



Expertise
and insight
for the future

Jaya Ram Khatri Chhetri

Automobile tracking system using GPS and GSM

Metropolia University of Applied Sciences

Bachelor of Engineering

Degree Programme in Information Technology

Bachelor's Thesis

25 May 2020

Author(s)	Jaya Ram Khatri Chhetri
Title	Automobile tracking system using GPS and GSM
Number of Pages	38 pages + 4 appendices
Date	25 May 2020
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Specialization option	Software Engineering
<p>An automobile tracking system is an essential tool for monitoring the vehicle. This project mainly illustrates the development and implementation of an automobile tracking system. The tracking system uses GPS and GSM technology for providing the data of the movement of the vehicle in a real-time. The main three parts of the complete automobile tracking system are a GPS receiver, a microcontroller, and a GSM module.</p> <p>This project covers both hardware and software development parts. Hardware development includes all the necessary wirings with other related physical components like Arduino UNO, SIM808, LCD, power connection etc. and software development covers all the essential coding for microcontroller and GSM message command using Arduino C language.</p> <p>The tracking device provides the opportunity to remotely access the real-time data of the vehicle's position with latitude and longitude at any point on the earth regardless of the weather. Besides, it delivers the google map link to determine the location of the automobile.</p> <p>The anti-theft device is more reliable, cost-efficient, and easy-to-use. It provides security over any unauthorized use of the vehicle.</p>	
Keywords	The Automobile tracking system, GPS module, GSM module, microcontroller, Arduino UNO, SIM808

ABBREVIATIONS

GPS- Global Positioning System

GSM- Global System for Mobile Communication

SMS- Short Message Service

GNSS- Global Navigation Satellite System

AC- Alternating Current

DC- Direct Current

IC- Integrated Circuit

MCU- Microcontroller Unit

LCD- Liquid Crystal Display

RS- Register Select

UTC- Universal Time Coordinated

SIM- Subscriber Identity Module or Subscriber Identification Module

V- Volts

USB- Universal Serial Bus

IDE- Integrated Development Environment

I/O- Input/Output

LEDs- Light-Emitting Diodes

RISC- Reduced Instruction Set Computer

ALU- Arithmetic Logic Unit

PWM- Pulse Width Modulation

UART- Universal Asynchronous Receiver/Transmitter

TTL- Transistor-Transistor Logic

USART- Universal Synchronous Asynchronous Receiver Transmitter

SRAM- Static Random-Access Memory

EEPROM- Electrically Erasable Programmable Read-only Memory

ADC- Analog-to-Digital Converter

LCC- Leadless Chip Carrier

GPRS- General Packet Radio Service

AT- Attention

SMT- Surface-Mount Technology

TTFF- Time-To-First Fix

GPIO- General Purpose Input/Output

IMSI- International Mobile Subscriber Identity

RMC- Recommended Minimum sentence C

Contents

1	Introduction	1
2	Objectives	2
3	Concept	2
3.1	Regulated power supply concept	4
3.2	Circuit Diagram	6
4	Scope	7
5	Arduino, a brief history	8
6	Hardware Requirement	8
6.1	Arduino UNO	8
6.1.1	Shield	9
6.1.2	ATMEGA328P microcontroller of Arduino UNO	10
6.2	SIM808 Module	10
6.2.1	Port and its Function	12
6.2.2	Serial Port (UART) Communication	13
6.3	16X2 LCD Display module	13
6.4	USB Cable	13
6.5	Subscriber Identity Module or Subscriber Identification Module	14
7	NMEA data	14
8	AT COMMANDS	14
9	GSM Technology	15
10	GPS Technology	16
11	Software Requirement	16
11.1	Arduino IDE	16
11.1.1	Arduino Web Editor	17
11.1.2	Desktop IDE	17
11.2	Libraries	18

11.3	Serial Monitor	19
12	Data Stream	19
13	Implementation	20
13.1	Structural and operational design	20
13.2	Components	21
13.2.1	Hardware testing	23
13.3	Serial Connections	24
13.3.1	Serial Testing with Arduino IDE	24
13.4	Software Implementation	26
13.4.1	Software Implementation for SIM808 module	26
13.4.2	Fetch GPS data	26
13.4.3	Data Display of Liquid Crystal Display	27
13.4.4	Sending Message	28
13.4.5	Read message	29
14	Automobile tracking system Implementation by Software	30
15	LCD status	32
16	Final Testing	33
16.1	SMS on GSM phone	33
16.2	Location determination by google map	34
17	Results	35
18	Conclusion	35
	References	37

Appendices

Appendix 1. SIM808 shield test

Appendix 2. Power control through software trigger

Appendix 3. Sending and Receiving the message

1 Introduction

The automobile tracking system is one of the essential devices which locates the current position of the vehicle in real-time using GPS and GSM technology at any location of the earth. The tracking system is an integrated combination of a GPS receiver, a microcontroller and GSM module. The GPS technology, termed as a navigation system based on satellite communication, provides information on time, speed, and location to the GPS receiver. It gives the position of a vehicle in terms of latitude and longitude at any pinpoint on the earth. The device also retrieves the data of the geographic location on Google map of a car. It hence can be adopted anywhere in the earth surface regardless of time and weather condition. GSM technology transmits voice and data services operating on a different frequency band, i.e. 850MHz, 900MHz, 1800MHz and 1900MHz [1]. It is possible to access the GPS Coordinates remotely through the mobile network. The tracking device places into the automobile in a hidden and safe location.

Vehicle security is the most exciting and concerning topic as it always has a significant threat and risk of theft. The priority of the automobile's protection against robbery is more demanding all over the world as it can be traceable in a real-time into the cell phone via the mobile network. It is highly motivated to develop a tracking system using GPS and GSM technologies.

It provides security and acts as a safeguard to all the automobile owners. It is the most reliable, flexible, user-friendly, and cost-efficient device. The device is easy to install.

The usage of the automobile tracking device is for both individuals and commercial purposes. Individuals use the tracking system when their vehicle is missing or stolen. The usage of the tracking system also varies from small scale to big scale companies. The companies can quickly locate the cars and find whether if it has used correctly. It avoids the risk of using autos for any personal purposes without any official notice.

Furthermore, it helps to manage the automobiles in systematic ways. It helps to avoid chaos in the companies. The system widely uses in the field of logistics, rental companies, day-cares, schools and public transportations like buses, trains, boats, ships. The technology system minimizes the illegal activities.

2 Objectives

The objectives of the whole project are to build a GPS and GSM based vehicle tracking system with selective hardware and software and apply it in everyday life for supervising the automobile. The following are a few more objectives of the whole project.

- To enhance the knowledge on the operation of the GPS and GSM module
- To retrieve the vehicle's location information using GPS in a real-time.
- To study the understandings on data transmission using a GSM modem
- To know more about the property and functionality of the hardware chosen
- To build an economical vehicle tracking device

3 Concept

The overall idea of the thesis project is to build the automobile tracking device, as illustrated in the figures below. In this type of tracking system, Global Navigation Satellite System (GNSS) network combines with many satellites to provide data signals to a GPS device. GPS/GSM module, LCD Display and Power supply interface with Arduino to give a complete structure of tracking device. Arduino Uno consists of ATMEGA328P microcontroller (MCU), which is the central processing unit as it controls all other peripheral components according to the instruction given. GPS receiver in tracking device is the main elements since it gets the geographical Coordinates from the satellites in a real UTC with the date, direction, and vehicle speed. GSM modem sends the location of the vehicle in a real-time to the designated phone through a text message on-demand after a microcontroller executes received information from a GPS receiver. Figure 1 is the architecture of an automobile tracking device.

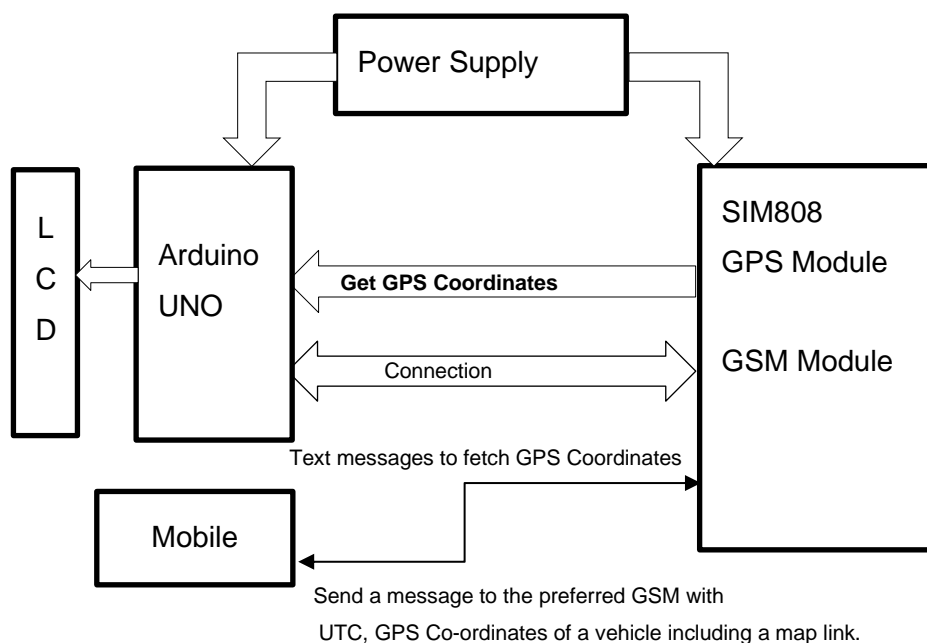


Figure 1. The architecture of an automobile tracking device.

The device fetches GPS information of a vehicle as soon as the automobile owner text message to the tracking devices. 16x2 LCD screen operating in 4-bit mode configuration displays various information that retrieves from Arduino UNO during the execution of the program. GSM module sends the GPS position of the vehicle on a real-time to the owner of the automobile via SMS. However, If the GPS does not get any signal from the satellite, the device sends just an empty coordinate, considers being an error. Figure 2 illustrates the flow chart of the tracking system.

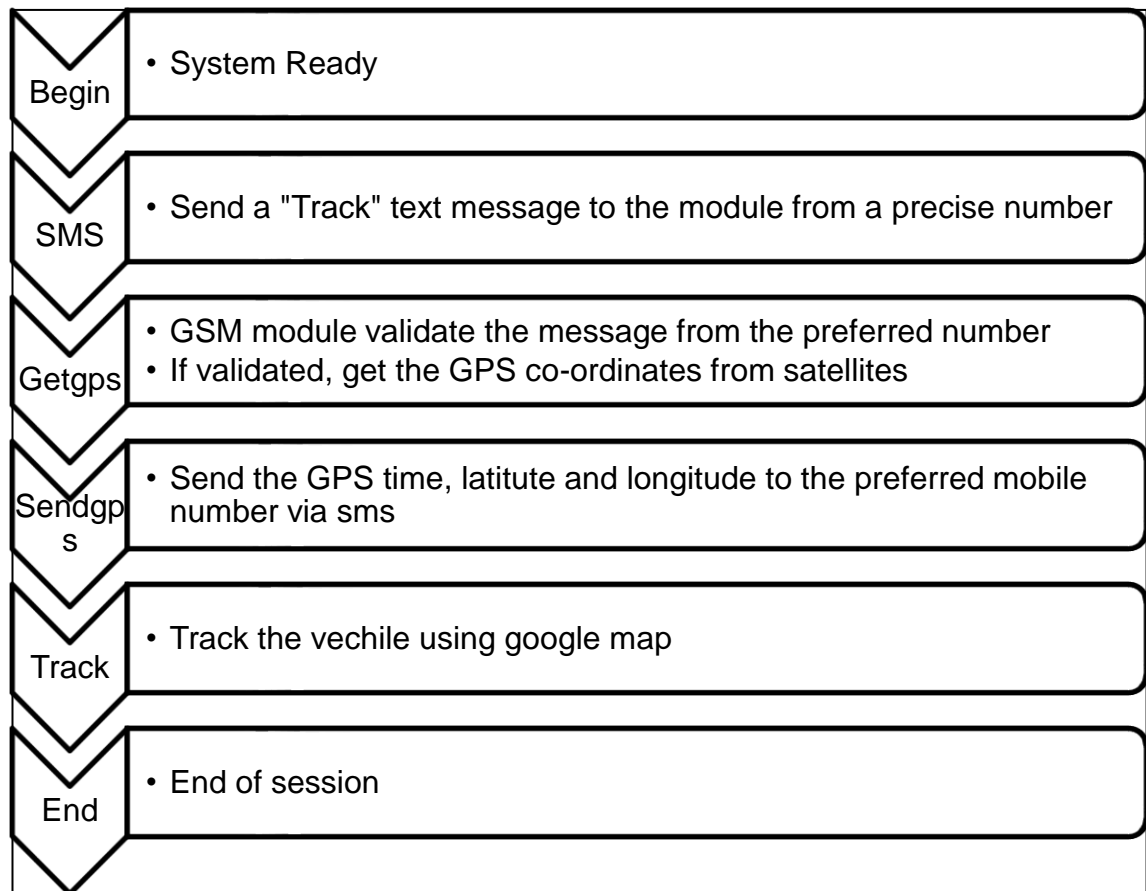


Figure 2. Flowchart of the Automobile Tracking System.

