011

Nitish Malviya, 17ETCEC013

Sonal Kumari, 17ETCEC018

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

ENGINEERING

TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

Abstract

Monitoring in IoT infra is some kind of logging a Temperature and Humidity Monitoring in a specific environment and store data in a location database and then displaying the user values on the Thing Speak. The core function of our proposed system is to monitor and pass the real-time values of the temperature and humidity of a particular place from any location via internet. The inspection of the current conditions can be visualized in the channels of the things speak IoT platform privately or publicly. It also focuses on the controlling of the IoT devices depend on the situation of this system.

Chapter 1: INTRODUCTION

1.1 Overview

Today, the increased demand of service over the internet necessitated the data collection and exchange in efficient manner. In this sense internet of things (IoT) had promised the ability to provide the efficient data storage and exchange by connecting the physical devices and vehicles via electronic sensors and internet.

As the expeditious of Internet of Things (IoT) is emerging and is accustom for remote monitoring of the surrounding parameters and other stuffs with the use of sensors that acquaint for wireless sensing of real time data and transfer them into the desired form and help to forward the sensed data across the network cloud via 'Internet Connection'. Here the project work deals with The IoT 'ThingSpeak' web service which is a generous open API service that act as a host for the variety of sensors to monitor the sensed data at cloud.

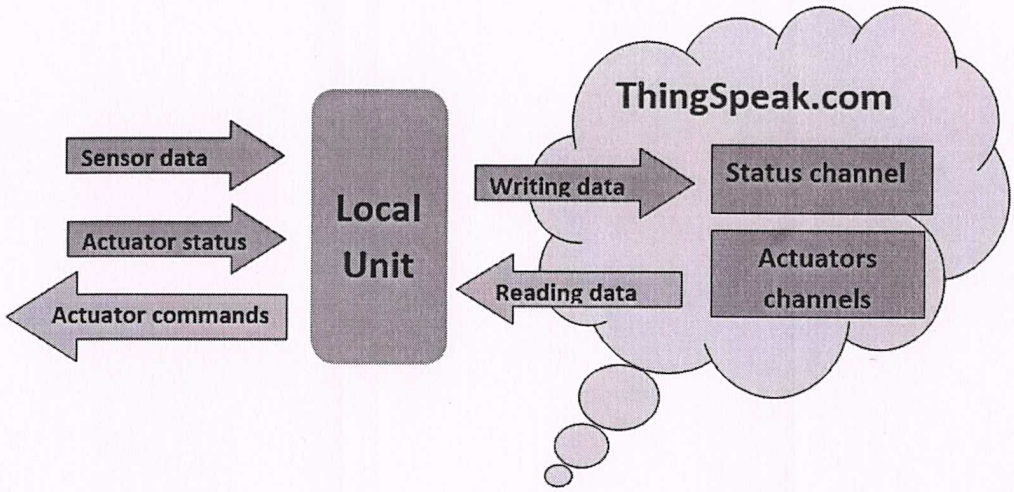


Fig. 1.1: Working Flow

In this system, temperature, humidity values of soil and environment will be monitored, stored and displayed on the web via the Internet.

The "Centre of our IoT project" is ThingSpeak.com. we will capture data from sensors and actuator status; send them to the Internet, and "writing" on a specific ThingSpeak.com Status Channel. Our project will also receive data from the internet, "reading" them from specific ThingSpeak Actuator Channels.

1.2 Process:

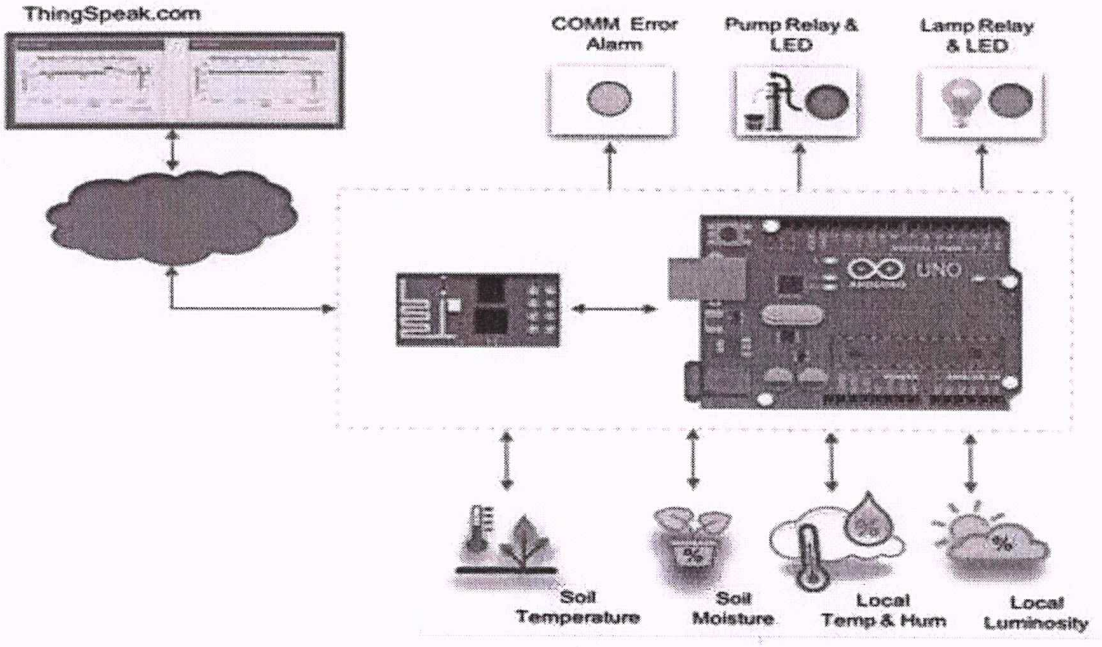
The "Centre of our IoT project" will be the ThingSpeak.com. The local unit (UNO/ESP-01) will capture data from sensors and actuator status; send them to the Internet, "writing" on a specific ThingSpeak.com Status Channel. The local unit will also receive data from the internet, "reading" them from specific ThingSpeak Actuator Channels.

(See the above Block diagram to better understand the flow of data).

Using common sensors, project will capture several data, sending them to the cloud, where everyone can see them thru the internet. To work those data, we will use the service provided by ThingSpeak.com, an open IoT platform that will permit us to collect, analyse and act on those data.

The data to be collected by sensors will be:

- Air Temperature and relative humidity
- Soil Temperature and humidity
- Luminosity



Sensors for data capture

Fig. 1.2: Working Flow Diagram

The project will have 2 actuators:

- Electrical Pump
- Electrical Lamp

The status of those actuators ("ON/OFF"), should be also sent to the cloud.

So, the idea will be to capture those data from the sensors, for example, a plantation and send them to cloud. Based on those data, a user should take the decision based on those statements:

- Turn ON the Pump if the soil humidity is too low
- Turn ON the Lamp if the soil temperature is too low

To remotely command our actuators, we will use an Android App.

Chapter 2: COMPONENT DESCRIPTION

2.1 DC Mini Submersible Water Pump

2.1.1 Description:

3V to 6V submersible pump micro mini submersible water pump 3V to 6V DC water pump for DIYDC pump for hobby kit mini submersible pump motor this is a low cost, small size submersible pump motor which can be operated from a 2.5 ~ 6V power supply. It can take up to 120 litres per hour with very low current consumption of 220ma. Just connect tube pipe to the motor outlet, submerge it in water and power it. Make sure that the water level is always higher than the motor. The dry run may damage the motor due to heating and it will also produce noise.

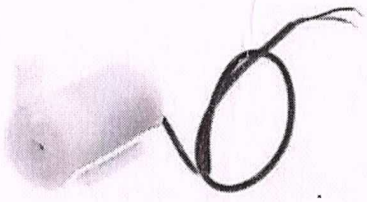


Fig. 2.1: DC Mini Submersible Water Pump

2.1.2 Features:

- Voltage: 2.5-6V
- Maximum lift: 40-110cm / 15.75"-43.4"
- Flow rate: 80-120L/H
- Outside diameter: 7.5mm / 0.3"
- Inside diameter: 5mm / 0.2"
- Diameter: Approx. 24mm / 0.95"
- Length: Approx. 45mm / 1.8"
- Height: Approx. 30mm / 1.2"
- Material: Engineering plastic

- Driving mode: DC design, magnetic driving

2.1.3 Applications:

- Controlled fountain water flow
- Controlled Garden watering systems
- Hydroponic Systems
- Fresh water intake or exhaust systems fish aquarium

2.1 Air and Relative Humidity Sensor (DHT22)

2.2.1 Description:

AM2302 output calibrated digital signal. It applies exclusive digital-signal-collecting-technique and humidity sensing technology, assuring its reliability and stability. Its sensing elements is connected with 8-bit single-chip computer. Every sensor of this model is temperature compensated and calibrated in accurate calibration chamber and the calibration-coefficient is saved in type of programme in OTP memory, when the sensor is detecting, it will cite coefficient from memory. Small size & low consumption & long transmission distance(100m) enables AM2302 to be suited in all kinds of harsh application occasions. Single-row packaged with four pins, making the connection very convenient.

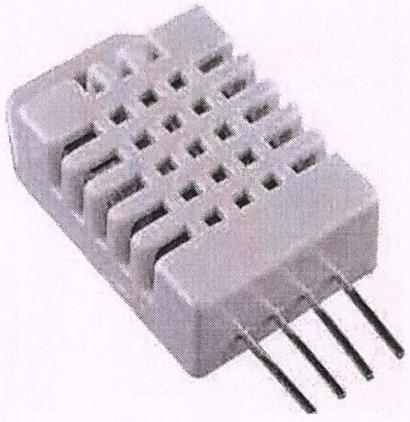


Fig. 2.2: DHT22

2.2.2 Technical Specification:

1.	Model	AM2302
2.	Power supply	3.3-5.5V DC
3.	Output signal	Digital signal via 1-wire bus
5.	Sensing element	Polymer humidity capacitor
6.	Accuracy	Humidity +-2%RH(Max +-5%RH)
7.	Temperature	40~80Celsius

Table 2.1: Technical specification

2.2.3 Operating Specifications:

- Power and Pins:

Power's voltage should be 3.3-5.5V DC. When power is supplied to sensor, don't send any instruction to the sensor within one second to pass unstable status. One capacitor valued 100nF can be added between VDD and GND for wave filtering.

- Communication and Signal:

1-wire bus is used for communication between MCU and AM2302. (Our 1-wire bus is specially designed, it's different from Maxim/Dallas 1-wire bus, so it's incompatible with Dallas 1-wire bus.)

2.2.3 Features & Applications:

- High precision
- Capacitive type
- Full range temperature compensated
- Relative humidity and temperature measurement
- Calibrated digital signal
- Outstanding long-term stability
- Extra components not needed

- Long transmission distance, up to 100 meters
- Low power consumption

2.3 Relay

It is an electronic switch is used to switch AC through DC. The working of relay is based on electromagnetic principle .Relay is an electromagnetically device which is used to isolate two circuit electrically and connected them magnetically. It is very useful device and allow one circuit to other one while they are completely separate.

2.3.1 Relay construction

There are only four main parts in a relay. They are

- Electromagnet
- Movable Armature
- Switch point contacts
- Spring The figures given below show the actual design of a simple relay.

