CHAPTER III

MATERIALS AND METHODS

A field experiment was conducted to study "Development of smart system for runoff measurement" during monsoon season (October - November) 2019. The details of experimental materials used, procedures followed and techniques adopted during the present investigation are presented in this chapter.

3.1 DESCRIPTION OF STUDY AREA

3.1.1 Location

The experiment was conducted at Instructional Farm, Department of Soil and Water Engineering, College of Agricultural Engineering and Technology, JAU, Junagadh. It is located between 69.40° to 71.05° East Longitude and 20.44° to 21.40° North Latitude. It was at an elevation of 83 m above MSL (Mean Sea Level) on the western side foothills of mount Girnar. The location is situated in the South Saurashtra agro-climatic zone of Gujarat state.

3.1.2 Climate

The study area is having typically subtropical and semi-arid climate, characterized by fairly cold and dry winter, hot and dry summer and warm and moderately humid during monsoon. The partial failure of monsoon once in three to four years is common in this region. Winter sets in November and continues until the end of February. January is the coldest month of winter. Summer commences in the second fortnight of February and ends in the middle of June. April and May are the hottest months of summer. The monsoon season runs between June to October. The last 35-year weather data recorded at the Junagadh Agricultural University meteorological observatory located near to the experimental site showed that the variation in the weekly mean of daily maximum temperature, minimum temperature, relative humidity, wind speed, bright sunshine hours and pan evaporation were 29.5 "C to 39.4 °C, 10 °C to 26.7 °C, 51 % to 81 %, 10.1 km/h, 4.2 to 13.4 hours and 3.6 to 10.7 mm, respectively. The average annual rainfall and evaporation are 937 mm and 2482 mm, respectively.

3.2 FIELD EXPERIMENTAL DETAILS

3.2.1. Runoff plot

The experiment was conducted by selecting existing runoff plot at Instructional Farm, College of Agricultural Engineering and Technology, JAU, Junagadh. All the boundary walls of beds of runoff plot and collecting tanks were constructed by 0.23 m Bella stone with 20 mm plaster. At the end of each runoff bed runoff guiding masonry ways having 1/6h width of runoff bed for guiding runoff water from runoff bed to runoff collecting tank were constructed. Therefore the runoff diverted from runoff bed n collecting tank was considered 1/6" of total runoff from each bed. The runoff tanks and waterways were covered with a metallic sheet to prevent addition of direct rainfall into actual runoff water coming from bed. The dimensions of the runoff plot given as per following detail and the runoff plot are shown.

Fig. 3.1 Runoff plot

Fig. 3.2 Dimension measurement of a runoff collection tank

- **Specification of runoff plot:**

Number of bed = 13

Length of each bed= 22.0 m

Width of each bed= 2.5 m.

Area of each bed = 55 m^2.

- **Specification of runoff collection tank :**

Number of collection tank = 13

Length of collection tank= 3.25 m

Width of collection tank = 1.7 m.

Height of collection tank = 0.75 m.

Area of each collection tank = 5.525 m^2.

3.2.2. MEASUREMENT OF RUNOFF

The treatment wise runoff can be measured after each runoff event. The runoff tank collects 1/6thof runoff generated from each bed and total runoff from bed Q_i,

$Q_i = a_T \times \Delta h \times 6$ ……….(3.1)

[Because 1/6"part of runoff diverted into tank]

V = Total Runoff $= \sum_{i=1}^{n} (\Delta v_i)$ ……….(3.2)

Where,

V = Cumulative Runoff from bed, m^3

i = An index for the time

n = Total number of sub-periods of storm event.

a_T = Bottom area of tank, m^2

Δh = Rise in water level in tank after runoff event, m

The water level in each tank during runoff event was measured by measuring tape as well as by developed smart system at 10 minutes interval.

The runoff depth was computed using the Eqn. as below.

$$RO_i = 1000 * \frac{\Delta V_i}{A}$$

Where, RO_i is the runoff depth during ith sub-period of rainfall event, A is the area of the runoff plot and ΔV_i is the incremental runoff volume collected during ith sub-period of rainfall event.

The discharge flow rate(Q_i) during ith sub-period of the rainfall event can be calculated as below.

$$Q_i = \frac{\Delta V_i}{\Delta t}$$

Where, Δt is the duration of runoff collection(seconds)

3.3 EXPERIMENTAL MATERIAL

The resources and materials used for development and performance evaluation of smart runoff measurement system during the experiment are given below.

3.3.1 Smart Runoff Measurement System Hardware

3.3.1.1 Arduino Mega 2560

Arduino is an open-source computer hardware and software company, project and user community that designs and manufacture single-board microcontrollers and microcontrollers kit for building digital devices and interactive objects that can sense and control objects in the physical and digital world.

Arduino is an open-source physical computing platform based on a simple ilo board and a development environment that implements the Processing/Wiring language. Arduino can be used to develop stand-alone interactive objects or can be connected to software on your computer (e.g. Flash, Processing, MaxMSP). The open-source IDE can be downloaded for free (currently for Mac OS X, Windows, and Linux).

The Arduino Mega is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs 4 UARTS (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started. Never fear for accidental electrical discharge, either since the Mega also includes a plastic base plate to protect it.

The Mega 2560 R3 also adds SDA and SCL pins next to the AREE. Also, there are two new pins placed near the RESET pin, one is the IOREF that allows the shields to adapt to the voltage provided from the board The other is a not connected and is reserved for future purposes. The Mega 2560R3 Works with all existing shields but can adapt to new shields which use these additional pins.

Specification of Arduino Mega 2560

Microcontroller	: ATmega2560
Operating Voltage	: 5V
Input Voltage (recommended)	: 7-12V
Input Voltage (limit)	: 6-20V
Digital 1/0 Pins	: 54 (of which 15 provide PWM output)
Analog Input Pins	: 16
DC Current per 1/O Pin	: 20 mA
DC Current for 3.3V Pin	: 50 mA
Flash Memory	: 256 KB of which 8 KB used by boot loader
SRAM	: 8 KB
EEPROM	: 4 KB
Clock Speed	: 16 MHz
LED BUILTIN	: 13
Length	: 101.52 mm
Width	: 53.3 mm
Weight	: 37g

Arduino Mega 2560 Pin Configuration

The pin configuration of this Arduino mega 2560 board is shown below. Every pin of this board comes by a particular function which is allied with it. All analog pins of this board can be used as digital I/O pins. By using this board, the Arduino mega projected can be designed. These boards offer flexible work memory space is the more & processing power that permits to work with different types of sensors without delay. When we compare with other types of Arduino boards, these boards are physically superior.

- **Pin 3.3V & 5V**

These pins are used for providing o/p regulated voltage approximately 5v. This RPS(regulated power supply) provides the power to the microcontroller as well as other components which are used over the Arduino mega board. It can be attained from Vin-pin of the board or one more regulated voltage supply-5V otherwise USB cable, whereas another voltage regulation can be offered by 33v 0-pin. The maximum power can be drawn by this is 50mA.

- **GND Pin**

The Arduino Mega board includes 5-GND pins where one of these pins can be used whenever the project requires.

