VEHICLE THEFT DETECTION AND SECURE SYSTEM USING ARDUINO

A Project report submitted in partial fulfilment of the requirements for the

award of the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

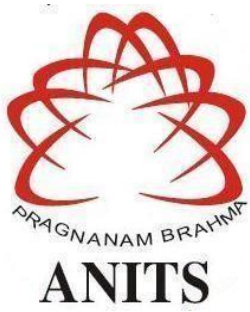
B.LOKESH (318126512011) **V.TIRUPATHIRAO(318126512056)**

R.SUPRA DEEPIKA (319126512L02) **P.JHANSI(318126512040)**

Under the guidance of

Mr.D. Anil Prasad

Assistant Professor



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES
(UGC AUTONOMOUS)

(Permanently Affiliated to AU, Approved by AICTE, and Accredited by NBA & NAAC with an 'A' Grade)
Sangivalasa, Bheemili Mandal, Visakhapatnam dist. (A.P)

(2021-2022)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES

(UGC AUTONOMOUS)

(Permanently Affiliated to AU, Approved by AICTE, and Accredited by NBA & NAAC with 'A-Grade)

Sangivalasa, Bheemili Mandal, Visakhapatnam dist. (A.P)

PRAGNANAM BRAHMA

ANITS

CERTIFICATE

This is to certify that the project report entitled "**VEHICLE THEFT DETECTION AND SECURE SYSTEM USING ARDUINO**" submitted by **B. LOKESH(318126512011),V. TIRUPATHI RAO(318126512056), R. SUPRA DEEPIKA(319126512L02), P. JHANSI(318126512040)** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Electronics & Communication Engineering** of Andhra University, Visakhapatnam is a record of bonafide work carried out under my guidance and supervision.

Project Guide

Mr. D. Anil Prasad
Assistant Professor
Department of E.C.E
ANITS

Assistant Professor
Department of E.C.E.
Anil Neerukonda
Institute of Technology & Sciences
Sangivalasa, Visakhapatnam-531 162

Head of Department

Dr. V. Rajyalakshmi
Professor & HOD
Department of E.C.E
ANITS

Head of the Department
Department of E C E
Anil Neerukonda Institute of Technology & Sciences
Sangivalasa-531 162

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our project guide **Mr. D. Anil Prasad,** Assistant professor, Department of Electronics and Communication Engineering, ANITS, for his/her guidance with unsurpassed knowledge and immense encouragement. We are grateful to **Dr. V. Rajyalakshmi**, Head of the Department of Electronics and Communication Engineering, for providing us with the required facilities for the completion of the project work.

We are very much thankful to the **Principal and Management, ANITS, Sangivalasa ,** for their encouragement and cooperation to carry out this work.

We express our thanks to all **teaching faculty** of the Department of ECE, whose suggestions during reviews helped us in I accomplishment of our project. We would like to thank teaching staff of the Department of ECE, ANITS for providing great assistance in accomplishing our project.

We would like to thank our parents, friends, and classmates for their encouragement throughout our project period. Last but not these, we thank everyone for supporting us directly or indirectly in completing this project successfully.

PROJECT STUDENTS

B. LOKESH (318126512011),
V.TIRUPATHI RAO (318126512056),
R.SUPRA DEEPIKA (319126512L02),
P. JHANSI (318126512040).

ABSTRACT

Vehicle theft is a common problem due to which people have lost their vehicles, faced many difficulties to find them, and sometimes failed even after a lot of struggles. To overcome this problem, we introduce the “VEHICLE DETECTION SYSTEM USING ARDUINO” with which we can find the vehicles which were either theft or met with an accident without any strenuous efforts. A using fingerprint sensor, RFID reader, and GSM GPS module it is made easy to identify and track the vehicle.

This project aims to authenticate and inform the details of the vehicle to the owner through SMS. If the owner is aware of that person he can give access through a return reply- motor on otherwise he can deny by replying –motor off. This is possible using GSM and fingerprint sensors. If the vehicle is theft by any chance, we can detect the vehicle using a reader where an RFID tag is placed in the device. On the other hand, the owner can also trace the location of the vehicle where he gets the information in terms of latitude and longitude using GPS. All these operations are controlled us using an ATMEGA328 microcontroller.

CONTENTS

ABSTRACT ii
LIST OF FIGURES v-vi
LIST OF TABLES **vii**
CHAPTER 1 INTRODUCTION **1-2**
1.1 Project Objective
1.2 Project Outline
CHAPTER 2 HARDWARE COMPONENTS
2.1 Arduino 3-7
2.1.1 Pin Diagram
2.1.2 Working
2.1.3 Applications
2.1.4 Features of the Arduino
2.2 GSM SIM900A 7-9
2.2.1 Pin Configuration
2.2.2 Working
2.2.3 Applications and Features
2.3 GPS NEO 6M 10-11
2.3.1 Pin Diagram
2.4 Node MCU 12-15
2.4.1 Pin Diagram and Functions
2.4.2 Applications
2.4.3 Features
2.5 Fingerprint Sensor 16-17
2.5.1 Features and Applications
2.6 RFID(MFRC522) 18-20
2.6.1 Pin Diagram
2.6.2 Features
2.7 LCD 21-23
2.7.1 LCD Registers
2.7.2 Pin Diagram
2.7.3 Features
2.8 MOTOR L289N 24
2.8.1 Features
CHAPTER3 INTERFACING
3.1 Interfacing Arduino with Motor 25-30
3.1.1 Controlling Dc motor
3.1.2 L298n motor driver Ic
3.1.3 Control Pins of Motors

3.1.4 Directional control pins
3.1.5 Speed control pins
3.1.6 Wiring L289N motor driver with Arduino
3.2 Interfacing Fingerprint sensor with Arduino 31-35
3.2.1 Installing the Adafruit Fingerprint Sensor Library
3.2.2 Finding a Match
3.3 Interfacing SIM900A GSM with Arduino 35-37
3.4 Interfacing RFID with Node MCU 38
3.5 Interfacing LCD with Node MCU 39-40
3.6 Interfacing GPS with Arduino 40-41

CHAPTER4 ARDUINO PROGRAMMING

4.1 Arduino Pin Diagram 42
4.2 Programming
4.2.1 Basics 43-44
4.2.2 Syntax and Program Flow 45-46
4.2.3 Serial Functions 45
4.2.4 AnalogRead () 48
4.2.5 Arduino Data types 48-49
4.2.6 Arduino Variables 49
4.2.7 Arduino Operators 49-53
4.2.8 Arduino IF Statements 53-54

CHAPTER5 RESULTS AND DISCUSSION

5.1 Arduino Simulator 54-58
5.1.1 Advantages of using Simulators
5.1.2 Types of Simulators
5.1.3 Accessing Simulators
5.1.4 Features of Simulators
5.2 Simulation Results 59-61
5.2.1 Motor
5.2.1.1 connections of motor
5.2.2 LCD
5.2.2.1 connections of LCD
5.2.2.2 Results of LCD
5.3Hardware implementation 61-62
5.4Discussions 63

CHAPTER 6 CONCLUSIONS 64
Reference 65-66

LIST OF FIGURES

Figure No	Name of the Figure	Page No
Fig 2.1	Arduino UNO Board	03
Fig 2.2	Arduino Pin Diagram	04
Fig 2.3	GSM SIM900A	08
Fig 2.4	GPS NEO 6M	08
Fig 2.5	SIM 900A Configuration	10
Fig 2.6	GPS Pin Configuration	11
Fig 2.7	Node MCU Pin Configuration	13
Fig 2.8	Fingerprint Sensor	16
Fig 2.9	RFID Reader and tag	18
Fig 2.10	RFID Pin Diagram	19
Fig 2.11	LCD Pin Diagram	22
Fig 2.12	Motor L289	24
Fig 3.1	Motor L289 H-Bridge Module	26
Fig 3.2	Motor L289 output pins	27
Fig 3.3	Directional Control Pins	28
Fig 3.4	Speed Control Pins	29
Fig 3.5	Interfacing Motor with Arduino	30
Fig 3.6	Fingerprint sensor pin diagram	31
Fig 3.7	Fingerprint sensor connections	31
fig 3.8	Fingerprint enrollment	33
Fig 3.9	scanning finger using sensor	33
Fig 3.10	output of fingerprint sensor when finger placed	34
Fig 3.11	serial monitor	34

Fig 3.12	output of fingerprint sensor in serial monitor when matched	35
Fig 3.13	Interfacing GSM with Arduino (without library)	37
Fig 3.14	Interfacing RFID with Node MCU	38
Fig 3.15	Interfacing LCD with Node MCU	39
Fig 3.16	GPS NEO 6M	38
Fig 3.17	Interfacing GPS with Arduino	41
Fig 4.1	Arduino Pin Diagram	42
Fig 4.2	coding screen	43
Fig 4.3	Program Flow chart	46
Fig 4.4	Flow Chart of IF Statements	53
Fig 4.5	Flow Chart of IF- ELSE Statements	54
Fig 5.1	Auto Desk Tinker CAD Window	56
Fig 5.2	Menu Window	56
Fig 5.3	Sign in Window	57
Fig 5.4	Join Window	57
Fig 5.5	Tinker CAD Window	58
Fig 5.6	Interfacing of motor with Arduino using motor driver	59
Fig 5.7	Interfacing LCD with Arduino	60
Fig 5.8	Output of LCD	61
Fig 5.9	Hardware prototype	62

LIST OF TABLES

Table No	Table Description	Page No
Table 2.1	pin description	05
Table 3.1	Interfacing Fingerprint sensor with Arduino	32
Table 3.2	Connections LCD with Node MCU	39
Table 3.3	Connections of GPS with Arduino	41

CHAPTER 1

INTRODUCTION

Nowadays, security has become one of the critical issues in the present world. People always want their things to be secured at any cost. People work a lot in their daily life, and most of them work in different places. We have many tools and software designed to ensure the safety of our belongings. The security issue has become one of the alarming problems in society. The researchers and vehicle manufacturers throughout the world have been working on various theories to come up with a device that helps to curb this menace. The Bluetooth-enabled cost-effective solution has been made to protect the vehicles which uses arduino, fingerprint sensor and Bluetooth.[1] When a vehicle is stolen, it becomes hard to locate and track it, which considerably decreases the chances of recovering it. An Anti-Theft vehicle security has been developed to mitigate this problem.[2] One of the methods which can be applied for a security system is based on biometric identification system. Fingerprint recognition is one of the biometric systems that can be applied to the security system. The fingerprint of the owner and other authorized persons will be stored into the database, then while the time of starting the engine of the vehicle, the fingerprint will be validated with the database.[3] A fingerprint cum GPS/GSM tracking prototype is reported in 2019 where the location of the vehicle, i.e., latitude and longitude of the vehicle, and the tracing of the vehicle is done.[4] RFID based security and access control system is more secure and responds fast as compared to the other systems like biometric system.[5] The advantage of the RFID system is it works without-line-of-sight and is contact less. By using Arduino access is easy and works can be performed very quickly. Users can change the function as per convenience by using Arduino. Many researchers are working relentlessly to make solutions which can solve the various security problem of the community. Vehicles are ubiquitous in everyone's life. So we need elements which can ensure the safety of the vehicles. Our latest technologies bring a lot of hardware and software tools that makes the security of the vehicles. Hence providing cost-effective solution to protect the vehicles which uses Arduino, fingerprint sensor, GPS/GSM and RFID is of primary concern.

1.1 PROJECT OBJECTIVE:

This project aims to authenticate and inform the details of the vehicle to the owner through SMS. If the owner is aware of that person he can give access through a return reply- motor on otherwise he can deny by replying –motor off. This is possible using GSM and fingerprint sensors. If the vehicle is theft by any chance, we can detect the vehicle using a reader where an RFID tag is placed in the device. On the other hand, the owner can also trace the location of the vehicle where he gets the information in terms of latitude and longitude using GPS. All these operations are controlled us using an ATMEGA328 microcontroller.

1.2 PROJECT OUTLINE:

This project report is presented over the four remaining chapters. Chapter 2 describes the Hardware Components used in the project. Chapter 3 describes the Interfacing of Hardware components used in the project. Chapter 4 explains the concepts of Arduino programming with different Components. Chapter 5 presents the simulation results of the Components using the Arduino. Finally, conclusions are drawn in chapter 6.

CHAPTER 2

Hardware Components

2.1 **ARDUINO**:

Arduino Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 Analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with n h AC-to-DC adapter or battery to get started. The unit comes with 32KB flash memory that is used to store the number of instructions while the SRAM is 2KB and EEPROM is 1KB. The operating voltage of the unit is 5V which projects the microcontroller on the board and its associated circuitry operates at 5V while the input voltage ranges between 6V to 20V and the recommended input voltage ranges from 7V to 12V.

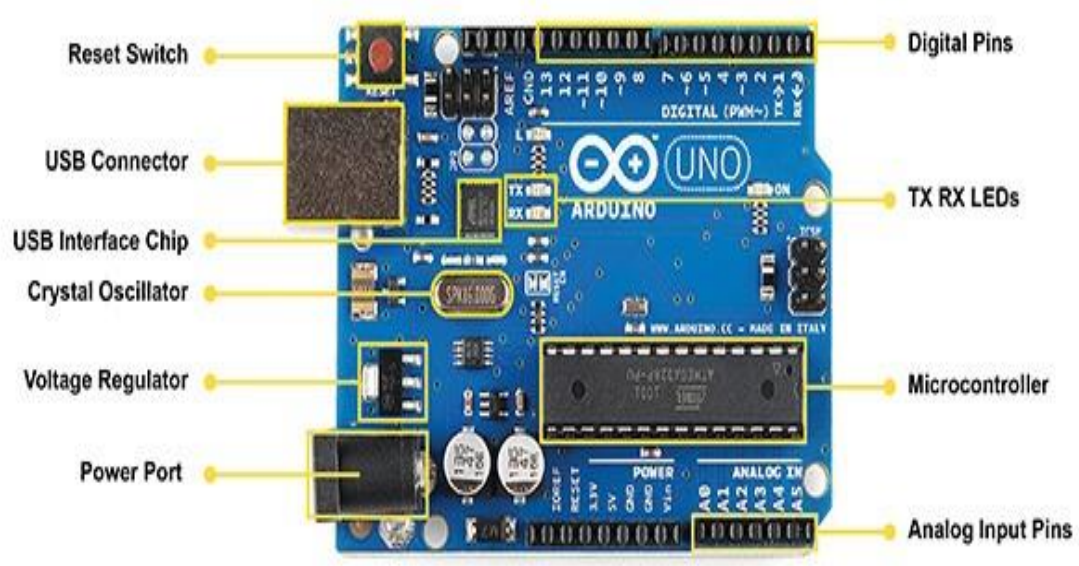


Fig 2.1: ARDUINO UNO BOARD

2.1.1 Pin Diagram:

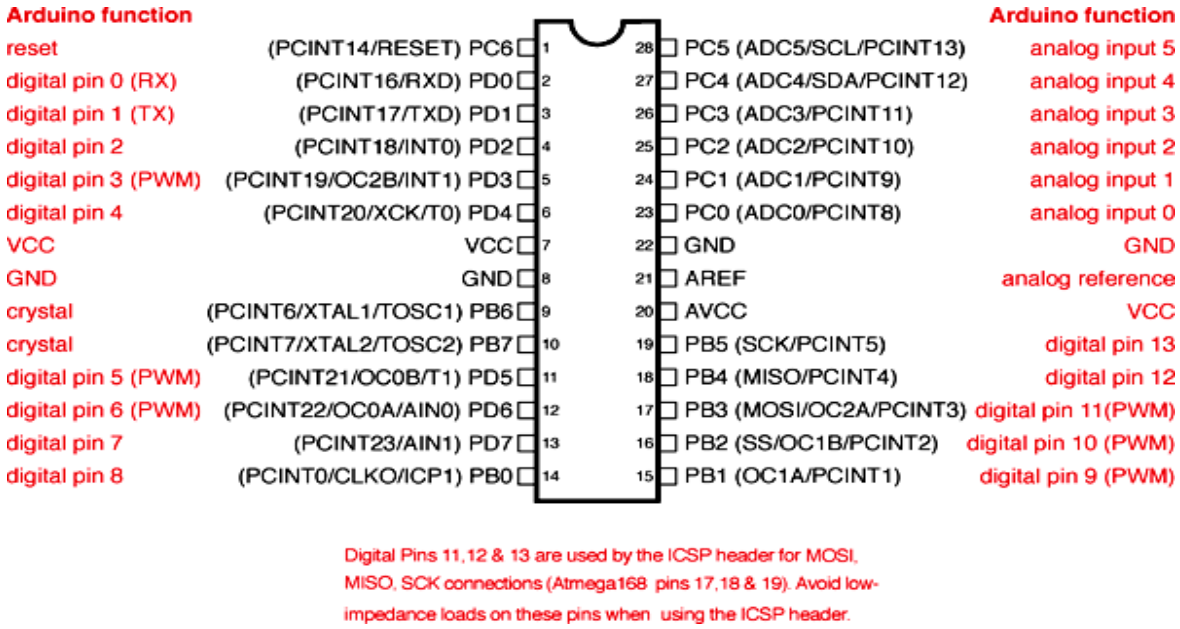


Fig 2.2: Arduino Pin diagram

Table 2.1: PIN CONFIGURATION

Pin Category	**Pin Name**	**Details**
Power	Vin, 3.3V, 5V, GND	Vin: Input voltage to Arduino when using an external source. 5V: Regulated power supply used to power microcontrol other components on the board. 3.3V: 3.3V supply generated by the onboard voltage reg The maximum current draw is 50mA. GND: ground pins.
Reset	Reset	Resets the microcontroller.
Analog Pins	A0 – A5	Used to provide analog input in the range of 0-5V

Input/Output Pins	Digital Pins 0 - 13	Can be used as input or output pins.
Serial	0(Rx), 1(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.
TWI	A4 (SDA), A5 (SCA)	Used for TWI communication.
AREF	AREF	To provide a reference voltage for input voltage.