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

2.3.2 How relay works?

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

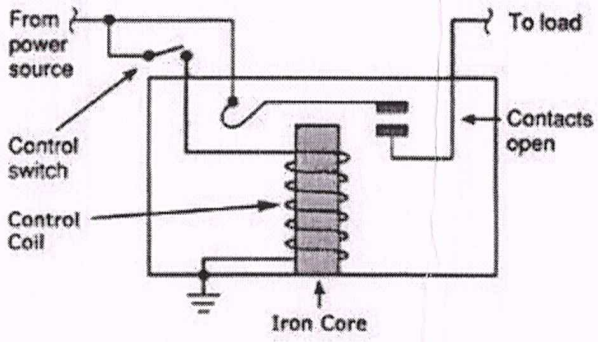


Fig. 2.3: Working of Relay

The diagram shows an inner section diagram of a relay. An iron core is surrounded by a control coil. As shown, the power source is given to the electromagnet through a control switch and through contacts to the load. When current starts flowing through the control coil, the electromagnet starts energizing and thus intensifies the magnetic field. Thus, the upper contact arm starts to be attracted to the lower fixed arm and thus closes the contacts causing a short circuit for the power to the load. On the other hand, if the relay was already de-energized when the contacts were closed, then the contact move oppositely and make an open circuit. As soon as the coil current is off, the movable armature will be returned by a force back to its initial position. This force will be almost equal to half the strength of the magnetic force. This force is mainly provided by two factors. They are the spring and also gravity. Relays are mainly made for two basic operations. One is low voltage application and the other is high voltage. For low voltage applications, more preference will be given to reduce the noise of the whole circuit. For high voltage applications, they are mainly designed to reduce a phenomenon called arcing.

2.3.3 Relay Basics:

The basics for all the relays are the same. Take a look at a 4 – pin relay shown below. There are two colours shown. The green colour represents the control circuit and the red colour represents the load circuit. A small control coil is connected onto the control circuit. A switch is connected to the load. This switch is controlled by the coil in the control circuit. Now let us take the different steps that occur in a relay.

Pole and Throw Relays have the exact working of a switch. So, the same concept is also applied. A relay is said to switch one or more poles. Each pole has contacts that can be thrown in mainly three ways. They are Normally Open Contact (NO) – NO contact is also called a make contact. It closes the circuit when the relay is activated. It disconnects the circuit when the relay is inactive. Normally Closed Contact (NC) – NC contact is also known as break contact. This is opposite to the NO contact. When the relay is activated, the circuit disconnects. When the relay is deactivated, the circuit connects. Change-over (CO) / Double-throw (DT) Contacts – This type of contacts is used to control two types of circuits. They are used to control a NO contact and also a NC contact with a common terminal. According to their type they are called by the names break before make and make before break contacts.

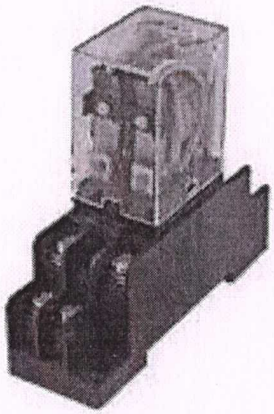


Fig. 2.4: Basic Relay

- **Energized Relay (ON)**:

As shown in the circuit, the current flowing through the coils represented by pins 1and 3 causes a magnetic field to be aroused. This magnetic field causes the closing of the pins 2 and 4. Thus the switch plays an important role in the relay working. As it is a part of the load circuit, it is used to control an electrical circuit that is connected to it. Thus, when the relay in energized the current flow will be through the pins 2 and 4.

- **De – Energized Relay (OFF)**

As soon as the current flow stops through pins 1 and 3, the switch opens and thus the open circuit prevents the current flow through pins 2 and 4. Thus the relay becomes de-energized and thus in off position.

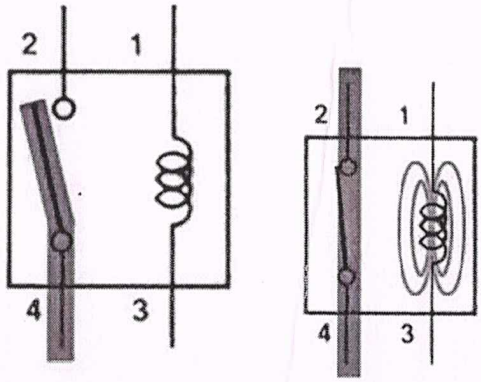


Fig. 2.5: Energized and De-Energized Relay

2.3.4 Types of Relay

Relays are also named with designations like

- **Single Pole Single Throw (SPST)** – This type of relay has a total of four terminals. Out of these two terminals can be connected or disconnected. The other two terminals are needed for the coil.
- **Single Pole Double Throw (SPDT)** – This type of a relay has a total of five terminals. Out f these two are the coil terminals. A common terminal is also included which connects to either of two others. Double Pole Single Throw (DPST) – This relay has a total of six terminals. These terminals are further divided into two pairs. Thus, they can act as two SPST's which are actuated by a single coil. Out of the six terminals two of them are coil terminals. Double Pole Double Throw (DPDT) – This is the biggest of all. It has mainly eight relay terminals. Out of these two rows are designed to be change over terminals. They are designed to act as two SPDT relays which are actuated by a single coil. A double pole double throw relay has two poles and double throws and it can be used to connect two terminals of a circuit at a time. For example, this relay is used for connecting both phase and neutral terminals to the load at a time.

2.4 Optocoupler

An optocoupler is an optical link and it connects two circuits via this link. The optical link is contained within a chip. A Light Emitting Diode inside the chip shines on a photo-diode, photo-transistor or others photo device.

When the photo device sees illumination, the resistance between its terminals reduces. This reduced resistance can activate another circuit.

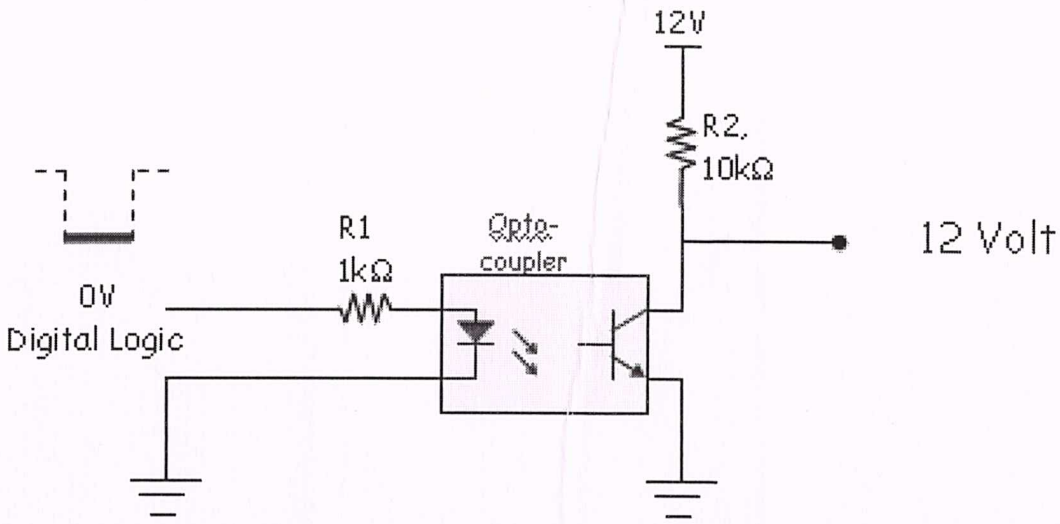


Fig. 2.6: Optocoupler

2.4.1 Operating Principle of Optoisolator

The **working principle of optoisolator** is simple and interesting. The output signal of one circuit can be controlled by varying input signal in another circuit, where the two circuits are electrically isolated. A powerful light emitting diode (LED) is connected across a variable voltage source. By adjusting the input voltage across the LED, the intensity of the light emitted from the LED can be controlled. The variable source and the LED form the input circuit of the optocoupler or optoisolator.

A photodiode is present in front of the LED so that the light from the LED directly strikes the junction of the photodiode. The photodiode is in reverse biased condition. The reverse biased circuit of the photodiode forms the output circuit of the system. It is also

ensured that there is no other light falling on the photodiode junction and the system is protected from any external light, except the light coming from the LED. Initially, no voltage is applied to the LED; hence the LED does not glow. In this condition as no light falls on the photodiode, there would be only dark current flowing through the output circuit. Dark current is the reverse saturation current of the reverse biased photodiode when it entire dark. This is the unavoidable reverse leakage current of the diode. Now, if we increase the voltage across the LED, the LED starts glowing and at same time intensity of the light increases with increasing input voltage across the LED. With increasing light intensity, the reverse current in the photodiode increases, since the reverse current in a photodiode is linearly proportional to the intensity of light falling on the photodiode junction. Also, if we decrease the intensity of light in the input, the output photodiode current will decrease.

2.4.2 Applications of Optoisolator

Optocouplers or Optoisolators are used in

- Lamp Ballasts
- Light Dimmers
- Valve or Motor Controllers
- Microcontrollers for interfacing with High Voltage Circuits.

2.5 Light Dependent Resistor

A light dependent resistor also known as a LDR, photo resistor, photoconductor or photocell, is a resistor whose resistance increases or decreases depending on the amount of light intensity. LDRs (Light Dependent Resistors) are a very useful tool in a light/dark circuits. For example, it can be used to turn on a light when the LDR is in darkness or to turn off a light when the LDR is in light. It can also work the other way around so when the LDR is in light it turns on the circuit and when it's in darkness the resistance increases and disrupts the circuit.

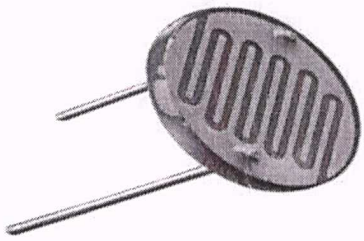


Fig. 2.7: LDR

2.5.1 Working principle of LDR:

This resistor works on the principle of photo conductivity. It is nothing but, when the light falls on its surface, then the material conductivity reduces and also the electrons in the valance band 21 of the device are excited in the conduction band. These photons in the incident light must have energy greater than the band gap of the semi-conductor material. This makes the electrons to jump from the valence band to conduction.

2.5.2 Working principle of LDR:

This resistor works on the principle of photo conductivity. It is nothing but, when the light falls on its surface, then the material conductivity reduces and also the electrons in the valance band 21 of the device are excited in the conduction band. These photons in the incident light must have energy greater than the band gap of the semi-conductor material. This makes the electrons to jump from the valence band to conduction.

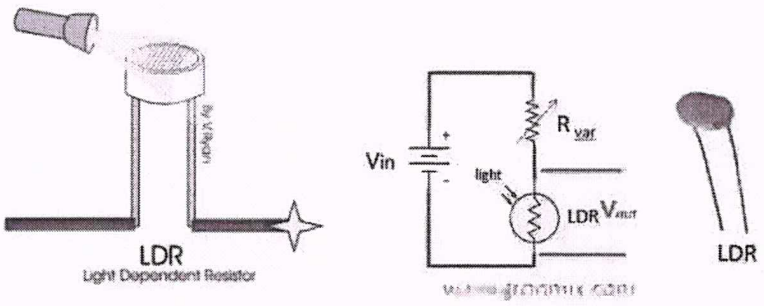


Fig 2.8: Working Of LDR

These devices depend on the light ,when light falls on the LDR then the resistance decreases, and increases in the dark .when LDR is kept in the dark place, its resistance is high and ,when the LDR is kept in light its resistance will decrease.

