**VEHICLE TO VEHICLE COMMUNICATION USING LORA**

**OBJECTIVE:**

The objective of this project is to develop a vehicle communication system using Lora technology, integrating sensors for detecting gas, fire, and acceleration. The system aims to provide real-time data transmission from one vehicle to another, enhancing safety and awareness on the road. By employing Internet of Things (IoT) capabilities, the project aims to alert vehicle owners about potential hazards or anomalies detected by the sensors.

**ABSTRACT:**

In today's world, ensuring road safety is paramount. This project introduces a novel approach to vehicle communication utilizing LoRa (Long Range) technology. The system consists of two main components: a transmitter and a receiver. The transmitter incorporates sensors for gas, fire, and acceleration detection, alongside a button for manual status indication. Upon sensing an anomaly, data is transmitted via LoRa to the receiver unit. The receiver, equipped with an LCD display and a buzzer, promptly alerts the vehicle owner. Through this setup, critical information about the vehicle's surroundings can be efficiently communicated, aiding in accident prevention and overall road safety.

**INTRODUCTION:**

The integration of advanced technologies in the automotive sector has significantly contributed to enhancing safety standards. In line with this trend, our project proposes the implementation of a vehicle communication system utilizing LoRa technology. LoRa, known for its long-range and low-power characteristics, offers an ideal solution for transmitting data between vehicles in real-time, overcoming the limitations of traditional communication methods.

The project comprises two interconnected modules: the transmitter and the receiver. The transmitter module is responsible for collecting data from sensors monitoring gas, fire, and acceleration levels. Additionally, a manual status indication button allows users to convey their vehicle's condition instantly. Upon detection of an anomaly, the transmitter transmits this data using Lora communication to the receiver unit.

On the receiving end, the receiver module decodes the transmitted data and displays it on an LCD screen. Simultaneously, an audible alert, generated by a buzzer, notifies the vehicle owner of the detected anomaly. This seamless communication loop ensures prompt awareness and action, enabling vehicle operators to respond effectively to potential hazards.

Through the integration of IoT capabilities, our system goes beyond simple sensor data transmission, enabling remote monitoring and alerting. By providing timely and accurate information about the vehicle's surroundings, the project aims to enhance road safety and contribute to the realization of a smarter, safer automotive ecosystem.

**PROBLEM STATEMENT**

* Current vehicle communication systems lack real-time data exchange capabilities and often rely on short-range technologies. This hinders driver awareness of surrounding hazards, potentially leading to accidents.

**LITERATURE SURVEY**

| **SI.NO** | **TITLE OF THE PROJECT** | **AUTHOR** | **DRAWBACKS** |
| --- | --- | --- | --- |
| 1. | Feature-Based Vehicle Identification Framework for  Optimization of Collective Perception Messages in  Vehicular Networks | HIDETAKA MASUDA, OUSSAMA EL MARAI , MANABU TSUKADA, TARIK TALEB  AND HIROSHI ESAKI | Limited range |
| 2. | Computationally Efficient Nonlinear One- and  Two-Track Models for Multi trailer Road Vehicles | TOHEED GHANDRIZ,  BENGT JACOBSON,  PETER NILSSON,  LEO LAINE AND NIKLAS FRÖJD | Scalability and Compatibility |
| 3. | An Improved Motion Control With Cyber-Physical Uncertainty Tolerance for Distributed Drive  Electric Vehicle | WANKE CAO , ZHIWEN ZHU, JINRUI NAN | Power Consumption |

**EXISTING SYSTEM:**

Vehicle communication systems lacked efficient long-range transmission capabilities and comprehensive sensor integration. Traditional systems primarily relied on short-range technologies like Bluetooth or Wi-Fi, limiting their effectiveness in transmitting data between vehicles. Sensor integration was also minimal, often focusing on basic functions like collision detection. Moreover, IoT capabilities were not extensively utilized for real-time monitoring and alerting. Overall, the existing systems lacked the robustness and flexibility required to provide comprehensive vehicle-to-vehicle communication for enhancing road safety.

**DISADVANTAGES:**

1. Limited Range
2. Interference Issues
3. Data Security Concerns
4. High Power Consumption
5. Limited Sensor Integration
6. Lack of Real-time Monitoring
7. Complexity and Cost

**PROPOSED SYSTEM:**

The proposed system introduces a vehicle communication solution leveraging LoRa technology, offering extended range transmission and low-power consumption. Integrated sensors for gas, fire, and acceleration detection enhance safety monitoring capabilities, while a manual status indication button provides additional context. Utilizing IoT functionalities, the system enables real-time data transmission to a receiver unit, equipped with an LCD display and buzzer for immediate alerts. By combining LoRa's advantages with comprehensive sensor integration and IoT capabilities, the proposed system aims to revolutionize vehicle-to-vehicle communication, fostering enhanced road safety and awareness.

**ADVANTAGES:**

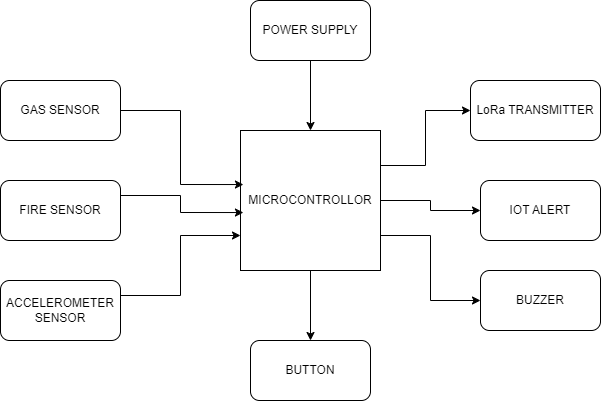
1. Extended Range Transmission
2. Low Power Consumption
3. Comprehensive Sensor Integration
4. Real-time Monitoring and Alerting
5. Manual Status Indication
6. Cost-effectiveness
7. Scalability and Flexibility

TECHNOLOGIES USED IN PROPOSED SYSTEM

* LoRa Low-power, long-range wireless communication technology
* Sensors Gas, fire, and acceleration sensors for hazard detection
* Microcontroller Processes sensor data and facilitates LoRa communication
* LCD Display (Receiver)  Visually displays received data
* Buzzer (Receiver)  Provides audible alerts for detected anomalies

**BLOCK DIAGRAM (HARDWARE):**

**TRANSMITTER PART:**



**RECEIVER PART:**

****

**BLOCK DIAGRAM DESCRIPTION:**

The block diagram for the proposed vehicle communication system consists of two main sections: the transmitter and the receiver. The transmitter section includes gas, fire, and accelerometer sensors, along with a manual status indication button. Data from these sensors is processed by a microcontroller and transmitted via a LoRa transmitter. The receiver section comprises a LoRa receiver, which receives the transmitted data and forwards it to another microcontroller. An LCD display and a buzzer are connected to the receiver microcontroller to provide real-time alerts to the vehicle owner. This modular setup ensures efficient data transmission and immediate response to detected anomalies, enhancing overall road safety.

**SOFTWARE REQUIREMENTS:**

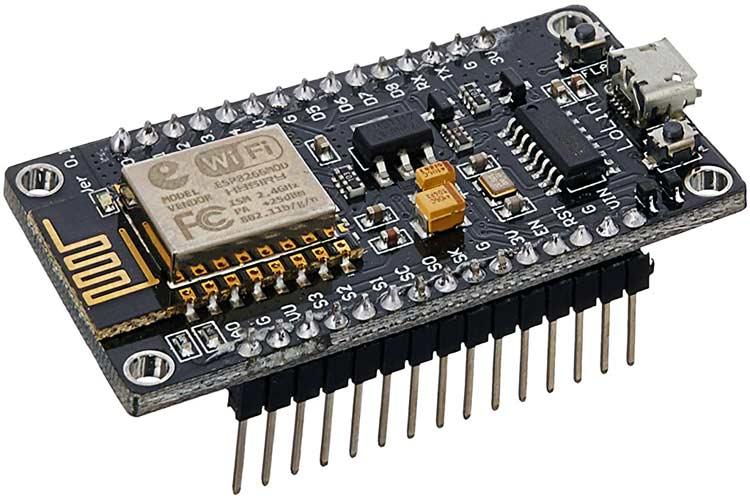
* Arduino ide
* Embedded C
* IOT

**HARDWARE REQUIREMENTS:**

* Microcontroller
* Gas sensor
* Fire sensor
* Accelerometer sensor
* LoRa Transmitter
* LoRa Receiver
* I2C LCD
* Button
* Buzzer

**HARDWARE MODULE:**

**NODE MCU**

****

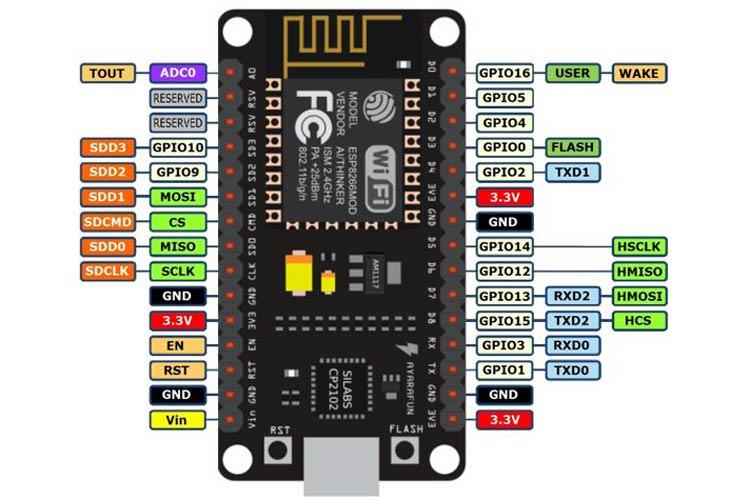
NodeMCU is an open-source Lua based firmware and **development board** specially targeted for IoT based Applications. It includes firmware that runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.