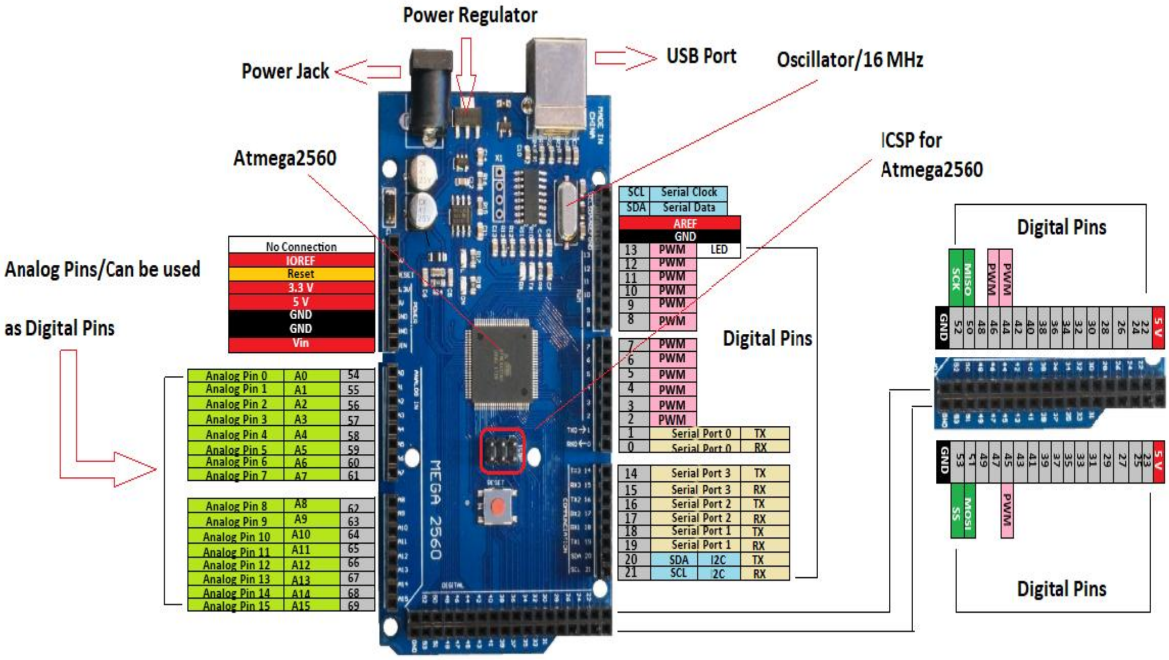


Fig. 3.3 Arduino Mega 2560

- **Reset (RST) Pin**

The RST pin of this board can be used for rearranging the board. The board can be rearranged by setting this pin to low.

- **Vin Pin**

The range of supplied input voltage to the board ranges from 7volts to 20volts. The voltage provided by the power jack can be accessed through this pin. However, the output voltage through this pin to the board will be automatically set up to 5V.

- **Serial Communication**

The serial pins of this board like TXD and RXD are used to transmit & receive the serial data. Tx indicates the transmission of information whereas the RX indicates receive data. The serial pins of this board have four combinations. For serial

0, it includes Tx (1) and Rx (0), for serial 1, it includes Tx(18) & Rx(19), for serial 2 it includes Tx(16) & Rx(17), and finally for serial 3, it includes Tx(14) & Rx(15).

- **External Interrupts**

The external interrupts can be formed by using 6-pins like interrupt 0(0), interrupt 1(3), interrupt 2(21), interrupt 3(20), interrupt 4(19), interrupt 5(18). These pins produce interrupts by a number of ways i.e. Providing LOW value, rising or falling edge or changing the value to the interrupt pins.

- **LED**

This Arduino board includes an LED and that is allied to pin-13 which is named as digital pin 13. This LED can be operated based on the high and low values of the pin. This will give you to modify the programming skills in real-time.

- **AREF**

The term AREF stands for Analog Reference Voltage which is a reference voltage for analog inputs

- **Analog Pins**

There are 16-analog pins included on the board which is marked as A0-A15. It is very important to know that all the analog pins on this board can be utilized like digital I/O pins. Every analog pin is accessible with the 10-bit resolution which can gauge from GND to 5 volts. But, the higher value can be altered using AREF pin as well as the function of analog Reference ().

- **I2C**

The I2C communication can be supported by two pins namely 20 & 21 where 20-pin signifies Serial Data Line (SDA) which is used for holding the data & 21-pin signifies Serial Clock Line (SCL) mostly utilized for offering data synchronization among the devices

- **SPI Communication**

The term SPI is a serial peripheral interface which is used to transmit the data among the controller & other components. Four pins like MISO (50), MOSI (51), SCK (52), and SS (53) are utilized for the communication of SPI

3.3.1.2 Wi-Fi Module

The NodeMCU (Node MicroController Unit) is an open-source software and hardware development environment that is built around a very inexpensive System-on-a-Chip (SoC) called the ESP8266. This small module used to allow a microcontroller to connect to a Wi-Fi network and make a simple TCP/IP connection using Hayes-style commands.

Specification of NodeMCU

Operating system	: XTOS
CPU	: ESP8266
Memory	: 128kBytes
Strorage	: 4MBytes
Power By	: USB
Power Voltage	: 3v 5v (used with 3.3v Regulator which inbuilt on Board using Pin VIN)
IDE Used	: Arduino IDE
GPIO	: 10

ESP8266 NodeMCU Pinout

The ESP8266 NodeMCU has total 30 pins that interface it to the outside world. The connections are as follows:

- **Power Pin**

There are four power pins viz. one VIN pin & three 3.3V pins. The VIN pin can be used to directly supply the ESP8266 and its peripherals if you have a regulated 5V voltage source. The 3.3V pins are the output of an on-board voltage regulator. These pins can be used to supply power to external components.

- **GND**

GND is a ground pin of the ESP8266 NodeMCU development board.

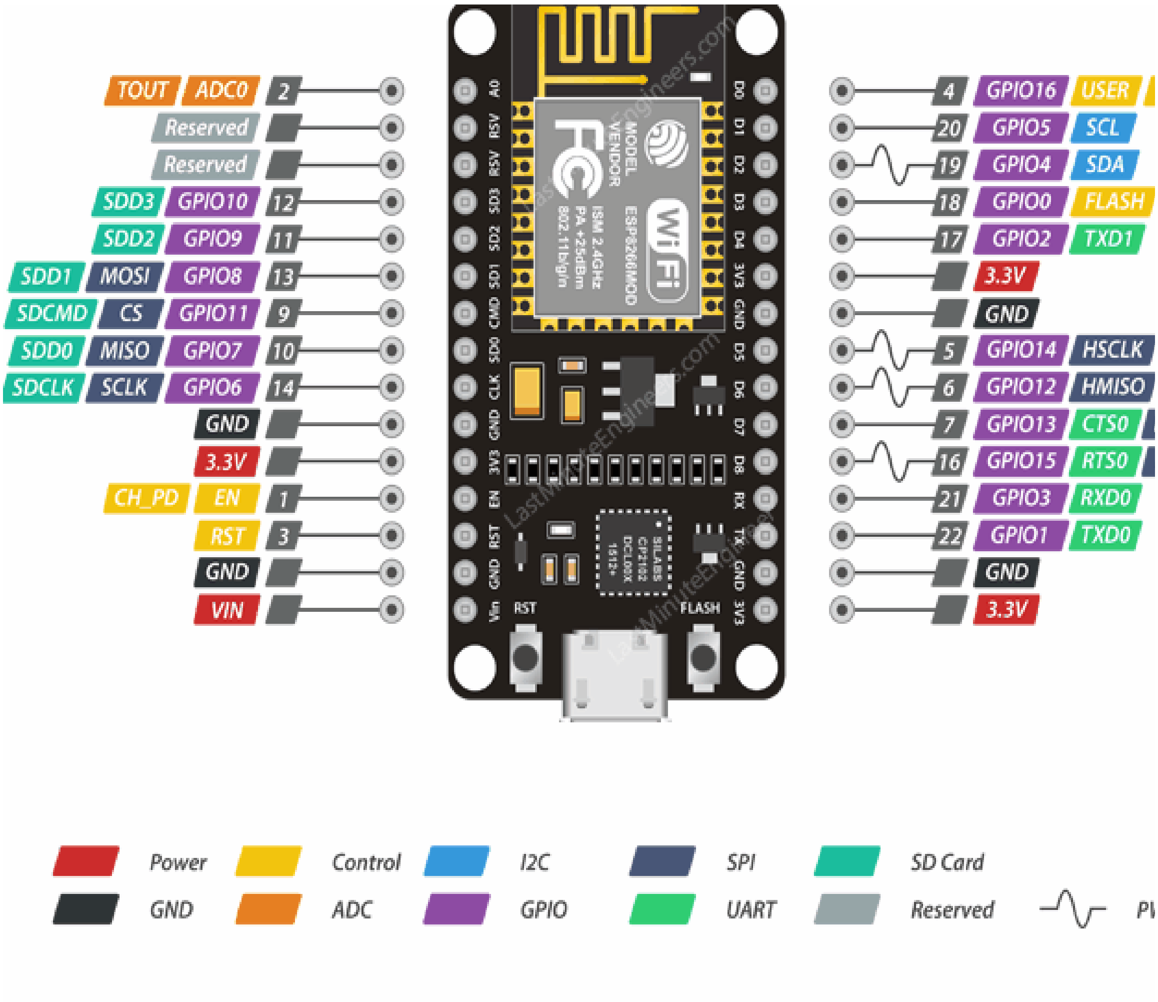


Fig. 3.4 NodeMCU

- **I2C Pin**

I2C pins are used to hook up all sorts of I2C sensors and peripherals in your project. Both I2C Master and I2C Slave are supported. I2C interface functionality can be realized programmatically, and the clock frequency is 100 kHz at a maximum. It should be noted that I2C clock frequency should be higher than the slowest clock frequency of the slave device.