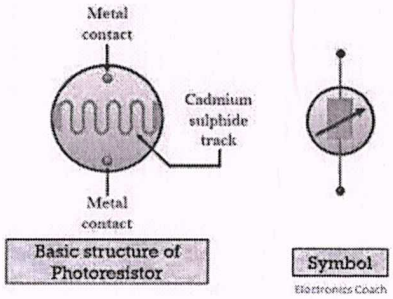


Fig. 2.9: Structure of LDR

2.6 ESP8266

2.6.1 Description:

The ESP8266 Wi-Fi Module is a self-contained SOC with integrated TCP/IP protocol stack that can give any microcontroller access to your Wi-Fi network. The ESP8266 is capable of either hosting an application or offloading all Wi-Fi networking functions from another application processor. Each ESP8266 module comes pre-programmed with an AT command set firmware, meaning, you can simply hook this up to your Arduino device and get about as much Wi-Fi-ability as a Wi-Fi Shield offers (and that's just out of the box)! The ESP8266 module is an extremely cost-effective board with a huge, and ever growing, community.

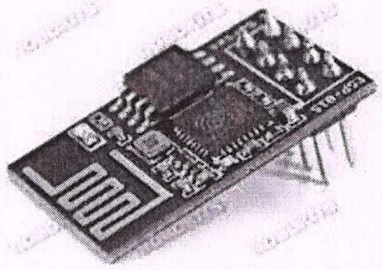


Fig. 2.10: ESP8266

This module has a powerful enough on-board processing and storage capability that allows it to be integrated with the sensors and other application specific devices through its GPIOs with minimal development up-front and minimal loading during runtime. Its high degree of on-chip integration allows for minimal external circuitry, including the front-end module, is designed to occupy minimal PCB area. The ESP8266 supports APSD for VoIP applications and Bluetooth co-existence interfaces, it contains a self-calibrated RF allowing it to work under all operating conditions, and requires no external RF parts. There is an almost limitless fountain of information available for the ESP8266, all of which has been provided by amazing community support. In the Documents section below you will find many resources to aid you in using the ESP8266, even instructions on how to transforming this module into an IoT (Internet of Things) solution!

2.6.2 Features:

- 802.11 b/g/n
- Wi-Fi Direct (P2P), soft-AP
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- Integrated PLLs, regulators, DCXO and power management units
- +19.5dBm output power in 802.11b mode
- Power down leakage current of
- Integrated low power 32-bit CPU could be used as application processor
- SDIO 1.1 / 2.0, SPI, UART
- STBC, 1×1 MIMO, 2×1 MIMO
- A-MPDU & A-MSDU aggregation & 0.4ms guard interval
- Wake up and transmit packets in < 2ms
- Standby power consumption of < 1.0mW (DTIM3)

2.6.3 Specification of ESP 8266:

- Wi-Fi Direct (P2P), soft-AP
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network

- Integrated PLLs, regulators, DCXO and power management units
- 19.5dBm output power in 802.11b mode
- Power down leakage current of
- Standby power consumption of < 1.0mW (DTIM3)

2.7 2 Channel 5v 10a Relay Module:

2.7.1 Description:

The relay module is an electrically operated switch that allows you to turn on or off a circuit using voltage and/or current much higher than a microcontroller could handle. There is no connection between the low voltage circuit operated by the microcontroller and the high-power circuit. The relay protects each circuit from each other. Each channel in the module has three connections named NC, COM, and NO. Depending on the input signal trigger mode, the jumper cap can be placed at highlevel effective mode which 'closes' the normally open (NO) switch at high level input and at low level effective mode which operates the same but at low level input.

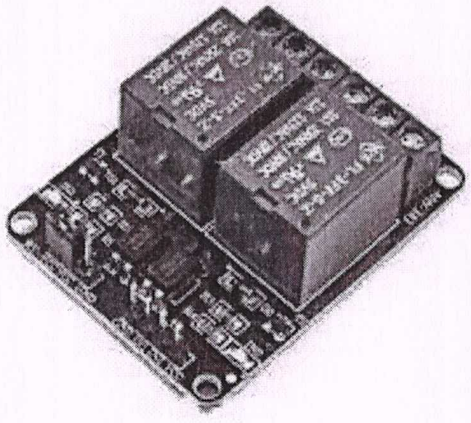


Fig. 2.11: 2 Channel 5v 10a Relay Module

2.7.2 Specifications:

- On-board EL817 photoelectric coupler with photoelectric isolating ant interference ability strong
- On-board 5V, 10A / 250VAC, 10A / 30VDC relays
- Relay long life can absorb 100000 times in a row
- Module can be directly and MCU I/O link, with the output signal indicator
- Module with diode current protection, short response time
- PCB Size: 45.8mm x 32.4mm

2.7.3 Pin Configuration:

- VCC: 5V DC
- COM: 5V DC
- IN1: high/low output
- IN2: high/low output
- GND: ground

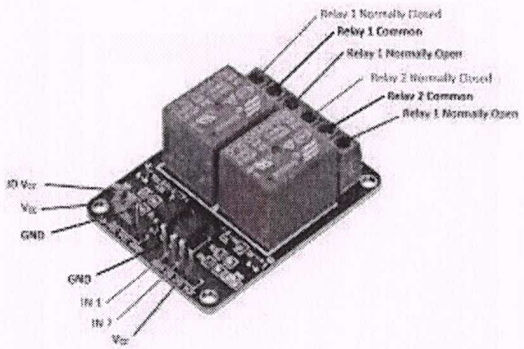


Fig. 2.12: Pin configuration of 2 Channel 5V 10A Relay Module

2.8 DS18B20:

(1-Wire Digital Temperature Sensor for Use on Soil)

2.8.1 Description:

The DS18B20 is one type of temperature sensor and it supplies 9-bit to 12-bit readings of temperature. These values show the temperature of a particular device. The

communication of this sensor can be done through a one-wire bus protocol which uses one data line to communicate with an inner microprocessor. Additionally, this sensor gets the power supply directly from the data line so that the need for an external power supply can be eliminated. The applications of the DS18B20 temperature sensor include industrial systems, consumer products, systems which are sensitive thermally, thermostatic controls, and thermometers.

2.8.2 DS18B20 Pin Configuration:

- Pin1 (Ground): This pin is used to connect to the GND terminal of the circuit
- Pin2 (VCC): This pin is used to give the power to the sensor which ranges from 3.3V or 5V
- Pin3 (Data): The data pin supplies the temperature value, which can communicate with the help of 1-wire method.

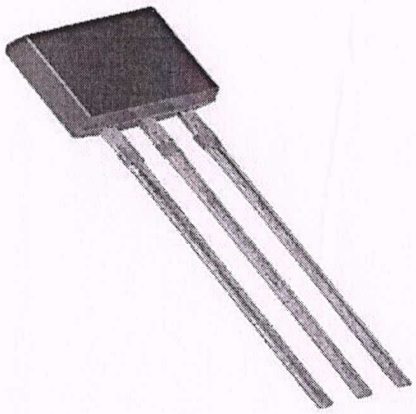


Fig. 2.13: DS18B20 Pin Configuration

2.8.3 Specifications:

- The specifications of this sensor include the following.
- This sensor is a programmable and digital temperature sensor
- The communication of this sensor can be done with the help of a 1-Wire method
- The range of power supply is 3.0V – 5.5V
- Fahrenheit equal s to -67°F to +257°F

- The accuracy of this sensor is ±0.5°C
- The o/p resolution will range from 9-bit to 12-bit
- It changes the 12-bit temperature to digital word within 750 ms time
- This sensor can be power-driven from the data line
- Alarm options are programmable
- The multiplexing can be enabled by Unique 64-bit address
- The temperature can be calculated from -55°C to +125°C.
- These are obtainable like SOP, To-92, and also as a waterproof sensor

2.8.4 Working Principle

The working principle of this DS18B20 temperature sensor is like a temperature sensor. The resolution of this sensor ranges from 9-bits to 12-bits. But the default resolution which is used to power-up is 12-bit. This sensor gets power within a low-power inactive condition. The temperature measurement, as well as the conversion of A-to-D, can be done with a convert-T command. The resulting temperature information can be stored within the 2-byte register in the sensor, and after that, this sensor returns to its inactive state.

If the sensor is power-driven by an exterior power supply, then the master can provide read time slots next to the Convert T command. The sensor will react by supplying 0 though the temperature change is in the improvement and reacts by supplying 1 though the temperature change is done.

2.8.5 Applications:

- The applications of DS18B20 include the following.
- This sensor is extensively used to calculate temperature within rigid environments which includes mines, chemical solutions, otherwise soil, etc.
- This sensor is used to measure the liquid temperature.
- We can use it in the thermostat controls system.
- It can be used in industries as a temperature measuring device.
- This sensor is used as a thermometer.

- It can be used in devices like which are sensitive to thermal.
- These are used in HVAC systems.
- Applications where the temperature has to be measured at multiple points.

2.9 Soil Moisture Sensor(YL-69):

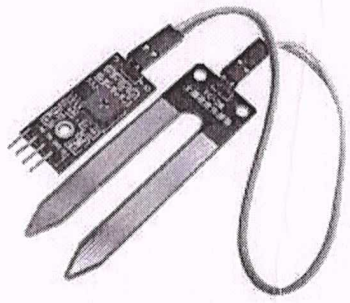


Fig. 2.14: Soil Moisture Sensor

2.9.1 Working of Soil moisture sensor

Let's now see how to build this simple project the two large exposed pads function as probes for the sensor the more water in the soil the better the conductivity between the pads that results in a lower resistance the sensor is an analogue one so in the analog output. we get about that as the soil gets drier, we get more voltage at the analog output since the resistance between the probes gets higher, we can set a threshold in order to enable the digital output at a certain moisture level using this potentiometer given with electronics part of sensor. But in this tutorial, I am using only the analog output of the sensor module.

2.9.2 Pin out of Soil moisture sensor

The pin out soil moisture sensor is given below. The connection to pic microcontroller extremely easy as follows:

- we connect the ground of the sensor to ground pin of power supply.
- VCC pin of sensor to 5v of power supply voltage.

- Next, we connect the analog output of the sensor to analog pin 0 of the pic microcontroller.

2.9.2 Applications:

- Thermostatic Controls
- Industrial Systems
- Consumer Products
- Thermometers
- Thermally Sensitive Systems

2.10 Arduino UNO

2.10.1 Description:

Arduino Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins, 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It also supports 3 communication protocols named Serial, I2C and SPI protocol.

2.10.1 Main features of Arduino UNO:

No.	Parameter Name	Parameter Value
1.	Microcontroller	Atmega328
2.	Crystal Oscillator	16 MHz
3.	Operating Voltage	5 V
4.	Input Voltage	5 - 12V
5.	Digital I/O Pins	14 (D0 - D15)
6.	Analog I/O Pins	6 (A0 – A5)

7.	PWM Pins	6 (Pin # 3, 5, 6, 9, 10, and 11)
8.	Power Pins	5V, 3.3V, Vin, GND
9.	Communication	UART (1), SPI (1), I2C (1)
10.	Flash Memory	32KB (0.5KB is used by bootloader)
11.	SRM	2 KB
12.	EEPROM	1 KB
13.	ICSP Header	Yes
14.	Power Sources	DC Power Jack and USB Port

Table 2.2: Features of Arduino

2.10.3 Arduino Communication

- Apart from USB, a battery or AC to DC adopter can also be used to power the board.
- Arduino UNO comes with a USB interface i.e.; USB port is added on the board to develop serial communication with the computer.
- Atmega328 microcontroller is placed on the board that comes with a number of features like timers, counters, interrupts, PWM, CPU, I/O pins and based on a 16MHz clock that helps in producing more frequency and number of instructions per cycle.

