# **Arduino GPS Project Report**

******

[**AT NUST GIS DAY 14NOV 2018: MUHAMMAD HAMZA EXPLAINING THE NMEA CODE TO DR. NAEEM MALIK (UAAR),WAQAS IN BLACK COAT**](https://www.facebook.com/photo/?fbid=2009159929381529&set=pob.100010011044786)

DATE: 18/NOV/2018

**SEMESTER PROJECT: BS GIS**

**INSTITUTE OF GEO-INFORMATION AND EARTH OBSERVATION**

**UNIVERSITY OF ARID AGRICULTURE RAWALPINDI,PAKISTAN**

**Abstract:**

The Arduino GPS project aimed to develop a GPS tracking system using the NEO-6M GPS module and Arduino Uno. This report details the process of connecting the GPS module, obtaining raw GPS data, parsing the data to extract relevant location information, and the historical background of the project.

## **1. Introduction**

I am [MUHAMMAD HAMZA (mhwahla)](https://www.linkedin.com/in/muhammad-hamza-mhwahla-7a9b12135/), a GIS contestant. The Arduino GPS project was undertaken during the 4th semester at BS GIS ARID University Rawalpindi, Pakistan, in November 2018. The project team consisted of Muhammad Hamza, Waqas Iftikhar, Khurram Khan, and Afsheen Sheikh. The project was presented at NUST GIS Day 2018, held at NUST Main Campus Islamabad.

## **2. Historical Background**

The inspiration for this project came from a desire apart from just doing a semester project, we should contribute to the field of Geographic Information Systems (GIS) and make a positive impact on the world. The project was initiated by a discussion with a faculty member, Dr. NAEEM MALIK who encouraged students to explore innovative GIS-related projects. This led to the formation of project groups, and our team began brainstorming ideas for a unique and creative project that would also encompass GIS concepts.

## **3. Project Development**

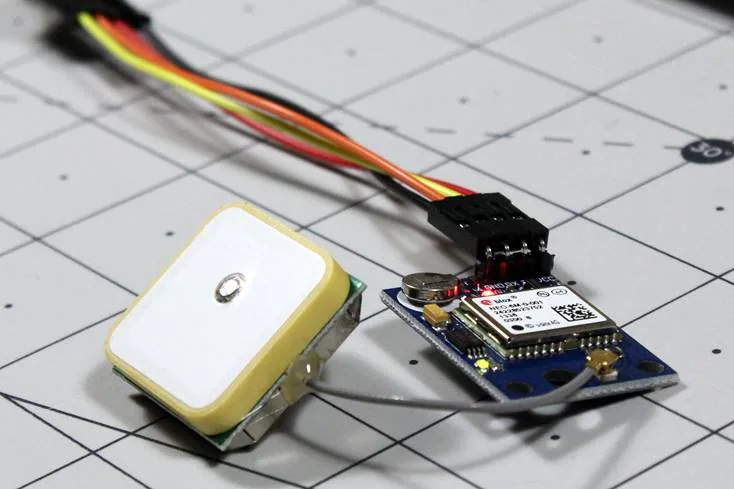
### **3.1 Procurement of Components**

To kickstart the project, essential components were procured. These components included an Arduino Uno, NEO-6M GPS module, various cables, boards, and electronic components. The team also acquired an external antenna to enhance GPS signal reception.

### **3.2 Familiarization with Arduino**

As students with limited hardware experience, the initial phase of the project involved getting acquainted with Arduino. Tutorials, resources from YouTube, Google, and the official Arduino website were extensively utilized. The team conducted experiments with LEDs, bulbs, and motors to build foundational knowledge in hardware programming.

### **3.3 GPS Module Exploration**

The next step focused on understanding the NEO-6M GPS module. Key specifications of the module, including its RS232 TTL interface, power supply requirements, default baud rate, and compatibility with NMEA sentences, were studied in detail. An external antenna was attached to the module to improve signal reception.

### **3.4 Challenges and Research**

The team encountered challenges in obtaining precise GPS data and struggled to find specific code examples for the NEO-6M module. Extensive research was conducted on platforms like Google, Quora, Reddit, and Stack Overflow. The breakthrough came when the team discovered a function for collecting NMEA data from satellites.

