ACKNOWLEDGEMENT

We sincerely acknowledge the help received from various persons and sources in collecting data and information completing this project satisfactorily.

We thank our parents for their continuous encouragement, motivation and blessings without which any of us could not have reached this achievement.

We express our profound thanks to our principal Shri V. N. DESAI for providing us ample opportunities to prepare this project.

We take the immense pleasure to thank our Head of Department (HOD) Mr. M. V. Brahma Prakash sir for permitting us to undertake this project.

We take a great opportunity to express our deep scene of gratitude and affectionate respect to guide Mr. M. V. Brahma Prakash, not only for his full support and guidance, but also for his valuable suggestions and help everytime, which led us to complete this project in a successful way.

Last but not the least we would like to thank all the faculties of our institute, parents, friends and well-wishers who have helped us directly or indirectly in bringing out this project work successfully.

INDEX

ABSTRACT

Feeling sleepy while driving could cause hazardous traffic accidents. However, when driving alone on highway or driving over a long period of time, drivers are inclined to feel bored and sleepy, or even fall asleep. Nowadays most of the products of driver anti-sleep detection sold in the market are simply earphone making intermittent noises, which are quite annoying and inefficient. As such, there is a high demand for cheap and efficient driver sleep detection. Therefore, we camp up with an idea and successfully developed a sleepy detection and alarming system, which could effectively meet this demand.

In the present day's alcohol-attributable accidents increasing rapidly where the concern of alcohol is a factor in many categories of injury. Every year it is reported about 2.3 million premature deaths due to harmful consumption of alcohol. In this paper we have proposed an improved alcohol detection system for the use in an automobile ignition locking system using Arduino. A temperature senser is used to measure the temperature of the breath sample to ensure that it is the same temperature as human breath. A senser is used for a specific volume of the breath sample, which is used to determine the alcohol content. A Micro Controller is used to convert the output into a reading which represents the breath alcohol content of the breath sample. This analysis is used as part of an overall automobile ignition locking system which prohibits starting the car when the operator is intoxicating. The system also requires rolling retests to ensure that the driver is still sober.

1. INTRODUCTION

Anti-Sleep system is constructed using Arduino and some very basic components. This system Alerts the driver whenever he is s;eepy while driving the vehicle. Since sleeping on wheels is dangerous sometimes it may convert into fettle accidents which can lead to death.

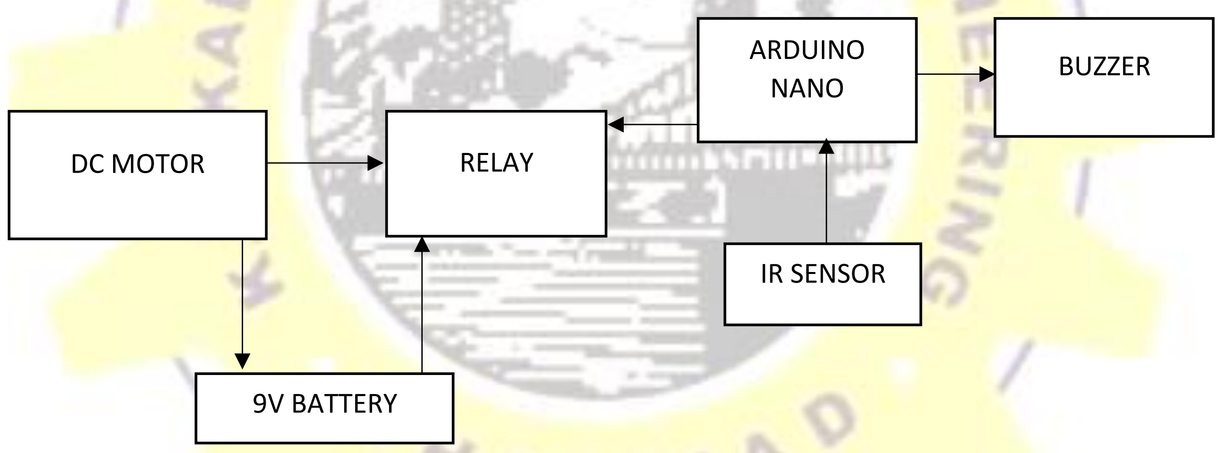
So, to prevent such consequences of accident we can use this gadget alert the driver when he feels drowsiness. So, let’s see how the anti-sleep system works.

Now a days a lot of accidents occur due to the Drink and Drive & it needs to stop people to drive after drinking the alcohol. The alcohol car safety system provides a system that prevents such accidents in intoxicated position.

This system uses the sensers and use it to detect whether the driver is intoxicated or not, if it detects such thing then it automatically sets the warning alert to driver that he is not in the position to drive the car and this system automatically turns off the ignition of car.

2. BLOCK DIAGRAM:

2.1 ANTI SLEEP DETECTION DEVICE:



2.1.1 DC MOTOR

A DC Motor is any of a class of rotary electrical motors that converts direct current (DC) electrical energy into mechanical energy. The most common types rely on the forces produced by magnetic fields. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the direction of current in part of the motor.

DC motors were the first form of motor widely used, as they could be powered from existing direct-current lighting power distribution systems. A DC motor's speed can be controlled over a wide range, using either a variable supply voltage or by changing the strength of current in its field windings. Small DC motors are used in tools, toys, and appliances. The universal motor can operate on direct current but is a lightweight brushed motor used for portable power tools and appliances. Larger DC motors are currently used in propulsion of electric vehicles, elevator and hoists, and in drives for steel rolling mills. The advent of power electronics has made replacement of DC motors with AC motors possible in many applications.

Some Types of DC Motors Available at market:

1. Permanent Magnet DC Motors

The permanent magnet motor uses a permanent magnet to create field flux. This type of DC motor provides great starting torque and has good speed regulation, but torque is limited so they are typically found on low horsepower applications.

2. Series DC Motors

In a series DC motor, the field is wound with a few turns of a large wire carrying the full armature current. Typically, series DC motors create a large amount of starting torque, but cannot regulate speed and can even be damaged by running with no load. These limitations mean that they are not a good option for variable speed drive applications.

3. Shunt DC Motors

In shunt DC motors the field is connected in parallel (shunt) with the armature windings. These motors offer great speed regulation due to the fact that the shunt field can be excited separately from the armature windings, which also offers simplified reversing controls.

4. Compound DC Motors

Compound DC motors, like shunt DC motors, have a separately excited shunt field. Compound DC motors have good starting torque but may experience control problems in variable speed drive applications.

Between the 4 types of DC motors, the potential applications are numerous. Each type of DC motor has its strengths and weaknesses. Understanding these can help you understand which types may be good for your application.

2.1.2 <u>RELAY:</u>

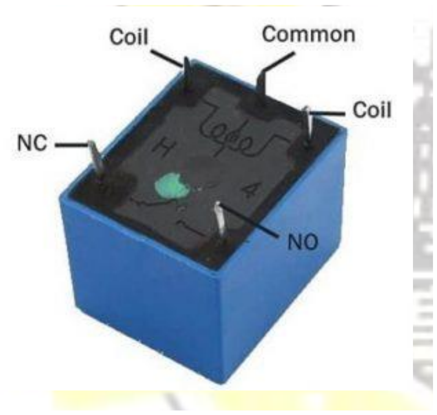
Relay is one kind of electro-mechanical component that functions as a switch. The relay coil is energized by DC so that contact switches can be opened or closed. A single channel 5V relay module generally includes a coil, and two contacts like normally open (NO) and normally closed (NC). This article discusses an overview of the 5V relay module & it's working but before going to discuss what is relay module is, first we have to know what is relay and its pin configuration.

What is a 5V Relay?

A 5v relay is an automatic switch that is commonly used in an automatic control circuit and to control a high-current using a low-current signal. The input voltage of the relay signal ranges from 0 to 5V.

5V Relay Pin Configuration

The pin configuration of the 5V relay is shown below. This relay includes 5-pins where each pin and its functionality are shown below.



Pin1 (End 1): It is used to activate the relay; usually this pin one end is connected to 5Volts whereas another end is connected to the ground.
Pin2 (End 2): This pin is used to activate the Relay.
Pin3 (Common (COM)): This pin is connected to the main terminal of the Load to make it active.

Pin4 (Normally Closed (NC)): This second terminal of the load is connected to either NC/ NO pins. If this pin is connected to the load, then it will be ON before the switch.
Pin5 (Normally Open (NO)): If the second terminal of the load is allied to the NO pin, then the load will be turned off before the switch.

Features

The **features of the 5V relay** include the following.

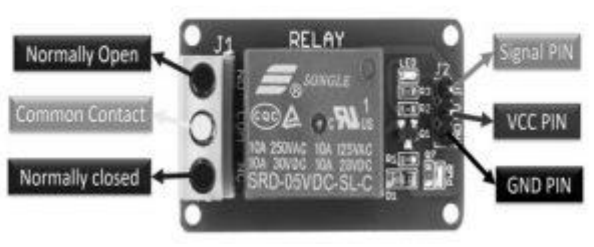
- Normal Voltage is 5V DC
- Normal Current is 70mA
- AC load current Max is 10A at 250VAC or 125V AC
- DC load current Max is 10A at 30V DC or 28V DC
- It includes 5-pins & designed with plastic material
- Operating time is 10msec
- Release time is 5msec
- Maximum switching is 300 operating per minute

5V Relay Module

The relay module with a single channel board is used to manage high voltage, current loads like solenoid valves, motor, AC load & lamps. This module is mainly designed to interface through different microcontrollers like PIC, Arduino, etc.

5V Relay Module Pin Configuration

The pin configuration of the 5V relay module is shown below. This module includes 6-pins where each pin and its functionality are discussed below.



Relay Module Pin Diagram

Normally Open (NO): This pin is normally open unless we provide a signal to the relay modules signal pin. So, the common contact pin smashes its link through the NC pin to make a connection through the NO pin

Common Contact: This pin is used to connect through the load that we desire to switch by using the module.

Normally Closed (NC): This NC pin is connected through the COM pin to form a closed circuit. However, this NC connection will break once the relay is switched through providing an active high/low signal toward the signal pin from a microcontroller.

Signal Pin: The signal pin is mainly used for controlling the relay. This pin works in two cases like active low otherwise active high. So, in active low case, the relay activates once we provide an active low signal toward the signal pin, whereas, in an active high case, the relay will trigger once we provide a high signal toward the signal pin.

However, these modules generally work on an active high signal which will strengthen the relay coil to make contact with the common terminal with the normally open terminal.

5V VCC: This pin needs 5V DC to work. So 5V DC power supply is provided to this pin.

Ground: This pin connects the GND terminal of the power supply.

2.1.3 ARDUINO NANO:

The **Arduino Nano** is a small, complete, and breadboard-friendly board based on the ATmega328P released in 2008. It offers the same connectivity and specs of the Arduino Uno board in a smaller form factor.

The Arduino Nano is equipped with 30 male I/O headers, in a DIP-30-like configuration, which can be programmed using the Arduino Software integrated development environment (IDE), which is common to all Arduino boards and running both online and offline. The board can be powered through a type-B mini-USB cable or from a 9 V battery.