2.1.2 Description of Arduino Uno Board:

Arduino is a single-board microcontroller meant to make the application more accessible which are interactive objects and their surroundings. The hardware features an open-source hardware board designed around an 8-bit Atmel AVR microcontroller . Current models consist of a USB interface, 6 analog input pins, and 14 digital I/O pins that allow the user to attach various extension boards.

The Arduino Uno board is a microcontroller based on the ATmega328. It has 14 digital input/output pins of which 6 can be used as PWM outputs, a 16 MHz ceramic resonator, an ICSP header, a USB connection, 6 analog inputs, a power jack, and a reset button. This contains all the required support needed for the microcontroller. To get started, they are simply connected to a computer with a USB cable. Arduino Uno Board varies from all other boards and they will not use the FTDI USB-to-serial driver chip in them. It is featured by the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

Arduino Uno R3 version is used in this project. Arduino projects can be stand-alone or they can communicate with software running on a computer. The board is clocked by a 16 MHz ceramic resonator and has a USB connection for power and communication. You can easily add micro SD/SD card storage for bigger tasks

2.1.3Applications:

- Prototyping of Electronics Products and Systems
- Projects requiring Multiple I/O interfaces and communications.

2.1.4Features of the Arduino UNO:

- Microcontroller: ATmega328.
- Operating Voltage: 5V.
- Input Voltage (recommended): 7-12V.
- Input Voltage (limits): 6-20V.
- Digital I/O Pins: 14 (of which 6 provide PWM output)
- Analog Input Pins: 6.
- DC Current per I/O Pin: 40 mA.
- DC Current for 3.3V Pin: 50 mA.
- Flash Memory: 32 KB of which 0.5 KB is used the by the bootloader
- SRAM: 2 KB (ATmega328)
- EEPROM: 1 KB (ATmega328)

2.2 GSM:

SIM900A GSM Module is the smallest and cheapest GPRS/GSM communication module. It is common with Arduino and microcontrollers in most. The module offers GPRS/GSM technology for communication with the use of a mobile. It uses a 900 and 1800MHz frequency band and allows users to receive/send mobile calls and SMS. The keypad and display interface allows the developers to make the customize application with it. It also has modes, command mode and data mode. SIM900A GSM is shown in the below figure.

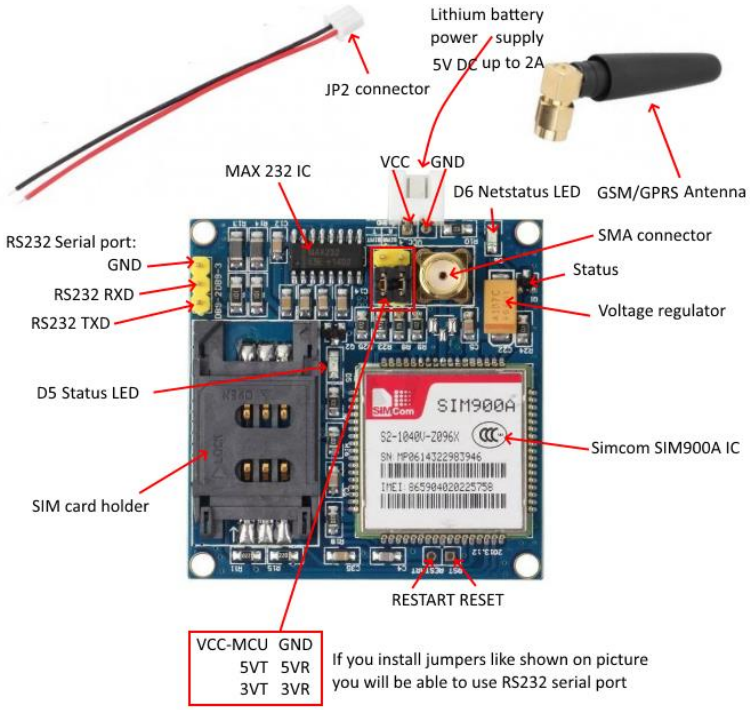


Fig 2.3: GSM SIM900A

2.2.1SIM900A Pin Configuration:

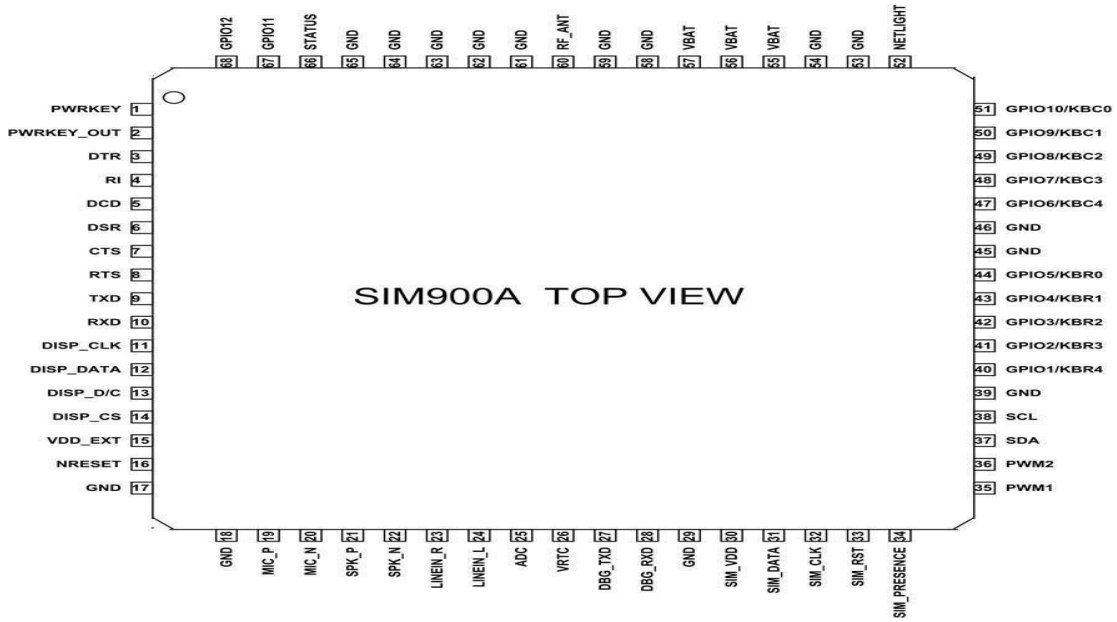


Fig 2.4: SIM900A Pin Configuration

2.2.1 Working:

SIM900A GSM Module is the smallest and cheapest GPRS/GSM communication module. It is common with Arduino and microcontrollers in a most embedded application. The module offers GPRS/GSM technology for communication with the use of a mobile sim. It uses a 900 and 1800MHz frequency band and allows users to receive/send mobile calls and SMS. The keypad and display interface allows the developers to make the customize application with it. Furthermore, it also has modes, command mode and data mode. In every country, they are GPRS/GSM and different protocols/frequencies to operate. Command mode helps the developers get the default setting according to their requirements. The baud rate is configurable from 1200-115200 through AT command. The GSM/GPRS Modem is having internal TCP/IP stack to enable you to connect to the internet via GPRS.. SIM900A is an ultra-compact and reliable wireless module. This is a complete GSM/GPRS module in an SMT type and designed with a very powerful single-chip processor integrating AMR926EJScore.

2.2.2 Applications:

- The module is best application to design a graphic for Voice calls and SMS applications.
- Some IoT applications, mostly in an emergency have the module.
- The location trancing system also uses SIM900A
- SIM900A can use for mobile communication.

2.2.3 SIM900A GSM module Features:

- The power supply of voltage is 3.4v-4.5v
- In sleep mode, current consumption is about 1.5mA
- Dual-frequency EGSM900 and DCS1800
- Keypad and display interface
- MIC and audio output
- UART interface
- AT command communication
- Data Transfer: 85.6Kbps (Download)/42.8Kbps (upload)

2.3 GPS NEO 6M:

At the heart of the module is a NEO-6M GPS chip from u-blox. The chip measures less than the size of a postage stamp but packs a surprising amount of features into its little frame. It can track up to 2 satellites on 50 channels and achieves the industry's highest level of sensitivity i.e. -161 dB tracking, while consuming only 45mA supply current. Unlike other GPS modules, it can do up to 5 location updates a second with 2.5m Horizontal position accuracy. The u-blox 6 positioning engine also boasts a Time-To-First-Fix (TTFF) of under 1 second .One of the best features the chip provides is Power Save Mode(PSM). It allows a reduction in system power consumption by selectively switching parts of the receiver ON and OFF. This dramatically reduces power consumption of the module to just 11mA making it suitable for power sensitive applications like GPS wristwatch. The necessary data pins of NEO-6M GPS

Fig 2.5: GPS NEO

chip are broken out to a 0.1″ pitch headers. This includes pins required for communication with a microcontroller over UART. The module supports baud rate from 4800bps to 230400bps with default baud of 9600.

2.3.1 NEO-6M GPS Module Pin Configuration:

The NEO-6M GPS module has a total of 4 pins that interface it to the outside world. The connections are as follows.

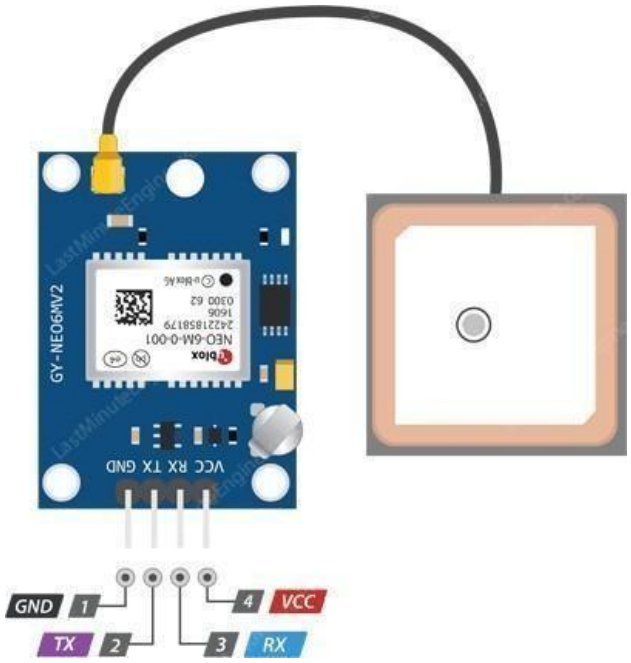


Fig 2.6: PIN CONFIGURATION OF GPS

GND is the Ground Pin and needs to be connected to the GND pin on the Arduino. TxD(Transmitter) pin is used for serial communication. RxD(Receiver) pin is used for serial communication.VCC supplies power for the module. You can directly connect it to the 5V pin on the Arduino.

2.3.2 Features:

- 5Hz position update rate
- EEPROM to save configuration settings
- Rechargeable battery for Backup
- The cold start time of 38 s and Hot start time of 1 s
- Supply voltage: 3.3 V
- Configurable from 4800 Baud to 115200 Baud rates. (default 9600)
- Super Sense ® Indoor GPS: -162 dBm tracking sensitivity

2.4 NODE MCU (ESP8366EX):

The Node MCU (Node Microcontroller Unit) is an open-source software and hardware development environment built around an inexpensive System-on-a-Chip (SoC) called the ESP8266. The ESP8266, designed and manufactured by Espressif Systems, contains the crucial elements of a computer: CPU, RAM, networking (WIFI), and even a modern operating system and SDK. That makes it an excellent choice for Internet of Things (IoT) projects of all kinds. However, as a chip, the ESP8266 is also hard to access and use. You must solder wires, with the appropriate analog voltage, to its pins for the simplest tasks such as powering it on or sending a keystroke to the "computer" on the chip. You also have to program it in low-level machine instructions that can be interpreted by the chip hardware. This level of integration is not a problem using the ESP8266 as an embedded controller chip in mass-produced electronics. It is a huge burden for hobbyists, hackers, or students who want to experiment with it in their own IoT projects. The Arduino project created an open-source hardware design and software SDK for their versatile IoT controller. Similar to Node MCU, the Arduino hardware is a microcontroller board with a USB connector, LED lights, and standard data pins. It also defines standard interfaces to interact with sensors or other boards. But unlike Node MCU, the Arduino board can have different types of CPU chips (typically an ARM or Intel x86 chip) with memory chips, and a variety of programming environments. There is an Arduino reference design for the ESP8266 chip as well. However, the flexibility of Arduino also means significant variations across different vendors. For example, most Arduino boards do not have WiFi capabilities, andsome even have a serial data port instead of a USB port.

2.4.1 NodeMCU Pinout and Functions Explained:

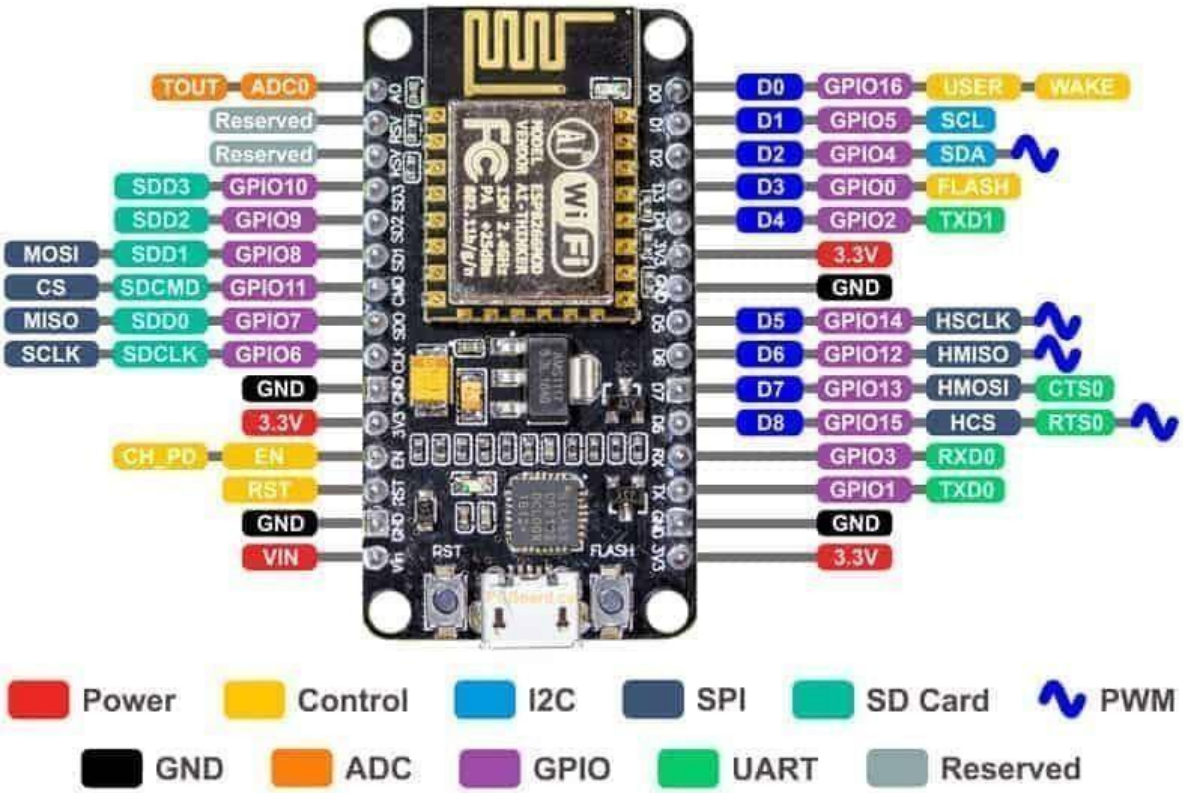


Fig 2.7: Node MCU pinout

- Power Pins There are four power pins. **VIN** pin and three **3.3V** pins.
- **VIN** can be used to directly supply the NodeMCU/ESP8266 and its peripherals. Power delivered on **VIN** is regulated through the onboard regulator on the NodeMCU module – we can also supply 5V regulated to the **VIN** pin
- **3.3V** pins are the output of the onboard voltage regulator and can be used to supply power to external components.