3.1 Regulated power supply concept

The power supply is a crucial part of any electronic component. Unregulated alternate supply damage the whole circuits. Thus, a regulated power supply, which converts AC to steady DC, must apply to electrical units.

Figure 3 presents a sample model of the overall concept of the regulated power supply. Step down transformer decreases a high primary voltage 230V, 50Hz AC mains to a secondary output of 12V, 500mA. Bridge D1 rectifies a 12V transformer output which uses 1N4007 diodes. Then C1, the capacitor filters the output where IC 7805, which is a three-terminal voltage regulator, regulates to generate steady 5V DC. Capacitor C2

filters the remains present in the regulator. R1 limits the current through the power indicator, LED D2.[2]

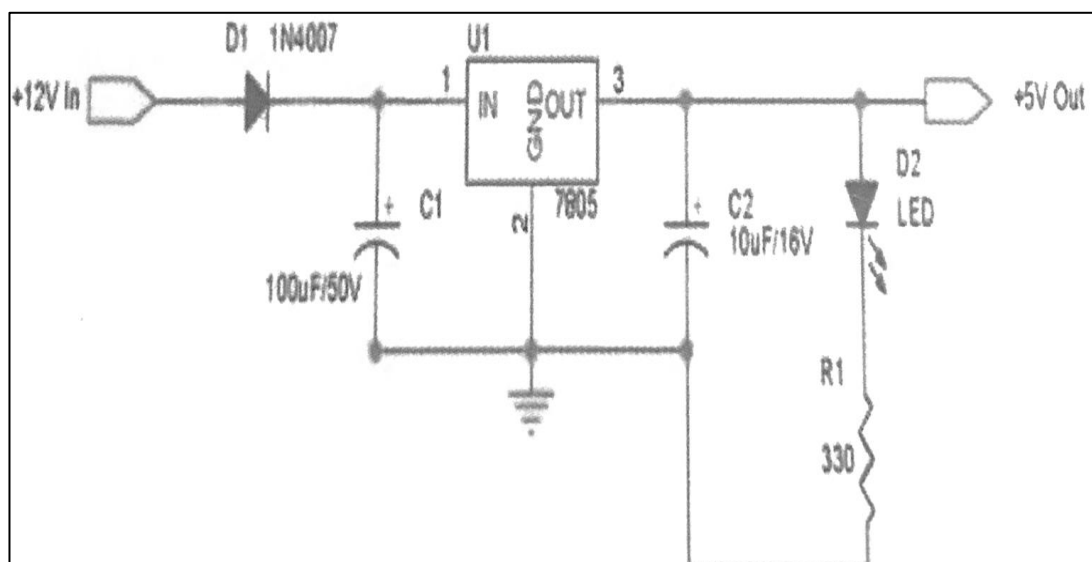


Figure 3. Power Supply [2].

Figure 4 shows the power supply section on the Arduino UNO. The input voltage ranges from 7-12V from DC power jack or VIN pin and 5V from the USB connector. 12V, 1A DC adapter must use if AC power applies to Arduino to run. +5V gets as an output as a regulated power supply.

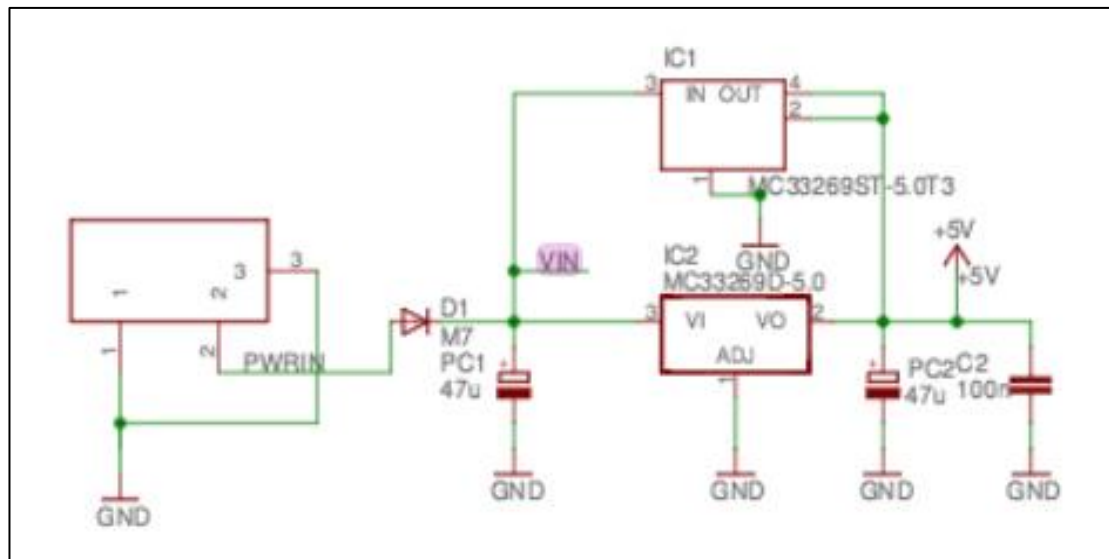


Figure 4. Power supply on the Arduino UNO [3].

3.2 Circuit Diagram

Figure 5 below demonstrates the complete circuit schematic of the automobile tracking system. Arduino UNO, LCD 16x2 and SIM808 shield interface each other. LCD pins RS (PIN 4), Enable (PIN 6), DB4, DB5, DB6 and DB7 wire into Arduino UNO digital I/O pins 12, 11, 5, 4, 3 and 2, respectively. +5V and GND connect across a 10k ohm potentiometer with its wiper, links to VO pin of LCD to adjust the contrast of the display. The display gets appropriate voltage supply via Arduino to Pins VSS (GND) and VDD (+5V). 220ohm resistor connects through the positive terminal of power supply and PIN 15. PIN 16 connects to GND. LED- and LED+ pins display the backlight of LCD.

Digital I/O Pins 7 and 8 create software serial communication between Arduino and the shield. RX and TX pins of SIM808 connects to pins D8 and D7 separately. Arduino IDE's serial monitor baud rate sets at 9600 bits/second. It is essential to set the exact baud rate for successful software serial communication between two modules. SIM808 shield gets power from the Arduino board. Arduino UNO receives an external power supply of 12V.

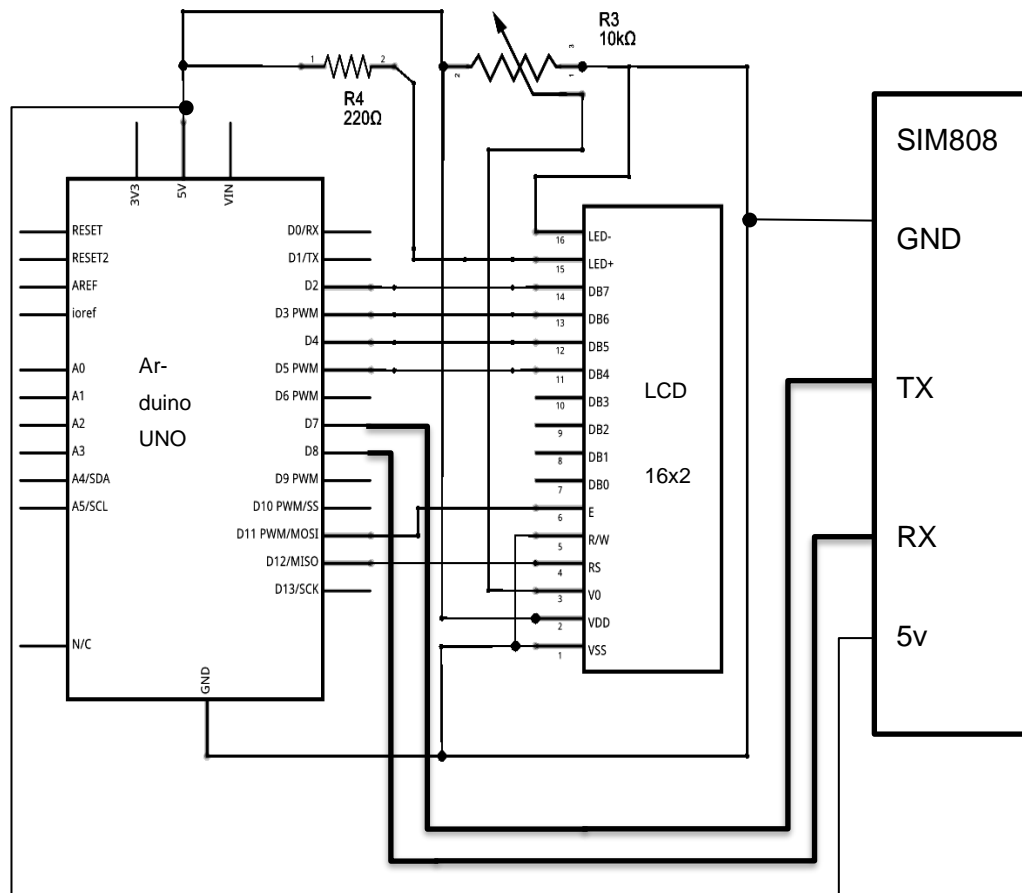


Figure 5. Circuit diagram of the tracking system.

4 Scope

The scope of this project is to study and construct a location tracker for an automobile that uses the adopted GPS and GSM technology. The design of the device allows extending the knowledge on GPS and GSM technology, including SIM interface. It provides a range of information on vehicle position, speed, direction, and date in real-time using GPS receiver. The microcontroller is another significant part to study, which shows how the instruction in it executes with the codes. AT commands control the GSM module.

5 Arduino, a brief history

Massimo Banzi and David Guartilles created and introduced Arduino in 2005 at the Interaction Design Institution Ivera, Italy. It was a simple, inexpensive, easy-to-connect and reliable 8-bit microcontroller of the AVR family programmable device for interactive art design projects. It aimed for the students as a helpful tool for fast prototyping who lacked the knowledge of electronics and programming. David Mellis developed the Arduino Software. [4, p.1] The AVR family microcontrollers possess many unique features which are highly configurable and highly versatile.

There are many popular Arduino boards designed at Arduino.cc such as Arduino Nano, Arduino Mini/Mini Pro, Arduino Mega, Arduino UNO etc. With the help of these popular inexpensive Arduino boards, it is possible to build the different applicable applications, monitoring devices, small robots, automation, performance art and many more these days.

