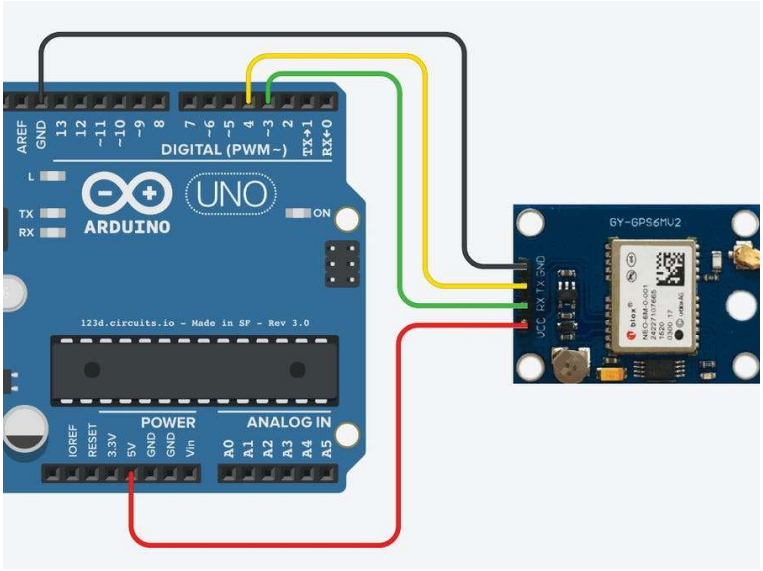
NEO-6M GPS

Content

1. About GPS

What is GPS

The Global Positioning System (GPS) is a satellite-based navigation system made up of at least 24 satellites. GPS works in any weather conditions, anywhere in the world, 24 hours a day, with no subscription fees or setup charges.



How GPS works

GPS satellites circle the Earth twice a day in a precise orbit. Each satellite transmits a unique signal and orbital parameters that allow GPS devices to decode and compute the precise location of the satellite. GPS receivers use this information and trilateration to calculate a user's exact location. Essentially, the GPS receiver measures the distance to each satellite by the amount of time it takes to

receive a transmitted signal. With distance measurements from a few more satellites, the receiver can determine a user's position and display it.

To calculate your 2-D position (latitude and longitude) and track movement, a GPS receiver must be locked on to the signal of at least 3 satellites. With 4 or more satellites in view, the receiver can determine your 3-D position (latitude, longitude and altitude). Generally, a GPS receiver will track 8 or more satellites, but that depends on the time of day and where you are on the earth.

Once your position has been determined, the GPS unit can calculate other information, such as:

- Speed
- Bearing
- Track
- Trip dist
- Distance to destination

What's the signal?

GPS satellites transmit at least 2 low-power radio signals. The signals travel by line of sight, meaning they will pass through clouds, glass and plastic but will not go through most solid objects, such as buildings and mountains. However, modern receivers are more sensitive and can usually track through houses.

A GPS signal contains 3 different types of information:

- Pseudorandom code is an I.D. code that identifies which satellite is transmitting information. You can see which satellites you are getting signals from on your device's satellite page.
- Ephemeris data is needed to determine a satellite's position and gives important information about the health of a satellite, current date and time.
- Almanac data tells the GPS receiver where each GPS satellite should be at any time throughout the day and shows the orbital information for that satellite and every other satellite in the system.

2. Download and install required libraries for GPS

- SoftwareSerial library
- TinyGPS library

3. NEO-6M GPS module

The NEO-6M GPS module is shown in the figure below. It comes with an external antenna and does not come with header pins. So, you will need to solder it.

Overview of NEO-6M GPS Module

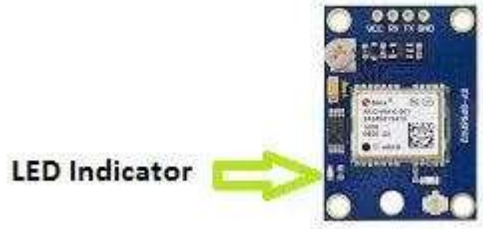
NEO-6M GPS Chip
The heart of the module is a NEO-6M GPS chip from u-blox. It can track up to 22 satellites on 50 channels and achieves the industry's highest level of sensitivity i.e. -161 dB tracking, while consuming only 45mA supply current. The u-blox 6 positioning engine also boasts a Time-To-First-Fix (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 chip are broken out to a "0.1" pitch headers. This includes pins required for communication with a microcontroller over UART.

Note: The module supports baud rate from 4800bps to 230400bps with default baud of 9600.