- GND is the ground pins of NodeMCU/ESP8266.
- I2C Pins are used to connecting I2C sensors and peripherals. Both I2C Master and I2C Slave are supported. I2C interface functionality can be realized programmatically, and the clock frequency is 100 kHz at a maximum. It should be noted that the I2C clock frequency should be higher than the slowest clock frequency of the slave device.

- GPIO Pins NodeMCU/ESP8266 has 17 GPIO pins that can be assigned to functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light, and Button programmatically. Each digital-enabled GPIO can be configured to internal pull-up or pull-down, or set to high impedance. When configured as an input, it can also be set to edge-trigger or level-trigger to generate CPU interrupts.
- ADC Channel the NodeMCU is embedded with a 10-bit precision SAR ADC. The two functions can be implemented using ADC. Testing power supply voltage of VDD3P3 pin and testing input voltage of TOUT pin. However, they cannot be implemented at the same time.
- UART Pins NodeMCU/ESP8266 has 2 UART interfaces (UART0 and UART1) which provide asynchronous communication (RS232 and RS485) and can communicate at up to 4.5 Mbps. UART0 (TXD0, RXD0, RST0 & CTS0 pins) can be used for communication. However, UART1 (TXD1 pin) features only data transmit signal so, it is usually used for printing logs
- SPI Pins NodeMCU/ESP8266 features two SPIs (SPI and HSPI) in slave and master modes. These SPIs also support the following general-purpose SPI features:
- 4 timing modes of the SPI format transfer
- Up to 80 MHz and the divided clocks of 80 MHz
- Up to 64-Byte FIFO
- SDIO Pins NodeMCU/ESP8266 features Secure Digital Input/Output Interface (SDIO) which is used to directly interface SD cards. 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported.
- PWM Pins The board has 4 channels of Pulse Width Modulation (PWM). The PWM output can be implemented programmatically and used for driving digital motors and LEDs. PWM frequency range is adjustable from 1000 μs to 10000 μs(100Hz and 1 kHz).
- Control Pins are used to controlling the NodeMCU/ESP8266. These pins include the Chip Enable pin (EN), Reset pin (, RST), and WAKE pin.
- **EN:** The ESP8266 chip is enabled when the EN pin is pulled HIGH. When pulled LOW the chip works at minimum power.
- **RST:** RST pin is used to reset the ESP8266 chip.

2.4.2 Node MCU 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

2.4.3 Applications:

- Prototyping of IoT devices
- Low power battery operated applications
- Network projects
- Projects requiring multiple I/O interfaces with Wi-Fi and Bluetooth functionalities

2.5 FINGER PRINT SENSOR(RJ307):

This is the R307 Optical Fingerprint Reader Sensor Module. R307 fingerprint module is a fingerprint sensor with a TTL UART interface for direct connections to microcontroller UART or PC through MAX232 / USB-Serial adapter. The user can store the fingerprint data in the module and can configure it in 1:1 or 1: N mode for identifying the person.

The FP module can directly interface with a 3.3 or 5v Microcontroller. A level converter (like MAX232) is required for interfacing with the PC serial port.

Integrated image collecting and algorithm chip together, the All-in-one Fingerprint reader can conduct secondary development and can be embedded into a variety of end products. Users can conduct secondary development, which can be embedded into a variety of end products, such as access control, attendance, safety deposit box, and car door locks.

Low power consumption, low cost, small size, excellent performance, Professional optical technology, precise module manufacturing technics. Good image processing capabilities can successfully capture an image up to a resolutionof n 500 dpi Finger detection function.

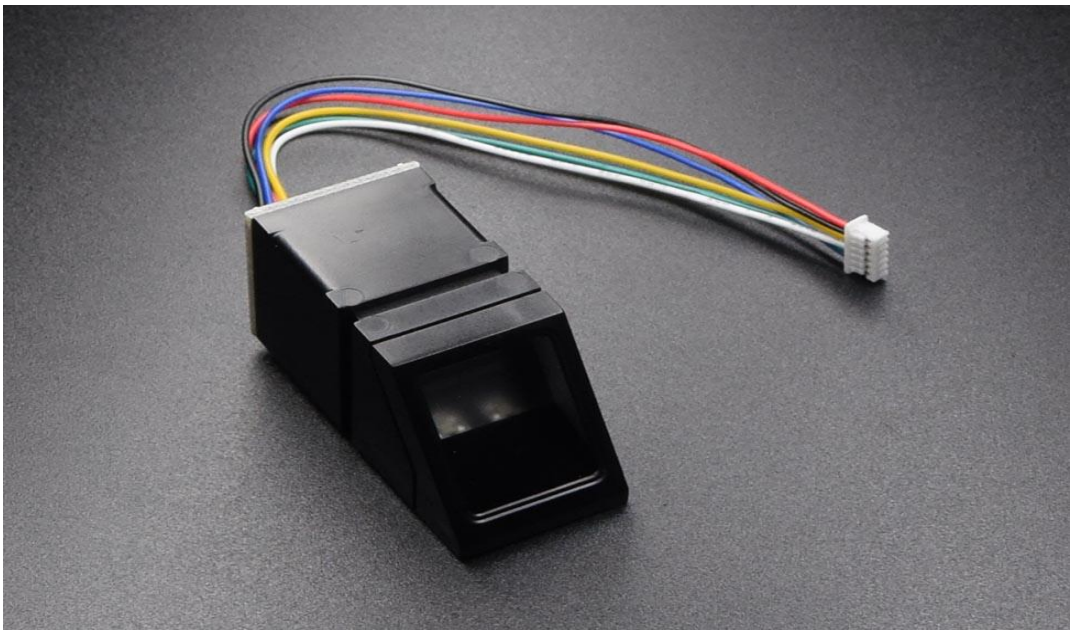


Fig 2.8: Finger Print Sensor

2.5.1 Features:

- Integrated image collecting and algorithm chip together, ALL-in-One.
- The fingerprint reader can conduct secondary development and can be embedded into a variety of end products.
- Low power consumption, low cost, small size, excellent performance.
- Professional optical technology, precise module manufacturing techniques.
- Good image processing capabilities can successfully capture an image up to a resolution of n 500 dpi.

2.5.2 Specification of RJ (307):

- The power supply is 6V.
- The current value is 50mA.
- The interface can be UART and USB
- Baud rate is 9600*N bps, N=1 to 12 and image can be taken in 0.5s.
- The character file size is 256 Bytes
- The template size is 512 Byte.
- The false acceptance rate is 0.001%
- The false recognition rate is 0.1%

2.6 RFID(MFRC522):

RFID or Radio Frequency Identification system consists of two main components, a transponder/tag attached to an object to be identified, and a Transceiver also known as interrogator/Reader. A Reader consists of a Radio Frequency module and an antenna that generates a high-frequency electromagnetic field. On the other hand, the tag is usually a passive device, meaning it doesn't contain a battery instead e, ad it contains a microchip that stores and processes information, and an antenna to receive and transmit a signal.

To read the information encoded on a tag, it is placed near the Reader (does not need to be within a direct line of sight of the reader). A Reader generates an electromagnetic field that causes electrons to move through the tag's antenna and subsequently power the chip. The powered chip inside the tag then responds by sending its stored information back to the reader in the form of another radio signal. This is called backscatter. The backscatter, or change in the electromagnetic/RF wave, is detected and interpreted by the reader which then sends the data out to a computer or microcontroller.

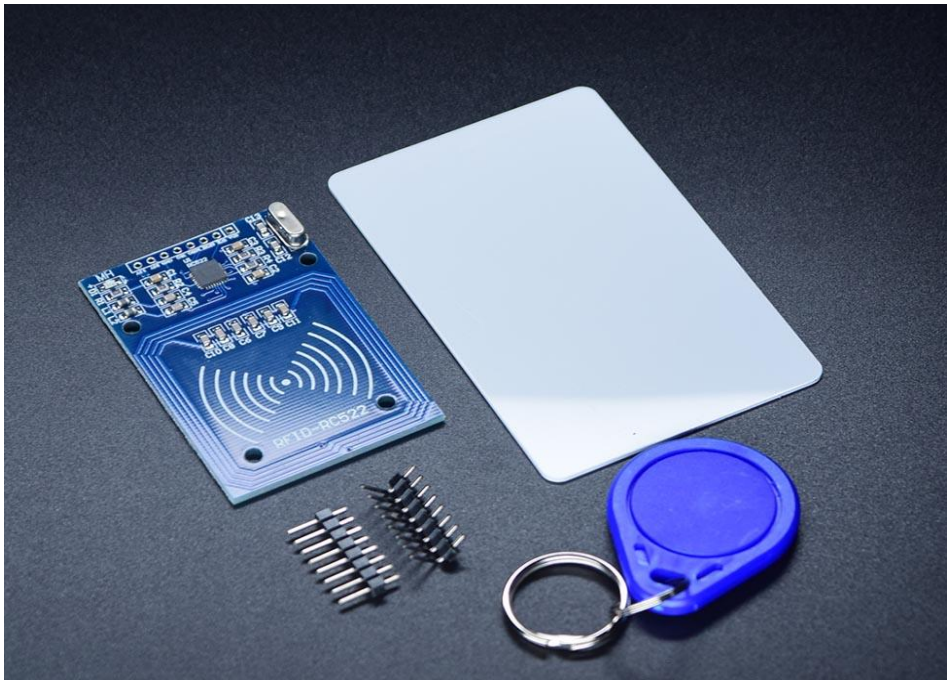


Fig 2.9: RFID Reader and Tag

The RC522 RFID Reader module is designed to create a 13.56MHz electromagnetic field that it uses to communicate with the RFID tags (ISO 14443A standard tags). The reader can communicate with a microcontroller over a 4-pin Serial Peripheral Interface (SPI) with a maximum data rate of 10Mbps. It also supports communication over I2C and UART protocols. The operating voltage of the module is from 2.5 to 3.3V, but the good news is that the logic pins are 5-volt tolerant, so we can easily connect it to an Arduino or any 5V logic microcontroller without using any logic level converter.

2.6.1 RC522 RFID Module Pin Configuration:

The RC522 module has a total of 8 pins that interface it to the outside world. The connections are as follows:

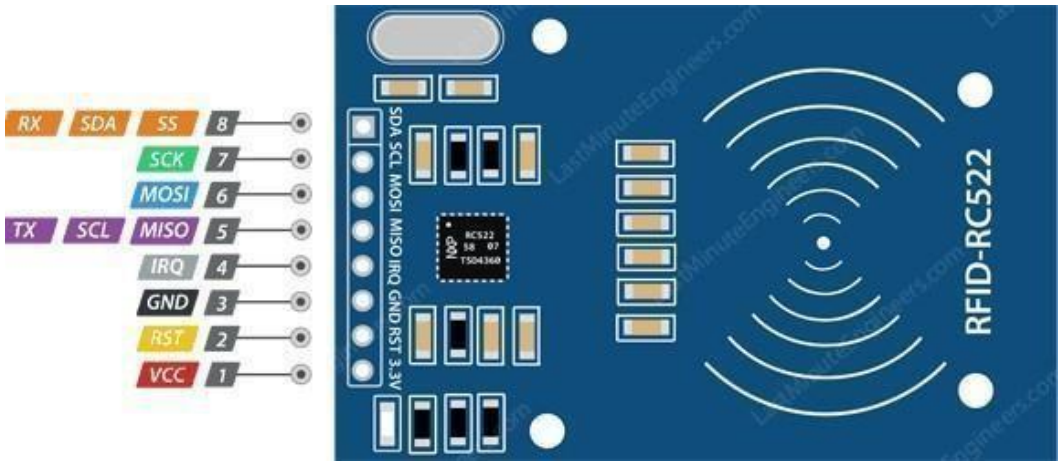


Fig 2.10: RFID Pin Diagram

- VCC supplies power for the module. This can be anywhere from 2.5 to 3.3 volts. You can connect it to 3.3V output from your Arduino. Remember connecting it to a 5V pin will likely destroy your module!
- RST is an input for Reset and power-down. When this pin goes low, a hard power-down is enabled. This turns off all internal current sinks including the oscillator and the input pins are disconnected from the outside world. On the rising edge, the module is reset.
- GND is the Ground Pin and needs to be connected to the GND pin on the Arduino.
- IRQ is an interrupt pin that can alert the microcontroller when an RFID tag comes into its vicinity.
- MISO/SCL/Tx pin acts as Master-In-Slave-Out when SPI interface is enabled, acts as a serial clock when I2C interface is enabled, ed and acts as serial data output when UART interface is enabled.

- MOSI(Master Out Slave In) is SPI input to the RC522 module.
- SCK(serial clock) accepts clock pulses provided by the SPI bus Master i.e. Arduino.
- SS/SDA/Rx pin acts as Signal input when the SPI interface is enabled, acts as serial data the hen I2C interface is enabled and acts as serial data input when the UART interface is enabled. This pin is usually marked by encasing the pin in a square so it can be used as a reference for identifying the other pins.

2.6.2 Features:

- 13.56MHz RFID module
- Operating voltage: 2.5V to 3.3V
- Communication: SPI, I2C protocol, UART
- Maximum Data Rate: 10Mbps
- Read Range: 5cm
- Current Consumption: 13-26mA
- Power-down mode consumption: 10uA (min)

2.7 LCD:

The term LCD stands for liquid crystal display. It is one kind of electronic display module used in an extensive range of applications like various circuits & devices like mobile phones, calculators, computers, TV sets, etc. These displays are mainly preferred for multi-segment light-emitting diodes and seven segments. The main benefits of using this module are inexpensive; simply programmable, animations, and there are no limitations for displaying custom characters, special and even animations, etc.

2.7.1 Registers of LCD:

A 16×2 LCD has two registers a data register and a command register. The RS (register select) is mainly used to change from one register to another. When the register set is '0', then it is known as the command register. Similarly, when the register set is '1', then it is known as the data register.

Command Register:

The main function of the command register is to store the instructions of commands which are given to the display. So that predefined tasks can be performed such as clearing the display, initializing, setting the cursor place, and display control. Here commands processing can occur within the register.

Data Register:

The main function of the data register is to store the information which is to be exhibited on the LCD screen. Here, the ASCII value of the character is the information which is to be exhibited on the screen of LCD. Whenever we send the information to LCD, it transmits to the data register, and then the process will be starting there. When register set =1, then the data register will be selected.

2.7.2 LCD 16×2 Pin Diagram:

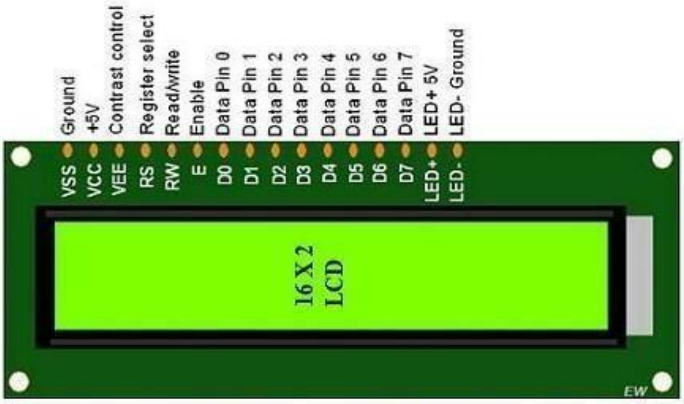


Fig 2.11: LCD-16×2-pin-diagram

- Pin1 (Ground/Source Pin): This is a GND pin of display, used to connect the GND terminal ofthe microcontroller unit or power source.
- Pin2 (VCC/Source Pin): This is the voltage supply pin of the display, used to connect the supplypin of the power source.
- Pin3 (V0/VEE/Control Pin): This pin regulates the difference of the display, used to connect achangeable POT that can supply 0 to 5V.
- Pin4 (Register Select/Control Pin): This pin toggles among command or data registers, used to connect a microcontroller unit pin , and obtains either 0 or 1(0 = data mode, and 1 = command mode).
- Pin5 (Read/Write/Control Pin): This pin toggles the display among the read or writes operation, and it is connected to a microcontroller unit pin to get either 0 or 1 (0 = Write Operation, and 1=Read Operation).