6 Hardware Requirement

6.1 Arduino UNO

Arduino UNO, the first series of USB Arduino board and the reference model of the Arduino platform, is the most robust, used, and documented board of the whole Arduino family [5]. It is the open-source microcontroller small board based on ATMEGA328P microcontroller of AVR family. There are 14 digital I/O pins numbered from 0 to 13. Six digital pins marked with a tilde (~), i.e. pins 3, 5, 6, 9,10, and 11 provide PWM output. It also consists of the power and analogue sockets in which the analogue plugs contain six analogue Input (A0-A5 pins). The UNO board has the USB connector which connects to a computer for either supplying power, uploading the sketch, or transmitting data. The recommended input voltage is 7-12V whereas as the operating voltage is 5V. However, the input voltage limits to 6-20V. The frequency or clock speed is 16 megahertz. A separate power connector is available in the board for external power supply. The appropriate voltage supply to corresponding digital I/O pins reduces the risk to

damage hardware. The reset button of the Arduino UNO restarts the system on any technical issue upon pressing on it. Figure 6 shows the Arduino UNO board.

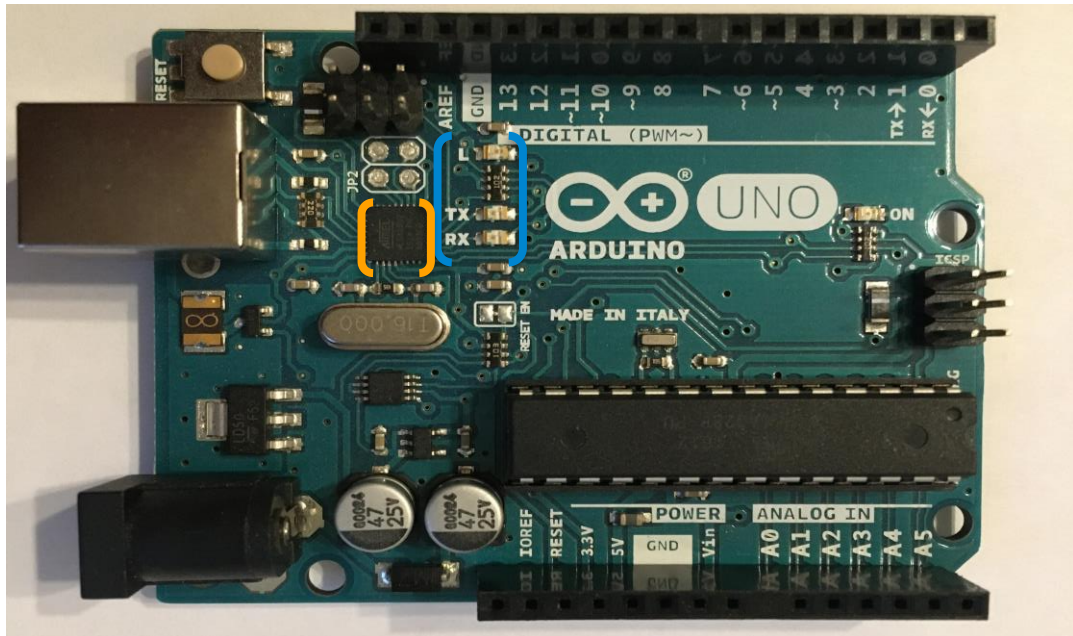


Figure 6. An Arduino UNO board.

From the above picture, LEDs labelled with TX and RX inside the blue bracket light up at the time of receiving and sending data via the serial port or USB between the Arduino and other attached devices. LED L connects to digital PIN 13 for own use. The small microcontroller inside the orange bracket controls over the USB interface between Arduino and the computer during data transmission. [6, p.22-23]

Arduino UNO has one serial port, UART for serial communication to communicate with the pc via USB or another external serial device. PIN 0 (RX) receives data, and PIN 1 (TX) transmits data. The microcontroller ATmega328P uses 5V TTL serial communication.

6.1.1 Shield

A shield is a circuit board that plugs on top of the Arduino PCB via pins to extend its capabilities and functionality [7]. There are many shields available like ethernet shield, GPS receiver shield, MicroSD shield, SIM808 Shield etc.

Arduino shield is stack in design. It allows connecting with the external board or another device through the sockets on its sides. [8, p.162]

6.1.2 ATMEGA328P microcontroller of Arduino UNO

ATMEGA328P is a high-performance 8-bit microcontroller of AVR family with low power consumption based on RISC architecture which has 32 general-purpose working registers connected to the arithmetic logic unit (ALU) directly. The microchip contains 23 general-purpose programmable digital I/O lines and three significant amounts of memory.

- Flash memory - 32 K bytes (non-volatile memory)
- SRAM – 2K bytes (volatile memory)
- EEPROM – 1K byte (non-volatile memory)

Two 8-bit and one 16-bit timer/counter, six PWM channels, real-time counter with a separate oscillator, programmable serial USART and watchdog timer, two-wire serial interface, a six- or eight-channel 10-bit ADC etc. are some of the essential features of ATMEGA328P microchip. The operating voltage for the ATMEGA328P type of microcontroller ranges from 2.7V to 5.5V other than 1.8V – 5.5V [9]. Figure 7 shows the ATMEGA328P chips.

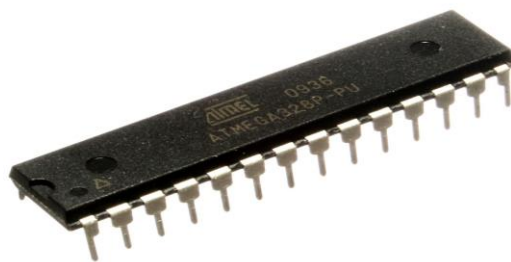


Figure 7. Snapshot of ATMEGA328P microcontroller.

6.2 SIM808 Module

SIM808 module is a compact design developed by SIMCOM. It is the integration of GPRS and GPS in an LCC package reducing time, energy and money for anyone to

establish GPS based applications.[10] SIM808 is a combination of GSM/GPRS, GPS and BT module which supports Quad-band frequencies of 850/900/1800/1900MHz and merges with GPS technology to get satellite navigation. AT commands control SIM808 and support 3.3V and 5V. [11] Figure 8 shows the SIM808 module.

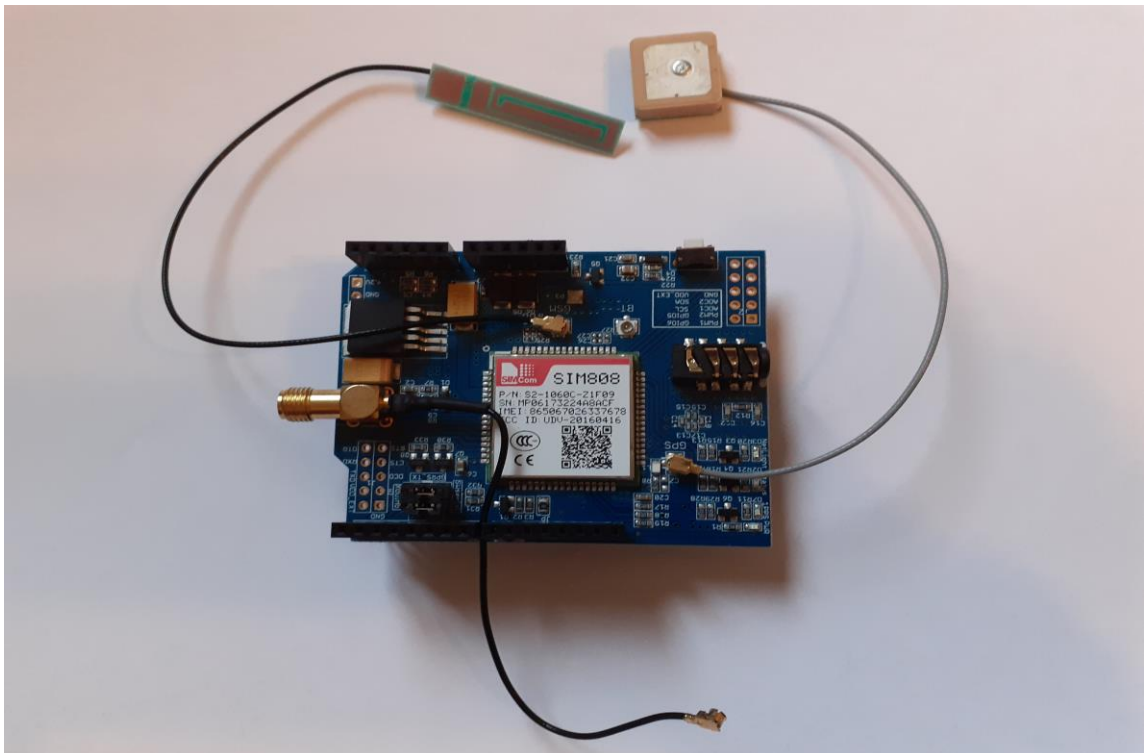


Figure 8. SIM808 Module.

GPRS module SIM808 is compatible with Arduino UNO, that delivers and receives GPRS data, Voice and SMS messages. It gets GPS along with A-GPS data. [11]

SIM808 has many different features as listed below.

- SIM808 powered through Arduino UNO
- One SIM Card interface and holder of standard size
- GPRS mobile station class B.
- Operated via AT commands
- 68 pins SMT pads
- Time-to-first fix (TTFF) and accuracy
- UART interface
- Reduce power consumption

- Support various functions like GPS, NMEA protocol, BT, Real-time Clock and 3V-5V logic level
- Supply Voltage 5V ~ 12V
- GSM, Bluetooth, and GPS Antenna pad
- USB interface
- GPIO. [12]

6.2.1 Port and its Function

SIM808 interface function shows the connection, performance, and status of the relative hardware. Bluetooth, GSM and GPM port connect with an individual outer antenna. 5-12V direct current supplies via Vin. Serial select port gives the possibility to choose hardware serial or software serial select for connecting GPS and GSM shield with Arduino. The different LED lights show the different status of the device.

- PWR- notifies the device is on
- 1PPS- tells about the GPS condition.
- Status- shows the status of power on
- Net light- shows the status of network connection to SIM808 module.

The power key is for switching the device on or off.[13]

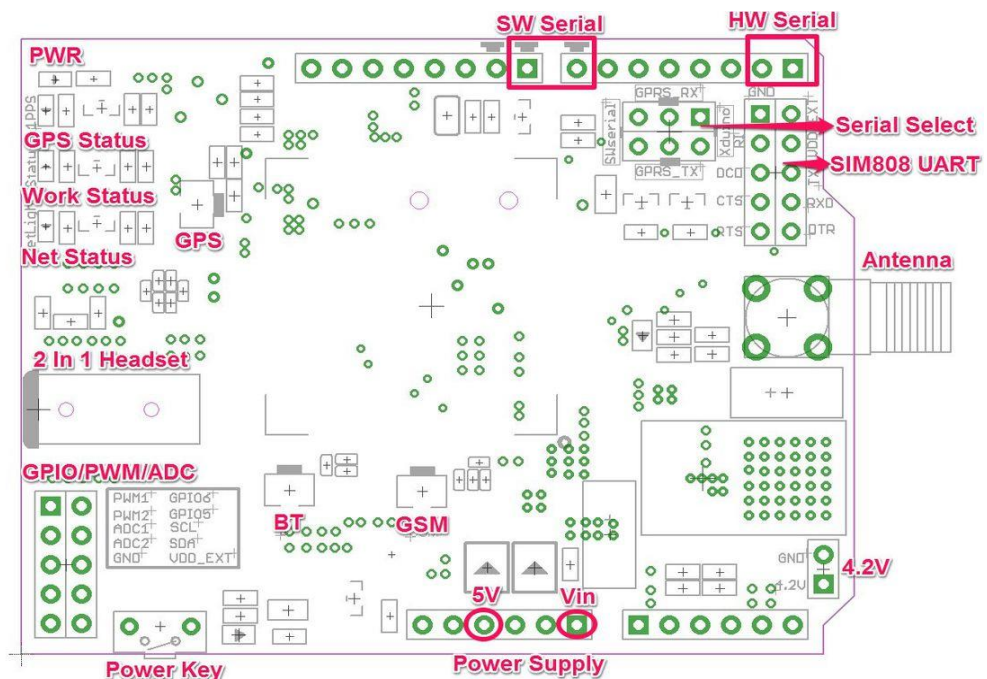


Figure 9. Interface function [13].