- **GPIO Pins**

ESP8266 NodeMCU has 17 GPIO pins which can be assigned to various functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light and Button programmatically. Each digital-enabled GPIO can be configured to internal pull-up or pull-down, or set to high impedance. When configured as an input, it can also be set to edge-trigger or level-trigger to generate CPU interrupts.

- **ADC Channel**

The NodeMCU is embedded with a 10-bit precision SAR ADC. The two functions can be implemented using ADC viz. Testing power supply voltage of VDD3P3 pin and testing input voltage of TOUT pin. However, they cannot be implemented at the same time.

- **UART Pins**

ESP8266 NodeMCU has 2 UART interfaces, i.e, UARTO and UART1, which provide asynchronous communication (RS232 and RS485), and can communicate at up to 4.5 Mbps. UARTO (TXD0, RXD0, RSTO0 & CTSO pins) can be used for communication. It supports fluid control. However, UART1 (TXD1 pin) features only data transmit signals so, it is usually used for printing log.

- **SPI Pins**

ESP8266 features two SPIS (SPI and HSPI) in slave and master modes. These SPIS also support the following general-purpose SPI features:

a. 4 timing modes of the SPI format transfer
b. Up to 80 MHz and the divided clocks of 80 MHz
c. Up to 64-Byte FIFO

- **SDIO Pins**

ESP8266 features Secure Digital Input/Output Interface (SDIO) which is used to directly interface SD cards 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO V2.0 are supported.

- **PWM Pins**

The board has 4 channels of Pulse Width Modulation (PWM). The PWM output can be implemented programmatically and used for driving digital motors and LEDs PWM frequency range is adjustable from 1000 us to 10000 us ie, between 100 Hz and 1 kHz.

- **Control Pins**

Control Pins are used for control ESP8266. These pins include Chip Enable pin (EN), Reset pin (RST) and WAKE pin.

a. EN pin - The ESP8266 chip is enabled when EN pin is pulled HIGH, When pulled LOW the chip works at minimum power.
b. RST pin RST pin is used to reset the ESP8266 chip.
c. WAKE pin - Wake pin is used to wake the chip from a deep sleep.

3.3.1.3. Ultrasonic Sensor

One of the most useful components in these projects is a distance sensor. The HC-SR04 Ultrasonic Distance Sensor is an inexpensive device that is very useful for test equipment projects. This tiny sensor is capable of measuring the distance between itself and the nearest solid object.

At its core, the HC-SR04 Ultrasonic distance sensor consists of two ultrasonic transducers. The one acts as a transmitter which converts an electrical signal into 40 kHz ultrasonic sound pulses. The receiver listens for the transmitted pulses. If it receives them it produces an output pulse whose width can be used to determine the distance the pulse traveled.

The HC-SR04 can be hooked directly to an Arduino or other microcontroller and it operates on 5 volts. This ultrasonic distance sensor is capable of measuring distances between 2 cm to 400 cm (that’s about an inch to 13 feet for those of you who don’t “speak” Metric). It’s a low current device so it’s suitable for battery-powered devices.

Specification of HC-SR04

Operating Voltage	: DC 5V
Operating Current	: 15mA
Operating Frequency	: 40KHz
Max Range	: 4m
Min Range	: 2cm
Ranging Accuracy	: 3mm
Measuring Angle	: 15 degree
Trigger Input Signal	: 10µS TTL pulse
Dimension	: 45 x 20 x 15mm

HC-SR04 Pin Configuration

Table no 3.1 HC-SR04 Pin Configuration

Pin Number	Pin Name	Description
1	VCC	The Vcc pin powers the sensor, typically with +5V
2	Trigger	Trigger pin is an input pin. This pin has to be kept high for 10us to initialize measurement by sending US wave.
3	Echo	Echo pin is an output pin. This pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor.
4	Ground	This pin is connected to the Ground of the system.

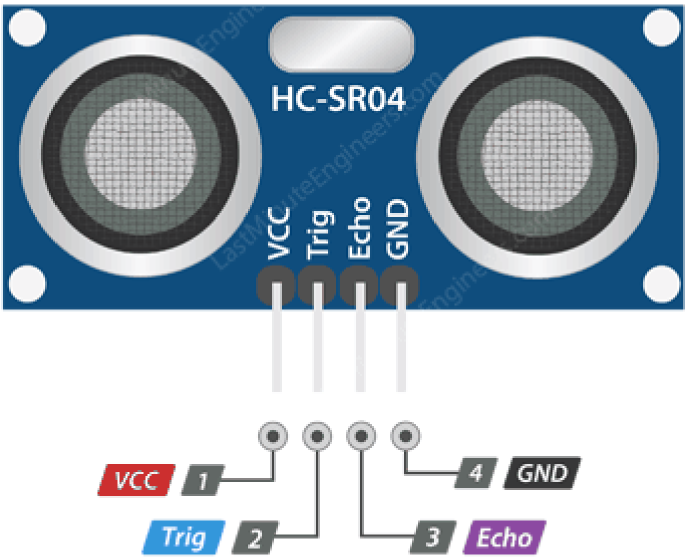


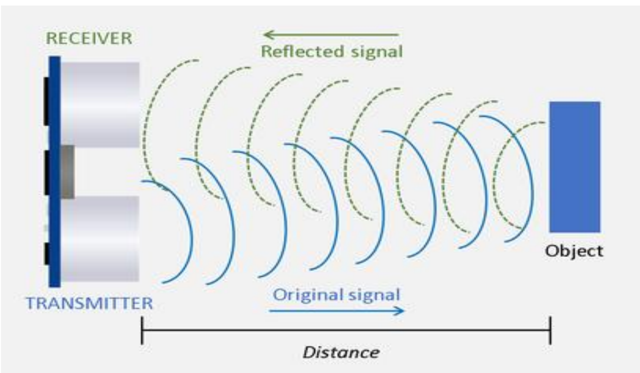
Fig. 3.5 HC-SR04

How the HC-SR04 Works

Ultrasonic distance sensors use pulses of ultrasonic sound (sound above the range of human hearing) to detect the distance between them and nearby solid objects. The sensors consist of two main components:

i. **An Ultrasonic Transmitter** – This transmits the ultrasonic sound pulses, it operates at 40 kHz

ii. **An Ultrasonic Receiver** – The receiver listens for the transmitted pulses. If it receives them it produces an output pulse whose width can be used to determine the distance the pulse travelled.



The device operates as follows :

1. A 5-volt pulse of at least 10 uS (10 microseconds) in duration is applied to the Trigger pin.
2. The HC-SR04 responds by transmitting a burst of eight pulses at 40 kHz. This 8-pulse pattern makes the “ultrasonic signature” from the device unique, allowing the receiver to discriminate between the transmitted pattern and the ultrasonic background noise.
3. The eight ultrasonic pulses travel through the air away from the transmitter. Meanwhile, the Echo pin goes high to start forming the beginning of the echo-back signal.
4. If the pulse in NOT reflected back then the Echo signal will timeout after 38 mS (38 milliseconds) and return low. This produces a 38 mS pulse that indicates no obstruction within the range of the sensor.
5. If the pulse IS reflected back the Echo pin goes low when the signal is received. This produces a pulse whose width varies between 150 uS to 25 mS, depending upon the time it took for the signal to be received.
6. The width of the received pulse is used to calculate the distance to the reflected object. Remember that the pulse indicates the time it took for the signal to be sent out and reflected back so to get the distance you’ll need to divide your result in half.

Distance Calculation :

It is known that,

Distance = Speed × Time

The speed of sound waves is 343 m/s. So,

$$\text{Total Distance} = \frac{343 \times \text{Time of High (Echo) Pulse}}{2}$$

Total distance is divided by 2 because signal travels from HC-SR04 to object and returns to the module HC-SR-04.

3.3.1.4. Breadboard

A breadboard is a solderless device for a temporary prototype with electronics and test circuit designs. Most electronic components in electronic circuits can be interconnected by inserting their leads or terminals into the holes and then making connections through wires where appropriate. The breadboard has strips of metal underneath the board and connects the holes on the top of the board.