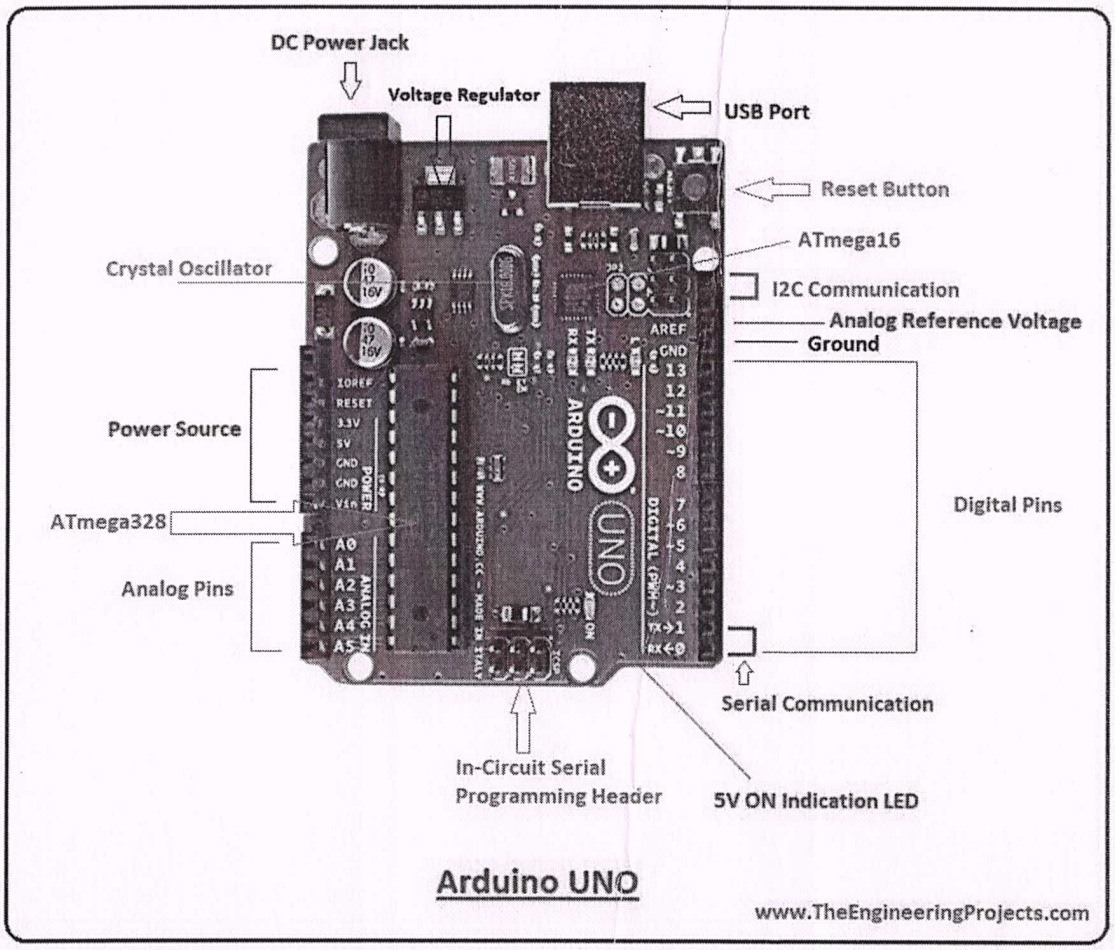


Fig.2.15: Arduino UNO

2.10.4 Atmega Microcontroller

- It is an open-source platform where anyone can modify and optimize the board based on the number of instructions and tasks they want to achieve.
- This board comes with a built-in regulation feature that keeps the voltage under control when the device is connected to the external device.
- A reset pin is present in the board that resets the whole board and takes the running program in the initial stage. This pin is useful when the board hangs up in the middle of the running program; pushing this pin will clear everything up in the program and starts the program right from the beginning.

- The 6 analog pins are marked as A0 to A5 and come with a resolution of 10bits. These pins measure from 0 to 5V, however, they can be configured to the high-range using analogReference() function and AREF pin.
- Only 5 V is required to turn the board on, which can be achieved directly using a USB port or external adopter, however, it can support an external power source up to 12 V which can be regulated and limit to 5 V or 3.3 V based on the requirement of the project.

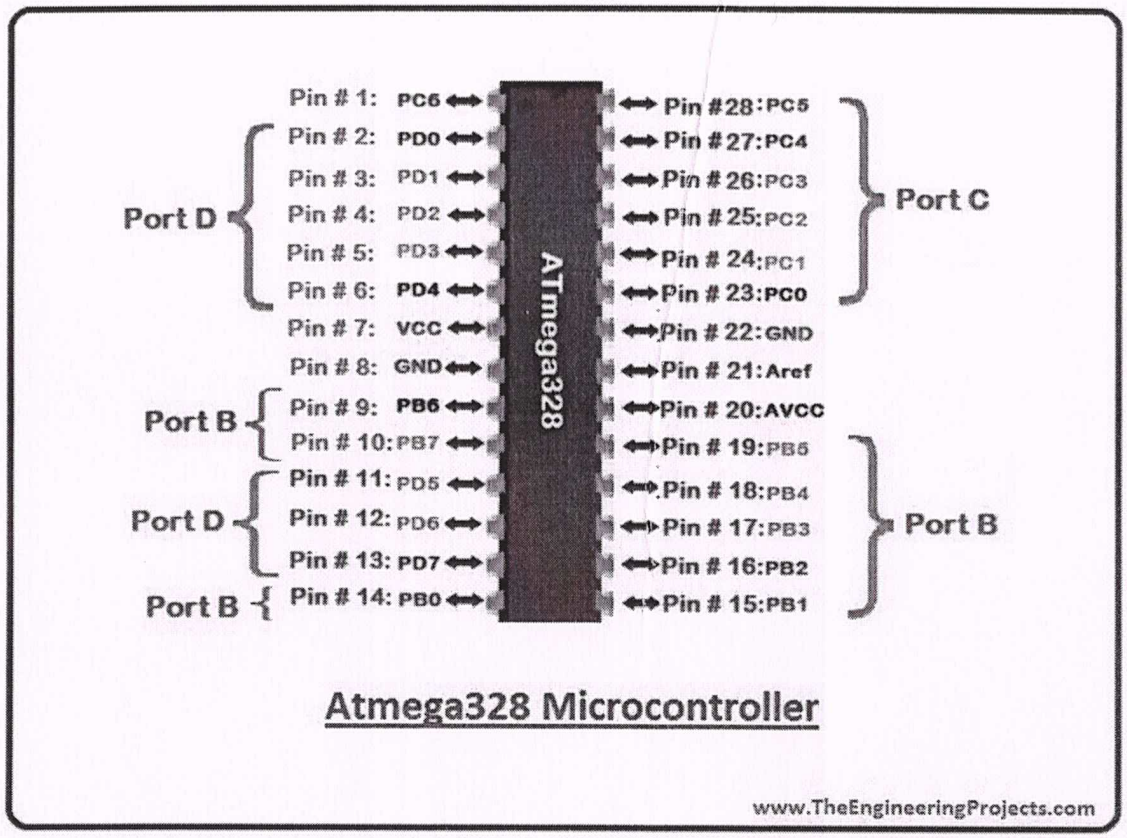


Fig. 2.16: Atmega328 Microcontroller

2.10.5 Arduino Uno Pinout

Arduino Uno is based on an AVR microcontroller called Atmega328. This controller comes with 2KB SRAM, 32KB of flash memory, 1KB of EEPROM. Arduino

Board comes with 14 digital pins and 6 analog pins. ON-chip ADC is used to sample these pins. A 16 MHz frequency crystal oscillator is equipped on the board. The following figure shows the pinout of the Arduino Uno Board.

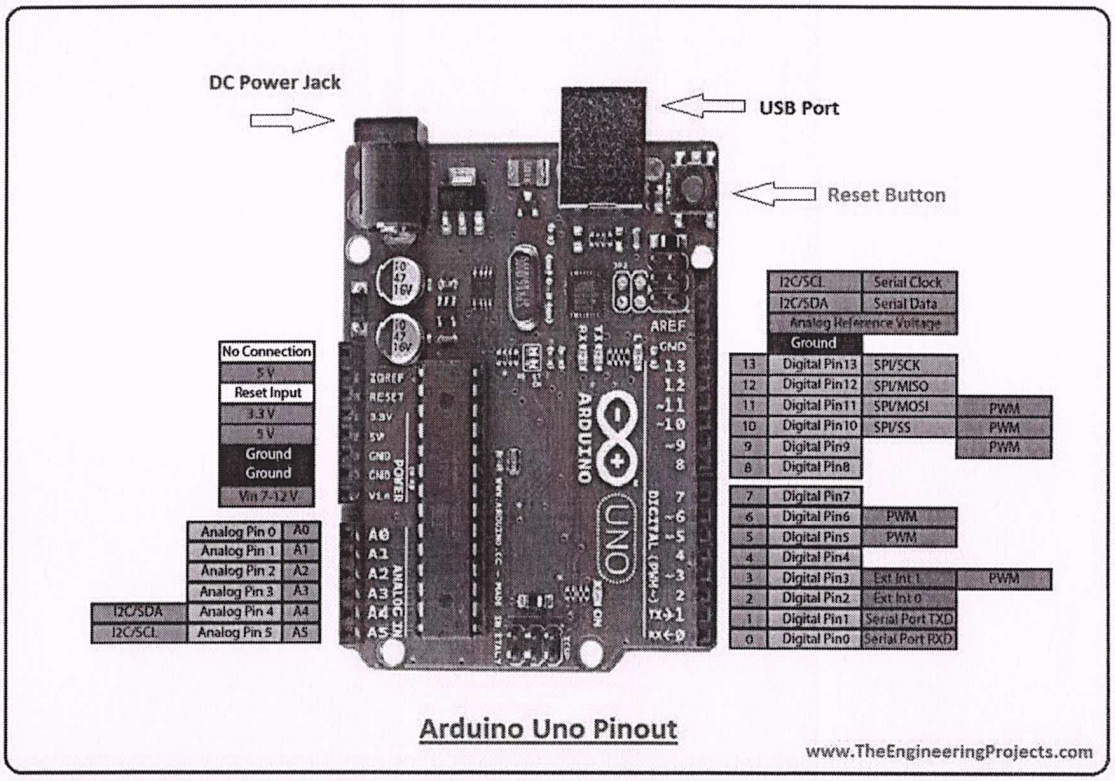


Fig. 2.17: Arduino UNO Pinout

There are several I/O digital and analog pins placed on the board which operates at 5V. These pins come with standard operating ratings ranging between 20mA to 40mA. Internal pull-up resistors are used in the board that limits the current exceeding the given operating conditions. However, too much increase in current makes these resisters useless and damages the device.

- **LED:** Arduino Uno comes with a built-in LED which is connected through pin 13. Providing HIGH value to the pin will turn it ON and LOW will turn it OFF.

