

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

DATE: 18/NOV/2018

SEMESTER PROJECT: BS GIS

INSTITUTE OF GEO-INFORMATION AND EARTH OBSERVATION

UNIVERSITY OF ARID AGRICULTURE RAWALPINDI, PAKISTAN

GIS HARDWARE

ARDUINO GPS NEO 6M

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 \(mhwahla\)](#), 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

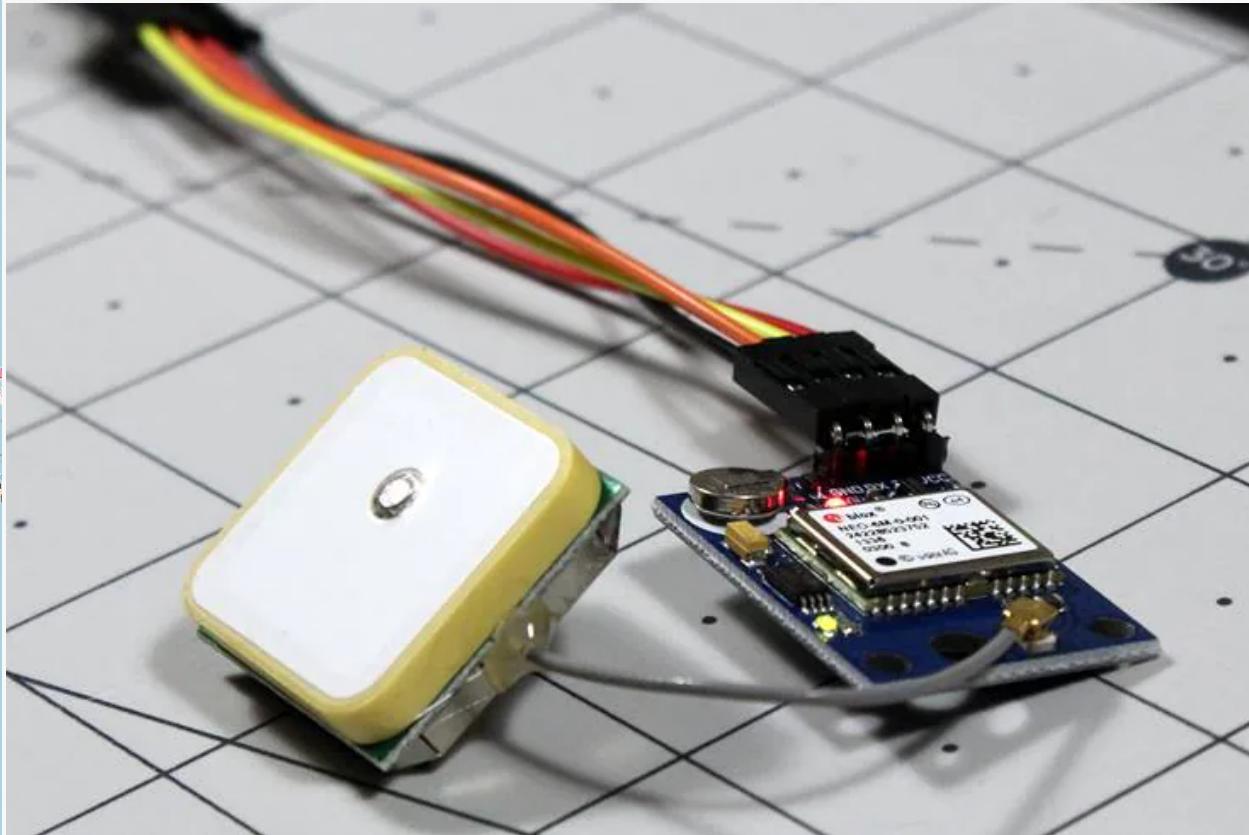
To kickstart the project, essential components were procured. These components included an Arduino Uno, NEO-6M GPS module, various cables, boards, and electronic components. The team also acquired an external antenna to enhance GPS signal reception.

3.2 Familiarization with Arduino

GIS HARDWARE ARDUINO GPS NEO 6M

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.