The top and bottom rows (the rows indicated by the blue) and are usually the (+) and (-) power supply holes and these move horizontally across the breadboard, while the holes for the components move vertically Each hole is connected to the many metal strips that are running underneath.

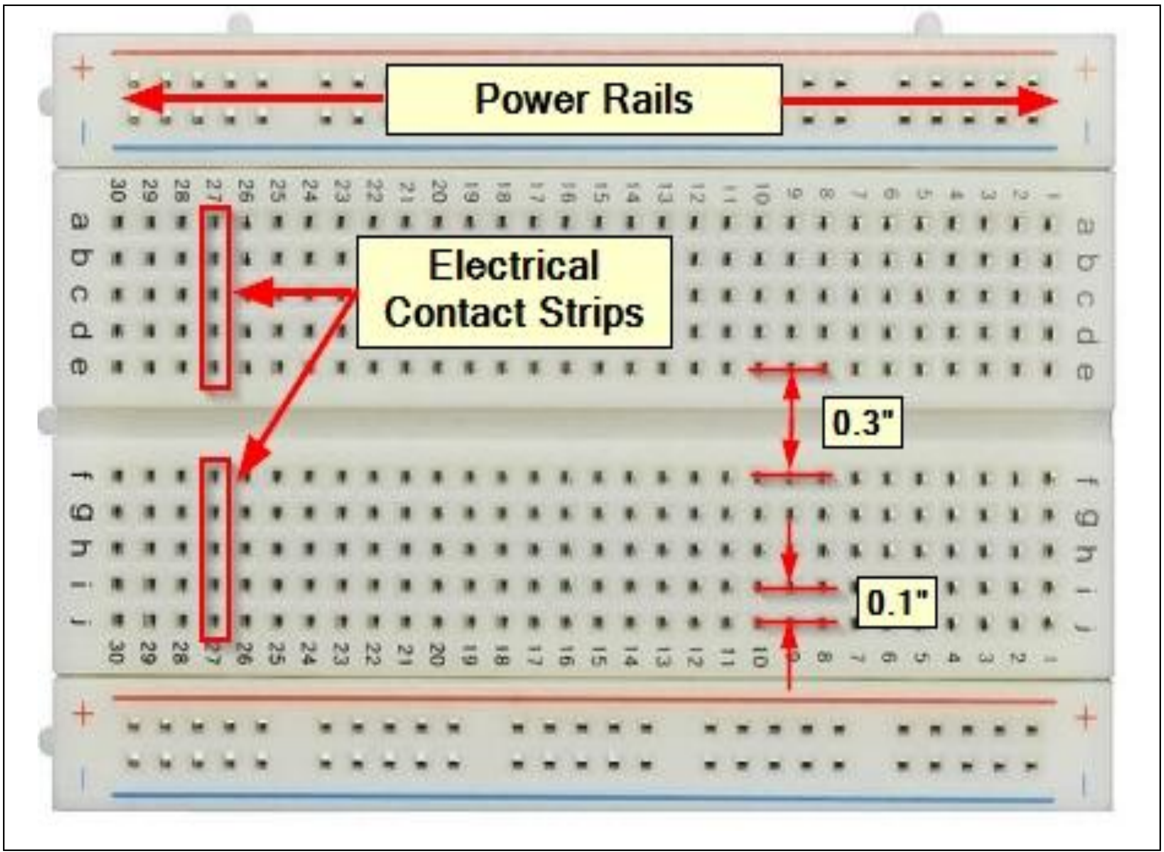


Fig. 3.6 Breadboard with interconnection

3.3.1.5 Wire

Jumper wire

Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each other without soldering. Jumper wires are typically used with breadboards and other prototyping tools in order to make it easy to change a circuit as needed. Individual jumper wire is fitted by inserting its "end connectors" into the slots provided in a breadboard, the header connector of a circuit board, or a piece of the test equipment.

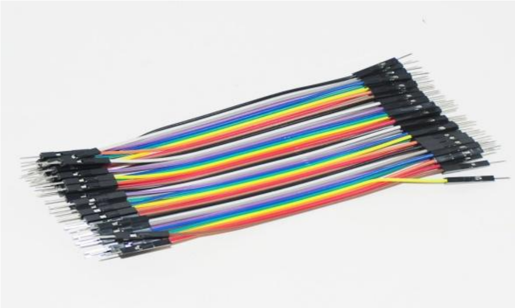


Fig. 3.7 Jumper wire

Single-core wire

It is also electrical wire having a single-core metal thread. It is used for connecting sensors and microcontroller which are far away from each other. single-core coated with PVC insulation. It is used for Breadboard, Electronic Lab Purpose, Electronic Project.

Fig. 3.8 Single core wire

3.3.2 Connection Diagram of Smart Runoff Measurement System

A wiring diagram is a simplified conventional pictorial representation of an electrical circuit. It shows the components of the circuit as simplified shapes and the power and signal connections between the devices. A wiring diagram is mainly intended to convey the wiring or connection between the components properly without any confusion so that one can create prototype easily. The connection diagram of different components of a smart system is given in Fig. 4.2 and pin configuration in Table 3.2.

Table 3.2 Pin configuration of different components of smart runoff measurement system

Arduino Mega	**NodeMCU**	**Work**
SDA	D2	Serial Data Line
SCL	D1	Serial Clock Line
Gnd	Gnd	Ground Line
Arduino Mega	**Ultrasonic Sensors**	**Work**
5 V	VCC	Power Line
D2 to D13,D22*	Trig	Activate Transmitter for Ultrasonic Pulses.
D2 to D13,D22*	Echo	Activate Receiver for listen Ultrasonic Pulses.
Gnd	Gnd	Ground Line

* Pin D2 to D13, D22 are used as a single wire line for both trigger and echo because we used three wire interface