- **Vin:** It is the input voltage provided to the Arduino Board. It is different than 5 V supplied through a USB port. This pin is used to supply voltage. If a voltage is provided through a power jack, it can be accessed through this pin.
- **5V:** This board comes with the ability to provide voltage regulation. 5V pin is used to provide output regulated voltage. The board is powered up using three ways i.e., USB, Vin pin of the board or DC power jack.
- USB supports voltage around 5V while Vin and Power Jack support a voltage range between 7V to 20V. It is recommended to operate the board on 5V. It is important to note that, if a voltage is supplied through 5V or 3.3V pins, they result in bypassing the voltage regulator that can damage the board if the voltage surpasses its limit.
- **GND:** These are ground pins. More than one ground pins are provided on the board which can be used as per requirement.
- **Reset:** This pin is incorporated on the board which resets the program running on the board. Instead of physical reset on the board, IDE comes with a feature of resetting the board through programming.
- **IOREF:** This pin is very useful for providing voltage reference to the board. A shield is used to read the voltage across this pin which then selects the proper power source.
- **PWM:** PWM is provided by 3,5,6,9,10, 11pins. These pins are configured to provided 8-bit output PWM.
- **SPI:** It is known as Serial Peripheral Interface. Four pins 10(SS), 11(MOSI), 12(MISO), 13(SCK) provide SPI communication with the help of the SPI library.
- **AREF:** It is called Analog Reference. This pin is used for providing a reference voltage to the analog inputs.
- **TWI:** It is called Two-wire Interface. TWI communication is accessed through Wire Library. A4 and A5 pins are used for this purpose.
- **Serial Communication:** Serial communication is carried out through two pins called Pin 0 (Rx) and Pin 1 (Tx).
- Rx pin is used to receive data while Tx pin is used to transmit data.

- **External Interrupts:** Pin 2 and 3 are used for providing external interrupts. An interrupt is called by providing LOW or changing value.

Arduino Uno comes with the ability of interfacing with other Arduino boards, microcontrollers and computers. The Atmega328 placed on the board provides serial communication using pins like Rx and Tx.

The Atmega16U2 incorporated on the board provides a pathway for serial communication using USB com drivers. A serial monitor is provided on the IDE software which is used to send or receive text data from the board. If LEDs placed on the Rx and Tx pins will flash, they indicate the transmission of data.

Arduino Uno is programmed using Arduino Software which is a cross-platform application called IDE written in Java. The AVR microcontroller Atmega328 laid out on the base comes with built-in bootloader that sets you free from using a separate burner to upload the program on the board.

2.10.6 Arduino Uno Applications:

- Embedded System
- Security and Defence System
- Digital Electronics and Robotics
- Parking Lot Counter
- Weighing Machines
- Traffic Light Count Down Timer
- Medical Instrument
- Emergency Light for Railways
- Home Automation
- Industrial Automation

Chapter 3: WORKING OF PROJECT

3.1 Objective:

Our goal will be to basically collect information from a localunit and send it to the internet. A user anywhere on the planet looking at this information will make decisions by sending remote commands to the actuators, which will also be in this local unit. Any sensor or actuator could be used.

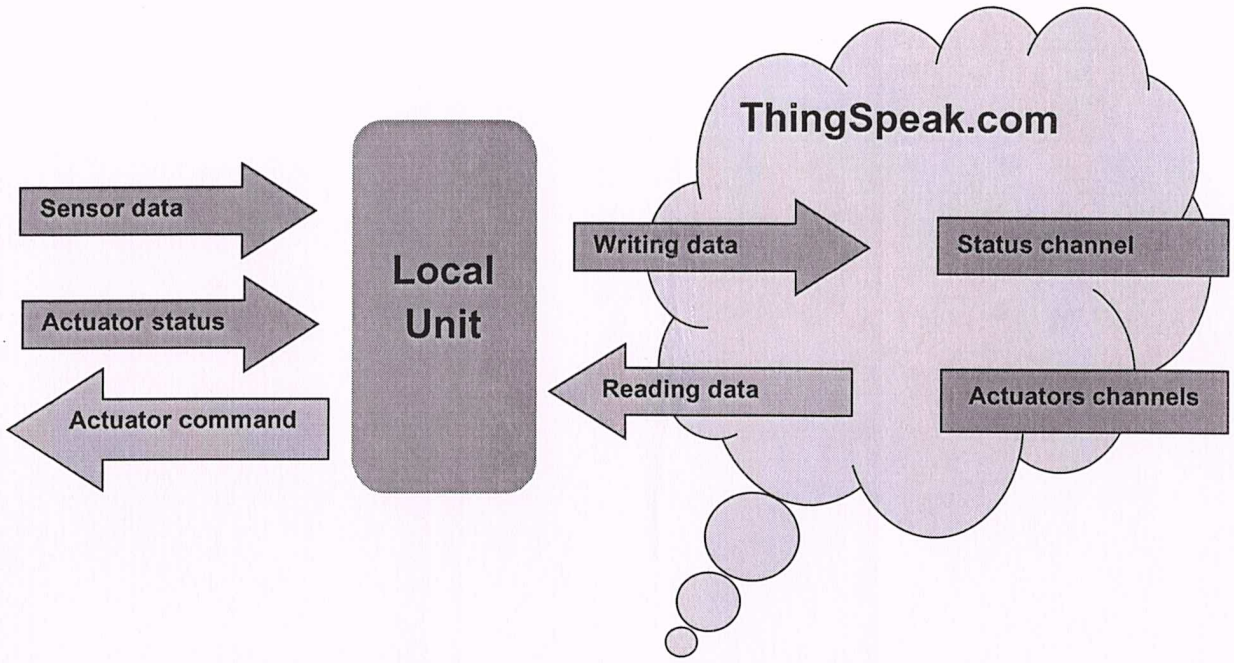


Fig. 3.1: Block Diagram of Overall System

The "Centre of our IoT project" will be the ThingSpeak.com. The local unit (UNO/ESP-01) will capture data from sensors and actuator status; send them to the Internet, "writing" on a specific ThingSpeak.com Status Channel. The local unit will also receive data from the internet, "reading" them from specific ThingSpeak Actuator Channels.

(See the above Block diagram to better understand the flow of data).

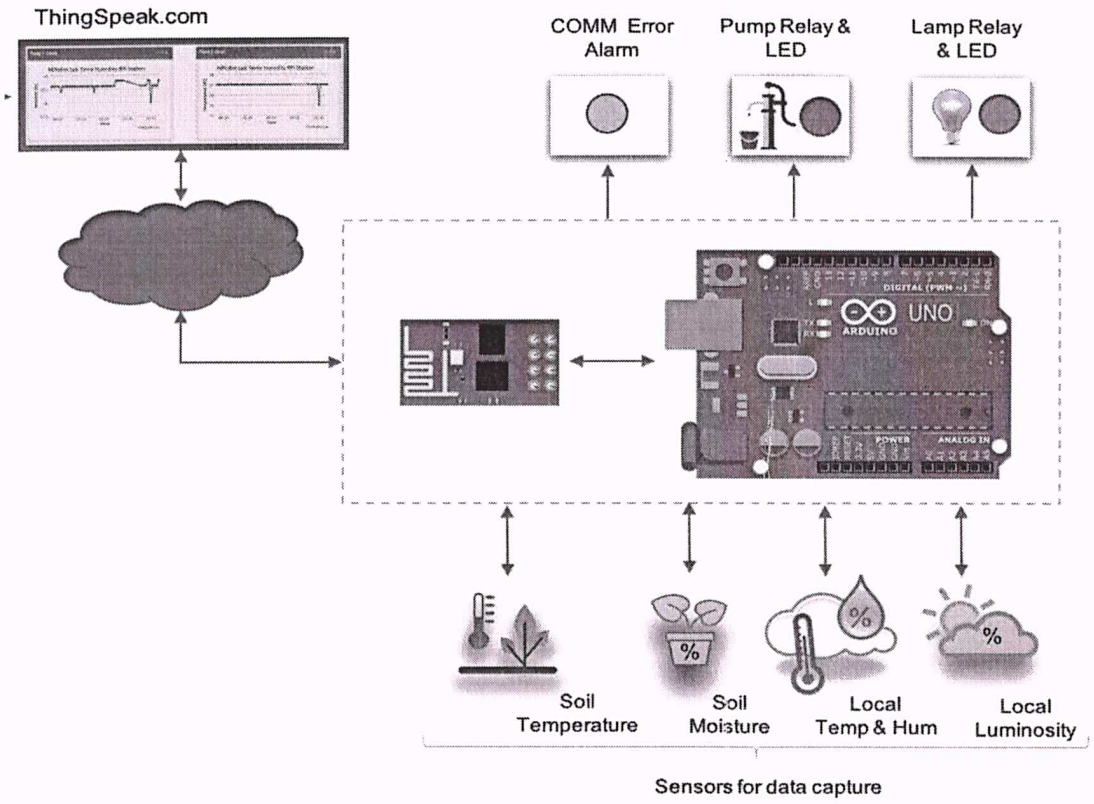


Fig. 3.2: A Proposed model

The Above Fig 3.2 Shows the proposed model for ThingSpeak based Sensing and Monitoring system where the model exhibit all the sensors and internet cloud, ThingSpeak cloud are interfaced with one another using the Arduino UNO.
Using common sensors, project will capture several data, sending them to the cloud, where everyone can see them thru the internet. To work those data, we will use the service provided by ThingSpeak.com, an open IoT platform that will permit us to collect, analyse and act on those data.

The data to be collected by sensors will be:

- Air Temperature and relative humidity
- Soil Temperature and humidity
- Luminosity

The project will have 2 actuators:

- Electrical Pump
- Electrical Lamp

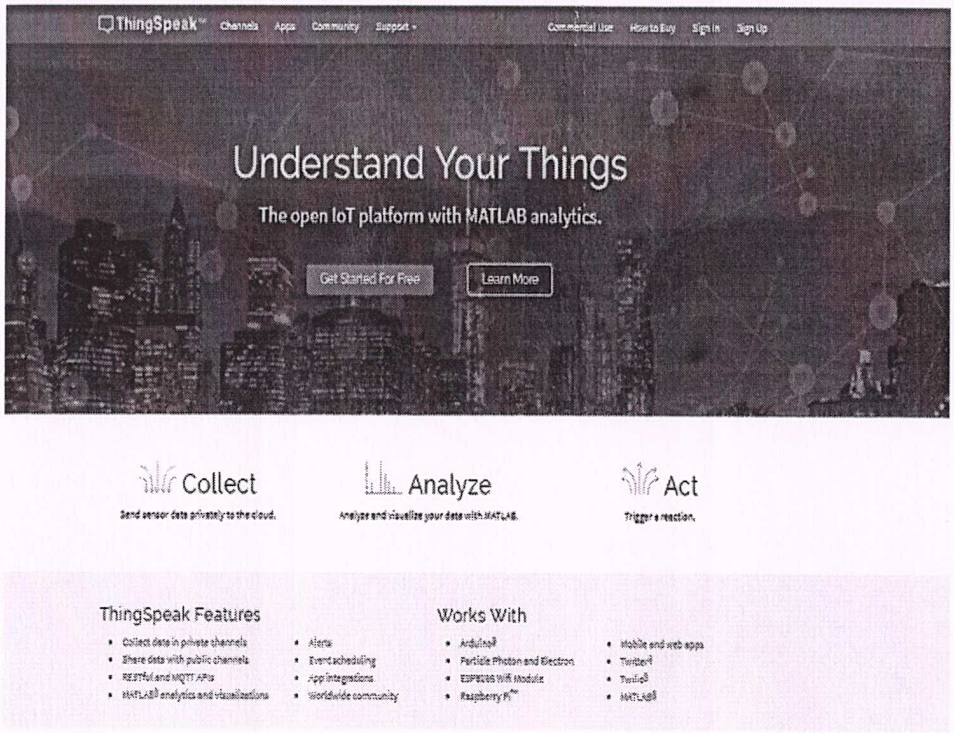
The status of those actuators ("ON/OFF"), should be also sent to the cloud.