NodeMCU Development Board Pinout Configuration

| **Pin Category** | **Name** | **Description** |
| --- | --- | --- |
| Power | Micro-USB, 3.3V, GND, Vin | **Micro-USB:** NodeMCU can be powered through the USB port    **3.3V:** Regulated 3.3V can be supplied to this pin to power the board    **GND:** Ground pins    **Vin:**External Power Supply |
| Control Pins | **EN, RST** | The pin and the button resets the microcontroller |
| Analog Pin | A0 | Used to measure analog voltage in the range of 0-3.3V |
| GPIO Pins | GPIO1 to GPIO16 | NodeMCU has 16 general purpose input-output pins on its board |
| SPI Pins | SD1, CMD, SD0, CLK | NodeMCU has four pins available for SPI communication. |
| UART Pins | TXD0, RXD0, TXD2, RXD2 | NodeMCU has two UART interfaces, UART0 (RXD0 & TXD0) and UART1 (RXD1 & TXD1). UART1 is used to upload the firmware/program. |
| I2C Pins |  | NodeMCU has I2C functionality support but due to the internal functionality of these pins, you have to find which pin is I2C. |

NodeMCU ESP8266 Specifications & Features

* Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106
* Operating Voltage: 3.3V
* Input Voltage: 7-12V
* Digital I/O Pins (DIO): 16
* Analog Input Pins (ADC): 1
* UARTs: 1
* SPIs: 1
* I2Cs: 1
* Flash Memory: 4 MB
* SRAM: 64 KB
* Clock Speed: 80 MHz
* USB-TTL based on CP2102 is included onboard, Enabling Plug n Play
* PCB Antenna
* Small Sized module to fit smartly inside your IoT project

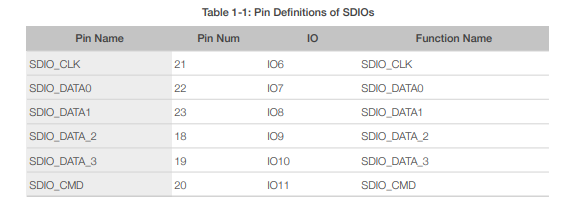


**General Purpose Input/Output Interface (GPIO)**

ESP8266EX has 17 GPIO pins which can be assigned to various functions by programming the appropriate registers. Each GPIO can be configured with internal pull-up or pull-down, or set to high impedance, and when configured as an input, the data are stored in software registers; the input can also be set to edge-trigger or level trigger CPU interrupts. In short, the IO pads are bidirectional, non-inverting and tristate, which includes input and output buffer with tristate control inputs. These pins can be multiplexed with other functions such as I2C, I2S, UART, PWM, IR Remote Control, etc.

**Secure Digital Input/Output Interface (SDIO)**

ESP8266EX has one Slave SDIO, the definitions of which are described below. 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported.



**Serial Peripheral Interface (SPI/HSPI)**

ESP8266EX has 3 SPIs.

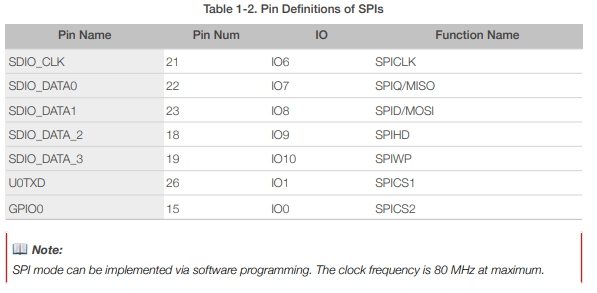
One general Slave/Master SPI

One Slave SDIO/SPI

One general Slave/Master HSPI

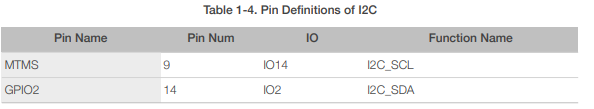
Functions of all these pins can be implemented via hardware. The pin definitions are described as below.

**General SPI (Master/Slave)**



I2C Interface

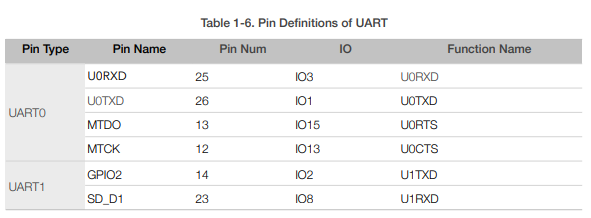
ESP8266EX has one I2C used to connect with micro-controller and other peripheral equipments such as sensors. The pin definition of I2C is as below.



Both I2C Master and I2C Slave are supported. I2C interface functionality can be realized via software programming, the clock frequency reaches 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.

**Universal Asynchronous Receiver Transmitter (UART)**

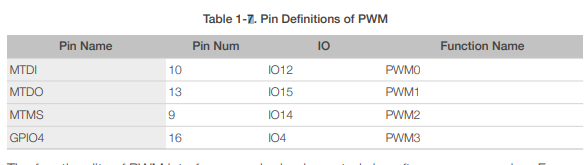
ESP8266EX has two UART interfaces UART0 and UART, the definitions are as below



Data transfers to/from UART interfaces can be implemented via hardware. The data transmission speed via UART interfaces reaches 115200 x 40 (4.5 Mbps). UART0 can be used for communication. It supports fluid control. Since UART1 features only data transmit signal (Tx), it is usually used for printing log.

**Pulse-Width Modulation (PWM)**

ESP8266EX has four PWM output interfaces. They can be extended by users themselves. The pin definitions of the PWM interfaces are defined as below.



The functionality of PWM interfaces can be implemented via software programming. For example, in the LED smart light demo, the function of PWM is realized by interruption of the timer, the minimum resolution reaches as much as 44 ns. PWM frequency range is adjustable from 1000 μs to 10000 μs, i.e., between 100Hz and 1 kHz. When the PWM frequency is 1 kHz, the duty ratio will be 1/22727, and over 14 bit resolution will be achieved at 1 kHz refresh rate.

**Functional Overview**

This protocol uses the SDIO mode of the ESP8266 to communicate with other processor's SPI hosts. The electrical interface is connected through signal line No.4, including the SCLK, MOSI, MISO and interrupt signal No.1 in the SPI protocol (note: no CS signal). Downloading the ESP8266 SDIO can be different from downloading other programs. When the ESP8266 starts, the system reads the pin shared by the SPI interface and the SDIO interface by default. Therefore, the SDIO module communication protocol should be used. The ESP8266 should start in the SDIO mode, and then, the host will start the chip in the ESP8266 RAM through the SDIO downloaded programs. The majority of the programs that directly use CPU CACHE to call FLASH can be burnt to the FLASH chip connected to the HSPI interface beforehand. Data received or sent by the ESP8266 SDIO is processed directly by the DMA module that supports linked list index. The ESP8266 can receive and send the SDIO packets efficiently without using the CPU. It does so through the address of the memory map linked list.

and re-transfer sketch, prior to interfacing outside voltage to AREF

**SOFTWARE TIPS**

While bootloadingna Atmega8 chip with Arduino 0010, there is an order (- i800) that makes bootloader defer 10 minutes. Thus, in the event that you need to utilize bootloader, use order line rather than IDE, eliminating "– i800" order and adding "– F" order, or use Arduino 0007 IDE. To transfer draws Arduino 0010 turns out great.

**ARDUINO S3v3 NEW FEATURES**

* full viable with Shield Boards (Version 2 is the main Arduino Board not viable with Shield Boards as a result of ICSP header wrong position, and tall parts);
* AVcc LP channel to lessen commotion level on ADC;
* auto reset include;
* auto reset empower/impair jumper, to dodge not wanted reseting;
* arduinoDiecimila viable reset pin;
* pin13 locally available drove, with current limiter resistor;
* TX and RX locally available leds;
* power drove with suitable current limiter resistor (less 20mA of comsumption);
* jumper to impair sequential correspondence and to empower RX outer draw down resistor, to evade "RX skimming blunder". This element permits to utilize computerized pin0 and pin1 as an ordinary pin, when sequential correspondence isn't required;
* all comparative segments (diodes, semiconductors, leds, capacitors) has a similar board direction (to commits simpler to mount with less errors);
* no wires between cushions, more space between wires, bigger wires, bigger cushions (better for drawing, binding and penetrating, with no shortcircuits, patching extensions or open wires in erosion);
* just 3 wire spans;
* electrolitic capacitor (in sequential to TTL circuit) changed to bipolar sort (to keep away from rearranged voltage issue when sequential link isn't associated);

All jumpers are correct point type, to permit Shield Boards use.

