# CHAPTER-1

# ABSTRACT

## 1.1 ABSTRACT:

The prime goal is to develop a reliable automated agriculture system so that the farmers can easily handle the cultivation. In this work we present an IOT based smart irrigation system that can provide real time online information on their PC about crop condition and they can take necessary action from their devices through IOT.

Internet of Things (IOT) is a concept that envisions all objects around us as part of internet. IOT coverage is very wide and includes variety of objects like smart phones, tablets, pc and sensors. Once all these devices are connected to each other, they enable more and more smart processes and services that support our basic needs, environment and health. Such enormous number of devices connected to internet provides many kinds of services. They also produce huge amount of data and information.

The monitoring of crop condition is really helpful in various applications like in critical conditions like soil drying, heavy moisture in the crop. The need for this project came from the support of a fact of very low cost devices and instruments are available that can provide you live updates of the crop and accordingly take actions from anywhere.

# 1.2 LITERATURE SURVEY:

The Internet of Things is a novel paradigm shift in IT arena. The phrase “Internet of Things” which is also shortly well-known as IOT is coined from the two words i.e. the first word is “Internet” and the second word is “Things”. The Internet is a global system of interconnected computer networks that use the standard Internet protocol suite (TCP/IP) to serve billions of users worldwide. It is a network of networks that consists of millions of private, public, academic, business, and government networks, of local to global scope, that are linked by a broad array of electronic, wireless and optical networking technologies.

In this paper, soil moisture sensor, temperature sensors placed in root zone of plant and gateway unit handles the sensor information and transmit data to a web application. One algorithm was developed for measure threshold values of temperature sensor and soil moisture sensor that was programmed into a microcontroller to control water quantity. For power photovoltaic panel was used. Another facto like cellular-Internet interface used that allowed for data inspection and irrigation scheduling to be programmed through a web page.

- GSM Based Automated Irrigation Control using Rain gun Irrigation System.R.suresh , S.Gopinath , K.Govindaraju , T.Devika , N.SuthanthiraVanitha.
- Irrigation Control System Using Android and GSM for Efficient Use of Water and Power - LaxmiShabadi, NandiniPatil, Nikita. M, Shruti. J, Smitha. P & Swati.
- Remote Sensing and Control of an Irrigation System Using a Distributed Wireless Sensor Network Yunseop (James) Kim, Member, IEEE, Robert G. Evans, and William M. Iversen.

Automatic irrigation system control using Zigbee and GPRS technique has some disadvantage viz speed, distance factor, reliability, so GPRS is not used in our project. Zigbee also have disadvantage i.e. low transmission rate. It is only use for smaller distance.

# CHAPTER-2

# INTRODUCTION

## 2.1 OBJECTIVE:

India’s major source of income is from agriculture sector and 70% of farmers and general people depend on the agriculture. In India most of the irrigation systems are operated manually. These outmoded techniques are replaced with semi-automated and automated techniques. The available traditional techniques are like ditch irrigation, terraced irrigation, drip irrigation, sprinkler system. The global irrigation scenario is categorized by increased demand for higher agricultural productivity, poor performance and decreased availability of water for agriculture. These problems can be appropriately rectified if we use automated irrigation system in agriculture using IOT.

Internet of Things (IOT) is a concept that envisions all objects around us as part of internet. IOT coverage is very wide and includes variety of objects like smart phones, tablets, pc and sensors. Once all these devices are connected to each other, they enable more and more smart processes and services that support our basic needs, environment and health. Such enormous number of devices connected to internet provides many kinds of services. They also produce huge amount of data and information.

## 2.2 PRINCIPLE OF OPERATION:

The system mainly consists of sensors and microcontroller for data acquisition. The device works by taking readings from various sensors at different pins in raspberry pi microcontroller and it will also give outputs to sprinklers. For this purpose we’ve used an raspberry pi compatible WiFi shield stacked upon our raspberry pi microcontroller which adds up extra functionality to our board. It increases the scope of this project.

The receiver unit also has a duplex communication link based on a Internet interface. The Internet connection allows the data inspection in real time on a website, where the soil-moisture and temperature levels are graphically displayed through an application interface and farmers can control the sprinklers through the application interface website.

## 2.3 INTRODUCTION TO IOT:

The Internet of Things, also called The Internet of Objects, refers to a wireless network between objects, usually the network will be wireless and self-configuring, such as household appliances.

The Internet of Things (IOT) is the network of physical objects or "things" embedded with electronics, software, sensors, and network connectivity, which enables these objects to collect and exchange data. IOT allows objects to be sensed and controlled remotely across existing network infrastructure, creating opportunities for more direct integration between the physical world and computer-based systems, and resulting in improved efficiency, accuracy and economic benefit.

### 2.3.1 IOT EVOLUTION:

The term “Internet of Things” (IOT) was first used in 1999 by British technology pioneer Kevin Ashton to describe a system in which objects in the physical world could be connected to the Internet by sensors.12 Ashton coined the term to illustrate the power of connecting Radio-Frequency Identification (RFID) tags13 used in corporate supply chains to the Internet in order to count and track goods without the need for human intervention. Today, the Internet of Things has become a popular term for describing scenarios in which Internet connectivity and computing capability extend to a variety of objects, devices, sensors, and everyday items. While the term “Internet of Things” is relatively new, the concept of combining computers and networks to monitor and control devices has been around for decades. By the late 1970s, for example, systems for remotely monitoring meters on the electrical grid via telephone lines were already in commercial use.14 In the 1990s, advances in wireless technology allowed “machine–to–machine” (M2M) enterprise and industrial solutions for equipment monitoring and operation to become widespread. Many of these early M2M solutions, however, were based on closed purpose–built networks and proprietary or industry–specific standards,15 rather than on Internet Protocol (IP)–based networks and Internet standards. Using IP to connect devices other than computers to the Internet is not a new idea. The first Internet “device”—an IP–enabled toaster that could be turned on and off over the Internet—was featured at an Internet conference in 1990.16 Over the next several years, other “things” were IP–enabled, including a soda machine17 at Carnegie Mellon University in the US and a coffee pot18 in the Trojan Room at the University of Cambridge in the UK (which remained Internet–connected until 2001). From these whimsical beginnings, a robust field of research and development into “smart object networking”19 helped create the foundation for today’s Internet of Things.

## 2.3.2 IOT TECHNOLOGY OVERVIEW:

A thing, in the Internet of Things, can be a person with a heart monitor implant, a farm animal with a biochip transponder, an automobile that has built-in sensors to alert the driver when tire pressure is low -- or any other natural or man-made object that can be assigned an IP address and provided with the ability to transfer data over a network.

IOT architecture layers can be divided into four layers, they are.

- Sense and Identification Layer
- Network Construction Layer
- Information Processing Layer
- Integrated Application Layer

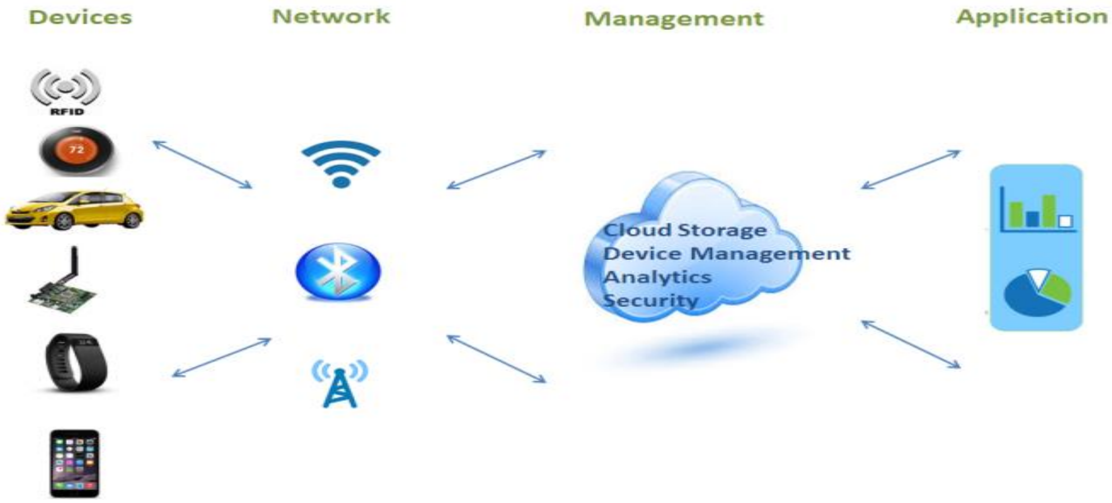


**Fig2.1 Architecture layer of IOT**

# CHAPTER-3

# NODEMCU ESP8266-12E

## 3.1 INTRODUCTION:

NodeMCU is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.

ESP-12E WiFi module is developed by Ai-thinker Team. core processor ESP8266 in smaller sizes of the module encapsulates Tensilica L106 integrates industry-leading ultra low power 32-bit MCU micro, with the 16-bit short mode, Clock speed support 80 MHz, 160 MHz, supports the RTOS, integrated Wi-Fi MAC/BB/RF/PA/LNA, on-board antenna. The module supports standard IEEE802.11 b/g/n agreement, complete TCP/IP protocol stack. Users can use the add modules to an existing device networking, or building a separate network controller. ESP8266 is high integration wireless SOCs, designed for space and power constrained mobile platform designers. It provides unsurpassed ability to embed Wi-Fi capabilities within other systems, or to function as a standalone application, with the lowest cost, and minimal space requirement.

ESP8266EX offers a complete and self-contained Wi-Fi networking solution; it can be used to host the application or to offload Wi-Fi networking functions from another application processor. When ESP8266EX hosts the application, it boots up directly from an external flash. In has integrated cache to improve the performance of the system in such applications.

ESP8266EX also integrates an enhanced version of Tensilica’s L106 Diamond series 32-bit processor, with on-chip SRAM, besides the Wi-Fi functionalities. ESP8266EX is often integrated with external sensors and other application specific devices through its GPIOs

## BLOCK DIAGRAM OF ESP8266:

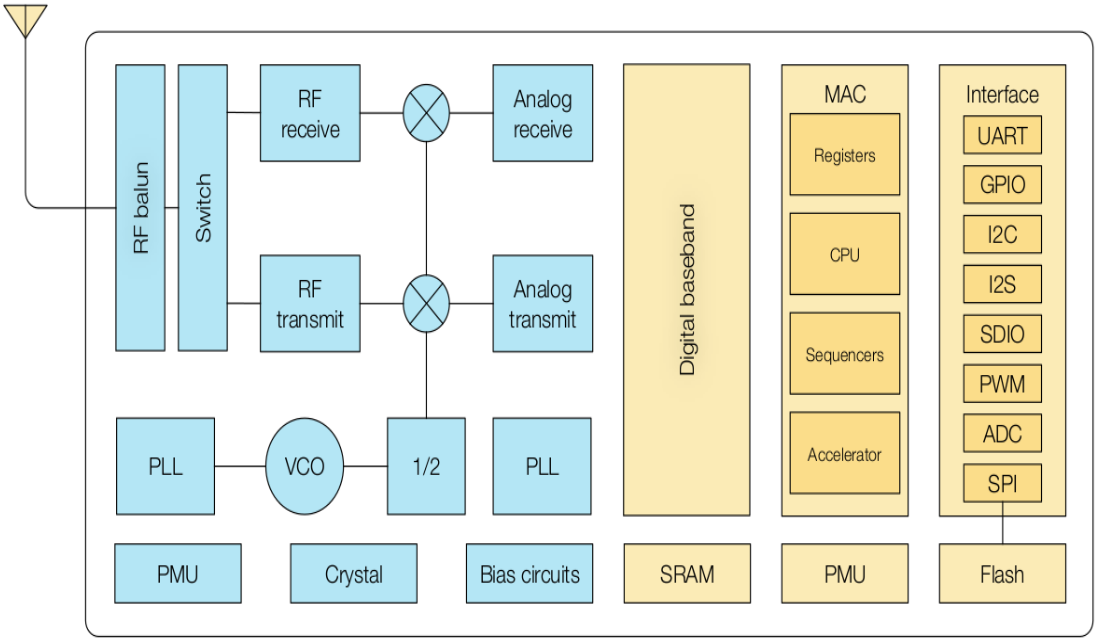


**Fig 3.1: Block Diagram of ESP8266**

## 3.2 FEATURES:

- 802.11 b/g/n
- Integrated low power 32-bit MCU
- Integrated 10-bit ADC
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- Integrated PLL, regulators, and power management units
- Supports antenna diversity
- Wi-Fi 2.4 GHz, support WPA/WPA2
- Support STA/AP/STA+AP operation modes
- Support Smart Link Function for both Android and iOS devices
- Support Smart Link Function for both Android and iOS devices
- SDIO 2.0, (H) SPI, UART, I2C, I2S, IRDA, PWM, GPIO
- STBC, 1x1 MIMO, 2x1 MIMO
- A-MPDU & A-MSDU aggregation and 0.4s guard interval

- Deep sleep power <10uA, Power down leakage current < 5uA
- Wake up and transmit packets in < 2ms
- Standby power consumption of < 1.0mW (DTIM3)
- +20dBm output power in 802.11b mode
- Operating temperature range -40C ~ 125C

## ESP 12E PIN DESIGN:

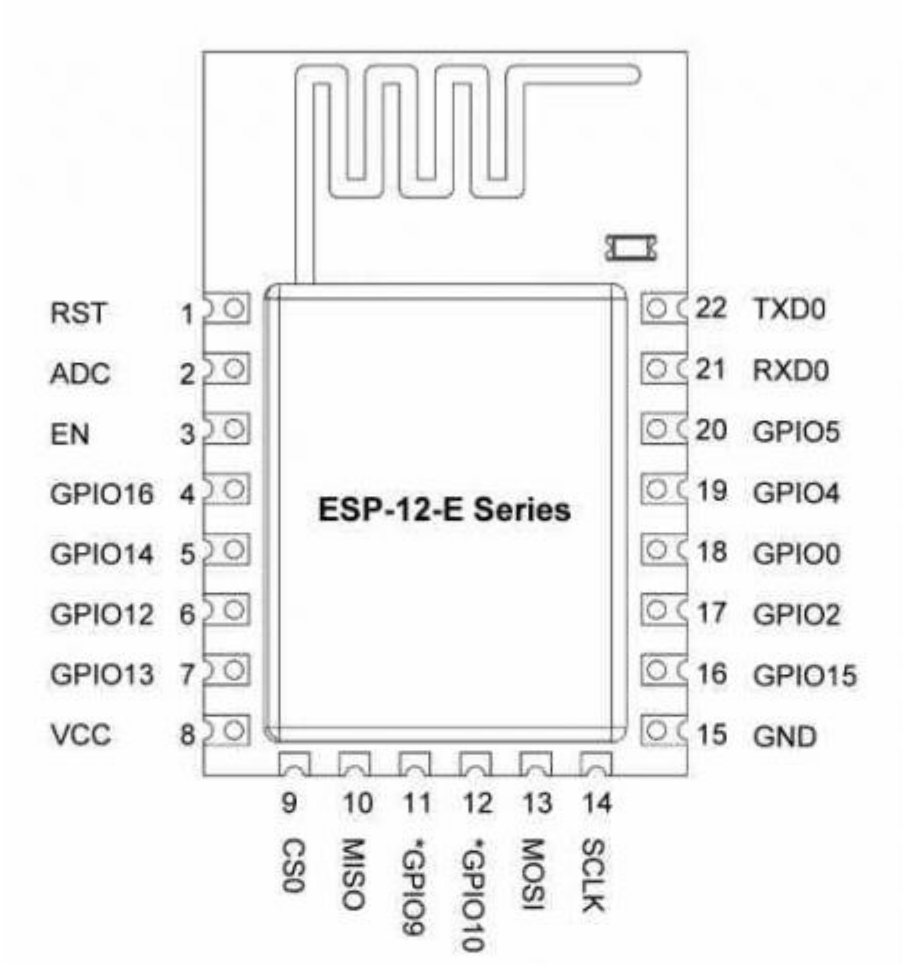
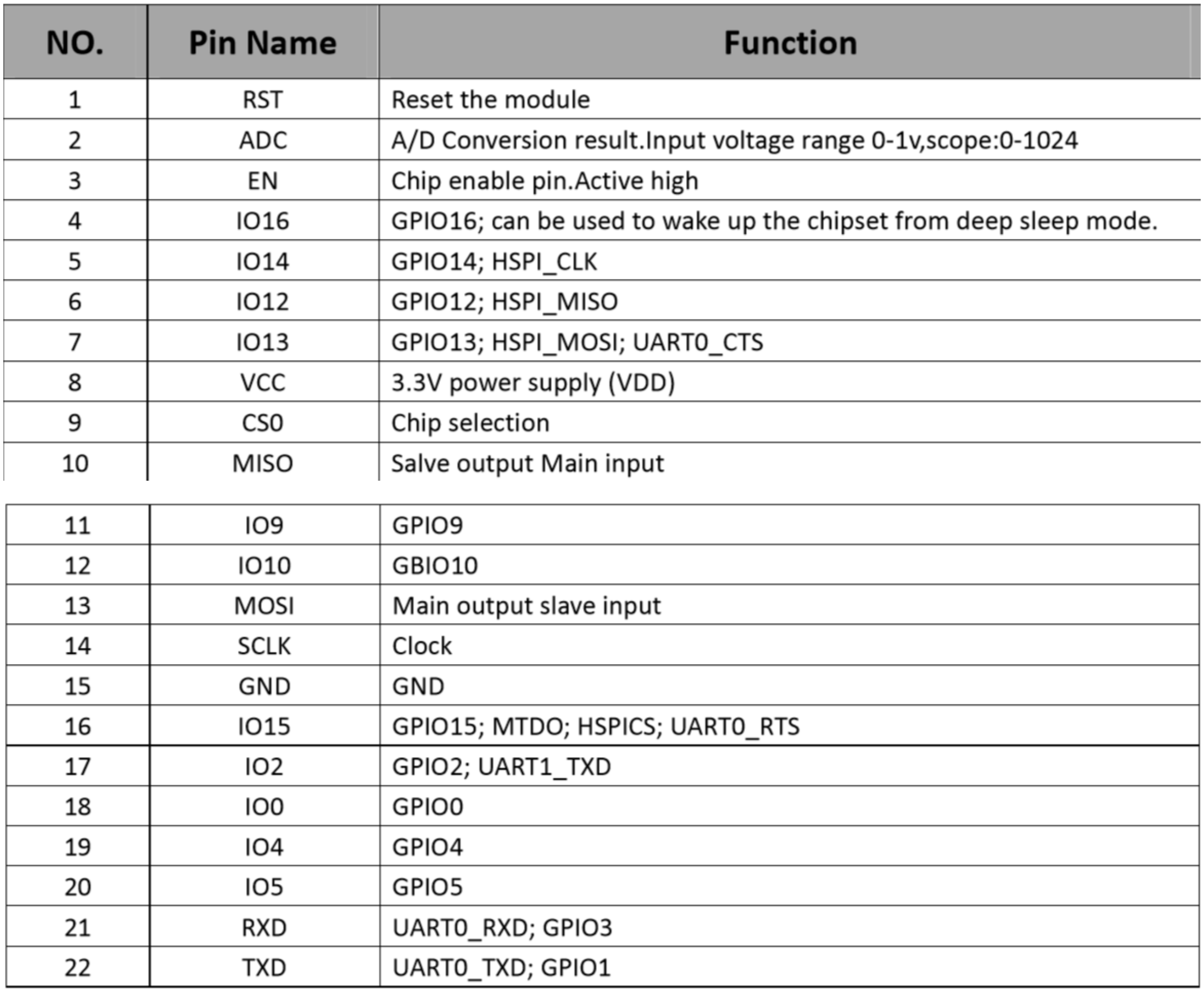