Chapter 4: THINGSPEAK: A IoT WEB SERVICE

4.1 Description:

ThingSpeak provides a very good tool for IoT based projects. By using the ThingSpeak site, we can monitor our data and control our system over the Internet, using the Channels and web pages provided by ThingSpeak.

A IoT web Service ThingSpeak is a web based open API IoT source information platform that comprehensive in storing the sensor data of varied 'IoT applications' and conspire the sensed data output in graphical form at the web level.
One of the most important parts of our project is the ThingSpeak, an open IoT platform that will permit us to collect, analyse and act on collected data.
For creating your channel on ThingSpeak, you first need to Sign up on ThingSpeak. In case if you already have an account on ThingSpeak, just sign in using your id and password.
For creating your account go to www.thinspeak.com



Then create a new channel and create the **API keys**.

- This key is required for programming modifications and setting your data.
- To send data to ThingSpeak, we need a unique API key, which we will use later in our code to upload our sensor data to ThingSpeak Website.
- The below Fig 11 highlights the unique ThingSpeak channel ID and Read API key which is used in this project.

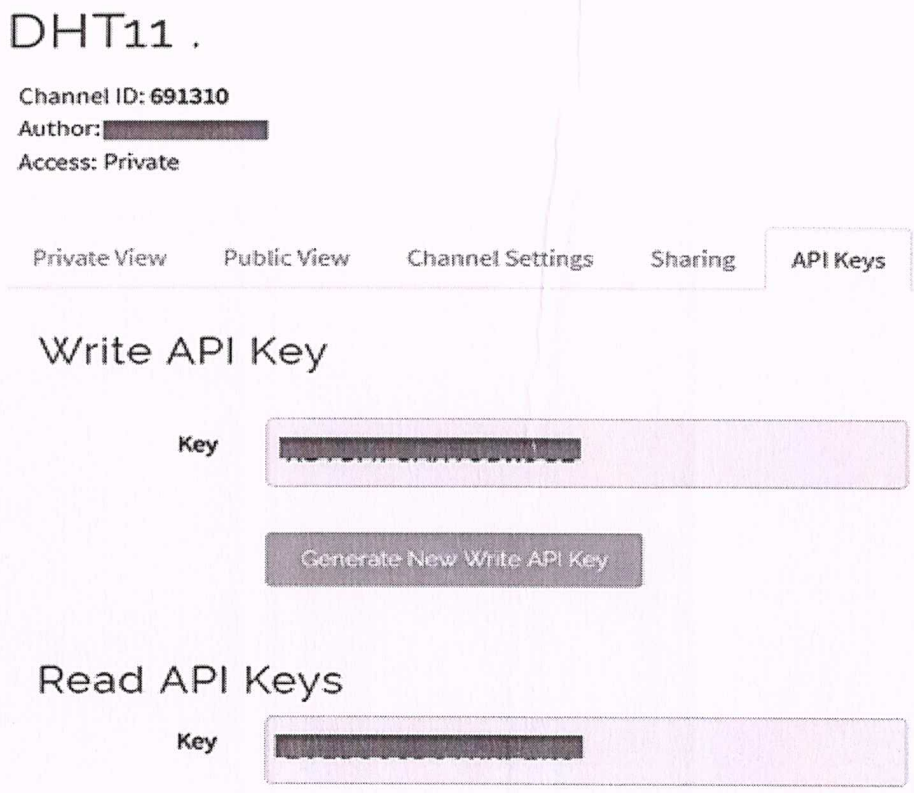


Fig. 4.1: Channel ID and API Key

Chapter 5: HARDWARE

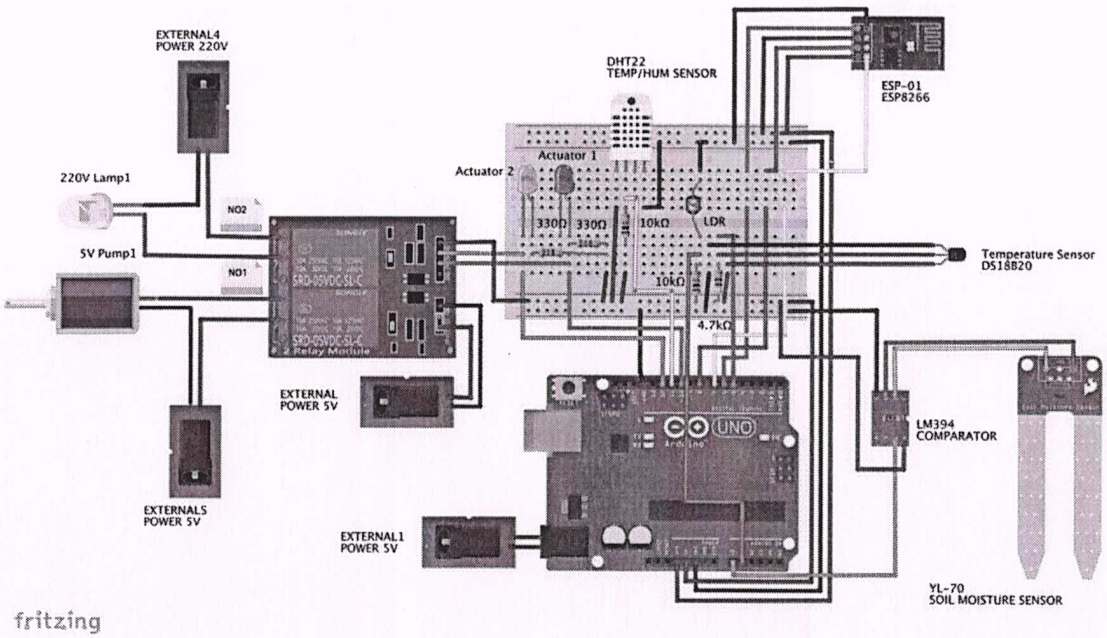


Fig. 5.1 : A Hardware Setup

We can follow the steps:

- Install and test locally all sensors
- Install and configure the ESP-01 (Bare Minimum)
- Change the ESP-01 installation for its final configuration and test it
- Configure the ThingSpeak Status Channel
- Install ThingSpeak code in your Arduino and check the Sensors status on the Cloud.

5.1 Connecting Sensors:

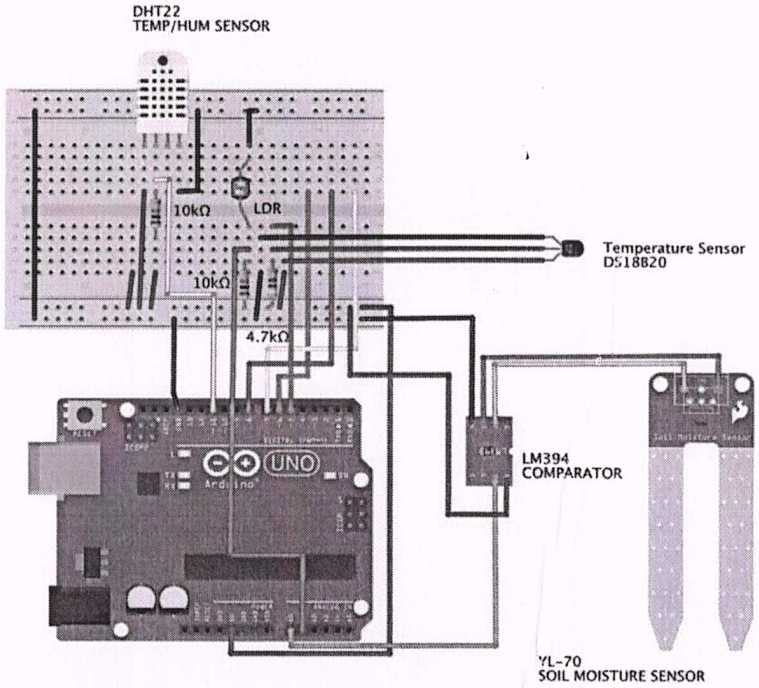


Fig. 5.2: Connection of Sensors with Arduino Uno

```
// DS18B20
#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS 5 // DS18B20 on pin D5
OneWireoneWire(ONE_WIRE_BUS);
DallasTemperature DS18B20(&oneWire);
int soilTemp = 0;

//DHT
#include "DHT.h"
#include <stdlib.h>
int pinoDHT = 11;
int tipoDHT =  DHT22;
DHT dht(pinoDHT, tipoDHT);
int airTemp = 0;
int airHum = 0;

// LDR (Light)
```

```
#defineldrPIN 1
int light = 0;

// Soil humidity
#definesoilHumPIN 0
int soilHum = 0;

void setup()
{
Serial.begin(9600);
  DS18B20.begin();
dht.begin();
}

void loop()
{
readSensors();
displaySensors();
  delay (10000);
}
```

```
/********* Read Sensors value *************/
void readSensors(void)
{
airTemp = dht.readTemperature();
airHum = dht.readHumidity();

  DS18B20.requestTemperatures();
soilTemp = DS18B20.getTempCByIndex(0);
soilHum = map(analogRead(soilHumPIN), 1023, 0, 0, 100);

  light = map(analogRead(ldrPIN), 1023, 0, 0, 100);
}

/********* Display Sensors value *************/
void displaySensors(void)
{
Serial.print ("airTemp  (oC): ");
Serial.println (airTemp);
Serial.print ("airHum    (%): ");
Serial.println (airHum);
Serial.print ("soilTemp (oC): ");
```

```
Serial.println (soilTemp);
Serial.print ("soilHum   (%): ");
Serial.println (soilHum);
Serial.print ("light     (%): ");
Serial.println (light);
Serial.println ("");
}
```

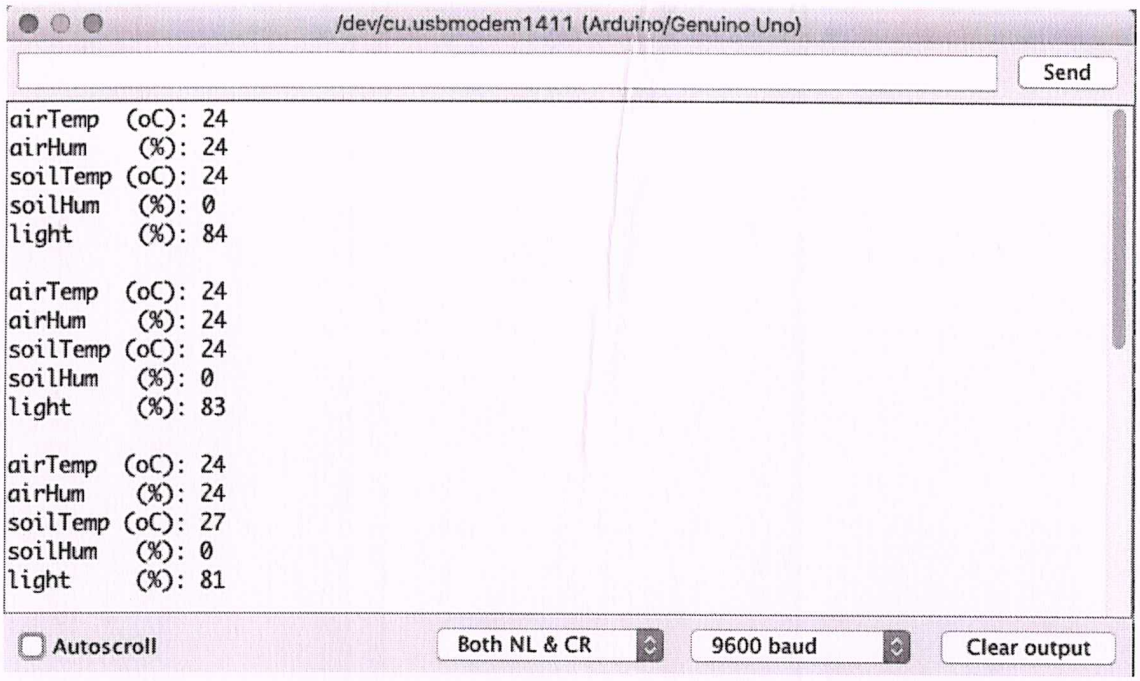


Fig: 5. Output of all sensors (on Serial Monitor)