In 2019, Arduino released the **Arduino Nano Every**, a pin-equivalent evolution of the Nano. It features a more powerful ATmega4809 processor and twice the RAM

Technical Specifications:

- Microcontroller: Microchip ATmega328P
- Operating voltage: 5 volts
- Input voltage: 6 to 20 volts
- Digital I/O pins: 14 (6 optional PWM outputs)
- Analog input pins: 8
- DC per I/O pin: 40 mA
- DC for 3.3 V pin: 50 mA
- Flash memory: 32 KB, of which 0.5 KB is used by bootloader
- SRAM: 2 KB
- EEPROM: 1 KB
- Clock speed: 16 MHz
- Length: 45 mm
- Width: 18 mm
- Mass: 7 g
- USB: Mini-USB Type-B [5]
- ICSP Header: Yes
- DC Power Jack: No

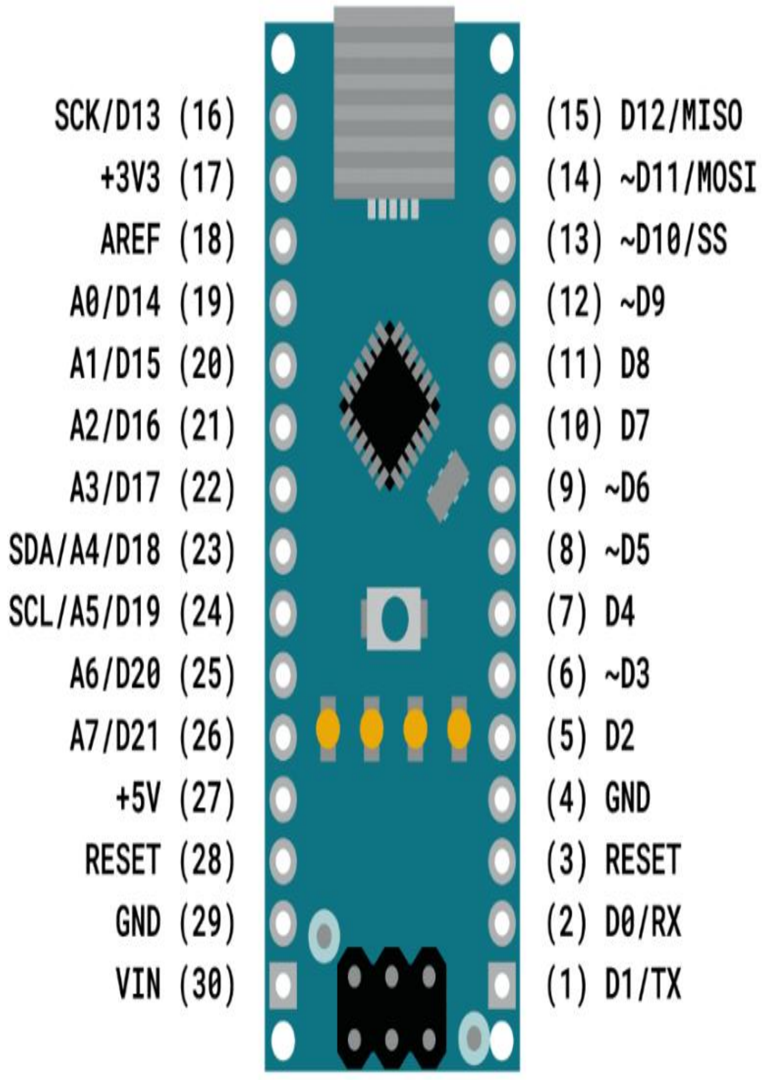
Automatic (software) reset

Rather than 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 nano-farad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip.

This setup has other implications. When the Nano is connected to 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.

The **Arduino Nano** is another popular Arduino development board very much similar to the Arduino UNO. They use the same Processor (Atmega328p) and hence they both can share the same program.

Arduino Nano Pinout Configuration



Pin Category	Pin Name	Details
Power	**Vin, 3.3V, 5V, GND**	**Vin:** Input voltage to Arduino when using an external power source (6-12V). **5V:** Regulated power supply used to power microcontroller and other components on the board. **3.3V:** 3.3V supply generated by on-board voltage regulator. Maximum current draw is 50mA. **GND:** Ground pins.
Reset	**Reset**	Resets the microcontroller.
Analog Pins	**A0 – A7**	Used to measure analog voltage in the range of 0-5V
Input/Output Pins	**Digital Pins D0 - D13**	Can be used as input or output pins. 0V (low) and 5V (high)
Serial	Rx, **Tx**	Used to receive and transmit TTL serial data.
External Interrupts	2, 3	To trigger an interrupt.
PWM	3, 5, 6, 9, 11	Provides 8-bit PWM output.
SPI	10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK)	Used for SPI communication.
Inbuilt LED	**13**	To turn on the inbuilt LED.
IIC	A4 (SDA), A5 (SCA)	Used for TWI communication.
AREF	**AREF**	To provide a reference voltage for input voltage.

Arduino Nano Technical Specifications

Microcontroller	ATmega328P – 8-bit AVR family microcontroller
Operating Voltage	5V
Recommended Input Voltage for Vin pin	7-12V
Analog Input Pins	6 (A0 – A5)
Digital I/O Pins	14 (Out of which 6 provide PWM output)
DC Current on I/O Pins	40 mA
DC Current on 3.3V Pin	50 mA
Flash Memory	32 KB (2 KB is used for Bootloader)
SRAM	2 KB
EEPROM	1 KB
Frequency (Clock Speed)	16 MHz
Communication	IIC, SPI, USART

Applications

- Prototyping of Electronics Products and Systems
- Multiple DIY Projects.
- Easy to use for beginner-level DIYers and makers.
- Projects requiring Multiple I/O interfaces and communications.

2.1.4 BUZZER:

This buzzer is an active buzzer, which basically means that it will buzz at a predefined frequency (2300 ±300 Hz) on its own even when you just apply steady DC power. If you are looking for a buzzer can produce varied tones from an oscillating input signal, then take a look at our passive buzzer.

Some people prefer to get active buzzers since they can use them with steady DC power but also be able to produce some variety of tones by applying an oscillating signal. Some consider them to be more versatile than their cousin, the passive buzzer, which is the type that requires an oscillating signal to create any tone.

It is possible, and often done, to still create different tones through an active buzzer when you apply an oscillating signal to the buzzer, but the spectrum of possible different tones is very limited and not as crisp or clean of sound as can be produced with a passive buzzer.

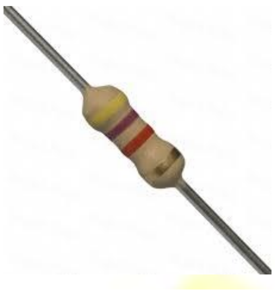
One advantage to an active buzzer is that you can still produce a sound from the buzzer connected to a microcontroller, such as an Arduino, by just driving a standard high output on the connected pin. The benefits of this are that you don't need to use processing power, hardware timers, or additional code to produce sound.

SPECIFICATIONS:

Longer pin is the positive pin

Rated Voltage	5 V
Operating Voltage	4~8 V
Max Rated Current	≤32 mA
Min. Sound Output at 10cm	85 dB
Resonant Frequency	2300 ±300 Hz
Operating Temperature	-20°C to 45°C
Dimensions (Excluding Pins)	
Height	9.16 mm (0.36")
Diameter	11.78 mm (0.46")
Weight	1.6 g (0.057 oz)

2.1.5 RESISTOR:



The 4.7k ohm (4700 ohm) resistor is one of the most common resistors in electronics. The four band 4.7k resistor is easy to recognize with its' distinctive pattern of yellow, violet, and red colour bands. The 5 and 6 band versions have a pattern of yellow, violet, black, brown, followed by the tolerance band and temperature coefficient band on the 6-band version.

One difficulty is that the 4.7k ohm resistor is easy to confuse with a 470-ohm resistor, so keep this in mind and verify using the colour code or a mustimeter if possible.

In this article, we'll learn how to identify 4 band, 5 band, *and* 6 band 4.7k Ohm resistors.
4.7k Ohm Resistor – 4, 5, 6 Band Colour Code Chart

	Band One	**Band Two**	**Band Three**	**Band Four**	**Band Five**	**Band Six**
Four Band	Yellow	Violet	Red	± %	–	–
Five Band	Yellow	Violet	Black	Brown	± %	–
Six Band	Yellow	Violet	Black	Brown	± %	R(T°)

4.7k Ohm Resistor Colour Bands

Each band on the 4.7k ohm resistor has a specific purpose.

All color-coded resistors (regardless of how many bands they have) have at least two-digit bands, one multiplier band, and one tolerance. This is the structure of a four-band resistor colour code; five and six band resistors add to this basic structure.

In a four-band resistor, the first three bands give us the resistor's **nominal value**.

The 4th band gives us the **tolerance** of the resistor. Even though the resistor is given a 4.7k ohm nominal value, the actual resistance will vary a bit above or below the nominal value. The tolerance tells us the range that we can expect; higher quality resistors will have an improved tolerance over lower quality ones.

In a five-band resistor, the first four bands tell us the nominal value of the resistor. Note that **a five-band resistor's extra band is used to add a digit to the nominal value**. Think of it as an extra digit band added to the front, with the other bands retaining the same function as those of a four-band resistor. The last band (i.e., the fifth band) still identifies the tolerance.

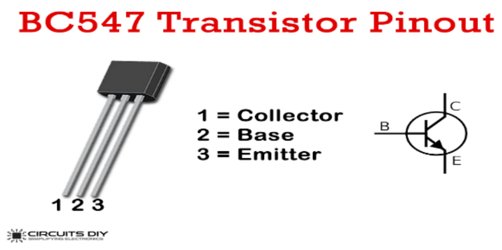
Six band resistors add another colour band at the end of the resistor that tells us the **temperature coefficient**. This is an indication of how sensitive the resistor is to temperature changes.

All resistors have a tolerance value, which means that the value is unlikely to be exactly 4.7k Ohms. Higher quality resistors have tighter tolerances.

2.1.6 <u>BC547 TRANSISTOR:</u>

When asking BC547 belongs to which type of the triode, basically, we think that BC547 is a normal NPN (Negative-Positive-Negative) junction transistor. And he bc547 parameter and the bc547 pin diagram are very important in the basic knowledge of BC547.

Below I will discuss the new electronic component called BC547. It is a BJT transistor and is often used to satisfy the need of quick switching. If you use this transistor in an engineering project, I want to suggest you to download BC547 Proteus Simulation. As we all know, simulating before entering the hardware is always a better strategy. So just let us to take a look at the basic knowledge of BC547.



Operational Status of BC547 Transistor

BC547 has two operation status: forward bias and reverse bias.

In the status of the forward bias, the current can pass when the collector and emitter are connected.

While in the status of the reverse bias, it acts as a disconnect switch and current cannot pass.

Pinout of BC547: Pin diagram

The BC547 transistor pinout shows 3 pins from left: Collector, base, and emitter respectively.

Pin Number	**Pin Name**	**Description**
1	Collector	This pin act as an inlet as the current enters the transistor from here. The collector is denoted by ‘C’.
2	Base	This pin controls the transistor biasing. The base is denoted by ‘B’.
3	Emitter	This pin act as an outlet and the current comes out of the transistor from here. The emitter is denoted by ‘E’.

Technical specifications of BC547:

- Package-Type: **TO-92**
- Transistor Type: **NPN**
- Max Collector Current (I_C): **100mA**
- Max Collector-Emitter Voltage (V_{CE}): **45V**
- Max Collector-Base Voltage (V_{CB}): **50V**
- Max Emitter-Base Voltage (VEBO): **6V**
- Max Collector Dissipation (Pc): **500 milliwatt**
- Max Transition Frequency (ft): **300 MHz**
- Minimum & Maximum DC Current Gain (h_{eft}): **110 – 800**
- Max Storage & Operating temperature Should Be: **-65 to +150 Centigrade**
- Low Noise: **2-10 dB**

BC547 transistor Uses and Applications:

- The highest transition frequency of BC547 is 300MHz. Thus, it can also be used in RF circuits.
- Amplification of current
- Audio Amplifiers
- Switching Loads < 100mA
- Transistor Darlington Pairs
- Amplifiers like Audio, signal, etc.
- Darlington pair
- Quick switching
- PWM (Pulse Width Modulation)

2.1.7 <u>IR SENSOR:</u>

IR technology is used in daily life and also in industries for different purposes. For example, TVs use an **IR sensor** to understand the signals which are transmitted from a remote control. The main benefits of IR sensors are low power usage, their simple design & their convenient features. IR signals are not noticeable by the human eye. The IR radiation in the **electromagnetic spectrum** can be found in the regions of the visible & microwave. Usually, the wavelengths of these waves range from 0.7 µm 5 to 1000µm. The IR spectrum can be divided into three regions like near-infrared, mid, and far-infrared. The near IR region's wavelength ranges from 0.75 – 3µm, the mid-infrared region's wavelength ranges from 3 to 6µm & the far IR region's infrared radiation's wavelength is higher than 6µm.

What is an IR Sensor/Infrared Sensor?

An infrared sensor is an electronic device, that emits in order to sense some aspects of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion. These types of sensors measure only infrared radiation, rather than emitting it that is called a passive IR sensor. Usually, in the infrared spectrum, all the objects radiate some form of thermal radiation.