**Fig 3.2: ESP 12E pin design**

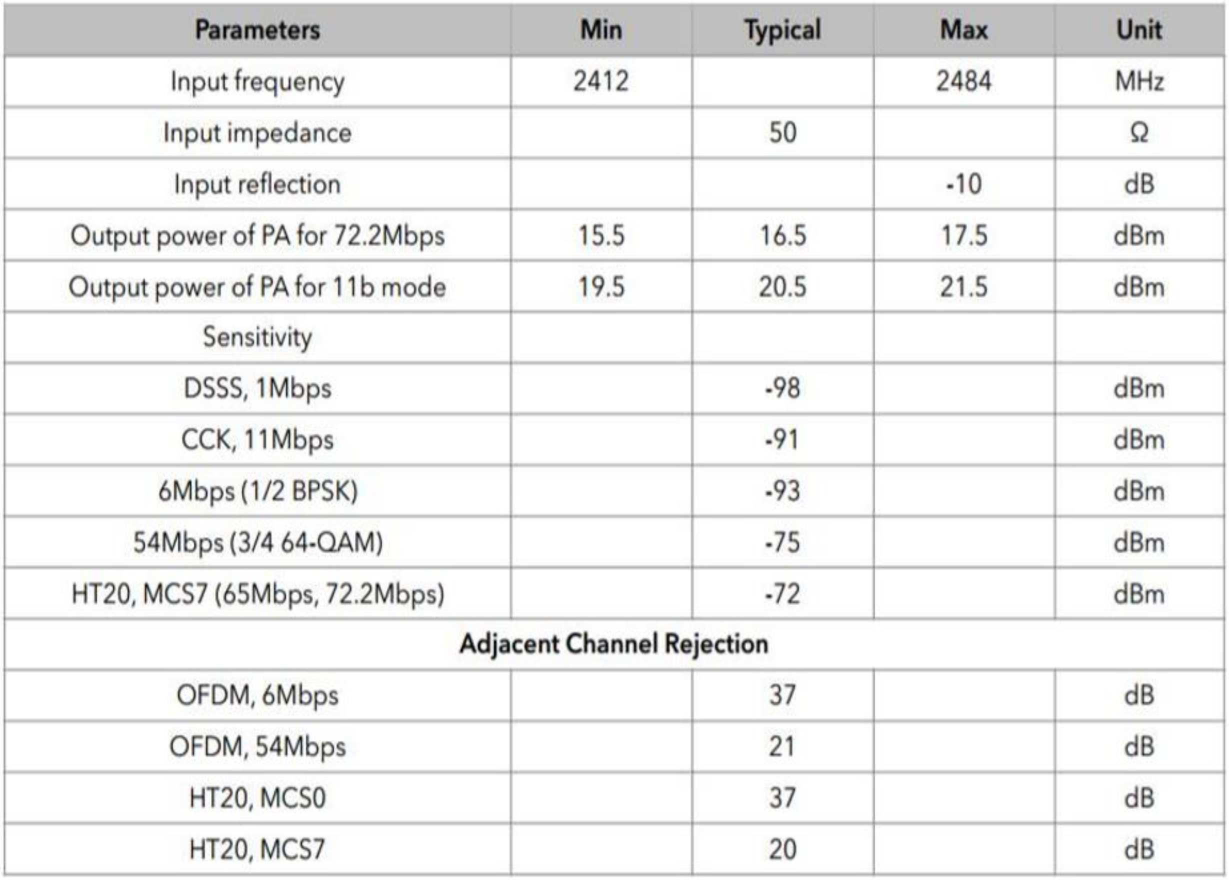
## PIN Description Table:



| NO. | Pin Name | Function |
|---|---|---|
| 1 | RST | Reset the module |
| 2 | ADC | A/D Conversion result.Input voltage range 0-1v,scope:0-1024 |
| 3 | EN | Chip enable pin.Active high |
| 4 | IO16 | GPIO16; can be used to wake up the chipset from deep sleep mode. |
| 5 | IO14 | GPIO14; HSPI_CLK |
| 6 | IO12 | GPIO12; HSPI_MISO |
| 7 | IO13 | GPIO13; HSPI_MOSI; UART0_CTS |
| 8 | VCC | 3.3V power supply (VDD) |
| 9 | CS0 | Chip selection |
| 10 | MISO | Salve output Main input |
| 11 | IO9 | GPIO9 |
| 12 | IO10 | GBIO10 |
| 13 | MOSI | Main output slave input |
| 14 | SCLK | Clock |
| 15 | GND | GND |
| 16 | IO15 | GPIO15; MTDO; HSPICS; UART0_RTS |
| 17 | IO2 | GPIO2; UART1_TXD |
| 18 | IO0 | GPIO0 |
| 19 | IO4 | GPIO4 |
| 20 | IO5 | GPIO5 |
| 21 | RXD | UART0_RXD; GPIO3 |
| 22 | TXD | UART0_TXD; GPIO1 |

**Table 3.1: PIN Description Table**

## Receiver Sensitivity:



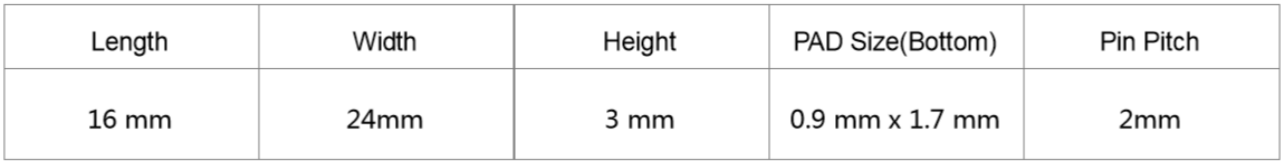
| Parameters | Min | Typical | Max | Unit |
|---|---|---|---|---|
| Input frequency | 2412 | | 2484 | MHz |
| Input impedance | | 50 | | Ω |
| Input reflection | | | -10 | dB |
| Output power of PA for 72.2Mbps | 15.5 | 16.5 | 17.5 | dBm |
| Output power of PA for 11b mode | 19.5 | 20.5 | 21.5 | dBm |
| Sensitivity | | | | |
| DSSS, 1Mbps | | -98 | | dBm |
| CCK, 11Mbps | | -91 | | dBm |
| 6Mbps (1/2 BPSK) | | -93 | | dBm |
| 54Mbps (3/4 64-QAM) | | -75 | | dBm |
| HT20, MCS7 (65Mbps, 72.2Mbps) | | -72 | | dBm |
| **Adjacent Channel Rejection** | | | | |
| OFDM, 6Mbps | | 37 | | dB |
| OFDM, 54Mbps | | 21 | | dB |
| HT20, MCS0 | | 37 | | dB |
| HT20, MCS7 | | 20 | | dB |

**Table 3.2: Receiver Sensitivity**

## 3.3 PACKAGING AND DIMENSIONS:

The external size of the module is 16mm*24mm*3mm . The type of flash integrated in this module is an SPI flash, the capacity of which is 4 MB, and the package size of which is SOP-210mil. The antenna applied on this module is a 3DBi PCB-on-board antenna.

**Dimensions of ESP 12E module:**



| Length | Width | Height | PAD Size(Bottom) | Pin Pitch |
|---|---|---|---|---|
| 16 mm | 24mm | 3 mm | 0.9 mm x 1.7 mm | 2mm |

**Table 3.3: Dimensions of ESP 12E module**

## 3.4 FUNCTIONAL DESCRIPTIONS:

### 3.4.1 MCU :

ESP8266EX is embedded with Ten silica L106 32-bit micro controller (MCU), which features extra low power consumption and 16-bit RSIC. The CPU clock speed is 80MHz. It can also reach a maximum value of 160MHz. ESP8266EX is often integrated with external sensors and other specific devices through its GPIOs; codes for such applications are provided in examples in the SDK.

### 3.4.2 Memory Organization

### Internal SRAM and ROM :

ESP8266EX WiFi SOC is embedded with memory controller, including SRAM and ROM. MCU can visit the memory units through iBus, dBus, and AHB interfaces. All memory units can be visited upon request, while a memory arbiter will decide the running sequence according to the time when these requests are received by the processor.

According to our current version of SDK provided, SRAM space that is available to users is assigned as below:

- RAM size < 36kB, that is to say, when ESP8266EX is working under the station mode and is connected to the router, programmable space accessible to user in heap and data section is around 36kB.)
- There is no programmable ROM in the SoC, therefore, user program must be stored in an external SPI flash.

### 3.4.3 External SPI Flash :

This module is mounted with an 4 MB external SPI flash to store user programs. If larger definable storage space is required, a SPI flash with larger memory size is preferred. Theoretically speaking, up to 16 MB memory capacity can be supported. Suggested SPI Flash memory capacity:

- **OTA is disabled**: the minimum flash memory that can be supported is 512 kB;
- **OTA is enabled**: the minimum flash memory that can be supported is 1 MB. Several SPI modes can be supported, including Standard SPI, Dual SPI, and Quad SPI.

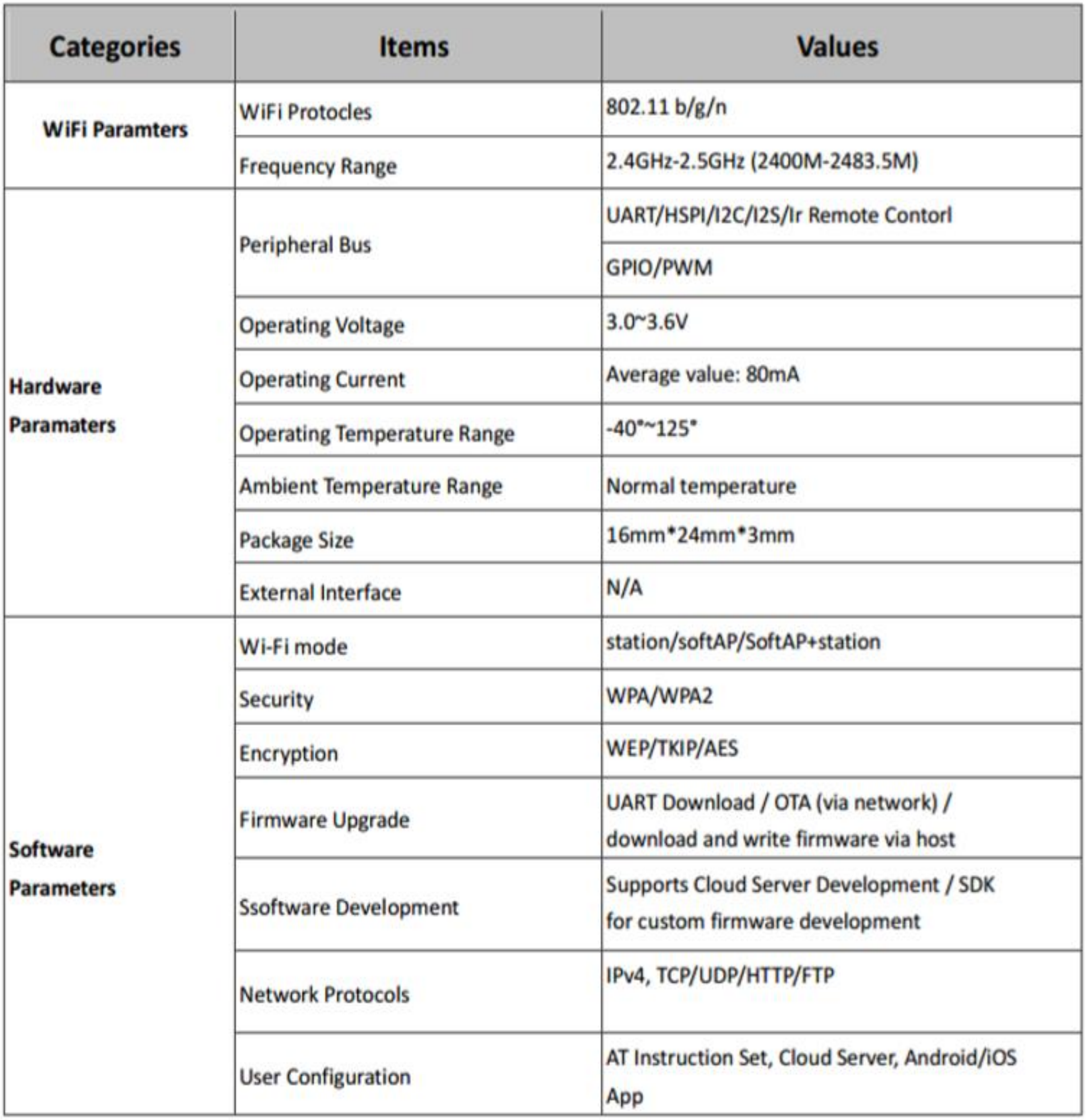
Therefore, please choose the correct SPI mode when you are downloading into the flash, otherwise firmwares/programs that you downloaded may not work in the right way.

### 3.4.4 Crystal :

Currently, the frequency of crystal oscillators supported include 40MHz, 26MHz and 24MHz. The accuracy of crystal oscillators applied should be ±10PPM, and the operating temperature range should be between -20°C and 85°C.

When using the downloading tools, please remember to select the right crystal oscillator type. In circuit design, capacitors C1 and C2, which are connected to the earth, are added to the input and output terminals of the crystal oscillator respectively. The values of the two capacitors can be flexible, ranging from 6pF to 22pF, however, the specific capacitive values of C1 and C2 depend on further testing and adjustment on the overall performance of the whole circuit. Normally, the capacitive values of C1 and C2 are within 10pF if the crystal oscillator frequency is 26MHz, while the values of C1 and C2 are 10pF<C1, C2<22pF if the crystal oscillator frequency is 40MHz.

