A logo of a university

Description automatically generated A close-up of a computer code

Description automatically generated

*Universitatea Politehnica din Bucureşti*

*Facultatea de Electronică şiTelecomunicaţii*

# Proiect 3 - Proiectarea unui modul GPS GNSS

Studenti:Chivoiu Gabriel

Nita Robert-Andrei

Belinsca Stefan-Alexandru

Grupa:444C

An universitar 2023-2024

# Cuprins

[Proiect 3 - Proiectarea unui modul GPS GNSS 1](#_Toc187783107)

[Cuprins 2](#_Toc187783108)

[Listă figuri 3](#_Toc187783109)

[Tema proiectului : 4](#_Toc187783110)

[Schema bloc funcţională 4](#_Toc187783111)

[Descriere hardware: 5](#_Toc187783112)

[Descriere scurtă a configurației: 7](#_Toc187783113)

[Rezultate și Aplicații 9](#_Toc187783114)

[Proiectarea unui Sistem GNSS pentru Afișarea Locației pe o Interfață Web 12](#_Toc187783115)

[Bibliografie 13](#_Toc187783116)

[Anexă cod 14](#_Toc187783117)

# Listă figuri

Figura1- poză cu placa Arduino Uno

Figura2,3-poze co montajul între Arduino Uno și antenă

Figura4-rezultate serial Arduino

***Proiectarea unui modul GPS GNSS***

# Tema proiectului :

Sa se realizeze un sistem de localizare și monitorizare GPS/GNSS, utilizând un modul GPS/GNSS conectat la un adaptor USB-TTL pentru comunicație cu un computer sau un microcontroler. Scopul acestui proiect este de a permite recepția și procesarea coordonatelor de poziționare globală pentru a afișa în timp real locația unui dispozitiv sau pentru a înregistra traseul parcurs într-o aplicație specifică. Sistemul trebuie să fie capabil să colecteze date precise de poziționare (latitudine, longitudine, altitudine) și să le transmită către o interfață software pentru afișare și analiză ulterioară.

# Schema bloc funcţională

 **Antenă GPS**

* Captarea semnalelor satelit GPS/GNSS.

 **Circuit RF (Radio-Frecvență)**

* Amplifică și filtrează semnalul de la antenă pentru procesare ulterioară.

 **Modul de procesare GNSS**

* Unitatea centrală care decodifică și interpretează semnalele pentru a extrage informațiile de poziționare (latitudine, longitudine, altitudine).

 **Memorie/Buffer**

* Poate fi inclusă pentru stocarea temporară a datelor procesate.

 **Interfața de comunicație (TX/RX)**

* Comunicare serială TTL pentru transmiterea datelor procesate către alte dispozitive.

 **Alimentare**

* Intrare de alimentare (VCC) și masă (GND).

# Descriere hardware:

Acest ansamblu constă dintr-un modul GPS/GNSS conectat la un adaptor USB-TTL pentru comunicație serială cu un computer sau alt dispozitiv compatibil. Modulul GPS/GNSS permite recepția semnalelor de poziționare globală pentru aplicații de navigație și localizare, iar adaptorul USB-TTL servește drept interfață între modul și un port USB, convertind semnalele seriale TTL în semnale compatibile USB.

**Modulul GPS/GNSS**

* Modulul este echipat cu o antenă externă pentru o recepție îmbunătățită a semnalului.
* Conectorii de pe modul includ pinii TX (transmitere date) și RX (recepționare date), necesari pentru comunicația serială.

**Adaptorul USB-TTL**

* Adaptorul include o interfață USB standard și pinii de ieșire TX, RX, GND și VCC pentru conectarea cu modulele externe.
* Un indicator LED roșu confirmă alimentarea și activitatea.

**Conexiuni și schema de cablare:**

* Pinul TX al modulului GPS/GNSS se conectează la pinul RX al adaptorului USB-TTL.
* Pinul RX al modulului se conectează la pinul TX al adaptorului USB-TTL.
* GND (masă) de pe modul se conectează la GND de pe adaptor.
* VCC de pe modul este conectat la VCC de pe adaptor pentru alimentare (asigură-te că tensiunile sunt compatibile).