- Pin 6 (Enable/Control Pin): This pin should be held high to execute the Read/Write process, andit is connected to the microcontroller unit & constantly held high.

- Pins 7-14 (Data Pins): These pins are used to send data to the display. These pins are connected in two-wire modes like 4-wire mode and 8-wire mode. In 4-wire mode, only four pins are connected to the microcontroller unit like 0 to 3, whereas in 8-wire mode, 8-pins are connected to the microcontroller unit like 0 to 7.Pin15 (+ve pin of the LED): This pin is connected to +5V.Pin 16 (-ve pin of the LED): This pin is connected to GND

2.7.3 Features of LCD16x2:

- The operating voltage of this LCD is 4.7V-5.3V
- It includes two rows where each row can produce 16-characters.
- The utilization of current is 1mA with no backlight
- Every character can be built with a 5×8pixel box
- The alphanumeric LCDs alphabets & numbers
- Is display can work on two modes like 4-bit & 8-bit
- These are obtainable in Blue & Green Backlight
- It displays a few custom generated characters

2.8 MOTORL298N:

This L298N Motor Driver Module is a high-power motor driver module for driving DC and Stepper Motors. This module consists of an L298 motor driver IC and a 78M05 5V regulator.

The L298N motor driver is based on the H-bridge configuration (an H-bridge is a simple circuit that lets us control a DC motor to go backward or forward.), which is useful in controlling the direction of rotation of a DC motor. L298N Module can control up to 4 DC motors, or 2 DC motors with directional and speed control. It is a high current dual full H-bridge driver that is constructed to receive standard TTL logic levels. It can also be used to control inductive loads

e.g. relays, solenoids, motors (DC and stepping motor), etc.

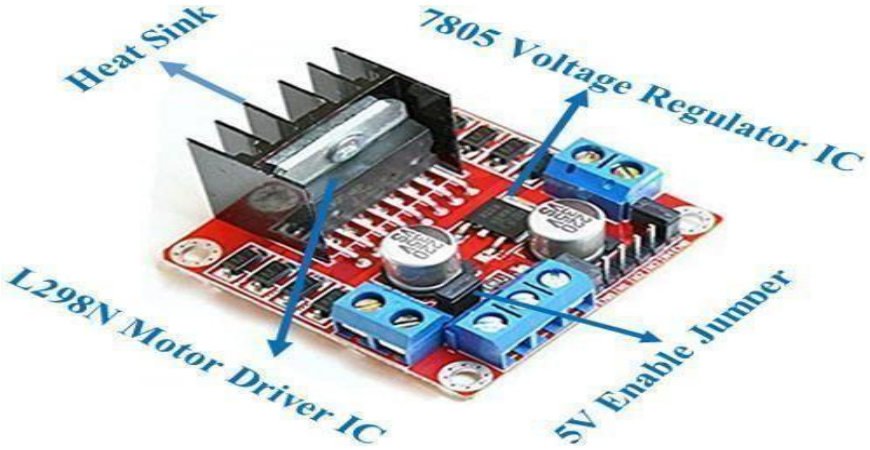


Fig 2.12: MotorL289 Diagram

2.7.4 Features & Specifications:

- Driver Model: L298N 2A
- Driver Chip: Double H Bridge L298N
- Motor Supply Voltage (Maximum): 46V
- Motor Supply Current (Maximum): 2A
- Logic Voltage: 5V
- Driver Voltage: 5-35V
- Driver Current:2A
- Logical Current:0-36mA
- Maximum Power (W): 25W

CHAPTER 3

INTERFACING

3.1 INTERFACING ARDUINO WITH MOTOR:

One of the easiest and most inexpensive ways to control DC motors is to interface L298N Motor Driver with Arduino. It can control both the speed and spinning direction of two DC motors.

3.1.1 Controlling a DC Motor:

To have complete control over the DC motor here we have to control its speed and rotation direction. This can be achieved by combining these two techniques.

- PWM – For controlling speed
- H-Bridge – For controlling rotation direction

PWM – For controlling speed:

The speed of a DC motor can be controlled by varying its input voltage. A common technique for doing this is to use PWM (Pulse Width Modulation). PWM is a technique where the average value of the input voltage is adjusted by sending a series of ON-OFF pulses. The average voltage is proportional to the width of the pulses known then as the Duty Cycle. The higher the duty cycle, the greater the average voltage being applied to the dc motor (High rede, ed), and the lower the duty cycle, the less the average voltage being applied to the dc motor (Low Speed).

H-Bridge – For controlling rotation direction:

The DC motor's spinning direction can be controlled by changing the polarity of its input voltage A common technique for doing this is to use an H-Bridge. An H-Bridge circuit contains four switches with the motor at the center forming an H-like arrangement. Closing two particular switches at the same time reverse the polarity of the voltage applied to the motor. This causes a change in the spinning direction of the motor. Below animation, illustrates H-Bridge circuit working.

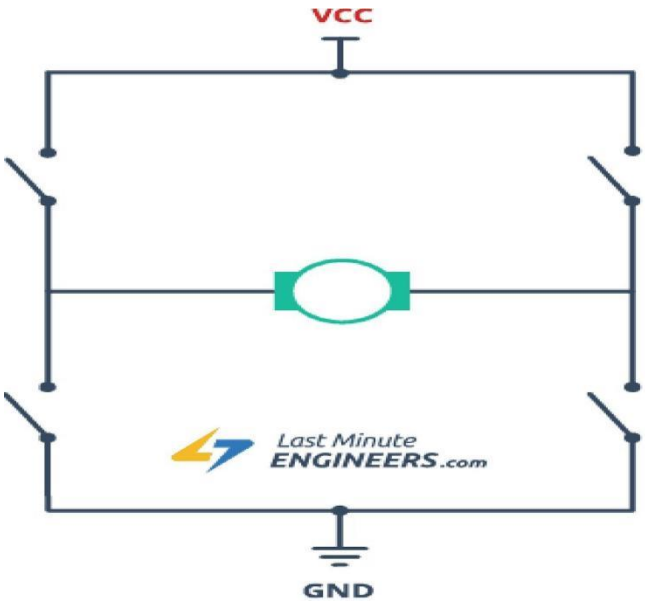


Fig 3.1: Motor H-Bridge Module

3.1.2 L298N Motor Driver IC:

- At the heart of the module is the big, black chip with a chunky heat sink is an L298N.
- The L298N is a dual-channel H-Bridge motor driver capable of driving a pair of DC motors.
- That means it can individually drive up to two motors making it ideal for building two-wheel robot platforms.

Power Supply:

The L298N motor driver module is powered through 3-pin 3.5mm-pitch screw terminals. It consists of pins for motor power supply (Vs), ground, and 5V logic power supply (Vss). The L298N motor driver IC has two input power pins viz. 'Vss' and 'Vs'.

From Vs pin, the H-Bridge gets its power for driving the motors which can be 5 to 35V.

Vss is used for driving the logic circuitry which can be 5 to 7V. And they both sink to a common ground named 'GND'. The module has an onboard 78M05 5V regulator from STMicroelectronics**.** It can be enabled or disabled through a jumper. When the jumper is removed, the 5V regulator gets disabled and we have to supply 5 Volts separately through a 5 Volt input terminal. You can put the jumper in place if the motor power supply is below 12V. If it is greater than 12V, you should remove the jumper to avoid the onboard 5V regulator from getting damaged. Also, D O does NOT supply power to both the motor power supply input and

5V power supply input when the jumper is in place.

Output Pins:

Fig 3.2: Motor L289 Output Pins

- The L298N motor driver's output channels for motors A and B are broken out to the edge of themodule with two 3.5mm-pitch screw terminals.
- You can connect two DC motors having voltages between 5 to 35V to these terminals.
- Each channel on the module can deliver up to 2A to the DC motor.
- However, the amount of current supplied to the motor depends on the system's power supply.

3.1.3 Control Pins:

- For each of the L298N's channels, there are two types of control pins that allow us to control the speed and spinning direction of the DC motors at the same time viz. Direction control pins& Speed control pins.

3.1.4 Direction Control Pins:

- Using the direction control pins, we can control whether the motor spins forward or backward.
- These pins control the switches of the H-Bridge circuit inside L298N IC.

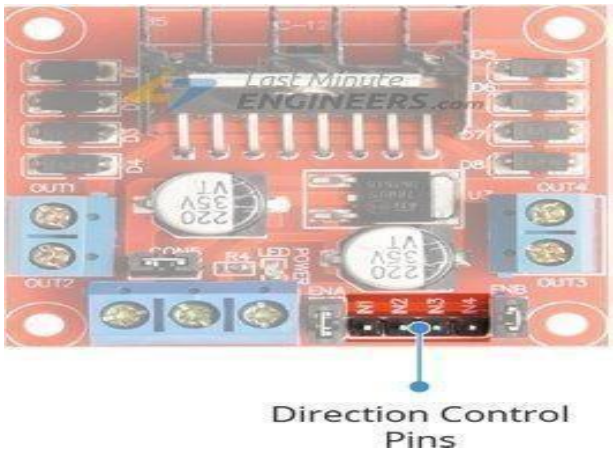


Fig 3.3: Direction Control Pins

- The module has two direction control pins for each channel.
- The IN1 and IN2 pins control the spinning direction of motor A while IN3 and IN4 control motor B.
- The spinning direction of a motor can be controlled by applying either a logic HIGH(5 Volts) orlogic LOW(Ground) to these inputs.
- The below chart illustrates how this is done.

Input1	Input2	Spinning Direction
Low(0)	Low(0)	Motor OFF
High(1)	Low(0)	Forward
Low(0)	High(1)	Backward
High(1)	High(1)	Motor OFF

3.1.5 Speed Control Pins:

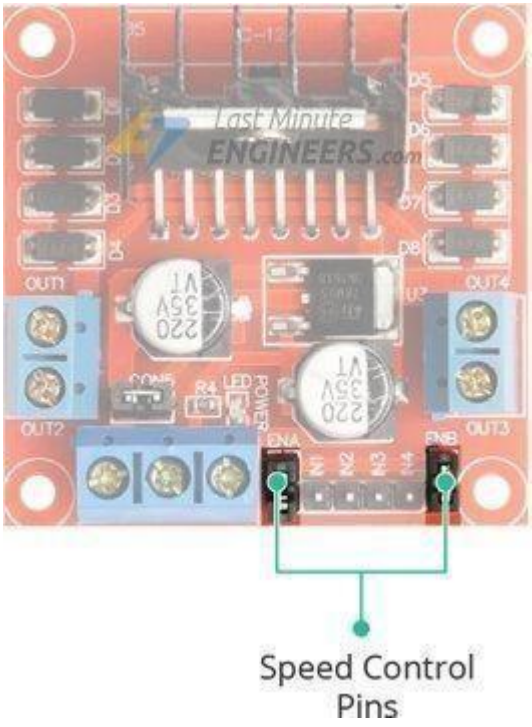


Fig 3.4: Speed Control Pins

- The speed control pins viz. ENA and ENB are used to turn the motors ON, and OFF and controltheir speed.
- Pulling these pins HIGH will make the motors spin, pulling them LOW will make them stop.
- But, with Pulse Width Modulation (PWM), we can control the speed of the motors.
- The module usually comes with a jumper on these pins. When this jumper is in place, the motoris enabled and spins at maximum speed.
- If you want to control the speed of motors programmatically, you need to remove the jumpersand connect them to PWM-enabled pins on Arduino.

3.1.6 Wiring L298N motor driver module with Arduino UNO:

- Start by connecting the power supply to the motors.
- They are rated for 3 to 12V. So, we will connect the external 12V power supply to the VCCterminal.
- Considering the internal voltage drop of L298N IC, the motors will receive 10V and will spin atslightly lower RPM. But, that's OK.
- Next, we need to supply 5 Volts for the L298N's logic circuitry.
- We will make use of the onboard 5V regulator and derive the 5 volts from the motor power supply so, keep the 5V-EN jumper in place.
- Now, the input and enable pins (ENA, IN1, IN2, IN3, IN4, and ENB) of the L298N module areconnected to six Arduino digital output pins(9, 8, 7, 5, ,4, and 3).
- Note that the Arduino output pins 9 and 3 are both PWM-enabled.
- Finally, connect one motor to terminal A(OUT1 & OUT2) and the other motor to terminal B(OUT3 & OUT4).

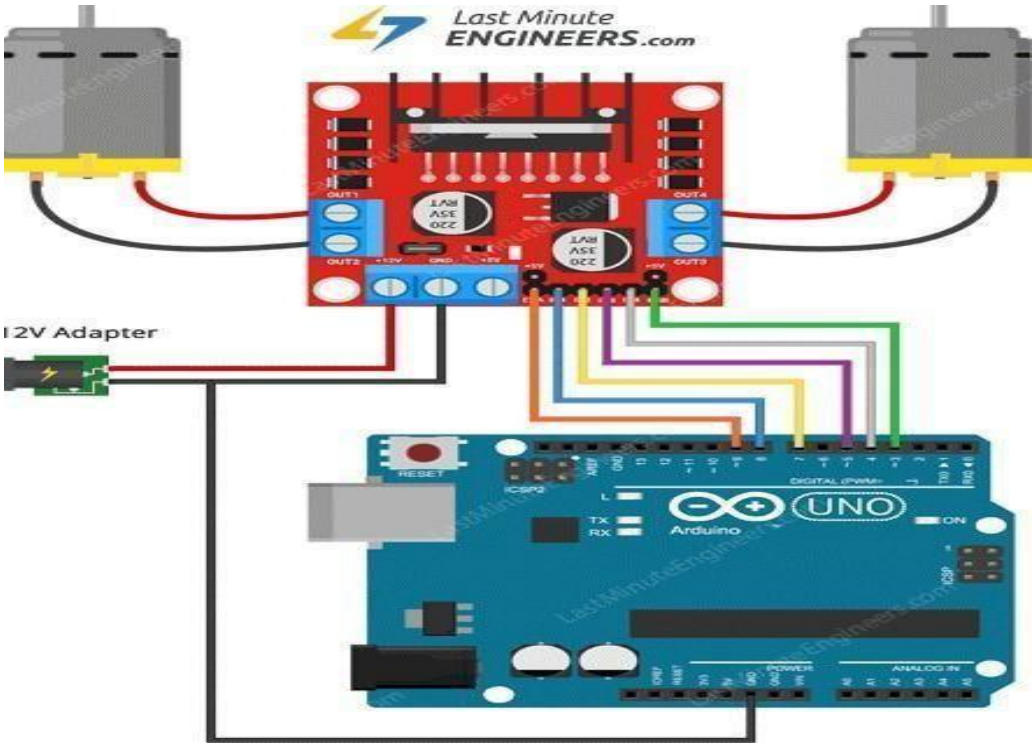


Fig 3.5: Interfacing L298N motor driver module with Arduino UNO

3.2 Interfacing Fingerprint sensor with Arduino:

3.2.1 Sensor Pinout:

The sensor has six pins that are labeled in the figure below.