**GAS SENSOR**

Ideal sensor for use to detect the presence of a dangerous LPG leak in your car or in a service station, storage tank environment. This unit can be easily incorporated into an alarm unit, to sound an alarm or give a visual indication of the LPG concentration. The sensor has excellent sensitivity combined with a quick response time. The sensor can also sense iso-butane, propane, LNG and cigarette smoke.

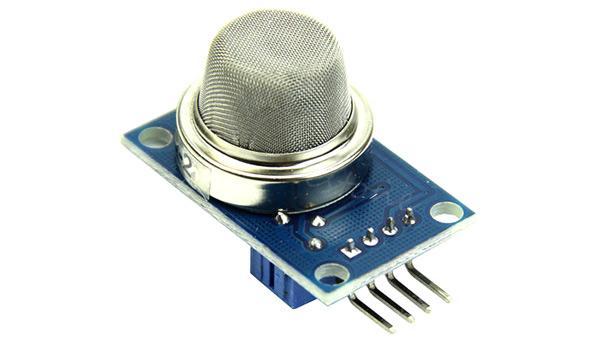


A Typical human nose has 400 types of scent receptors enabling us to smell about 1 trillion different odours. But still many of us do not have the capacity to identify the type or concentration of gas present in our atmosphere. This is where Sensors comes in, there are many [types of sensors](https://components101.com/sensors/mq-6-gas-sensor-pinout-equivalent-datasheet) to measure different parameters and a Gas sensor is one which comes handy in applications where we have to detect the variation in the concentration of toxic gases in order to maintain the system safe and avoid/caution any unexpected threats. There are various gas sensors to detect gases like oxygen, Carbon Dioxide, Nitrogen, methane etc. They can also be commonly found in devices that are used to detect the leakage of the harmful gases, monitor the air quality in industries and offices etc.

In this article, we will learn more about gas sensors, their construction, types, working and how they can be used to measure the required type and concentration of Gas in our atmosphere. There are [many types of Gas sensors](https://components101.com/tags/gas-sensor) but the MQ type gas sensors are commonly used and widely popular so will focus more on these types of sensors for this article.

**Introduction**

A gas sensor is a device which detects the presence or concentration of gases in the atmosphere. Based on the concentration of the gas the sensor produces a corresponding potential difference by changing the resistance of the material inside the sensor, which can be measured as output voltage. Based on this voltage value the type and concentration of the gas can be estimated.



Gas Sensor

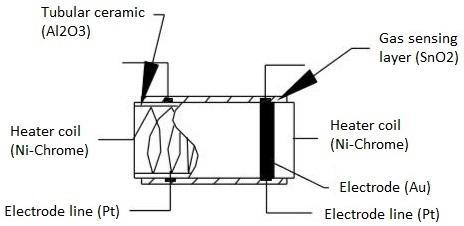
The type of gas the sensor could detect depends on the sensing material present inside the sensor. Normally these sensors are available as modules with comparators as shown above. These comparators can be set for a particular threshold value of gas concentration. When the concentration of the gas exceeds this threshold the digital pin goes high. The analog pin can be used to measure the concentration of the gas.

**Construction**

All Of the above-listed types, the most commonly used gas sensor is the Metal oxide semiconductor based gas sensor. All Gas sensors will consist of a sensing element which comprises of the following parts.

1. Gas sensing layer
2. Heater Coil
3. Electrode line
4. Tubular ceramic
5. Electrode

The below image illustrates the parts present in a metal oxide gas sensor



The purpose of each of these elements is as below:

**Gas sensing layer:** It is the main component in the sensor which can be used to sense the variation in the concentration of the gases and generate the change in electrical resistance. The gas sensing layer is basically a chemiresistor which changes its resistance value based on the

The concentration of particular gas in the environment. Here the sensing element is made up of a Tin Dioxide (SnO2) which is, in general, has excess electrons (donor element). So whenever toxic gases are being detected the resistance of the element changes and the current flown through it varies which represents the change in concentration of the gases.

**Heater coil:** The purpose of the heater coil is to burn-in the sensing element so that the sensitivity and efficiency of the sensing element increases. It is made of Nickel-Chromium which has a high melting point so that it can stay heated up without getting melted.

**Electrode line:** As the sensing element produces a very small current when the gas is detected it is more important to maintain the efficiency of carrying those small currents. So Platinum wires come into play where it helps in moving the electrons efficiently.

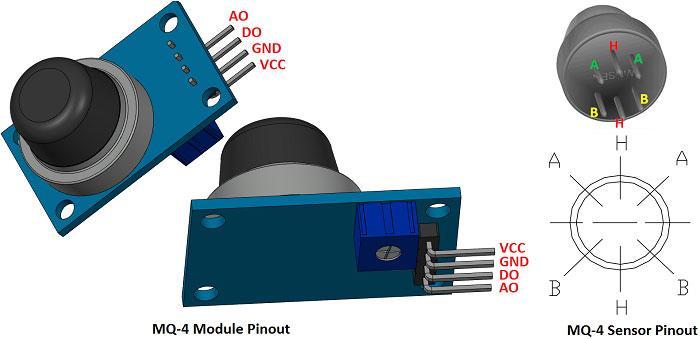
**Electrode:** It is a junction where the output of the sensing layer is connected to the Electrode line. So that the output current can flow to the required terminal. An electrode here is made of Gold (Au –Aurum) which is a very good conductor.

**Tubular ceramic:** In between the Heater coil and Gas sensing layer, the tubular ceramic exists which is made of Aluminum oxide (Al2O3). As it has high melting point, it helps in maintaining the burn-in (preheating) of the sensing layer which gives the high sensitivity for the sensing layer to get efficient output current.

**Mesh over the sensing element:** In order to protect the sensing elements and the setup, a metal mesh is used over it, which is also used to avoid/hold the dust particles entering into the mesh and prevent damaging the gas sensing layer from corrosive particles.

**How to use a Gas sensor?**

A basic gas sensor has 6 terminals in which 4 terminals (A, A, B, B) acts input or output and the remaining 2 terminals (H, H) are for heating the coil. Of these 4 terminals, 2 terminals from each side can be used as either input or output (these terminals are reversible as shown in the circuit diagram) and vice versa.



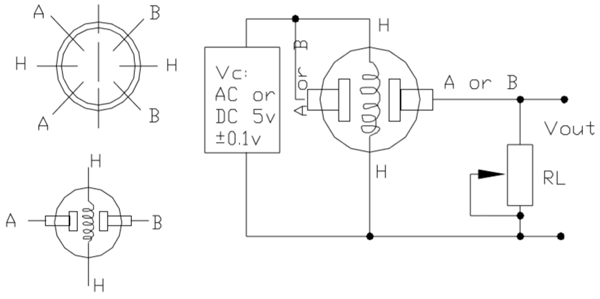
These sensors are normally available as modules (shown right), these modules consist of the gas sensor and a [comparator IC](https://components101.com/tags/voltage-comparator). Now let’s see the pin description of the gas sensor module which we will generally use with an Arduino. The gas sensor module basically consists of 4 terminals

* Vcc – Power supply
* GND – Power supply
* Digital output – This pin gives an output either in logical high or logical low (0 or 1) that means it displays the presence of any toxic or combustible gases near the sensor.
* Analog output – This pin gives an output continuous in voltage which varies based on the concentration of gas that is applied to the gas sensor.

As discussed earlier the output of a gas sensor alone will be very small (in mV) so an external circuit has to be used in order to get a digital high low output from the sensor. For this purpose, a comparator ([LM393](https://components101.com/ics/lm393-low-offset-voltage-dual-comparators)), [adjustable potentiometer](https://components101.com/potentiometer), [some resistors](https://components101.com/resistors) and [capacitors](https://components101.com/capacitors) are used.

The purpose of LM393 is to get the output from the sensor, compare it with a reference voltage and display whether the output is logically high or not. Whereas the purpose of the potentiometer is to set the required threshold value of the gas above which the digital output pin should go high.

The below diagram shows the basic circuit diagram of a gas sensor in a gas sensor module



Here A and B are the input and output terminals (these are reversible - means any of the paired terminals can be used as input or output) and H is the Heater coil terminal. The purpose of the variable resistor is to adjust the output voltage and to maintain high sensitivity.

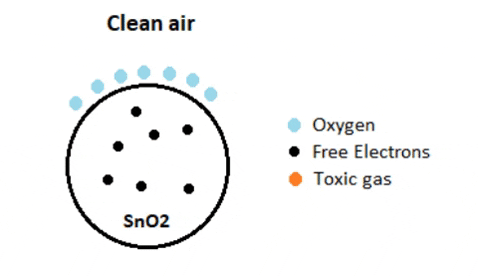
If no input voltage is applied to the heater coil, then the output current will be very less (which is negligible or approximately 0). When sufficient voltage is applied to the input terminal and heater coil, the sensing layer wakes up and is ready to sense any combustible gases nearby it. Initially let’s assume that there is no toxic gas near the sensor, so the resistance of the layer doesn’t change and the output current and voltage are also unchanged and are negligible (approximately 0).