Nu am reusit conexiune si vizualizarea datelor cu ajutorul TTL-ului asa ca am trecut la o placa de test Arduino Uno pentru a putea realiza montajul si citirea pe serial a coordonatelor date de Gps-ul nostru.

**Arduino Uno** este o placă de dezvoltare microcontroler, extrem de populară în rândul pasionaților de electronică și programare. Este bazată pe microcontrolerul **ATmega328P** și oferă o platformă simplă, dar puternică, pentru dezvoltarea de proiecte electronice.

**Specificații principale:**

* **Microcontroler**: ATmega328P.
* **Număr de pini I/O digitali**: 14 (dintre care 6 pot fi folosiți ca ieșiri PWM).
* **Număr de pini analogici**: 6.
* **Memorie Flash**: 32 KB, dintre care 0.5 KB este folosită pentru bootloader.
* **RAM (SRAM)**: 2 KB.
* **EEPROM**: 1 KB.
* **Alimentare**: 5V (alimentare prin USB) sau 7-12V (alimentare externă prin pin sau jack).
* **Interfață de programare**: USB-B.

**Caracteristici:**

1. **Conectivitate ușoară**: Poate fi conectat rapid la un computer prin cablu USB pentru programare.
2. **Compatibilitate extinsă**: Este suportat de mediul Arduino IDE, ceea ce permite scrierea de coduri și încărcarea lor direct pe placă.
3. **Pini multipurpose**: Pinii săi pot fi folosiți atât pentru input, cât și pentru output, făcând posibilă conectarea diverselor componente (LED-uri, senzori, motoare).
4. **Dimensiune compactă**: Este suficient de mică pentru proiecte de dimensiuni reduse.

**Utilizări tipice:**

* Proiecte educaționale și prototipuri.
* Sisteme de iluminat cu LED-uri.
* Robotică și automatizări simple.
* Aplicații IoT de bază.



Figura 1

Configurația hardware folosește o placă **Arduino Uno** conectată la un modul **GNSS (Global Navigation Satellite System)** proiectat manual. Modulul GNSS este atașat pentru a primi date de poziționare prin satelit (precum locația, altitudinea sau viteza). Cablurile conectate între GNSS și Arduino asigură comunicarea între componente.

# Descriere scurtă a configurației:

1. **Arduino Uno**:
   * Placă principală pentru procesarea datelor și interfațarea cu computerul.
   * Primește date de la modulul GNSS prin porturile I/O.
2. **Modulul GNSS**:
   * Creat manual pentru a recepționa semnalele GPS/GNSS.
   * Trimite informații despre locație către Arduino.
3. **Legături între componente**:
   * Cablurile conectează GNSS la pinii corespunzători de pe Arduino (de exemplu, TX și RX pentru comunicație serială).
4. **Cablu USB**:
   * Folosit pentru alimentarea Arduino și pentru transferul de date către un computer.

**Ce ar trebui să facă:**

1. **Recepționează date GNSS**: Modulul GNSS preia semnale de la sateliți și calculează

datele de poziție (latitudine, longitudine, altitudine).

1. **Trimite date către Arduino**: Datele GNSS sunt trimise către Arduino pentru procesare.
2. **Aplicații posibile**:
   * Afisarea locației pe un computer.
   * Realizarea unui tracker GPS.
   * Utilizarea pentru automatizarea unor sisteme de navigație.

Această configurație poate fi folosită pentru a testa performanțele modulului GNSS și pentru a-l integra în aplicații mai complexe.

**GNSS (Global Navigation Satellite System)** este o tehnologie globală utilizată pentru a determina poziția, viteza și timpul unui obiect sau utilizator cu o precizie ridicată. Modulul GNSS prezent în această configurație este conceput pentru a recepționa semnale de la constelațiile de sateliți GPS, GLONASS, Galileo sau BeiDou și pentru a transmite informațiile prelucrate către o placă de dezvoltare Arduino Uno, care procesează și afișează datele sub forma coordonatelor geografice.