5.2 Connecting Sensors and ESP-01:

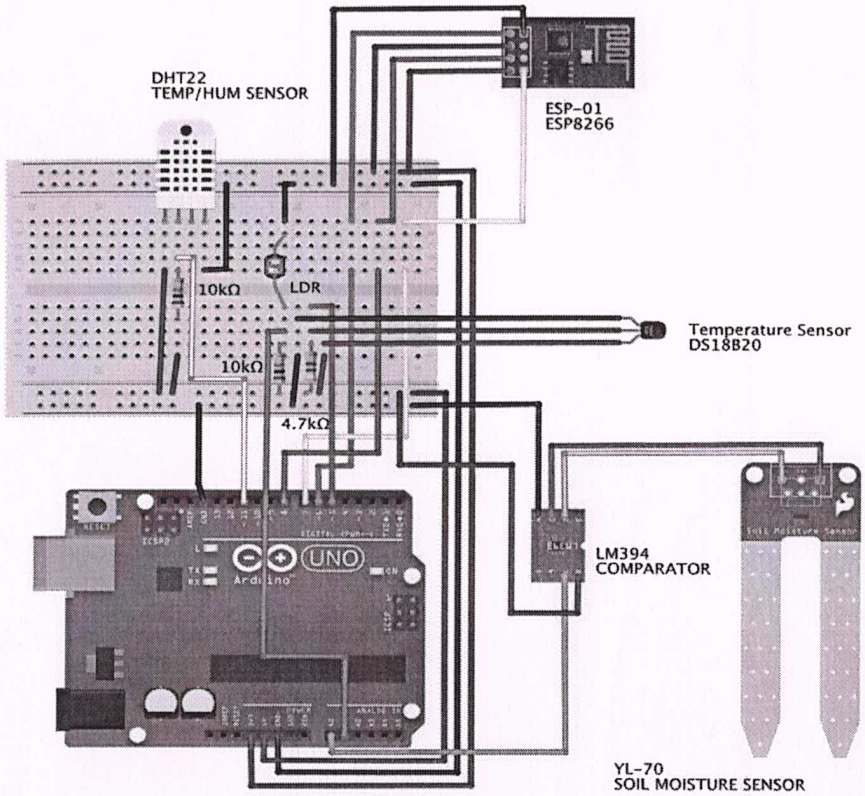


Fig. 5.3:Connecting Sensors and ESP-01

```
#include <SoftwareSerial.h>
SoftwareSerial esp8266(6,7);  //Rx ==> Pin 6; TX ==> Pin7
#define speed8266 9600

void setup()
{
  esp8266.begin (speed8266);
Serial.begin(speed8266);
Serial.println("ESP8266 Setup test - use AT coomands");
}
```

```
void loop()
{
  while(esp8266.available())
  {
Serial.write(esp8266.read());
  }
  while(Serial.available())
  {
    esp8266.write(Serial.read());
  }
}
```

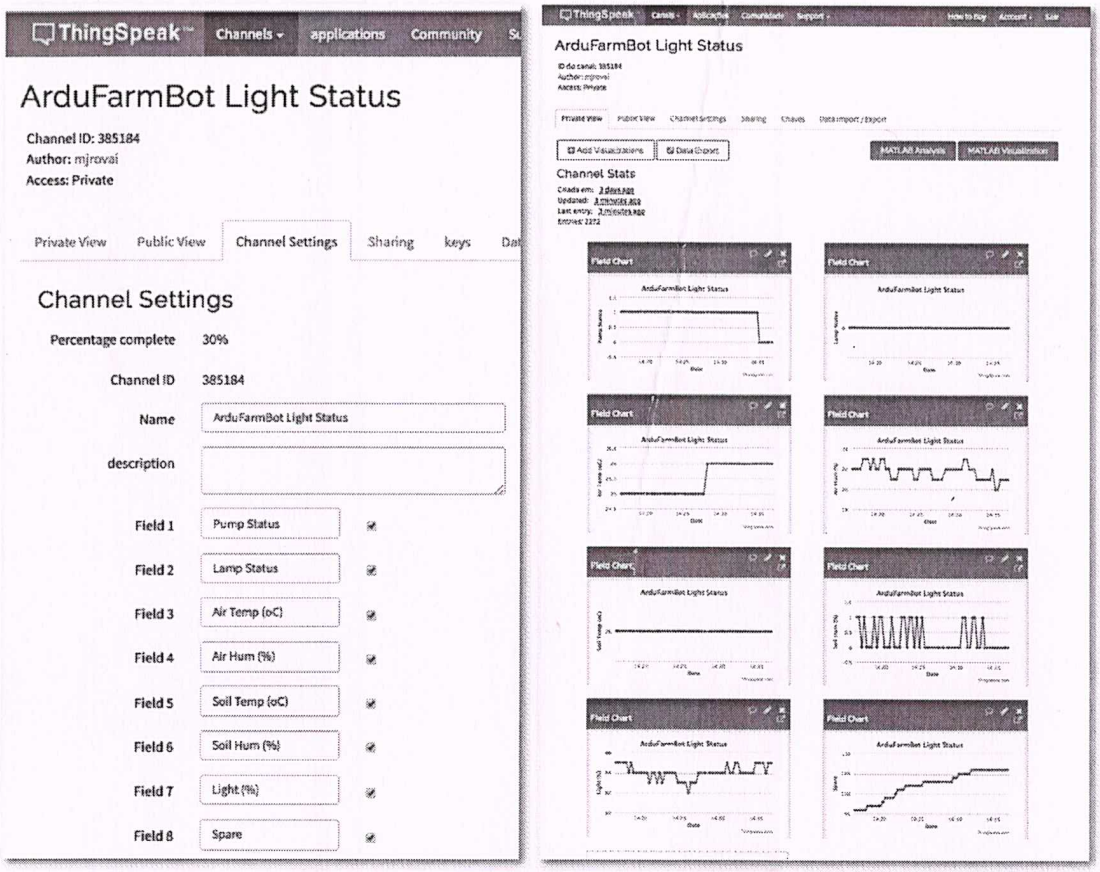


Fig. 5.4: Sending Status to the Cloud

One of the most important parts of our project is the ThingSpeak, an open IoT platform that will permit us to collect, analyse and act on collected data. If you do not have yet, please go to ThingSpeak sign up and create your account.
Next, create a new Channel where we will have our 2 actuators, 5 sensors and a spare field status:

- Field 1: Actuator 1
- Field 2: Actuator 2
- Field 3: Air Temperature in oC
- Filed 4: Air Relative Humidity in %
- Field 5: Soil Temperature in oC
- Field 6: Soil Humidity in %
- Field 7: Luminosity in %

```
// ThingSpeak
String statusChWriteKey = "YOUR WRITE KEY HERE";  // Status Channel
id: 385184

#include <SoftwareSerial.h>
SoftwareSerialEspSerial(6, 7); // Rx,  Tx
#define HARDWARE_RESET 8

// DS18B20
#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS 5 // DS18B20 on pin D5
OneWireoneWire(ONE_WIRE_BUS);
DallasTemperature DS18B20(&oneWire);
int soilTemp = 0;

//DHT
#include "DHT.h"
#include <stdlib.h>
int pinoDHT = 11;
int tipoDHT =  DHT22;
DHT dht(pinoDHT, tipoDHT);
```

```
int airTemp = 0;
int airHum = 0;

// LDR (Light)
#defineldrPIN 1
int light = 0;

// Soil humidity
#definesoilHumPIN 0
int soilHum = 0;

// Variables to be used with timers
long writeTimingSeconds = 17; // ==> Define Sample time in seconds to
send data
long startWriteTiming = 0;
long elapsedWriteTime = 0;

// Variables to be used with Actuators
boolean pump = 0;
boolean lamp = 0;

int spare = 0;
boolean error;

void setup()
{
Serial.begin(9600);

pinMode(HARDWARE_RESET,OUTPUT);
digitalWrite(HARDWARE_RESET, HIGH);
  DS18B20.begin();
dht.begin();

EspSerial.begin(9600); // Comunicacao com Modulo WiFi
EspHardwareReset(); //Reset do Modulo WiFi
startWriteTiming = millis();
}

void loop()
{
  start: //label
  error=0;
elapsedWriteTime = millis()-startWriteTiming;
  if (elapsedWriteTime> (writeTimingSeconds*1000))
  {
readSensors();
```

```
writeThingSpeak();
startWriteTiming = millis();
  }
  if (error==1) //Resend if transmission is not completed
  {
Serial.println(" <<<< ERROR >>>>");
    delay (2000);
goto start; //go to label "start"
  }
}
```

```
/********* Read Sensors value *************/
void readSensors(void)
{
airTemp = dht.readTemperature();
airHum = dht.readHumidity();

  DS18B20.requestTemperatures();
soilTemp = DS18B20.getTempCByIndex(0);
  light = map(analogRead(ldrPIN), 1023, 0, 0, 100);
soilHum = map(analogRead(soilHumPIN), 1023, 0, 0, 100);
}

/********* Conexao com TCP com ThingSpeak *******/
void writeThingSpeak(void)
{
startThingSpeakCmd();

  String getStr = "GET /update?api_key=";
getStr += statusChWriteKey;
getStr +="&field1=";
getStr += String(pump);
getStr +="&field2=";
getStr += String(lamp);
getStr +="&field3=";
getStr += String(airTemp);
getStr +="&field4=";
getStr += String(airHum);
getStr +="&field5=";
getStr += String(soilTemp);
```

```
getStr +="&field6=";
getStr += String(soilHum);
getStr +="&field7=";
getStr += String(light);
getStr +="&field8=";
getStr += String(spare);
getStr += "\r\n\r\n";

sendThingSpeakGetCmd(getStr);
}
```

```
/********* Reset ESP *************/
void EspHardwareReset(void)
{
Serial.println("Reseting.......");
digitalWrite(HARDWARE_RESET, LOW);
  delay(500);
digitalWrite(HARDWARE_RESET, HIGH);
  delay(8000);//Tempo necessário para começar a ler
Serial.println("RESET");
}

/********* Start communication with ThingSpeak*************/
void startThingSpeakCmd(void)
{
EspSerial.flush();
  String cmd = "AT+CIPSTART=\"TCP\",\"";
  cmd += "184.106.153.149";
  cmd += "\",80";
EspSerial.println(cmd);
Serial.print("enviado ==> Start cmd: ");
Serial.println(cmd);

  if(EspSerial.find("Error"))
  {
Serial.println("AT+CIPSTART error");
return;
}
}
```

```
/********* send a GET cmd to ThingSpeak ************/
String sendThingSpeakGetCmd(String getStr)
{
  String cmd = "AT+CIPSEND=";
  cmd += String(getStr.length());
EspSerial.println(cmd);
Serial.print("enviado ==>lenght cmd: ");
Serial.println(cmd);

  if(EspSerial.find((char *)">"))
  {
EspSerial.print(getStr);
Serial.print("enviado ==>getStr: ");
Serial.println(getStr);
    delay(500

    String messageBody = "";
    while (EspSerial.available())
    {
      String line = EspSerial.readStringUntil('\n');
      if (line.length() == 1)
messageBody = EspSerial.readStringUntil('\n');
    }
Serial.print("MessageBody received: ");
Serial.println(messageBody);
    return messageBody;}
  else
  {
EspSerial.println("AT+CIPCLOSE");     // alert user
Serial.println("ESP8266 CIPSEND ERROR: RESENDING"); //Resend...
    spare = spare + 1;
    error=1;
return "error";
}
 }
```