6.2.2 Serial Port (UART) Communication

The SIM808 UART joins with select serial ports to interface either with software serial or hardware serial communications. Pin7 and pin8 of Arduino connect with GPRS RX and TX shield for Software serial communication separately. Pin0 and pin1 pins of Arduino connects for hardware serial communication.

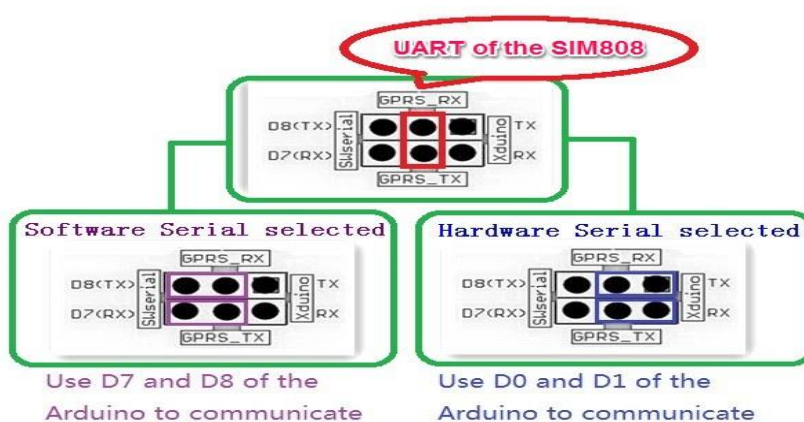


Figure 10. Serial Port Connection. [13]

6.3 16X2 LCD Display module

16x2 Liquid Crystal Display is alphanumeric that has an 8-bit and 4-bit mode. It has 16 columns, two rows and 32 characters with 1280 pixels. It has HD44780U dot-matrix small LCD controller which gets the instruction from the microcontroller unit and execute it to provide information on the display. There are 16 characters in each row, and each character has a 40-pixel box. The operating voltage is 4.7V to 5.3V with 1mA current, not including the backlight.[14]

6.4 USB Cable

Universal Serial Bus connects different components to the computer to exchange data between them. The USB use standard interface to communicate with other devices. Different types of USB are available.

6.5 Subscriber Identity Module or Subscriber Identification Module

Subscriber Identity Module (SIM), is a detachable smart plastic card that stores user's information including network, ID, data plan, phone number and country code. It has a specific memory to store contacts and messages. It is for cellular phones, mobile computers, tablets and so on. It uses IMSI 64-bit number to recognize every subscriber on a cellular network distinctively. SIM is transferable to any GSM mobiles. It uses a four-digit PIN to activate on the net. Standard, Micro and Nano are different sizes of SIM card available in the market. GSM requires a SIM card.

7 NMEA data

NMEA, National Marine Electronics Association, is a US-based non-profit organization to improve the technology and security of marine electronics. It provides a standard file extension format for all electronics protocol industries for electrical interface and data communication.

NMEA data parse into a sentence in ASCII characters that uses a serial communication protocol, including carriage return (CF) and line feed (LF). NMEA 0183 electrical interface uses 4800 baud rate, 8 bits of data in total, zero parity including stop bits one or multiple. NMEA sentence always begins with a "\$" sign and finished with a carriage return and line feed. [15]

8 AT COMMANDS

AT commands, also known as Hayes Smartmodem commands (by Hayes Microcomputer Products, Inc.), controls any popular modems through the series of instructions for various operations. [16] AT stands for attention. The AT command performs different actions on supporting modems like dialling phone, sending SMS, powering module, inquiring the quality of GSM, fax, information and configuration on SIM, module and so on. The command line starts with the prefix AT which tells modem about the beginning

of the instruction or code. The device uses those commands to communicate with computers.

The table 1 below shows a few required samples AT command and their description used in the project.

AT Command	Description
AT	Match baud rate/Check interface
ATI	Recognition of shield
AT+CCID	SIM confirmation
AT+CREG	Check network registration
AT+CSQ	Provide signal quality
AT+CMGF	Text message format
AT+CNMI	To specify how to handle the newly arrived message
AT+CMGS	Send message
AT+CMGR	Read message
AT+CGNSTST	Send GNSS data to AT UART
AT+CGNSPWR	Power control of GNSS
AT+CGNSSEQ	Define the last NMEA sentences that parsed
AT+CGNSINF	Read NMEA sentences

Table 1. AT command and their description.[17]

9 GSM Technology

GSM, a short form of Global System for Mobile Communication, is a cellular network that operates in various frequency ranges, i.e. 850MHz, 900MHz, 1800MHz and 1900MHz for transmitting data and broadly used by mobile phone customers in Europe and Asia including other parts of the world. The technology is accepted globally for digital cellular communication.

10 GPS Technology

GPS (Global Positioning System) is a navigation system based on the network of 24 satellites initially and positioned into trajectory or orbit. The technology was adopted by US military force (American Department of Defence, DOD) and known as the creator of the GPS. The satellites send signals or data to the GPS receiver on the earth to find the position, time, speed, and direction on any point on the planet. GPS works without any subscription fee at any situation regardless of weather condition everywhere in the world at any time. The system is worldwide accepted by aviation, navy, civilians, land survey and many more.

11 Software Requirement

11.1 Arduino IDE

Arduino IDE (Integrated Development Environment) introduced by Arduino.cc is open-source development software, uses for writing and compiling programs then uploading them to any Arduino shields. Arduino IDE software runs in a different operating system like Windows, Linux, and MAC that supports both C and C++ programming languages. Arduino IDE is available online and offline.

Arduino IDE has two main parts, setup () and loop () parts. Setup () function runs once the power is on or pressing the reset button. The function uses to initialize pin modes, setting the initial values, variables and so on. The opening brace ({) and closing brace (}) in a setup function tells what the program executes as a result. Loop () part uses for looping purposes that allows the application to change and react controlling the Arduino Board uninterruptedly until the power is off or presses the reset button.

11.1.1 Arduino Web Editor

Arduino Web Editor is an online open-source software development platform that allows to write programs and upload them to any genuine Arduino devices. Online IDE allows to backup and saves the sketches into the cloud. Therefore, it opens from all possible devices after a successful login. The latest features of this IDE support for all genuine Arduino Board. Figure 11 shows the online IDE.

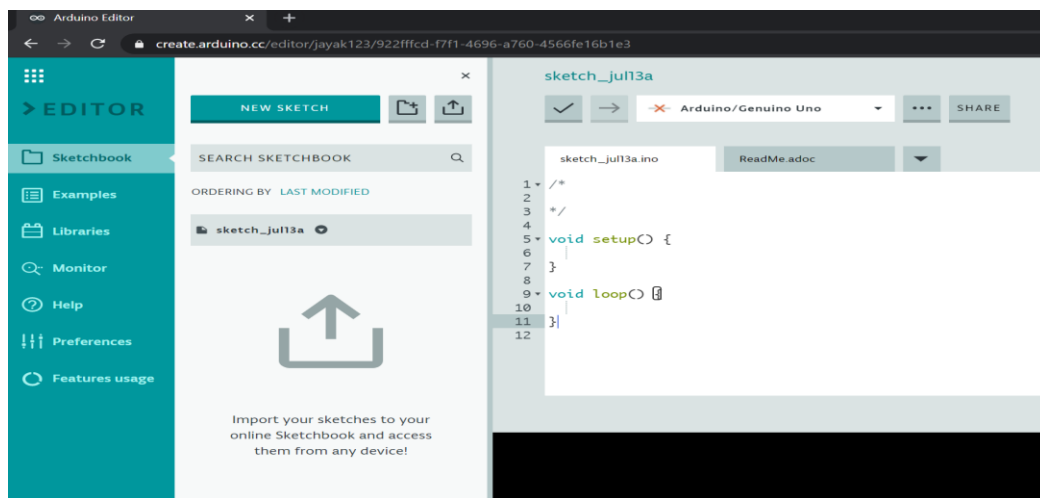


Figure 11. Screenshot of Arduino Web Editor Integrated Development Environment.

11.1.2 Desktop IDE

Desktop Integrated Development Environment (IDE) is cross-platform. It runs on a different operating system. The software IDE works well on Windows, Macintosh OSX and Linux operating system after successful installation. The proper guidelines need to follow for successful installation of the software. Figure 12 shows the Desktop IDE for windows operating system.

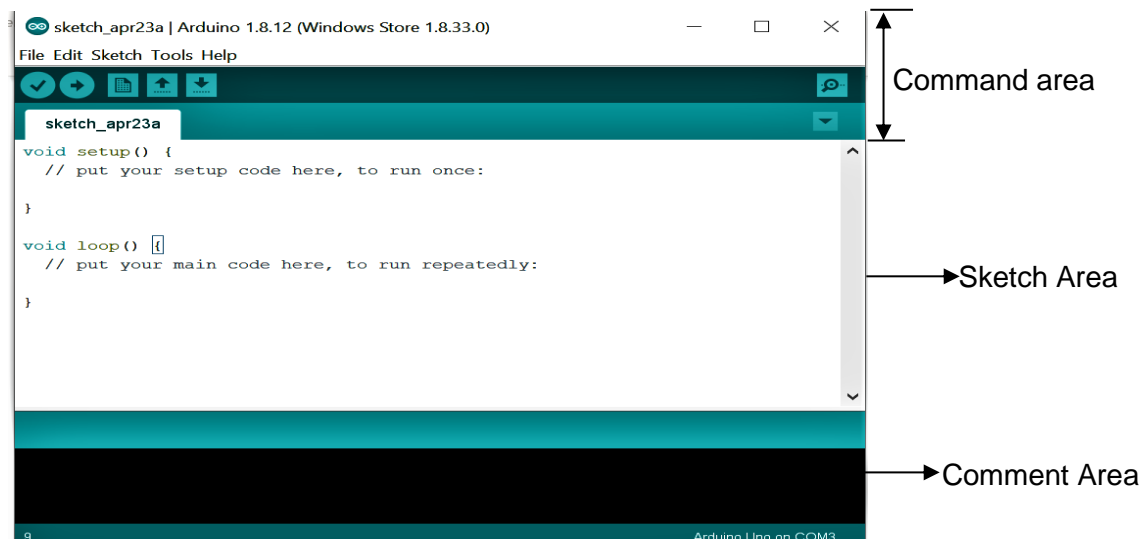


Figure 12. Screenshot of Desktop Arduino Integrated Development Environment.

11.2 Libraries

Libraries are a vast collection of functions precompiled into a library file. They are the header files and marked a filename as .h at the end. The programmer can extract and put the code from the library to call any additional functionality in their system. Arduino IDE includes different standard libraries that are available to sketch the program. Some of the essential libraries supplied are LiquidCrystal, EEPROM, Ethernet, SoftwareSerial and so on. LiquidCrystal libraries have all the required functions needed for liquid crystal displays. There are various libraries available provided by the hardware vendor, supplier or another developer. It is also possible to create and add your library. In the Arduino IDE, anyone can insert library by just selecting sketch > Import Library. [18, p.277-290]

SoftwareSerial Library contains all the operational instructions for serial communication between Arduino and other board.