**Principiul de Funcționare al Modulului GNSS**

Modulul GNSS se bazează pe principiile **trilaterației**, folosind semnale radio recepționate de la sateliți aflați pe orbită. Funcționarea acestuia implică mai multe etape esențiale:

1. **Recepția Semnalelor de la Sateliți**:
   * Modulul GNSS utilizează o antenă specializată pentru a capta semnalele emise de sateliți.
   * Fiecare satelit transmite periodic pachete de date ce includ informații precum:
     + Poziția exactă a satelitului (efemeride).
     + Timpul precis de emisie a semnalului (timestamp).
   * Pentru o localizare corectă, modulul trebuie să recepționeze semnale de la cel puțin **4 sateliți simultan**.
2. **Calculul Poziției**:
   * Modulul măsoară timpul necesar semnalului să parcurgă distanța dintre satelit și receptor.
   * Distanțele calculate față de mai mulți sateliți sunt utilizate pentru determinarea coordonatelor (latitudine, longitudine, altitudine) prin trilaterație.
3. **Prelucrarea Datelor**:
   * Modulul GNSS procesează semnalele interne și extrage coordonatele geografice în format standard, de exemplu, **NMEA (National Marine Electronics Association)**.
   * Linii de date specifice, precum $GPGGA sau $GPRMC, conțin informații precum:
     + Latitudine și longitudine.
     + Ora universală (UTC).
     + Viteza de deplasare.
     + Numărul sateliților folosiți.

**Interfațarea GNSS cu Arduino Uno**

Interfațarea GNSS cu Arduino se face utilizând pinii de comunicație serială, precum **TX** (Transmisie) și **RX** (Recepție), astfel încât datele prelucrate de modulul GNSS să poată fi citite de Arduino.

1. **Configurația Hardware**:
   * Modulul GNSS este conectat la pinii **RX** și **TX** ai Arduino pentru transferul de date.
   * Alimentarea modulului GNSS este asigurată prin pinii de 5V și GND ai plăcii Arduino.
2. **Software și Procesare**:
   * Arduino utilizează o bibliotecă specifică GNSS (cum ar fi **TinyGPS++**) pentru a interpreta datele brute recepționate.
   * Programul încărcat pe Arduino urmărește fluxul de date seriale și extrage informațiile utile (de exemplu, latitudinea și longitudinea).
3. **Afișarea Coordonatelor**:
   * Datele procesate de Arduino sunt afișate pe monitorul serial al IDE Arduino.
   * Formatul afișat conține coordonatele geografice și alte informații relevante, cum ar fi ora sau numărul sateliților activi.

# Rezultate și Aplicații

Această implementare permite obținerea coordonatelor în timp real, care pot fi utilizate pentru diverse aplicații, precum:

* **Navigație și orientare**.
* **Tracking de obiecte sau persoane**.
* **Sisteme autonome de localizare**, cum ar fi drone sau vehicule robotizate.

Proiectul integrează tehnologia GNSS cu puterea de procesare a microcontrolerului Arduino Uno, oferind o soluție practică și accesibilă pentru explorarea și utilizarea sistemelor globale de poziționare.

