HARDWARE REQUIRED

1. Arduino Nano – 1 pcs
2. Arduino Nano USB Cable – 1 pcs
3. Ultrasonic Sensor Module HC-SR04 - 1 pcs
4. Rain Drop Sensor Module – 1 pcs
5. 9v Battery - 1 pcs
6. Battery snapper - 1 pcs
7. B20 Buzzer - 2 pcs
8. Jumper wire (Female to Female) - As per requirement
9. Mini Rocker switch – 1 pcs
10. Wire – As per requirement
11. Pipe or Stick
12. Arduino IDE Software

HARDWARE DESCRIPTION

1.Arduino Nano

Synopsis Description

Arduino is an open source microcontroller board. The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328P (Arduino Nano 3.x).The microcontroller on the board is programmed using Arduino software.

The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards or breadboards and other circuit. The Microcontrollers are typically programmed using a dialect of features from programming language C & C++. In addition to using traditional compiler tool chains, the Arduino project provides an integrated development environment (IDE) bases on the processing language project.

Power

The Arduino Nano can be powered via the Mini-B USB connection, 6-20V unregulated external power supply (pin 30), or 5V regulated external power supply (pin 27). The power source is automatically selected to the highest voltage source.

Memory

The ATmega328 has 32 KB, (also with 2 KB used for the bootloader. The ATmega328 has 2 KB of SRAM and 1 KB of EEPROM.

Input and Output

Each of the 14 digital pins on the Nano can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the FTDI USB-to-TTL Serial chip.
- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
- LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Nano has 8 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the analogReference() function. Analog pins 6 and 7 cannot be used as digital pins. Additionally, some pins have specialized functionality:

- I2C: A4 (SDA) and A5 (SCL). Support I2C (TWI) communication using the Wire library (documentation on the Wiring website).

There are a couple of other pins on the board:

- AREF. Reference voltage for the analog inputs. Used with analogReference().
- Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Communication

The Arduino Nano has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provide UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An FTDI FT232RL on the board channels this serial communication over USB and the FTDI drivers (included with the Arduino software) provide a virtual com port to software on the computer. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the FTDI chip and USB connection to the computer (but not for serial communication on pins 0 and 1).
A SoftwareSerial library allows for serial communication on any of the Nano's digital pins.
The ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus. To use the SPI communication, please see ATmega328 datasheet.

Programming

The Arduino Nano can be programmed with the Arduino software (download). Select "Arduino Duemilanove or Nano w/ ATmega328" from the Tools > Board menu (according to the microcontroller on your board).
The ATmega328 on the Arduino Nano comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol.
You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using Arduino ISP or similar.

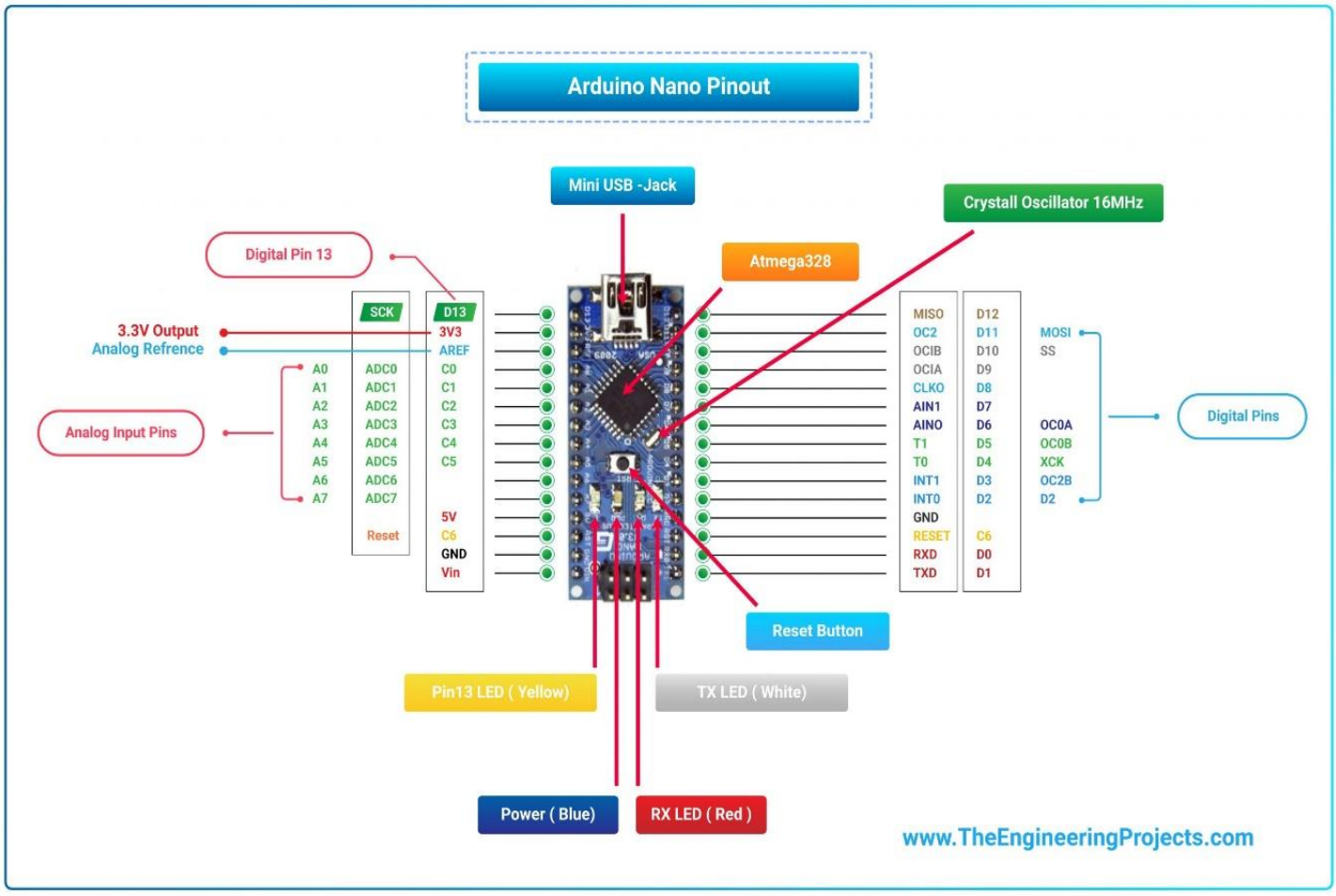
Automatic (Software) Reset

> Rather then requiring a physical press of the reset button before an upload, the Arduino Nano is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of

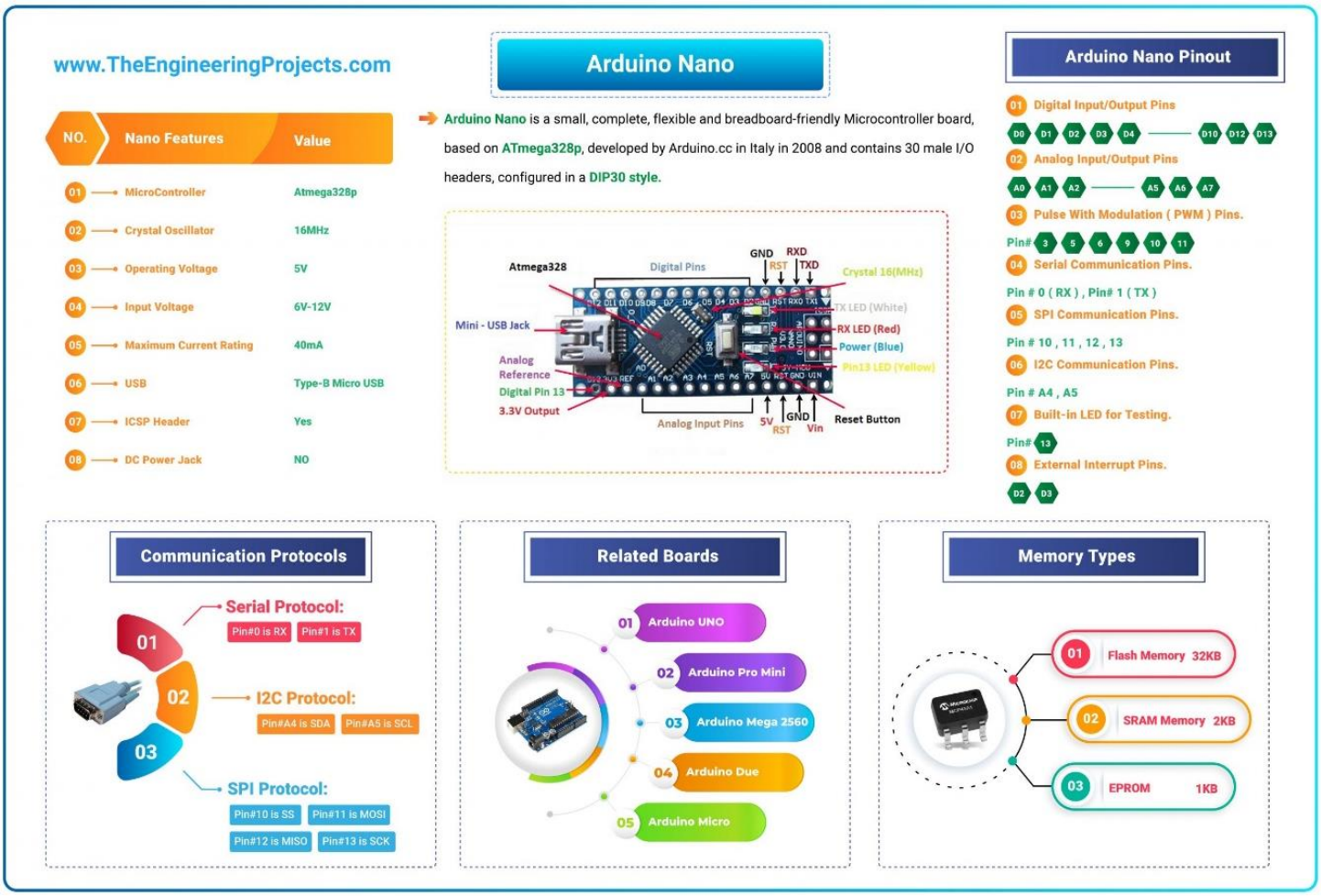
the FT232RL is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload. This setup has other implications. When the Nano is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Nano. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

Brief Description :- Arduino nano in very detailed explanation with Graphics

Arduino Nano Pinout, datasheet, drivers & applications. It is a Microcontroller board developed by arduino.cc and based on Atmega328p / Atmega168. Arduino boards are widely used in robotics, embedded systems, automation, Internet of Things (IoT) and electronics projects. These boards were initially introduced for the students and non-technical users but nowadays Arduino boards are widely used in industrial projects.