### **3.5 Code Development**

The team developed code for Arduino to communicate with the GPS module via Software Serial. The code enabled the extraction of raw GPS data, including NMEA sentences, which were subsequently displayed on the serial monitor.

## **4. Obtaining GPS Raw Data**

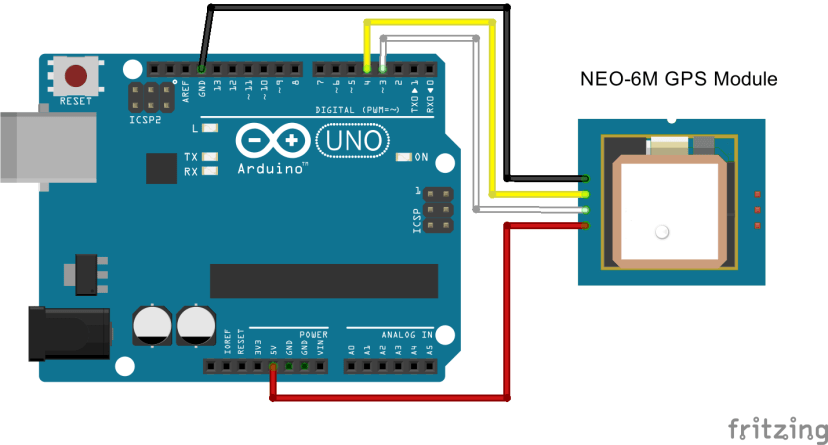
This section detailed the process of obtaining raw GPS data, which involved setting up serial communication with the GPS module using Software Serial. The necessary components for testing were listed.

### **4.1 Schematics**

A schematic diagram illustrated the connection between the NEO-6M GPS module and the Arduino Uno.

### **Schematics**

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



* The module GND pin is connected to 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 Arduino 5V pin

### **4.2 Code**

The Arduino code for establishing communication with the GPS module and displaying raw GPS data on the serial monitor was provided. Key portions of the code, such as software serial pin definitions and baud rates, were explained.

#include <TinyGPS++.h>

#include <SoftwareSerial.h>

SoftwareSerial GPS\_SoftSerial(4, 3); // Create a SoftwareSerial object for GPS communication on pins 4 (RX) and 3 (TX)

TinyGPSPlus gps; // Create a TinyGPSPlus object for parsing GPS data

volatile float minutes, seconds;

volatile int degree, secs, mins;

void setup() {

Serial.begin(115200); // Initialize serial communication with a baud rate of 115200

GPS\_SoftSerial.begin(9600); // Initialize software serial communication with the GPS module at 9600 baud

}

void loop() {

smartDelay(1000); // A function to manage serial communication and data parsing

unsigned long start;

double lat\_val, lng\_val, alt\_m\_val;

uint8\_t hr\_val, min\_val, sec\_val;

bool loc\_valid, alt\_valid, time\_valid;

// Retrieve GPS data

lat\_val = gps.location.lat(); // Get latitude data

loc\_valid = gps.location.isValid(); // Check if valid location data is available

lng\_val = gps.location.lng(); // Get longitude data

alt\_m\_val = gps.altitude.meters(); // Get altitude data in meters

alt\_valid = gps.altitude.isValid(); // Check if valid altitude data is available

hr\_val = gps.time.hour(); // Get hour

min\_val = gps.time.minute(); // Get minutes

sec\_val = gps.time.second(); // Get seconds

time\_valid = gps.time.isValid(); // Check if valid time data is available

// Display GPS data

if (!loc\_valid) {

Serial.print("Latitude : \*\*\*\*\*\n");

Serial.print("Longitude : \*\*\*\*\*\n");

} else {

DegMinSec(lat\_val);

Serial.print("Latitude in Decimal Degrees : ");

Serial.println(lat\_val, 6);

Serial.print("Latitude in Degrees Minutes Seconds : ");

Serial.print(degree);

Serial.print("\t");

Serial.print(mins);

Serial.print("\t");

Serial.println(secs);

DegMinSec(lng\_val); // Convert the decimal degree value into degrees minutes seconds form

Serial.print("Longitude in Decimal Degrees : ");

Serial.println(lng\_val, 6);

Serial.print("Longitude in Degrees Minutes Seconds : ");

Serial.print(degree);

Serial.print("\t");