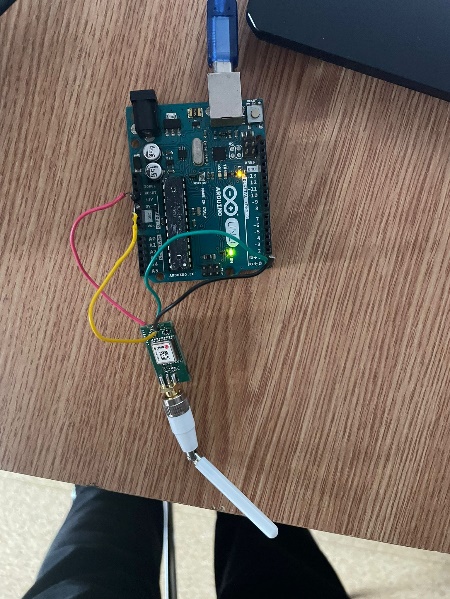


Figura 2

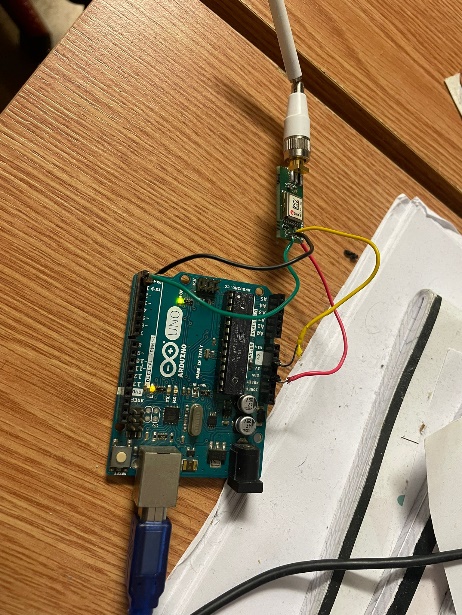


Figura 3

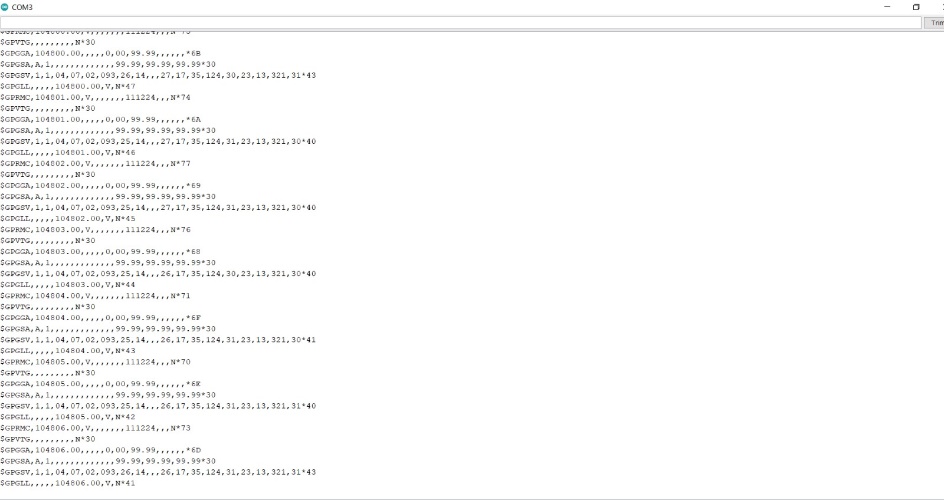


Figura 4

# Proiectarea unui Sistem GNSS pentru Afișarea Locației pe o Interfață Web

Proiectul extinde funcționalitatea sistemului GNSS, utilizând datele de localizare procesate de un Arduino Uno pentru a afișa locația în timp real pe o interfață web. Acest sistem integrează tehnologii hardware și software pentru a realiza un flux de date complet: de la captarea coordonatelor GNSS până la vizualizarea acestora pe o hartă interactivă.

**Descrierea Sistemului**

1. **Captarea datelor GNSS cu Arduino**:
   * Modulul GNSS transmite către Arduino coordonatele geografice în format serial (de exemplu, **NMEA**, cu linii precum $GPGGA și $GPRMC).
   * Programul pe Arduino interpretează aceste date, extrage informațiile esențiale (latitudine, longitudine, ora UTC) și le transmite către un computer prin conexiunea USB.
2. **Transferul Datelor către un Server Web**:
   * Un script rulând pe computer (de exemplu, în **Python**) citește fluxul de date serial de la Arduino și transmite informațiile către un server web.
   * Protocolul utilizat poate fi **HTTP POST** pentru transferul datelor către un endpoint web.
3. **Stocarea Datelor pe Server**:
   * Serverul web primește datele GNSS și le stochează într-o bază de date (de exemplu, **MySQL**) sau într-un fișier JSON. Acest pas permite salvarea istorică a locațiilor pentru utilizări ulterioare, cum ar fi analizarea traseului.
4. **Vizualizarea Datelor pe un Site Web**:
   * Un site web este creat pentru a afișa locația în timp real folosind API-uri de hărți, cum ar fi **Google Maps API** sau **Leaflet.js**.
   * Datele din baza de date sunt preluate și utilizate pentru a plasa un marker pe hartă, indicând poziția curentă.