Here’s the figure showing key points of Arduino Nano:



- Here’s the table showing important features of Arduino Nano:

1	Microcontroller	Atmega328p
2	Crystal Oscillator	16MHz
3	Operating Voltage	5V
4	Input Voltage	6V-12V
5	Maximum Current Rating	40mA
6	USB	Type-B Micro USB
7	ICSP Header	Yes
8	DC Power Jack	No

Here’s the quick overview of Arduino Nano Pinout:

1	D0 – D13	Digital Input / Output Pins.
2	A0 – A7	Analog Input / Output Pins.
3	Pin # 3, 5, 6, 9, 10, 11	Pulse Width Modulation (PWM) Pins.
4	Pin # 0 (RX) , Pin # 1 (TX)	Serial Communication Pins.
5	Pin # 10, 11, 12, 13	SPI Communication Pins.
6	Pin # A4, A5	I2C Communication Pins.
7	Pin # 13	Built-In LED for Testing.
8	D2 & D3	External Interrupt Pins.

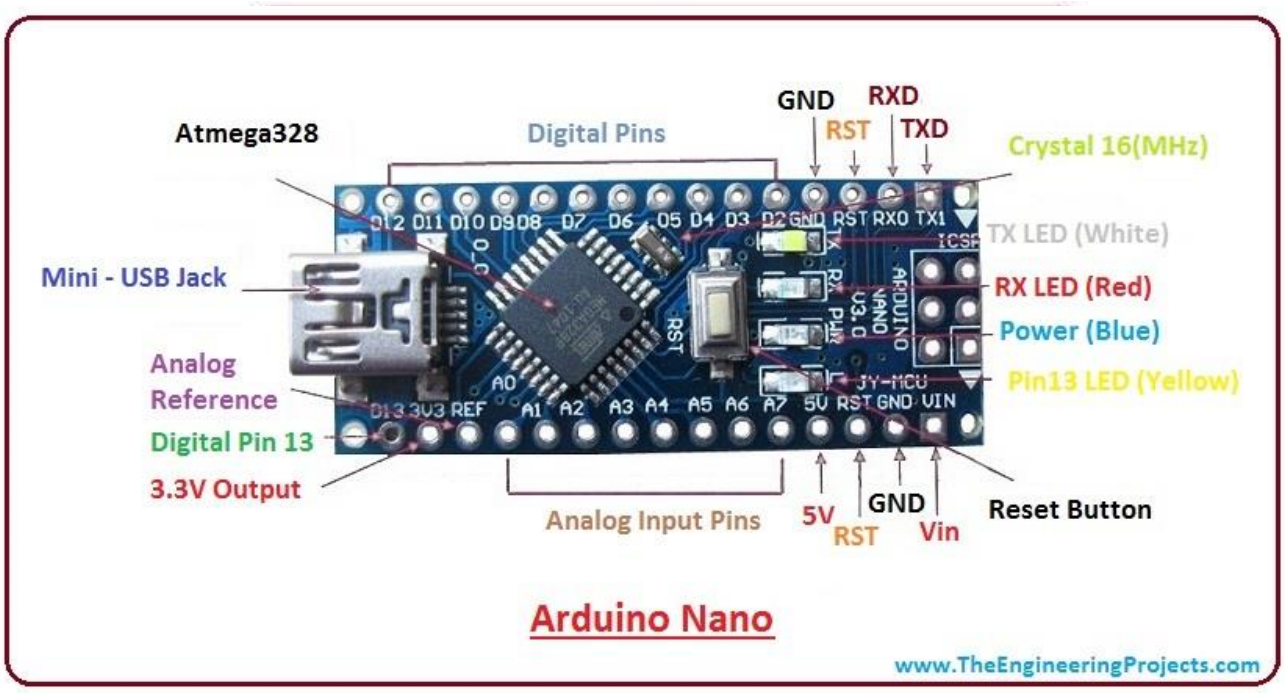
- Arduino Nano offers three types of communications protocols, shown in the below table:

6	Serial Port	1 (Pin#0 is RX, Pin#1 is TX).
7	I2C Port	1 (Pin#A4 is SDA, Pin#A5 is SCL).
8	SPI Port	1 (Pin#10 is SS, Pin#11 is MOSI, Pin#12 is MISO, Pin#13 is SCK).

- Here's the memory details present in Arduino Nano:

7	Flash Memory	32KB
8	SRAM Memory	2KB
7	EEPROM	1KB

Introduction to Arduino Nano



- **Arduino Nano** is a small, complete, flexible and breadboard-friendly Microcontroller board, based on **ATmega328p**, developed by Arduino.cc in Italy in 2008 and contains 30 male I/O headers, configured in a **DIP30 style**.
- **Arduino Nano Pinout** contains 14 digital pins, 8 analog Pins, 2 Reset Pins & 6 Power Pins.
- It is programmed using **Arduino IDE**, which can be downloaded from Arduino Official site.
- Arduino Nano is simply a smaller version of Arduino UNO, thus both have almost the same functionalities.
- It comes with an **operating voltage of 5V**, however, the input voltage can vary from **7 to 12V**.
- Arduino Nano's **maximum current rating is 40mA**, so the load attached to its pins shouldn't draw current more than that.
- Each of these Digital & Analog Pins is assigned with multiple functions but their main function is to be configured as **Input/Output**.

- Arduino Pins are acted as **Input Pins** when they are interfaced with sensors, but if you are driving some load then we need to use them as an **Output Pin**.
- Functions like **pinMode()** and **digitalWrite()** are used to control the operations of digital pins while **analogRead()** is used to control analog pins.
- The analog pins come with a total **resolution of 10-bits** which measures the value from 0 to 5V.
- Arduino Nano comes with a **crystal oscillator of frequency 16 MHz**. It is used to produce a clock of precise frequency using constant voltage.
- There is one limitation of using Arduino Nano i.e. it doesn't come with a **DC power jack**, which means you can not supply an external power source through a battery.
- This board doesn't use standard USB for connection with a computer, instead, it comes with **Type-B Micro USB**.
- The tiny size and breadboard-friendly nature make this device an ideal choice for most applications where the size of the electronic components is of great concern.
- **Flash memory is 16KB or 32KB** that all depends on the Atmega board i.e Atmega168 comes with 16KB of flash memory while Atmega328 comes with a flash memory of 32KB. Flash memory is used for storing code. The 2KB of memory out of total flash memory is used for a bootloader.
- The **SRAM memory of 2KB** is present in Arduino Nano.
- Arduino Nano has an **EEPROM memory of 1KB**.

- The following figure shows the specifications of the Arduino Nano board.

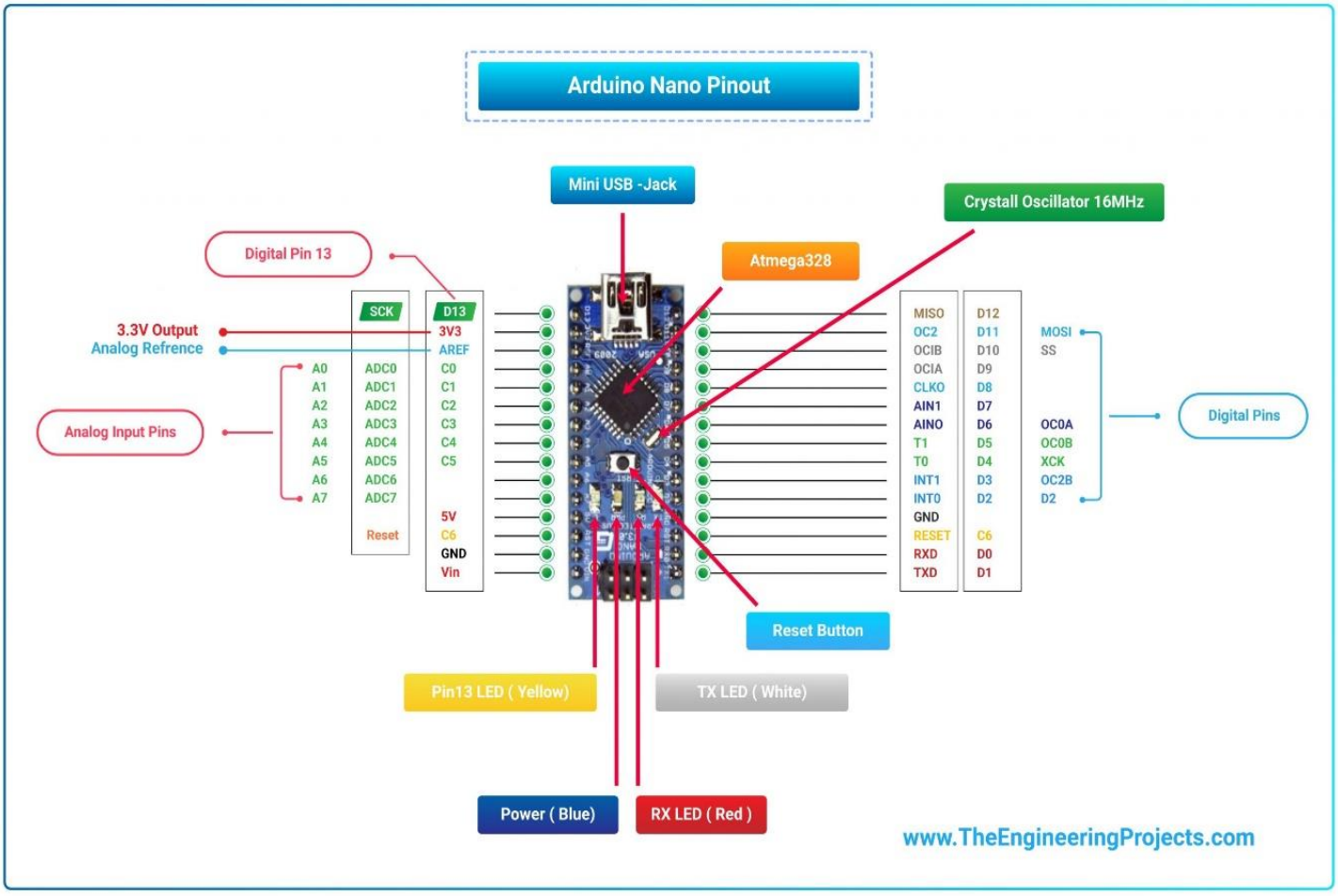
Arduino Nano Spacification

The following figure shows the specifications of the Arduino Nano board.