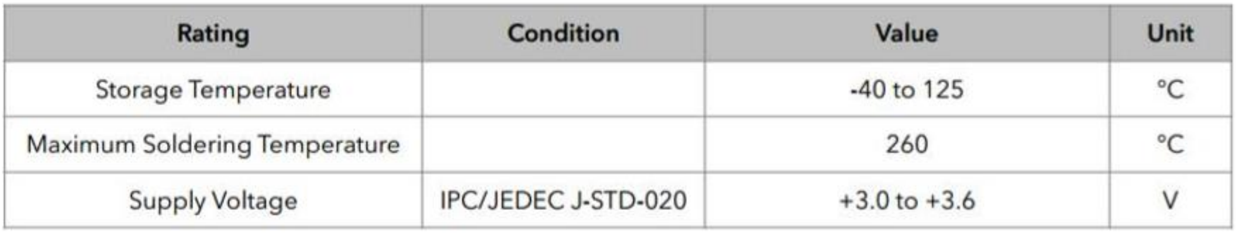
## PARAMETERS:



| Categories | Items | Values |
|---|---|---|
| WiFi Paramters | WiFi Protocles | 802.11 b/g/n |
| | Frequency Range | 2.4GHz-2.5GHz (2400M-2483.5M) |
| Hardware Paramaters | Peripheral Bus | UART/HSPI/I2C/I2S/Ir Remote Contorl<br>GPIO/PWM |
| | Operating Voltage | 3.0~3.6V |
| | Operating Current | Average value: 80mA |
| | Operating Temperature Range | -40°~125° |
| | Ambient Temperature Range | Normal temperature |
| | Package Size | 16mm*24mm*3mm |
| | External Interface | N/A |
| Software Parameters | Wi-Fi mode | station/softAP/SoftAP+station |
| | Security | WPA/WPA2 |
| | Encryption | WEP/TKIP/AES |
| | Firmware Upgrade | UART Download / OTA (via network) / download and write firmware via host |
| | Ssoftware Development | Supports Cloud Server Development / SDK for custom firmware development |
| | Network Protocols | IPv4, TCP/UDP/HTTP/FTP |
| | User Configuration | AT Instruction Set, Cloud Server, Android/iOS App |

**Table 3.4: Parameters of ESP8266**

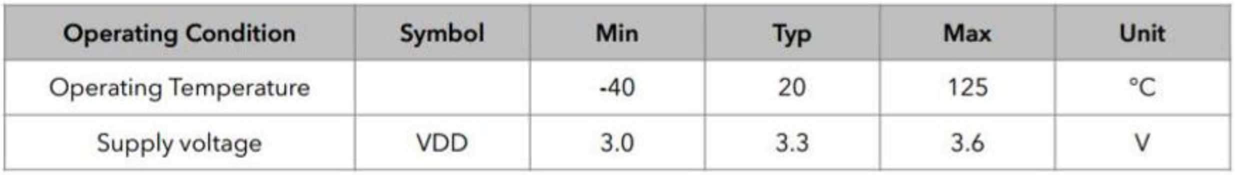
## Absolute maximum rating:



| Rating | Condition | Value | Unit |
|---|---|---|---|
| Storage Temperature | | -40 to 125 | °C |
| Maximum Soldering Temperature | | 260 | °C |
| Supply Voltage | IPC/JEDEC J-STD-020 | +3.0 to +3.6 | V |

Table 3.5: Absolute maximum rating

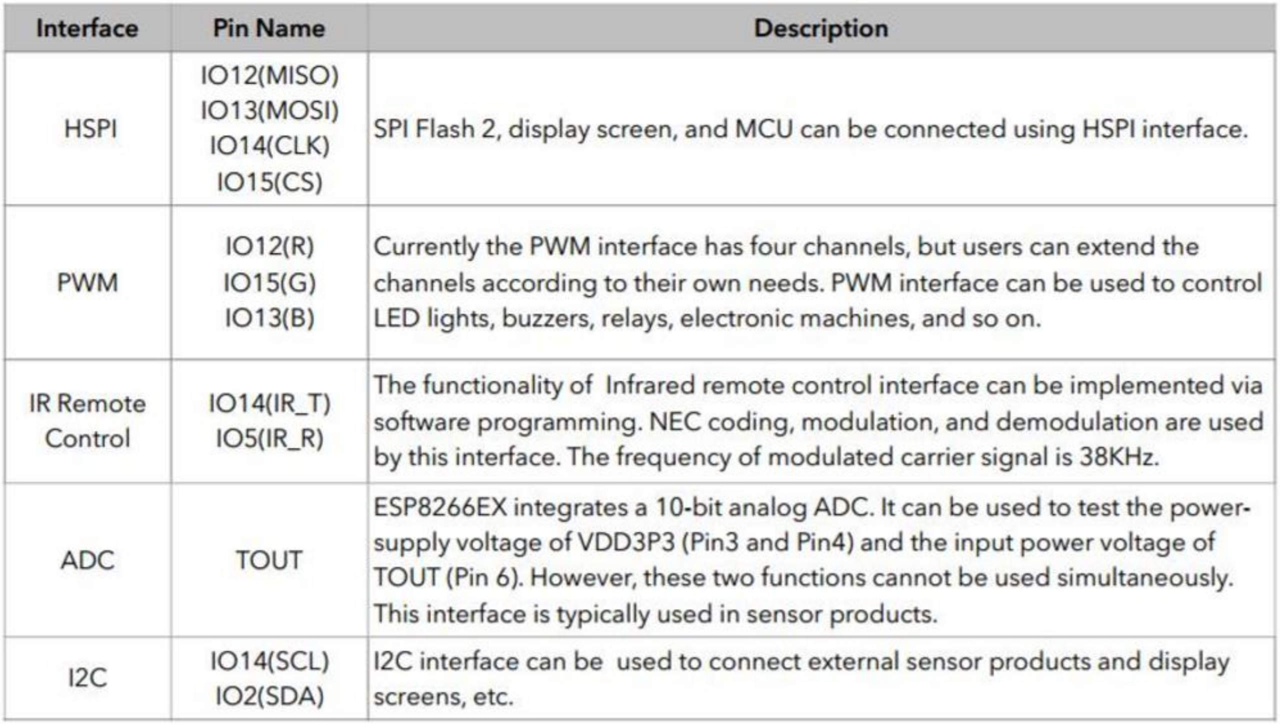
## Recommended Operating Conditions:



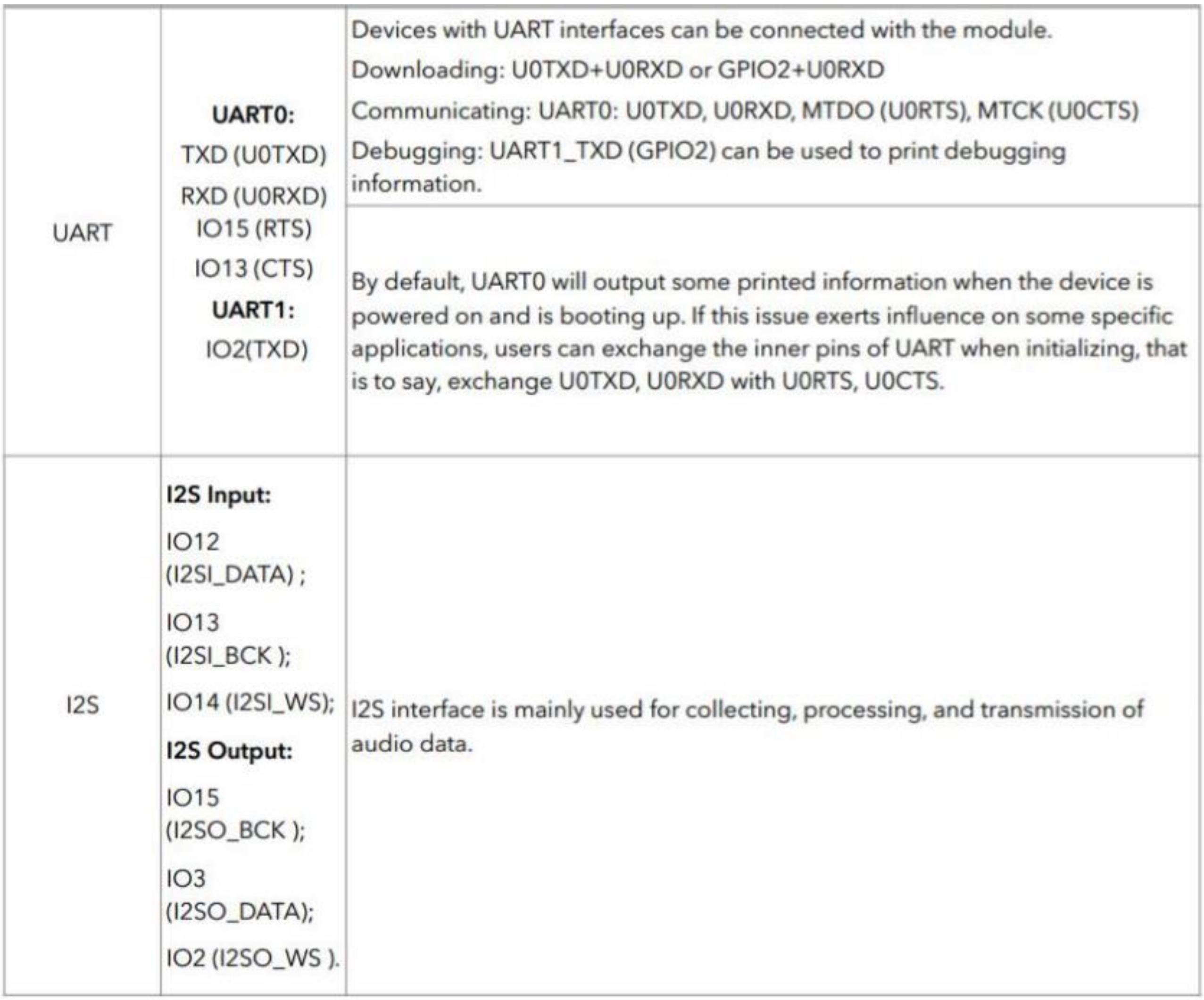
| Operating Condition | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Operating Temperature | | -40 | 20 | 125 | °C |
| Supply voltage | VDD | 3.0 | 3.3 | 3.6 | V |

Table 3.6: Recommended Operating Conditions

## Interfaces:



| Interface | Pin Name | Description |
|---|---|---|
| HSPI | IO12(MISO)<br>IO13(MOSI)<br>IO14(CLK)<br>IO15(CS) | SPI Flash 2, display screen, and MCU can be connected using HSPI interface. |
| PWM | IO12(R)<br>IO15(G)<br>IO13(B) | Currently the PWM interface has four channels, but users can extend the channels according to their own needs. PWM interface can be used to control LED lights, buzzers, relays, electronic machines, and so on. |
| IR Remote Control | IO14(IR_T)<br>IO5(IR_R) | The functionality of Infrared remote control interface can be implemented via software programming. NEC coding, modulation, and demodulation are used by this interface. The frequency of modulated carrier signal is 38KHz. |
| ADC | TOUT | ESP8266EX integrates a 10-bit analog ADC. It can be used to test the power-supply voltage of VDD3P3 (Pin3 and Pin4) and the input power voltage of TOUT (Pin 6). However, these two functions cannot be used simultaneously. This interface is typically used in sensor products. |
| I2C | IO14(SCL)<br>IO2(SDA) | I2C interface can be used to connect external sensor products and display screens, etc. |



| | | |
|---|---|---|
| UART | **UART0:**<br>TXD (U0TXD)<br>RXD (U0RXD)<br>IO15 (RTS)<br>IO13 (CTS)<br>**UART1:**<br>IO2(TXD) | Devices with UART interfaces can be connected with the module.<br>Downloading: U0TXD+U0RXD or GPIO2+U0RXD<br>Communicating: UART0: U0TXD, U0RXD, MTDO (U0RTS), MTCK (U0CTS)<br>Debugging: UART1_TXD (GPIO2) can be used to print debugging information. |
| | | By default, UART0 will output some printed information when the device is powered on and is booting up. If this issue exerts influence on some specific applications, users can exchange the inner pins of UART when initializing, that is to say, exchange U0TXD, U0RXD with U0RTS, U0CTS. |
| I2S | **I2S Input:**<br>IO12 (I2SI_DATA) ;<br>IO13 (I2SI_BCK );<br>IO14 (I2SI_WS);<br>**I2S Output:**<br>IO15 (I2SO_BCK );<br>IO3 (I2SO_DATA);<br>IO2 (I2SO_WS ). | I2S interface is mainly used for collecting, processing, and transmission of audio data. |

**Table 3.7: Interfaces for ESP8266**

## Schematic:

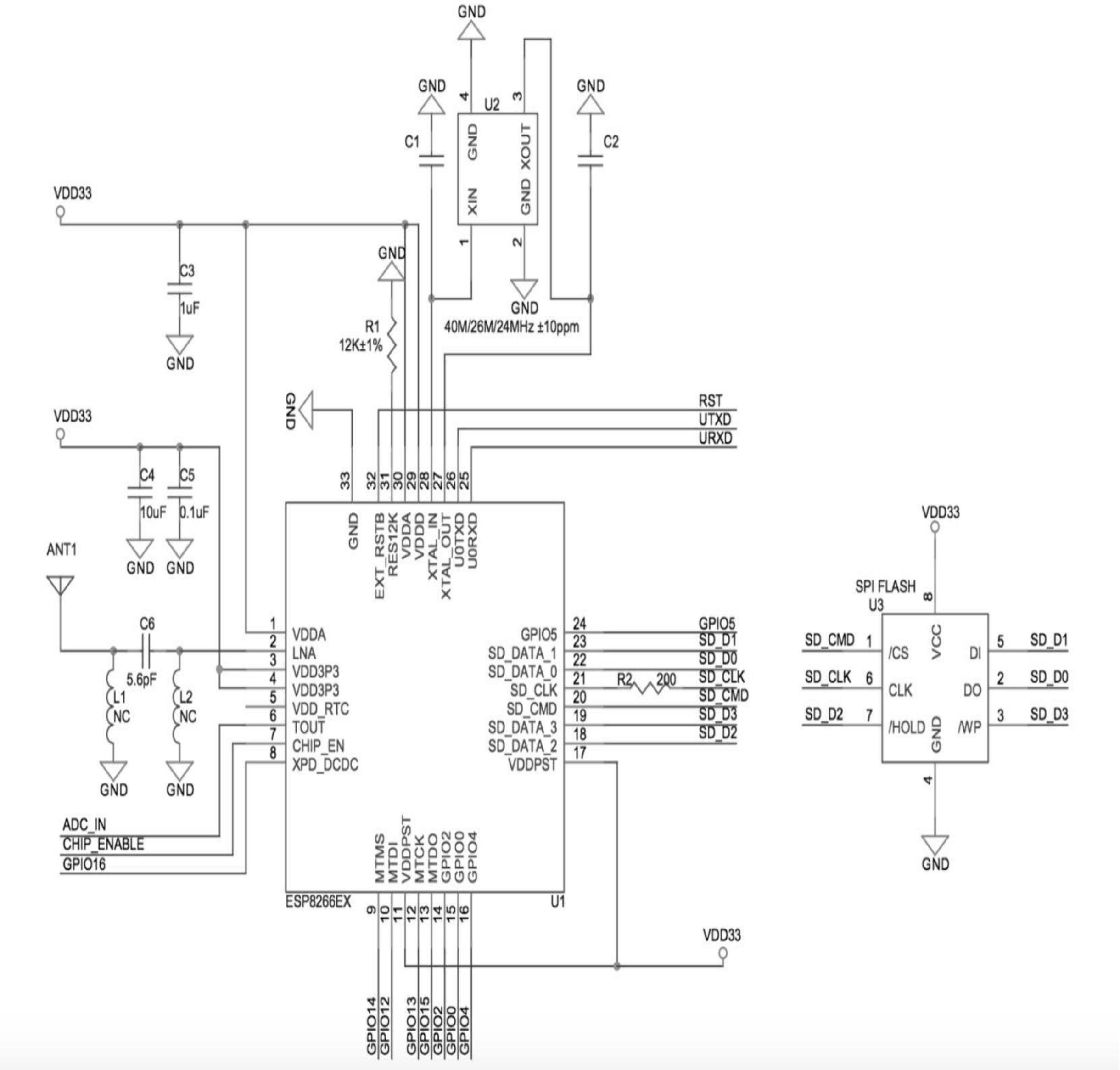


**Fig 3.3: Schematic diagram for ESP8266**

# CHAPTER-4
# CP2102 USB TO UART BRIDGE

## 4.1 INTRODUCTION:

The CP2102/9 is a highly-integrated USB-to-UART Bridge Controller providing a simple solution for updating RS- 232 designs to USB using a minimum of components and PCB space. The CP2102/9 includes a USB 2.0 full speed function controller, USB transceiver, oscillator, EEPROM or EPROM, and asynchronous serial data bus (UART) with full modem control signals in a compact 5 x 5 mm QFN-28 package. No other external USB components are required.