Serial.print(mins);

Serial.print("\t");

Serial.println(secs);

}

if (!alt\_valid) {

Serial.print("Altitude : \*\*\*\*\*\n");

} else {

Serial.print("Altitude : ");

Serial.println(alt\_m\_val, 6);

}

if (!time\_valid) {

Serial.print("Time : \*\*\*\*\*\n");

} else {

char time\_string[32];

sprintf(time\_string, "Time : %02d/%02d/%02d\n", hr\_val, min\_val, sec\_val);

Serial.print(time\_string);

}

}

static void smartDelay(unsigned long ms) {

unsigned long start = millis();

do {

while (GPS\_SoftSerial.available()) {

gps.encode(GPS\_SoftSerial.read()); // Parse GPS data from the software serial

}

} while (millis() - start < ms);

}

void DegMinSec(double tot\_val) {

degree = (int)tot\_val;

minutes = tot\_val - degree;

seconds = 60 \* minutes;

minutes = (int)seconds;

mins = (int)minutes;

seconds = seconds - minutes;

seconds = 60 \* seconds;

secs = (int)seconds;

}

## **5. Understanding NMEA Sentences**

NMEA sentences, the standard data format used by GPS modules, were introduced. Each NMEA sentence was explained, with examples provided. The report clarified the structure of NMEA sentences, including their starting character ($) and comma-separated data fields.

## **6. Parsing NMEA Sentences with TinyGPS++ Library**

This section demonstrated how to parse NMEA sentences using the TinyGPS++ library. The library's installation process was outlined.

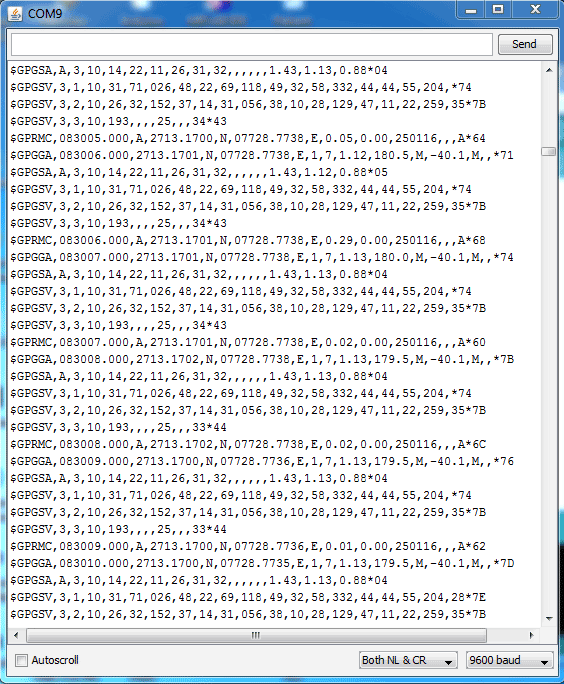
### **6.1 Getting Location Using the NEO-6M GPS Module and the TinyGPS++ Library**

The report featured Arduino code that utilized the TinyGPS++ library to extract location data (latitude and longitude) from the GPS module. The code was explained step by step, including library import, pins definition, and data extraction.

### **6.2 Getting More GPS Information Using the TinyGPS++ Library**

An extended code shows how to retrieve additional GPS information, including date, time, speed, course, altitude, satellites, and HDOP (Horizontal Dilution of Precision). Explanations for each data field were included.

## **7. Conclusion**

The Arduino GPS project successfully demonstrated the integration of the NEO-6M GPS module with an Arduino Uno and the use of the TinyGPS++ library to extract meaningful GPS information. The knowledge gained from this project laid the foundation for future applications of GPS technology.

## **8. Acknowledgments**

The project team expressed gratitude to project mentors, faculty members, and individuals who provided assistance during the development process.

## **9. References**

**MH WAHLA**  
LINKEDIN (<https://www.linkedin.com/in/muhammad-hamza-mhwahla-7a9b12135/>)

FB (<https://www.facebook.com/humzahwahla/>)

CELL +923167333888

PAUL (<https://www.youtube.com/watch?v=fJWR7dBuc18>)

Arduino is an Italian open-source hardware and software company, project, and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices.

ALL RIGHTS RESERVED BY MHWAHLA