Microcontroller	Atmega328/Atmega168
Operating Voltage	5V
Input Voltage	7 - 12 V
Digital I/O Pins	14
PMW	6 Out of 14 Digital Pins
Max. Current Rating	40mA
USB	Mini
Analog Pins	8
Flash Memory	16KB or 32KB
SRAM	1KB or 2KB
Crystal Oscillator	16 MHz
EEPROM	512byte or 1KB
USART	Yes

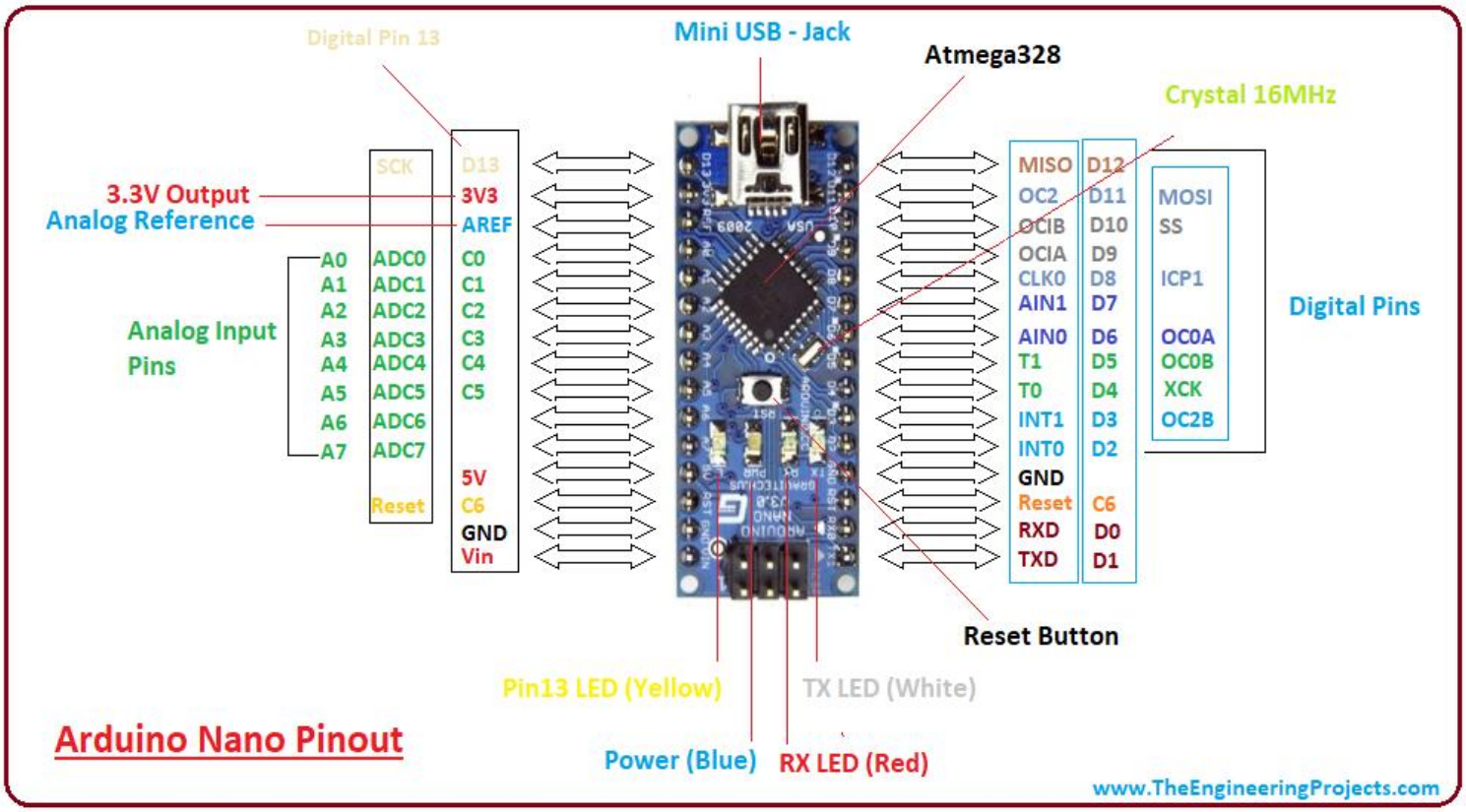
- It is programmed using Arduino IDE which is an Integrated Development Environment that runs both offline and online.
- No prior arrangements are required to run the board. All you need is a board, mini USB cable and Arduino IDE software installed on the computer.
- USB cable is used to transfer the program from the computer to the board.
- No separate burner is required to compile and burn the program as this board comes with a built-in boot-loader.

Now, let's have a look at Arduino Nano Pinout in detail:



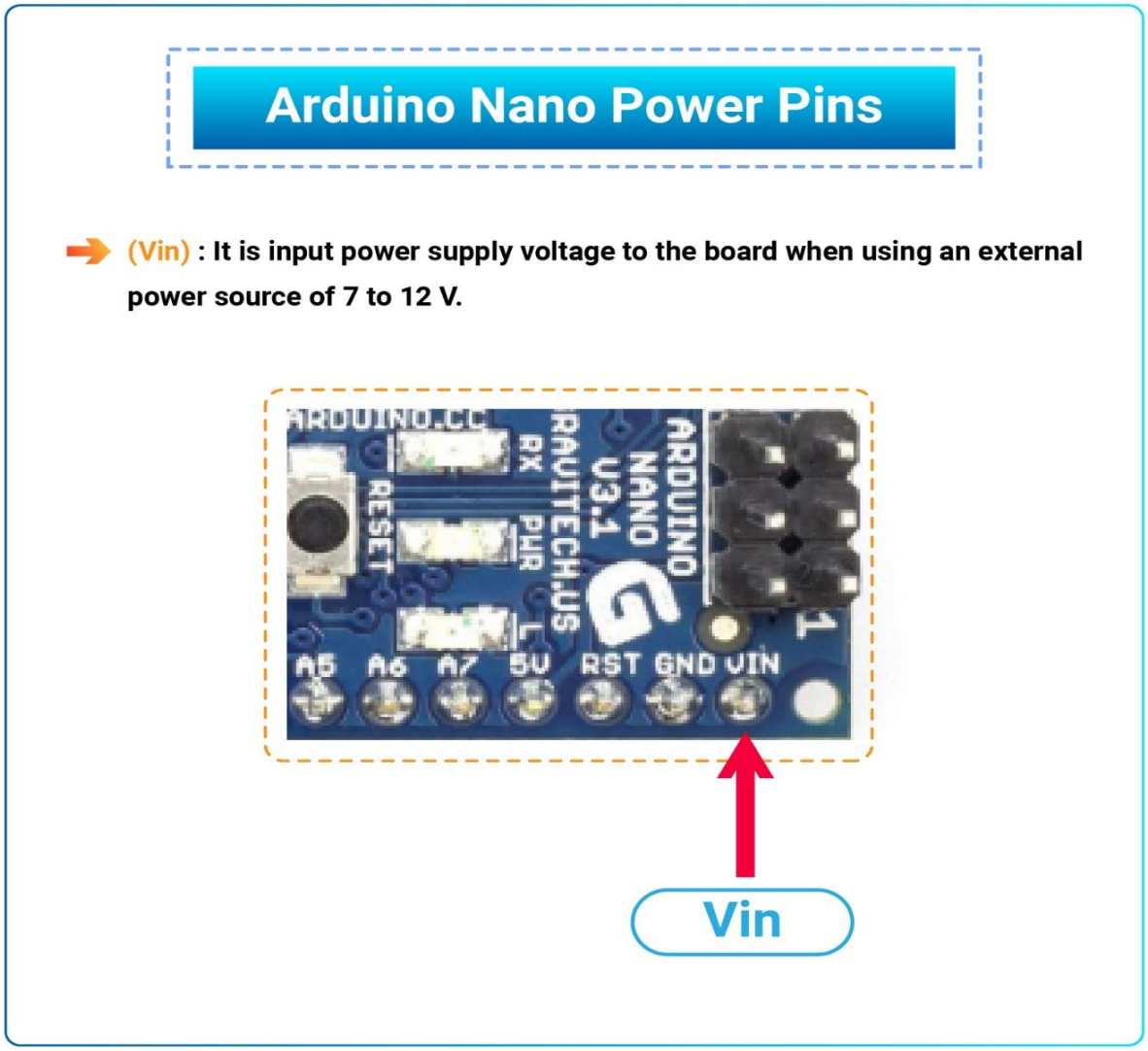
Arduino Nano Pinout

- The following figure shows the pinout of the Arduino Nano Board:



- Each pin on the Nano board comes with a specific function associated with it.
- We can see the analog pins that can be used as an analog to a digital converter, where A4 and A5 pins can also be used for I2C communication.
- Similarly, there are 14 digital pins, out of which 6 pins are used for generating PWM.

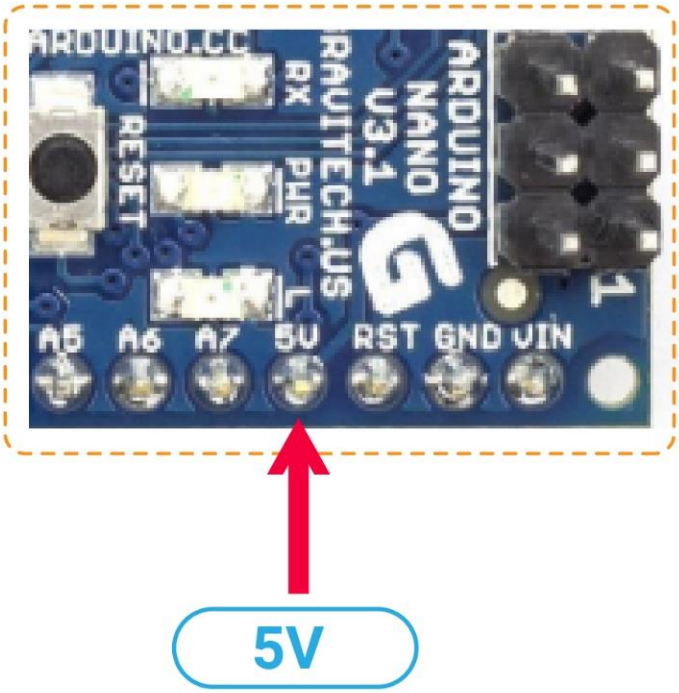
Let’s have a look at the Arduino Nano Pinout in detail:



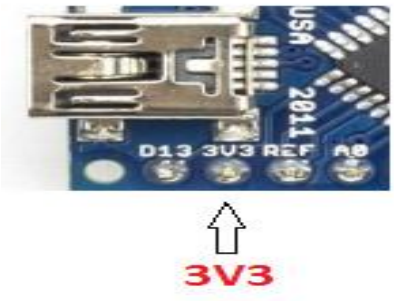
- **Vin:** It is input power supply voltage to the board when using an external power source of 7 to 12 V.
- **5V:** It is a regulated power supply voltage of the board that is used to power the controller and other components placed on the board.

Arduino Nano Power Pins

5V: It is a regulated power supply voltage of the board that is used to power the controller and other components placed on the board..