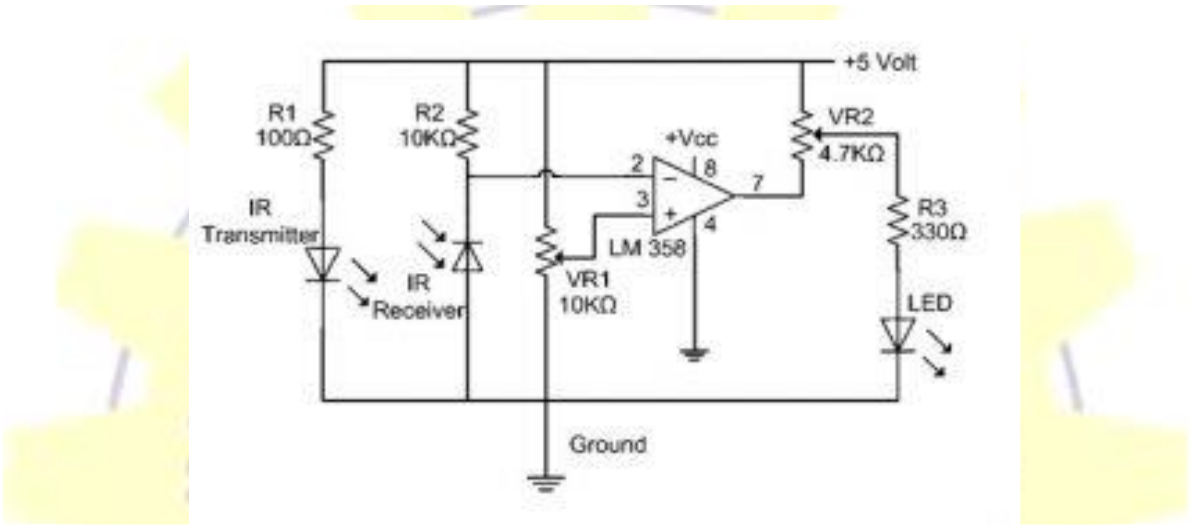
These types of radiations are invisible to our eyes, which can be detected by an infrared sensor. The emitter is simply an IR LED (Light Emitting Diode) and the detector is simply an IR photodiode that is sensitive to IR light of the same wavelength as that emitted by the IR LED. When IR light falls on the photodiode, the resistances and the output voltages will change in proportion to the magnitude of the IR light received.

IR Sensor Circuit Diagram

An infrared sensor circuit is one of the basic and popular sensor modules in an electronic device. This sensor is analogous to human's visionary senses, which can be used to detect obstacles and it is one of the common applications in real-time. This circuit comprises the following components

- LM358 IC 2 IR transmitter and receiver pair
- Resistors of the range of kilo-ohms.
- Variable resistors.
- LED (Light Emitting Diode).

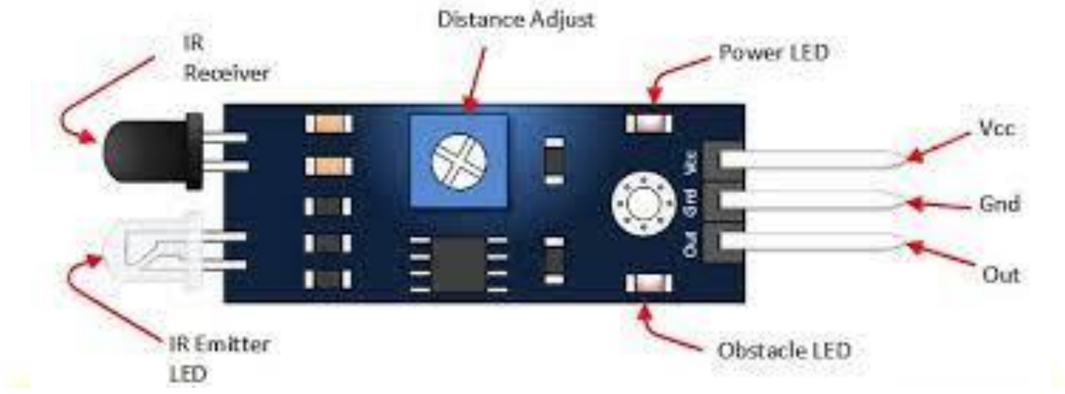
Circuit Diagram of IR Sensor:



In this project, the transmitter section includes an IR sensor, which transmits continuous IR rays to be received by an IR receiver module. An IR output terminal of the receiver varies depending upon its receiving of IR rays. Since this variation cannot be analysed as such, therefore this output can be fed to a comparator circuit. Here an operational amplifier (op-amp) of LM 339 is used as a comparator circuit.
When the IR receiver does not receive a signal, the potential at the inverting input goes higher than that non-inverting input of the comparator IC (LM339). Thus, the output of the comparator goes low, but the LED does not glow. When the IR receiver module receives a signal to the potential at the inverting input goes low. Thus, the output of the comparator (LM 339) goes high and the LED starts glowing.

Resistor R1 (100), R2 (10k), and R3 (330) are used to ensure that a minimum of 10 mA current passes through the IR LED Devices like Photodiode and normal LEDs respectively. Resistor VR2 (preset=5k) is used to adjust the output terminals. Resistor VR1 (preset=10k) is used to set the sensitivity of the circuit Diagram. Read more about IR sensor

IR Sensor Pin Diagram



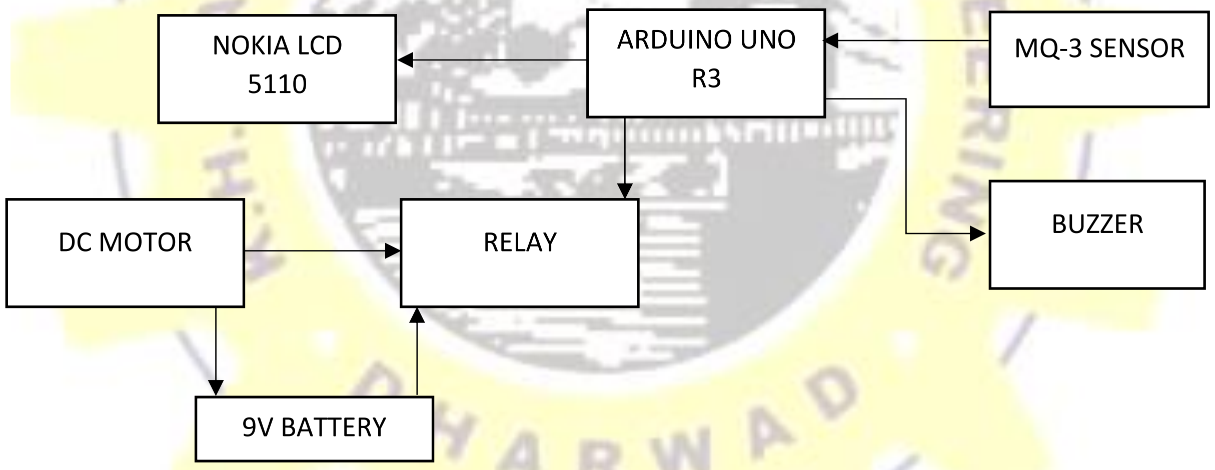
Pin Name	Description
VCC	Power Supply +5v
GND	Power Supply Ground
OUTPUT	Active High Output

Features and Applications of IR Sensors:

IR Sensor Module Features:
1. 5VDC Operating voltage
2. I/O pins are 5V and 3.3V compliant
3. Range: Up to 20cm
4. Adjustable Sensing range
5. Built-in Ambient Light Sensor
6. 20mA supply current
7. Mounting hole

Applications:
1. Obstacle Detection
2. Industrial safety devices
3. Wheel encoder.

2.2 ALCOHOL DETECTION WITH IGNITION LOCK:



2.2.1 Nokia LCD 5110:

Component description:

This Nokia 5110 LCD Display Module is mounted on an easy to solder PCB. The Nokia 5110 LCD Module uses a Philips PCD8544 LCD driver, which is designed for mobile phones.

Nokia 5110 LCD Display Module is a low-cost monochrome LCD module comprised of 84 X 48 pixels that can be used to display rich graphics and text content. This module is a revision that accepts 5V input. So no extra level shifter is needed.

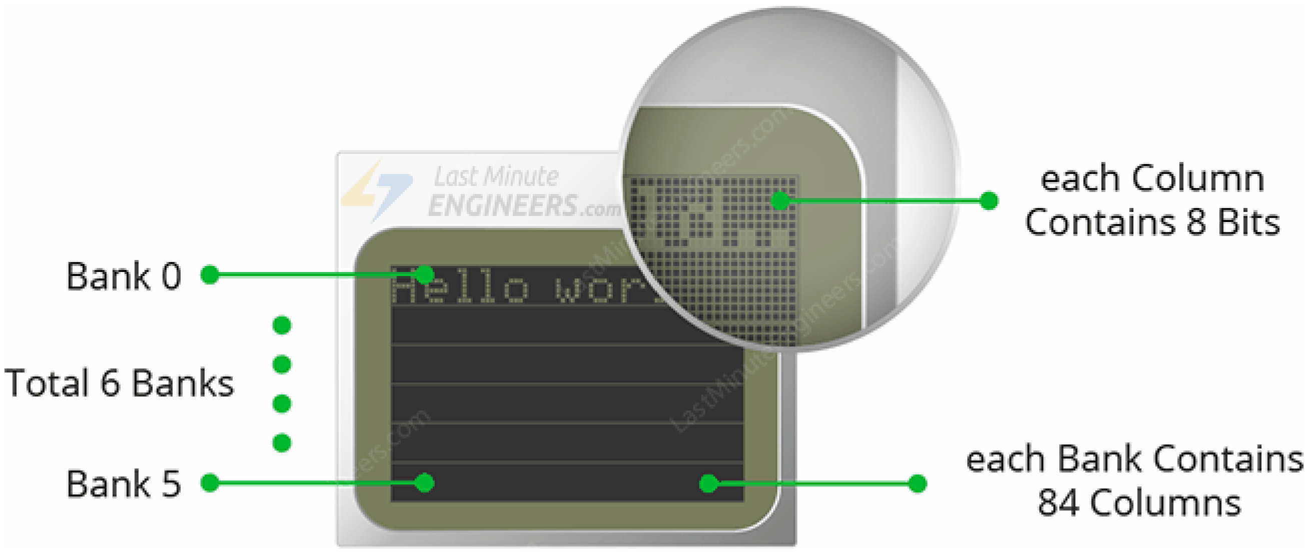
It uses the PCD8544 controller, which is the same used in the Nokia 3310 LCD. The PCD8544 is a low power CMOS LCD controller/driver, designed to drive a graphic display of 48 rows and 84 columns. All necessary functions for the display are provided in a single chip, including on-chip generation of LCD supply and bias voltages, resulting in a minimum of external components and low power consumption. The PCD8544 interfaces to microcontrollers through a serial bus interface**.**

Nokia 5110 LCD Memory Map:

The PCD8544 LCD driver has a built-in 504 bytes Graphic Display Data RAM (GDDRAM) for the screen which holds the bit pattern to be displayed. This memory area is organized in 6 banks (from 0 to 5). Each bank contains 84 columns/segments (from 0 to 83). And each column can store 8 bits of data (from 0 to 7). That surely tells us we have

6 banks x 84 segments x 8 bits of data = 4032 bits = 504 bytes

The whole memory map with banks, segments and data is highlighted below.



Each bit represents particular pixel on the screen which can be turned ON or OFF programmatically.

Working of Nokia 5110 LCD display module to Arduino Uno:

Before we get to uploading code and sending data to the display, let's hook the display up to the Arduino.

Connections are fairly simple. As we are implementing software SPI, we have flexible pin options. You can connect data transmission pins to any digital I/O pin. In our case the serial clock(CLK), serial data(DIN), data/command(DC), chip enable(CE) and reset(RST) pins are connected from pin 7 all the down to pin 3 on Arduino.

But unfortunately, the LCD has 3v communication levels, so we cannot directly connect these pins to the Arduino. We need some protection. This can be done by shifting levels.

One of the cheap and easiest way to shift levels is to add resistors inline with each data transmission pin. Just add 10kΩ resistors between the CLK, DIN, D/C, and RST pins and a 1kΩ resistor between CE.

Finally, The backlight(BL) pin is connected to 3.3V via 330Ω current limiting resistor. You can add a potentiometer or connect this pin to any PWM-capable Arduino pin, if you wish to control its brightness.

Here are the complete specifications:

Display Technology	Dot Matrix LCD
MCU Interface	SPI
Screen Size	1.5 Inch Across
Resolution	84×48 pixels
Operating Voltage	2.7V – 3.3V
Operating Current	50mA max
Viewing Angle	180°

Working of Nokia 5110 LCD display module to Arduino Uno:

Before we get to uploading code and sending data to the display, let's hook the display up to the Arduino.