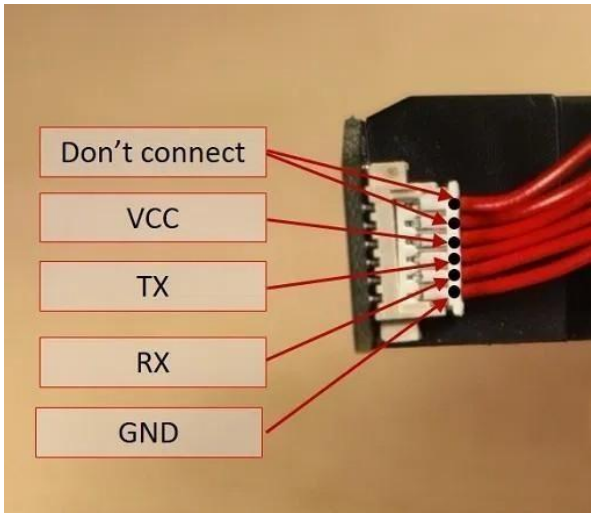


Fig 3.6: Fingerprint pin diagram

The fingerprint sensor module used in this project came with really thin wires, so soldering breadboard-friendly wires were needed. We recommend using different colors according to the pin function. In our case:

- DNC – white wires
- VCC – red wire
- TX – blue wire
- RX – green wire
- GND – black wire

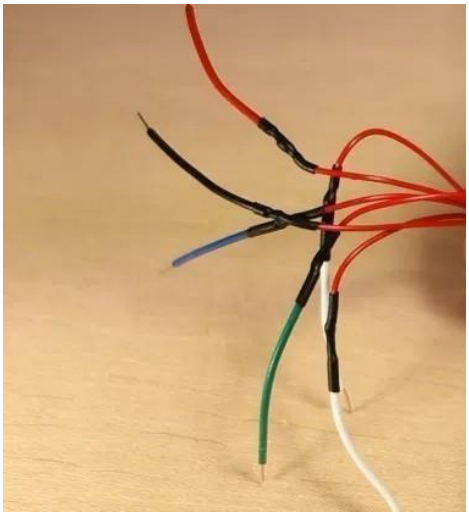


Fig 3.7: Fingerprint sensor connections

The following table shows how to wire the sensor to the Arduino

Table 3.1: Interfacing fingerprint sensor with Arduino

Fingerprint Sensor	**Arduino**
VCC	5v (it also works with 3.3V)
TX	RX (digital pin 2, software serial)
RX	TX (digital pin 3, software serial)
GND	GND

3.2.2 Installing the Adafruit Fingerprint Sensor Library:

The easiest way to control the fingerprint sensor module with the Arduino is by using the Adafruit library for this sensor. Follow the next instructions to install the library:

- Click here to download the Adafruit Fingerprint Sensor library. You should have a .zip folder in your Downloads folder
- Unzip the .zip folder and we should get the Adafruit-Fingerprint-Sensor-Library-master folder
- Rename your folder from Adafruit-Fingerprint-Sensor-Library-master the folder the to Adafruit**_Fingerprint_Sensor_Library folder**
- Move the folder to your Arduino IDE installation libraries folder
- Finally, re-open your Arduino IDE.

Enroll a New Fingerprint:

Having the fingerprint sensor module wired to the Arduino, follow the next steps to enroll a newfingerprint. Make sure you've installed the Adafruit Fingerprint Sensor library previously.

1. In the Arduino IDE, go to **File** > **Examples** > **Adafruit Fingerprint Sensor Library** > **Enroll**.
2. Upload the code, and open the serial monitor at a baud rate of 9600.
3. You should enter an ID for the fingerprint. As this is your first fingerprint, type 1 at the top leftcorner, and then, click the **Send** button.

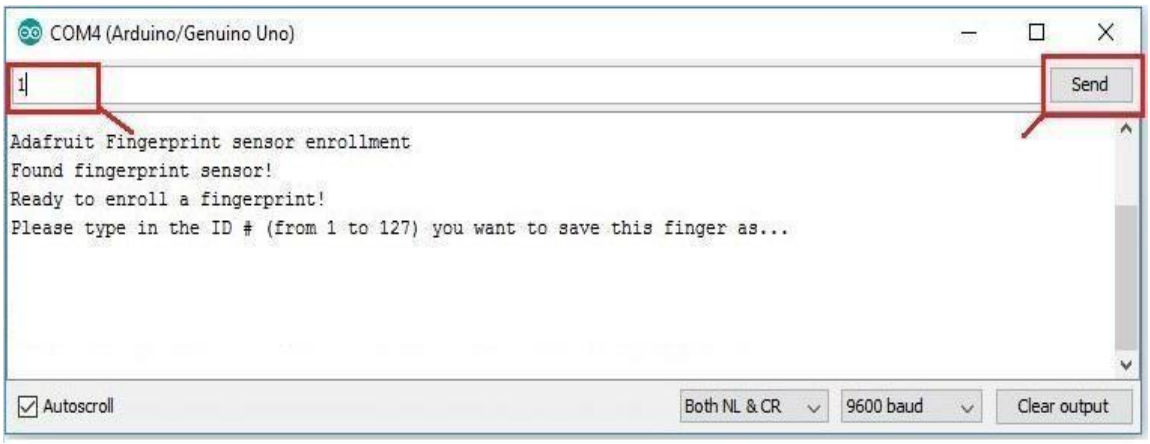


Fig 3.8: Fingerprint Enrollment

4. Place your finger on the scanner and follow the instructions on the serial monitor.

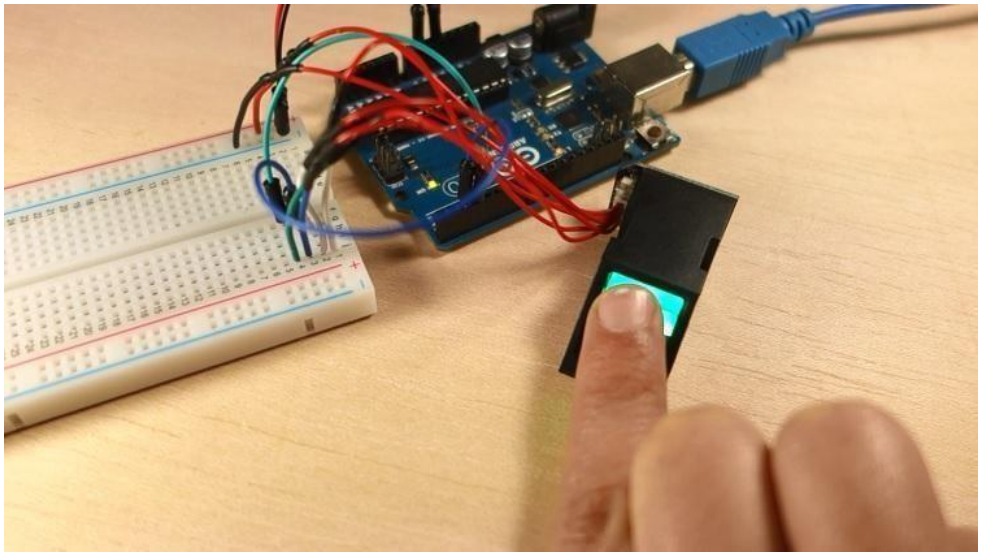


Fig 3.9:Scanning finger on fingerprint sensor

Finger print need to be placed twice on the scanner. If you get the "**Prints matched!**" message, as shown below, your fingerprint was successfully stored. If not, repeat the process, until you succeed.

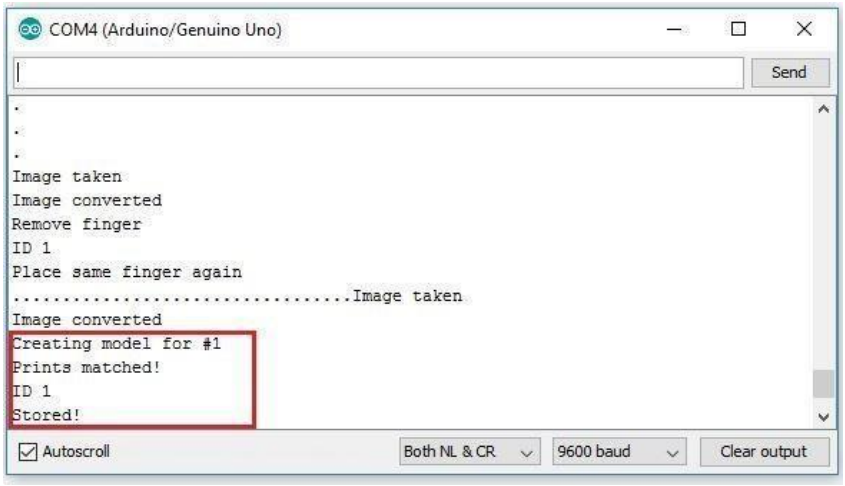


Fig 3.10: Output of sensor

Store as many fingerprints you want using this method. In this we can store upto 128 finger prints.

3.2.3 Finding a Match:

We can save fingerprints saved on different IDs. To find a match with the fingerprintsensor, follow the next instructions.

In the Arduino IDE, go to **File** > **Examples** > **Adafruit Fingerprint Sensor Library** > **Fingerprint**

and upload the code to your Arduino board.

1. Open the Serial Monitor at a baud rate of 9600. You should see the following message:

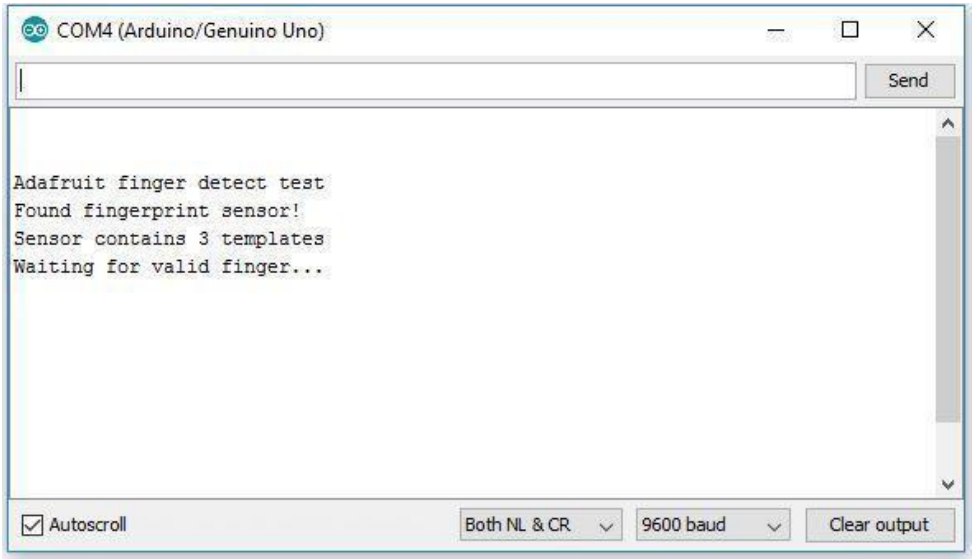


Fig 3.11: Serial monitor

2. Place the finger to be identified on the scan.
3. On the serial monitor, you can see the ID that matches the fingerprint. It also shows theconfidence – the higher the confidence, the similar the fingerprint is with the stored fingerprint.

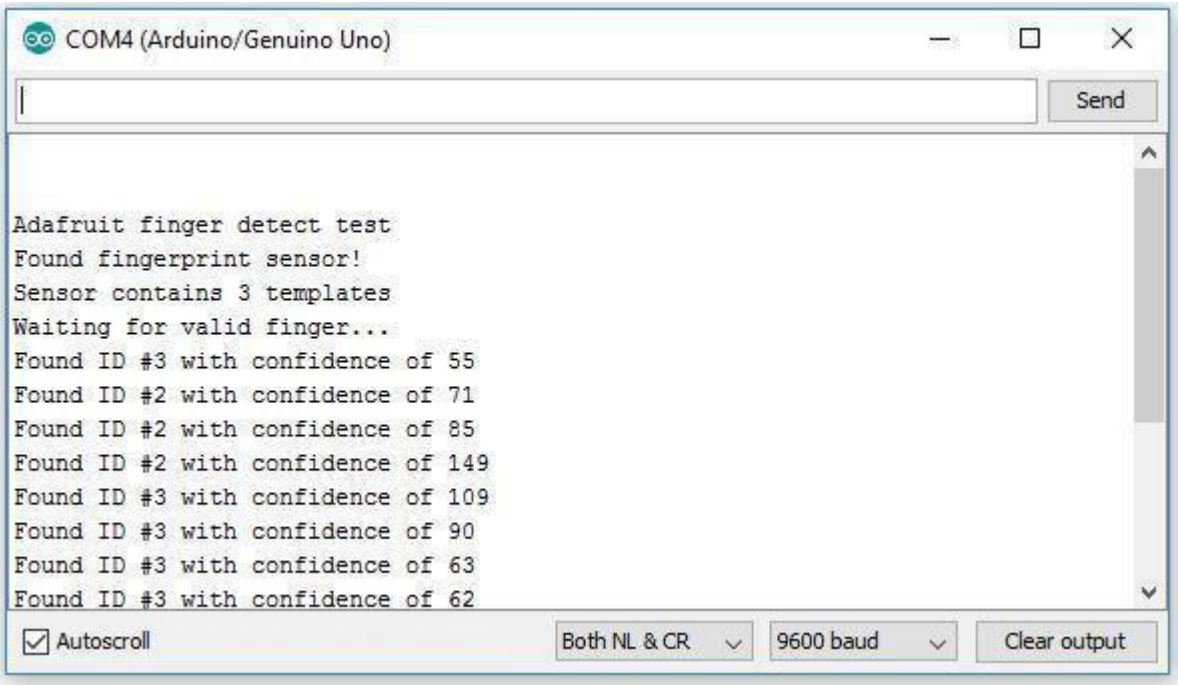


Fig 3.12: Output of fingerprint sensor in serial monitor

3.3 Interfacing Sim 900A GSM With Arduino:

we are going to see how to interface the GSM Module with Arduino. There are different kinds of GSM modules available on the market. We are using the most popular module based on Simcom SIM900 and Arduino Uno for this tutorial. Interfacing a GSM module to Arduino is pretty simple. We only need to make 3 connections between the gsm module and Arduino.

A **GSM Module** is a GSM Modem (like SIM 900) connected to a PCB with different types of output taken from the board – say TTL Output (for Arduino, 8051, and other, microcontrollers) and RS232 Output to interface directly with a PC (personal computer). The board will also have pins or provisions to attach the mic and speaker, to take out +5V or other values of power and ground connections.

Booting the GSM Module:

- Insert the SIM card into the GSM module and lock it.
- Connect the adapter to the GSM module and turn it ON!
- Now wait for some time (say 1 minute) and see the blinking rate of 'status LED' or 'network LED' (GSM module will take some time to establish a connection with mobile network)
- Once the connection is established successfully, the status/network LED will blink continuously every 3 seconds. You may try making a call to the mobile number of the sim card inside the GSM module. If you hear a ring back, the gsm module has successfully established a network connection.

Connecting GSM Module to Arduino:

There are two ways of connecting the GSM module to Arduino. In any case, the communicationbetween
Arduino and GSM modules is serial.

1. Without Software Serial Library
2. With Software Serial Library

Method 1:

If you are going with the first method, you have to connect the Tx pin of the GSM module to the Rx pin of Arduino and the Rx pin of the GSM module to the Tx pin of Arduino. GSM Tx –> Arduino Rx and GSM Rx –> Arduino Tx.

Now connect the ground pin of Arduino to the ground pin of the gsm module!.

Circuit diagram of interfacing GSM module with Arduino:

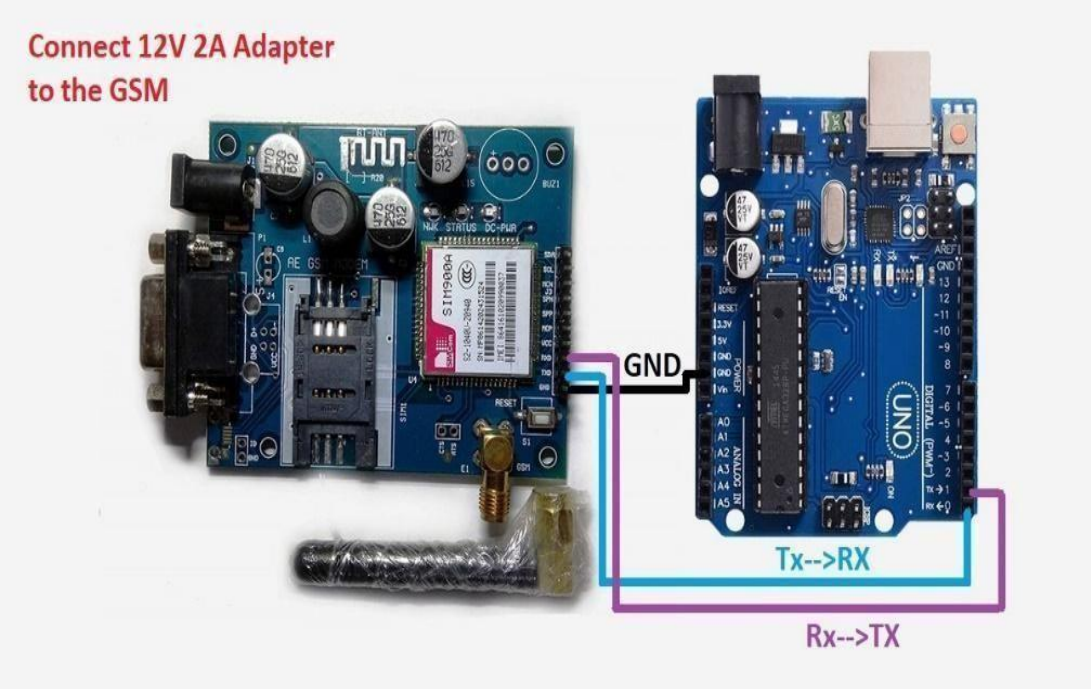


Fig 3.13: Interfacing GSM module with Arduino

Method 2:

To avoid the difficulty mentioned above, we are using an alternate method in which two digital pins of Arduino are used for serial communication. We need to select two **PWM-enabled pins of Arduino** for this method. So choose pins **9** and **10** (which are PWM-enabled pins). This method is made possible with the **SoftwareSerial Library** of Arduino. Software Serial is a library of Arduino which enables serial data communication through other digital pins of Arduino. The library replicates hardware functions and handles the task of serial communication.