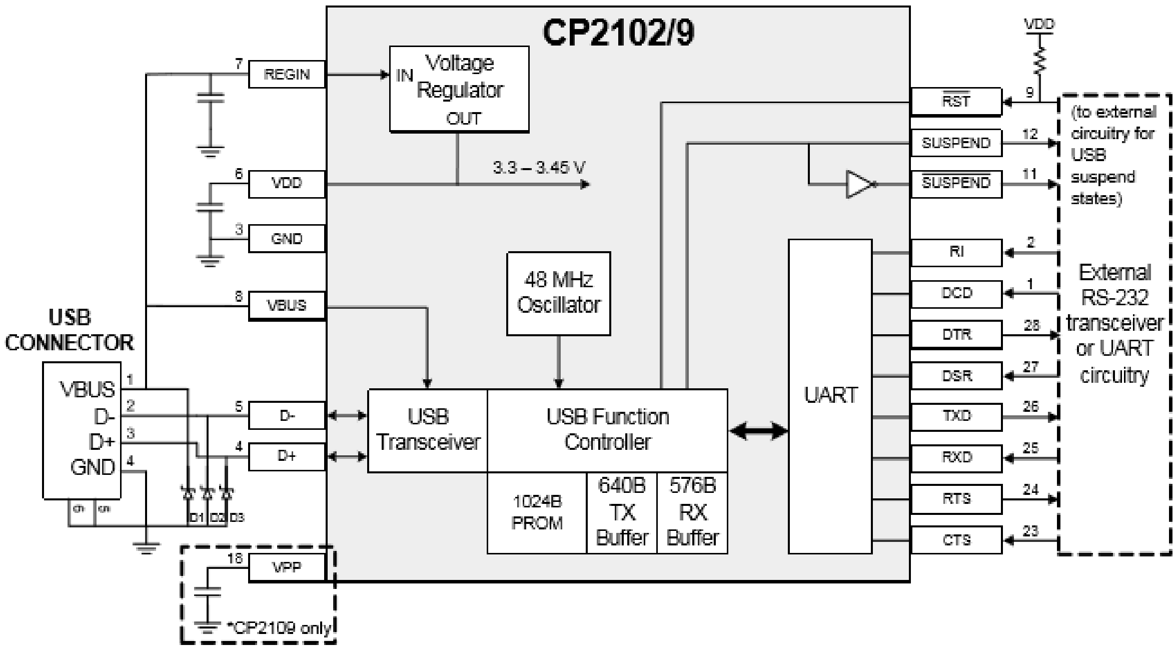


**Fig 4.1: Block Diagram of CP2102/9**

## 4.2 FEATURES:

### 4.2.1 Single-Chip USB to UART Data Transfer

- Integrated USB transceiver; no external resistors required
- Integrated clock; no external crystal required
- Internal 1024-byte programmable ROM for vendor ID, product ID, serial number, power descriptor, release number, and product description strings

- EEPROM (CP2102)
- On-chip power-on reset circuit
- On-chip voltage regulator
- 3.3 V output (CP2102)
- 100% pin and software compatible with CP2101

### 4.2.2 USB Function Controller

- USB Specification 2.0 compliant; full-speed (12 Mbps)
- USB suspend states supported via SUSPEND pins

### 4.2.3 Asynchronous Serial Data BUS (UART)

- All handshaking and modem interface signals
- Data formats supported:
- Data bits: 5, 6, 7, and 8
- Stop bits: 1, 1.5, and 2
- Parity: odd, even, mark, space, no parity
- Baud rates: 300 bps to 1 Mbps
- 576 Byte receive buffer; 640 byte transmit buffer
- Hardware or X-On/X-Off handshaking supported
- Event character support
- Line break transmission

### 4.2.4 Supply Voltage

- Self-powered: 3.0 to 3.6 V
- USB bus powered: 4.0 to 5.25 V

### 4.2.5 Package

- RoHS-compliant 28-pin QFN (5x5 mm)

**4.2.6 Temperature Range:** –40 to +85 °C

## PIN DIAGRAM OF CP2102/9:

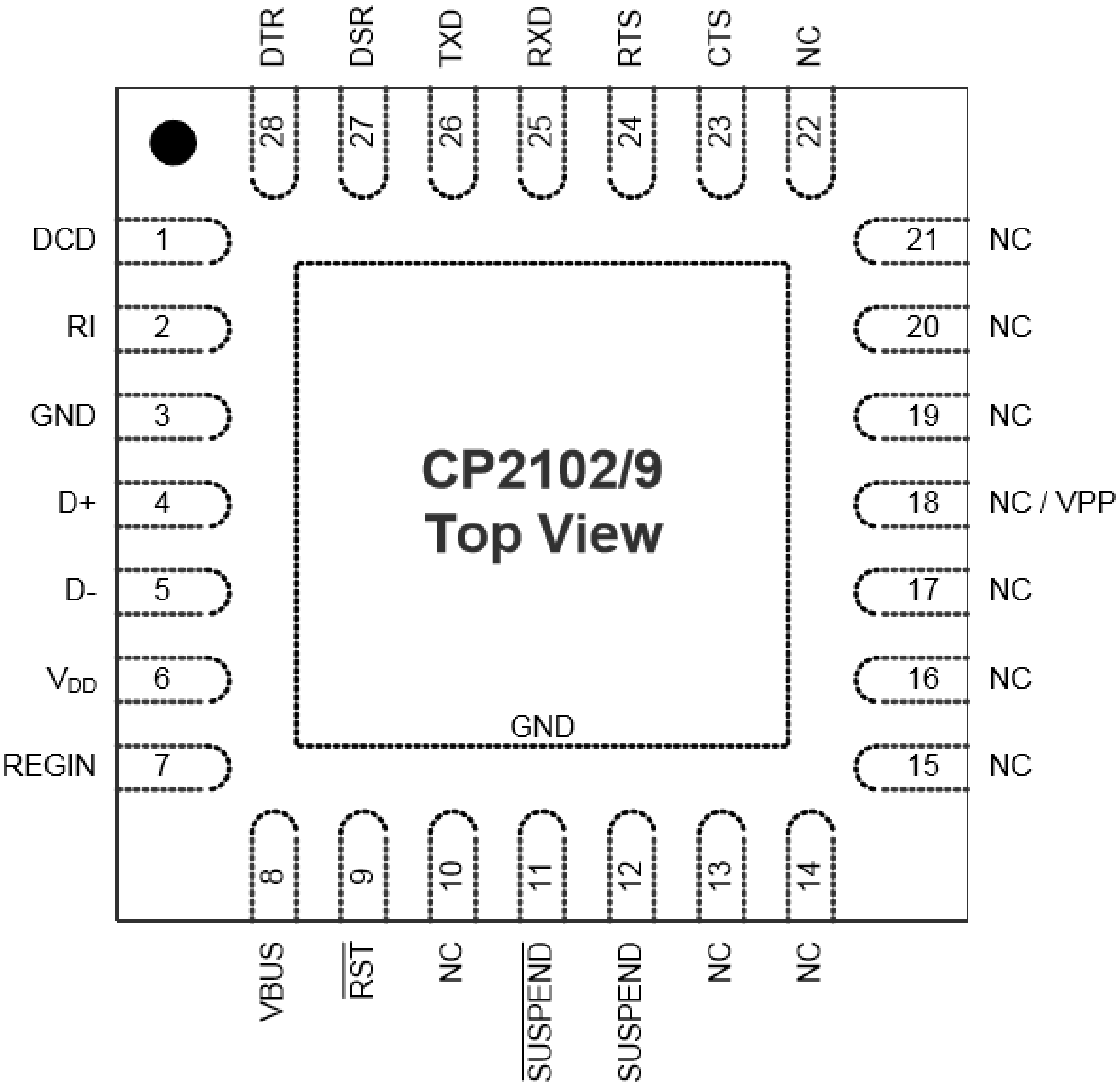
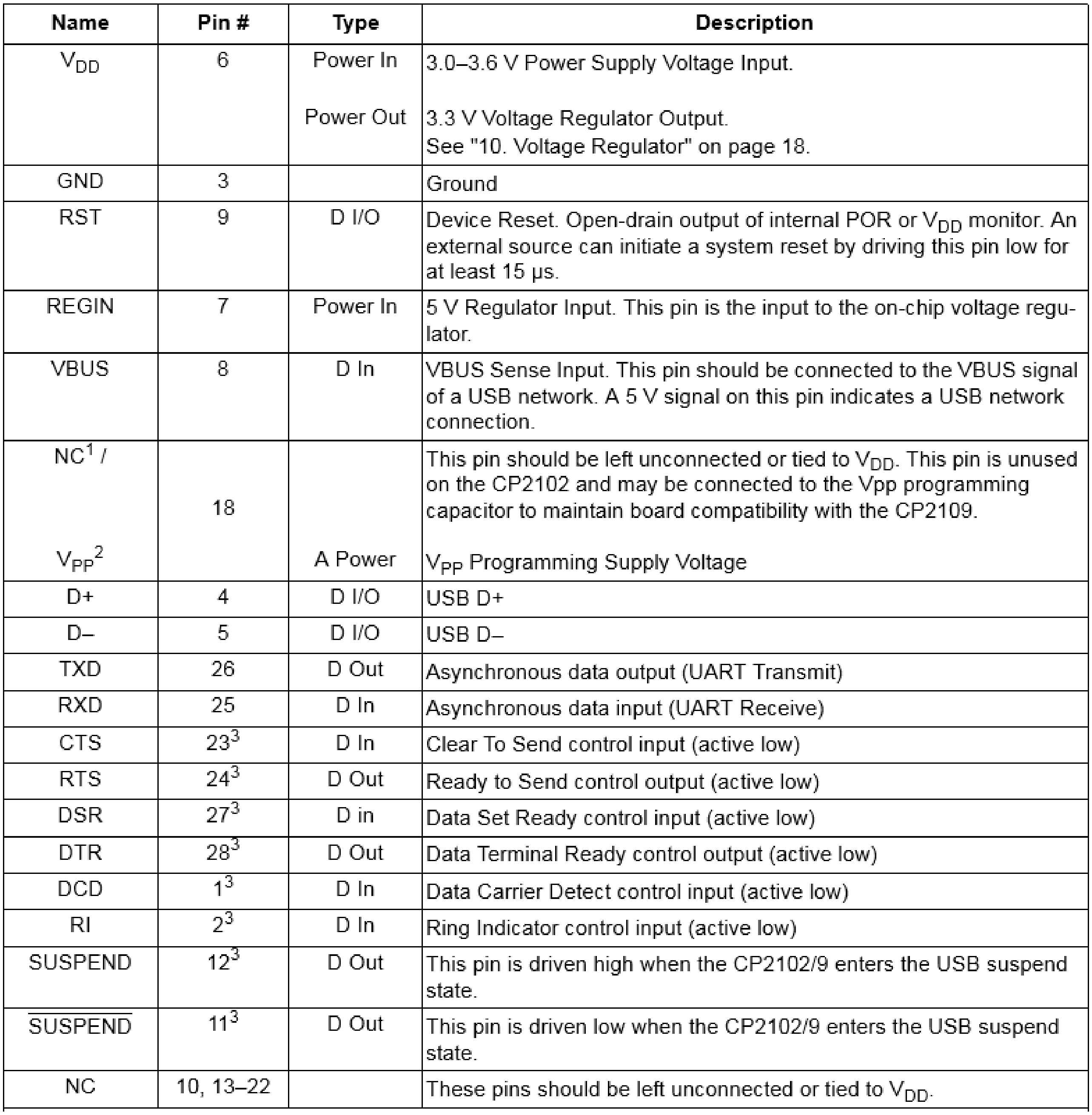


**Fig 4.2: Pin Diagram of CP2102**

## PINOUT AND PACKAGE DEFINITIONS:



| Name | Pin # | Type | Description |
|---|---|---|---|
| $V_{DD}$ | 6 | Power In<br>Power Out | 3.0–3.6 V Power Supply Voltage Input.<br>3.3 V Voltage Regulator Output. See "10. Voltage Regulator" on page 18. |
| GND | 3 | | Ground |
| RST | 9 | D I/O | Device Reset. Open-drain output of internal POR or $V_{DD}$ monitor. An external source can initiate a system reset by driving this pin low for at least 15 µs. |
| REGIN | 7 | Power In | 5 V Regulator Input. This pin is the input to the on-chip voltage regulator. |
| VBUS | 8 | D In | VBUS Sense Input. This pin should be connected to the VBUS signal of a USB network. A 5 V signal on this pin indicates a USB network connection. |
| NC[1] /<br>$V_{PP}$[2] | 18 | <br>A Power | This pin should be left unconnected or tied to $V_{DD}$. This pin is unused on the CP2102 and may be connected to the Vpp programming capacitor to maintain board compatibility with the CP2109.<br>$V_{PP}$ Programming Supply Voltage |
| D+ | 4 | D I/O | USB D+ |
| D– | 5 | D I/O | USB D– |
| TXD | 26 | D Out | Asynchronous data output (UART Transmit) |
| RXD | 25 | D In | Asynchronous data input (UART Receive) |
| CTS | 23[3] | D In | Clear To Send control input (active low) |
| RTS | 24[3] | D Out | Ready to Send control output (active low) |
| DSR | 27[3] | D in | Data Set Ready control input (active low) |
| DTR | 28[3] | D Out | Data Terminal Ready control output (active low) |
| DCD | 1[3] | D In | Data Carrier Detect control input (active low) |
| RI | 2[3] | D In | Ring Indicator control input (active low) |
| SUSPEND | 12[3] | D Out | This pin is driven high when the CP2102/9 enters the USB suspend state. |
| $\overline{\text{SUSPEND}}$ | 11[3] | D Out | This pin is driven low when the CP2102/9 enters the USB suspend state. |
| NC | 10, 13–22 | | These pins should be left unconnected or tied to $V_{DD}$. |

**Table 4.1: Pinout and package definitions.**

# CHAPTER-5

# DHT11 - HUMIDITY AND TEMPERATURE SENSOR

## 5.1 INTRODUCTION:

The DHT11 is a basic, low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin (no analog input pins needed). It is fairly simple to use, but requires careful timing to grab data. The only real downside of this sensor is you can only get new data from it once every 2 seconds.

## 5.2 DETAILS:

This sensor includes a resistive-type humidity measurement component and an NTC temperature measurement component, and connects to a high-performance 8-bit microcontroller, offering excellent quality, fast response, anti-interference ability and cost-effectiveness. The calibration coefficients are stored as programmers in the OTP memory, which are used by the sensor's internal signal detecting process. The single-wire serial interface makes system integration quick and easy. Its small size, low power consumption and up-to-20 meter signal transmission making it the best choice for various applications, including those most demanding ones. The component is 4-pin single row pin package.

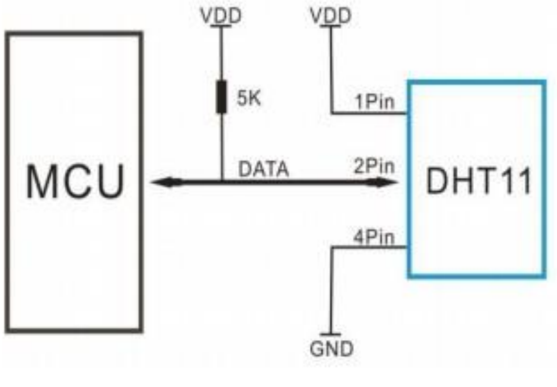


**Fig 5.1: Block Diagram of DHT11.**

DHT11's power supply is 3-5.5V DC. When power is supplied to the sensor, do not send any instruction to the sensor in within one second in order to pass the unstable status. One capacitor valued 100nF can be added between VDD and GND for power filtering.

## 5.3 FEATURES:

- Full range temperature compensated
- Relative humidity and temperature measurement
- Calibrated digital signal
- Outstanding long-term stability
- Extra components not needed
- Long transmission distance
- Low power consumption
- 4 pins packaged and fully interchangeable

## 5.4 COMMUNICATION PROCESS:

Serial Interface (Single-Wire Two-Way) The interesting thing in this module is the protocol that uses to transfer data. All the sensor readings are sent using a single wire bus which reduces the cost and extends the distance. In order to send data over a bus you have to describe the way the data will be transferred, so that transmitter and receiver can understand what says each other. This is what a protocol does. It describes the way the data are transmitted. On DHT-11 the 1-wire data bus is pulled up with a resistor to VCC. So if nothing is occurred the voltage on the bus is equal to VCC.