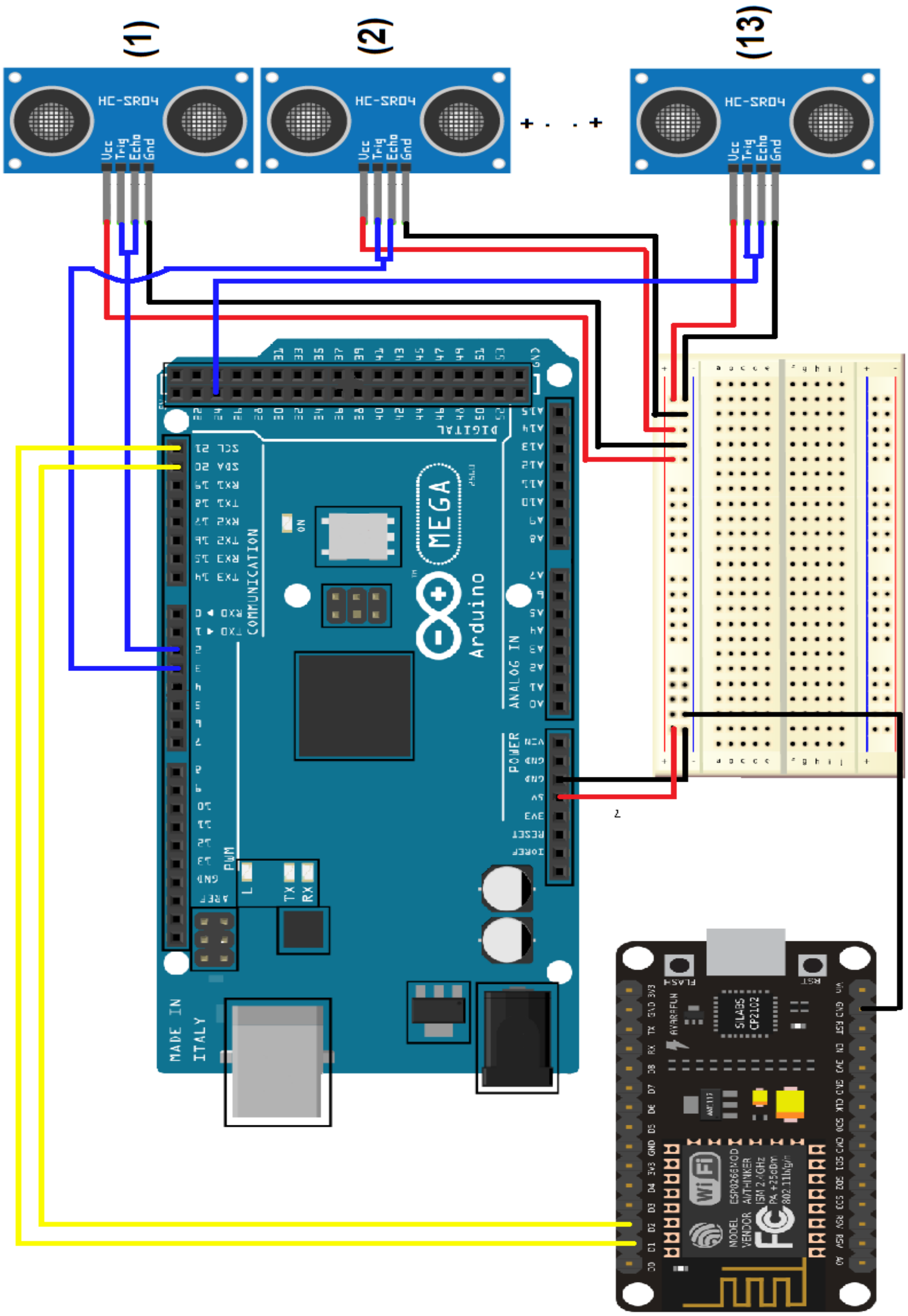


Fig 3.9 Circuit diagram of smart runoff measurement system

3.3.3 Smart Runoff Measurement System Software

3.3.3.1 Arduino IDE

Arduino IDE (Integrated Development Environment) is official software introduced by Arduino.cc that is mainly used for writing, compiling and uploading the code in the Arduino Device. Almost all Arduino modules are compatible with this software that is an open-source and is readily available to install and start compiling the code on the go.

The main code, also known as a sketch, created on the IDE platform will ultimately generate a Hex File which is then transferred and uploaded in the controller on the board. The IDE environment mainly contains two basic parts: Editor and Compiler where former is used for writing the required code and later is used for compiling and uploading the code into the given Arduino Module. This environment supports both C and C++ languages.

The IDE environment is mainly distributed into three sections.

1. Menu Bar
2. Text Editor
3. Output Pan

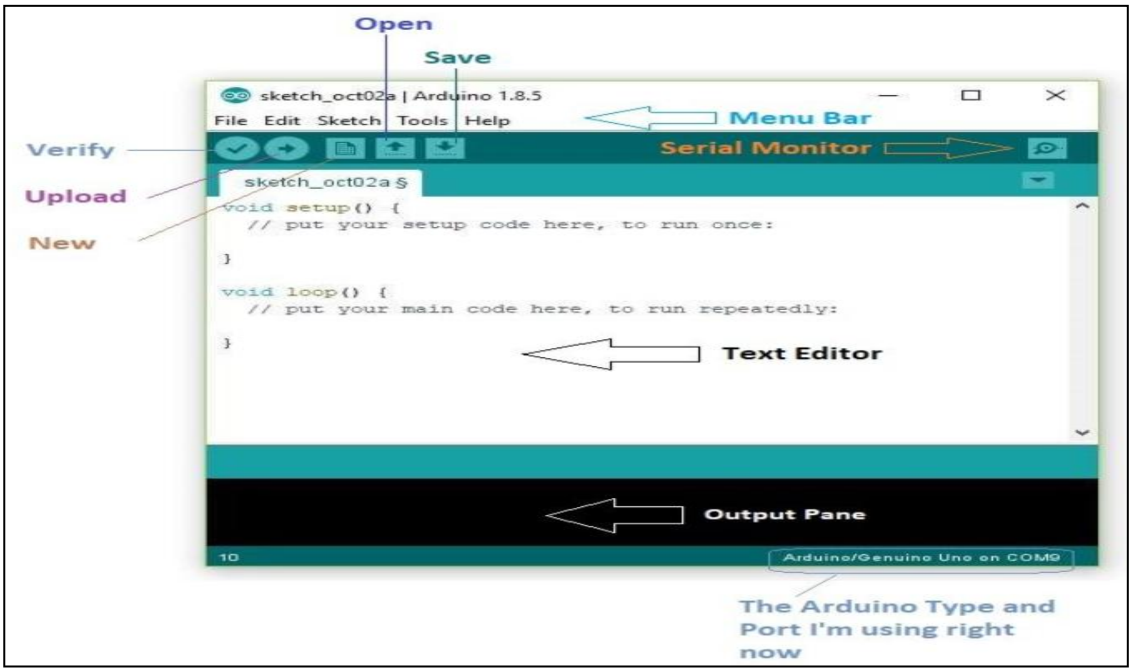


Fig. 3.10 Arduino IDE Software

1. Menu bar

The bar appearing on the top is called Menu Bar that comes with five different options as follows.

- **File** – You can open a new window for writing the code or open an existing one. File option is categorized into a subdivision like New, Open, Save, Quit, etc.
- **Edit** – Used for copying and pasting the code with further modification for font
- **Sketch** – For compiling and programming
- **Tools** – Mainly used for testing projects. The Programmer section in this panel is used for burning a boot loader to the new microcontroller.
- **Help** – In case you are feeling skeptical about software, complete help is available from getting started to troubleshooting.

2. Text Editor

The main screen below the Menu bar is known as a simple text editor used for writing the required code.

3. Output Pane

The bottom of the main screen is described as an Output Pane that mainly highlights the compilation status of the running code: the memory used by the code, and errors occurred in the program. You need to fix those errors before you intend to upload the hex file into your Arduino Module.

3.3.3.2 STRUCTURE OF ARDUINO SKETCH

Arduino programs made in Arduino IDE are known as sketches. So if we talk about the structure of basic Arduino sketch, then we can say that it consists of two mandatory functions known as setup () and loop () functions. Whenever we open a new window in Arduino IDE we can see that these two functions are already present in new sketch.

- **SETUP () FUNCTION:**

  It is always executed first. This function is only executed once every time that sketch is run and it will start it's execution of instruction present in it once it has been programmed in our Arduino board. It can be re-run by using reset button or by connecting Arduino again by disconnecting it.

- **LOOP () FUNCTION:**

  Statements present in loop () function will continuously run from top to bottom and then back to top. But even if we have nothing to do in loop () we still have to use this function in our sketch otherwise our microcontroller will start executing whatever it finds next in memory. So in order to prevent this issue, we have to define loop function in our sketch. The operational functionality of Arduino takes place in loop () function.

3.3.3.3 Arduino Libraries

Libraries are files written in C or C++ (.c, .cpp) which provide your sketches with extra functionality.

Libraries Used

1. **New Ping Library**

   The New Ping library written by Tim Eckel can be used with many ultrasonic distance sensors. the New Ping library does include some other nice features. It allows you to set a max distance to read, it won't lag for a full second when no echo is received and it has a built-in median filter.

2. **Wire Library**

   This library allows you to communicate with I2C / TWI devices. On the Arduino boards with the R3 layout (1.0 pinout), the SDA (data line) and SCL (clock line) are on the pin headers close to the AREF pin.

3. **ESP8266WiFi**

The Wi-Fi library for ESP8266 has been developed based on ESP8266 SDK, using the naming conventions and overall functionality philosophy of the Arduino Wi-Fi library. This library allows a nodeMCU board to connect to the internet. It can serve as either a server accepting incoming connections or a client making outgoing ones.

3.3.3.4 I2C Serial Communication

The I2C communication bus is very popular and broadly used by many electronic devices because it can be easily implemented in many electronic designs which require communication between a master and multiple slave devices or even multiple master devices. The easy implementations come with the fact that only two wires are required for communication between up to almost 128 (112) devices when using 7 bits addressing and up to almost 1024 (1008) devices when using 10 bits addressing. I2C is a serial communication protocol, so data is transferred bit by bit along a single wire (the SDA line).