3.4 Interfacing RFID With Node MCU:

As explained in the initial stage of the instructable an RFID reader is a wireless technology usedfor wireless communication where it is used to read or write data over an RFID tag. a simple NodeMCU RFID reader using the MFRC522 module, and program the NodeMCU to provide access when the right card is detected

Hardware Components

- NodeMCU
- MFRC522 RFID Reader
- RFID Tags (**13.56 MHz**)
- Bread Board
- Jumper Wires
- Micro USB
- Cable Software Components
- Arduino IDE

A **RFID reader** is a device used to gather information from an RFID tag, which is used to trackindividual objects. Radio waves are used to transfer data from the tag to a reader.

A **passive tag** is an RFID tag that does not contain a battery, the power is supplied by the reader. When radio waves from the reader are encountered by a passive rfid tag, the coiled antenna within the tag forms a magnetic field. The tag draws power from it, energizing the circuits in thetag.

Specifications

- Input voltage: 3.3v
- Frequency: 13.56MHz

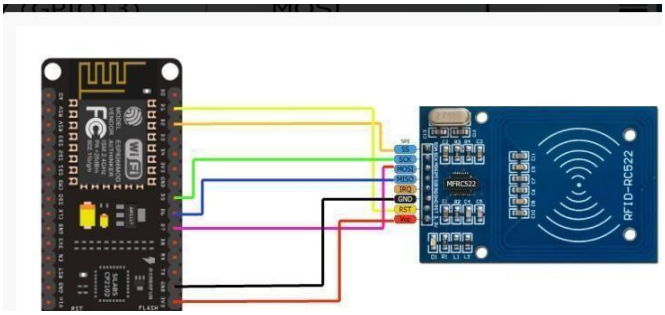


Fig 3.14: Interfacing Node MCU with RFID

3.5 Interfacing LCD With NODEMCU:

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 a 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 an LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling the 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.

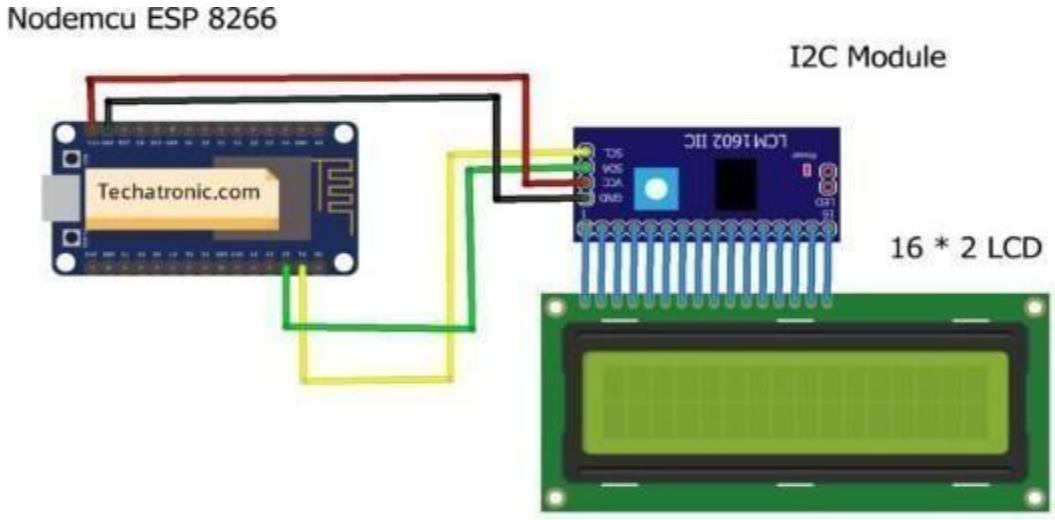


Fig 3.15: Interfacing LCD with Node MCU

Connection Table:

Table 3.2: Connections LCD with Node MCU

Nodemcu esp8266	I2C Module
Vin, VV	VCC
GND	GND
D2 Pin (SDA)	SDA Pin
D1 Pin (SCL)	SCL Pin
16*2 LCD Display	**I2C Module**
16 Pin connect	16 Pin connect

- Mount the Node MCU board on a breadboard so it is easy to make connections.

- Attach the Pins of the I2C module with the 16×2 LCD.
- Connect the VIN or 3.3-volts pin of the Node MCU with the VCC pin of the I2C module andthe GND pin of the Node MCU with the GND pin of the I2C module.
- Join the SDA pin of the I2C module with the digital-2 pin of the Node MCU for the node MCUTutorial.
- Connect the SCL pin of the I2C module with the digital-1 pin of the Node MCU.

3.6 Interfacing GPS NEO 6M With Arduino:

The NEO-6M GPS module is shown in the figure below. It comes with an external antenna and doesn't come with header pins.

Fig 3.16: GPS NEO 6M

- This module has an external antenna and built-in EEPROM.
- Interface: RS232 TTL
- Power supply: 3V to 5V
- Default baud rate: 9600 bps
- Works with standard NMEA sentences

Pin Wiring:

The NEO-6M GPS module has four pins: VCC,RX,TX and GND. The module communicates with the Arduino via serial communication using the TX and RX pins, so the wiring couldn't besimpler:

Table 3.3:connections GPS with Arduino

NEO-6M GPS Module	**Wiring to Arduino UNO**
VCC	5V
RX	TX pin defined in the software serial
TX	RX pin defined in the software serial
GND	GND

Schematics:

Wire the NEO-6M GPS module to your Arduino by following the schematic below.

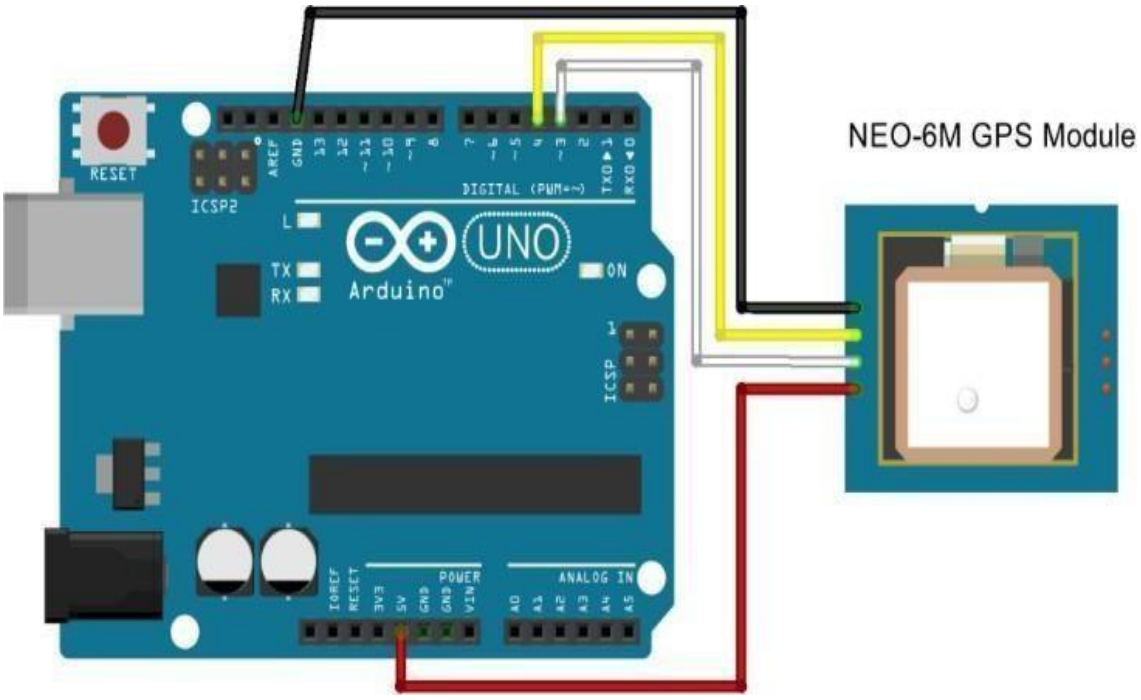


Fig 3.16:Interfacing of GPS with Arduino

- The module GND pin is connected to the Arduino GND pin.
- The module RX pin is connected to Arduino pin 3.
- The module TX pin is connected to Arduino pin 4.
- The module VCC pin is connected to the Arduino 5V pin.

CHAPTER4

ARDUINO PROGRAMMING

Arduino is a prototype platform (open-source) based on easy-to-use hardware and software. It consists of a circuit board, which can be preprogramme deferred to as a microcontroller, and a ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board. Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

The key features are:

- Arduino boards can read analog or digital input signals from different sensors and turn them into an output such as activating a motor, turning LED on/off, connecting to the cloud, and many other actions.
- You can control your board functions by sending a set of instructions to the microcontroller on the board via Arduino IDE (referred to as uploading software).
- Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.
- Finally, Arduino provides a standard form factor that breaks the functions of the microcontroller into a more accessible package.

ARDUINO PIN DIAGRAM:

The Arduino UNO Board, with the specification of pins, is shown below:

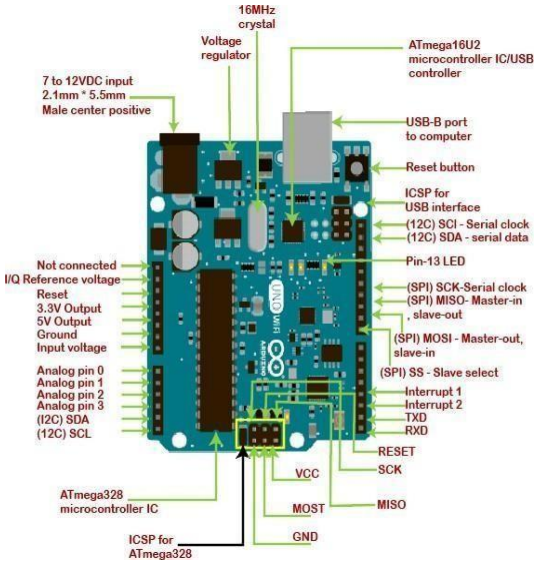


Fig 4.1: Pin diagram

4.2 Programming:

4.2.1 BASICS:

Coding Screen:
The set of statements in the setup and loop blocks are enclosed with the curly brackets. We can write multiple statements depending on the coding requirements for a particular project.

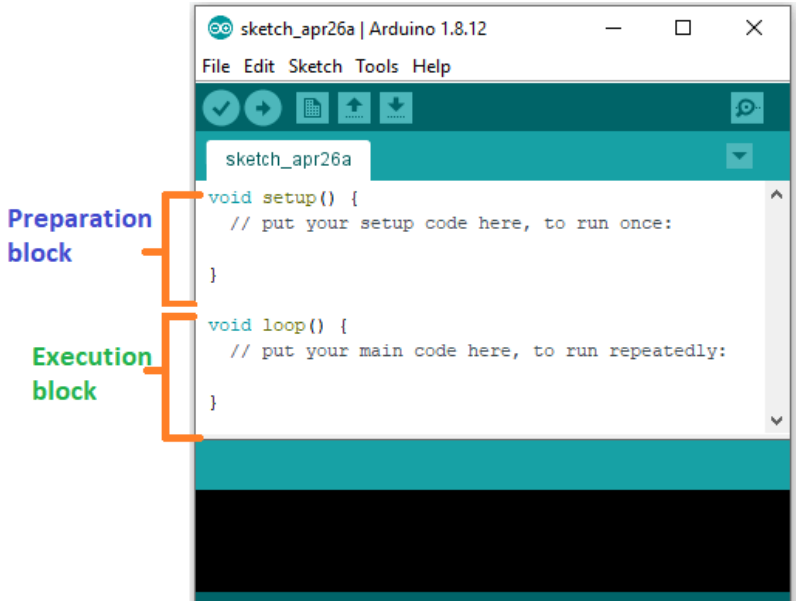


Fig 4.2:coding screen

Set-Up (): It contains an initial part of the code to be executed. The pin modes, libraries, variables, etc., are initialized in the setup section. It is executed only once during the uploading of the program and after reset or power-up of the Arduino board. Zerosetupsp () resides at the top of each sketch. As soon as the program starts running, the code inside the curly bracket is executed in the setup and it executes only once.

Loop (): The loop contains statements that are executed repeatedly. The section of code inside the curly brackets is repeated depending on the value of the variables.

Pin Mode ():

The specific PIN is set as the INPUT or OUTPUT in the pinMode () function. The Syntax is:
pinMode (pin, mode)
Where,

pin: It is the PIN. We can select the PIN according to the requirements. **Mode:** We can set the mode as INPUT or OUTPUT according to the corresponding PIN.

The OUTPUT mode of a specific PIN provides a considerable amount of current to other circuits, which is enough to run a sensor or to light the LED brightly. The output state of a pin isconsidered the low-impedance state.

The high current and short circuit of a pin can damage the Atmel chip. So, it is recommended toset the mode as OUTPUT.

Digital Write ():

The digitalWrite ( ) function is used to set the value of a pin as HIGH or LOW.

HIGH: It sets the value of the voltage. For the 5V board, it will set the value of 5V, while for3.3V, it will set the value of 3.3V.
LOW: It sets the value = 0 (GND).

If we do not set the pinMode as OUTPUT, the LED may light dim. The syntax is:
digitalWrite(pin, value HIGH/LOW).

The digitalRead () function will read the HIGH/LOW value from the digital pin, and the digitalWrite () function is used to set the HIGH/LOW value of the digital pin.
pin: We can specify the PIN or the declared variable. Delay ():

The delay () function is a blocking function to pause a program from doing a task during the specified duration in milliseconds.

4.2.2 Syntax and Programme Flow

Syntax:

Syntax in Arduino signifies the rules that need to be followed for the successful uploading of the Arduino program to the board. The syntax of Arduino is similar to the grammar in English. It means that the rules must be followed to compile and run our code successfully. If we break those rules, our computer program may compile and run, but with some bugs.

Functions:

The functions in Arduino combine many pieces of lines of code into one.

The functions usually return a value after finishing execution. But here, the function does not return any value due to the presence of a void.

The setup and loop function have **void** keywords present in front of their function name. The multiple lines of code that a function encapsulates are written inside curly brackets. Every closing curly bracket '} ' must match the opening curly bracket '{'in the code.

We can also write our functions, which will be discussed later in this tutorial.

Spaces:

Arduino ignores the white spaces and tabs before the coding statements.

The coding statements in the code are intended (empty spacing at the starting) for easy reading.In the function definition, loop, and conditional statements, 1 intent = 2 spaces.

The compiler of Arduino also ignores the spaces in the parentheses, commas, blank lines, etc.

Uses of Parenthesis ():

It denotes the function like void setup ( ) and void loop ( ).

The parameter's inputs to the function are enclosed within the parentheses.

It is also used to change the order of operations in mathematical operations. Semicolon;It is the statement terminator in C as well as C++.

A statement is a command given to the Arduino, which instructs it to take some kind of action. Hence, the terminator is essential to signify the end of a statement. We can write one or more statements in a single line, but with a semicolon indicatingthe end of each statement.

The compiler will indicate an error if a semicolon is absent in any of the statements.It is recommended to write each statement with a semicolon in a different line, which makes the code easier to read.