**Fluxul de Date și Interacțiune**

1. **Arduino → Serial Monitor**:
   * Arduino transmite coordonatele GNSS prin cablu USB către computer, unde pot fi citite în timp real.
2. **Serial Monitor → Server**:
   * Un script în limbaj Python colectează datele și le transmite către server printr-un apel HTTP, asigurând transferul continuu de informații.
3. **Server → Website**:
   * Datele prelucrate sunt trimise către interfața web, unde sunt integrate într-o hartă interactivă.
4. **Afișarea pe Hartă**:
   * Coordonatele sunt marcate pe hartă sub formă de **marker**, care se actualizează periodic pentru a reflecta locația în timp real.

Anexa cod:

# Bibliografie

[1]\*\*\*, [Arduino Team. Arduino Uno Rev3 Technical Specifications](https://www.arduino.cc),Accesat la 4.11.2022

[2] Kaplan, E.; Hegarty, C. (2005). Understanding GPS: Principles and Applications. Artech House. pp. 1–576. doi:10.1007/978-1-4684-9464-2.

[3] Chandola, V.; Banerjee, A.; Kumar, V. (2009). "Anomaly detection: A survey". ACM Computing Surveys. 41 (3): 1–58. doi:10.1145/1541880.1541882.

# Anexă cod

import folium

import serial

# Funcție pentru a extrage coordonatele din propozițiile NMEA

def extrage\_coordnate(line):

if line.startswith("$GPGGA") or line.startswith("$GPGLL") or line.startswith("$GPRMC"):

fields = line.split(",")

if fields[2] and fields[4]: # Verifică dacă latitudinea și longitudinea există

# Conversia coordonatelor în format decimal

lat = float(fields[2]) / 100

lon = float(fields[4]) / 100

lat\_deg = int(lat)

lon\_deg = int(lon)

lat\_min = (lat - lat\_deg) \* 100

lon\_min = (lon - lon\_deg) \* 100

lat\_decimal = lat\_deg + lat\_min / 60

lon\_decimal = lon\_deg + lon\_min / 60

# Corectează direcția (N/S, E/W)

if fields[3] == "S":

lat\_decimal = -lat\_decimal

if fields[5] == "W":

lon\_decimal = -lon\_decimal

return lat\_decimal, lon\_decimal

return None, None

# Configurare port serial

ser = serial.Serial('COM3', 9600, timeout=1) # Înlocuiește 'COM3' cu portul tău

while True:

try:

# Citește linia brută de la portul serial

raw\_line = ser.readline()

line = raw\_line.decode('latin-1').strip() # Decodifică linia

# Extrage coordonatele din linia curentă

lat, lon = extrage\_coordnate(line)

if lat and lon:

print(f"Coordonate valide: Latitudine = {lat}, Longitudine = {lon}")

# Creează harta centrată pe coordonatele date

harta = folium.Map(location=[lat, lon], zoom\_start=15)

# Adaugă un marker pe hartă

folium.Marker(

[lat, lon],

popup=f"Lat: {lat}, Lon: {lon}",

tooltip="Locație GPS"

).add\_to(harta)

# Salvează harta într-un fișier HTML

harta.save("harta.html")

print("Harta a fost generată și salvată ca 'harta.html'. Deschide acest fișier într-un browser!")

break # Ieși din buclă după ce salvezi harta

except Exception as e:

print(f"Eroare: {e}")