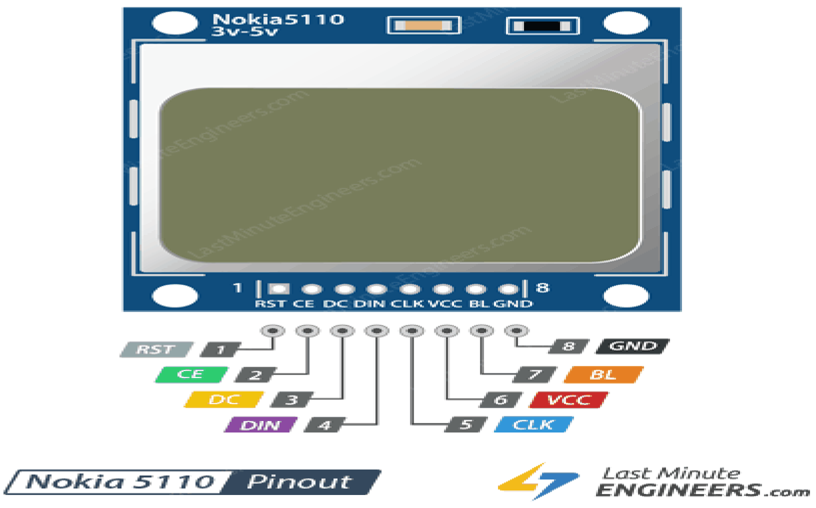
Connections are fairly simple. As we are implementing software SPI, we have flexible pin options. You can connect data transmission pins to any digital I/O pin. In our case the serial clock(CLK), serial data(DIN), data/command(DC), chip enable(CE) and reset(RST) pins are connected from pin 7 all the down to pin 3 on Arduino.

But unfortunately, the LCD has 3v communication levels, so we cannot directly connect these pins to the Arduino. We need some protection. This can be done by shifting levels.

One of the cheap and easiest way to shift levels is to add resistors inline with each data transmission pin. Just add 10kΩ resistors between the CLK, DIN, D/C, and RST pins and a 1kΩ resistor between CE.

Finally, The backlight(BL) pin is connected to 3.3V via 330Ω current limiting resistor. You can add a potentiometer or connect this pin to any PWM-capable Arduino pin, if you wish to control its brightness.

Nokia 5110 LCD Display Module Pinout:



RST pin resets the display. It's an active low pin meaning; you can reset the display by pulling it low. You can also connect this pin to the Arduino reset so that it will reset the screen automatically.

CE(Chip Enable) pin is used to select one of many connected devices sharing same SPI bus. It's an active low pin as well.

D/C(Data/Command) pin tells the display whether the data it's receiving is a command or displayable data.

DIN is a serial data pin for SPI interface.

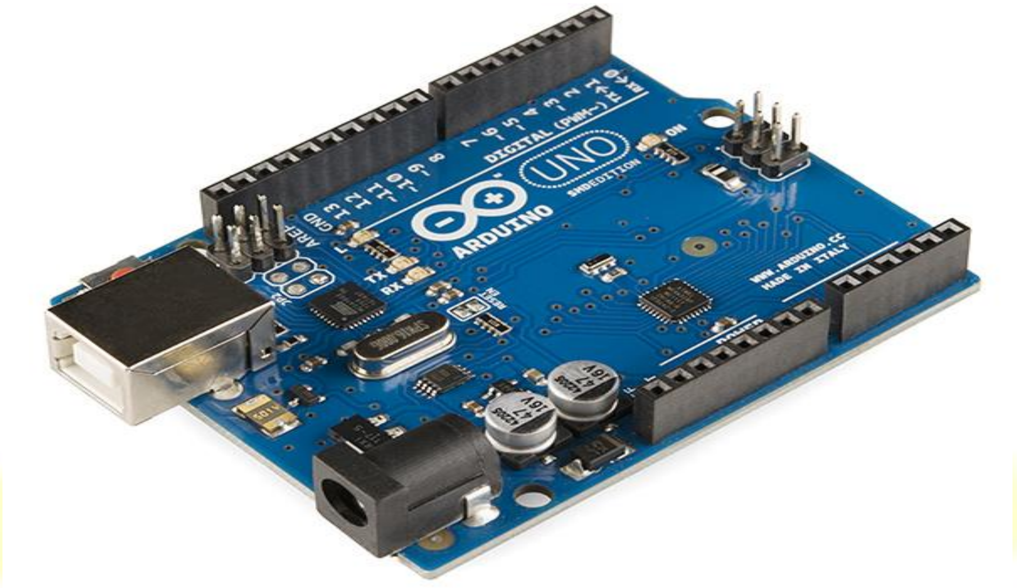
CLK is a serial clock pin for SPI interface.

VCC pin supplies power for the LCD which we connect to the 3.3V volts pin on the Arduino.

BL(Backlight) pin controls the backlight of the display. To control its brightness, you can add a potentiometer or connect this pin to any PWM-capable Arduino pin.

GND should be connected to the ground of Arduino

2.2.2 Arduino Uno R3:



The Arduino UNO R3 is the perfect board to get familiar with electronics and coding. This versatile microcontroller is equipped with the well-known ATmega328P and the AT Mega 16U2 Processor. This board will give you a great first experience within the world of Arduino.

Target areas: Maker, introduction, industries Arduino® UNO R3 2 / 13 Arduino® UNO R3 Modified: 03/08/2022 Features ATMega328P Processor Memory AVR CPU at up to 16 MHz 32KB Flash 2KB SRAM 1KB EEPROM Security Power On Reset (POR) Brown Out Detection (BOD) Peripherals 2x 8-bit Timer/Counter with a dedicated period register and compare channels 1x 16-bit Timer/Counter with a dedicated period register, input capture and compare channels 1x USART with fractional baud rate generator and start-of-frame detection 1x controller/peripheral Serial Peripheral Interface (SPI) 1x Dual mode controller/peripheral I2C 1x Analog Comparator (AC) with a scalable reference input Watchdog Timer with separate on-chip oscillator Six PWM channels Interrupt and wake-up on pin change ATMega16U2 Processor 8-bit AVR® RISC-based microcontroller Memory 16 KB ISP Flash 512B EEPROM 512B

SRAM debug WIRE interface for on-chip debugging and programming Power 2.7-5.5 volts

How to Use an Arduino Uno?

- Arduino Uno can detect the surroundings from the input. Here the input is a variety of sensors and these can affect its surroundings through controlling motors, lights, other actuators, etc. The ATmega328 microcontroller on the Arduino board can be programmed with the help of an Arduino programming language and the IDE (Integrated Development Environment). **Arduino projects** can communicate by software while running on a PC.

Arduino Programming

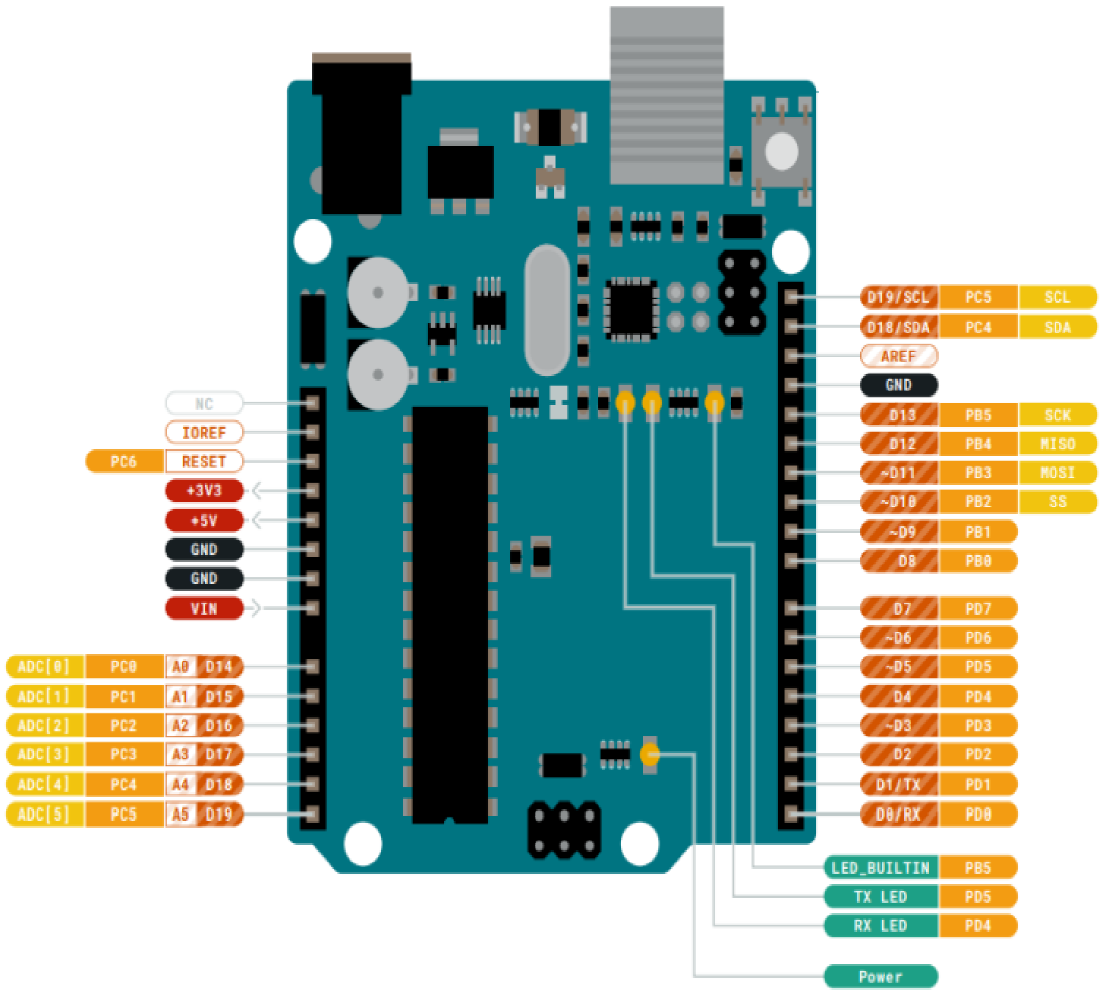
- Once the Arduino IDE tool is installed in the PC, attach the Arduino board to the computer with the help of USB cable. Open the Arduino IDE & select the right board by choosing Tools–>Board..>Arduino Uno, and select the right Port by choosing Tools–>Port. This board can be programmed with the help of an Arduino **programming language** depends on Wiring.
- To activate the Arduino board & flash the LED on the board, dump the program code with the selection of Files–> Examples.>Basics.>Flash. When the programming codes are dumped into the IDE, and then click the button 'upload' on the top bar. Once this process is completed, check the LED flash on the board.

High Voltage Protection of USB:

- The Arduino Uno board has a rearrangeable poly fuse that defends the USB port of the PC from the over-voltage. Though most of the PCs have their own inner protection, the fuse gives an additional coating of safety. If above 500mA is given to the USB port, then the fuse will routinely crack the connection until the over-voltage is removed.

Physical Characteristics:

- The physical characteristics of an Arduino board mainly include length and width. The **printed circuit board** of the Arduino Uno length and width are 2.7 X 2.1 inches, but the power jack and the USB connector will extend beyond the previous measurement. The board can be attached on the surface otherwise case with the screw holes.



Power Supply:

- The **Arduino Uno power supply** can be done with the help of a USB cable or an external power supply. The external power supplies mainly include AC to DC adapter otherwise a battery. The adapter can be connected to the Arduino Uno by plugging into the power jack of the Arduino board. Similarly, **the battery** leads can be connected to the Vin pin and the GND pin of the POWER connector. The suggested voltage range will be 7 volts to 12 volts.
- **Input & Output**
- The 14 digital pins on the Arduino Uno can be used as input & output with the help of the functions like pinMode (), digitalWrite(), & Digital Read().

- **Pin1 (TX) & Pin0 (RX) (Serial):** This pin is used to transmit & receive TTL serial data, and these are connected to the ATmega8U2 USB to TTL Serial chip equivalent pins.
- **Pin 2 & Pin 3 (External Interrupts):** External pins can be connected to activate an interrupt over a low value, change in value.
- **Pins 3, 5, 6, 9, 10, & 11 (PWM):** This pin gives 8-bit PWM o/p by the function of analogWrite().
- **SPI Pins (Pin-10 (SS), Pin-11 (MOSI), Pin-12 (MISO), Pin-13 (SCK):** These pins maintain SPI-communication, even though offered by the fundamental hardware, is not presently included within the Arduino language.
- **Pin-13(LED):** The inbuilt LED can be connected to pin-13 (digital pin). As the HIGH-value pin, the light emitting diode is activated, whenever the pin is LOW.
- **Pin-4 (SDA) & Pin-5 (SCL) (I2C):** It supports TWI-communication with the help of the Wire library.

- **AREF (Reference Voltage):** The reference voltage is for the analog i/ps with analog Reference().
- **Reset Pin:** This pin is used for reset (RST) the microcontroller.
- **Memory**
- The memory of this Atmega328 Arduino microcontroller includes flash memory-32 KB for storing code, SRAM-2 KB EEPROM-1 KB.