We are not required to place a semicolon after the curly braces of the setup and loop function

Arduino processes each statement sequentially. It executes one statement at a time before moving to the next statement.

Program Flow:

The program flow in Arduino is similar to the flowcharts. It represents the execution of a program in order.

The Arduino coding process in the form of the flowchart is shown below:

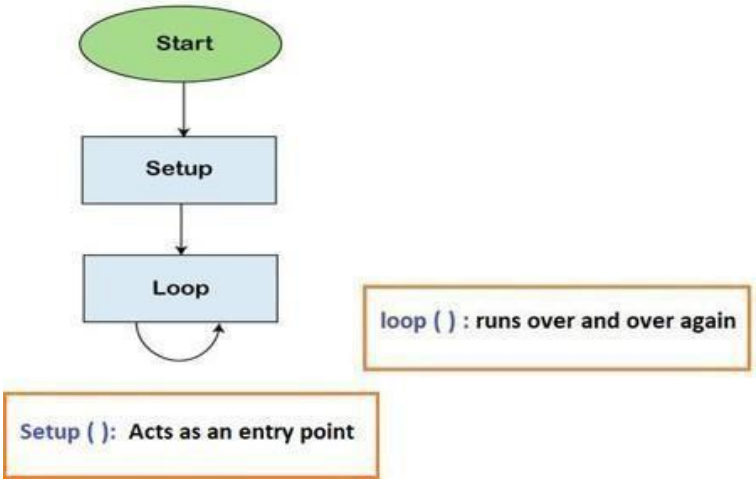


Fig 4.3 :Program Flow chart

4.2.3 Serial Functions:

Serial.begin():
The serial. Begin () sets the baud rate for serial data communication. The **baud** rate signifies the data rate in bits per second.
The default baud rate in Arduino is **9600 bps (bits per second**). We can specify other baud rates as well, such as 4800, 14400, 38400, 28800, etc.

serial: It signifies the serial port object.

speed: It signifies the baud rate or bps (bits per second) rate. It allows ***long*** data types.

conFig: It sets the stop, parity, and data bits. Serial. print:

The serial. print ( ) in Arduino prints the data to the serial port. The printed data is stored in the ASCII (American Standard Code for Information Interchange) format, which is a human-readable text.

Each digit of a number is printed using the ASCII characters.

The printed data will be visible in the **serial monitor**, which is present in the right corner of the toolbar.

The Serial. Print( ) is declared in two formats, which are shown below:

print(value)

print(value, format)

serial: It signifies the serial port object.

print: The print ( ) returns the specified number of bytes written.

value: It signifies the value to print, which includes any data type value.

4.2.4 analogRead ():

The **analogRead ()** function reads the value from the specified analog pin present on the particular Arduino board. The ADC (Analog to Digital Converter) on the Arduino board is a multichannel converter. It maps the input voltage and the operating voltage between the values 0 and 1023. The operating voltage can be **5V or 3.3V.** The values from 0 to 1023 are the integer values. It can also be written as 0 to (2^10) -1. The time duration to read an analog input signal on the boards (UNO, Mega, Mini, and Nano) is about 100 microseconds or 0.0001 seconds. Hence, the maximum reading rate of analog input is about 10000 times per second.

Let's discuss the operating voltage and resolution of some Arduino boards.

The Operating voltage of Arduino UNO, Mini, Mega, Nano, Leonardo, and Micro is**5V,** and the resolution is **10 bits**.

The Operating voltage of MKR family boards, Arduino Due, and Zero is **3 V,** and the resolution is 12 bits.

4.2.5ArduinoDatatypes:

The data types are used to identify the types of data and the associated functions for handling the data. It is used for declaring functions and variables, which determines the bit pattern and the storage space. The data types that we will use in the Arduino are listed below.

- void Data Type
- int Data Type
- Char Data Type
- Float Data Type
- Double Data Type
- Unsigned int Data Type
- short Data Type
- long Data Type
- Unsigned long Data Type
- byte data type
- word data type

4.2.4Arduino Variables

The variables are defined as the place to store the data and values. It consists of a name, value, and type. The variables can belong to any data type such as int, float, char, etc. Consider the URL -Arduino data types for detailed information.

Ex: **int pin=8**

Here, the **int** data type is used to create a variable named **pin** that stores the value **8**. It also means thatvalue 8 is initialized to the variable **pin**.

We can modify the name of the variable according to our choice.
The variables can be declared in two ways in Arduino, which are listed below:

- Local variables
- Global variables

Local Variables

The local variables are declared within the function. The variables have scope only within the function. These variables can be used only by the statements that lie within that function.

4.2.5Arduino Operators:

The operators are widely used in Arduino programming from basics to advanced levels. It plays a crucial role in every programming concept like C, C++, Java, etc. The operators are used to solve logical and mathematical problems. For example, to calculate the temperature given by the sensor based on some analog voltage. The arithmetic operators are used to perform basic mathematical functions. Based on these arithmetic operators the logic for the desired program can be devised. There are eleven operators used for the mathematical calculations that are explained in below.

The types of Operators classified in Arduino are:

1. Arithmetic Operators
2. Boolean Operators
3. Bitwise Operators
4. Comparison Operator.

1.Arithmetic Operators:

There are six basic operators responsible for performing mathematical operations in Arduino, which are listed below:

Assignment Operator (=):

The Assignment operator in Arduino is used to set the variable's value. It is quite different from the equal symbol (=) normally used in mathematics.

Addition (+):

The addition operator is used for the addition of two numbers. For example, P + Q.

Subtraction (-):

Subtraction is used to subtract one value from another. For example, P - Q. Multiplication (*):

Multiplication is used to multiply two numbers. For example, P * Q. Division Division (/):

The division is used to determine the result of one number divided by another.
For example, P/Q.

Modulo (%):

The Modulo operator is used to calculate the remainder after the division of one number by another number

2.Boolean Operators:

The Boolean Operators are NOT (!), Logical AND (& &), and Logical OR (| |). Let's discuss theabove operators in detail.

Logical AND (& &):

The result of the condition is true if both the operands in the condition are true.

Logical OR (| |):

The result of the condition is true if either of the variables in the condition is true.

NOT (!):

It is used to reverse the logical state of the operand.

1.Comparison Operators:

The comparison operators are used to compare the value of one variable with the other. The comparison operators are listed below:

less than (<):

The less than operator checks that the value of the left operand is less than the right operand. The statement is true if the condition is satisfied.

greater than (>):

The less than operator checks that the value of the left side of a statement is greater than the right side. The statement is true if the condition is satisfied.

equal to (= =):

It checks the value of two operands. If the values are equal, the condition is satisfied. not equal

to (! =):

It checks the value of two specified variables. If the values are not equal, the condition will be correct and satisfied.

less than or equal to (< =):

The less or equal operator checks that the value of the left side of a statement is less or equal to the value on the right side. The statement is true if either of the condition is satisfied.

greater than or equal to (> =):

The greater or equal operator checks that the value of the left side of a statement is greater or equal to the value on the right side of that statement. The statement is true if the condition is satisfied.

2.Bitwise Operators:

The Bitwise operators operate at the binary level. These operators are quite easy to use. There

are various bitwise operators. Some of the popular operators are listed below:

bitwise NOT (~):

The bitwise NOT operator acts as a complement for reversing the bits.

bitwise XOR (^):

The output is 0 if both the inputs are the same, and it is 1 if the two input bits are different.

bitwise OR (|):

The output is 0 if both of the inputs in the OR operation are 0. Otherwise, the output is 1. The two input patterns are of 4 bits.

bitwise AND (&):

The output is 1 if both the inputs in the AND operation are 1. Otherwise, the output is 0. The two input patterns are of 4 bits.

bitwise left shift (< <):

The left operator is shifted by the number of bits defined by the right operator.

bitwise right shift (> >):

The right operator is shifted by the number of bits defined by the left operator.

4.2.9 Arduino IF Statement:

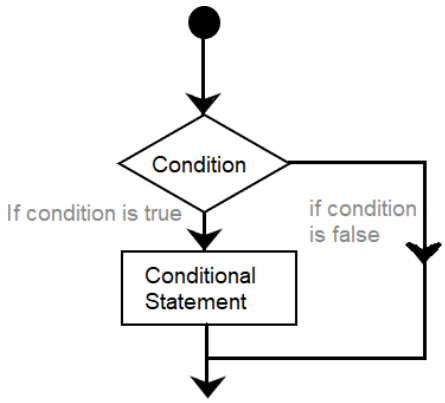


Fig 4.4: Flow chart of IF statement

The if () statement is the conditional statement, which is the basis for all types of programming languages. If the condition in the code is true, the corresponding task is performed.

The function is performed accordingly. It returns one value if the condition in a program is **true**. It further returns another value if the condition is **false**. It means that if ( ) statement checks for the condition and then executes a statement or a set of statements.

2.If-Else:

The if-else condition includes if ( ) statement and else ( ) statement. The condition in the else statement is executed if the result of the If ( ) statement is false.

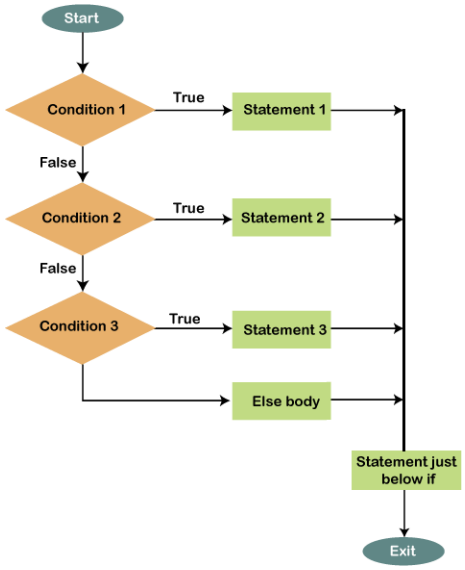


Fig 4.5: Flow Chart of If-Else

The statements will be executed one by one until the true statement is found. When the true statement is found, it will skip all other if and else statements in the code and runs the associated blocks of code.

CHAPTER 5
RESULTS AND DISCUSSIONS

5.1 ARDUINO SIMULATOR:

The Arduino simulator is a virtual portrayal of the circuits of Arduino in the real world. We can create many projects using a simulator without the need for any hardware. The Simulator helps to learn, program, and create their projects without wasting time on collecting hardware equipment.

5.1.1ADVANTAGES OF USING A SIMULATOR:

There are various advantages of using the simulator, which are listed below:

- It saves money because there is no need to buy hardware equipment to make a project.
- The task to create and learn Arduino is easy for beginners.
- We need not worry about the damage to the board and related equipment.
- No messy wire structure required.
- It supports line-to-line debugging and helps to find out the errors easily.
- Coding can be learned and build projects anywhere with our computer and internet connection.
- We can also share our design with others.

5.1.2 TYPES OF SIMULATOR:

There are various simulators available. Some are available for free, while some require a licenseto access the simulators. Some types of simulators are listed below:

- Autodesk Tinker cad
- Emulator Arduino Simulator
- Autodesk Eagle
- Proteus Simulator
- Virtronics Arduino Simulator

5.1.2 ACESSING SIMULATOR:

Here, we are using the **Autodesk Tinker cad Simulator**. The steps to access the TINKERCADare listed below:

1. Open the official website of tinker cad. **URL: https://www.tinkercad.com/**

A window will appear, as shown below:

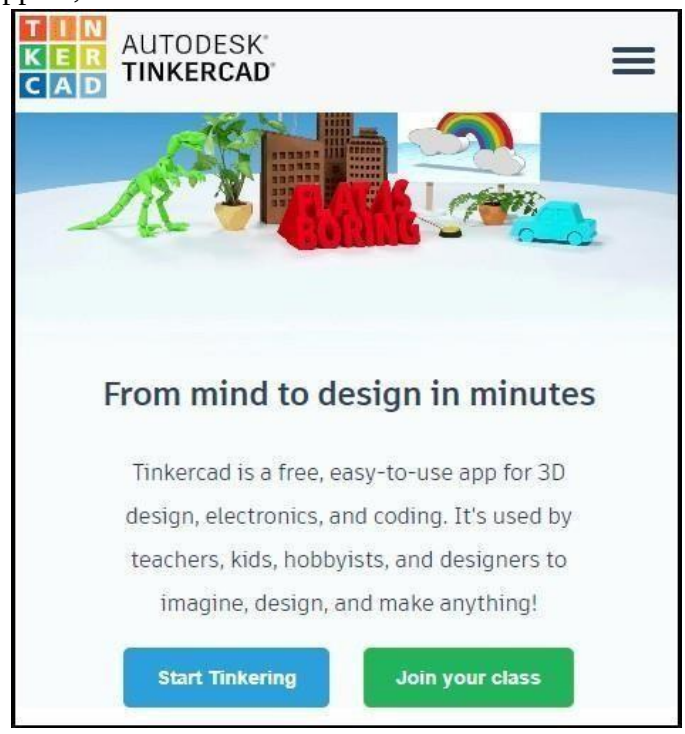


Fig 5.1: Autodesk Tinkercad window

2. Click on the three horizontal lines present in the upper right corner.

3. Click on the '**Sign in**' option, if you have an account in Autodesk. Otherwise, click on the '**JOIN NOW**' option if you don't have an account, as shown below:

Fig 5.2: Menu Window

The **SIGN IN** window will appear as:

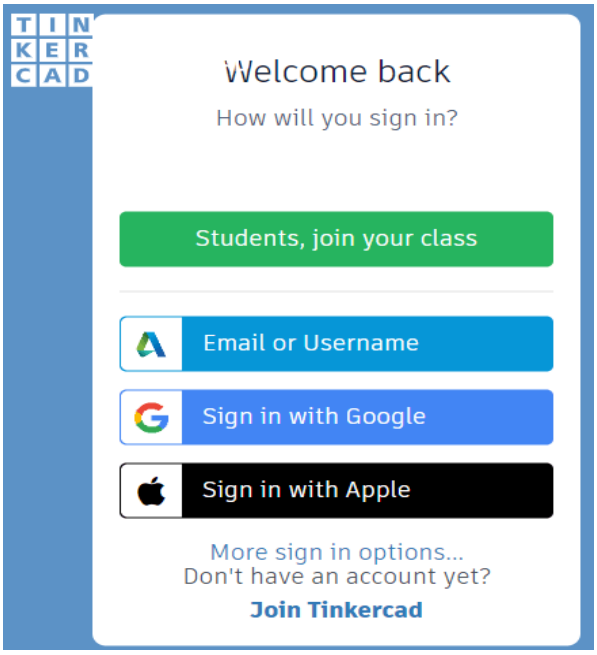


Fig 5.3: Sign in Window

We can select any sign-in method. Specify the username and password. We already havean account in Autodesk, so we will sign indirectly with the username and password.

The **JOIN** window will appear as:

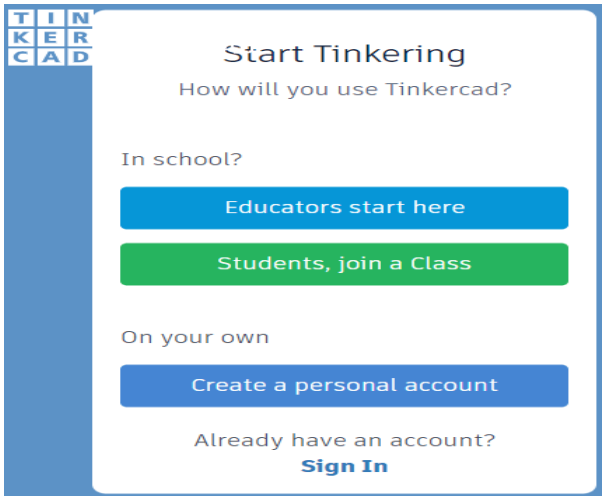


Fig 5.4: Join window

Select the preference according to the requirements and sign in using Gmail, etc.

