# GPS: Transportation System

1.Acquire GPS coordinates from the Ublox Neo-m7/m8 GPS module.
2. Determine the GPS heading using the built-in magnetometer.

We used GPS ublox neo m6 which has 2 i2c pins, nothing but a inbuilt magnetometer. and 2 i2c pins are connected to a4 and a5 of Arduino.

Code:

```
#include <Wire.h>          // Include the Wire library for I2C communication
#include <Adafruit_Sensor.h>
#include <Adafruit_HMC5883_U.h>  // Include the Adafruit HMC5883L Magnetometer
library
#include <TinyGPS++.h>     // Include the TinyGPS++ library for GPS data
parsing

TinyGPSPlus gps;           // Create a TinyGPS++ object to store GPS data
Adafruit_HMC5883_Unified mag;  // Create an Adafruit magnetometer object

void setup() {
  Serial.begin(9600);       // Initialize the Serial communication
  Wire.begin();             // Initialize I2C communication on pins A4 and A5

  if (!mag.begin()) {
    Serial.println("Could not find a valid magnetometer! Check wiring.");
    while (1);
  }
}

void loop() {
  // Read GPS data
  while (Serial.available() > 0) {
    if (gps.encode(Serial.read())) {
      if (gps.location.isValid()) {
        float latitude = gps.location.lat();
        float longitude = gps.location.lng();

        // Read magnetometer data
        sensors_event_t event;
```

```
        mag.getEvent(&event);
        float heading = atan2(event.magnetic.y, event.magnetic.x) * 180 / PI;
        if (heading < 0) {
          heading += 360;
        }

        // Print GPS coordinates and heading
        Serial.print("Latitude: "); Serial.println(latitude, 6);
        Serial.print("Longitude: "); Serial.println(longitude, 6);
        Serial.print("Heading: "); Serial.println(heading, 2);
      }
    }
  }
}
```

Output:

Longitude: 77.755950

Heading: 225.00

Latitude: 12.984527

Longitude: 77.755950

Heading: 225.00

Latitude: 12.984527

Longitude: 77.755950

Heading: 225.00

Latitude: 12.984527

# 3. Calculate speed and acceleration using appropriate formulas.

Code:

```
#include <Wire.h>
#include <TinyGPS++.h>

TinyGPSPlus gps;

float prevLat = 0.0;
float prevLon = 0.0;
unsigned long prevTime = 0;
float prevSpeed = 0.0;
```

```
void setup() {
  Serial.begin(9600);
  Wire.begin();
}

void loop() {
  while (Serial.available() > 0) {
    if (gps.encode(Serial.read())) {
      if (gps.location.isValid() && gps.hdop.isValid() && gps.hdop.hdop() <
2.0) {
        float latitude = gps.location.lat();
        float longitude = gps.location.lng();

        // Calculate distance using Haversine formula
        float distance = haversine(prevLat, prevLon, latitude, longitude);

        // Calculate time difference in seconds
        unsigned long currentTime = millis();
        float deltaTime = (currentTime - prevTime) / 1000.0; // Convert
milliseconds to seconds

        // Calculate speed (in meters per second)
        float speed = distance / deltaTime;

        // Calculate acceleration (in meters per second squared)
        float acceleration = (speed - prevSpeed) / deltaTime;

        // Update previous values for the next iteration
        prevLat = latitude;
        prevLon = longitude;
        prevTime = currentTime;
        prevSpeed = speed;

        // Print speed, acceleration, and HDOP
        Serial.print("Speed: "); Serial.println(speed, 6);
        Serial.print("Acceleration: "); Serial.println(acceleration, 6);
        Serial.print("HDOP: "); Serial.println(gps.hdop.hdop(), 6);
        delay(1000);
      }
    }
  }
}

float haversine(float lat1, float lon1, float lat2, float lon2) {
  // Haversine formula to calculate distance between two GPS coordinates
  float dLat = radians(lat2 - lat1);
  float dLon = radians(lon2 - lon1);
```

```
  float a = sin(dLat / 2.0) * sin(dLat / 2.0) + cos(radians(lat1)) *
cos(radians(lat2)) * sin(dLon / 2.0) * sin(dLon / 2.0);
  float c = 2.0 * atan2(sqrt(a), sqrt(1.0 - a));

  // Radius of the Earth in meters (change it based on your requirements)
  float radius = 6371000.0;

  return radius * c;
}
```

Output:

Speed: 0.822760

Acceleration: 0.040913

HDOP: 1.460000

Speed: 0.000000

Acceleration: -0.820299

HDOP: 1.460000

Speed: 0.000000

Acceleration: 0.000000

HDOP: 1.460000