SIM808 library includes various features of SIM808 GPS and GPRS module. It allows accessing the property of SIM808 like sending SMS, getting geo-location, sending GET and POST HTTP(s) requests etc. The library applies 64 bytes buffer to transmit data

with a GPS-GSM module and consume less memory. SIM808 parse response confirms that the executed instructions from the module are accurate.[19]

11.3 Serial Monitor

Arduino IDE has a separate pop-up window called serial monitor, in which the computer interface with Arduino to communicate with GPRS/GPS/GSM module. The serial monitor interacts by sending and receiving serial data. It shows the status of the component. The serial monitor makes it easier to debug Arduino sketches.

12 Data Stream

Data Stream protocol is the regular generation of data in real-time at a regular interval in a wide variety and velocity from various sources. It is necessary to understand how the data streams between PC, Arduino and GPRS module. Arduino IDE installs into a laptop and connects via hardware UART to Arduino UNO from USB to establish a communication. GPRS module links via software UART with Arduino through jumper wires by a cross-connection to receive and transmit data through the terminals. The baud rate is 19200 bps at 8-N-1 for synchronization.

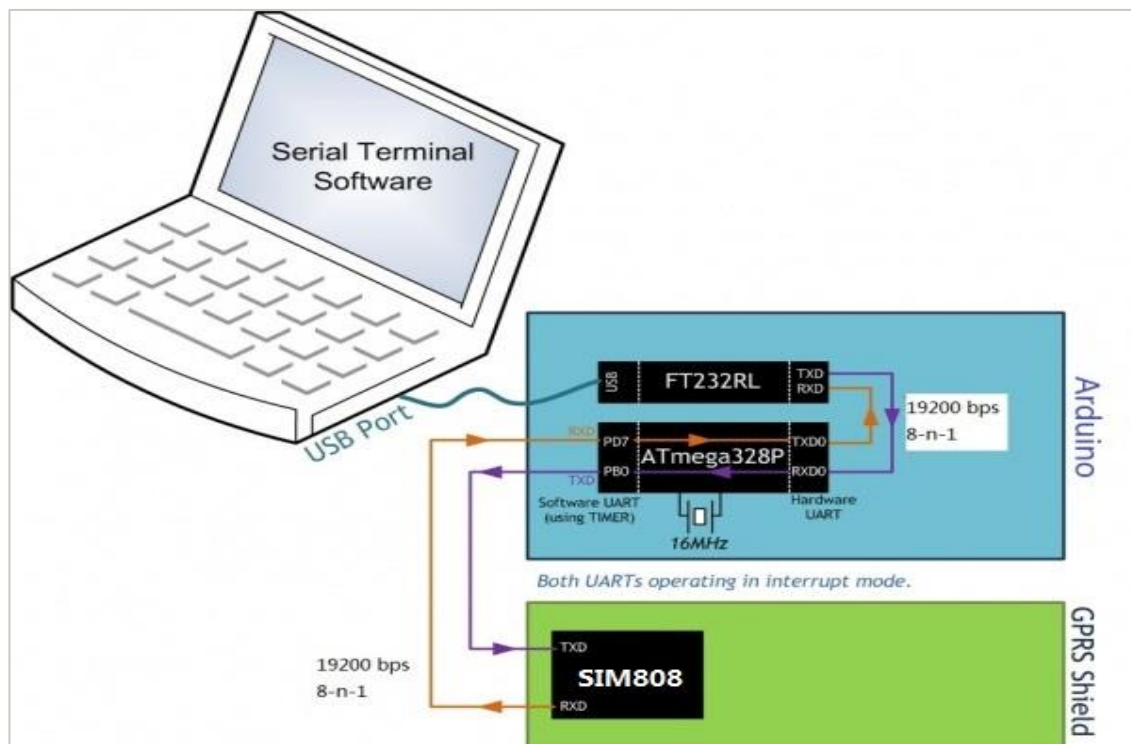


Figure 13. Data Stream between PC, Arduino, and Shield [20].

13 Implementation

13.1 Structural and operational design

The implementation of the whole project started by giving a concrete design to the automobile tracking system using a Global Positioning System and Short Message Service concept. In the initial phase, the model was drafted and outlined the operational structure.

Figure 14 showed the operational block diagram of a GPS and GSM based tracking module, which revealed the functional overview of the system. 5V regulated power supply applied to every unit in the system. GPS receiver received the signal from the satellite in the form of latitude and longitudes of the vehicle. The retrieved data would show

into the display. The accumulated data would execute in the microcontroller and processed to the GSM modem that contained the SIM card. GSM modem would then forward the data to mobile once it got a command from the phone.

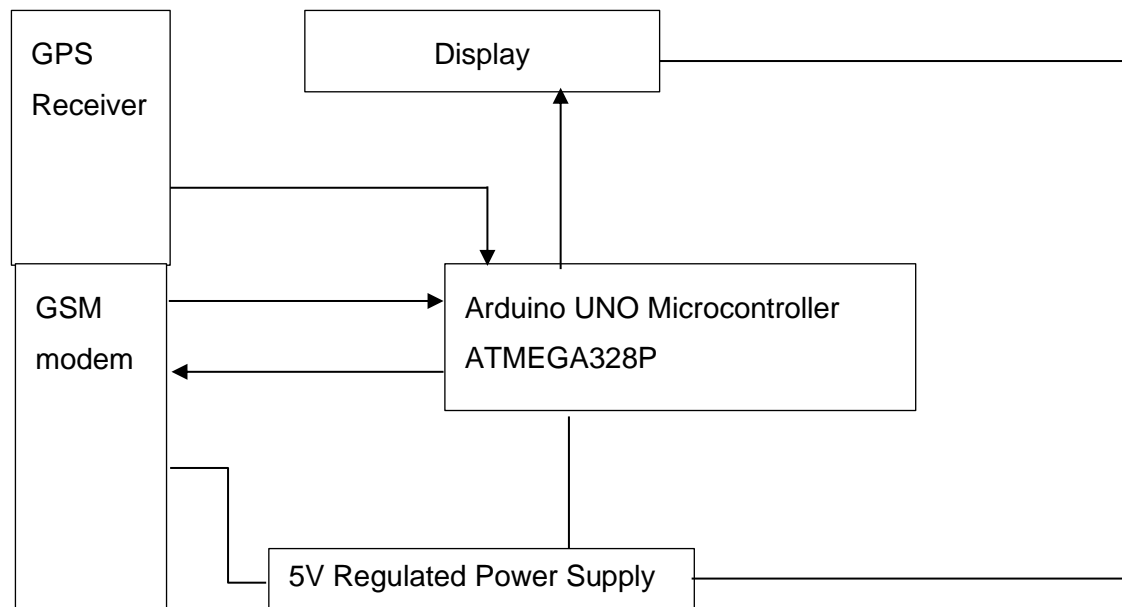


Figure 14. Operational block diagram of GPS and GSM based tracking module.

13.2 Components

The setup began with assembling the hardware parts used in the project. The components used in building the device are listed below in Table 2.

Quantity	Components
1	Arduino Uno
1	SIM808 module
1	Liquid Crystal Display(16X2)
2	USB
1	9V Battery and 3V Lithium battery (Optional)

1	GPS Antenna
1	GSM Antenna
1	SIM card
18 pieces male to male	Jumper wires

Table 2. Components Required

The essential step of the project was to install all the parts relating to each other. Arduino UNO connected to LCD through jumper wires with the help of their circuit diagram studied from "Arduino Projects Book" that came with Arduino starter kit as a manual. The supplied voltage to the display was 5V provided from Arduino.

Serial port selection in SIM808 was essential to configure the communication port whether the connection was a software serial or hardware serial. SIM808 and Arduino UNO connected through jumper wires into their RX and TX terminals. SIM808_TX and SIM808_RX pins of the module plugged into D7(RX) and D8(TX) pins of Arduino UNO separately. The noticeable part was the stackable design of the shield that gave an option to plug into the Arduino UNO.

A six-pin holder valid unlocked SIM card inserted into SIM808 module which automatically detects the required voltage. But as an option, a 3V CR1220 Lithium battery inserted into the battery holder at the backside of the shield. Powering of SIM through this battery would provide the internal Real-Time Clock. The GPS and GPRS antenna connection to SIM808 was necessary to get the signals from the towers.

The USB linked Laptop, and Arduino UNO provided power supply to Arduino beside data transformation. SIM808 received the voltage of 5V another USB port separately. Arduino powered up the LCD supplying 5V energy.

Figure 11. shows the necessary parts used for constructing a device.

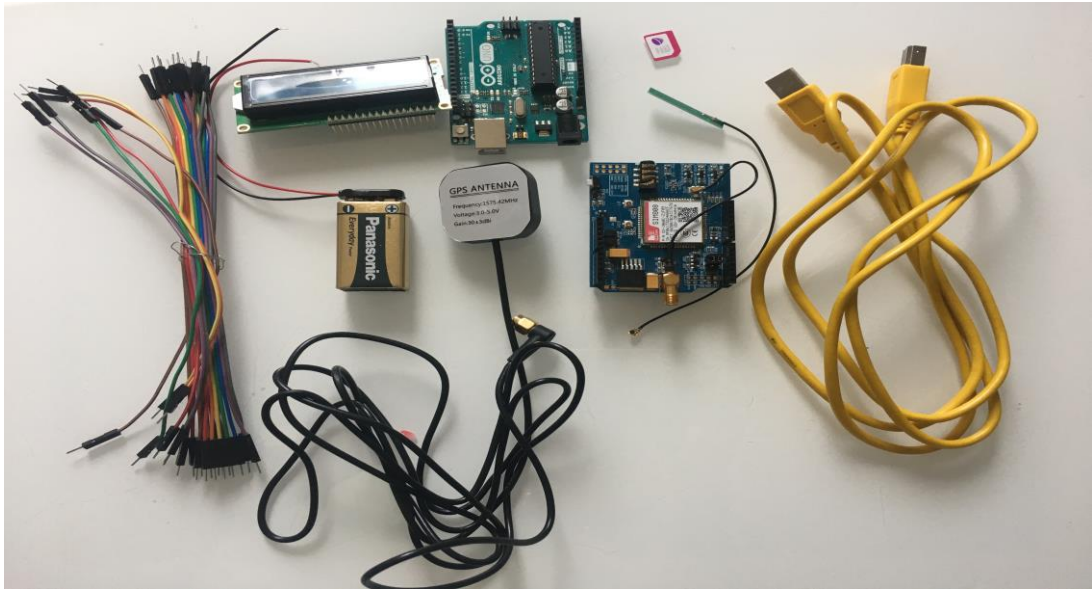


Figure 11. Components

13.2.1 Hardware testing

The next phase was to establish a hardware test to check whether all the assembled elements configured correctly. Two LED lights, LED ON and LED L blinked on Arduino UNO after a 9V source applied along with successful power-up on a display screen. Red LED in the modem flashed on after it got the voltage. Work and NET status LED light switched on when the power key triggered for 2 seconds. The GPS status LED light (blue) started functioning after the modem connected to the satellite signal. The blinking nature of the network led provided the following information for visual confirmation.