- Now, a window will appear, as shown below:

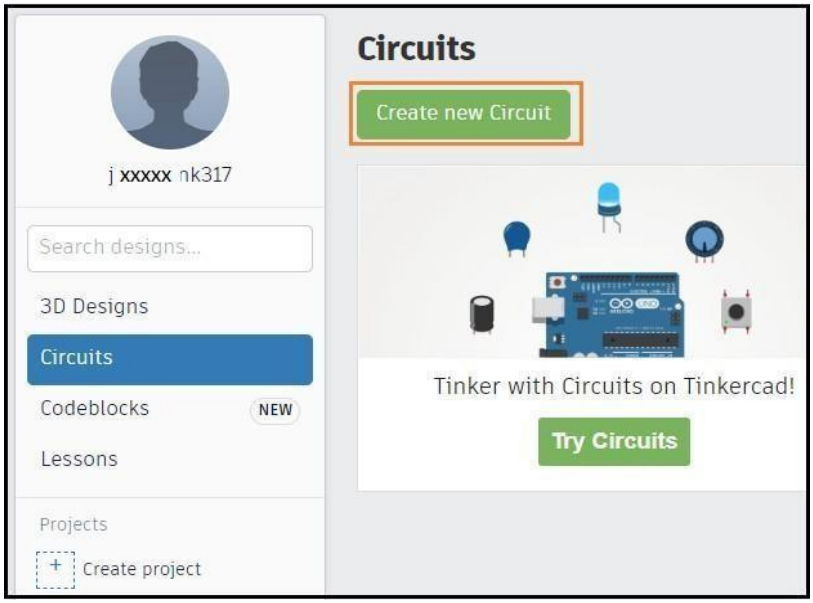


Fig 5.5: Tinker cad Window

- Click on the '**Create new circuit**' option to start designing the Arduino circuit, as shown above. The '**Circuits**' option will also show the previous circuits created by the user. The design option is used for creating the 3D design, which is of no use in Arduino.
- We are now ready to start with the Autodesk Tinkercad. We can start creating our projects.

5.1.3 FEATURES OF SIMULATOR:

- Glow and move circuit assembly. It means we can use the components of a circuit according to the project requirement.
- Glow here signifies the glowing of LED.Integrated product design. It means the electronic components used in the circuitry are real.Arduino Programming.
- We can directly write the program or code in the editor of the simulator.

- We can also consider some ready-made examples provided by thetinker cad for better understanding.
- We can prototype our designs withinthe browser before implementing them in real-time.

5.2 SIMULATION RESULTS:

Here the results obtained after interfacing the various hardware components using Arduino were presented.

5.1.4 MOTOR

5.2.1.1 Connections of motor:

Initially the motor is interfaced by connecting through the motor driver L298n and Arduino uno board as shown in fig.6.1. Two motors were operated at a time using motor driver L298n. Motor driver has four input pins, four output pins and two enable pins(ENA and ENB). Enable pins can control speed of motor. Here we use only one motor, output 1 and output 2 of motor driver is connected to terminal 1 and terminal 2 of motor respectively. Enable pin1 (ENA) of motor driver is connected to 5^{th} pin of Arduino. Input 1 and input 2 of motor driver

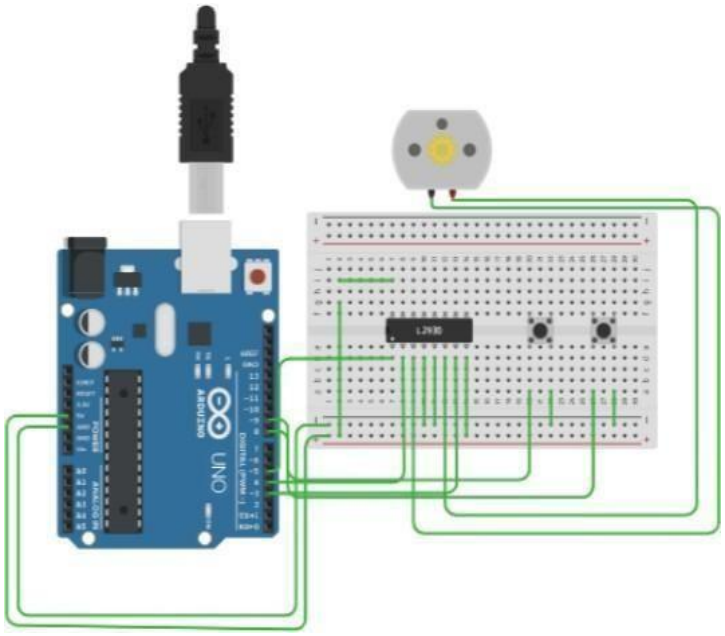


Fig. 5.6: Interfacing of motor with Arduino through motor driver L298n

is connected to 4th pin and 3rd pin of Arduino respectively. Gnd and V_{CC} of motor driver were connected to is connected to Gnd, 5v of Arduino respectively.

After completion of the simulation. The motor starts rotating in both directions by using the pushbutton and we can control the speed of the motor.

5.2 LCD

5.2.2.1 LCD connections:

They are 16 pins in LCD. First two pins is connected to ground, vcc .Next pin used to adjust the contrast of LCD. Next three pins are register select , read and write pins and enable pins. Register select pin is connected to A0 of Arduino. Enable pin is connected to A5 of Arduino. Pin 4 to pin 7 are used as data transfer pins which are connected to A1 to A4 respectively. Next anode and cathode pins are connected 5v and ground respectively.

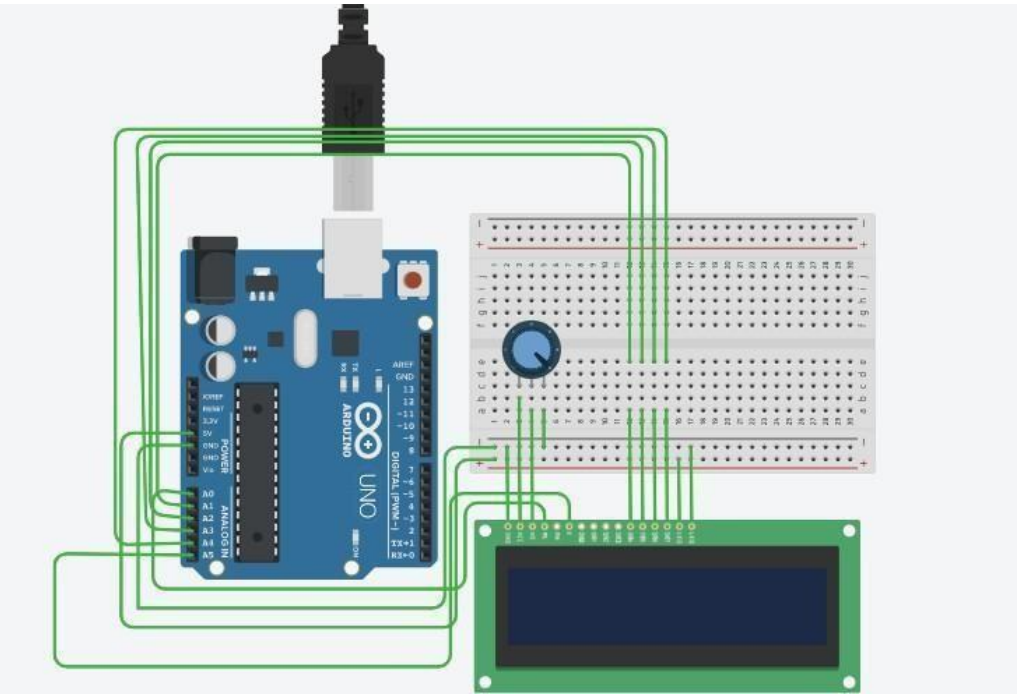


Fig 5.7: Interfacing LCD with Arduino

5.2.2.2 Result of Lcd:

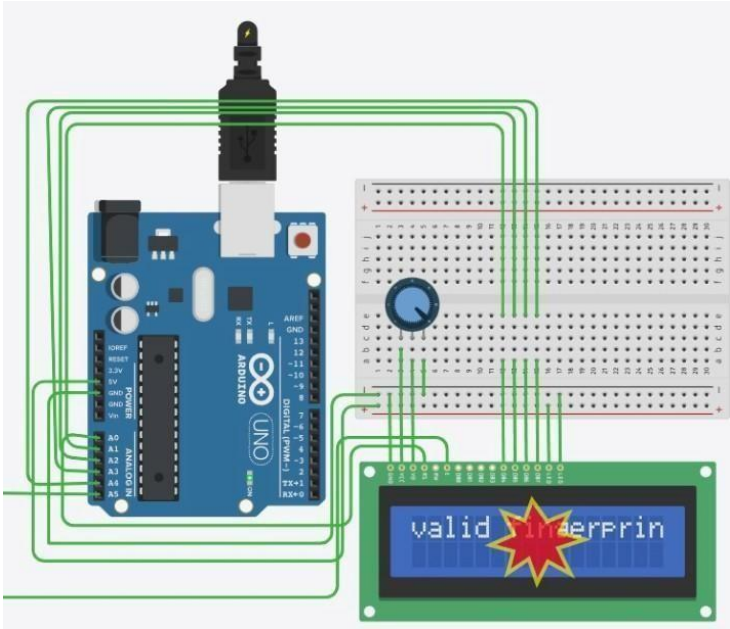


Fig 5.8: Output of lcd

And the output is displayed on Lcd using Tinker cad simulation Software. If a person place his/her finger on the Fingerprint sensor then fingerprint sensor checks the image of the finger with the stored data.If it matches with any stored image then it sends message to LCD display as it is a “valid fingerprint” as shown in the fig.6.3. If it does not match with any stored image then it shows “InvalidFingerprint” on the LCD.

5.2.2 Hardware implementation :

The main aim of this project is to prevent the vehicle from probable theft. To achieve this we are incorporating security by including biometrics, i.e a fingerprint. In the beginning the owner of the vehicle must store his/her own fingerprint in the finger print module. The GSM modem is used to send and receive messages to and from the owner. The owner’s mobile number has to be set fixed during the coding. To start the ignition of the four wheeler one should enter the authorized fingerprint. If anyone enters an unregistered fingerprint, the owner will immediately receive a message. Then real time tracking begins and the GPS location of the vehicle is sent to the owner by SMS. In this proposed project we used GPS module to find the current latitude and longitude of the present location, the GPS

module is UART (Universal Asynchronous Receiver/Transmitter) with a baud rate of 9600 bps. We are using two serial ports. One, for the GSM modem and another one for the GPS modem. The code is written in Arduino IDE was used to program it. It is a fitted device on the automobile. The whole monitoring of entire device is done by the mobile phone which delivers wireless connection among the vehicle tracking system device and the owner. The vehicle tracking device also has a dedicated sim card slot in which a GSM SIM card is inserted in for receiving and sending SMS. The user can send an SMS through his mobile phone, know the location of its vehicle and also the facility to safeguard the vehicle.

During implementation, the Hardware components like GSM, motor, GPS, and RFID are connected to Arduino Uno board and Node MCU using jumper wires. LCD is connected to Node MCU. The connections are made perfectly, Arduino takes input from the fingerprint sensor and then if Authentication of the person is matched the vehicle starts moving.

If the Authentication of a person is mismatched, it sends an SMS as “access denied” to the owner through GSM module. If the owner is aware of a person, he can give access through an SMS reply. In case if the vehicle is theft, owner can track the location of the vehicle using GPS. We can also track the vehicle using RFID technology where the reader is installed at toll gates and tags are placed on vehicles. The registered tag can be detected when the vehicle is passed through the toll gate. The status of vehicle is displayed on the LCD.

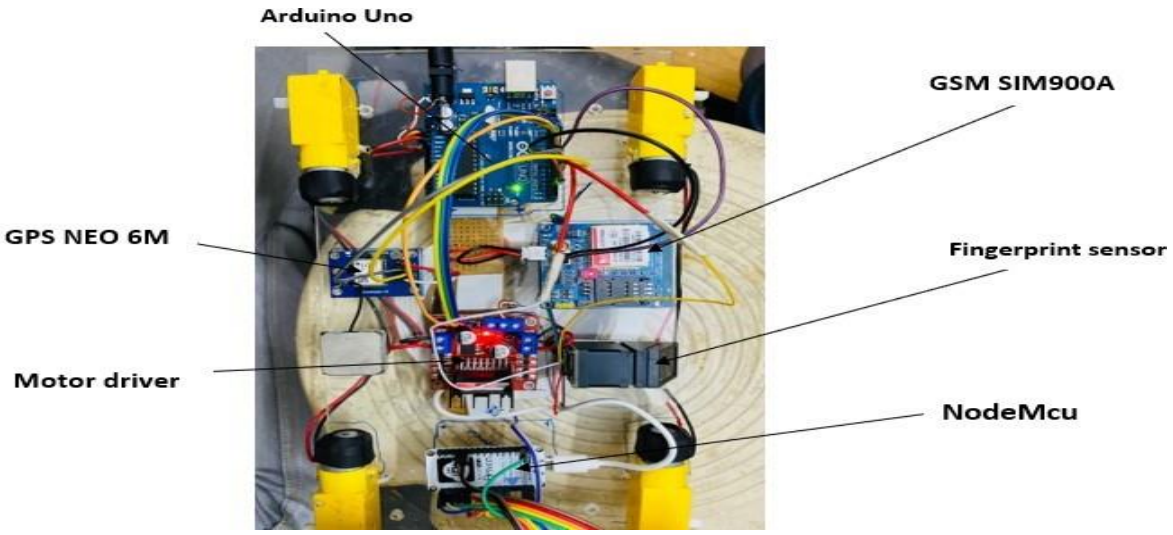


Fig 5.9: Hardware Prototype

5.2.3 DISCUSSIONS:

During the implementation, each and every single part was tested alone the following points have been observed:

Initially, interfacing of individual components to the controllers is done and the outputs are observed separately. And then serial communication is established among all the components. The fingerprint sensor functions correctly and accurately. This is useful to identify the correct authentication of the owner. If the authentication is valid, then Vehicle starts moving. When the authentication is Invalid then it sends an SMS to the owner using the GSM module.

The RFID reader and tag are working correctly. They are useful to trace the location of the Vehicle at the toll gate in case of Theft. We are using GPS in our project for tracking the location of vehicles in terms of longitude and latitude. The output of GPS is sent to the registered mobile number through SMS.

CHAPTER 6

CONCLUSION

In this work we built a hardware model by using Arduino as main component as it is an open-source electronics platform based on easy-to-use hardware and software. It is verified the functionality of developed model that provides authentication and security by using fingerprint sensor to start a vehicle. It can also send a message to the owner if any unauthorized person tries to start the vehicle. Vehicle location can also be tracked by using GPS and RFID.

REFERENCES

1.Vidhyotma; Jaiteg Singh," Bluetooth Enabled Anti-Theft System Using Android Based Handheld Device", 2018 6th Edition of International Conference on Wireless Networks & Embedded Systems (WECON), 16-17 Nov. 2018.

2.Hussain Samad and Md. Careful Design and Implementation of Microcontroller Based Anti-Theft Vehicle Security System using GPS, GSM and RFID International Conference Electrical Engineering and Information & Communication Technology (IEEE ICT 2018),

3.Pranoko Rivandi , Dewanto Satrio , "Automotive Start-Stop Engine Based on Fingerprint Recognition System" , (2018) Web of Conference 130,010222.

4.Aryan Verma,Shikhar Seth," Vehicle Theft Identification and License Authentication Using IoT",Journal of Physics: Conference Series, Volume 164, Advances in Computational Electronics and Communication Engineering Citation Aryan Verma et al 2021 J. Phys.: Conf. Ser. 164 062068 (2019).

5.Akash Khare, Jitendra Yadav; "RFID Based Security System Using Arduino Module", 2016 10th International Conference on Intelligent Systems and Control (ISCO). Volume-3, Issue-5, May-2020.

6.Prabal Deep Das and Sharmila Sengupta, "Proposing the systems to provide protection ofvehicles against theft and accident", 2016 IEEE Conference on Recent Trends in Electronics Information Communication Technology, pp. 1681-1685.

7.Vidhyotma and Jaiteg Singh "Bluetooth Enabled Anti-Theft System Using Android Based Handheld Device"6th Edition of International Conference on Wireless Networks and Embedded Systems,July 2018.

8.Kunal Maurya and Mandeep Singh "A Study of Biometric Approach for Vehicle Security System Using Fingerprint Recognition" International Journal of Advanced Research Trends in Engineering and Technology (IJARTET) Vol. 1, Issue 2, October 2019

9.S.Mohanasundaram (2019); “Vehicle Theft Tracking, Detecting and Locking System Using Open CV”.

10.Mohammad Jahangir Alam, Dhiman Sarma,” A Smart Security Solution to Prevent any Mischief with the Vehicles”, 2021 IEEE Region 10 Symposium (TENSYMP), 23-25 Aug. 2021.

11.D. Anil Prasad, “Vehicle Theft Detection and Secure System” was published in International Journal of Scientific Research in Engineering and Management (IJSREM) Volume 06, Issue 05 May 2022