Communication:

- The Arduino Uno ATmega328 offers UART TTL-**serial communication**, and it is accessible on digital pins like TX (1) and RX (0). The software of an Arduino has a serial monitor that permits easy data. There are two LEDs on the board like RX & TX which will blink whenever data is being broadcasted through the USB.

A SoftwareSerial library permits for serial communication on Arduino Uno digital pins and the ATmega328P supports TWI (I2C) as well as **SPI-communication**. The Arduino software contains a wired library for simplifying the utilization of the I2C bus

Features of Arduino Uno Board:

The **features of Arduino Uno ATmega328** includes the following.

- The operating voltage is 5V
- The recommended input voltage will range from 7v to 12V
- The input voltage ranges from 6v to 20V
- Digital input/output pins are 14
- Analog i/p pins are 6
- DC Current for each input/output pin is 40 mA
- DC Current for 3.3V Pin is 50 mA
- Flash Memory is 32 KB
- SRAM is 2 KB
- EEPROM is 1 KB
- CLK Speed is 16 MHz

Applications of Arduino Uno ATmega328:

The **applications of Arduino Uno** include the following.

- **Arduino Uno** is used in Do-it-Yourself projects prototyping.
- In developing projects based on code-based control
- Development of Automation System
- Designing of basic circuit designs.

2.2.3 MQ-3 Sensor

Give your next Arduino project a nose for alcohol with the MQ3 alcohol sensor module. This sensor detects the presence and the concentration of alcohol present in the air. So, if you are planning to make your own breathalyzer to measure the amount of alcohol in the human body, the MQ3 alcohol sensor module is a great option.

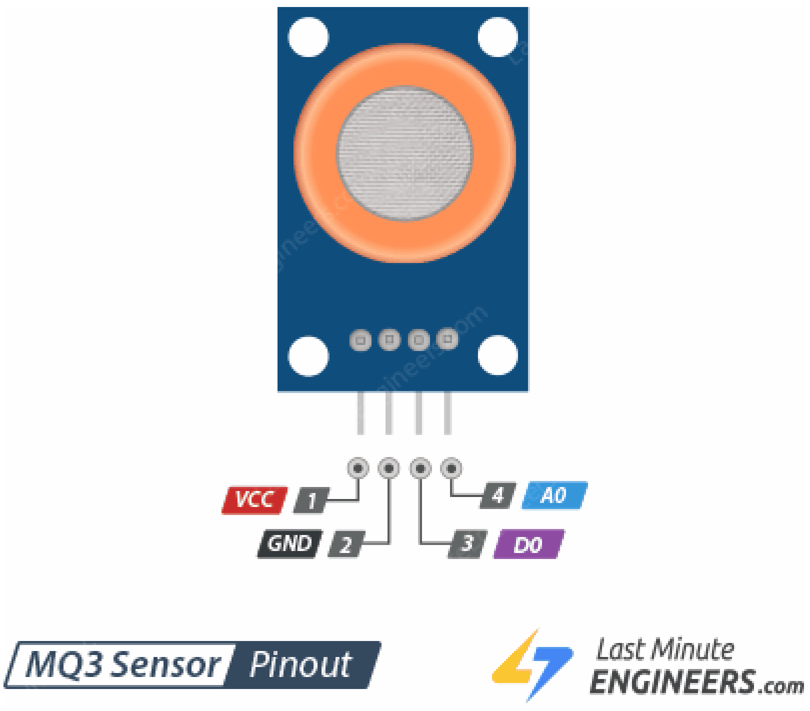
MQ3 Alcohol Sensor:

MQ3 is one of the most commonly used sensors in the MQ sensor series. It is a Metal Oxide Semiconductor (MOS) type of sensor. Metal oxide sensors are also known as Chemiresistors, because sensing is based on the change of resistance of the sensing material when exposed to alcohol. So by placing it in a simple voltage divider network, alcohol concentrations can be detected.

MQ3 alcohol sensor works on 5V DC and draws around 800mW. It can detect Alcohol concentrations anywhere from 25 to 500 ppm.

MQ3 Alcohol Sensor Module Pinout:

Now let's have a look at the pinout.



VCC supplies power for the module. You can connect it to 5V output from your Arduino.

GND is the Ground Pin and needs to be connected to GND pin on the Arduino.

D0 provides a digital representation of the presence of alcohol.

Technical Specifications:

The MQ3 alcohol sensor technical specifications are listed below.

- It requires a power supply of 5VDC (@ 165mA heater ON / 60mA heater off).
- Consumes 150mA current.
- Digital output Do: 0 and 1 TTL digital (0.1V and 5V).
- Analog output Ao: 0.1V to 0.3V (relates to pollution), voltage concentration is maximum of 4V.
- Alcohol Concentration detection: 0.05 mg/L to 10 mg/L.
- Interface: one TTL compatible input (HSW) and one TTL compatible output (ALR).
- Heater consumes: <750mW.
- Resistance of the heater: 33ohms±5%.

Use/Applications:

The **applications of the MQ3 alcohol sensor** are given below,

- Used as a gas level over-limit alarm.
- Portable alcohol detector.
- Breathalyzer.
- Stand-alone sensing module.
- Used in environmental monitoring equipment.
- Vehicle alcohol detector.

2.2.4 Buzzer

This buzzer is an active buzzer, which basically means that it will buzz at a predefined frequency (2300 ±300 Hz) on its own even when you just apply steady DC power. If you are looking for a buzzer can produce varied tones from an oscillating input signal, then take a look at our passive buzzer.

Some people prefer to get active buzzers since they can use them with steady DC power but also be able to produce some variety of tones by applying an oscillating signal. Some consider them to be more versatile than their cousin, the passive buzzer, which is the type that requires an oscillating signal to create any tone.

It is possible, and often done, to still create different tones through an active buzzer when you apply an oscillating signal to the buzzer, but the spectrum of possible different tones is very limited and not as crisp or clean of sound as can be produced with a passive buzzer.

One advantage to an active buzzer is that you can still produce a sound from the buzzer connected to a microcontroller, such as an Arduino, by just driving a standard high output on the connected pin. The benefits of this are that you don't need to use processing power, hardware timers, or additional code to produce sound.

SPECIFICATIONS:

- Longer pin is the positive pin

Rated Voltage	5 V
Operating Voltage	4~8 V
Max Rated Current	≤32 mA
Min. Sound Output at 10cm	85 dB
Resonant Frequency	2300 ±300 Hz
Operating Temperature	-20°C to 45°C
Dimensions (Excluding Pins)	
Height	9.16 mm (0.36")
Diameter	11.78 mm (0.46")
Weight	1.6 g (0.057 oz)

Advantages

The **advantages of a buzzer** include the following.

- Simply Compatible
- Frequency Response is Good
- Size is small
- Energy Consumption is less
- The Range of Voltage usage is Large
- Sound Pressure is high

2.2.5 DC MOTOR

A DC Motor is any of a class of rotary electrical motors that converts direct current (DC) electrical energy into mechanical energy. The most common types rely on the forces produced by magnetic fields. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the direction of current in part of the motor.

DC motors were the first form of motor widely used, as they could be powered from existing direct-current lighting power distribution systems. A DC motor's speed can be controlled over a wide range, using either a variable supply voltage or by changing the strength of current in its field windings. Small DC motors are used in tools, toys, and appliances. The universal motor can operate on direct current but is a lightweight brushed motor used for portable power tools and appliances. Larger DC motors are currently used in propulsion of electric vehicles, elevator and hoists, and in drives for steel rolling mills. The advent of power electronics has made replacement of DC motors with AC motors possible in many applications.

Some Types of DC Motors Available at market:

1. Permanent Magnet DC Motors

The permanent magnet motor uses a permanent magnet to create field flux. This type of DC motor provides great starting torque and has good speed regulation, but torque is limited so they are typically found on low horsepower applications.

2. Series DC Motors

In a series DC motor, the field is wound with a few turns of a large wire carrying the full armature current. Typically, series DC motors create a large amount of starting torque, but cannot regulate speed and can even be damaged by running with no load. These limitations mean that they are not a good option for variable speed drive applications.

3. Shunt DC Motors

In shunt DC motors the field is connected in parallel (shunt) with the armature windings. These motors offer great speed regulation due to the fact that the shunt field can be excited separately from the armature windings, which also offers simplified reversing controls.

4. Compound DC Motors

Compound DC motors, like shunt DC motors, have a separately excited shunt field. Compound DC motors have good starting torque but may experience control problems in variable speed drive applications.

Between the 4 types of DC motors, the potential applications are numerous. Each type of DC motor has its strengths and weaknesses. Understanding these can help you understand which types may be good for your application.

2.2.6 RELAY:

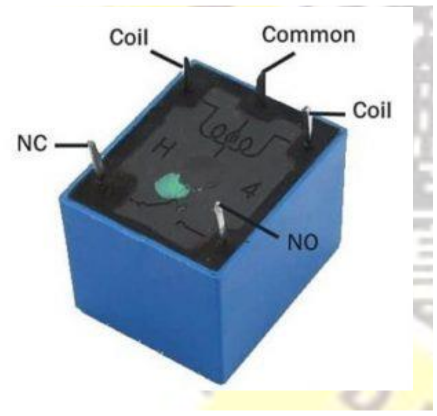
Relay is one kind of electro-mechanical component that functions as a switch. The relay coil is energized by DC so that contact switches can be opened or closed. A single channel 5V relay module generally includes a coil, and two contacts like normally open (NO) and normally closed (NC). This article discusses an overview of the 5V relay module & it's working but before going to discuss what is relay module is, first we have to know what is relay and its pin configuration.

What is a 5V Relay?

A 5v relay is an automatic switch that is commonly used in an automatic control circuit and to control a high-current using a low-current signal. The input voltage of the relay signal ranges from 0 to 5V.

5V Relay Pin Configuration

The pin configuration of the 5V relay is shown below. This relay includes 5-pins where each pin and its functionality are shown below.



Pin1 (End 1): It is used to activate the relay; usually this pin one end is connected to 5Volts whereas another end is connected to the ground.
Pin2 (End 2): This pin is used to activate the Relay.
Pin3 (Common (COM)): This pin is connected to the main terminal of the Load to make it active.

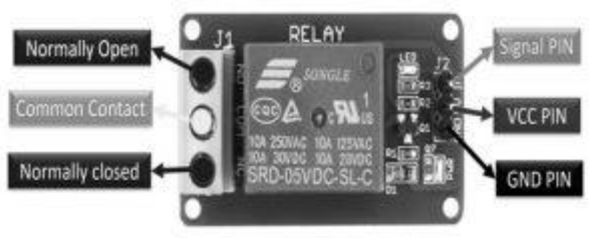
Pin4 (Normally Closed (NC)): This second terminal of the load is connected to either NC/ NO pins. If this pin is connected to the load, then it will be ON before the switch.
Pin5 (Normally Open (NO)): If the second terminal of the load is allied to the NO pin, then the load will be turned off before the switch.

Features
The **features of the 5V relay** include the following.

- Normal Voltage is 5V DC
- Normal Current is 70mA
- AC load current Max is 10A at 250VAC or 125V AC
- DC load current Max is 10A at 30V DC or 28V DC
- It includes 5-pins & designed with plastic material
- Operating time is 10msec
- Release time is 5msec
- Maximum switching is 300 operating per minute

5V Relay Module
The relay module with a single channel board is used to manage high voltage, current loads like solenoid valves, motor, AC load & lamps. This module is mainly designed to interface through different microcontrollers like PIC, Arduino, etc.

5V Relay Module Pin Configuration
The pin configuration of the 5V relay module is shown below. This module includes 6-pins where each pin and its functionality are discussed below.



Relay Module Pin Diagram

Normally Open (NO): This pin is normally open unless we provide a signal to the relay modules signal pin. So, the common contact pin smashes its link through the NC pin to make a connection through the NO pin
Common Contact: This pin is used to connect through the load that we desire to switch by using the module.
Normally Closed (NC): This NC pin is connected through the COM pin to form a closed circuit. However, this NC connection will break once the relay is switched through providing an active high/low signal toward the signal pin from a microcontroller.
Signal Pin: The signal pin is mainly used for controlling the relay. This pin works in two cases like active low otherwise active high. So, in active low case, the relay activates once we provide an active low signal toward the signal pin, whereas, in an active high case, the relay will trigger once we provide a high signal toward the signal pin.
However, these modules generally work on an active high signal which will strengthen the relay coil to make contact with the common terminal with the normally open terminal.

5V VCC: This pin needs 5V DC to work. So 5V DC power supply is provided to this pin.
Ground: This pin connects the GND terminal of the power supply.