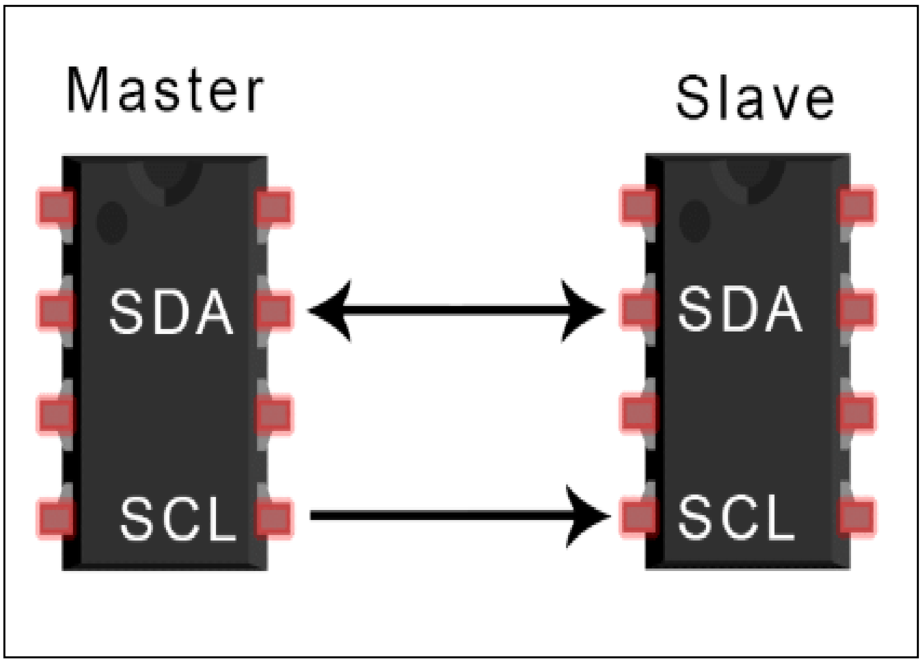


Fig. 3.11 I2C Serial Communication

SDA (Serial Data) – The line for the master and slave to send and receive data.

SCL (Serial Clock) – The line that carries the clock signal.

3.3.3.4.1 Working of I2C

Each device has a preset ID or a unique device address so the master can choose with which devices will be communicating. With I2C, data is transferred in *messages*. Messages are broken up into *frames* of data. Each message has an address frame that contains the binary address of the slave, and one or more data frames that contain the data being transmitted. The message also includes start and stop conditions, read/write bits, and ACK/NACK bits between each data frame:

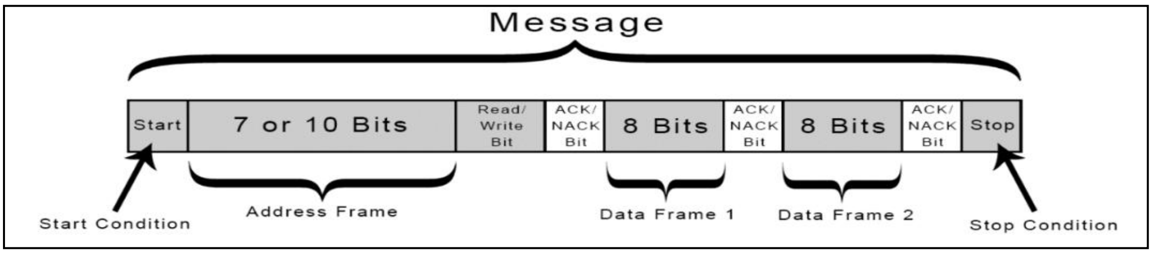


Fig. 3.12 Working of I2C

Start Condition: The SDA line switches from a high voltage level to a low voltage level *before* the SCL line switches from high to low.

Stop Condition: The SDA line switches from a low voltage level to a high voltage level *after* the SCL line switches from low to high.

Address Frame: A 7 or 10-bit sequence unique to each slave that identifies the slave when the master wants to talk to it.

Read/Write Bit: A single bit specifying whether the master is sending data to the slave (low voltage level) or requesting data from it (high voltage level).

ACK/NACK Bit: Each frame in a message is followed by an acknowledge/no-acknowledge bit. If an address frame or data frame was successfully received, an ACK bit is returned to the sender from the receiving device.

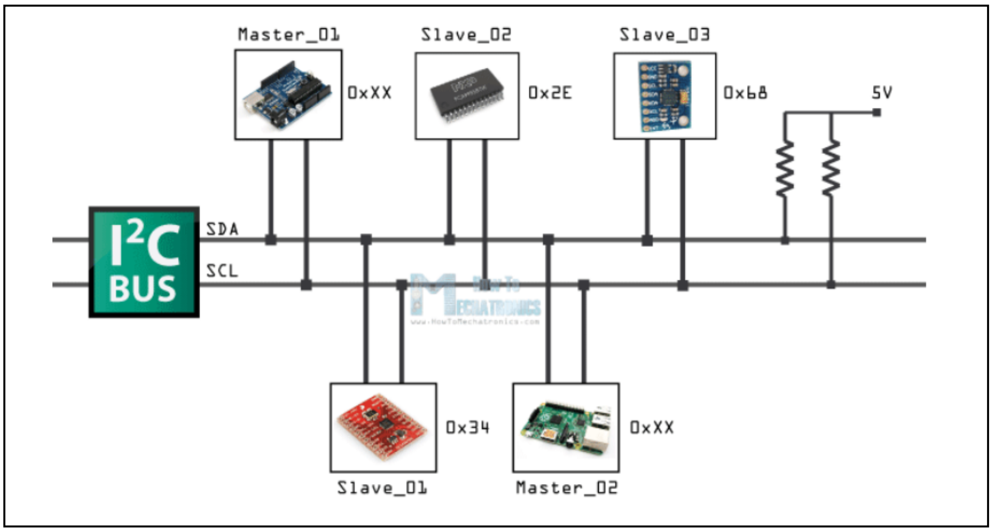


Fig. 3.13 I2C Serial Communication Bus

3.3.3.4.2 Addressing

I2C doesn't have slave select lines like SPI, so it needs another way to let the slave know that data is being sent to it, and not another slave. It does this by *addressing*. The address frame is always the first frame after the start bit in a new message.

The master sends the address of the slave it wants to communicate with every slave connected to it. Each slave then compares the address sent from the master to its address. If the address matches, it sends a low voltage ACK bit back to the master. If the address doesn't match, the slave does nothing and the SDA line remains high.

3.3.3.4.3 READ/WRITE BIT

The address frame includes a single bit at the end that informs the slave whether the master wants to write data to it or receive data from it. If the master wants to send data to the slave, the read/write bit is a low voltage level. If the master is requesting data from the slave, the bit is a high voltage level.

3.3.3.4.4 The data frame

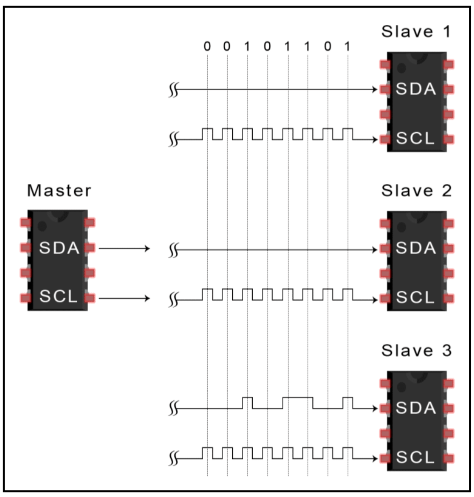
After the master detects the ACK bit from the slave, the first data frame is ready to be sent. The data frame is always 8 bits long and sent with the most significant bit first. Each data frame is immediately followed by an ACK/NACK bit to verify that the frame has been received successfully. The ACK bit must be received by either the master or the slave (depending on who is sending the data) before the next data frame can be sent.

After all of the data, frames have been sent, the master can send a stop condition to the slave to halt the transmission. The stop condition is a voltage transition from low to high on the SDA line after a low to high transition on the SCL line, with the SCL line remaining high.

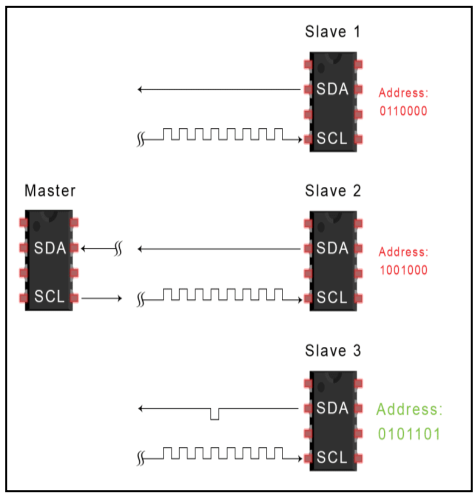
3.3.3.4.5 STEPS OF I2C DATA TRANSMISSION

1. The master sends the start condition to every connected slave by switching the SDA line from a high voltage level to a low voltage level before switching the SCL line from high to low.

2. The master sends each slave the 7 or 10 bit address of the slave it wants to communicate with, along with the read/write bit.