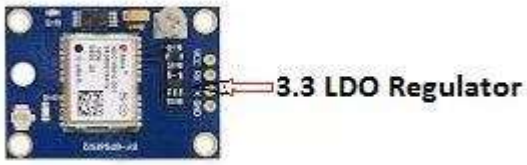
Position Fix LED Indicator

There is an LED on the NEO-6M GPS Module which indicates the status of Position Fix. It'll blink at various rates depending on what state it's in

- No Blinking ==> means It is searching for satellites
- Blink every 1s – means Position Fix is found
- 3.3V LDO Regulator



The operating voltage of the NEO-6M chip is from 2.7 to 3.6V. But the module comes with MIC5205 ultra-low dropout 3V3 regulator from MICREL. The logic pins are also 5-volt tolerant, so we can easily connect it to an Arduino or any 5V logic microcontroller without using any logic level converter.



Battery & EEPROM

The module is equipped with an HK24C32 two wire serial EEPROM. It is 4KB in size and connected to the NEO-6M chip via I2C.The module also contains a rechargeable button battery which acts as a super-capacitor.

An EEPROM together with battery helps retain the battery backed RAM (BBR). The BBR contains clock data, latest position data (GNSS or bit data) and module configuration. But it is not meant for permanent data storage.

As the battery retains clock and last position, time to first fix (TTFF) significantly reduces to 1s. This allows much faster position locks.

Without the battery the GPS always cold-start so the initial GPS lock takes more time. The battery is automatically charged when power is applied and maintains data for up to two weeks without power.

Pinout

- GND is the Ground Pin and needs to be connected to 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.

4. Connection of Arduino UNO and GPS module

Connect the four pins from UBLOX to an Arduino as follows:

GPS module ==> Arduino

GND ==> GND
TX ==> Digital pin (D3)
RX ==> Digital pin (D4)
Vcc ==> 3.3 V

Here, I suggest you to use external power supply to power the GPS module because minimum power requirement for GPS module to work is 3.3 V and Arduino is not capable of providing that much voltage.

To provide voltage use prolific USB TTL.

- USB driver

One more thing I have found while working with GPS antenna comes with module is its not receiving signal inside the house so I used this antenna.

- https://linxtechnologies.com/wp/product/sh-series-gps-antenna/

5. JHD162a LCD (OPTIONAL)

- Ground ==> Ground pin of the LCD module.
- Pin2(Vcc) ==> Power to LCD module (+5V supply is given to this pin)
- Pin3(VEE) ==> Contrast adjustment pin. This is done by connecting the ends of a 10K potentimeter to +5V and ground and then connecting the slider pin to the VEE pin. The voltage at the VEE pin defines the contrast. The normal setting is between 0.4 and 0.9V.
- Pin4(RS) ==> Register select pin.The JHD162A has two registers namely command register and data register. Logic HIGH at RS pin selects data register and logic LOW at RS pin selects command register. If we make the RS pin HIGH and feed an input to the data lines (DB0 to DB7), this input will be treated as data to display on LCD screen. If we make the RS pin LOW

and feed an input to the data lines, then this will be treated as a command (a command to be written to LCD controller – like positioning cursor or clear screen or scroll).

- Pin5(R/W) ==> Read/Write modes. This pin is used for selecting between read and write modes. Logic HIGH at this pin activates read mode and logic LOW at this pin activates write mode.
- Pin6(E) ==> This pin is meant for enabling the LCD module. A HIGH to LOW signal at this pin will enable the module.
- Pin7(DB0) to Pin14(DB7) ==> These are data pins. The commands and data are fed to the LCD module though these pins.
- Pin15(LED+) ==> Anode of the back light LED. When operated on 5V, a 560 ohm resistor should be connected in series to this pin. In arduino based projects the back light LED can be powered from the 3.3V source on the arduino board.
- Pin16(LED-) ==> Cathode of the back light LED.

6. Connection of Arduino UNO and JHD162a LCD

LCD ==> Arduino

VSS ==> GND
VCC ==> 5V
VEE ==> 10K Resistor
RS ==> A0 (Analog pin)
R/W ==> GND
E ==> A1
D4 ==> A2
D5 ==> A3
D6 ==> A4
D7 ==> A5
LED+ ==> VCC
LED- ==> GND