Communication Format can be seperated into three stages

1) Request

2) Response

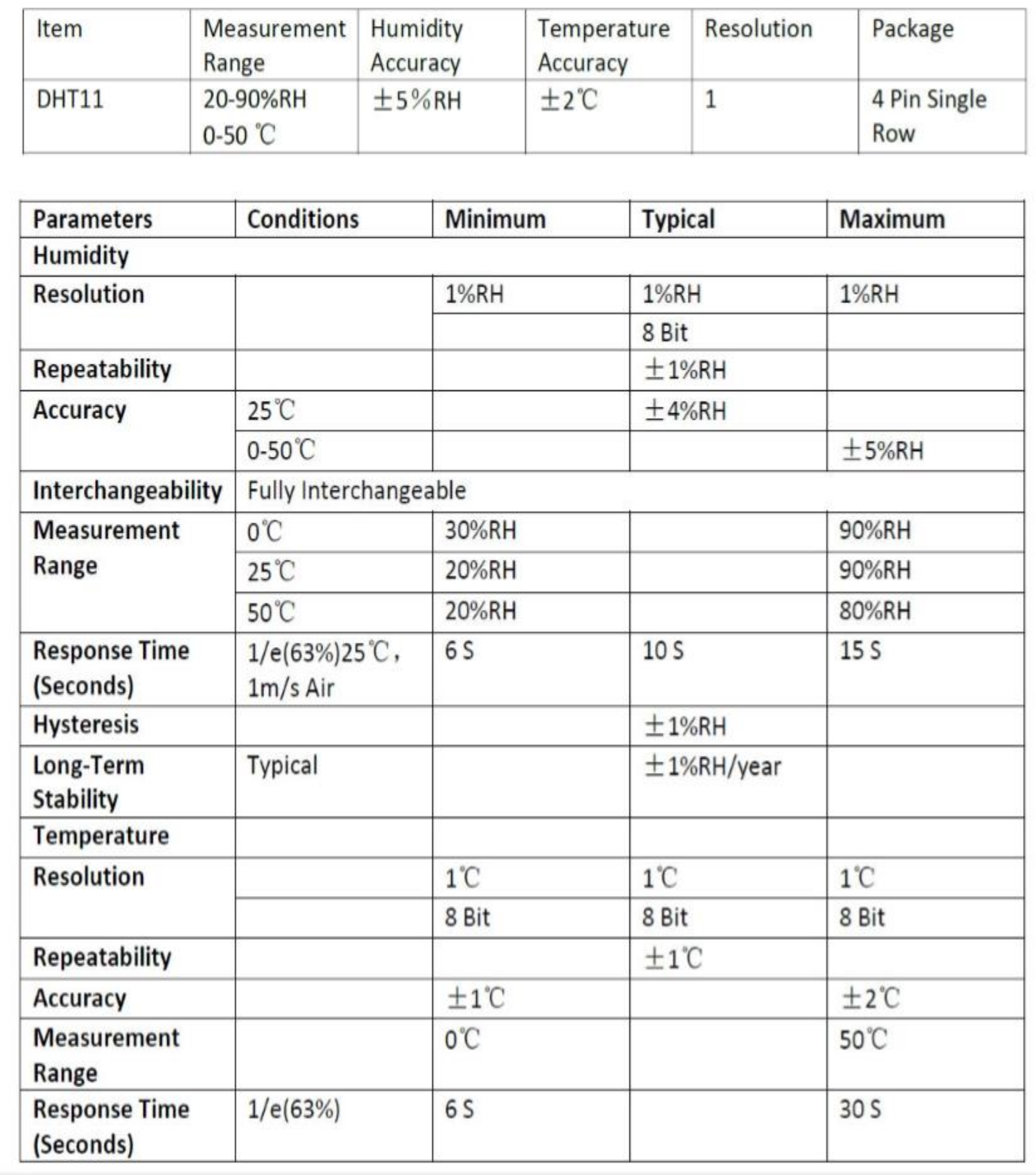
3) Data Reading

1) Request: To make the DHT-11 to send you the sensor readings you have to send it a request. The request is, to pull down the bus for more than 18ms in order to give DHT time to understand it and then pull it up for 40uS.

2) Response: What comes after the request is the DHT-11 response. This is an automatic reply from DHT which indicates that DHT received your request. The response is ~54uS low and 80uS high.

3) Data Reading: What will come after the response is the sensor data. The data will be packed in a packet of 5 segments of 8-bits each. Totally 5×8 =40bits.

# SPECIFICATIONS:



| Item | Measurement Range | Humidity Accuracy | Temperature Accuracy | Resolution | Package |
|---|---|---|---|---|---|
| DHT11 | 20-90%RH<br>0-50 ℃ | ±5%RH | ±2℃ | 1 | 4 Pin Single Row |

| Parameters | Conditions | Minimum | Typical | Maximum |
|---|---|---|---|---|
| Humidity | | | | |
| Resolution | | 1%RH | 1%RH | 1%RH |
| | | | 8 Bit | |
| Repeatability | | | ±1%RH | |
| Accuracy | 25℃ | | ±4%RH | |
| | 0-50℃ | | | ±5%RH |
| Interchangeability | Fully Interchangeable | | | |
| Measurement Range | 0℃ | 30%RH | | 90%RH |
| | 25℃ | 20%RH | | 90%RH |
| | 50℃ | 20%RH | | 80%RH |
| Response Time (Seconds) | 1/e(63%)25℃，1m/s Air | 6 S | 10 S | 15 S |
| Hysteresis | | | ±1%RH | |
| Long-Term Stability | Typical | | ±1%RH/year | |
| Temperature | | | | |
| Resolution | | 1℃ | 1℃ | 1℃ |
| | | 8 Bit | 8 Bit | 8 Bit |
| Repeatability | | | ±1℃ | |
| Accuracy | | ±1℃ | | ±2℃ |
| Measurement Range | | 0℃ | | 50℃ |
| Response Time (Seconds) | 1/e(63%) | 6 S | | 30 S |

**Table 5.1: Specifications of DHT11**

# CHAPTER-6
# SOIL MOISTURE SENSOR

## 6.1 INTRODUCTION:

The Soil Moisture Sensor is used to measure the volumetric water content of soil. This makes it ideal for performing experiments in courses such as soil science, agricultural science, environmental science, horticulture, botany, and biology.

Use the Soil Moisture Sensor to:

- Measure the loss of moisture over time due to evaporation and plant uptake.
- Evaluate optimum soil moisture contents for various species of plants.
- Monitor soil moisture content to control irrigation in greenhouses.

## 6.2 WORKING PRINCIPLE:

The Soil Moisture Sensor uses capacitance to measure dielectric permittivity of the surrounding medium. In soil, dielectric permittivity is a function of the water content. The sensor creates a voltage proportional to the dielectric permittivity, and therefore the water content of the soil.

## 6.3 SPECIFICATIONS:

Working voltage: 5V

Working current: <20ma

Interface: Analog

Arduino compatible interface

Output voltage signal: 0~4.2V

## CONNECTING DIAGRAM:

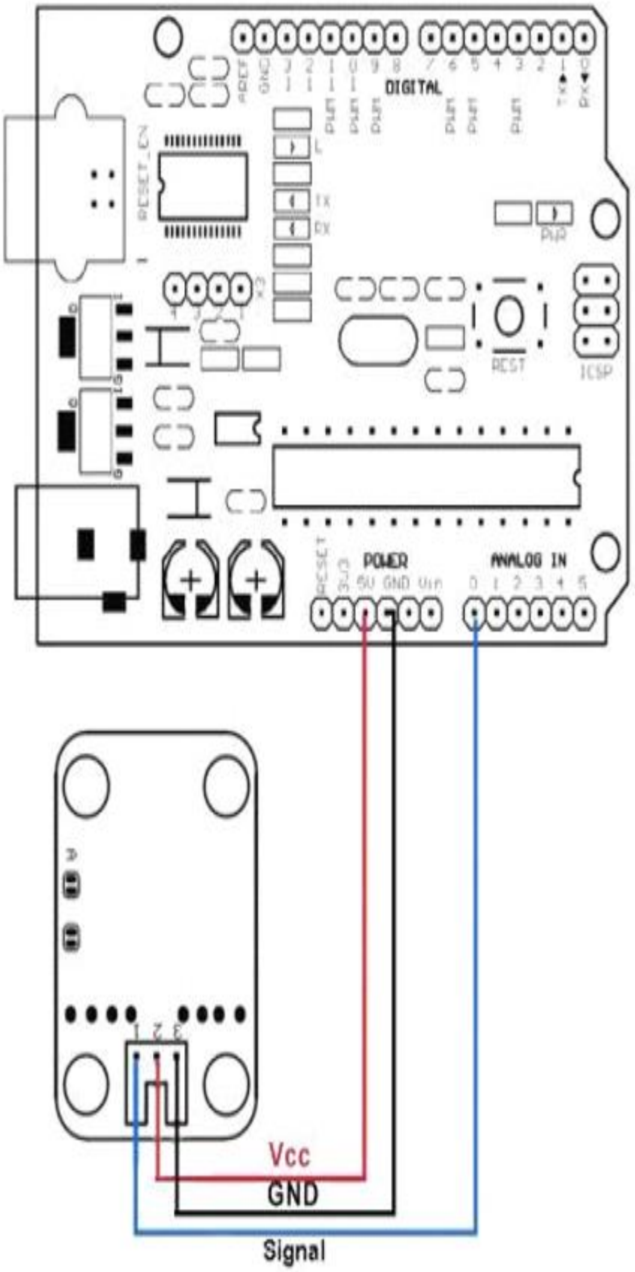


**Fig 6.1: Connecting Diagram of Soil moisture sensor**

# CHAPTER-7

# DC MOTOR AND RELAY

## 7.1 DC MOTOR PRINCIPLE:

A machine that converts dc power into mechanical energy is known as dc motor. Its operation is based on the principle that when a current carrying conductor is placed in a magnetic field, the conductor experiences a mechanical force. The direction of the force is given by Fleming's left hand rule.

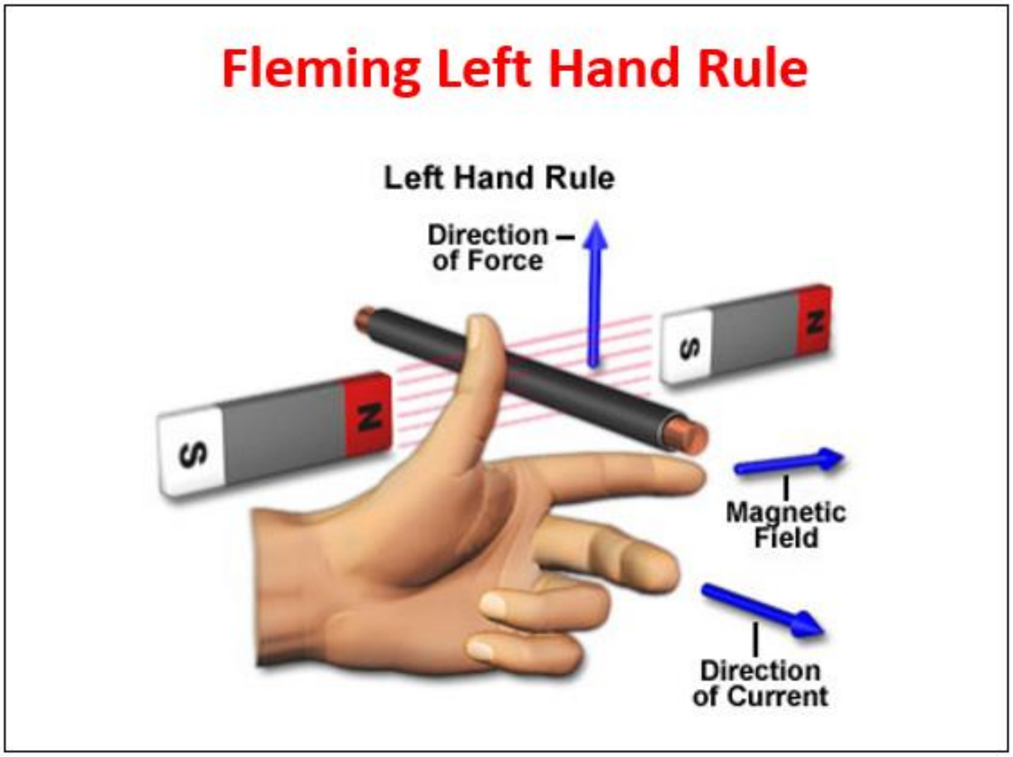


**Fig 7.1: Fleming Left Hand Rule.**

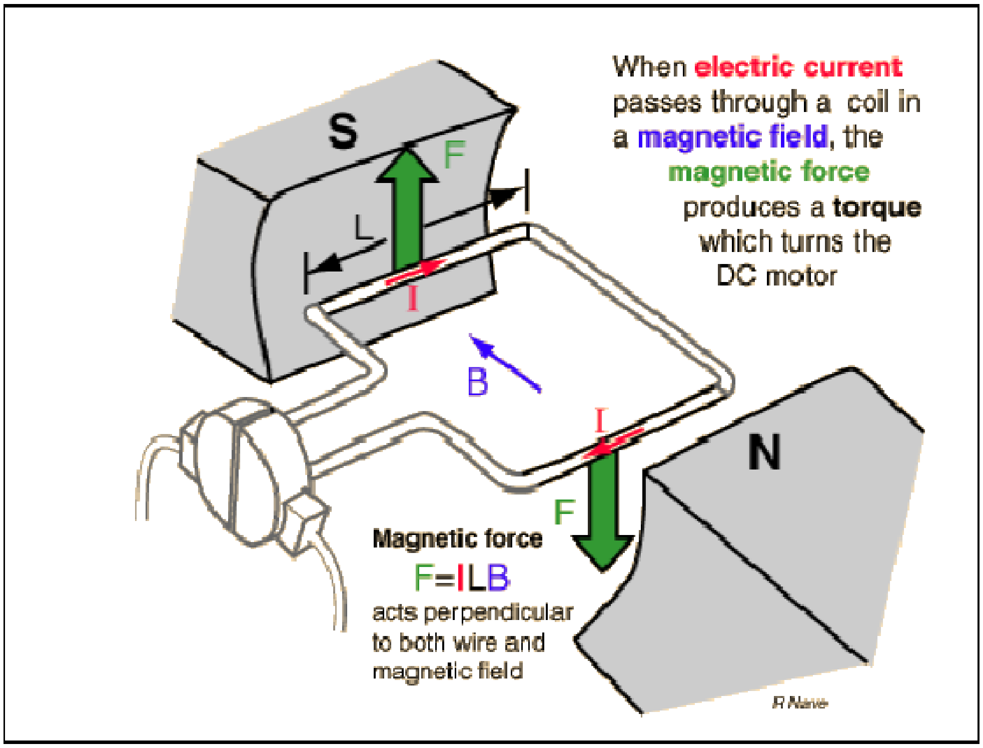

When electric current passes through a coil in a magnetic field, the magnetic force produces a torque which turns the DC motor
S
N
F
L
I
B
F
I
Magnetic force
F=ILB
acts perpendicular to both wire and magnetic field
R Nave

**Fig 7.2: Working Of DC Motor**

### 7.1.1 BACK OR COUNTER EMF:

When the armature of a d.c. motor rotates under the influence of the driving torque, the armature conductors move through the magnetic field and hence an e.m.f. is induced in them. The induced e.m.f. acts in opposite direction to the applied voltage V(Lenz's law) and is known as back or counter e.m.f. Eb.

### 7.1.2 SIGNIFICANCE OF BACK EMF:

The presence of back e.m.f. makes the d.c. motor a self-regulating machine i.e., it makes the motor to draw as much armature current as is just sufficient to develop the torque required by the load. Back e.m.f. in a d.c. motor regulates the flow of armature current i.e., it automatically changes the armature current to meet the load requirement.

## 7.2 PARTS OF A DC MOTOR:

A simple motor has six parts

- Armature/ Rotor
- Commutator
- Brushes
- Axle
- Permanent Magnet
- DC Power supply

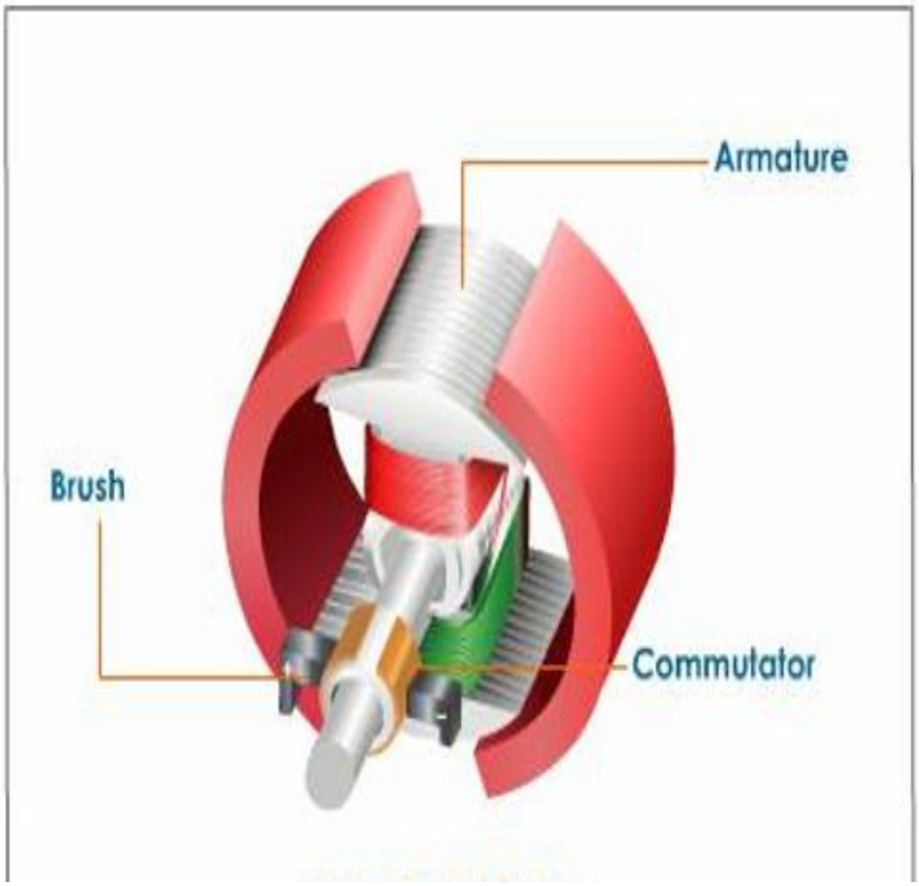


**Fig 7.3: Parts of a DC Motor**

## 7.3 RELAYS:

A relay is an electrically controllable switch widely used in industrial controls, automobiles and appliances.