5.3 Installing Actuators (LEDs and Relays)

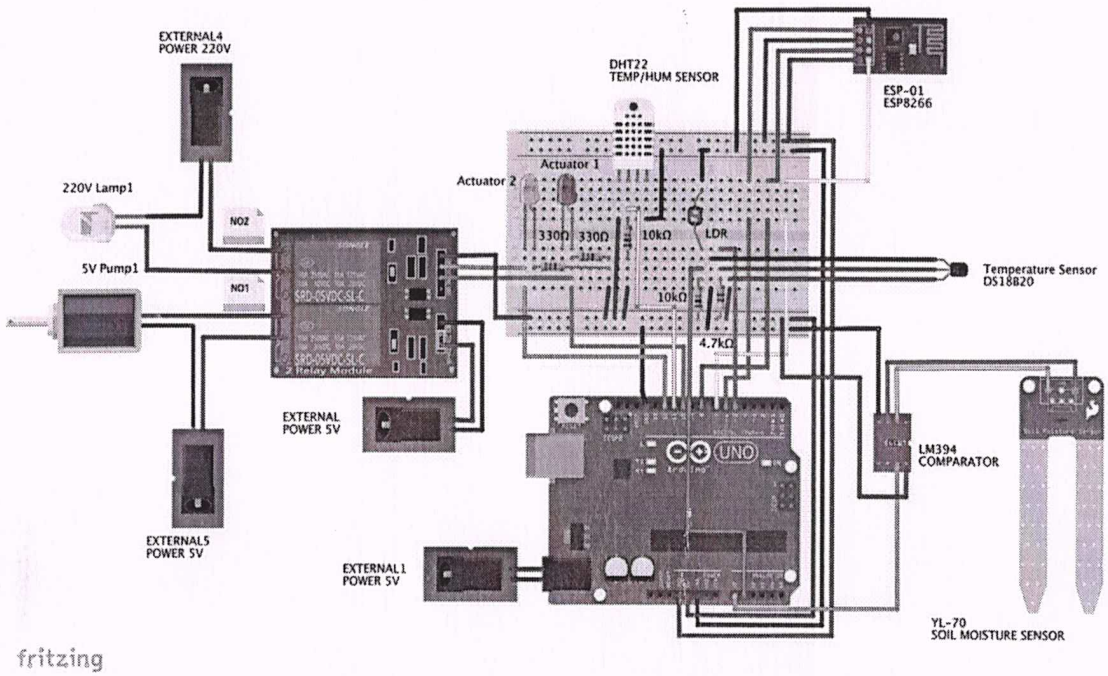


Fig. 5.5: Installing Actuators

To Turn ON and OFF a Pump and a Lamp. The Arduino output will activate a Relay (and a LED) to get those actions.
We will use a Relay module that has an Optocoupler Low-Level Trigger. Also, we will supply 5V to it through a separate pin.

The above figure shows how the actuators must be connected. Note that the Realy GND is NOT CONNECTED to Arduino GND. This will help not to introduce noise when the relay works.

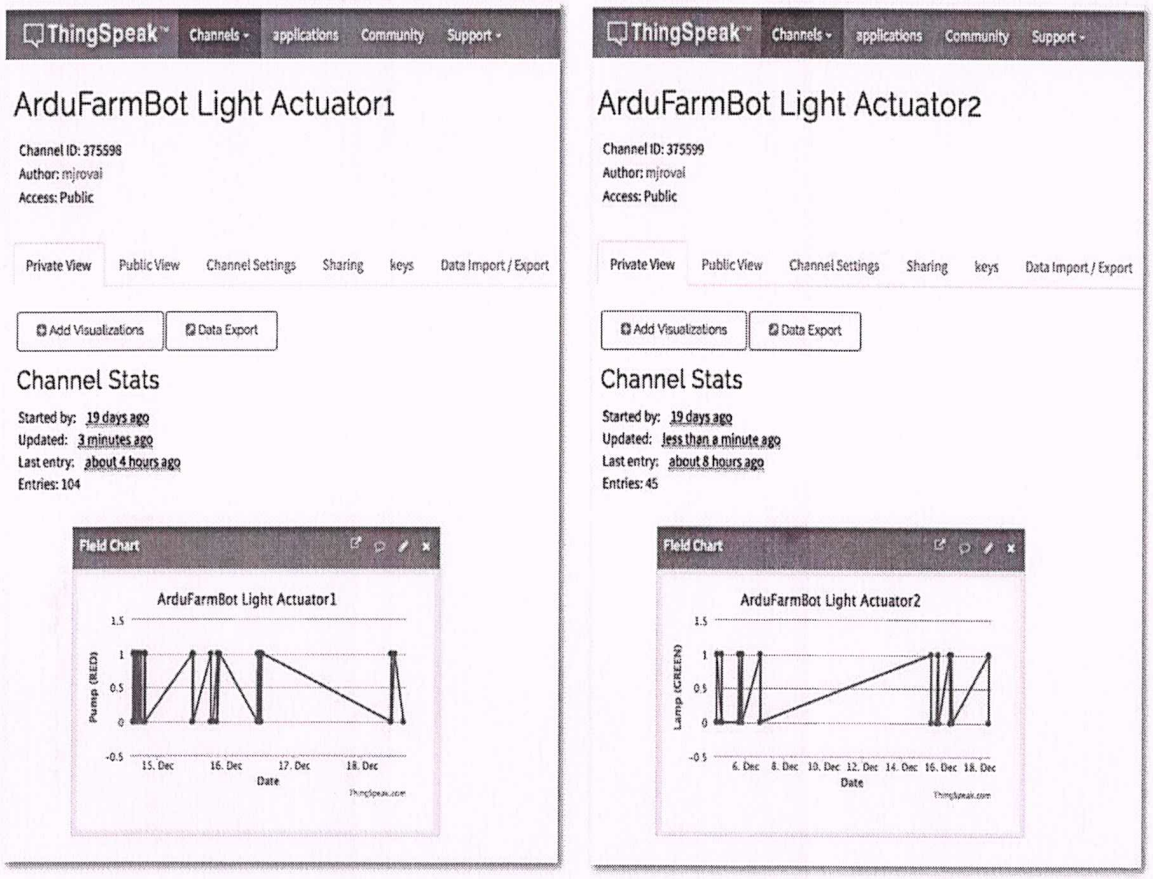


Fig. 5.6: Configuring the ThingSpeak Actuators Channels

The same way that we did for the Status, we will create 2 new channels one for each actuator.

From each one of them, take note of Channel ID, Read and Write keys.

We will write only on Field 1 of each one of those channels.

For example:

- Channel ID 375598 ==> LED Red (Pump)
    - Field1 = 0 ==> Pump OFF
    - Field1 = 1 ==> Pump ON

- Channel ID 375599 ==> LED Green (Lamp)
    - Field1 = 0 ==> Lamp OFF
    - Field1 = 1 ==> Lamp ON

When we sent data to the web, what we did was to WRITE on a ThingSpeak channel (Channel status) to "transmit" (upload) data.

Now we should READ from a ThingSpeak channel (Actuator Channel) to "receive" (download) data. We will READ from a ThingSpeak channel and for that, we will need to send a GET string. We will do in 3 parts:

We will send a "Start cmd":

```
AT+CIPSTART="TCP","184.106.153.149",80
```

Following the "length" of the command:

```
AT+CIPSEND=36
```

And the GET string itself, that will read from field 1 on each one of the Actuator Channel:

```
GET /channels/375598/fields/1/last
```

We will be reading from ThingSpeak channels on intervals of 10 seconds.

Chapter 6: FUTURE SCOPE

An Android App will be also be use for **reading** those data from the ThingSpeak.com Status Channel and displaying them for the user. Same way, the user, based on that status info, can send commands to actuators, **writing** them on ThingSpeak Actuator Channels .

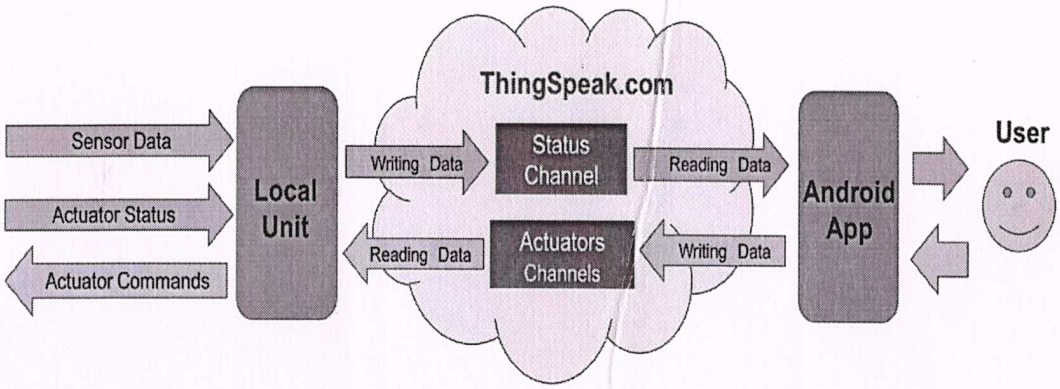


Fig. 6.1: Future Scope

For example, a plantation and send them to cloud. Based on those data, a user should take the decision based on those statements:

- Turn ON the Pump if the soil humidity is too low
- Turn ON the Lamp if the soil temperature is too low

To remotely command our actuators, we will use an Android App.

References

1.INTRODUCTION TO ARDUINO:-https://www.arduino.cc/en/guide/introduction

2.INTRODUCTION TO ESP8266:- https://microcontrollerslab.com/esp8266-getting-started-tutorial/

3.ARDUINO IDE:- https://en.m.wikipedia.org/wiki/Arduino_IDE

4. DHT22:- https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf

5. 5VDC SUBMERSIBLE PUMP :- https://5.imimg.com/data5/IQ/GJ/PF/SELLER-1833510/dc-mini-submersible-water-pump.pdf

6. LDR :- https://components101.com/sites/default/files/component_datasheet/LDR%20Datasheet.pdf

7. SOIL MOISTURE SENSOR :- https://www.electronicoscaldas.com/datasheet/OBSoil-01_ElecFreaks.pdf

8. 2 CHANNEL 5V 10A RELAY MODULE:- https://www.robodukkan.com/class/INNOVAEditor/assets/2_CHANNEL_5V_10A_RELAY_MODULE.pdf

9. INTRODUCTION TO THINKSPEAK :- https://www.codeproject.com/Articles/845538/An-Introduction-to-ThingSpeak

10.https://www.google.com/

11. https://www.wikipedia.org/