3. EMBEDDED SYSTEM:

An **embedded system** is a computer system a combination of a computer processor, computer memory, and input/output peripheral devices that has a dedicated function within a larger mechanical or electronic system. It is *embedded* as part of a complete device often including electrical or electronic hardware and mechanical parts. Because an embedded system typically controls physical operations of the machine that it is embedded within, it often has real-time computing constraints. Embedded systems control many devices in common use today. In 2009, it was estimated that ninety-eight percent of all microprocessors manufactured were used in embedded systems.

Modern embedded systems are often based on microcontrollers (i.e., microprocessors with integrated memory and peripheral interfaces), but ordinary microprocessors (using external chips for memory and peripheral interface circuits) are also common, especially in more complex systems. In either case, the processor(s) used may be types ranging from general purpose to those specialized in a certain class of computations, or even custom designed for the application at hand. A common standard class of dedicated processors is the digital signal processor (DSP).

HISTORY:

The origins of the microprocessor and the microcontroller can be traced back to the MOS integrated circuit, which is an integrated circuit chip fabricated from MOSFETs (metal-oxide-semiconductor field-effect transistors) and was developed in the early 1960s. By 1964, MOS chips had reached higher transistor density and lower manufacturing costs than bipolar chips. MOS chips further increased in complexity at a rate predicted by Moore's law, leading to large-scale integration (LSI) with hundreds of transistors on a single MOS chip by the late 1960s. The application of MOS LSI chips to computing was the basis for the first microprocessors, as engineers began recognizing that a complete computer processor system could be contained on several MOS LSI chips.

The first multi-chip microprocessors, the Four-Phase Systems AL1 in 1969 and the Garrett Ai Research MP944 in 1970, were developed with multiple MOS LSI chips. The first single-chip microprocessor was the Intel 4004, released in 1971. It was developed by Federico Fagin, using his silicon-gate MOS technology, along with Intel engineers Marian Hoff and Stan Major, and Busycon engineer Masatoshi Shema.

DEVELOPMENT:

Since these early applications in the 1960s, embedded systems have come down in price and there has been a dramatic rise in processing power and functionality. An early microprocessor, the Intel 4004 (released in 1971), was designed for calculators and other small systems but still required external memory and support chips. By the early 1980s, memory, input and output system components had been integrated into the same chip as the processor forming a microcontroller. Microcontrollers find applications where a general-purpose computer would be too costly. As the cost of microprocessors and microcontrollers fell, the prevalence of embedded systems increased.

Today, a comparatively low-cost microcontroller may be programmed to fulfil the same role as a large number of separate components. With microcontrollers, it became feasible to replace, even in consumer products, expensive knob-based analog components such as potentiometers and variable capacitors with up/down buttons or knobs read out by a microprocessor. Although in this context an embedded system is usually more complex than a traditional solution, most of the complexity is contained within the microcontroller itself. Very few additional components may be needed and most of the design effort is in the software. Software prototype and test can be quicker compared with the design and construction of a new circuit not using an embedded processor.

APPLICATIONS:

- Embedded systems are commonly found in consumer, industrial, automotive, home appliances, medical, telecommunication, commercial, aerospace and military applications.
- Telecommunications systems employ numerous embedded systems from telephone switches for the network to cell phones at the end user. Computer networking uses dedicated routers and network bridges to route data.
- Medical equipment uses embedded systems for monitoring, and various medical imaging (positron emission tomography (PET), single-photon emission computed tomography (SPECT), computed tomography (CT), and magnetic resonance imaging (MRI) for non-invasive internal inspections. Embedded systems within medical equipment are often powered by industrial computers.

CHARACTERISTICS:

- Embedded systems are designed to do some specific tasks, rather than be a general-purpose computer for multiple tasks. Some also have real-time performance constraints that must be met, for reasons such as safety and usability; others may have low or no performance requirements, allowing the system hardware to be simplified to reduce costs.
- Embedded systems are not always standalone devices. Many embedded systems consist of small parts within a larger device that serves a more general purpose. For example, the Gibson Robot Guitar features an embedded system for tuning the strings, but the overall purpose of the Robot Guitar is, of course, to play music. Similarly, an embedded system in an automobile provides a specific function as a subsystem of the car itself.

e-con Systems eSOM270 & eSOM300 Computer on Modules

- The program instructions written for embedded systems are referred to as firmware, and are stored in read-only memory or flash memory chips. They run with limited computer hardware resources: little memory, small or non-existent keyboard or screen.

PROCESSORS OF EMBEDDED SYSTEM:

Examples of properties of typical embedded computers when compared with general-purpose counterparts, are low power consumption, small size, rugged operating ranges, and low per-unit cost. This comes at the price of limited processing resources.

Numerous microcontrollers have been developed for embedded systems use. General-purpose microprocessors are also used in embedded systems, but generally, require more support circuitry than microcontrollers.

PERIPHERIALS:

Embedded systems talk with the outside world via peripherals, such as:

- Serial communication interfaces (SCI): RS-232, RS-422, RS-485, etc.
- Synchronous Serial Interface: I2C, SPI, SSC and ESSI (Enhanced Synchronous Serial Interface)
- Universal Serial Bus (USB)
- Media cards (SD cards, CompactFlash, etc.)
- Network interface controller: Ethernet, Wi-Fi, etc.
- Fieldbuses: CAN bus, LIN-Bus, PROFIBUS, etc.
- Timers: Phase-locked loops, programmable interval timers
- General Purpose Input/Output (GPIO)
- Analog-to-digital and digital-to-analog converters
- Debugging: JTAG, In-system programming, background debug mode interface port, BITP, and DB9 ports.

DEBUGGING:

Embedded debugging may be performed at different levels, depending on the facilities available. Considerations include: does it slow down the main application, how close is the debugged system or application to the actual system or application, how expressive are the triggers that can be set for debugging (e.g., inspecting the memory when a particular program counter value is reached), and what can be inspected in the debugging process (such as, only memory, or memory and registers, etc.).

From simplest to most sophisticated debugging techniques and systems be roughly grouped into the following areas:

- Interactive resident debugging, using the simple shell provided by the embedded operating system (e.g., Forth and Basic)
- Software-only debuggers have the benefit that they do not need any hardware modification but have to carefully control what they record in order to conserve time and storage space.[10]
- External debugging using logging or serial port output to trace operation using either a monitor in flash or using a debug server like the Remedy Debugger that even works for heterogeneous multicore systems.
- An in-circuit debugger (ICD), a hardware device that connects to the microprocessor via a JTAG or Nexus interface.[11] This allows the operation of the microprocessor to be controlled externally, but is typically restricted to specific debugging capabilities in the processor.

3.1 ATMEGA328P MICROPROCESSOR:

The **ATmega328** is a single-chip microcontroller created by Atmel in the megaAVR family (later Microchip Technology acquired Atmel in 2016). It has a modified Harvard architecture 8-bit RISC processor core.

SPECIFICATION:

The Atmel 8-bit AVR RISC-based microcontroller combines 32 KB ISP flash memory with read-while-write capabilities, 1 KB EEPROM, 2 KB SRAM, 23 general-purpose I/O lines, 32 general-purpose working registers, 3 flexible timer/counters with compare modes, internal and external interrupts, serial programmable USART, a byte-oriented 2-wire serial interface, SPI serial port, 6-channel 10-bit A/D converter (8 channels in TQFP and QFN/MLF packages), programmable watchdog timer with internal oscillator, and 5 software-selectable power-saving modes. The device operates between 1.8 and 5.5 volts. The device achieves throughput approaching 1 MIPS/MHz

FEATURES:

Parameter	Value
CPU type	8-bit AVR
Maximum CPU speed	20 MHz
Performance	20 MIPS at 20 MHz
Flash memory	32 KB
SRAM	2 KB
EEPROM	1 KB
Package pin count	28 or 32
Capacitive touch sensing channels	16
Maximum I/O pins	23
External interrupts	2
USB interface	No

FAMILY:

A common alternative to the ATmega328 is the "Pico Power" ATmega328P. A comprehensive list of all other members of the megaAVR series can be found on the Atmel website.[3]

- ATmega32
- ATmega328P and ATmega328P-AUTOMOTIVE
- ATmega328PB and ATmega328PB-AUTOMOTIVE (superset of ATmega328P) - has more UART, I2C, and SPI peripherals than ATmega328P

APPLICATIONS:

- ATmega328 is commonly used in many projects and autonomous systems where a simple, low-powered, low-cost micro-controller is needed.
- Perhaps the most common implementation of this chip is on the popular Arduino development platform, namely the Arduino Uno, Arduino Pro Mini and Arduino Nano models.

PROGRAMMING:

Reliability qualification shows that the projected data retention failure rate is much less than 1 PPM over 20 years at 85 °C or 100 years at 25 °C.

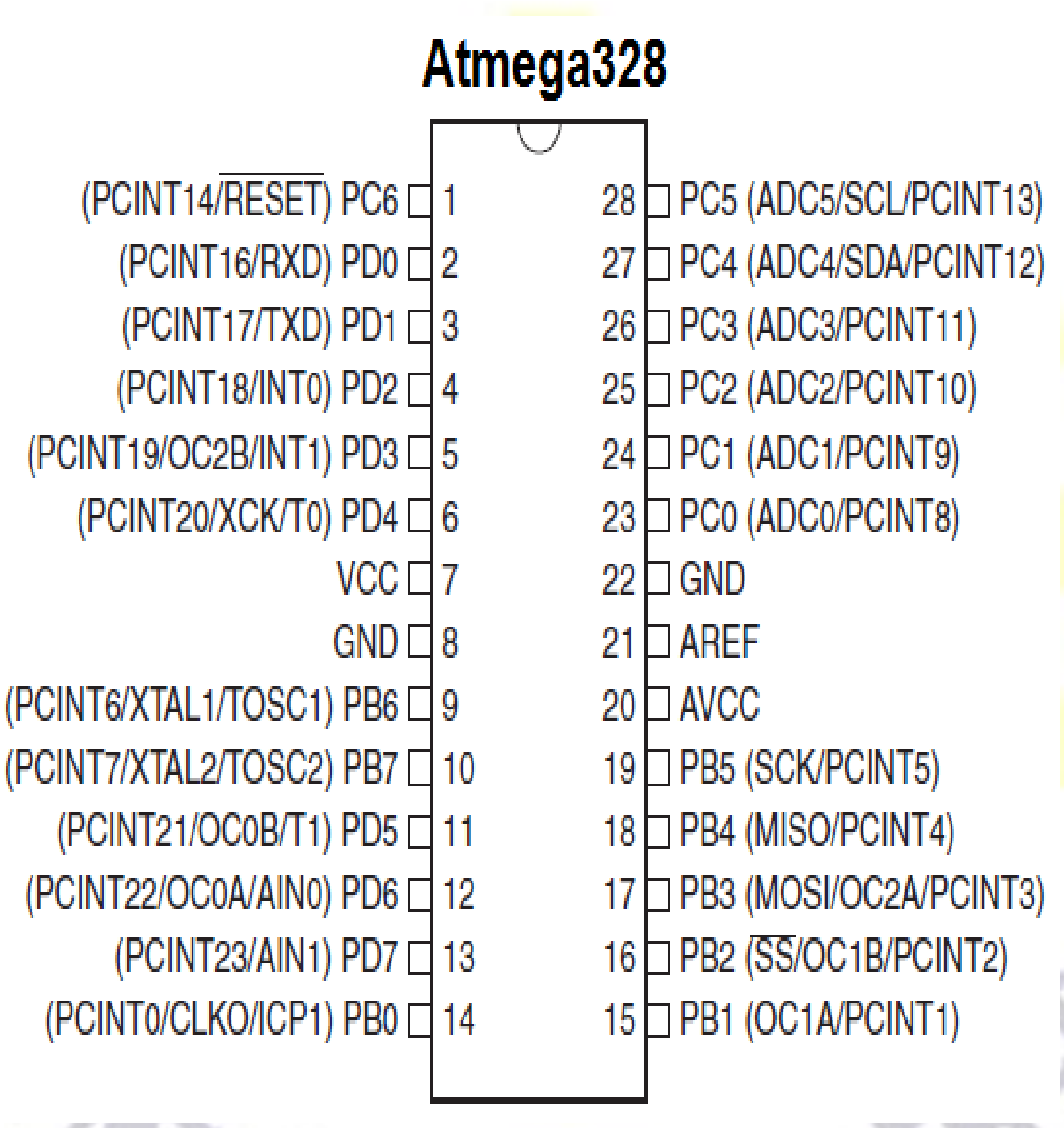
Parallel program mode			
Programming signal	**Pin Name**	**I/O**	**Function**
RDY/BSY	PD1	O	High means the MCU is ready for a new command, otherwise busy.
OE	PD2	I	Output enables (active low)
WR	PD3	I	Write pulse (active low)
BS1	PD4	I	Byte select 1 ("0" = Low byte, "1" = High byte)
XA0	PD5	I	XTAL action bit 0
XA1	PD6	I	XTAL action bit 1
PAGEL	PD7	I	Program memory and EEPROM data page load
BS2	PC2	I	Byte select 2 ("0" = low byte, "1" = 2nd high byte)
DATA	PC [1:0]: PB [5:0]	I/O	Bi-directional data bus (output when OE is low)

Programming mode is entered when PAGEL (PD7), XA1 (PD6), XA0 (PD5), BS1 (PD4) is set to zero. RESET pin to 0 V and V_{CC} to 0 V. V_{CC} is set to 4.5–5.5 V. Wait 60 µs, and RESET is set to 11.5–12.5 V. Wait more than 310 µs. Set XA1:XA0:BS1:DATA = 100 1000 0000, pulse XTAL1 for at least 150 ns, pulse WR to zero. This starts the chip erase. Wait until RDY/BSY (PD1) goes high. XA1:XA0:BS1:DATA = 100 0001 0000, XTAL1 pulse, pulse WR to zero. This is the flash write command. And so on.