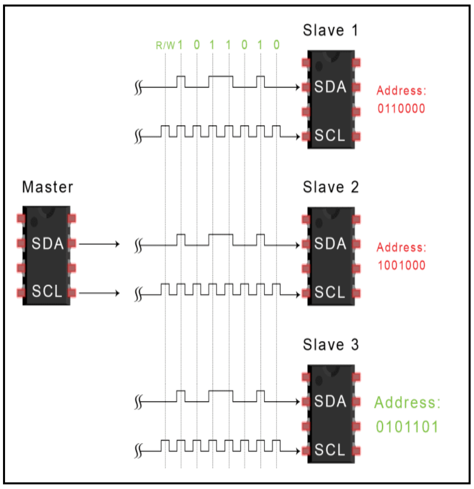
Step-1



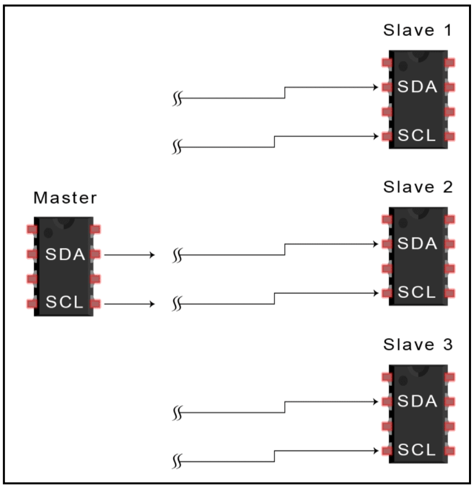
Step-2

3. Each slave compares the address sent from the master to its own address. If the address matches, the slave returns an ACK bit by pulling the SDA line low for one bit. If the address from the master does not match the slave's own address, the slave leaves the SDA line high.

4. The master sends or receives the data frame.



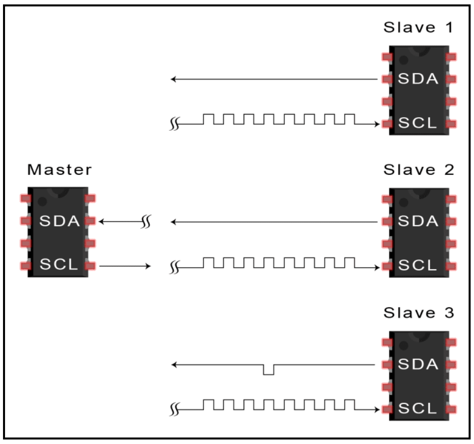
Step-3



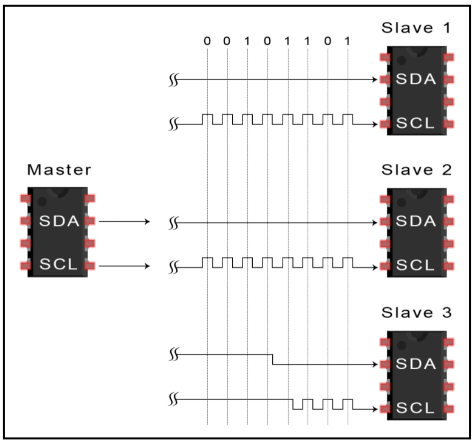
Step-4

5. After each data frame has been transferred, the receiving device returns another ACK bit to the sender to acknowledge successful receipt of the frame.

6. To stop the data transmission, the master sends a stop condition to the slave by switching SCL high before switching SDA high.



Step-5



Step-6

3.3.3.4.6 LIMITATION OF I2C AND IT'S SOLUTION

I2C normally only sends a single byte at a time so you are limited to 255 as the largest number. In this experiment, we have to send larger number than 255 from

Arduino Mega to nodeMCU. Number which is bigger than 255 have more than one byte so sending them over I2C will be a bit of a problem. For this particular problem, we found following solution. We split the number into individual bytes, put them into an array, and send it over. On the other end, you read the array in as individual bytes and rebuild it into the appropriately sized number.

3.3.3.5 Third-party Website for data storage and data visualization

"ThingSpeak" is an open-source Internet of Things (IoT) application and API to store and retrieve data from things using the HTTP and MQTT protocol over the Internet or via a Local Area Network. It is an IoT analytics platform service that allows you to aggregate, visualize, and analyze live data streams in the cloud. You can send data to ThingSpeak from your devices, create instant visualizations of live data.

Feature of ThinkSpeak

Some of the key capabilities of ThingSpeak include the ability to:

- Easily configure devices to send data to ThingSpeak using popular IoT protocols.
- Visualize your sensor data in real-time.
- Aggregate data on-demand from third-party sources.
- Use the power of MATLAB to make sense of your IoT data.
- Run your IoT analytics automatically based on schedules or events.
- Prototype and build IoT systems without setting up servers or developing web software.

3.3.4 Power Unit

Solar energy is the cleanest and most available renewable energy source. Solar energy can also be used to meet our electricity requirements. Through solar photovoltaic (SPV) cells, solar radiation gets converted into DC electricity directly. This electricity can either be used as it is or can be stored in the battery.

solar power systems consist of a core set of components

- Solar Panels
- Batteries
- Charge Controllers

3.3.4.1 Solar Panel

A solar panel is a collection of solar cells, the solar panel converts solar energy into electrical energy. The solar panel uses Ohmic material for interconnections as well as the external terminals. So the electrons created in the n-type material passes through the electrode to the wire connected to the battery. Through the battery. the electrons reach the p-type material. Here the electrons combine with the holes. So when the solar panel is connected to the battery. It behaves like another battery, and both the systems are in series just like two batteries connected serially. The material most often used for solar cells is silicon. This material can occur in three forms:

- monocrystalline silicon
- polycrystalline (or multi-crystalline) silicon
- amorphous silicon

The two types of solar panels most suitable for residential solar installations are monocrystalline and polycrystalline. These perform similarly although the monocrystalline is slightly more efficient and a little more expensive.

The output of the solar panel is its power which is measured in terms of Watts or Kilowatts. So before selecting the solar panel, it is necessary to find out the power requiring for the load Watt-hours or Kilowatt-hours is used for calculating the power requirement. So, we select monocrystalline solar panel output ratings is 10 watts.

Fig. 3.14 Solar Panel

Specification of solar panel:

Maximum Power (P_{max})	: 10Wp
Maximum Power Voltage (V_{mp})	: 19.25V
Maximum Power Current (I_{mp})	: 0.52A
Short Circuit Current (I_{sc})	: 0.55A
Open Circuit Voltage (V_{oc})	: 22.50V
Maximum System Voltage	: 600V

3.3.4.2 Solar Battery

For some problems, such as unstable grid energy over-charging or discharging ad irregular full recharging, is for the solar battery important to meet those demands. Solar batteries work by storing energy produced by your solar panels and storing it as for later use.

SMF battery which means Sealed Maintenance Free battery are sealed completely because there is no need to add water. The electrolyte used is in the form of gel which fills the cavity of plates. Just like other batteries, it also emit H2 and O2 gases and due to sealed batteries both these gases combine to form water.

SMF batteries measure created in an eco-friendly, ISO Certified & trendy plant with a huge producing capability and square measure being sold-out worldwide. There are differences between SMF batteries and other tubular batteries. In SMF Batteries no distilled water or effort is needed and requires only. There is a wide selection of SMF battery on the market to suit all applications of standby power needs like UPS, electrical converter and Emergency Lights, communication system, hearth Alarm & Security Systems, Railway communication, Electronic group action and money Registers, star Lanterns and Systems, etc.

So, for our project, we select 12v 7Amp solar battery to fulfill our requirement.