Now let’s assume that there is some toxic gas nearby. As the heater coil is pre-heated it is now easy to detect any combustible gases. When the sensing layer interacts with the gases, the resistance of the material varies and the current flowing through the circuit also varies. This change in variation can be then observed at the load resistance (RL).

The value of load resistance (RL) can be anywhere from 10KΩ to 47KΩ. The exact value of the load resistance can be selected by calibrating with the known concentration of the gas. If low load resistance is selected then the circuit has less sensitivity and if high load resistance is selected then the circuit has high sensitivity.

**Working**

The ability of a Gas sensor to detect gases depends on the chemiresister to conduct current. The most commonly used chemiresistor is Tin Dioxide (SnO2) which is an n-type semiconductor that has free electrons (also called as donor). Normally the atmosphere will contain more oxygen than combustible gases. The oxygen particles attract the free electrons present in SnO2 which pushes them to the surface of the SnO2. As there are no free electrons available output current will be zero. The below gif shown the oxygen molecules (blue color) attracting the free electrons (black color) inside the SnO2 and preventing it from having free electrons to conduct current.



When the sensor is placed in the toxic or combustible gases environment, this reducing gas (orange color) reacts with the adsorbed oxygen particles and breaks the chemical bond between oxygen and free electrons thus releasing the free electrons. As the free electrons are back to its initial position they can now conduct current, this conduction will be proportional the amount of free electrons available in SnO2, if the gas is highly toxic more free electrons will be available.

**Types of Gas sensors**

Gas sensors are typically classified into various types based on the type of the sensing element it is built with. Below is the classification of the various types of gas sensors based on the sensing element that are generally used in various applications:

* Metal Oxide based gas Sensor.
* Optical gas Sensor.
* Electrochemical gas Sensor.
* Capacitance-based gas Sensor.
* Calorimetric gas Sensor.
* Acoustic based gas Sensor.

**List of Gas Sensors**

| Sensor Name | Gas to measure |
| --- | --- |
| [MQ-2](https://components101.com/mq2-gas-sensor) | Methane, Butane, LPG, Smoke |
| MQ-3 | Alcohol, Ethanol, Smoke |
| [MQ-4](https://components101.com/sensors/mq-4-methane-gas-sensor-pinout-datasheet) | Methane, CNG Gas |
| MQ-5 | Natural gas, LPG |
| [MQ-6](https://components101.com/sensors/mq-6-gas-sensor-pinout-equivalent-datasheet) | LPG, butane |
| MQ-7 | Carbon Monoxide |
| MQ-8 | Hydrogen Gas |
| MQ-9 | Carbon Monoxide, flammable gasses |
| MQ131 | Ozone |
| [MQ135](https://components101.com/sensors/mq135-gas-sensor-for-air-quality) | Air Quality |
| MQ136 | Hydrogen Sulphide gas |
| [MQ137](https://components101.com/sensors/mq137-gas-sensor) | Ammonia |
| MQ138 | Benzene, Toluene, Alcohol, Propane, Formaldehyde gas, Hydrogen |
| MQ214 | Methane, Natural Gas |
| MQ216 | Natural gas, Coal Gas |
| MQ303A | Alcohol, Ethanol, smoke |
| MQ306A | LPG, butane |
| MQ307A | Carbon Monoxide |
| MQ309A | Carbon Monoxide, flammable gas |

**Features**

* High Sensitivity
* Detection Range: 100 - 10,000 ppm iso-butane propane
* Fast Response Time: <10s
* Heater Voltage: 5.0V
* Dimensions: 18mm Diameter, 17mm High excluding pins, Pins - 6mm High

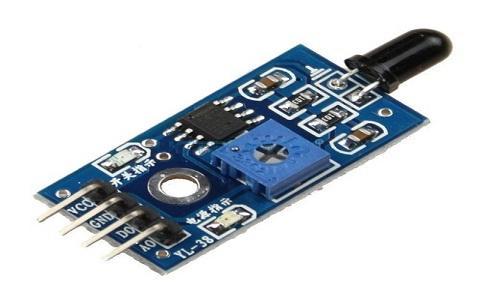
**Applications**

* Used in industries to monitor the concentration of the toxic gases.
* Used in households to detect an emergency incidents.
* Used at oil rig locations to monitor the concentration of the gases those are released.
* Used at hotels to avoid customers from smoking.
* Used in air quality check at offices.
* Used in air conditioners to monitor the CO2 levels.
* Used in detecting fire.
* Used to check concentration of gases in mines.
* Breath analyzer.

**Flame Sensor**

### A sensor which is most sensitive to a normal light is known as a flame sensor. That’s why this sensor module is used in flame alarms. This sensor detects flame otherwise wavelength within the range of 760 nm – 1100 nm from the light source. This sensor can be easily damaged to high temperature. So this sensor can be placed at a certain distance from the flame. The flame detection can be done from a 100cm distance and the detection angle will be 600. The output of this sensor is an analog signal or digital signal. These sensors are used in fire fighting robots like as a flame alarm

## What is a Flame Sensor?

A flame-sensor is one [kind of detector](https://www.elprocus.com/emf-detector-circuit-working-types-and-its-applications/) which is mainly designed for detecting as well as responding to the occurrence of a fire or flame. The flame detection response can depend on its fitting. It includes an [alarm system](https://www.elprocus.com/fire-alarm-circuit-using-thermistor/), a natural gas line, propane & a fire suppression system. This sensor is used in [industrial boilers](https://www.elprocus.com/what-is-steam-boiler-working-principle-types-of-steam-boilers/). The main function of this is to give authentication whether the boiler is properly working or not. The response of these sensors is faster as well as more accurate compare with a heat/smoke**** detector because of its mechanism while detecting the flame.

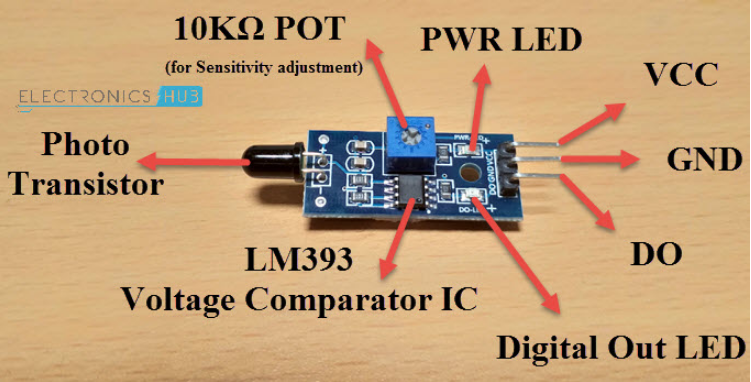
### A Brief Note on Flame Sensor

* A Flame Sensor is a device that can be used to detect presence of a fire source or any other bright light sources. There are several ways to implement a Flame Sensor but the module used in this project is an Infrared Radiation Sensitive Sensor.
* This particular flame sensor is based on YG1006 NPN Photo Transistor. The black object at the front of the module is this Photo Transistor.
* The YG1006 Photo Transistor looks like a black LED but it is a three terminal NPN Transistor, where the long lead is the Emitter and the shorter one is the collector (there is no base terminal as the light it detects will enable the flow of current).
* This photo transistor is coated with black epoxy, making it sensitive to Infrared radiations and this particular Photo Transistor (YG1006) is sensitive to Infrared Radiation in the wavelength range of 760nm to 1100nm.
* Using this particular type of Flame Sensor, you can detect Infrared Light up to a distance of 100cm within its 60 degrees of detection angle.
* There are two types of implementations of Flame Sensors using YG1006 Photo Transistor: one is with both Analog Output and Digital Output while the other is with only the Digital Output.
* Both these implementations require same components but the difference is that one module (the one with the Analog Output) provides the Sensor output as Analog Output.
* The Flame Sensor that I am using in this project has only Digital Output.

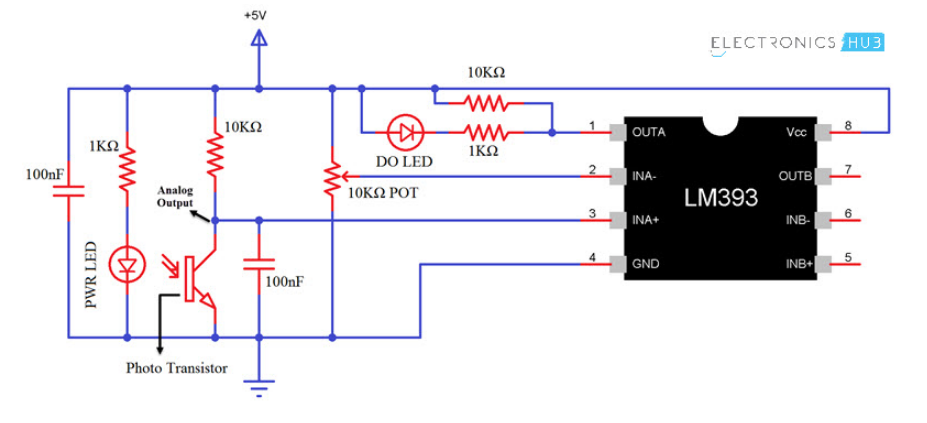
### Working Principle

This sensor/detector can be built with an [electronic circuit](https://www.elprocus.com/top-10-simple-electronic-circuits-for-beginners/) using a receiver like electromagnetic radiation. This sensor uses the infrared flame flash method, which allows the sensor to work through a coating of oil, dust, water vapor, otherwise ice.