Serial programming			
Symbol	**Pins**	**I/O**	**Description**
MOSI	PB3	I	Serial data in
MISO	PB4	O	Serial Data out
SCK	PB5	I	Serial Clock

Serial data to the MCU is clocked on the rising edge and data from the MCU is clocked on the falling edge. Power is applied to V_{CC} while RESET and SCK are set to zero. Wait for at least 20 ms and then the programming enables serial instruction 0xAC, 0x53, 0x00, 0x00 is sent to the MOSI pin. The second byte (0x53) will be echoed back by the MCU.

PINOUT DIAGRAM:



3.2 What is Arduino?

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments.

Why Arduino?

Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low-cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step-by-step instructions of a kit, or sharing ideas online with other members of the Arduino community.

There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Net media's BX-24, Phi gets, MIT's Handy board, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package. Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems:

- **Inexpensive** - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than $50.

- **Cross-platform** - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.

- **Simple, clear programming environment** - The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.

- **Open source and extensible software** - The Arduino software is published as open-source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.

- **Open source and extensible hardware** - The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module in order to understand how it works and save money.

Official Boards:

The original Arduino hardware was manufactured by the Italian company Smart Projects. Some Arduino-branded boards have been designed by the American companies Spark Fun Electronics and Adafruit Industries. As of 2016, 17 versions of the Arduino hardware have been commercially produced.

- Arduino RS232
- Arduino Diecimila
- Arduino Duemilanove
- Arduino UNO R2
- Arduino UNO SMD R3
- Arduino Leonardo
- Arduino Micro
- Arduino Pro Micro
- Arduino Pro
- Arduino Mega
- Arduino NANO
- Arduino Lilypad 00
- Arduino Robot
- Arduino Esplora
- Arduino Ethernet
- Arduino Yún
- Arduino Due

<u>SOFTWARE:</u>

A program for Arduino hardware may be written in any programming language with compilers that produce binary machine code for the target processor. Atmel provides a development environment for their 8-bit AVR and 32-bit ARM Cortex-M based microcontrollers: AVR Studio (older) and Atmel Studio (newer).

IDE:

The Arduino integrated development environment (IDE) is a cross-platform application (for Microsoft Windows, macOS, and Linux) that is written in the Java programming language. It originated from the IDE for the languages Processing and Wiring. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple one-click mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus. The source code for the IDE is released under the GNU General Public License, version 2.

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program argued to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

From version 1.8.12, Arduino IDE windows compiler supports only Windows 7 or newer OS. On Windows Vista or older one gets "Unrecognized Win32 application" error when trying to verify/upload program. To run IDE on older machines, users can either use version 1.8.11, or copy "Arduino-builder" executable from version 11 to their current install folder as it's independent from IDE.

IDE 2.0:

On October 18, 2019, Arduino Pro IDE (alpha preview) was released. Later, on March 1, 2021, the beta preview was released, renamed IDE 2.0. The system still uses Arduino CLI (Command Line Interface), but improvements include a more professional development environment, autocompletion support, and Git integration.[64] The application frontend is based on the Eclipse Theia Open-Source IDE. The main features available in the new release are:

- Modern, fully featured development environment
- Dual Mode, Classic Mode (identical to the Classic Arduino IDE) and Pro Mode (File System view)
- New Board Manager
- New Library Manager
- Board List
- Basic Auto-Completion (Arm targets only)
- Git Integration
- Serial Monitor
- Dark Mode

SKETCHES:

A sketch is a program written with the Arduino IDE. Sketches are saved on the development computer as text files with the file extension **.ino**. Arduino Software (IDE) pre-1.0 saved sketches with the extension **.pde**.

A minimal Arduino C/C++ program consists of only two functions:

- setup(): This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch. It is analogous to the function main().
- loop(): After setup() function exits (ends), the loop() function is executed repeatedly in the main program. It controls the board until the board is powered off or is reset. It is analogous to the function while(1).

Blink example

Power LED (red) and User LED (green) attached to pin 13 on an Arduino compatible board

Most Arduino boards contain a light-emitting diode (LED) and a current limiting resistor connected between pin 13 and ground, which is a convenient feature for many tests and program functions. A typical program used by beginners, akin to Hello, World!, is "blink", which repeatedly blinks the on-board LED integrated into the Arduino board. This program uses the functions `pinMode()`, `digitalWrite()`, and `delay()`, which are provided by the internal libraries included in the IDE environment. This program is usually loaded into a new Arduino board by the manufacturer.

Program to Blink LED:

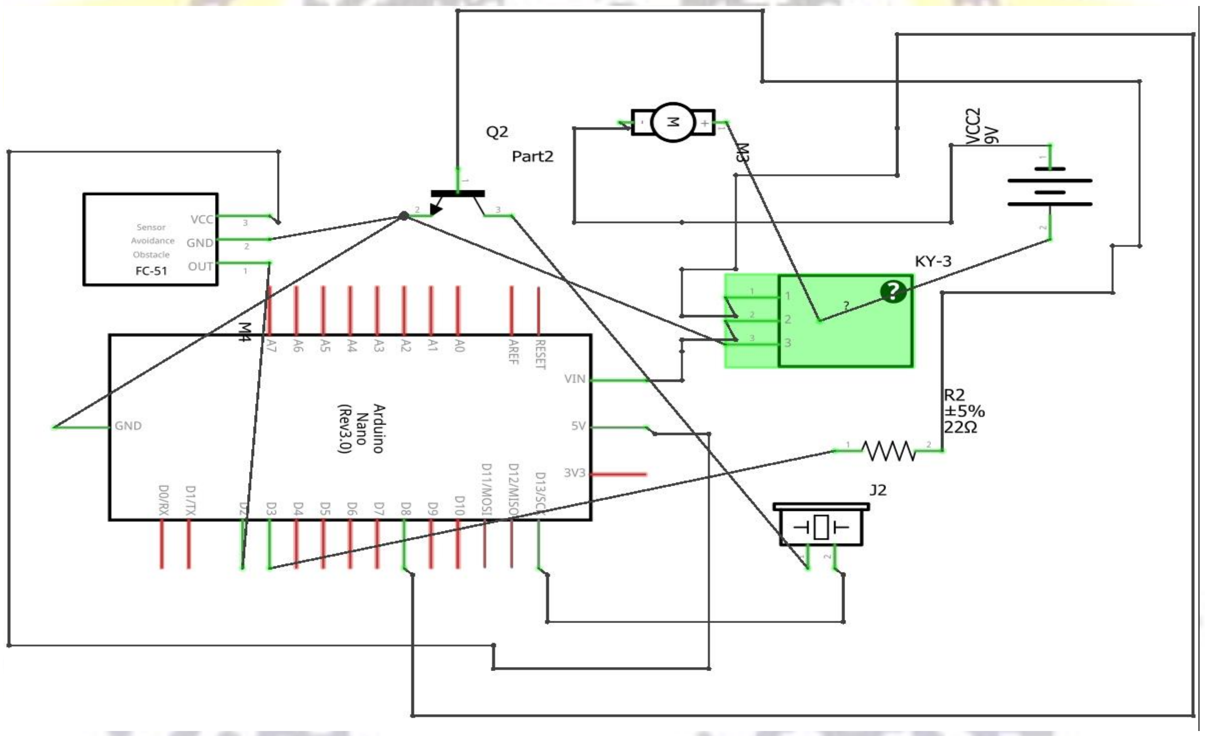
```
# define LED_PIN 13                   // Pin number attached to LED.

void setup () {
    pinMode(LED_PIN, OUTPUT);        // Configure pin 13 to be a digital
output.
}

void loop() {
    digitalWrite(LED_PIN, HIGH);     // Turn on the LED.
    delay(1000);                     // Wait 1 second (1000 milliseconds).
    digitalWrite(LED_PIN, LOW);      // Turn off the LED.
    delay(1000);                     // Wait 1 second.
}
```

4. PROGRAM FOR ANTI SLEEP DETECTION WITH IGNITION LOCK SYSTEM

```
#define sensor 2  //the AOUT pin of the IR sensor goes into analog pin D2 of the arduino
#define buz 13   //the DOUT pin of the buzzer goes into digital pin D13 of the arduino
#define relay 8  //the DOUT pin of the relay goes into digital pin D8 of the arduino
int Val;

void setup()
{
  Serial.begin(9600);  //sets the baud rate
  pinMode(sensor, INPUT);  //sets the pin as an input to the arduino
  pinMode(buz, OUTPUT);  //sets the pin as an output of the arduino
  pinMode(relay, OUTPUT);  //sets the pin as an output of the arduino
}
void loop() {
Val=digitalRead(sensor);  //reads the digital value from the IR sensor's DOUT pin


  if(Val==0)
  {
    delay (3000);  //sets the delay time
    digitalWrite(buz, HIGH);  //if limit has been reached, buzzer turns on as status indicator
    digitalWrite(relay, HIGH);  //if limit has been reached, relay turns on as status indicator
    Serial.println("DETECTED");  //prints DECTCTED on Serial Monitor
}
```

```
 else
 {
  digitalWrite(buz, LOW);  //if limit has been not reached, buzzer turns off as status indicator
  digitalWrite(relay, LOW);  //if limit has been not reached, relay turns on as status indicator
  Serial.println("NOT DETECTED");  //prints NOT DECTCTED on Serial Monitor


   }
}
```

4.1 CIRCUIT DIAGRAM OF ANTI SLEEP DETECTION DEVICE:



4.2 Connections of anti-sleep:

Steps:

- Connection of IR sensors to the Auridon nano

1. pin of VCC IR sensor is connected to the 5v pin of Arduino nano.
2. pin of GND IR sensor is connected to the GND pin of Arduino nano.
3. pin of OUT IR sensor is connected to the D2 pin of Arduino nano.

- Connection of transistor and resistor to the Arduino nano board
    1. Emitter pin of transistor is connected to the negative terminals of GND pin of Arduino nano
    2. Base pin is connected to the one end of 4.7k ohm resistor
    3. Another end of 4.7kohm resistor is connected to the D3 pin of Arduino nano
    4. Collector pin is connected to the negative terminal of buzzer, positive terminal of buzzer is connected to the D13 pin of Arduino nano

- Connection of relay and motor to the Arduino nano
    1. Positive terminal of relay is connected to the 5v pin of the Arduino nano.
    2. negative terminal of relay is connected to the GND pin of the Arduino nano.
    3. Signal terminal of relay is connected to the D8 pin of the Arduino nano.
    4. One end of the DC motor is connected to the common terminal of the relay module
    5. Negative terminal of the battery is connected to another end of the DC motor
    6. Positive terminal of the battery is connected to the open close terminal of the relay

4.3 Internal working:

At the beginning all the device gets power thought the Arduino board. the IR sensor switch's gets switch ON and starts the continuous IR ray's transmission.

The IR sensors signal is connected to the 'D2' pin of the Arduino board. It acts as the input Source. The relay and the buzzer are connected to the 'D8' and 'D13' pins of board Simultaneously.

As the IR sensor gets block by an object while transmitting the signal the photo-diode to the reflected rays and send the information to the electricity and the relay will start to flow the electricity and buzzer gets beeped continuously. Then running DC motor will instantly stop functioning which will indicate the turning off vehicle engine.

As the object is removed from the blocking The IR sensor, the complete circuit will gets reset and the device will go back to the initial state.