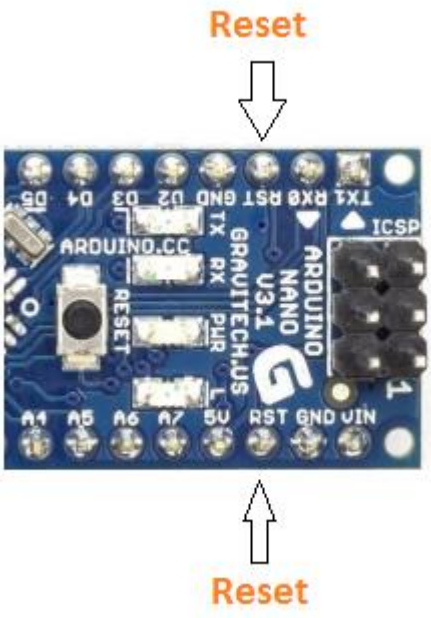


- **3V3:** This is a minimum voltage generated by the voltage regulator on the nano board.



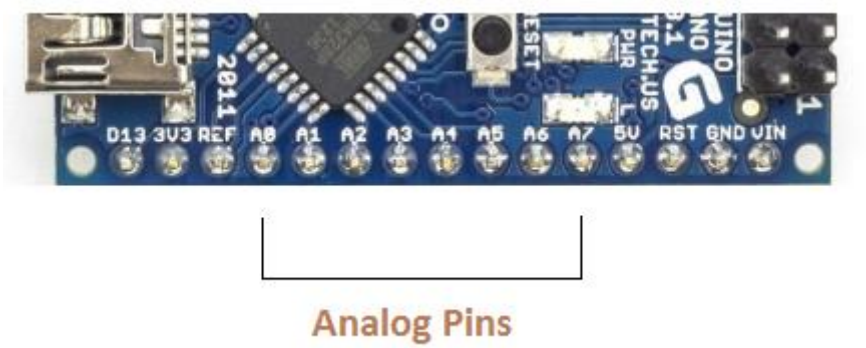
- **GND Pin:** These are the ground pins on the board.
- There are multiple ground pins on the board that can be interfaced accordingly when more than one ground pin is required.

- **Reset Pin:** Arduino Nano has 2 reset pins incorporated on the board, making any of these **Reset pins LOW** will reset the microcontroller.



- **Pin#13:** A built-in LED is connected to pin#13 of nano board.
- This LED is used to check the board i.e. it's working fine or not.
- **AREF:** This pin is used as a reference voltage for the input voltage.

- **Analog Pins:** There are 8 analog pins on the board marked as **A0 – A7**.



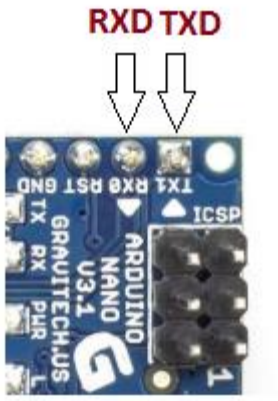
- These pins are used to measure the analog voltage ranging between **0 to 5V**.

- **Digital Pins**: Arduino Nano has 14 digital pins starting from D0 to D13.
- These digital pins are used for interfacing third-party digital sensors and modules with Nano board.
- **PWM Pins:** Arduino Nano has 6 PWM pins, which are Pin#3, 5, 6, 9, 10 and 11. (All are digital pins)
- These pins are used to generate an 8-bit PWM (Pulse Width Modulation) signal.
- **External Interrupts:** Pin#2 and 3 are used for generating external interrupts normally used in case of emergency, when we need to stop the main program and call important instructions.
- The main program resumes once interrupt instruction is called and executed.

- **Serial Pins:** These pins are used for serial communication where:

    1. Pin#0 is RX used for receiving serial data.

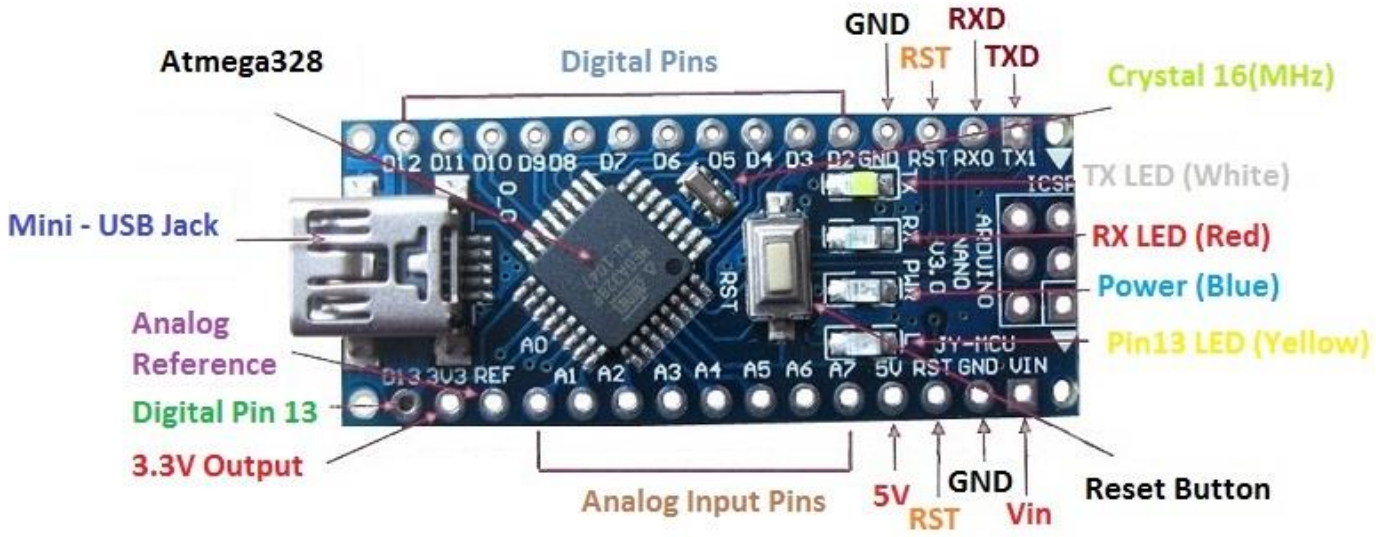
    2. Pin#1 is Tx used for transmitting serial data.



- **SPI Protocol:** Four pins 10(SS->Slave Select), 11(MOSI -> Master Out Slave In), 12(MISO -> Master In Slave Out) and 13(SCK -> Serial Clock) are used for SPI (Serial Peripheral Interface) Protocol.

- SPI is an interface bus and is mainly used to transfer data between microcontrollers and other peripherals like sensors, registers, and SD cards.

- **I2C Protocol:** I2C communication is developed using A4 and A5 pins, where **A4 represents the serial data line (SDA)** which carries the data and **A5 represents the serial clock line (SCL)** which is a clock signal, generated by the master device, used for data synchronization between the devices on an I2C bus.



Arduino Nano

Arduino Nano Programming & Communication

- The Nano board comes with the ability to set up communication with other controllers and computers.
- The serial communication is carried out by the digital pins, Pin 0(Rx) and Pin 1(Tx) where Rx is used for receiving data and Tx is used for the transmission of data.
- The serial monitor is added to the Arduino IDE, which is used to transmit textual data to or from the board.
- FTDI drivers are also included in the software which behaves as a virtual com port to the software.
- The Tx and Rx pins come with an LED which blinks as the data is transmitted between FTDI and USB connection to the computer.
- Arduino Software Serial Library is used for carrying out serial communication between the board and the computer.
- Apart from serial communication the Nano board also supports I2C and SPI communication. The Wire Library inside the Arduino Software is accessed to use the I2C bus.
- The Arduino Nano is programmed by Arduino Software called IDE which is a common software used for almost all types of board available. Simply download the software and select the board you are using.
- Uploading code to Arduino Nano is quite simple, as there's no need to use any external burner to compile and burn the program into the controller and you can also upload code by using ICSP (In-circuit serial programming header).
- Arduino board software is equally compatible with Windows, Linux or MAC, however, Windows are preferred to use.

Applications of Arduino Nano

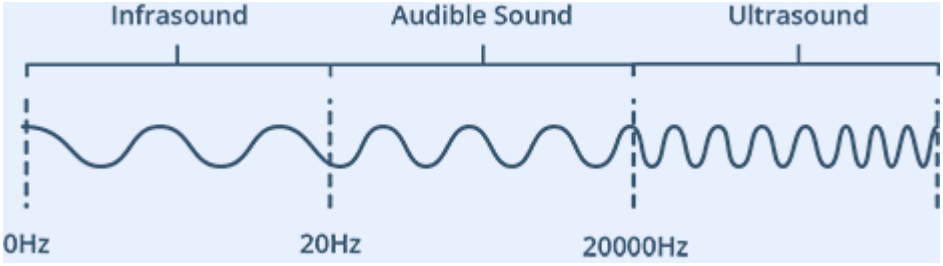
Arduino Nano is a very useful device that comes with a wide range of applications and covers less space as compared to other Arduino boards. Breadboard-friendly nature makes it stand out from other boards. Following are the main applications of Arduino Nano:

- Engineering Students' Projects.
- Medical Instruments
- Industrial Automation
- Android Applications
- GSM Based Projects
- Embedded Systems
- Automation and Robotics
- Home Automation and Defense Systems
- Virtual Reality Applications

HC-SR04 Ultrasonic Sensor Hardware Overview

What is Ultrasound?

Ultrasound is high-pitched sound waves with frequencies higher than the audible limit of human hearing.



Human ears can hear sound waves that vibrate in the range from about 20 times a second (a deep rumbling noise) to about 20,000 times a second (a high-pitched whistling). However, ultrasound has a frequency of over 20,000 Hz and is therefore inaudible to humans.

An ultrasonic sensor is a device that can measure the distance to an object by using sound waves. It measure distance by sending out a sound wave at a specific frequency and listening for that sound wave to bounce back. The ultrasonic transmitter an ultrasonic wave this wave travel in air and when it gets object by any material it gets reflected back toward the sensor this reflected wave is observed by the ultrasonic receiver module.

At its core, the HC-SR04 Ultrasonic distance sensor consists of two ultrasonic transducers. The one acts as a transmitter which converts 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 travelled. As simple as pie!

The sensor is small, easy to use in any robotics project and offers excellent non-contact range detection between 2 cm to 400 cm (that's about an inch to 13 feet) with an accuracy of 3mm. Since it operates on 5 volts, it can be hooked directly to an Arduino or any other 5V logic microcontrollers.