#### Components of Flame Sensor



#### Circuit Diagram of Flame Sensor



**Pin Description**

* Pin1 (VCC pin): Voltage supply rages from 3.3V to 5.3V
* Pin2 (GND): This is a ground pin
* Pin3 (AOUT): This is an analog output pin (MCU.IO)
* Pin4 (DOUT): This is a digital output pin (MCU.IO)

**Types of Flame-sensors**

* IR single frequency
* IR multi-spectrum
* UV flame detectors
* UV/ IR flame detectors

### Features & Specifications

* Photosensitivity is high
* Response time is fast
* Simple to use
* Sensitivity is adjustable
* Detection angle is 600,
* It is responsive to the flame range.
* Accuracy can be adjustable
* Operating voltage of this sensor is 3.3V to 5V
* Analog voltage o/ps and digital switch o/ps
* The PCB size is 3cm X 1.6cm
* Power indicator & digital switch o/p indicator
* If the flame intensity is lighter within 0.8m then the flame test can be activated, if the flame intensity is high, then the detection of distance will be improved.

**Applications**

* Hydrogen stations
* Industrial heating
* Fire detection
* [Fire alarm](https://www.elprocus.com/fire-alarm-circuit-using-thermistor/)
* [Fire fighting robot](https://www.elprocus.com/projects-on-fire-fighting-robotic-vehicle/)
* Drying systems
* Industrial gas turbines
* Domestic heating systems
* Gas-powered cooking devices

**ACCELERATION SENSOR:**

An acceleration sensor, also known as an accelerometer, is a device that measures proper acceleration. Proper acceleration is the acceleration experienced by an object relative to free fall. This type of sensor is commonly used in various applications, including automotive systems, aerospace, consumer electronics (such as smartphones and wearable devices), industrial machinery, and more.

Accelerometers typically work based on the principles of inertia. They detect changes in the velocity of the object they are attached to, which can be caused by changes in motion, orientation, or vibration. These changes are then converted into electrical signals that can be interpreted by electronic systems.

Accelerometers can measure acceleration along one, two, or three axes, depending on their design. Single-axis accelerometers measure acceleration along a single direction (typically the vertical axis), while dual-axis and tri-axis accelerometers can measure acceleration along two or three orthogonal axes, respectively.

Some common applications of accelerometers include:

1. Motion sensing: Used in smartphones, tablets, and gaming controllers to detect tilting, shaking, or tapping gestures.

2. Automotive applications: Used for airbag deployment, stability control systems, and anti-lock braking systems (ABS).

3. Aerospace: Used for flight data recording, attitude control systems, and vibration monitoring in aircraft.

4. Industrial machinery: Used for monitoring vibrations in rotating equipment, detecting impacts in manufacturing processes, and monitoring structural integrity.

5. Health and fitness: Integrated into wearable devices to track physical activity, measure steps, and monitor sleep patterns.

Accelerometers play a crucial role in many modern technologies, enabling precise motion detection, orientation sensing, and vibration monitoring across a wide range of applications.

**LORA (LONG RANGE):**

LoRa WAN is a low power wide area network technology (LP-WAN) designed for Internet of Things (IoT) and smart sensor applications. As the name implies, long range transmission capability with less power consumption makes LoRa a significant player in IoT networks.

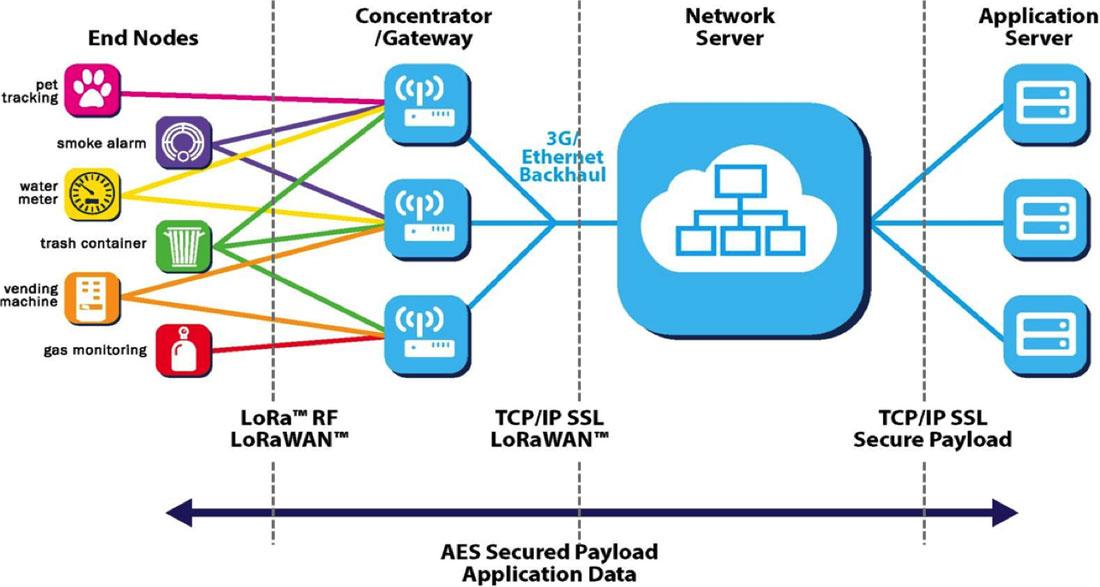




**WORKING**

In LoRa technology, a message transmitted by any device can be received by single or multiple gateways. The received messages will be forwarded to the central network for processing. Smart server architecture will handle these messages to each related applications.

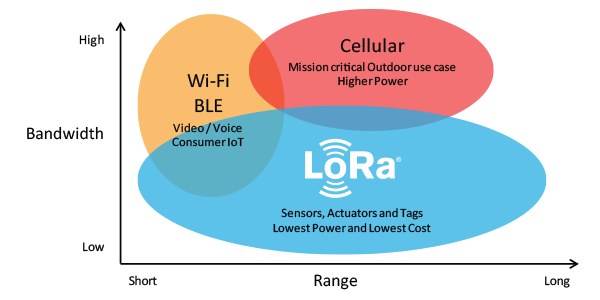
LoRa alliance is an open, non-profit organization of members that standardize, develop, monitors and improves LoRa standard. Internet of Things is one of the major drivers behind this highly efficient LP-WAN technology



**Why LoRa®?**

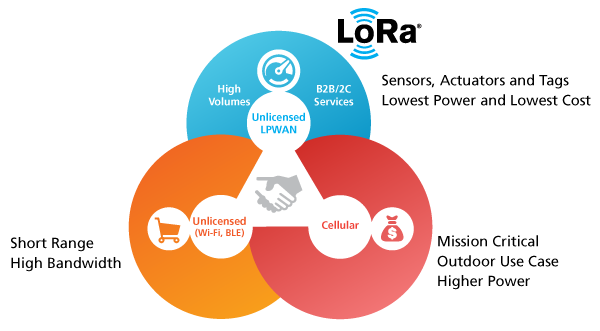
**LoRa Fills a Technology Gap**

LoRa Technology has revolutionized IoT by enabling data communication over a long range while using very little power. When connected to a non-cellular LoRaWAN network, LoRa devices accommodate a vast range of IoT applications by transmitting packets with important information. LoRaWAN fills the technology gap of Cellular and Wi-Fi/BLE based networks that require either high bandwidth or high power, or have a limited range or inability to penetrate deep indoor environments. In effect, LoRa Technology is flexible for rural or indoor use cases in smart cities, smart homes and buildings, smart agriculture, smart metering, and smart supply chain and logistics.



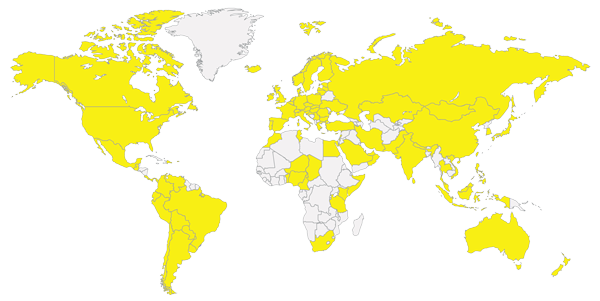
**LoRa Complements Wi-Fi, Bluetooth and Cellular**

Like Wi-Fi, LoRaWAN operates in the unlicensed band and supports indoor applications; like Cellular, LoRa Technology is highly secure from end devices to the application server, and is suitable for outdoor applications. LoRa devices and the LoRaWAN protocol combine these features of Wi-Fi and Cellular networks to offer an efficient, flexible and economical connectivity solution ideal for IoT applications whether indoor or outdoor and installed in public, private or hybrid networks. Simple sensor data can fuel analytics platforms, such as those for artificial intelligence and machine learning. These require data diversity which is made possible by low-cost LoRa-enabled sensors.