The relay allows the isolation of two separate sections of a system with two different voltage sources i.e., a small amount of voltage/current on one side can handle a large amount of voltage/current on the other side but there is no chance that these two voltages mix up.

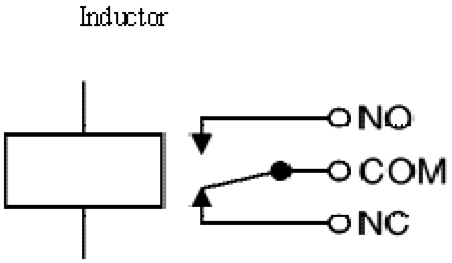


**Fig 7.4: Circuit symbol of a relay**

### 7.3.1Operation:

When current flows through the coil, a magnetic field is created around the coil i.e., the coil is energized. This causes the armature to be attracted to the coil. The armature’s contact acts like a switch and closes or opens the circuit. When the coil is not energized, a spring pulls the armature to its normal state of open or closed. There are all types of relays for all kinds of applications.

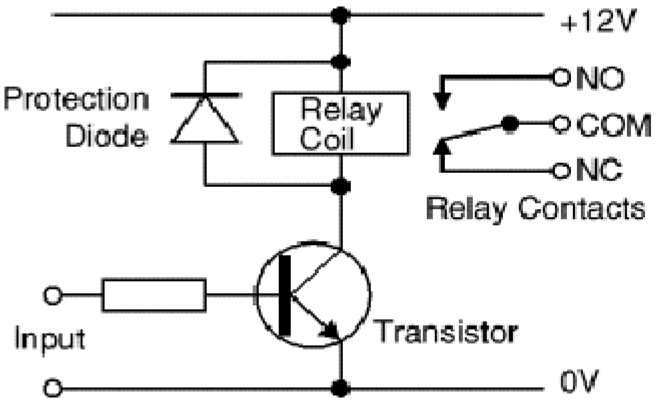


**Fig 7.5: Relay Operation and use of protection diodes.**

Transistors and ICs must be protected from the brief high voltage 'spike' produced when the relay coil is switched off. The above diagram shows how a signal diode (eg 1N4148) is connected across the relay coil to provide this protection. The diode is connected 'backwards' so that it will normally not conduct. Conduction occurs only when the relay coil is switched off, at this moment the current tries to flow continuously through the coil and it is safely diverted through the diode. Without the diode no current could flow and the coil would produce a damaging high voltage 'spike' in its attempt to keep the current flowing.

In choosing a relay, the following characteristics need to be considered:

1. The contacts can be normally open (NO) or normally closed (NC). In the NC type, the contacts are closed when the coil is not energized. In the NO type, the contacts are closed when the coil is energized.

2. There can be one or more contacts. i.e., different types like SPST (single pole single throw), SPDT (single pole double throw) and DPDT (double pole double throw) relays.

3. The voltage and current required to energize the coil. The voltage can vary from a few volts to 50 volts, while the current can be from a few milliamps to 20milliamps. The relay has a minimum voltage, below which the coil will not be energized. This minimum voltage is called the “pull-in” voltage.

4. The minimum DC/AC voltage and current that can be handled by the contacts. This is in the range of a few volts to hundreds of volts, while the current can be from a few amps to 40A or more, depending on the relay.

### 7.3.2 Driving a relay:

An SPDT relay consists of five pins, two for the magnetic coil, one as the common terminal and the last pins as normally connected pin and normally closed pin. When the current flows through this coil, the coil gets energized. Initially when the coil is not energized, there will be a connection between the common terminal and normally closed pin. But when the coil is energized, this connection breaks and a new connection between the common terminal and normally open pin will be established. Thus when there is an input from the microcontroller to the relay, the relay will be switched on. Thus when the relay is on, it can drive the loads connected between the common terminal and normally open pin. Therefore, the relay takes 5V from the microcontroller and drives the loads which consume high currents. Thus the relay acts as an isolation device.

Digital systems and microcontroller pins lack sufficient current to drive the relay. While the relay's coil needs around 10milli amps to be energized, the microcontroller's pin can provide a maximum of 1-2milli amps current. For this reason, a driver such as ULN2003 or a power transistor is placed in between the microcontroller and the relay. In order to operate more than one relay, ULN2003 can be connected between relay and microcontroller.

## 7.4 DRIVER CIRCUIT:

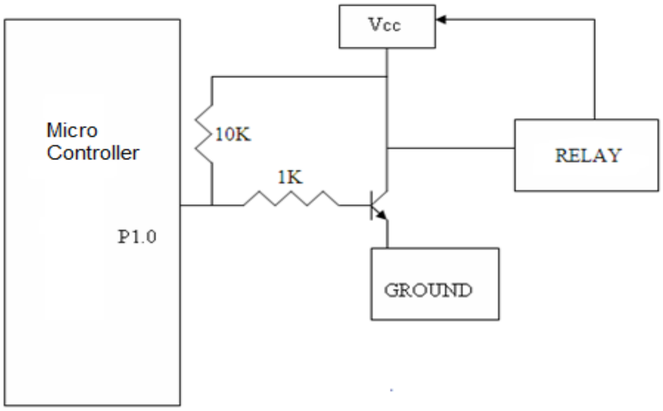


**Fig 7.6 Driver circuit**

The input to the base of the transistor is applied from the microcontroller port pin P1.0. The transistor will be switched on when the base to emitter voltage is greater than 0.7V (cut-in voltage). Thus when the voltage applied to the pin P1.0 is high i.e., P1.0=1 (>0.7V), the transistor will be switched on and thus the relay will be ON and the load will be operated.

When the voltage at the pin P1.0 is low i.e., P1.0=0 (<0.7V) the transistor will be in off state and the relay will be OFF. Thus the transistor acts like a current driver to operate the relay accordingly.

## Relay interfacing with the microcontroller:

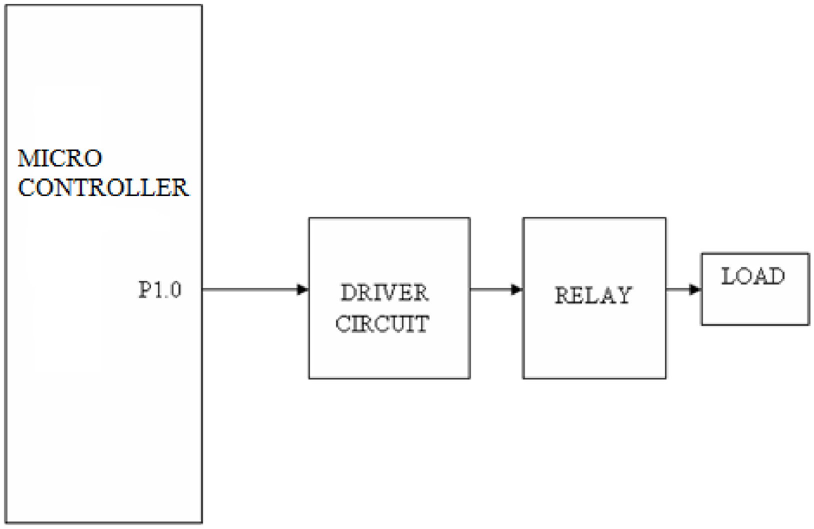


**Fig 7.7: Relay interfacing with micro controller.**

# CHAPTER-8

# ARDUINO IDE

## 8.1 ARDUINO IDE:

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.

## 8.2 WRITING SKETCHES:

**Verify**

Checks your code for errors compiling it.

**Upload**

Compiles your code and uploads it to the configured board.

See uploading below for details.

**New**

Creates a new sketch.

**Open**

Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content..

**Save**

Saves your sketch.

**Serial Monitor**

Opens the serial monitor.

## 8.2.1. FILE:

**New**

Creates a new instance of the editor, with the bare minimum structure of a sketch already in place.

**Open**

Allows to load a sketch file browsing through the computer drives and folders.

**Open Recent**

Provides a short list of the most recent sketches, ready to be opened.

**Sketchbook**

Shows the current sketches within the sketchbook folder structure; clicking on any name opens the corresponding sketch in a new editor instance.

**Examples**

Any example provided by the Arduino Software (IDE) or library shows up in this menu item. All the examples are structured in a tree that allows easy access by topic or library.

**Close**

Closes the instance of the Arduino Software from which it is clicked.

**Save**

Saves the sketch with the current name. If the file hasn't been named before, a name will be provided in a "Save as.." window.

**Save as…**

Allows to save the current sketch with a different name.

**Page Setup**

It shows the Page Setup window for printing.

**Print**

Sends the current sketch to the printer according to the settings defined in Page Setup.

**Preferences**

Opens the Preferences window where some settings of the IDE may be customized, as the language of the IDE interface.

**Quit**

Closes all IDE windows. The same sketches open when Quit was chosen will be automatically reopened the next time you start the IDE.

### 8.2.2. EDIT:

**Undo/Redo**

Goes back of one or more steps you did while editing; when you go back, you may go forward with Redo.

**Cut**

Removes the selected text from the editor and places it into the clipboard.

**Copy**

Duplicates the selected text in the editor and places it into the clipboard.

**Copy for Forum**

Copies the code of your sketch to the clipboard in a form suitable for posting to the forum, complete with syntax coloring.

**Copy as HTML**

Copies the code of your sketch to the clipboard as HTML, suitable for embedding in web pages.

**Paste**

Puts the contents of the clipboard at the cursor position, in the editor.

**Select All**

Selects and highlights the whole content of the editor.

**Comment/Uncomment**

Puts or removes the // comment marker at the beginning of each selected line.

**Increase/DecreaseIndent**

Adds or subtracts a space at the beginning of each selected line, moving the text one space on the right or eliminating a space at the beginning.

**Find**

Opens the Find and Replace window where you can specify text to search inside the current sketch according to several options.

**Find Next**

Highlights the next occurrence - if any - of the string specified as the search item in the Find window, relative to the cursor position.

**Find Previous**

Highlights the previous occurrence - if any - of the string specified as the search item in the Find window relative to the cursor position.

### 8.2.3 SKETCH:

**Verify/Compile**

Checks your sketch for errors compiling it; it will report memory usage for code and variables in the console area.

**Upload**

Compiles and loads the binary file onto the configured board through the configured Port

.**UploadUsingProgrammer**

This will overwrite the bootloader on the board; you will need to use Tools > Burn Bootloader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so a Tools -> Burn Bootloader command must be executed.

**Export Compiled Binary**

Saves a .hex file that may be kept as archive or sent to the board using other tools.

**Show Sketch Folder**

Opens the current sketch folder.

**Include Library**

Adds a library to your sketch by inserting #include statements at the start of your code. For more details, see libraries below. Additionally, from this menu item you can access the Library Manager and import new libraries from .zip files

.**Add File**

Adds a source file to the sketch (it will be copied from its current location). The new file appears in a new tab in the sketch window. Files can be removed from the sketch using the tab menu accessible clicking on the small triangle icon below the serial monitor one on the right side of the toolbar.

## 8.3 Tools:

**Archive Sketch**

Archives a copy of the current sketch in .zip format. The archive is placed in the same directory as the sketch.

**Fix Encoding & Reload**

Fixes possible discrepancies between the editor char map encoding and other operating systems char maps.

**Serial Monitor**

Opens the serial monitor window and initiates the exchange of data with any connected board on the currently selected Port. This usually resets the board, if the board supports Reset over serial port opening.

**Board**

Select the board that you're using. See below for descriptions of the various boards.

**Port**

This menu contains all the serial devices (real or virtual) on your machine. It should automatically refresh every time you open the top-level tools menu.

**Programmer**

For selecting a hardware programmer when programming a board or chip and not using the onboard USB-serial connection. Normally you won't need this, but if you're burning a bootloader to a new microcontroller, you will use this.

**BurnBootloader**

The items in this menu allow you to burn a bootloader onto the microcontroller on an Arduino board. This is not required for normal use of an Arduino or Genuino board but is useful if you purchase a new ATMEGA microcontroller (which normally come without a bootloader). Ensure that you've selected the correct board from the Boards menu before burning the bootloader on the target board. This command also set the right fuses.

## 8.4 SKETCH BOOK:

The Arduino Software (IDE) uses the concept of a sketchbook: a standard place to store your programs (or sketches). The sketches in your sketchbook can be opened from the File > Sketchbook menu or from the Open button on the toolbar. The first time you run the Arduino software, it will automatically create a directory for your sketchbook. You can view or change the location of the sketchbook location from with the Preferences dialog.

## 8.5 LIBRARIES:

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from the Sketch > Import Library menu. This will insert one or more #include statements at the top of the sketch and compile the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its #includestatements from the top of your code.

## 8.6 SERIAL MONITOR:

Displays serial data being sent from the Arduino or Genuino board (USB or serial board). To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down that matches the rate passed to Serial.begin in your sketch. Note that on Windows, Mac or Linux, the Arduino or Genuino board will reset (rerun your sketch execution to the beginning) when you connect with the serial monitor.

# CHAPTER-9
# THINGSPEAK AND BLYNK

## 9.1 INTRODUCTION:

ThingSpeak is an open source Internet of Things (IoT) application and API to store and retrieve data from things using the HTTP protocol over the Internet or via a Local Area Network. ThingSpeak enables the creation of sensor logging applications, location tracking applications, and a social network of things with status updates.

ThingSpeak was originally launched by ioBridge in 2010 as a service in support of IoT applications.

ThingSpeak has integrated support from the numerical computing software MATLAB from MathWorks. Allowing ThingSpeak users to analyze and visualize uploaded data using Matlab without requiring the purchase of a Matlab license from Mathworks.

ThingSpeak acts as the IoT platform for data collection and analytics that serves as a bridge connecting edge node devices such as temperature and pressure sensors to collect data and data exploratory analysis software to analyze data. ThingSpeak serves as the data collector which collects data from edge node devices and also enables the data to be pulled into a software environment for historical analysis of data.

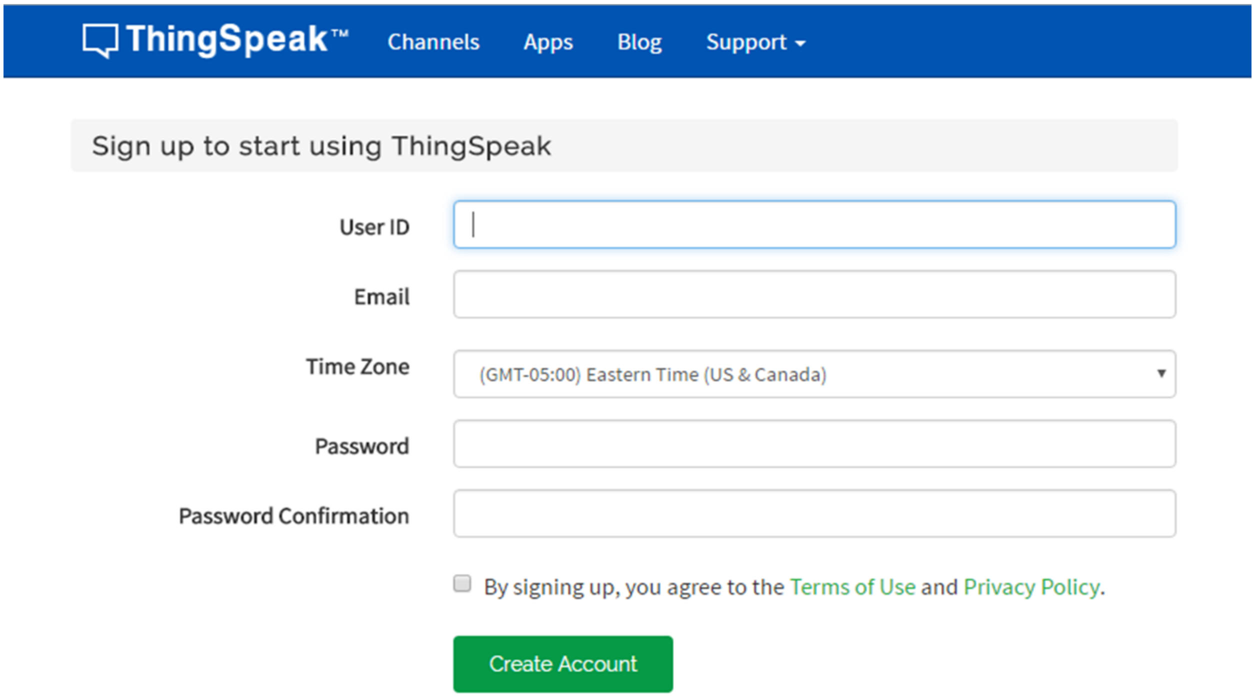
## 9.2 THINGSPEAK FUNCTIONALITY:

The primary element of ThingSpeak activity is the channel, which contains data fields, location fields, and a status field. After you create a ThingSpeak channel, you can write data to the channel, process and view the data with MATLAB® code, and react to the data with tweets and other alerts.