- 64ms on and 800ms off- the network was not connected
- 64ms on, 3 seconds off-cellular connection was successful and could send/receive voice and SMS
- 64ms on, 300ms off - the GPRS data connection was working.[21]

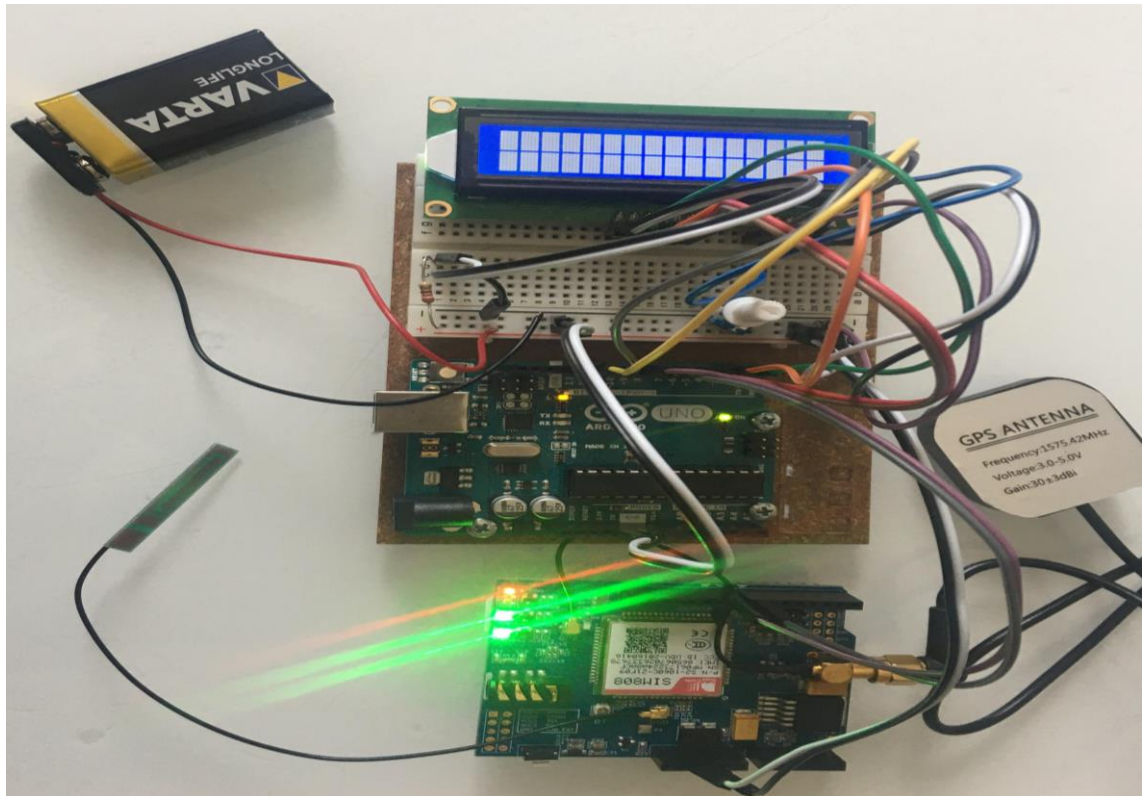


Figure 12. showed the connection status

13.3 Serial Connections

Software serial port selection from jumper cap was a crucial part of serial communication. SIM808 hooked up to Arduino through software UART connecting the RX and TX terminal of the shield to D8 and D7 of the board, so-called a cross-linked between ports.

13.3.1 Serial Testing with Arduino IDE

A serial monitor from Arduino Desktop IDE popped up once the connection was confirmed, and the sketch was uploaded to a device through USB to ATMEGA328P microcontroller. The shield remained shut down before uploading the program. Arduino UNO and SIM808 connected through jumper wires and monitored the status of the system, as shown in Figure 10. AT commands were sent to check the various conditions.

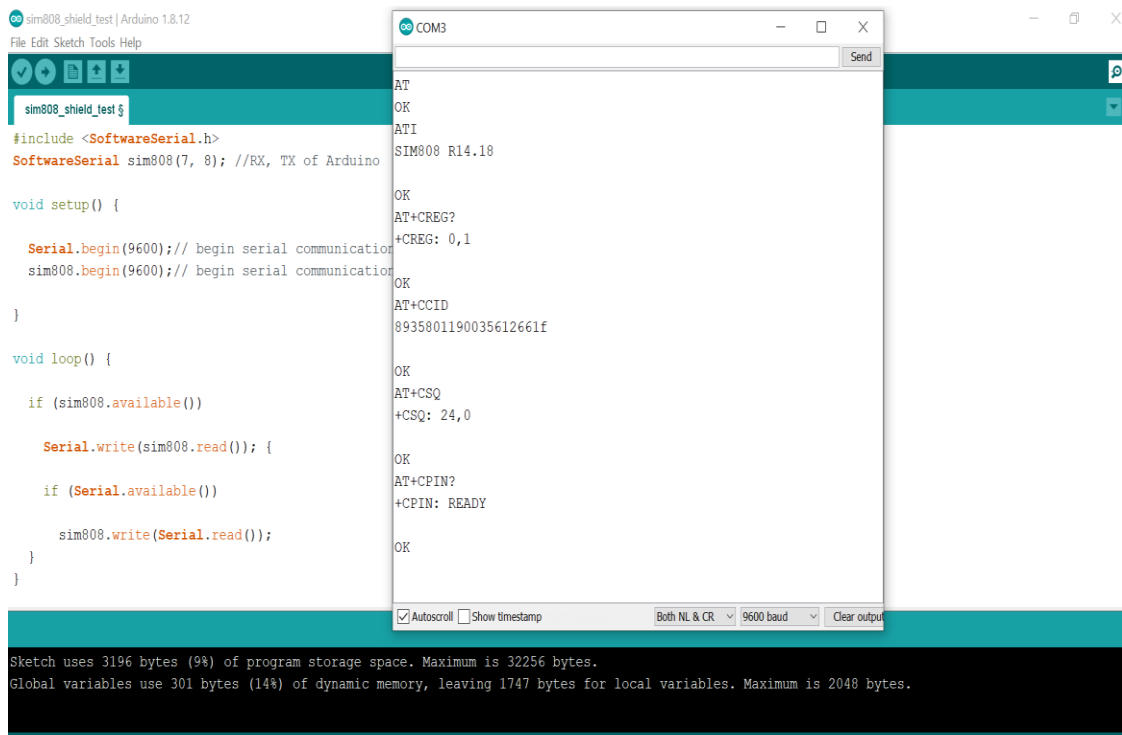


Figure 10. Screenshot of the Serial Monitor Test

Appendix 1 showed how the initial serial testing was adopted.

NMEA message format like RMC, GGA, GSA, GLL and VTG studied during the project research. RMC message format chosen while writing the programming language. The format of the RMC (Recommended Minimum sentence C) message is as shown below. \$GPRMC,hhhmss,Status,Latitude,N,Longitude,E,SOG,COG,ddmmy,MV,MVE,Mode *CS<CR>>LF>

In the example below the parse NMEA data sentence is in RMC. The sentence has a specific meaning explained under the GPS PVT (position, velocity, time) data.

\$GPRMC,184729.000, A,6017.9927, N,02503.8122, E,0.21,316.38, 190520, A*64

Where:

RMC	Recommended Minimum sentence C
184729.000	Fix taken at 18:47:29 UTC
A	Data Active, V- Data not valid
6017.9927 N	Latitude 60 deg 17.9927' N
02503.8122 E	Longitude 25 deg 3.8122' E

0.21	Speed over the ground in knots
316.38	Track angle in degrees True
190520	Date, 19 May 2020
A*64	The checksum data.[22]

13.4 Software Implementation

13.4.1 Software Implementation for SIM808 module

The GPRS/GPS module power triggered through software. Sim808 has a specified pin for the switch. Pin 9 assigned to write the corresponding program to control ON and OFF, as shown in Appendix 2.

SIM808 module contained libraries that defined software serial communication on pins D7 (RX to TX connection) and D8 (TX to RX connection) of Arduino UNO. The module configured with the name `sim808` in the software serial constructor where digital I/O pins were assigned to communicate with GPS and GPRS module. The setup part was most important, where the baud rate set as 9600 for both Serial and `sim808`. `Serial.begin()` started serial communication with Arduino and software monitor that popped up when the code uploaded to Arduino IDE for monitoring. The output observed after selecting "Both NL & CR" at 9600 baud rates. The function `sim808.begin()` started serial communication with Arduino and the shield. The delay with 150 milliseconds imposed. The `loop()` included the main program that decided how to control the device over and over.

The `sim808.available()` checked the presence of data in the pins 7 and 8. The module would return a-1 if no data were detected.

The `sim808.read()` read the returning data from the serial port.

13.4.2 Fetch GPS data

The SIM808 software UART provided the GPS signal that retrieved from the receiver through GPS antenna and sent to the ATMEGA328P. All the necessary data grabbed through AT commands.

```

void setup() {

gps_init("AT+CGNSPWR=1", 2000, DEBUG); //power control of GNSS

    delay(200);

    gps_init("AT+CGNSSEQ=RMC", 2000, DEBUG); //define the last NMEA that parsed
in RMC message format

    delay(200);

}

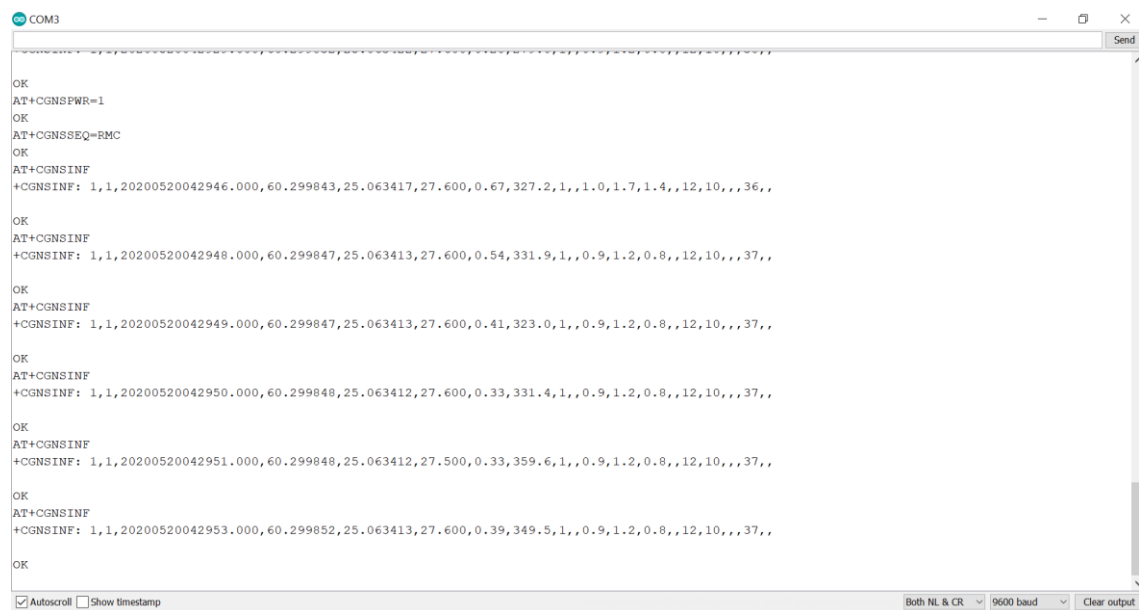
Void loop() {

send_Gps_Data("AT+CGNSINF", 2000, DEBUG); // read GNSS data

}

```

Figure 11 shows the return single RMC data received from sim808 to the microcontroller.