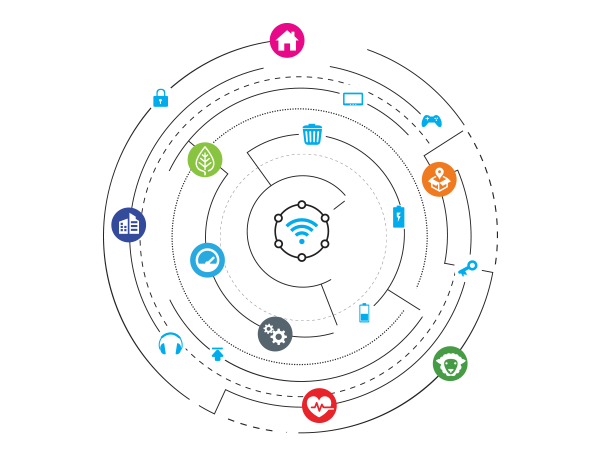
**Network Availability near You**

LoRa devices operate on the LoRaWAN open standard which is backed by the [LoRa Alliance®](https://www.lora-alliance.org/), an open, non-profit association committed to driving its adoption worldwide. The LoRa Alliance’s ecosystem of over 500 members includes over 100 public network operators with service in 58 countries and growing. Public, private and hybrid LoRaWAN networks are globally available and accessible in over 100 countries. This existing infrastructure makes it easy and efficient to deploy solutions immediately.



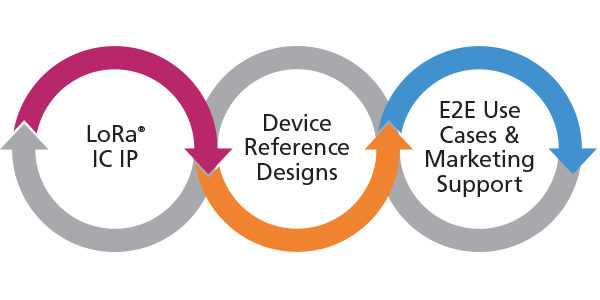
**A Rapidly Scaling Ecosystem**

The ecosystem supporting LoRa and LoRaWAN includes a comprehensive collection of network operators, hardware manufacturers, software designers, service providers, universities, and industry associations that play a key role in creating and enabling devices, networks and applications. The ecosystem is continuously growing ― and continues to grow at an impressive rate in conjunction with widespread adoption of LoRa Technology.



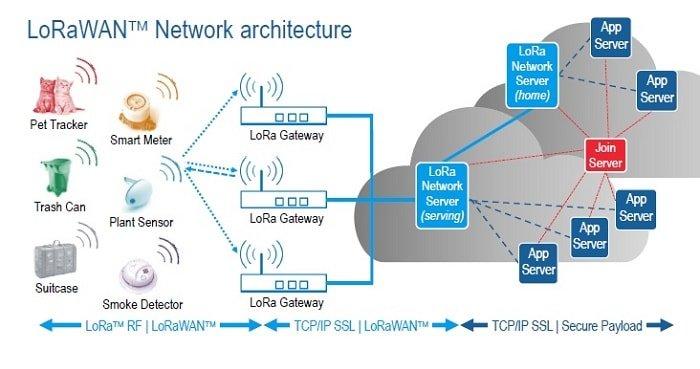
**Simplified Go-To-Market**

Semtech’s LoRa Technology enables innovators to overcome the age-old challenges that come with launching an IoT solution. The ecosystem supporting LoRa Technology is all-inclusive, providing end-to-end integration from silicon to services. As an open platform that operates on the unlicensed band, LoRa Technology is flexible for various business models to create profitability. The standardized LoRaWAN protocol is interoperable, enabling solutions to scale, bundle and evolve. Semtech and its ecosystem partners are committed to simplifying and accelerating the path for IoT innovators to bring products to market and demonstrate real-world use cases. A great place for innovators to start is in the LoRa Developer Portal – the online forum for IoT and M2M technologists to connect, learn and share.



**What is a LoRa Gateway?**

LoRa gateway is a sophisticated software architecture designed to carry out communication in LoRa network terminology. It acts as a smart hub between end devices and application network server. In future, it will be preinstalled on all IoT enabled hardware to support wide range of applications.



**LoRa WAN Specification**

Standard:            LoRa WAN technology follows IEEE 802.15.4 standard

Frequency:         License free ISM bands 433, 868, 915 MHz

Bandwidth:         125 KHz, 250 KHz and 500 KHz

Modulation:       Chirp spread spectrum based modulation (suitable for better sensitivity)

Data rate:            Up to 50 kbps

Range:                  Up to 20 KM

**Advantages of LoRa WAN (challenges of LP-WAN)**

* Long battery life for devices and sensors due to low power consumption
* Low cost implementation due to low cost hardware and unlicensed spectrum
* Long range coverage and in-building penetration
* Less complexity in programming
* Offers a secure transmission network
* Scalable network to support future upgrades
* Ease of access and connectivity to the cloud applications
* Remote management and control access
* Highly intelligent architecture

**Application and future of LoRa WAN technology**

**Smart City:** LoRa WAN will be inevitable technology in future smart city applications together with Internet of Things like:

* Smart lighting
* Air quality and pollution monitoring
* Smart parking and vehicle management
* Facilities and infrastructure management
* Fire detection and management
* Waste management

**Industrial Applications:** LoRa WAN is suitable for wide range of industrial applications.

* Radiation and leak detection
* Smart sensor technology
* Item location and tracking
* Shipping and transportation

**Smart home applications:** In future, billions of smart devices and home appliances will be connected to internet.

* Enhanced home security
* Home automation for IoT enables smart appliances

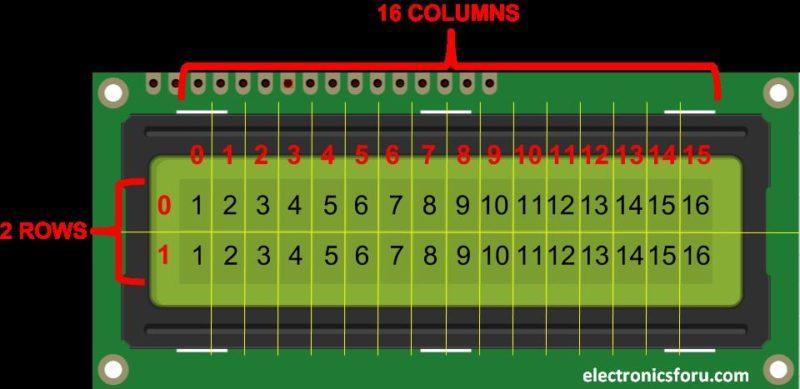
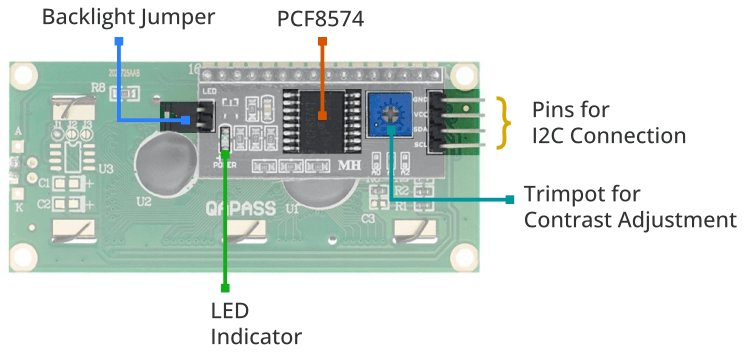
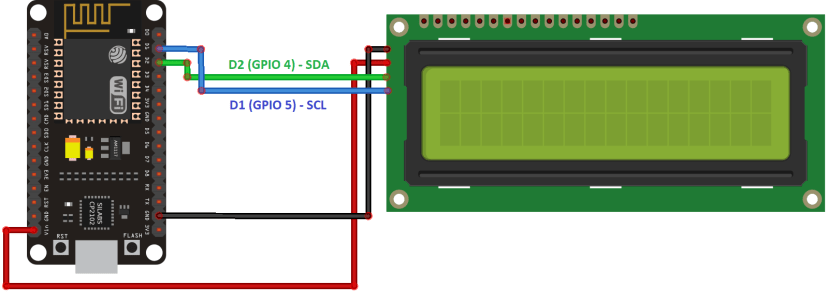
**Healthcare:** LoRa is one of the best solutions for connecting healthcare devices efficiently

* Health monitoring devices and management
* Wearable technology

**Agriculture:**LoRa technology can be used in smart agriculture and farming applications.

* Smart farming and livestock management
* Temperature and moisture monitoring
* Water level sensors and irrigation control