Here are complete specifications:

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

HC-SR04 Ultrasonic Sensor Pinout

HC-SR04 Pinout

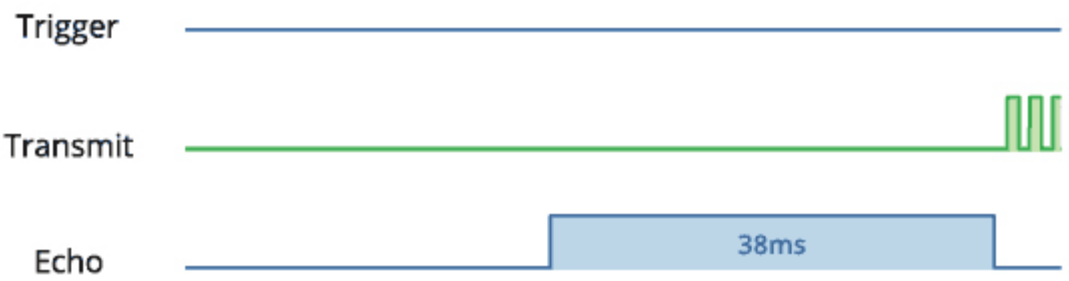
VCC is the power supply for HC-SR04 Ultrasonic distance sensor which we connect the 5V pin on the Arduino.
Trig (Trigger) pin is used to trigger the ultrasonic sound pulses.
Echo pin produces a pulse when the reflected signal is received. The length of the pulse is proportional to the time it took for the transmitted signal to be detected.
GND should be connected to the ground of Arduino.

How Does HC-SR04 Ultrasonic Distance Sensor Work?

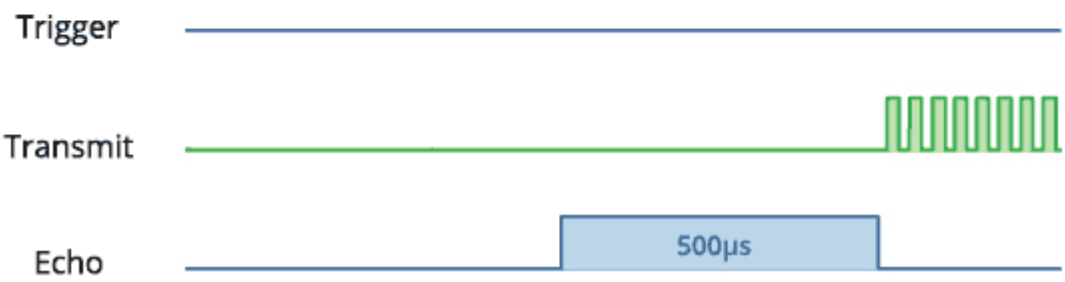
It all starts, when a pulse of at least 10 µS (10 microseconds) in duration is applied to the Trigger pin. In response to that the sensor transmits a sonic burst of eight pulses at 40 KHz. This 8-pulse pattern makes the "ultrasonic signature" from the device unique, allowing the receiver to differentiate the transmitted pattern from the ambient ultrasonic noise.

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.

In case, If those pulses are not reflected back then the Echo signal will timeout after 38 mS (38 milliseconds) and return low. Thus a 38 mS pulse indicates no obstruction within the range of the sensor.



If those pulses are reflected back the Echo pin goes low as soon as the signal is received. This produces a pulse whose width varies between 150 µS to 25 mS, depending upon the time it took for the signal to be received.



The width of the received pulse is then used to calculate the distance to the reflected object. This can be worked out using simple distance-speed-time equation, we learned in High school. In case you forgot, an easy way to remember the distance, speed and time equations is to put the letters into a triangle.

Distance = Speed x Time

$$\text{Time} = \frac{\text{Distance}}{\text{Speed}}$$

$$\text{Speed} = \frac{\text{Distance}}{\text{Time}}$$

Let's take an example to make it more clear. Suppose we have an object in front of the sensor at an unknown distance and we received a pulse of width 500 µS on the Echo pin. Now let's calculate how far the object from the sensor is. We will use the below equation.

$$\text{Distance} = \text{Speed} \times \text{Time}$$

Here, we have the value of Time i.e. 500 µs and we know the speed. What speed do we have? The speed of sound, of course! Its 340 m/s. We have to convert the speed of sound into cm/µs in order to calculate the distance. A quick Google search for "speed of sound in centimeters per microsecond" will say that it is 0.034 cm/µs. You could do the math, but searching it is easier. Anyway, with that information, we can calculate the distance!

$$\text{Distance} = 0.034\ \text{cm}/\mu\text{s} \times 500\ \mu\text{s}$$

But this is not done! Remember that the pulse indicates the time it took for the signal to be sent out and reflected back so to get the distance so, you'll need to divide your result in half.

$$\text{Distance} = (0.034\ \text{cm}/\mu\text{s} \times 500\ \mu\text{s}) / 2$$

$$\text{Distance} = 8.5\ \text{cm}$$

So, now we know that the object is 8.5 centimetres away from the sensor.

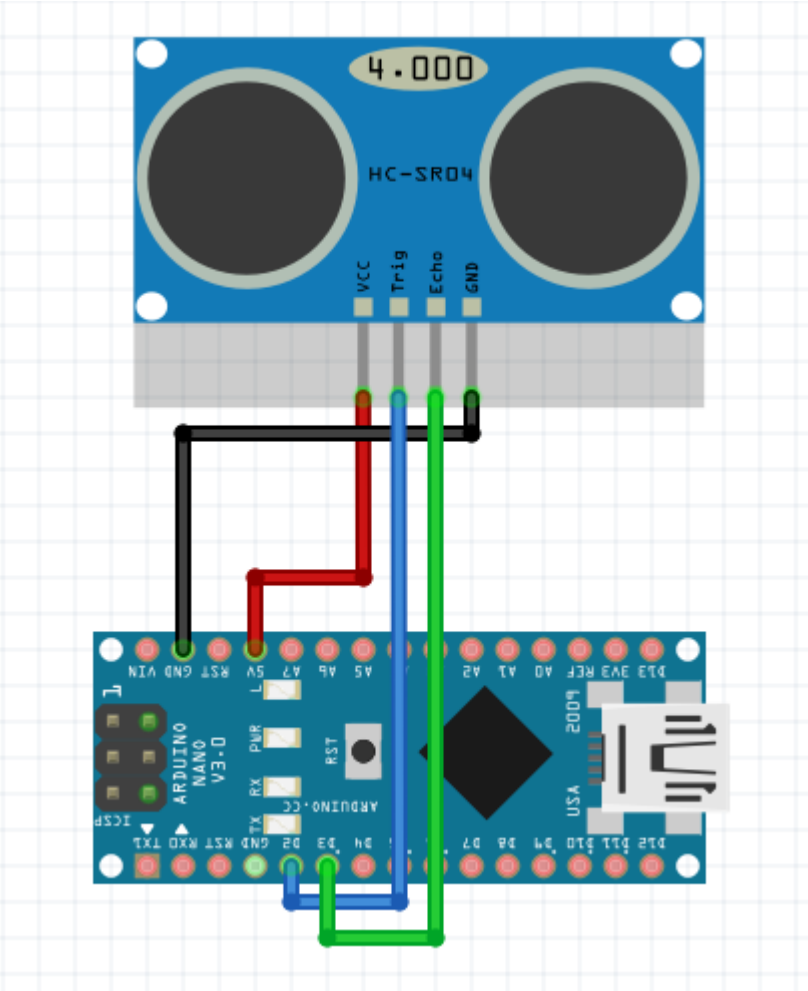
Wiring – Connecting HC-SR04 to Arduino Nano

Now that we have a complete understanding of how HC-SR04 ultrasonic distance sensor works, we can begin hooking it up to our Arduino.

Connecting the HC-SR04 to the Arduino is pretty easy.

Connect one GND pin of Arduino Nano with the GND of HC-SR04.
Connect 5V of Arduino Nano with the VCC pin of HC-SR04.

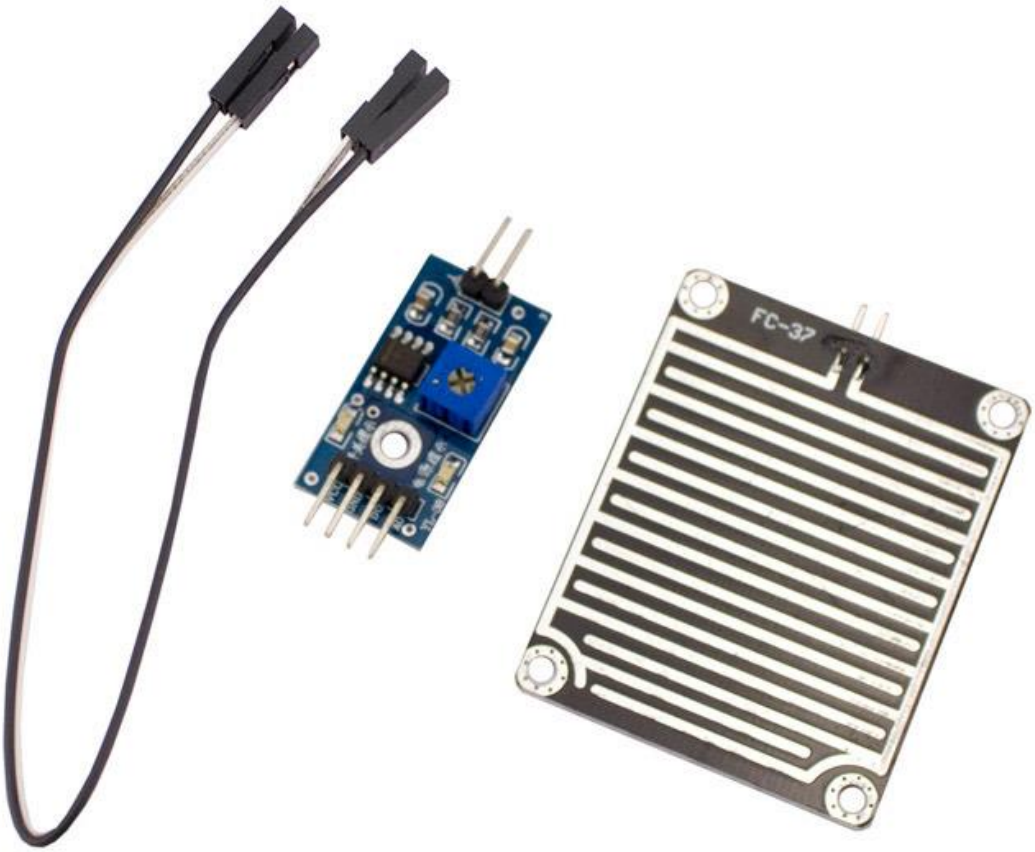
Connect ECHO pin of HC-SR04 to the D3 pin of Arduino Nano.
Connect the TRIG pin of HC-SR04 to the D2 pin of Arduino Nano.



3.5 Rain Sensors

The rain sensor module is and easy tool for rain detection. It can be used as a switch when raindrop falls through raining board and also for measuring rainfall intensity. The module features, a rain board and the control board that separated for more convenience power indicator LED and an adjustable sensitivity through a potentiometer.