The screenshot shows a serial monitor window titled 'COM3' with a 'Send' button in the top right corner. The window displays the following text:

```

OK
AT+CGNSPWR=1
OK
AT+CGNSSEQ=RMC
OK
AT+CGNSINF
+CGNSINF: 1,1,20200520042946.000,60.299843,25.063417,27.600,0.67,327.2,1,,1.0,1.7,1.4,,12,10,,,36,,
OK
AT+CGNSINF
+CGNSINF: 1,1,20200520042948.000,60.299847,25.063413,27.600,0.54,331.9,1,,0.9,1.2,0.8,,12,10,,,37,,
OK
AT+CGNSINF
+CGNSINF: 1,1,20200520042949.000,60.299847,25.063413,27.600,0.41,323.0,1,,0.9,1.2,0.8,,12,10,,,37,,
OK
AT+CGNSINF
+CGNSINF: 1,1,20200520042950.000,60.299848,25.063412,27.600,0.33,331.4,1,,0.9,1.2,0.8,,12,10,,,37,,
OK
AT+CGNSINF
+CGNSINF: 1,1,20200520042951.000,60.299848,25.063412,27.500,0.33,359.6,1,,0.9,1.2,0.8,,12,10,,,37,,
OK
AT+CGNSINF
+CGNSINF: 1,1,20200520042953.000,60.299852,25.063413,27.600,0.39,349.5,1,,0.9,1.2,0.8,,12,10,,,37,,
OK

```

At the bottom of the window, there are checkboxes for 'Autoscroll' (checked) and 'Show timestamp' (unchecked). On the right side, there are dropdown menus for 'Both NL & CR', '9600 baud', and a 'Clear output' button.

Figure 11. Current RMC location data in the serial monitor

13.4.3 Data Display of Liquid Crystal Display

The two initial lines built in the programming above the void setup() whose purpose was to include the library for the display.

```
#include <LiquidCrystal.h>
//LiquidCrystal lcd (12, 11, 5, 4, 3, 2); // pins for RS, E, DB4, DB5, DB6, DB7
```

The `lcd.begin(16, 2)` in the sketch in the void `setup()` alerted Arduino that the display got 2 rows and 16 characters in each row. The `lcd.clear()` cleared the screen whereas `lcd.setCursor(0,0)` or `lcd.setCursor(0,1)` function positioned and located the data.

13.4.4 Sending Message

In this project, Arduino UNO programmed to send a text message to the specified number ("my number= +358xxxxxxxx") followed by the preferred text message ("Track"). The module did not response anything until and unless the SMS and phone got validated. The functionality of the AT commands for sending message were studied and implemented.

`AT+CMGF=1` -This AT command set SMS message into text.

`AT+CMGS=" +358000000000"` – The module sent the text message to a number specified.

Figure 12 showed the message sent in the serial monitor.

```
COM3
AT+CMGF=1
OK
AT+CNMI=1,2,0,0,0
OK
AT+CMGD= 1,4
OK
AT+CGNSPWR=1
OK
AT+CGNSEQ=RMC
OK
Sending Message
AT+CMGF=1
AT+CMGS="+358000000000"
Location successfully retrived at UTC time 20200520084131.000
Latitude :60.299950
Longitude :25.06382
https://www.google.com/maps/place/60.299950,25.06382
Message Sent
```

Figure 12. Sending message

14 Automobile tracking system Implementation by Software

The design of the system started with including SoftwareSerial library and defined pins 7 and 8 as RX and TX of Arduino for serial communication. Display performance implemented by initially adding the LiquidCrystal library followed by assigning the Input and Output pins inside the LiquidCrystal constructor named with LCD. String variables were initialized at the time of array declaration and in the program loop.

The LCD showed the "Automobile Tracking Device" display information as a welcome message once LCD and Arduino UNO initialized with serial communication. The display showed the various status of the system like received the SMS, sending the text, sent a message including latitude and longitude. Software triggered the power "ON and OFF" of the module.

The received message first converted to text mode, assigned the command to read a live SMS. All the received message was then deleted.

When the GNSS power turned on, AT+CGNSSEQ=" RMC" command parsed the NMEA sentence in RMC message format. This functionality was set in the setup (). With the AT+CGNSINF (specific to SIM808 only) in the loop function, returned the single RMC data like UTC, latitude, longitude, speed etc. that included the required information for the project.

The device system got ready and waited for the message to receive. The module received the "Track" text from a specified number and returned with a fetched updated GPS data including google map link as SMS. The module accepted message only from the precise GSM number as demonstrated in the section of code below.

```
if (sim808.available() > 0) {
    String B = sim808.readString();
    B.trim();
    if (B.indexOf("Track") >= 0 && B.indexOf("my number") > 0)
```

And the module sent back a message to the same number as explained above in the sending message section.

Figure 14 showed the "Vehicle Tracking Device" as a welcome message.



Figure 14. Welcome message on LCD

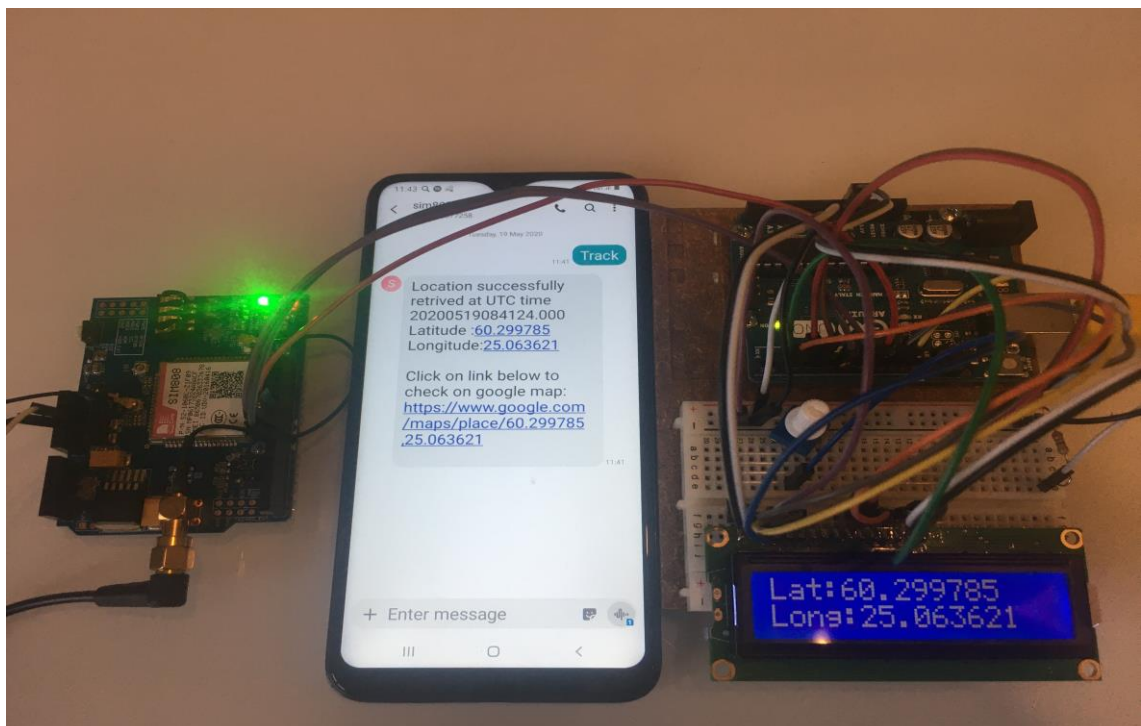


Figure 15. A complete tracking system

15 LCD status

The display showed various condition such as message status (received, sending, and sent result) together with latitude and longitude, as shown in the figures.

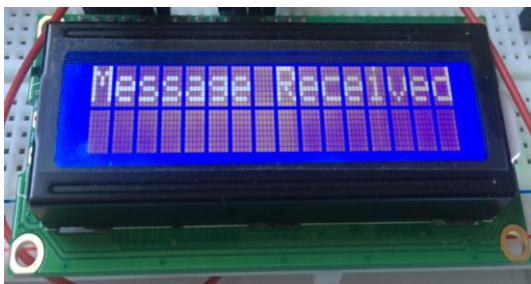


Figure 16.1. SMS received



Figure 16.2. Getting location from satellite

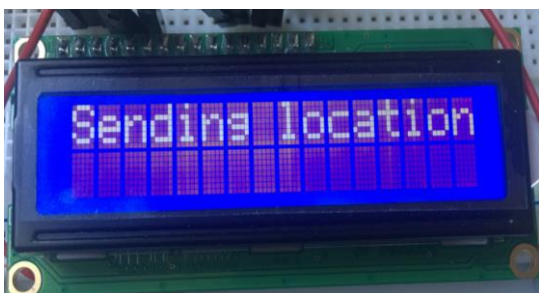


Figure 16.3. GPS location Sending to phone

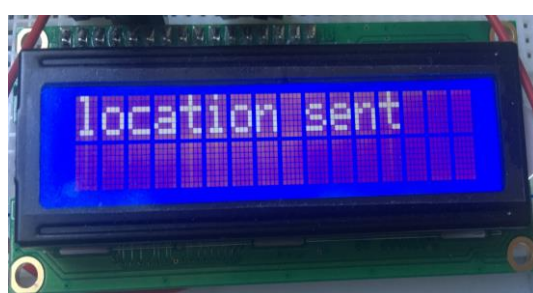


Figure 16.4. Latitude and longitude sent



Figure 16.5. GPS Coordinates shown in the display

16 Final Testing

The device fitted into the car and tested in various location. The system provided latitude and longitude with GPS time at every position in real-time. "Track" SMS sent to every point from my number and got a valid vehicle's position. The data below showed the final testing of the device.

- 1) UTC: 20200518102609.000 Latitude: 61.457318 Longitude: 23.75827
<https://www.google.com/maps/place/61.457318,23.75827>
- 2) UTC 20200518104210.000 Latitude: 61.296735 Longitude: 23.818750
<https://www.google.com/maps/place/61.296735,23.818750>
- 3) UTC: 20200518104547.000 Latitude: 61.231707 Longitude: 23.83712.
<https://www.google.com/maps/place/61.231707,23.83712>
- 4) UTC: 20200518110720.000 Latitude: 60.994627 Longitude: 24.45540.
<https://www.google.com/maps/place/60.994627,24.45540>
- 5) UTC: 20200518115106.000 Latitude: 60.296535 Longitude: 24.93370
<https://www.google.com/maps/place/60.296535,24.93370>
- 6) UTC: 20200518115521.000 Latitude: 60.291032 Longitude: 24.963441
<https://www.google.com/maps/place/60.291032,24.963441>
- 7) UTC: 20200518120055.000 Latitude: 60.294458 Longitude: 24.924266
<https://www.google.com/maps/place/60.294458,24.924266>

16.1 SMS on GSM phone

When "Track" message sent to the module from a number, the system sent back the complete information to the designated GSM showing UTC, latitude and longitude including a google map link. Figure 17 showed the result of automobile tracking system achieved on a mobile phone via SMS query.

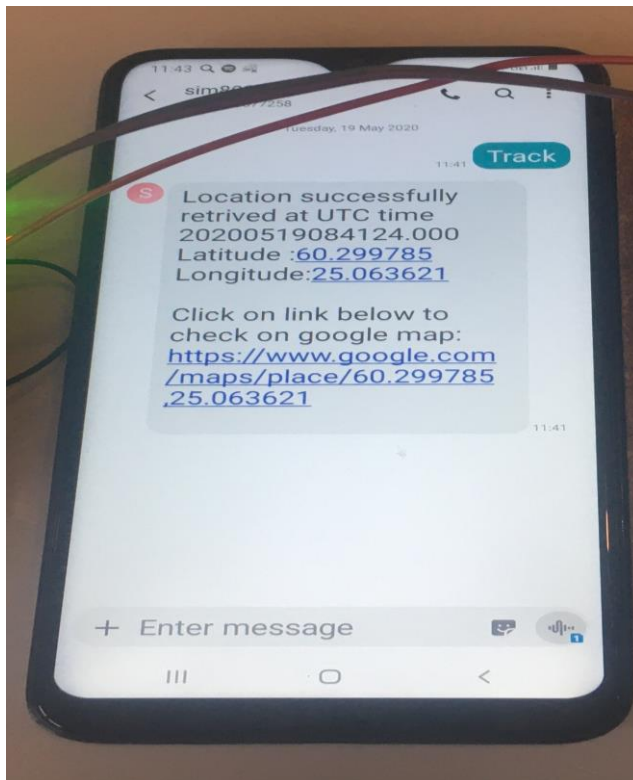


Figure 17. Result of SMS query on a mobile phone to get the location of a vehicle

16.2 Location determination by google map

The received data demonstrated as latitude (60.299785) and longitude (25.063621), as shown in figure 16.5. Google map determined the exact location of the retrieved coordinates entered into the search field of google map link (<https://www.google.com/maps>) separated by a comma. The returning site of manually entered data and the google map link in SMS of a phone were precise. Figure 17 showed the returning area of the map.