The typical ThingSpeak workflow :

- Create a Channel and collect data
- Analyze and Visualize the data
- Act on the data using any of several Apps

**THINGSPEAK SIGNUP PAGE:**



Sign up to start using ThingSpeak

Time Zone (GMT-05:00) Eastern Time (US & Canada) ▾

□ By signing up, you agree to the Terms of Use and Privacy Policy.

**Fig 9.1: ThingSpeak singup page**

## 9.3 INTRODUCTION OF BLYNK:

Blynk was designed for the Internet of Things. It can control hardware remotely, it can display sensor data, it can store data, vizualize it and do many other cool things.

There are three major components in the platform:

- Blynk App - allows to you create amazing interfaces for your projects using various widgets we provide.
- Blynk Server - responsible for all the communications between the smartphone and hardware. You can use our Blynk Cloud or run your private Blynk server locally. It’s open-source, could easily handle thousands of devices and can even be launched on a Raspberry Pi.
- Blynk Libraries - for all the popular hardware platforms - enable communication with the server and process all the incoming and outcoming commands.

## 9.4 FEATURES :

- Similar API & UI for all supported hardware & devices
- Connection to the cloud using:
  - Ethernet
  - WiFi
  - Bluetooth and BLE
  - USB (Serial)
  - …
- Set of easy-to-use Widgets
- Direct pin manipulation with no code writing
- Easy to integrate and add new functionality using virtual pins
- History data monitoring via History Graph widget
- Device-to-Device communication using Bridge Widget
- Sending emails, tweets, push notifications, etc.

## 9.5 CREATE A BLYNK ACCOUNT:

After you download the Blynk App, you'll need to create a New Blynk account. This account is separate from the accounts used for the Blynk Forums, in case you already have one.

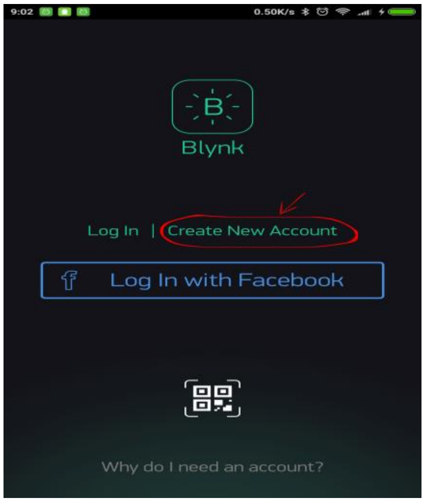


**Fig 9.2: Create a Blynk Account**

## 9.6 CREATE A NEW PROJECT:

After you’ve successfully logged into your account, start by creating a new project**.**

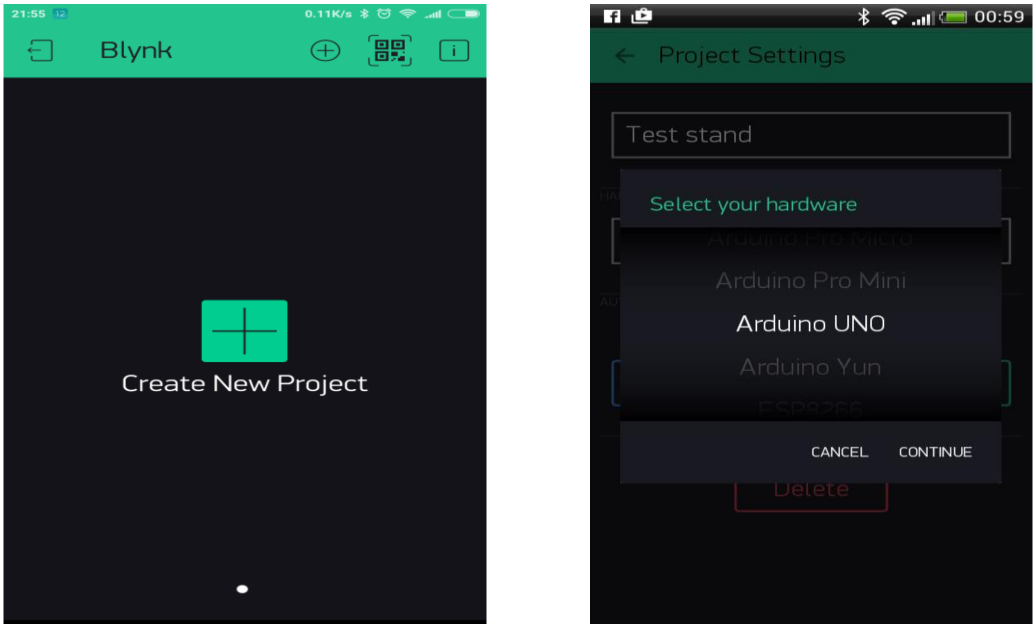
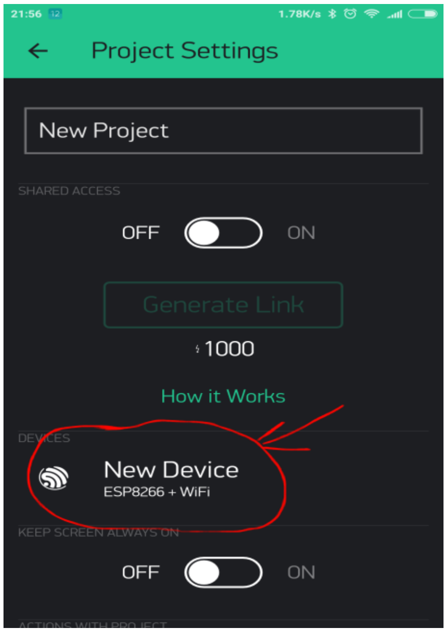


**Fig 9.3: Create a new project.**

## 9.7 AUTH TOKEN:

Auth Token is a unique identifier which is needed to connect your hardware to your smartphone. Every new project you create will have its own Auth Token. You’ll get Auth Token automatically on your email after project creation. You can also copy it manually.

**Click on devices section :**



New Project

SHARED ACCESS

OFF ON

1000

DEVICES

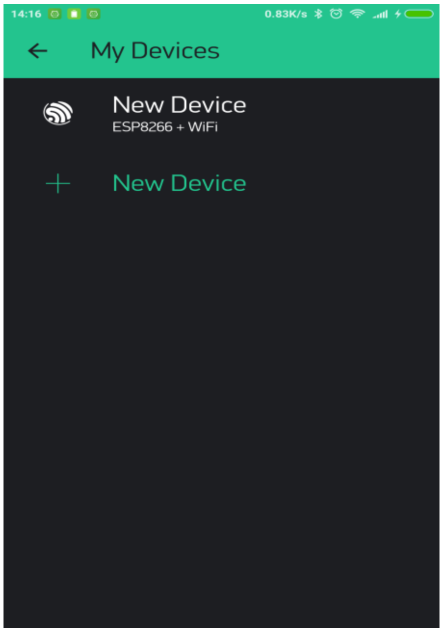
New Device
ESP8266 + WiFi

KEEP SCREEN ALWAYS ON

OFF ON

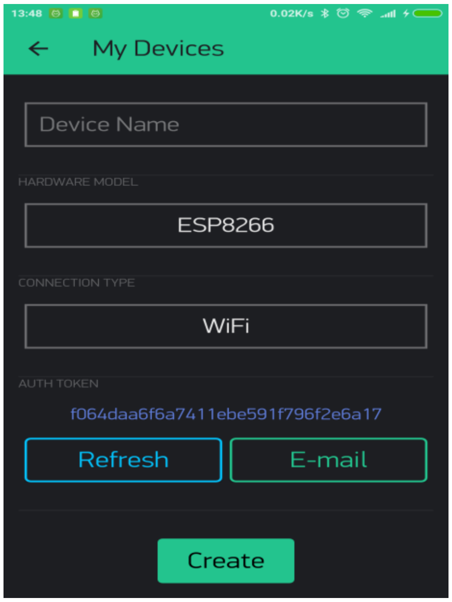
ACTIONS WITH PROJECT

**Click on device :**



New Device
ESP8266 + WiFi

**You will see token :**



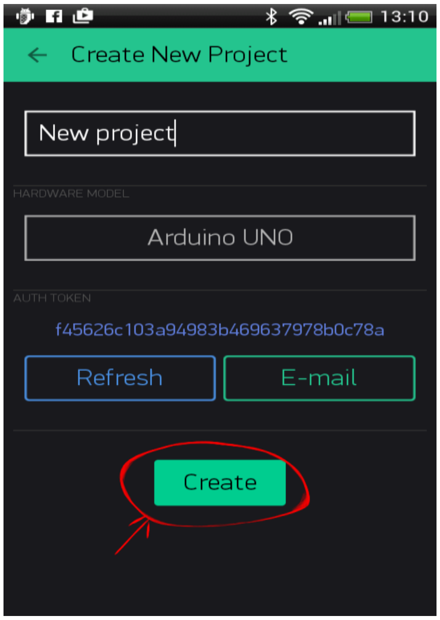
Device Name

HARDWARE MODEL
ESP8266

CONNECTION TYPE
WiFi

AUTH TOKEN
f064daa6f6a7411ebe591f796f2e6a17

**press the “Create” button:**



New project

HARDWARE MODEL
Arduino UNO

AUTH TOKEN
f45626c103a94983b469637978b0c78a

## 9.8 ADD A WIDGET :

Tap anywhere on the canvas to open the widget box. All the available widgets are located here. Now pick a button.

**Drag-n-Drop** - Tap and hold the Widget to drag it to the new position.

**Widget box:**

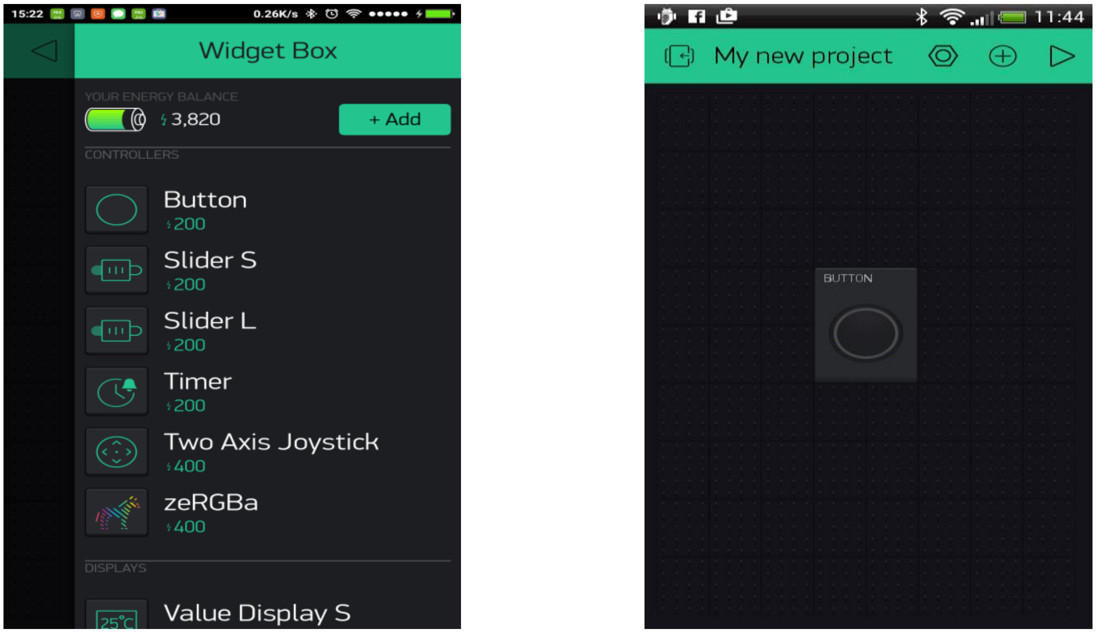


**Fig 9.4:Drag-n-Drop**.

**Widget Settings** - Each Widget has it's own settings. Tap on the widget to get to them.

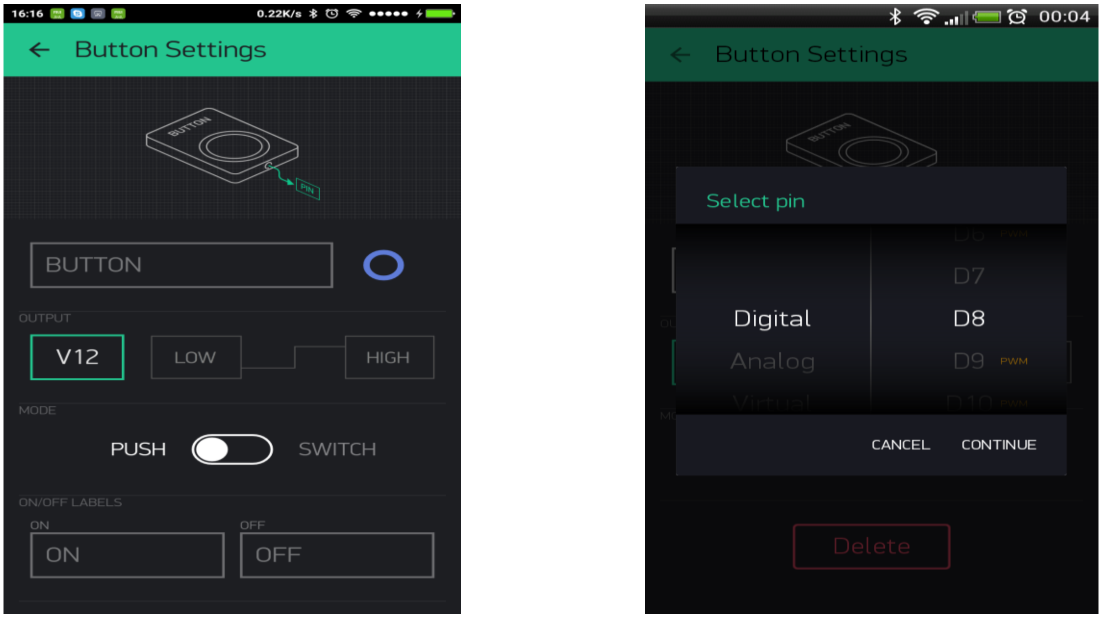


**Fig 9.5: Widget Settings.**

## 9.9 RUN THE PROJECT:

When you are done with the Settings - press the PLAY button. This will switch you from EDIT mode to PLAY mode where you can interact with the hardware.