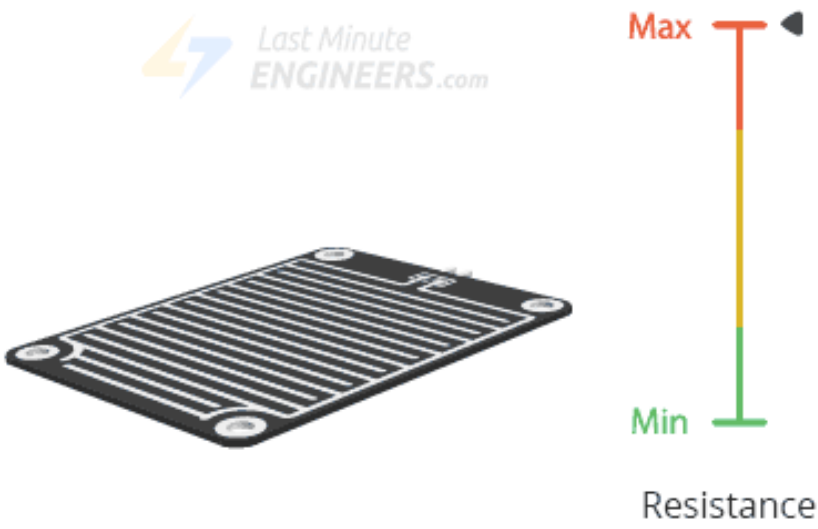
The rain sensor detects water that completes the circuits on its sensor board printed leads. The sensor board acts as a variable resistor that will change from 100 ohms when wet to 2M ohms when dry. In short the wetter the board the more current that will conducted.



How Rain Sensor works?

The working of the rain sensor is pretty straightforward.

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



This resistance is inversely proportional to the amount of water:

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

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

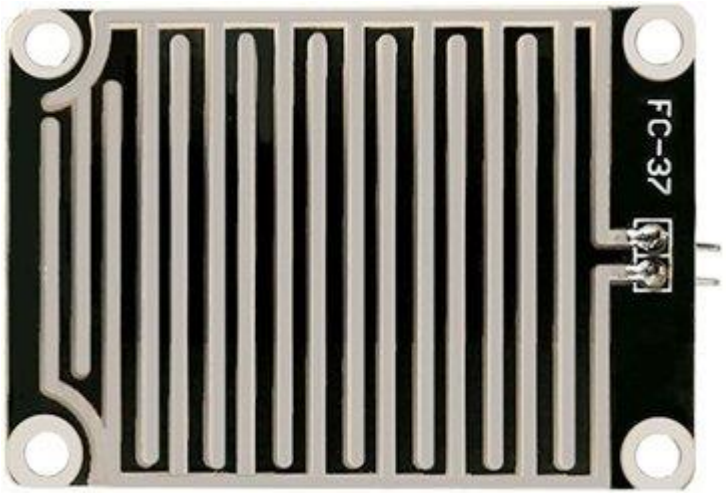
Hardware Overview

A typical rain sensor has two components.

The Sensing Pad

The sensor contains a sensing pad with series of exposed copper traces that is placed out in the open, possibly over the roof or where it can be affected by rainfall.

Usually, these traces are not connected but are bridged by water.



The Module

The sensor also contains an electronic module that connects the sensing pad to the Arduino.

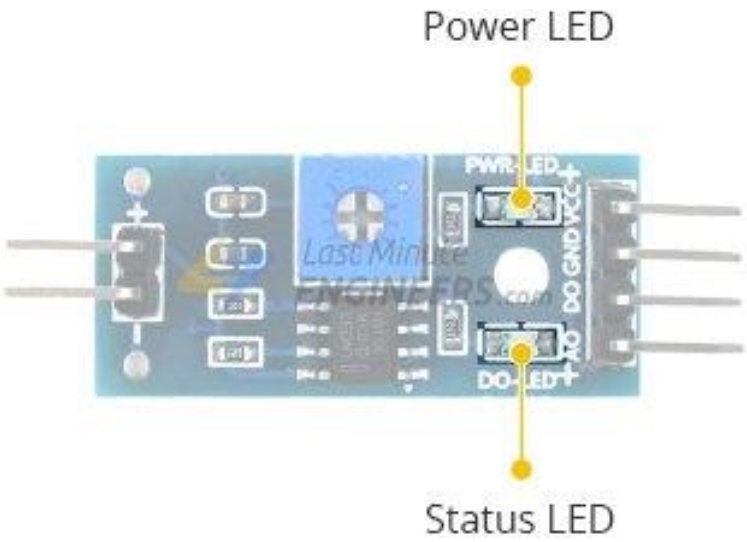
The module produces an output voltage according to the resistance of the sensing pad and is made available at an Analog Output (AO) pin.

The same signal is fed to a LM393 High Precision Comparator to digitize it and is made available at an Digital Output (DO) pin.

The module has a built-in potentiometer for sensitivity adjustment of the digital output (DO).

You can set a threshold by using a potentiometer; So that when the amount of water exceeds the threshold value, the module will output LOW otherwise HIGH.

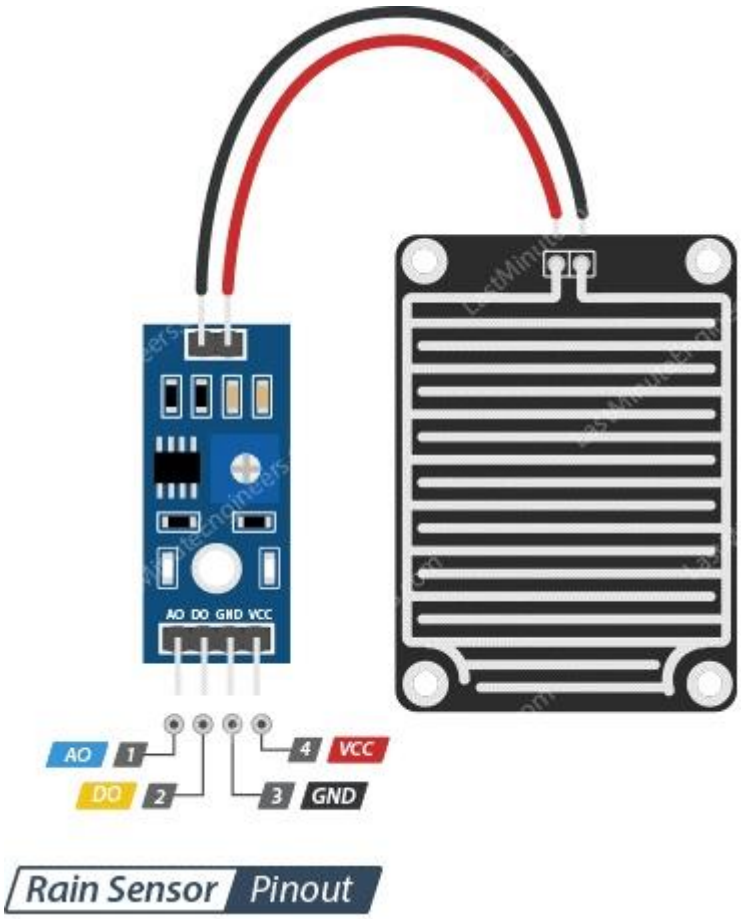
Rotate the knob clockwise to increase sensitivity and counterclockwise to decrease it.



Apart from this, the module has two LEDs. The Power LED will light up when the module is powered. The Status LED will light up when the digital output goes LOW.

Rain Sensor Pinout

The rain sensor is super easy to use and only has 4 pins to connect.



AO (Analog Output) pin gives us an analog signal between the supply value (5V) to 0V.

DO (Digital Output) pin gives Digital output of internal comparator circuit. You can connect it to any digital pin on an Arduino or directly to a 5V relay or similar device.

GND is a ground connection.

VCC pin supplies power for the sensor. It is recommended to power the sensor with between 3.3V – 5V. Please note that the analog output will vary depending on what voltage is provided for the sensor.

Wiring Rain Sensor with Arduino

Connect one GND pin of Arduino Nano with the GND of Rain Sensor.
Connect 5V of Arduino Nano with the VCC pin of Rain Sensor.
Connect AO pin of Rain Sensor to the A4 pin of Arduino Nano.

However, one commonly known issue with these sensors is their short lifespan

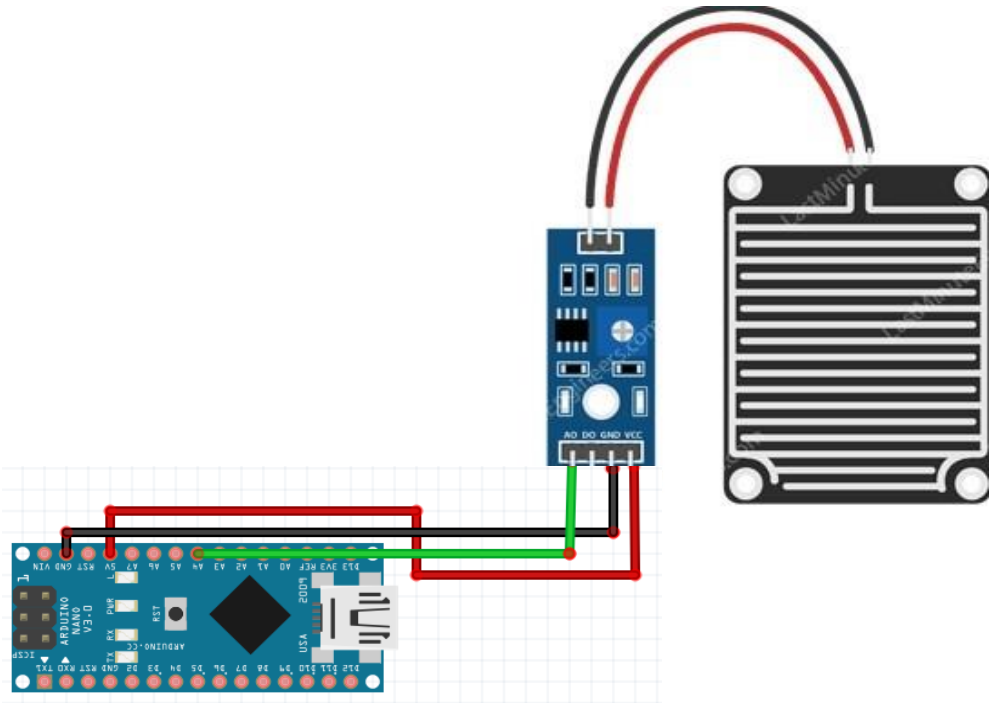
when exposed to a moist environment. Having power applied to the sensing pad constantly, speeds the rate of corrosion significantly.

To overcome this, we recommend that you do not power the sensor constantly, but power it only when you take the readings.

An easy way to accomplish this is to connect the VCC pin to a digital pin of an Arduino and set it to HIGH or LOW as as per your requirement.