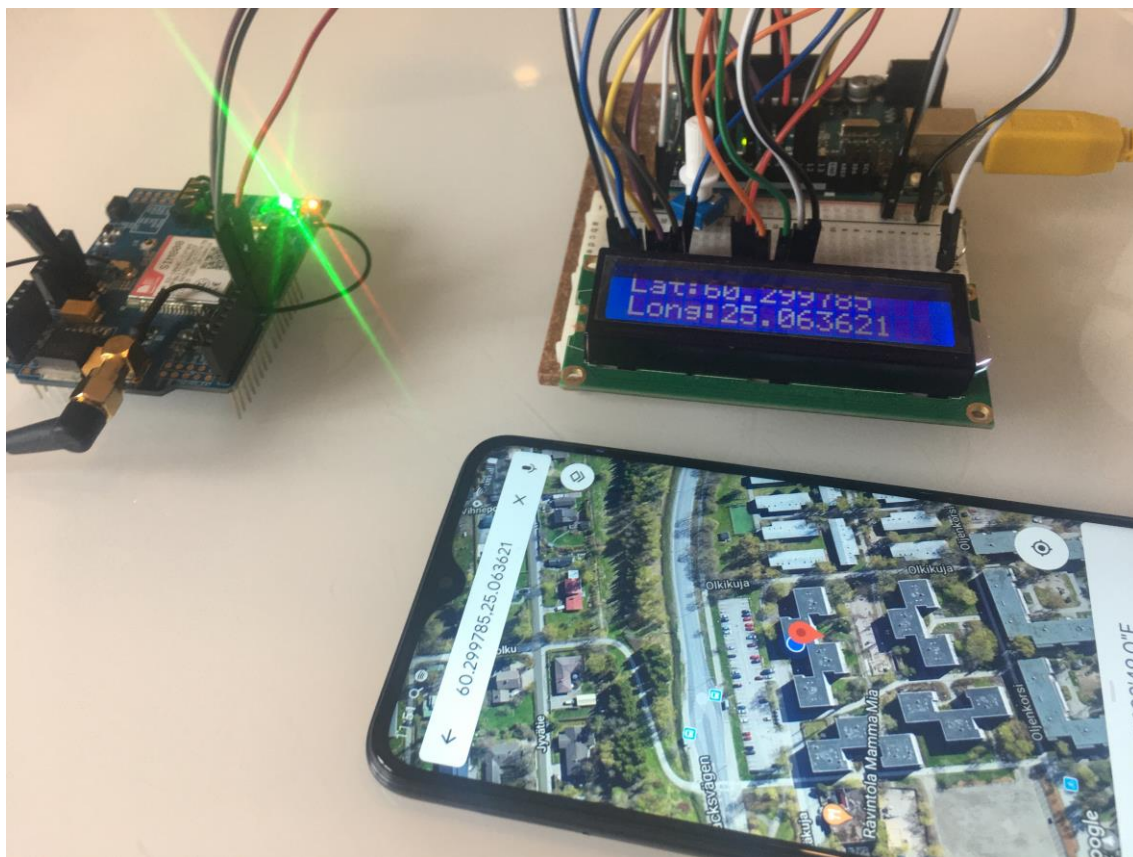


Figure 18. Google map returning location

17 Results

The sim808 shield was a complete, efficient package to build a tracking system with Arduino UNO as a primary operational part. The outcome of the entire method's performance observed was highly acceptable. The series of test conducted, and the result was excellent. Figure14 to 18 showed the set of the results of a tracking system.

18 Conclusion

The automobile tracking device was created and implemented successfully with the support of GPS and GSM technology from Arduino UNO and SIM808. The automobile was traceable at any location of the earth on a real-time via SMS request. The system can be used in the vehicle shortly as it is fully functioning.

Vehicle theft is a threat to security. The increasing risk of stealing a vehicle is incredibly rising. The development of such a tracker reduces the risk of robbing, which provides vehicle's position data at every interval of time. Hence, it provides security over the burglary of our car.

The implementation of the tracking device motivated to learn and experience into automation engineering.

All in all, the result obtained to meet the requirement of the project and provide a good result. However, much more works need for better performance.

References

1. SIM808 GPRS/GSM GPS Bluetooth Shield [Internet]. SIM808 GPRS/GSM GPS Shield ACS80801S Shield Arduino Compatible in Elecrow bazaar! [cited 23 April, 2020]. Available from: <https://www.elecrow.com/sim808-gprsgsmgps-shield-p-1389.html>
2. Sangeetha N, Suganthi K, Anitha PS, Bharathi A. Vehicle Tracking and Theft Control Using GSM and GPS. International Journal of Advances in Engineering. 21 March 2015;1(3):249–550.
3. Arduino UNO Reference Design [Internet]. Arduino UNO Reference Design. [cited 29 April, 2020]. Available from: <https://www.arduino.cc/en/uploads/Main/arduino-uno-schematic.pdf>
4. Hughes JM. The Arduino Family. In: Arduino: a technical reference: a handbook of technicians, engineers, and makers. Sebastopol: O'Reilly; 2016. p. 1.
5. Arduino Uno Rev3 [Internet]. Arduino Uno Rev3 | Arduino Official Store. [cited 24 April, 2020]. Available from: <https://store.arduino.cc/arduino-uno-rev3>
6. Boxall J. Exploring The Arduino Board And The IDE. In: Arduino workshop: a hands-on introduction with 65 projects. San Francisco: No Starch Press; 2013. p. 22–23.
7. Shields [Internet]. Arduino. [cited 25 April, 2020]. Available from: <https://www.arduino.cc/en/Main/arduinoShields>
8. Boxall J. Expanding Your Arduino. In: Arduino workshop: a hands-on introduction with 65 projects. San Francisco: No Starch Press; 2013. p. 162.
9. ATmega328P [Internet]. ATmega328P - 8-bit AVR Microcontrollers. [cited 24 April 2020]. Available from: <https://www.microchip.com/wwwproducts/en/ATmega328P>
10. Edric-深圳卓越迈创(<http://www.szmy.net/>). [Internet]. SIM808. [cited 25 April 2020]. Available from: <http://www.simcom.com/product/SIM808.html>
11. SIM808 GPRS/GSM GPS Shield v1.1 [Internet]. Elecrow. [cited 25 April 2020]. Available from: https://www.elecrow.com/wiki/index.php?title=SIM808_GPRS/GSM+GPS_Shield_v1.1
12. SIM808 Shield - GSM&GPRS GPS [Internet]. Seeed Studio. [cited 26 April 2020]. Available from: <https://www.seeedstudio.com/SIM808-Shield-GSM-GPRS-GPS-p-2524.html>

13. SIM808 GPRS/GSM+GPS Shield v1.1 [Internet]. Elecrow. [cited 2 May 2020]. Available from: https://www.elecrow.com/wiki/index.php?title=SIM808_GPRS%2FGSM%2BGPS_Shield_v1.1
14. 16x2 LCD Module [Internet]. Pinout, Diagrams, Description & Datasheet. [cited 28 April 2020]. Available from: <https://components101.com/16x2-lcd-pinout-datasheet>
15. National Marine Electronics Association [Internet]. NMEA. [cited 11 May 2020]. Available from: https://www.nmea.org/content/STAND-ARDS/NMEA_0183_Standard
16. AT commands [Internet]. IBM Knowledge Center. [cited 7 May 2020]. Available from: https://www.ibm.com/support/knowledgecenter/en/ssw_aix_72/network/asynch_atcommand.html
17. AT Commands [Internet]. AT Commands. [cited 7 May 2020]. Available from: <https://doc.qt.io/archives/qtextended4.4/atcommands.html#general-commands-3qpp-ts-27-007-section-5>
18. Purdum JJ. Arduino Libraries. In: Beginning C for Arduino: learn C programming for the Arduino. New York: Apress; 2015. p. 277–90.
19. Blemasle. blemasle/Arduino-sim808 [Internet]. GitHub. [cited 29 April 2020]. Available from: <https://github.com/blemasle/arduino-sim808>
20. SIM808 GPRS/GSM GPS Shield v1.1 [Internet]. Elecrow. [cited 9 May 2020]. Available from: https://www.elecrow.com/wiki/index.php?title=SIM808_GPRS/GSM+GPS_Shield_v1.1#Turn_on_the_SIM808_GPRS.2FGSM.2BGPS_shield
21. Mini GSM/GPRS GPS Breakout - SIM808. [cited 10 May, 2020]. Available from: https://seeeddoc.github.io/Mini_GSM-GPRS_Plus_GPS_Breakout-SIM808/
22. Read and Parse NMEA Data Directly From GPS Receiver [Internet]. Read and Parse NMEA Data Directly From GPS Receiver - MATLAB & Simulink. [cited 19 May, 2020]. Available from: <https://www.mathworks.com/help/fusion/examples/read-and-parse-nmea-data-directly-from-gps-receiver.html#d120e12070>

Appendices

Appendix 1. SIM808 shield test

```
#include <SoftwareSerial.h>
```

```
SoftwareSerial sim808(7, 8); //RX, TX of Arduino
```

```
void setup() {
```

```
    Serial.begin(9600); // begin serial communication between Arduino and PC (Serial monitor)
```

```
    sim808.begin(9600); // begin serial communication with Arduino and sim808
```

```
    delay(150);
```

```
}
```

```
void loop() {
```

```
    if (sim808.available())
```

```
Serial.write(sim808.read()); {  
  
if (Serial.available()  
  
    sim808.write(Serial.read());  
}  
}
```

Appendix 2. Power control through software trigger

```
Int pin= 9 ;  
  
void setup() {  
  
pinMode(pin, OUTPUT); // initialize pin as output  
  
void loop() {  
  
digitalWrite(pin, HIGH); //module turn on  
  
delay(1000);           // wait for 1 second  
  
digitalWrite(pin, LOW); //module turn off  
  
delay(500); // wait for 5 ms  
  
}
```

Appendix 3. Receiving and Sending message

```
//most important part of the code for receiving and sending message
```

```
#include <SoftwareSerial.h>
```

```
SoftwareSerial sim808(7, 8); //RX, TX of Arduino
```

```
void setup() {
```

```
    Serial.begin(9600);
```

```
    sim808.begin(9600);
```

```
    delay(150);
```

```
    sim808.print("AT+CMGF=1\r"); //text message format
```

```
    sim808.println("AT+CNMI=1,2,0,0,0");// newly arrived message indication
```

```
    delay(1000);
```

```
    sim808.println("AT+CMGD= 1,4"); //AT+CMGD=?+CMGD: (1-20),(0-4)// delete mes-  
    sage
```

```
}
```

```
void loop() {
```

```
    sim808.print("AT+CMGF=1\r"); //text message format
```

```
    delay(1000);
```

```
    sim808.print("AT+CMGS=\"+my_number\"\r"); // mobile number to be inserted
```

```
    delay(1000);
```



```
/******content of the message *****/  
  
Sim808.print("content of the sending message.");// UTC, latitude, longitude, google  
map  
  
sim808.write(0x1A); // sent message  
  
delay(1000);  
  
}
```