**I2C LCD:**

* LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16×2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even custom characters (unlike in seven segments), animations and so on.
* A 16×2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5×7 pixel matrix. This LCD has two registers, namely, Command and Data.
* The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD. Click to learn more about internal structure of a LCD.
* 
* At the center of this adapter, there is an 8-bit I/O expander chip – PCF8574. It takes the I2C data from the MCU (Arduino) and converts it into serial data required for an LCD display. At one side the I2C LCD adapter has four pins that can be connected to Arduino or any microcontroller that supports the I2C communication protocol. On another side, it has 16 pins that are connected to the LCD display. There are two header pins to control the backlight of the LCD display. One pin supplies a 5v power and another pin is for the backlight LED. These two pins are connected together by default. So the backlight will be always on. You can remove the jumper to turn off the backlight LED or you can use a potentiometer in between these two pins to control the intensity of the backlight LED.
* 
* **Working:**
* The I2C LCD operates by connecting to a microcontroller through two wires: SDA (Serial Data) and SCL (Serial Clock). It communicates using the I2C protocol, receiving commands and data from the microcontroller. An embedded controller within the LCD module interprets and processes this data to control the display, including showing characters, numbers, and symbols while efficiently using minimal hardware pins. This streamlined communication method simplifies wiring and hardware requirements, making it popular in various embedded systems.
* 
* **I2C LCD pinout:**

| **Pin Name** | **Description** |
| --- | --- |
| VCC | Power supply (usually +5V) |
| GND | Ground |
| SDA | I2C Serial Data Line |
| SCA | I2C Serial Clock Line |

* **Features of 16×2 LCD module**
* Operating Voltage is 4.7V to 5.3V
* Current consumption is 1mA without backlight
* Alphanumeric LCD display module, meaning can display alphabets and numbers
* Consists of two rows and each row can print 16 characters.
* Each character is build by a 5×8 pixel box
* Can work on both 8-bit and 4-bit mode
* It can also display any custom generated characters
* Available in Green and Blue Backlight
* **Advantages**
* LCD’s consumes less amount of power compared to CRT and LED
* LCD’s are consist of some microwatts for display in comparison to some mill watts for LED’s
* LCDs are of low cost
* Provides excellent contrast
* LCD’s are thinner and lighter when compared to cathode ray tube and LED
* **Disadvantages**
* Require additional light sources
* Range of temperature is limited for operation
* Low reliability
* Speed is very low
* LCD’s need an AC drive
* **Applications**
* Liquid crystal technology has major applications in the field of science and engineering as well on electronic devices.
* Liquid crystal thermometer
* Optical imaging
* The liquid crystal display technique is also applicable in visualization of the radio frequency waves in the waveguide
* Used in the medical applications

**BUTTON:**

****

Push Buttons are normally-open tactile switches. Push buttons allow us to power the circuit or make any particular connection only when we press the button. Simply, it makes the circuit connected when pressed and breaks when released. A push button is also used for triggering of the SCR by gate terminal. These are the most common buttons which we see in our daily life electronic equipment’s.

**Push Button Features**

* Prevent flux rise by the insert-molded terminal
* Snap-in mount terminal
* Contact Bounce: MAX 5mS
* Crisp clicking by tactile feedback
* Dielectric Withstanding Voltage 250V AC for 1 minute

**Technical Specifications**

* Mode of Operation: Tactile feedback
* Power Rating: MAX 50mA 24V DC
* Insulation Resistance: 100Mohm at 100v
* Operating Force: 2.55±0.69 N
* Contact Resistance: MAX 100mOhm
* Operating Temperature Range: -20 to +70 ℃
* Storage Temperature Range: -20 to +70 ℃

### **Applications**

* Calculators
* Push-button telephones
* Kitchen appliances
* Magnetic locks
* Various other mechanical and electronic devices, home and commercials.

**BUZZER:**

 A buzzer or beeper is an [audio](https://en.wikipedia.org/wiki/Sound) signalling device, which may be [mechanical](https://en.wikipedia.org/wiki/Machine), [electromechanical](https://en.wikipedia.org/wiki/Electromechanics), or [piezoelectric](https://en.wikipedia.org/wiki/Piezoelectricity) (piezo for short). Typical uses of buzzers and beepers include [alarm devices](https://en.wikipedia.org/wiki/Alarm_devices), [timers](https://en.wikipedia.org/wiki/Timer), and confirmation of user input such as a mouse click or keystroke



**How to use a Buzzer**

A buzzeris a small yet efficient component to add sound features to our project/system. It is very small and compact 2-pin structure hence can be easily used on breadboard, Perf Board and even on PCBs which makes this a widely used component in most electronic applications.

There are two types are buzzers that are commonly available. The one shown here is a simple buzzer which when powered will make a Continuous Beeeeeeppp.... sound, the other type is called a readymade buzzer which will look bulkier than this and will produce a Beep. Beep. Beep. Sound due to the internal oscillating circuit present inside it. But, the one shown here is most widely used because it can be customised with help of other circuits to fit easily in our application.

This buzzer can be used by simply powering it using a DC power supply ranging from 4V to 9V. A simple 9V battery can also be used, but it is recommended to use a regulated +5V or +6V DC supply. The buzzer is normally associated with a switching circuit to turn ON or turn OFF the buzzer at required time and require interval.

## Types

**Electromechanical**

**Early devices were based on an electromechanical system identical to an** [**electric bell**](https://en.wikipedia.org/wiki/Electric_bell) **without the metal gong. Similarly, a** [**relay**](https://en.wikipedia.org/wiki/Relay) **may be connected to interrupt its own** actuating [current](https://en.wikipedia.org/wiki/Electric_current), causing the [contacts](https://en.wikipedia.org/wiki/Switch) to buzz. Often these units were anchored to a wall or ceiling to use it as a sounding board. The word "buzzer" comes from the rasping noise that electromechanical buzzers made.

**Mechanical**

A [joy buzzer](https://en.wikipedia.org/wiki/Joy_buzzer) is an example of a purely mechanical buzzer and they require drivers. Other examples of them are doorbells.

**Piezoelectric**

**Piezoelectric disk beeper**

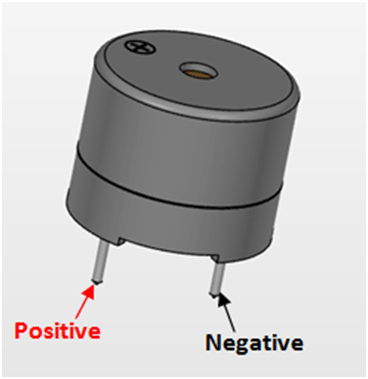
A [piezoelectric](https://en.wikipedia.org/wiki/Piezoelectric) element may be driven by an [oscillating](https://en.wikipedia.org/wiki/Oscillation) electronic circuit or other [audio signal](https://en.wikipedia.org/wiki/Audio_signal) source, driven with a [piezoelectric audio amplifier](https://en.wikipedia.org/wiki/Piezoelectric_audio_amplifier). Sounds commonly used to indicate that a button has been pressed are a click, a ring or a beep.



Interior of a readymade loudspeaker, showing a piezoelectric-disk-beeper (With 3 electrodes ... including 1 feedback-electrode (the central, small electrode joined with red wire in this photo), and an oscillator to self-drive the buzzer.

A piezoelectric buzzer/beeper also depends on acoustic cavity resonance or [Helmholtz resonance](https://en.wikipedia.org/wiki/Helmholtz_resonance) to produce an audible beep

### Buzzer Pin Configuration



| **Pin Number** | **Pin Name** | **Description** |
| --- | --- | --- |
| 1 | Positive | Identified by (+) symbol or longer terminal lead. Can be powered by 6V DC |
| 2 | Negative | Identified by short terminal lead. Typically connected to the ground of the circuit |

### Specificationshttps://upload.wikimedia.org/wikipedia/commons/thumb/d/d3/2007-07-24_Piezoelectric_buzzer.jpg/220px-2007-07-24_Piezoelectric_buzzer.jpg

* Rated Voltage: 6V DC
* Operating Voltage: 4-8V DC
* Rated current: <30mA
* Sound Type: Continuous Beep
* Resonant Frequency: ~2300 Hz
* Small and neat sealed package
* Breadboard and Perf board friendly

**Application**

* Alarming Circuits, where the user has to be alarmed about something
* Communication equipments
* Automobile electronics
* Portable equipments, due to its compact size.

**SOFTWARE MODULE:**

**Arduino Software (IDE):**

* [Writing Sketches](https://www.arduino.cc/en/Guide/Environment#toc1)
  + [File](https://www.arduino.cc/en/Guide/Environment#toc2)
  + [Edit](https://www.arduino.cc/en/Guide/Environment#toc3)
  + [Sketch](https://www.arduino.cc/en/Guide/Environment#toc4)
  + [Tools](https://www.arduino.cc/en/Guide/Environment#toc5)
  + [Help](https://www.arduino.cc/en/Guide/Environment#toc6)
* [Sketchbook](https://www.arduino.cc/en/Guide/Environment#toc7)
* [Tabs, Multiple Files, and Compilation](https://www.arduino.cc/en/Guide/Environment#toc8)
* [Uploading](https://www.arduino.cc/en/Guide/Environment#toc9)
* [Libraries](https://www.arduino.cc/en/Guide/Environment#toc10)
* [Third-Party Hardware](https://www.arduino.cc/en/Guide/Environment#toc11)
* [Serial Monitor](https://www.arduino.cc/en/Guide/Environment#toc12)
* [Preferences](https://www.arduino.cc/en/Guide/Environment#toc13)
* [Language Support](https://www.arduino.cc/en/Guide/Environment#toc14)
* [Boards](https://www.arduino.cc/en/Guide/Environment#toc15)



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 and Genuino hardware to upload programs and communicate with them.