Also the total power drawn by the module (with both LEDs lit) is about 8 mA, so it is okay to power the module off a digital pin on an Arduino.

The following illustration shows the wiring.



Buzzer

A buzzer is a small yet efficient component to add sound features to add sound to our project system. It is very small and compact 2 pin structure. Buzzer is in the lower portion of the audible frequency range of 20 Hz to 20 KHz. This is accomplished by covering an electric, oscillating signal in the audible range, into mechanical energy.

Buzzer

Wiring of Buzzer with Arduino

Connect D12 pin of Arduino Nano with the Negative pin of buzzer.
Connect A3 pin of Arduino Nano with the possitive pin of buzzer.

Jumper Wire

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

1. male to male,
2. male to female
3. female to male.

Jumper wire

SOFTWARE DESCRIPTION

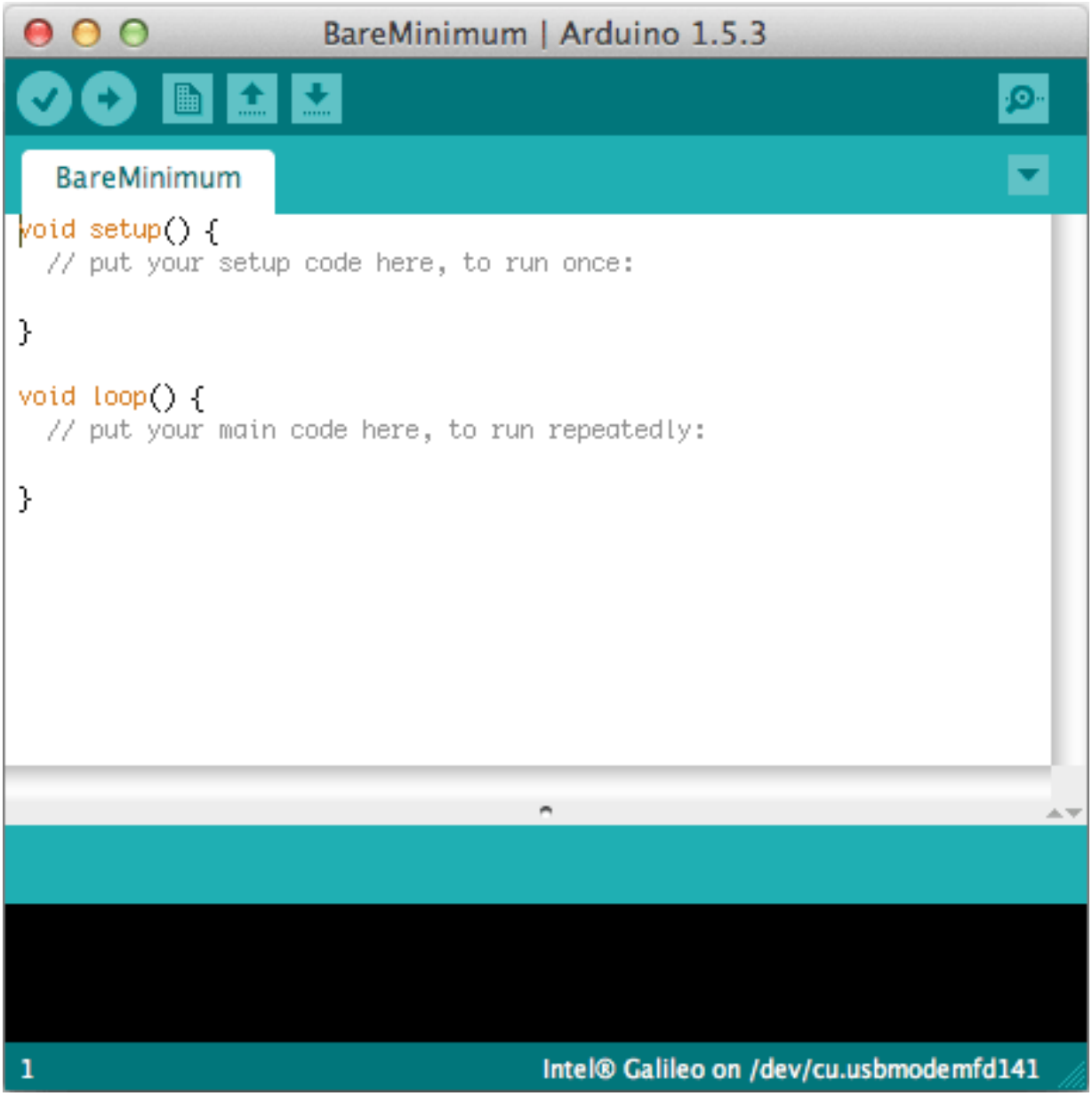
Arduino IDE

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, Mac OS and Linux) that is written in the programming language java. It is used and uploads programs to Arduino compatible boards.

The Arduino IDE supports the language C and C++ using special rules of codes structuring. The Arduino IDE supplies a software library from the wiring project which provides many common input and output input basic functions, for starting the sketch and the main program loop that are compiled and linked with a program

Arduino IDE is an open source that is mainly used for writing and compiling the code into the Arduino module. It is official software making code compilation too easy that even a common person with no prior technical knowledge can get their feet with the learning process. A different range of Arduino modules available including Arduino Uno, Arduino mega, Arduino Nano, and many more. Each of them consist a microcontroller on the board that is actually programmed and accepts the information in the form of code. 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.

Arduino IDE


BareMinimum | Arduino 1.5.3
BareMinimum
void setup() {
// put your setup code here, to run once:
}
void loop() {
// put your main code here, to run repeatedly:
}
1
Intel® Galileo on /dev/cu.usbmodemfd141

Arduino Project

SMART BLIND STICK WITH WATER DETECTION

A survey by WHO (World Health Organization) carried out in 2011 estimates that in the world, about 1% of the human population is visually impaired (about 70 million people) and amongst them, about 10% are fully blind (about 7 million people) and 90% (about 63 million people) with low vision. The main problem with blind people is how to navigate their way to wherever they want to go. Such people need assistance from others with good eyesight. As described by WHO, 10% of the visually impaired have no functional eyesight at all to help them move around without assistance and safely. This study proposes a new technique for designing a smart stick to help visually impaired people that will provide them navigation. The conventional and archaic navigation aids for persons with visual impairments are the walking cane (also called white cane or stick) and guide dogs which are characterized by a many imperfections. The most critical shortcomings of these aids include: essential skills and training phase, range of motion, and very insignificant information communicated been communicated. Our approach modified this cane with some electronics components and sensors, the electronic aiding devices are designed to solve such issues. The ultrasonic sensor, water sensor, buzzer are used to record information about the presence of obstacles on the road. Ultrasonic sensors have the capacity to detect any obstacle within the distance range of 2 cm-450 cm. Therefore whenever there is an obstacle in this range it will alert the user. Water sensor is used to detect if there is water in path of the user. Most blind guidance systems use ultrasound because of its immunity to the environmental noise. With the rapid advances of modern technology both in hardware and software it has become easier to provide intelligent navigation system to the visually impaired. Also, high-end technological solutions have been introduced recently to help blind persons navigate independently. Whenever the user wants to locate it, such a person will press a button on remote control and buzzer will ring, then the person can get the idea of where the stick is placed.

Vision is the most important part of human physiology as 83% of information human being gets from the environment is via sight. The 2011 statistics by the World Health Organization (WHO) estimates that there are 70 million people in the world living with visual impairment, 7 million of which are blind and 63 million with low vision. The conventional and oldest mobility aids for persons with visual impairments are characterized with many limitations. Some inventions also require a separate power supply or navigator which makes the user carry it in a bag every time they travel outdoor. These bulky designs will definitely make the user to be exhausted.

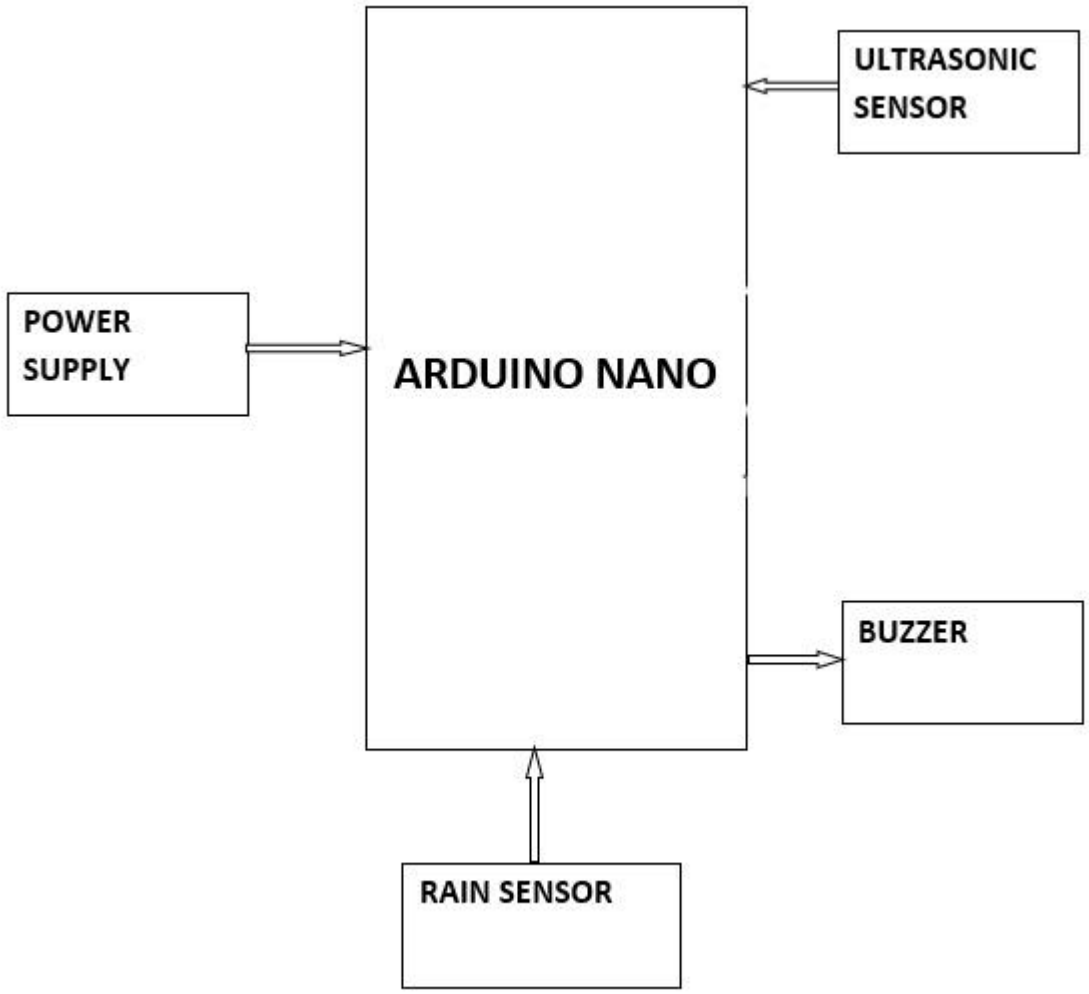
OBJECTIVES

Visually impaired people are the people who find it difficult to recognize the smallest detail with healthy eyes. The objectives of this research work include as follows:

1. To design an assistive technology for visually impaired people that can detect obstacles and provide alternative routes for the blind
2. To alarm the user through vibration to determine the obstacles direction sources
3. To help the user find his stick when he mistakenly loses it somewhere. Through this smart blind stick, visually impaired people will have so much of assistance.