4.4 Advantages:

- ➢ It can reduce the number of accidents, whether it is night or day time
- ➢ If this system is in practice, the driver will be always alert that he can not rely on that system because that system may fail at any time due to technical disasters. So the driver always keep his attention towards it.
- ➢ So, the waste of fuel will be lessened automatically because applying the brakes. Because applying brakes will cause lots of fuel consumption.
- ➢ By this system we can achieve low cost, compact size and transport process easily.

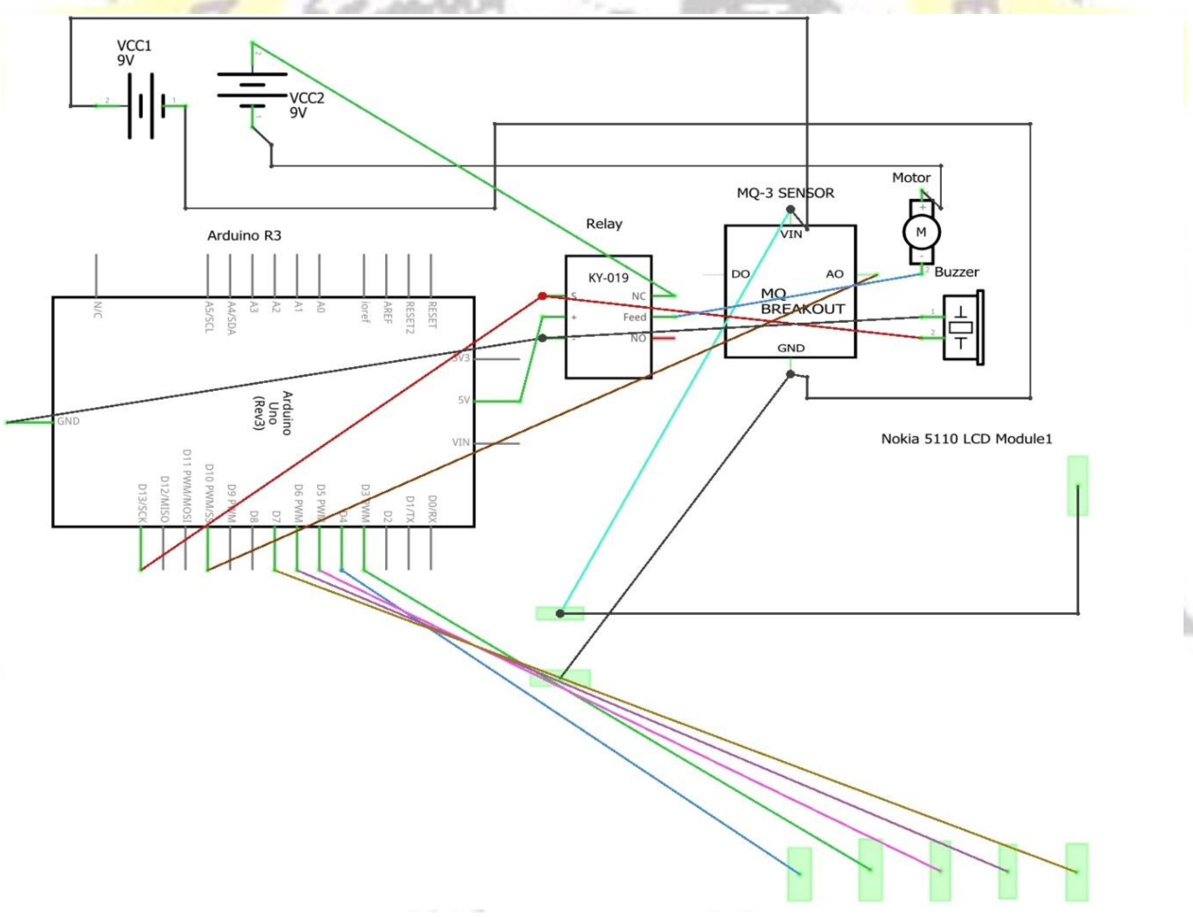
Applications:

- ➢ It is used to observe the Drowsy Drivers while they driving vehicles.
- ➢ It is used to communicate with physically handicap persons like Stiephen Hawkins.
- ➢ It is also used to detect the Coma Patients whenever they wake up in absence.

5. PROGRAM FOR ALCOHOL DETECTION WITH IGNITION LOCK SYSTEM:

```
//Alcohol Detector based on MQ3
/* Here are the list of I/O pins*/
#define MQ3 A0//the AOUT pin of the MO-3 sensor goes into analog pin Ao of the arduino
#define Relay 8//the DOUT pin of the relay goes into digital pin D8 of the arduino
#define Buzz 9//the DOUT pin of the buzzer goes into digital pin D9 of the arduino
//Threshold value of MQ3 reading above which it should trigger/
#define Thres_Val 300

//
// global int to store the analog reading from MQ3 (0 to 1023)
int value;
#include <SPI.h>
#include <Adafruit_GFX.h>
#include <Adafruit_PCD8544.h>

// Declare LCD object for software SPI
// Adafruit_PCD8544(CLK,DIN,D/C,CE,RST);
Adafruit_PCD8544 display = Adafruit_PCD8544(7, 6, 5, 4, 3);

int rotatetext = 1;/***/
void setup() {
  // declaring the input and output pins
  pinMode(MQ3, INPUT);;//sets the pin as an input to the arduino
  pinMode(Relay, OUTPUT);;//sets the pin as an output to the arduino
  pinMode(Buzz, OUTPUT);//sets the pin as an output to the arduino
  // Serial Output for debugging
  Serial.begin(9600);
  //Initialize Display
  display.begin();

  // you can change the contrast around to adapt the display for the best viewing
  display.setContrast(57);

  // Clear the buffer.

  display.clearDisplay();

  // Display Text
  display.setTextSize(2);
  display.setTextColor(BLACK);
  display.setCursor(0, 0);
  display.println("Alochol");
  display.display();
  delay(2000);
  display.clearDisplay();

}
void loop() {
  // reads the analog value from MQ3
  value = analogRead(MQ3);
  // sends the value to UART for debugging
  Serial.println(value);
  if ( value > Thres_Val )   //if alcohol is detected
  {
    digitalWrite ( Buzz , HIGH );    // turns the LED on
    digitalWrite(Relay, HIGH);   // turns on (uncomment if Relay is used)
    display.clearDisplay();
    display.println("limit is Exceed");// turns on lcd screen and indicate as limit is Excced
    display.display();
    delay(2000);
    display.clearDisplay();

    tone(Relay, 1000);            //Generate a 1000Hz tone only if you use speaker (comment out
if Relay is used)
  }
  else {
    digitalWrite(Buzz, LOW);      //  turns the LED off
    digitalWrite(Relay, LOW); //  turns off (uncomment if Relay is used)
    display.clearDisplay();
    display.println("limit  is not  Exceed");// turns on lcd screen and indicate as limit is not
Excced
    display.display();
    display.display();
    delay(2000);
    display.clearDisplay();
    noTone(Relay);            //  Removes the tone from speaker (comment out if Relay is used)
  }
  delay (500);           //  a 500ms delay before reading is taken again
}
```

5.1 CIRCUIT DIAGRAM FOR ALCOHOL DETECTION DEVICE:



5.2 Connections of Alcohol Detector

Step 1: - Connection of Nokia LCD 5110 to the Arduino Uno.

- RST pin is connected to the D3 pin of the Arduino board.
- CE pin is connected to the D4 pin of the Arduino board.
- DC pin is connected to the D5 pin of the Arduino board.
- DIN pin is connected to the D6 pin of the Arduino board.
- CLK pin is connected to the D7 pin of the Arduino board.
- VCC pin is connected to the 3.3V pin of the Arduino board.
- BL pin is connected to the 3.3V pin of the Arduino board.
- GND pin is connected to the GND pin of the Arduino board.

Step 2: - Connection of MQ3 Sensor to the Arduino Uno.

- A0 pin is connected to the A0 pin of the Arduino board.
- VCC pin is connected to the 5V pin of the Arduino board.
- GND pin is connected to the GND pin of the Arduino board.

Step 3: - Connection of Single-Phase Relay to the Arduino Uno.

- Signal pin is connected to the D8 pin of the Arduino board
- VCC pin is connected to the 5V pin of the Arduino board.
- GND pin is connected to the GND pin of the Arduino board.

Step 4: - Connection of DC Motor to Single-Phase Relay.

- Positive Terminal of the DC Motor is Connected to the Common Contact of the Single-Phase of the Relay.
- Negative Terminal of the DC Motor is Connected to the Negative Terminal of the 9V Battery.
- Positive Terminal of the Battery is Connected to the Normally open of the Single-Phase of the Relay.

Step 5: - Connection of Buzzer to the Arduino Uno.

- Positive Terminal of the Buzzer is Connected to the D9 pin of the Arduino board.
- Negative Terminal of the Buzzer is Connected to the GND pin of the Arduino board.

Step6: - Connection of Power Supply to the Arduino Uno.

- The both Positive and Negative terminal of the 9V Battery is Connected to the Battery Snap Connector & DC jack and it is Connected to the DC Power of the Arduino Board.

5.3 <u>Internal working:</u>

As the power supply gets switched ON every Single device used in the circuit gets power and they will turn ON. And The LCD will turn ON to show the indication of Dashboard, and it show the information like “Alcohol Detection” to show that device is turned ON.

Through Arduino NANO, the power will also be sent to MQ-3(alcohol sensor) with pin ‘A0’ as input. MQ-3 sensor is used as a “Breathalyzer”. Relay is connected to the pin ‘D8’, buzzer to pin ‘D9’ and finally the LCD is connected to pins ‘D5’ and ‘D6’.

In the device whenever it turns ON it Shows the name of circuit and the intimation named “Limit is not Exceed.” As the alcohol is detected in the range of MQ-3 sensor. There will be variation in the molecular Contamination, and it send the Signal to the Arduino board and LCD displays “Limit is Exceed.”

As the limit exceeds the relay diverts the power supply channel from ‘Normal Close’ to ‘Normal Open’ and the Buzzer starts beeping and the vibration starts vibrating and the DC Motor steps rotating as the indication of engine power loss.

If the vibration in environment comes down below the limit value the entire circuits goes reset and the device will get back to the initial state

5.4 Applications:

- It can be used in the various vehicles for detecting whether the driver has consumed alcohol or not.
- Breathing analyzer project can also be used in various companies or organizations to detect alcohol consumption of employees. An alcohol detection system in an automobile is a must feature which every cab or bus should have.

Advantages:

- It provides an automatic safety system for cars and other vehicles as well.

6. CONCLUSION

1) CONCLUDING THAT OUR PROJECT IS USEFUL FOR REAL TIME IMPLEMENTATION:

   This project is developed keeping in mind to help the society derivatives out of it. As the project has satisfied the requirement of the society, it is here by concluded that this project is always useful for realtime implementation.

2) DELIVERY OF THE STATEMENT OF ACCEPTING ALL THE SUGGETION GIVEN FOR THE PURPOSE OF IMPROVING THE PROPOSED SYSTEM:

   We the batchmates who developed this system for the public and the Government use, invite and accept all the suggetions given for the upgraded performance of the system development.

3) FINAL CONCLUSION BY THE BATCHMATES:

   With the intention of serving the society, all the sincere efforts will be put for the further development of system to achieve the results that are practically useful for day today life.

7. BIBLIOGRAPHY

- Arduino UNO board: - **https://www.elprocus.com/atmega328-arduino-uno-board-working-and-its-applications/**
  And **https://robu.in/arduino-pin-configuration/**
- Mq3 sensor: - **https://robu.in/arduino-pin-configuration/**
- Single phase 5v relay module: - **https://robu.in/arduino-pin-configuration/**
- Arduino Nano: - **https://robu.in/arduino-pin-configuration/**
- BC547 Transistor: - **https://robu.in/arduino-pin-configuration/**
- 4.7k resistor: - **https://robu.in/arduino-pin-configuration/**
- Nokia LCD 5110: - **https://lastminuteengineers.com/nokia-5110-lcd-arduino-tutorial/**
- 5v buzzer: - **https://www.elprocus.com/buzzer-working-applications/**
- Infrared Sensor Module: - **https://www.androiderode.com/ir-sensor-pinout-working-details/**
- Atmega328: - **http://www.learningaboutelectronics.com/Articles/Atmega328-pinout.php#:~:text=The%20Atmega328%20has%2028%20pins,the%20Atmega328%20is%20shown%20below**.
- Embedded System:- https://en.wikipedia.org/wiki/Embedded_system

- Arduino: - **https://en.wikipedia.org/wiki/Arduino** and https://www.arduino.cc/en/Guide/Introduction