Writing Sketches

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

NB: Versions of the Arduino Software (IDE) prior to 1.0 saved sketches with the extension .pde. It is possible to open these files with version 1.0, you will be prompted to save the sketch with the .ino extension on save.

Additional commands are found within the five menus: File, Edit, Sketch, Tools, Help. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

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.

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/Decrease Indent

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.

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.

• Upload Using Programmer

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 o the toolbar.

Tools

• Auto Format

This formats your code nicely: i.e. indents it so that opening and closing curly braces line up, and that the statements inside curly braces are indented more.

• 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 harware 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.

• Burn Bootloader

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.

Help

Here you find easy access to a number of documents that come with the Arduino Software (IDE). You have access to Getting Started, Reference, this guide to the IDE and other documents locally, without an internet connection. The documents are a local copy of the online ones and may link back to our online website.

• Find in Reference

This is the only interactive function of the Help menu: it directly selects the relevant page in the local copy of the Reference for the function or command under the cursor.

Sketchbook

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.

Beginning with version 1.0, files are saved with a .ino file extension. Previous versions use the .pde extension. You may still open .pde named files in version 1.0 and later, the software will automatically rename the extension to .ino.

Tabs, Multiple Files, and Compilation

Allows you to manage sketches with more than one file (each of which appears in its own tab). These can be normal Arduino code files (no visible extension), C files (.c extension), C++ files (.cpp), or header files (.h).

Uploading

Before uploading your sketch, you need to select the correct items from the Tools > Board and Tools > Port menus. The boards are described below. On the Mac, the serial port is probably something like /dev/tty.usbmodem241 (for an Uno or Mega2560 or Leonardo) or /dev/tty.usbserial-1B1 (for a Duemilanove or earlier USB board), or /dev/tty.USA19QW1b1P1.1 (for a serial board connected with a Keyspan USB-to-Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows Device Manager. On Linux, it should be /dev/ttyACMx , /dev/ttyUSBx or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the Upload item from the Sketch menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

When you upload a sketch, you're using the Arduino bootloader, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The bootloader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The bootloader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

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.

There is a list of libraries in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you do can import a library from a zip file and use it in an open sketch. See these instructions for installing a third-party library.

To write your own library, see this tutorial.

Third-Party Hardware

Support for third-party hardware can be added to the hardware directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, bootloaders, and programmer definitions. To install, create the hardware directory, then unzip the third-party platform into its own sub-directory. (Don't use "arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

For details on creating packages for third-party hardware, see the Arduino IDE 1.5 3rd party Hardware specification.

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.

You can also talk to the board from Processing, Flash, MaxMSP, etc (see the interfacing page for details).

Preferences

Some preferences can be set in the preferences dialog (found under the Arduino menu on the Mac, or File on Windows and Linux). The rest can be found in the preferences file, whose location is shown in the preference dialog.

Language Support

Since version 1.0.1 , the Arduino Software (IDE) has been translated into 30+ different languages. By default, the IDE loads in the language selected by your operating system. (Note: on Windows and possibly Linux, this is determined by the locale setting which controls currency and date formats, not by the language the operating system is displayed in.)

If you would like to change the language manually, start the Arduino Software (IDE) and open the Preferences window. Next to the Editor Language there is a dropdown menu of currently supported languages. Select your preferred language from the menu, and restart the software to use the selected language. If your operating system language is not supported, the Arduino Software (IDE) will default to English.

You can return the software to its default setting of selecting its language based on your operating system by selecting System Default from the Editor Language drop-down. This setting will take effect when you restart the Arduino Software (IDE). Similarly, after changing your operating system's settings, you must restart the Arduino Software (IDE) to update it to the new default language.

Boards

The board selection has two effects: it sets the parameters (e.g. CPU speed and baud rate) used when compiling and uploading sketches; and sets and the file and fuse settings used by the burn bootloader command. Some of the board definitions differ only in the latter, so even if you've been uploading successfully with a particular selection you'll want to check it before burning the bootloader. You can find a comparison table between the various boards here.

Arduino Software (IDE) includes the built in support for the boards in the following list, all based on the AVR Core. The Boards Manager included in the standard installation allows to add support for the growing number of new boards based on different cores like Arduino Due, Arduino Zero, Edison, Galileo and so on.

• Arduino Yùn

An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.

• Arduino/Genuino Uno

An ATmega328 running at 16 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.

• Arduino Diecimila or Duemilanove w/ ATmega168

An ATmega168 running at 16 MHz with auto-reset.

• Arduino Nano w/ ATmega328

An ATmega328 running at 16 MHz with auto-reset. Has eight analog inputs.

• Arduino/Genuino Mega 2560

An ATmega2560 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.

• Arduino Mega

An ATmega1280 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.

• Arduino Mega ADK

An ATmega2560 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.

• Arduino Leonardo

An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.

• Arduino/Genuino Micro

An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.

• Arduino Esplora

An ATmega32u4 running at 16 MHz with auto-reset.

• Arduino Mini w/ ATmega328

An ATmega328 running at 16 MHz with auto-reset, 8 Analog In, 14 Digital I/O and 6 PWM.

• Arduino Ethernet

Equivalent to Arduino UNO with an Ethernet shield: An ATmega328 running at 16 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.

• Arduino Fio

An ATmega328 running at 8 MHz with auto-reset. Equivalent to Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328, 6 Analog In, 14 Digital I/O and 6 PWM.

• Arduino BT w/ ATmega328

ATmega328 running at 16 MHz. The bootloader burned (4 KB) includes codes to initialize the on-board bluetooth module, 6 Analog In, 14 Digital I/O and 6 PWM..

• LilyPad Arduino USB

An ATmega32u4 running at 8 MHz with auto-reset, 4 Analog In, 9 Digital I/O and 4 PWM.

• LilyPad Arduino

An ATmega168 or ATmega132 running at 8 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.

• Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328

An ATmega328 running at 16 MHz with auto-reset. Equivalent to Arduino Duemilanove or Nano w/ ATmega328; 6 Analog In, 14 Digital I/O and 6 PWM.

• Arduino NG or older w/ ATmega168

An ATmega168 running at 16 MHz without auto-reset. Compilation and upload is equivalent to Arduino Diecimila or Duemilanove w/ ATmega168, but the bootloader burned has a slower timeout (and blinks the pin 13 LED three times on reset); 6 Analog In, 14 Digital I/O and 6 PWM.

• Arduino Robot Control

An ATmega328 running at 16 MHz with auto-reset.

• Arduino Robot Motor

An ATmega328 running at 16 MHz with auto-reset.

• Arduino Gemma

An ATtiny85 running at 8 MHz with auto-reset, 1 Analog In, 3 Digital I/O and 2 PWM.

**FUTURE SCOPE**

* Integration with existing ADAS systems for a comprehensive safety approach
* Development of a mobile application for remote vehicle monitoring and alerts
* Exploration of incorporating additional sensors for more comprehensive hazard detection
* Investigation of mesh networking capabilities for broader communication range

**CONCLUSION:**

In conclusion, the implementation of a vehicle communication system utilizing LoRa technology offers a significant advancement in enhancing road safety and awareness. By integrating comprehensive sensor arrays, real-time monitoring, and immediate alerting capabilities, the proposed system provides a robust solution for detecting and mitigating potential hazards on the road. The combination of extended range transmission, low-power consumption, and cost-effectiveness ensures scalability and accessibility across various vehicle types and applications. With its ability to facilitate timely communication between vehicles and enable prompt responses to critical situations, this project marks a significant step towards realizing a smarter and safer automotive ecosystem.

**REFERENCE**

* S. Iranmanesh, F. S. Abkenar, A. Jamalipour, and R. Raad, “A heuristic distributed scheme to detect falsification of mobility patterns in internet Of vehicles,” IEEE Internet Things J., vol. 9, no. 1, pp. 719–727, Jan. 2022.
* O. E. Marai and T. Taleb, “Smooth and low latency video streaming for autonomous cars during handover,” IEEE Netw., vol. 34, no. 6, pp. 302–309,Nov./Dec. 2020.
* T. Taleb, A. Ksentini, and F. Filali, “Wireless connection steering for vehicles,” in Proc. IEEE Glob. Commun. Conf., 2012, pp. 56–60.
* T. Taleb, A. Benslimane, and K. B. Letaief, “Toward an effective risk conscious and collaborative vehicular collision avoidance system,” IEEE Trans. Veh. Technol., vol. 59, no. 3, pp. 1474–1486, Mar. 2010.