BLOCK DIAGRAM

Figure:- Block diagram Smart Blind Stick



Block diagram description

The above block diagram shows the block diagram of our project. A DC 9voltage is supply though the power source to the Nano Arduino. All the sensor are interface with the Nano Arduino. Ultrasonic sensor is connected to the Arduino Nano then processes this data and calculates if the obstacle is close enough. If the obstacles are close the Nano Arduino sends a signal to sound a buzzer. It

also detected and sound a different buzzer if it where detects water and alerts the blind.

Circuit Diagram

Circuit diagram description :-

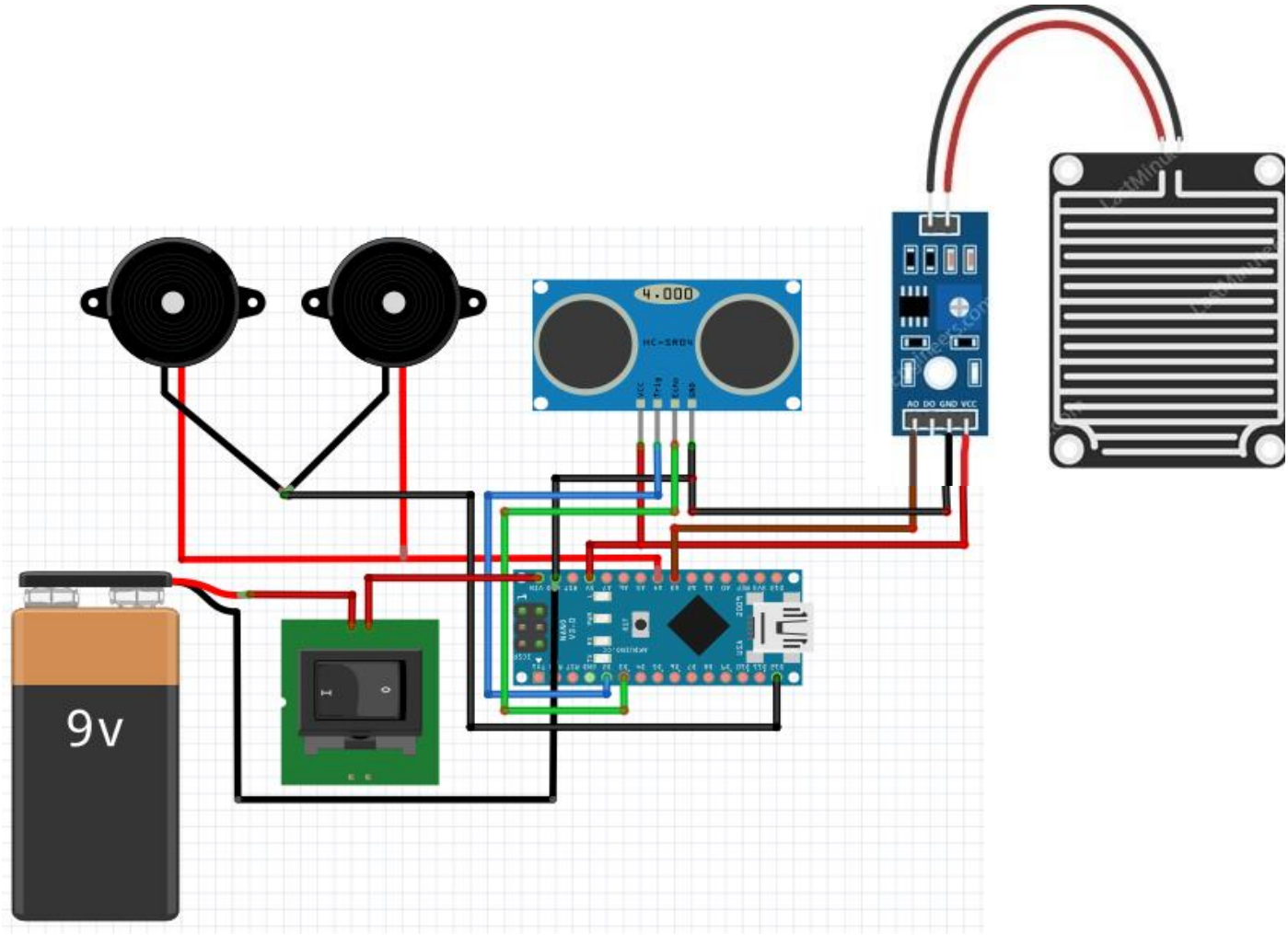


Figure:-Circuit diagram

The above figure shows the circuit diagram of Smart Blind Stick. We can see an **Arduino Nano** is used to control all the sensors.

The complete board is powered by a 9V battery which is regulated to +5V.

The **Ultrasonic Sensor** is powered by 5V and the trigger and Echo pin is connected to Arduino Nano pin D3 and D2 as shown above.

The **Rain sensor** is power by the 5v in VCC and Gnd is grounded and AO is connected to A4.

The output of the board is given by the **Buzzer** which is connected to pin D12 the negative of buzzer and A3 to the positive of buzzer.

Arduino Based project which consists of buzzer and Ultrasonic Sensor Module. The basic working principle of the Ultrasonic Sensor module is being described here. This Ultrasonic Sensor sends a signal through a trigger pin which when blocked by an obstacle returns high at the echo pin.

The time taken between this interval is used to calculate the distance in cm. The buzzer produces a sound that keeps on increasing as the distance from obstacle gets less after a certain set minimum distance limit.

SOURCE CODE

```
//Start of program

#include <SoftwareSerial.h>

#define pingPin 2 //trig pin of sr04
#define echoPin 3

void setup()
{
    Serial.begin(9600); // Starting Serial Terminal
    pinMode(pingPin, OUTPUT);
    pinMode(echoPin, INPUT);
    pinMode(12, OUTPUT); //pin12 is used as GND pin  for buzzer since arduino
nano has only two GND pins
    pinMode(A3, OUTPUT); //pin A3 provides the output on buzzer
    pinMode(A4, INPUT);  //pin A4 for Water Sensor input
}

void loop()
{
    long duration, cm;
    digitalWrite(12, LOW); //Buzzer GND is always low

    //send a signal at ping pin at an interval of 0.002 seconds to check for
an object
    digitalWrite(pingPin, LOW);
    delayMicroseconds(2);
    digitalWrite(pingPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(pingPin, LOW);

    duration = pulseIn(echoPin, HIGH); //check time using pulseIn function

    cm = microsecondsToCentimeters(duration); //functin call to find distance

    Serial.print("Distance : ");
    Serial.print(cm);
    Serial.print(" cm");
    Serial.println();
    delay(100);

    if (cm < 30 && cm > 20)
    {
        analogWrite(A3, 255);
        delay(1000);
        Serial.println("Alarming the Buzzer!!");
```

```
        analogWrite(A3, 0);
        delay(1000);
    } //sound buzzer every second if obstacle distance is between 20-30cm.

    else if (cm < 20 && cm > 10)
    {
        analogWrite(A3, 255);
        delay(500);
        Serial.println("Alarming the Buzzer!!");
        analogWrite(A3, 0);
        delay(500);
    } //sound buzzer every 0.5 seconds if obstacle distance is between 10-
20cm.

    else if (cm < 10 && cm > 0)
    {
        analogWrite(A3, 255);
        delay(100);
        Serial.println("Alarming the Buzzer!!");
        analogWrite(A3, 0);
        delay(100);
    } //sound buzzer every 0.1 seconds if obstacle distance is between 0-10cm.

    else
        analogWrite(A3, 0); //do not sound the buzzer

    int sensorValue = digitalRead(A4);

    if (sensorValue == 0)
    {
        Serial.print("Water Detected : ");
        Serial.println(sensorValue);
        analogWrite(A3, 255);
        delay(1500);
        Serial.println("Alarming the Buzzer!!");
        analogWrite(A3, 0);
        delay(1500);
    } //rain sensor
}

//function to return distance in cm from microseconds
long microsecondsToCentimeters(long microseconds)
{
    return microseconds / 29 / 2;
}

//End of Program
```

ADVANTAGES & LIMITATIONS

Advantage of blind stick is listed below:-

1. This system consists of different type of the sensor, which is used to measure the distance and alert the blind people.

2. It is simple to use and is affordable.

3. This system can navigate the location of the blind people when they find themselves in danger or some adverse situations.

4. Smart blind stick is robust and the stick is light, portable and reliable.

5. It consumes low power which makes it feasible to use.

Limitations of blind stick are listed below:-

1. Pits and bumps of the road cannot be detected using this device.

2. Smart stick is unfoldable.

3. As the sensor is sensitive, it should be handled in utmost care and prevented from contact with water.

APPLICATIONS

Following are the application of Smart Blind Stick

Facilitates the visually-impaired people through various user-friendly features such as water detection, Navigation, Obstacle alert and communication.

FUTURE IMPROVEMENTS

The certain modification of sensor and programming we can detect the small pits and dumps of road.

Water resistant can be eliminated

PROBLEM FORMULATION

While researching for this project, many problems were there. First problem was the unavailability of the materials required for the project. Because of this, much of the time was invested to search for them and talk to people outside Biratnagar. Secondly, there were some issues regarding interface. Other than that, resources could be utilized to make the research better.

CONCLUCSION

It is worth mentioning at this point that the aim of the of this study is design and implementation of a smart walking stick for the blind has been fully achieved. The smart stick as a basic platform for the coming generation of more adding devices to help the visually impaired to navigate safely both indoor and outdoor. It is effective and affordable. It leads the good result in detecting the obstacles on the path of the user in a range. This project offers low cost, reliable, portable, low power consumption and robust technology for navigation with obvious short response time. In this project, different types of sensors and other component with the light weight and the rain sensor is used to detect the water.

Biblography

https://www.theengineeringprojects.com/2018/06/introduction-to-arduino-nano.html

https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/

https://lastminuteengineers.com/rain-sensor-arduino-tutorial/

https://electronicsworkshops.com/2020/06/14/smart-blind-stick-using-gsm-module-gsm-moduleultrasonic-sensor-and-rain-sensor/