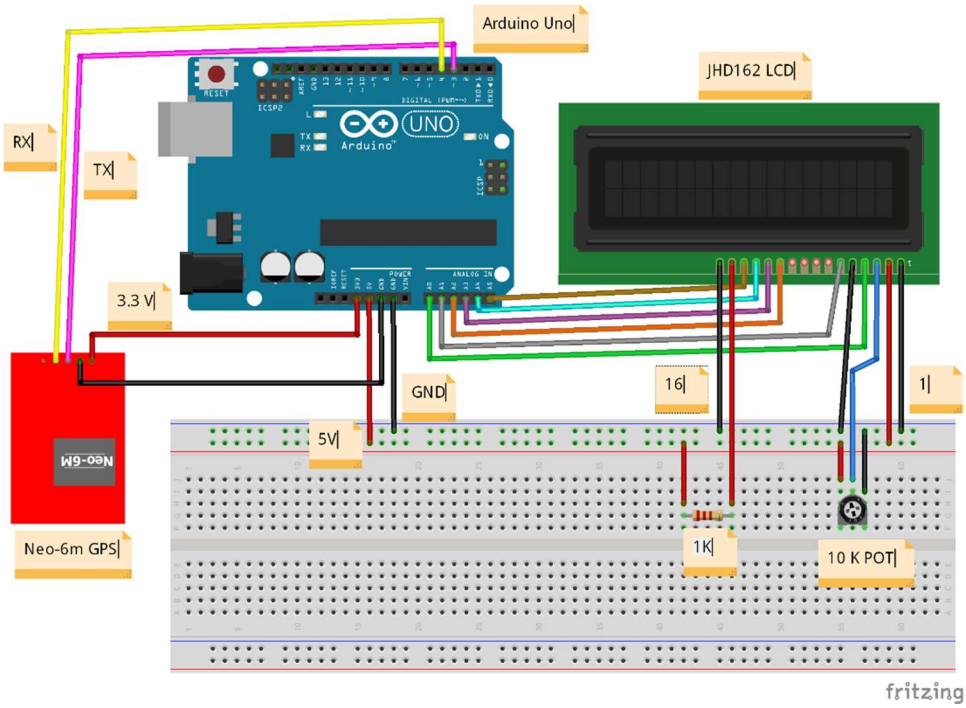
Programmingcode

```
#include <LiquidCrystal.h>
#include <SoftwareSerial.h>
#include <TinyGPS.h>
float lat = 28.5458,lon = 77.1703; // create variable for latitude and longitude object
SoftwareSerial gpsSerial(3,4);//rx,tx
LiquidCrystal lcd(A0,A1,A2,A3,A4,A5);
TinyGPS gps; // create gps object
void setup(){
Serial.begin(9600); // connect serial
//Serial.println("The GPS Received Signal:");
gpsSerial.begin(9600); // connect gps sensor
lcd.begin(16,2);
}
void loop(){
  while(gpsSerial.available()){ // check for gps data
  if(gps.encode(gpsSerial.read()))// encode gps data
  {
  gps.f_get_position(&lat,&lon); // get latitude and longitude
  // display position
  lcd.clear();
  lcd.setCursor(1,0);
  lcd.print("GPS Signal");
  //Serial.print("Position: ");
  //Serial.print("Latitude:");
  //Serial.print(lat,6);
  //Serial.print(";");
  //Serial.print("Longitude:");
  //Serial.println(lon,6);
  lcd.setCursor(1,0);
  lcd.print("LAT:");
  lcd.setCursor(5,0);
  lcd.print(lat);
  //Serial.print(lat);
  //Serial.print(" ");
  lcd.setCursor(0,1);
  lcd.print(",LON:");
  lcd.setCursor(5,1);
  lcd.print(lon);
 }
}
String latitude = String(lat,6);
  String longitude = String(lon,6);
Serial.println(latitude+";"+longitude);
delay(1000);
}
```

Main code

```
Arduino codeArduino
#include <LiquidCrystal.h>
#include <SoftwareSerial.h>
#include <TinyGPS.h>
//long   lat,lon; // create variable for latitude and longitude object
float lat = 28.5458,lon = 77.1703; // create variable for latitude and longitude object
SoftwareSerial gpsSerial(3,4);//rx,tx
LiquidCrystal lcd(A0,A1,A2,A3,A4,A5);
TinyGPS gps; // create gps object
void setup(){
Serial.begin(9600); // connect serial
//Serial.println("The GPS Received Signal:");
gpsSerial.begin(9600); // connect gps sensor
lcd.begin(16,2);
}

void loop(){
   while(gpsSerial.available()){ // check for gps data
   if(gps.encode(gpsSerial.read()))// encode gps data
   {
   gps.f_get_position(&lat,&lon); // get latitude and longitude
   // display position
   lcd.clear();
   lcd.setCursor(1,0);
   lcd.print("GPS Signal");
   //Serial.print("Position: ");
   //Serial.print("Latitude:");
   //Serial.print(lat,6);
   //Serial.print(";");
   //Serial.print("Longitude:");
   //Serial.println(lon,6);
   lcd.setCursor(1,0);
   lcd.print("LAT:");
   lcd.setCursor(5,0);
   lcd.print(lat);
   //Serial.print(lat);
   //Serial.print(" ");

   lcd.setCursor(0,1);
   lcd.print(",LON:");
   lcd.setCursor(5,1);
   lcd.print(lon);

  }
 }

 String latitude = String(lat,6);
   String longitude = String(lon,6);
 Serial.println(latitude+";"+longitude);
```

```
 delay(1000);

}
```


Arduino Uno
JHD162 LCD
RX
TX
3.3 V
GND
5V
16
1
Neo-6m GPS
1K
10 K POT
fritzing

Source

https://create.arduino.cc/projecthub/ruchir1674/how-to-interface-gps-module-neo-6m-with-arduino-8f90ad