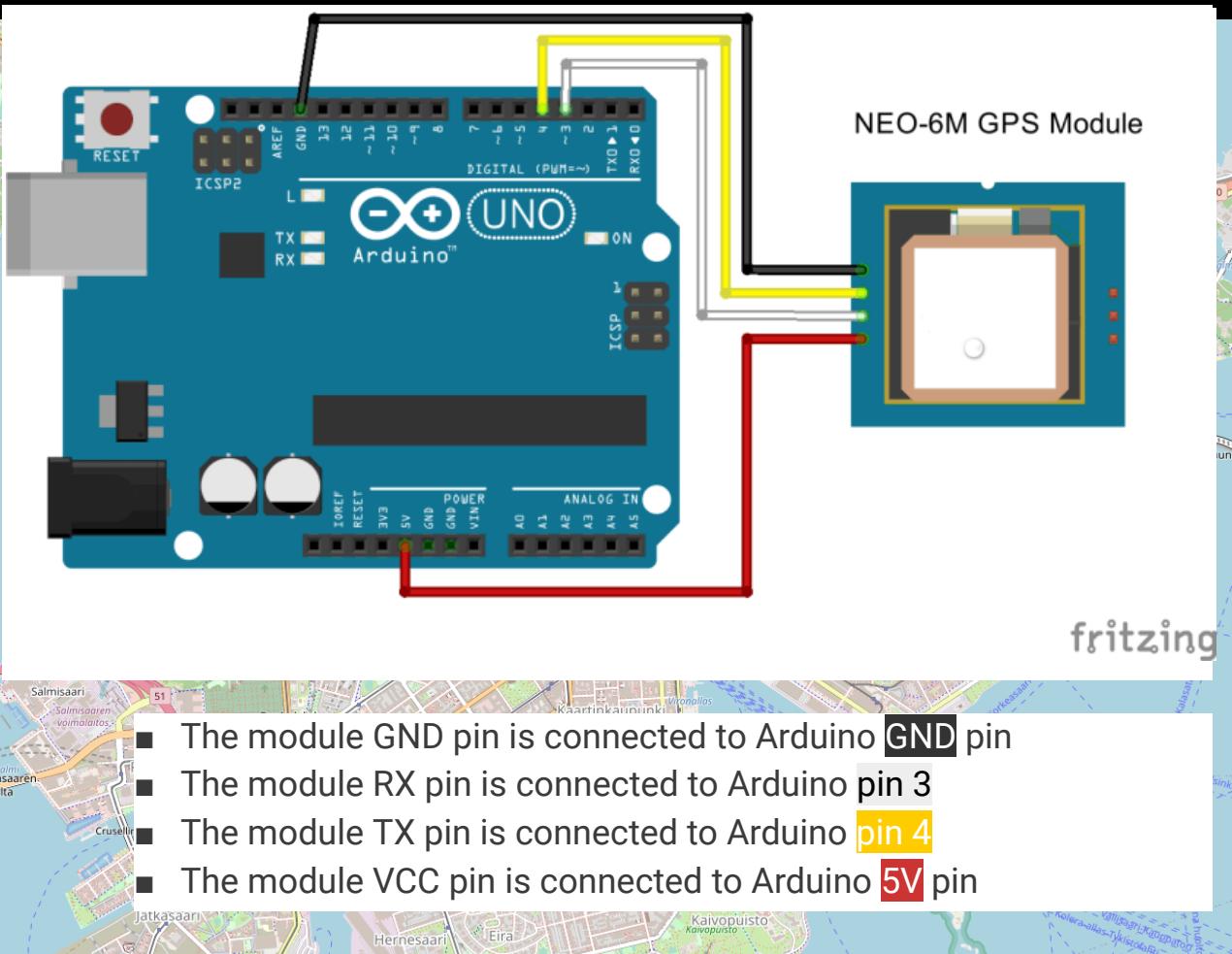
4.1 Schematics

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.