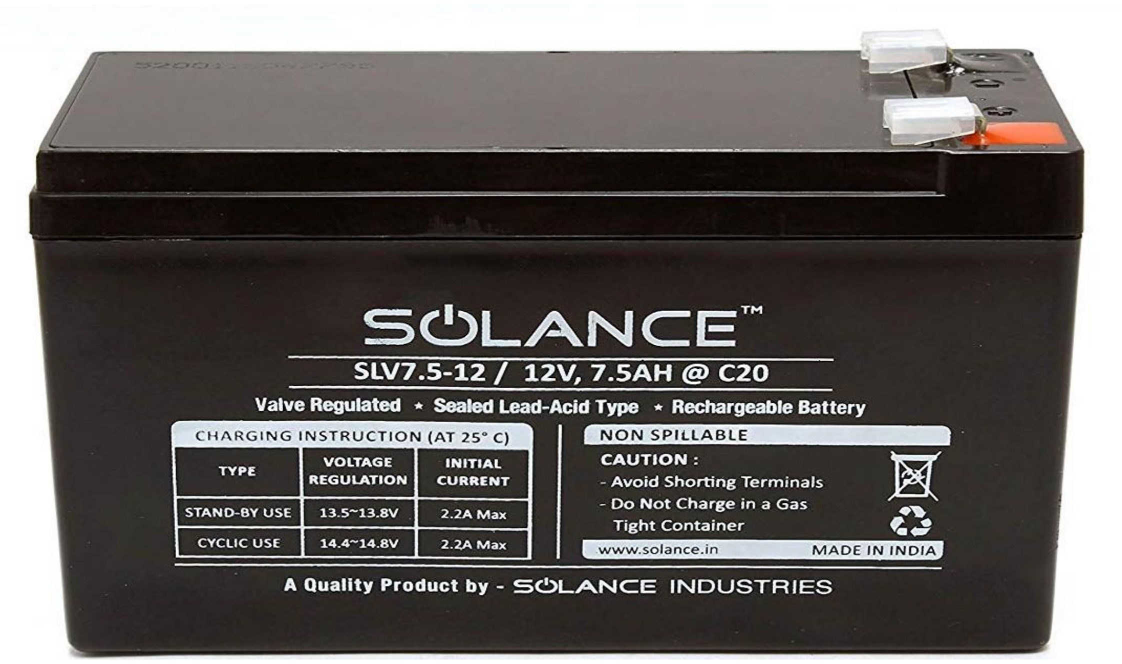


Fig. 3.15 Solar Battery

Specification of solar battery:

Available Charging Mode : 12V - 7AH

Stand By (Float Voltage) : 13.5 – 13.8 V

Cycle (Boost Voltage) : 14.4 – 15.0 V

Charging Current Max : 1.75 AMPS

3.3.3.3 Solar Charge Controller

A charge controller is an essential part of nearly all power systems that charge batteries. Whether the power source is PV, wind, hydro, fuel or utility grid. Its purpose is to keep your batteries properly fed and safe for the long term.

The basic functions of a controller are quite simple. Charge controllers block reverse current and prevent battery overcharge. Some controllers also prevent battery over-discharge. Protect from electrical overload and display battery status and the flow of power.

PWM (Pulse Width Modulation)

Pulse-Width Modulation (PWM) comes into play when the battery bank is full. During charging, the controller allows as much current as the PV panel/array can generate in order to reach the target voltage for the charge stage the controller is in. Once the battery approaches this target voltage, the charge controller quickly switches between connecting the battery bank to the panel array and disconnecting the battery bank, which regulates the battery voltage holding it constant. This quick switching is called PWM and it ensures your battery bank is efficiently charged while protecting it from being overcharged by the PV panel/array.

Fig. 3.16 Solar Charge Controller

FEATURES OF A SOLAR CHARGE CONTROLLER ARE:

Multistage charging of battery bank - changes the amount of power set to the batteries based on its charge level, for healthier batteries.

Reverse current protection - stops the solar panels from draining the batteries at night when there is no power coming from the solar panels.

Low voltage disconnect - turns off the attached load when battery is low and turns it back on when the battery is charged back up.

Lighting control - turns attached light on and off based on dusk and dawn. Many controllers are configurable, allowing settings for a few hours or all night, or somewhere in between.

Display- may show a voltage of battery bank, state of charge, amps coming in from solar panel.

Specification of solar charge controller:

Model	PWM SCC 12V
Voltage Rating	12 V
Charging Condition	Float cum boost 14.50V
Charging Current	10A
USB port	5V / 2A
Battery low load Disconnect	<= 10.4V Battery
Battery low load Recovery	>= 12.6V Battery
Battery High Voltage load Disconnect	>= 15.5V Battery
Battery High Voltage load Recovery	<= 14.5V Battery

Connection diagram for Solar PV System:

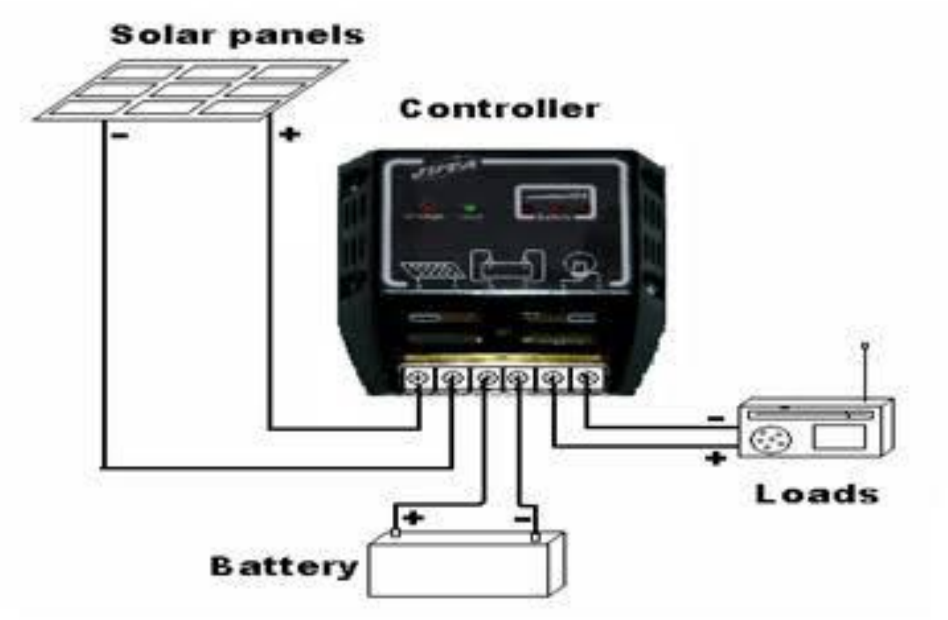


Fig.3.17 Connection diagram for Solar PV system

3.3.3.4 Solar PV system sizing

1. Determine power consumption demands

- The first step in designing a solar PV system is to find out the total power and energy consumption of loads that need to be supplied by the solar PV system.
- Electric power is the rate at which energy is transferred to or from a part of an electric circuit.

Power = Voltage x Current

Now power requirement for:

i. Arduino Mega 2560

- Output current = 200mA
- Output voltage= 12 volt
- Output wattage = 0.2 * 12 = 2.4 W

ii. NODEMCU- ESP8266

- Output current = 120 mA
- Output voltage= 5 volt
- Output wattage = 0.12 * 5 = 0.6 W

Power Requirement = 2.4 + 0.6 = 3 W

Total PV Panel energy needed =1.3 * 3 = 3.9 W

2. **Size the PV modules**

Different sizes of PV modules will produce different amounts of power. To find out the sizing of PV module, the total peak watt produced needs. The peak watt (Wp) produced depends on size of the PV module and climate of site location

Our experiment power requirement is 3.9 W

So, we can use the 10 watts and 12-volt solar panel to full fill over requirement.

3. Size of Battery

The battery type recommended for using in solar PV system is deep cycle battery. Deep cycle battery is specifically designed for to be discharged to low energy level and rapid recharged or cycle charged and discharged day after day years The battery should be large enough to store sufficient energy to operate the appliances at night and cloudy days. To find out the size of battery calculate as follows

Battery Capacity (Ah) =

$$\frac{\text{Total Watt} - \text{hours per day used by appliances} \times \text{Days of autonomy}}{(0.85 \times 0.6 \times \text{nominal battery voltage})}$$

Where,

Battery loss co-efficient = 0.85

Depth of discharge = 0.60

Day of autonomy = 5 days

Battery Capacity (Ah) = $\frac{3.9 \times 5}{(0.85 \times 0.6 \times 12)}$ = 3.18 Ah

So, we use 12 volts 7 Ah solar battery to full fill over requirement.

4. Selection of solar charge controller

The solar charge controller is typically rated against Amperage and Voltage capacities. Select the solar charge controller to match the voltage of PV array and batteries and then identify which type of solar charge controller is right for your application. Make sure that solar charge controller has enough capacity to handle the current from PV array.

For the series charge controller type, the sizing of controller depends on the total PV input current which is delivered to the controller and also depends on PV panel configuration (series or parallel configuration). We have 10W 12v solar panel and 12v 7ah solar battery according to that we select the solar charge controller.