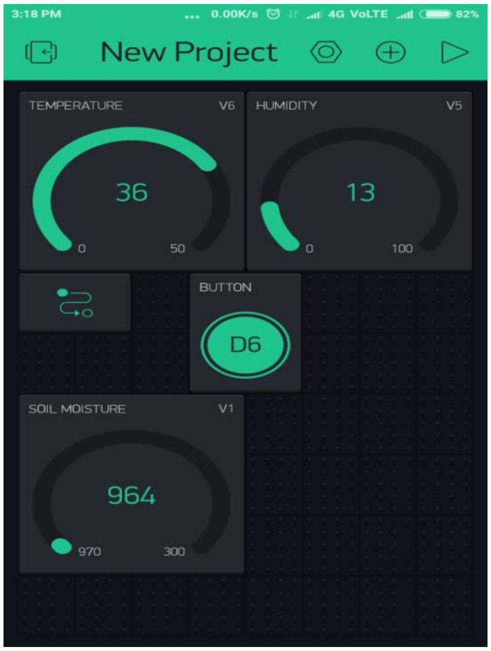


**Fig 9.6: Run the Project.**

# CHAPTER-10

# CIRCUIT DIAGRAM AND WORKING

## 10.1. Circuit diagram:

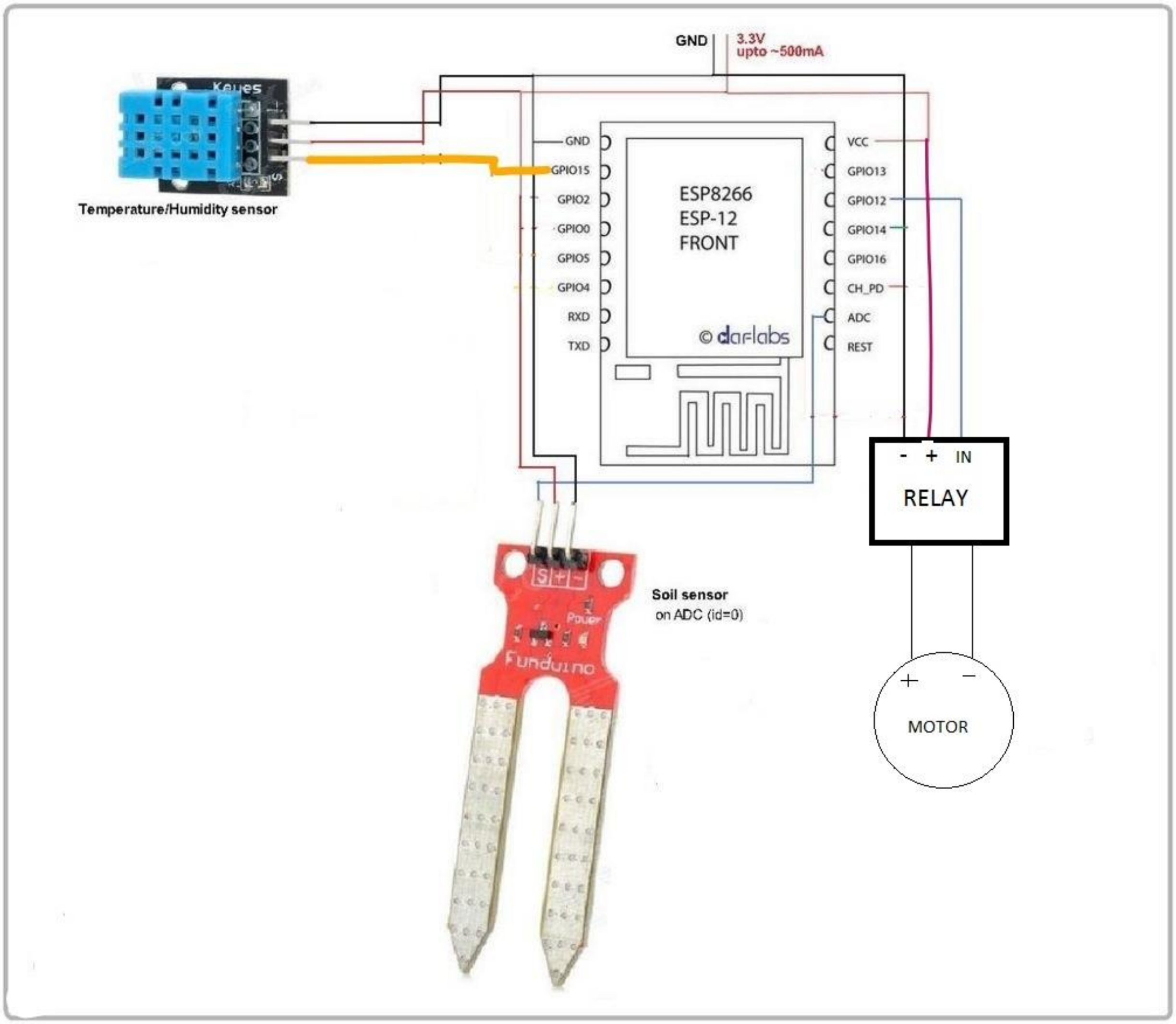


**Fig 10.1: Circuit block diagram for Automated Irrigation System in Agriculture using IOT**

## 10.2. CONNECTIONS:

### DHT 11 Connections:

Positive terminal (+) is connected to 3.3V pin of NodeMCU.

Negative terminal (-) is connected to gnd pin of NodeMCU.

Sensor terminal(S) is connected to GPIO 15 pin of NodeMCU.

### Soil Moisture Sensor Connections:

Pin 1 A0 is connected to A0pin of NodeMCU.

Pin 3 Gnd pin is connected to Gnd pin of NodeMCU.

Pin 4 Vcc pin is connected to 3.3V pin of NodeMCU.

### Relay and Motor Connections:

Vcc pin of relay is connected to 3.3 pin of NodeMCU.

Gnd pin of relay is connected to Gnd pin of NodeMCU.

IN1 pin of relay is connected to GPIO 12 pin of NodeMCU.

Relay output is given to motor with serial connection of 12V battery to motor.

## 10.3. WORKING:

The board can be supplied with power either the USB connector (**5V**), or the VIN pin of the board (**9V**). Supplying voltage via the **5V** or **3.3V** pins bypasses the regulator, and can damage the board.

The DHT 11 sensor and soil moisture sensor senses the temperature , humidity and soil moisture level in soil respectively and sends them to NodeMCU board. The microcontroller on the board takes these values as inputs and executes the program. ESP8266 acts as wifi module and connects our system to internet through a set of commands.

The network to which our module is to be connected is indicated with the help of SSID and password written in the arduino sketch (code).Therefore, our wifi module connects to the specified network.

Now, we need to open our Blynk account on our smart phone to see the output. It shows output in graphical form of meter gauge. In our project, we will get three graphs with field1 as temperature in Celsius, field2 as humidity in Relative humidity and field3 as soil moisture level. There will be another push button graphic on mobile labeled as motor. We can control the motor by switching that push button on and off.

In this way, our project helps the farmers to monitor the condition of crop and control equipment from anywhere in the world using the concept of Internet of Things.

## 10.4 FLOW CHART:

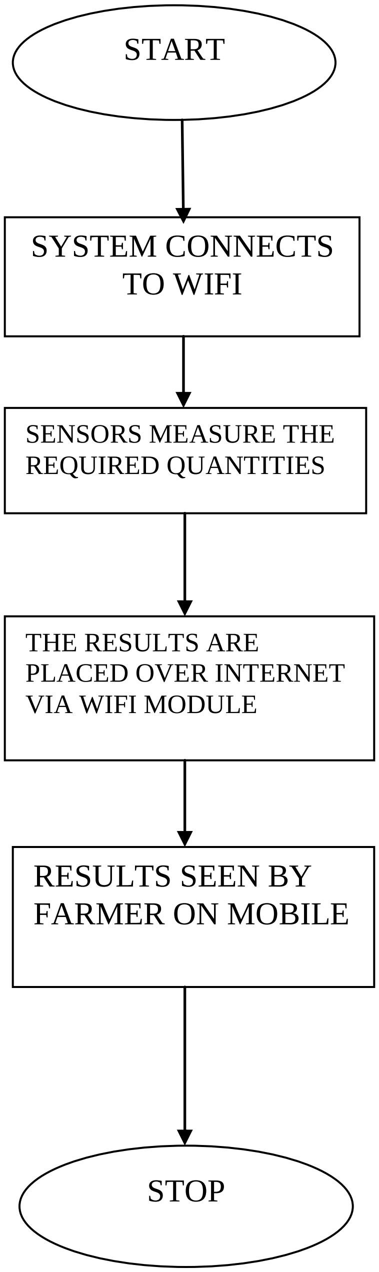

START
SYSTEM CONNECTS TO WIFI
SENSORS MEASURE THE REQUIRED QUANTITIES
THE RESULTS ARE PLACED OVER INTERNET VIA WIFI MODULE
RESULTS SEEN BY FARMER ON MOBILE
STOP

**Fig.10.2: Flow chart**

# CHAPTER-11

# PROJECT SNAPSHOTS

## 11.1. SYSTEM FOR AUTOMATED IRRIGATION SYSTEM:

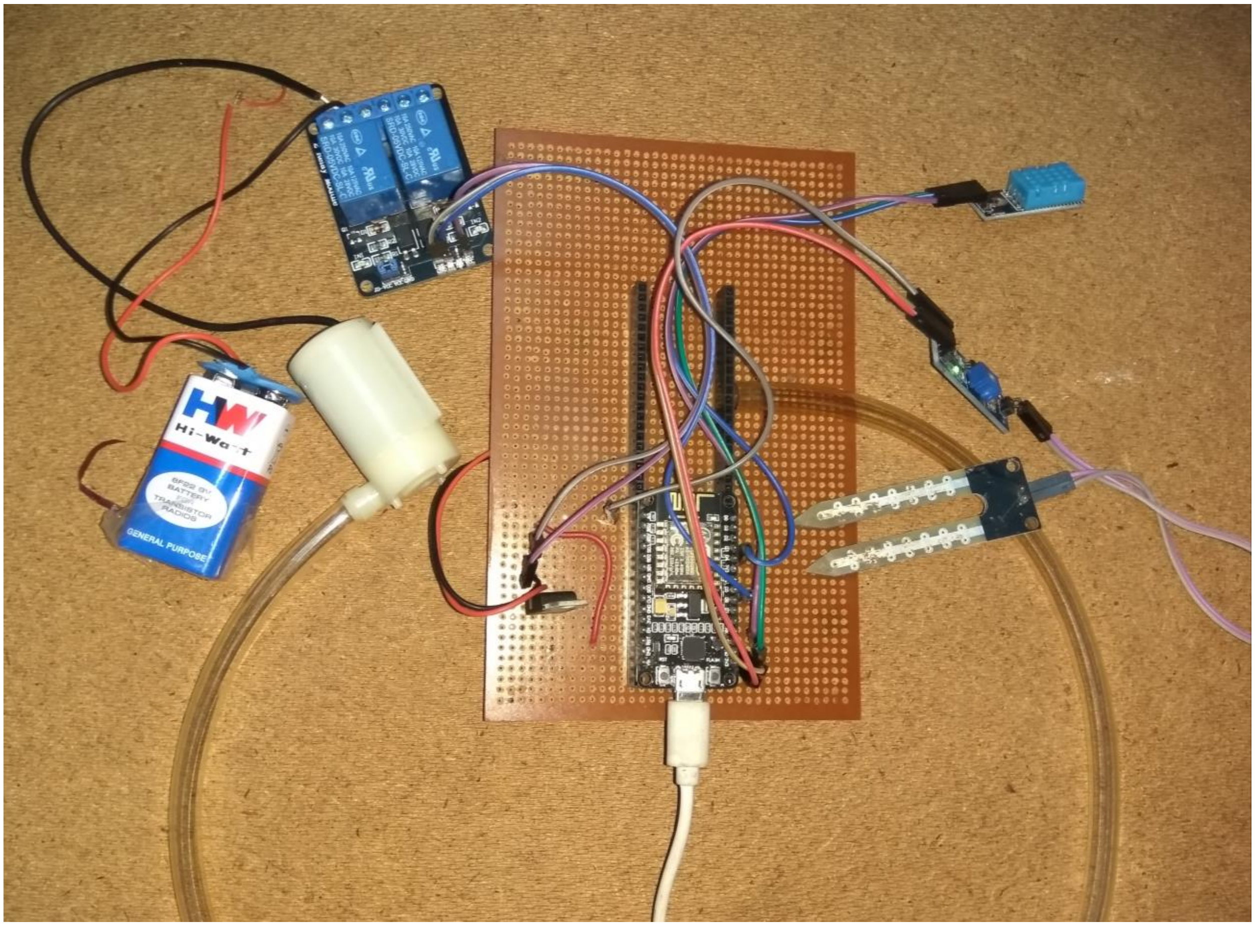


**Fig 11.1: System for Automated Irrigation System**

## 11.2 OUTPUT DISPLAYED ON MOBILE PHONE BLYNK APP:

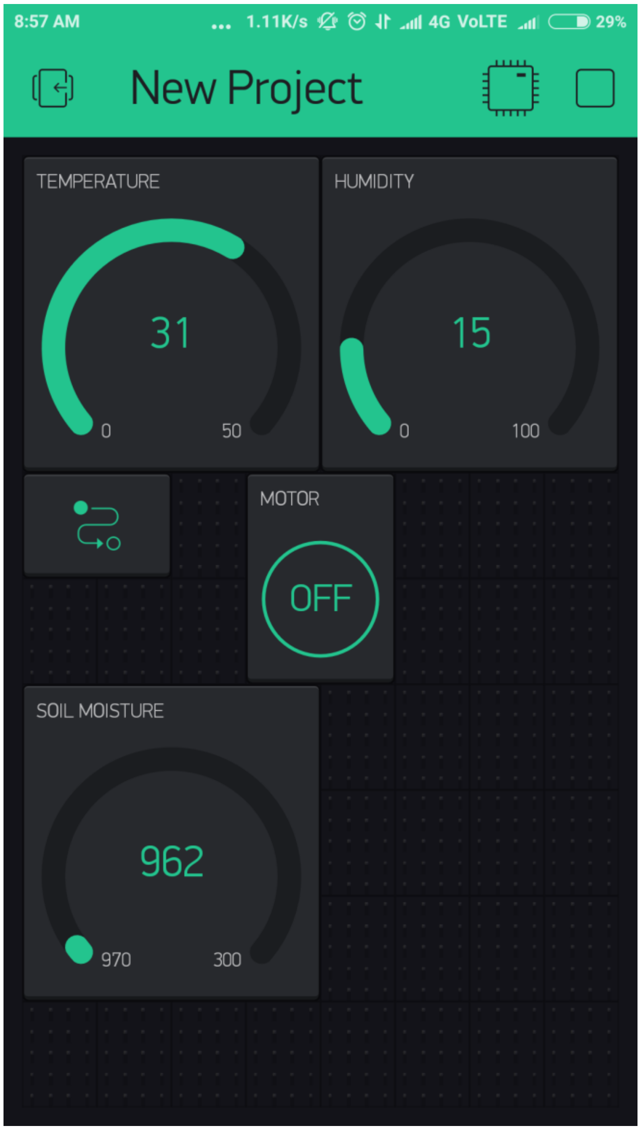


**Fig 11.2: Output displayed on Mobile Phone Using Blynk App**

# CHAPTER-12

# CONCLUSION

## 12.1 CONCLUSION:

The proposed model is to develop a reliable automated agriculture system so that the farmers can easily handle the cultivation. In this work we present an IOT based smart irrigation system that can provide real time online information on their PC about crop condition and they can take necessary action from their devices through IOT.

The device works by taking readings from various sensors at different pins in nodeMCU microcontroller and it will also give outputs to sprinklers. The receiver unit also has a duplex communication link based on a Internet interface. The Internet connection allows the data inspection in real time on a website, where the soil-moisture and temperature levels are graphically displayed through an application interface and farmers can control the sprinklers through the application interface website.

This project develops agriculture field and increase the growth of food production. This can also useful for increasing the economy and demand of food necessity.

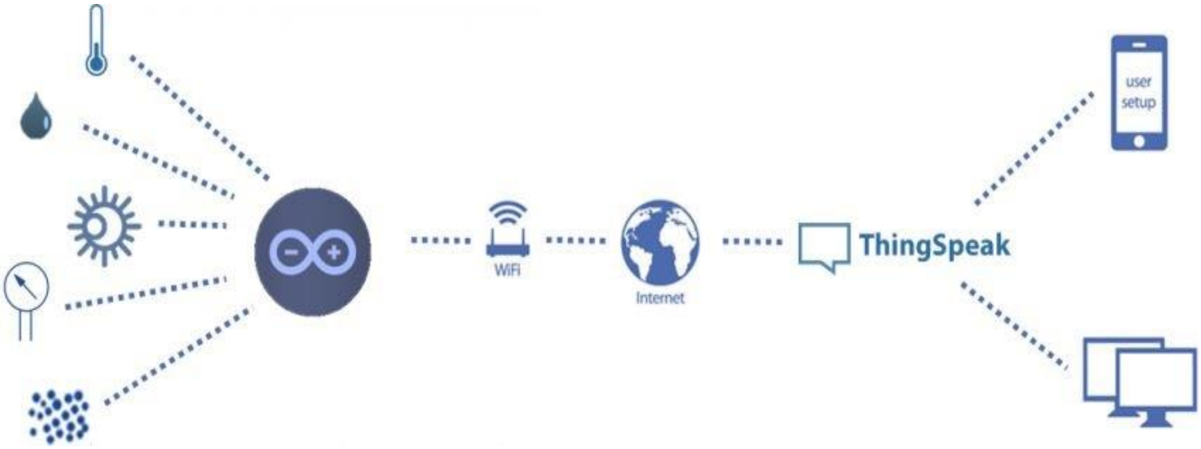


**Fig 12.1: Overview of the project**

## 12.2 REFERENCES:

- Joaquín Gutiérrez, Juan Francisco Villa-Medina, Alejandra NietoGaribay, and Miguel Ángel Porta- Gándara "Automated Irrigation System Using a Wireless Sensor Network and GPRS Module " IEEE 2013.
- Samy Sadeky, Ayoub Al-Hamadiy, Bernd Michaelisy, Usama Sayedz," An Acoustic Method for Soil Moisture Measurement ", IEEE 2004.
- Jia Uddin, S.M. Taslim Reza, Qader Newaz, Jamal Uddin, Touhidul Islam, and Jong-Myon Kim,"Automated Irrigation System Using Solar Power" ©2012 IEEE.
- Zhang Feng Yulin University Yulin University tfnew21@sina.com, " Research on water-saving irrigation automatic control system based on Internet of things Institute of Information Technology", 2011 IEEE.
- Awati J.S., Patil V.S., "Automatic Irrigation Control by using wireless sensor networks", Journal of Exclusive Management Science - June 2012.
- Shaohua Wan, "Research on the Model for Crop Water Requirements in Wireless Sensor Networks", 2012 International Conference on Management of e-Commerce and e-Government.
- Internet Of Things by K. Ashton and R. van Kranenburg .
- "SAW temperature sensor and remote reading system" Bao W. Burkhard V. V. Varadan and V. K. Varadan
- Arduino for agriculture applications by S Ferdoush, X Li - Procedia Computer Science.
- Soil moisture sensors by Edwin T.Engman, Narinder Chauhan.
- nodeMCU ESP8266 by Z. Tafa and R. Stojanovic