GIS HARDWARE ARDUINO GPS NEO 6M



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>
```

GIS HARDWARE ARDUINO GPS NEO 6M

```
#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
```

GIS HARDWARE ARDUINO GPS NEO 6M

```
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
```



GIS HARDWARE ARDUINO GPS NEO 6M

```
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);
```



GIS HARDWARE ARDUINO GPS NEO 6M

```
}
```

```
}
```

```
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

GIS HARDWARE ARDUINO GPS NEO 6M

applications of GPS technology.

```
$GPGSA,A,3,10,14,22,11,26,31,32, ,,, 1.43,1.13,0.88*04
$GPGSV,3,1,10,31,71,026,48,22,69,118,49,32,58,332,44,44,55,204,*74
$GPGSV,3,2,10,26,32,152,37,14,31,056,38,10,28,129,47,11,22,259,35*7B
$GPGSV,3,3,10,193, ,,, 25, ,,, 34*43
$GPRMC,083005.000,A,2713.1700,N,07728.7738,E,0.05,0.00,250116,,,A*64
$GPGGA,083006.000,2713.1701,N,07728.7738,E,1,7,1.12,180.5,M,-40.1,M,,,*71
$GPGSA,A,3,10,14,22,11,26,31,32, ,,, 1.43,1.12,0.88*05
$GPGSV,3,1,10,31,71,026,48,22,69,118,49,32,58,332,44,44,55,204,*74
$GPGSV,3,2,10,26,32,152,37,14,31,056,38,10,28,129,47,11,22,259,35*7B
$GPGSV,3,3,10,193, ,,, 25, ,,, 34*43
$GPRMC,083006.000,A,2713.1701,N,07728.7738,E,0.29,0.00,250116,,,A*68
$GPGGA,083007.000,2713.1701,N,07728.7738,E,1,7,1.13,180.0,M,-40.1,M,,,*74
$GPGSA,A,3,10,14,22,11,26,31,32, ,,, 1.43,1.13,0.88*04
$GPGSV,3,1,10,31,71,026,48,22,69,118,49,32,58,332,44,44,55,204,*74
$GPGSV,3,2,10,26,32,152,37,14,31,056,38,10,28,129,47,11,22,259,35*7B
$GPGSV,3,3,10,193, ,,, 25, ,,, 34*43
$GPRMC,083007.000,A,2713.1701,N,07728.7738,E,0.02,0.00,250116,,,A*60
$GPGGA,083008.000,2713.1702,N,07728.7738,E,1,7,1.13,179.5,M,-40.1,M,,,*7B
$GPGSA,A,3,10,14,22,11,26,31,32, ,,, 1.43,1.13,0.88*04
$GPGSV,3,1,10,31,71,026,48,22,69,118,49,32,58,332,44,44,55,204,*74
$GPGSV,3,2,10,26,32,152,37,14,31,056,38,10,28,129,47,11,22,259,35*7B
$GPGSV,3,3,10,193, ,,, 25, ,,, 33*44
$GPRMC,083008.000,A,2713.1702,N,07728.7738,E,0.02,0.00,250116,,,A*6C
$GPGGA,083009.000,2713.1700,N,07728.7736,E,1,7,1.13,179.5,M,-40.1,M,,,*76
$GPGSA,A,3,10,14,22,11,26,31,32, ,,, 1.43,1.13,0.88*04
$GPGSV,3,1,10,31,71,026,48,22,69,118,49,32,58,332,44,44,55,204,*74
$GPGSV,3,2,10,26,32,152,37,14,31,056,38,10,28,129,47,11,22,259,35*7B
$GPGSV,3,3,10,193, ,,, 25, ,,, 33*44
$GPRMC,083009.000,A,2713.1700,N,07728.7736,E,0.01,0.00,250116,,,A*62
$GPGGA,083010.000,2713.1700,N,07728.7735,E,1,7,1.13,179.5,M,-40.1,M,,,*7D
$GPGSA,A,3,10,14,22,11,26,31,32, ,,, 1.43,1.13,0.88*04
$GPGSV,3,1,10,31,71,026,48,22,69,118,49,32,58,332,44,44,55,204,28*7E
$GPGSV,3,2,10,26,32,152,37,14,31,056,38,10,28,129,47,11,22,259,35*7B
```

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

