UDK 004.78:629.052.9 • 005.94:629.052.9 • 621.3:629.052.9

GPS sustav za praćenje interaktivnom kartom

Luka Žužić[a], Deni Vale[b], Patrik Radolović[c], Ivan Simetić[d]

a Bacc. ing. mech., student, Istarsko Veleučilište – Università Istriana di scienze applicate, Preradovićeva 9D, 52100 Pula, lzuzic@iv.hr

b Mag. phys., predavač, Istarsko Veleučilište – Università Istriana di scienze applicate, Preradovićeva 9D, 52100 Pula, dvale@iv.hr

c Mag. ing. mech., Istarsko veleučilište – Università Istriana di scienze applicate, Preradovićeva 9D, 52100 Pula, pradolovi@iv.hr

d Univ. mag. ing. mech., predavač, Istarsko veleučilište – Università Istriana di scienze applicate, Preradovićeva 9D, 52100 Pula, isimetic@iv.hr

Sažetak

U radu je prikazan razvoj funkcionalnog prototipa GPS sustava za praćenje lokacije s prikazom podataka na interaktivnoj karti. U teorijskom dijelu rada analizirane su Arduino pločice, ESP32 i Raspberry Pi Zero 2 W kao moguće platforme za implementaciju sustava, dok je za praktičnu izvedbu prototipa odabran ESP32 mikroupravljač. Sustav se temelji na povezivanju NEO-7M GPS modula s ESP32 mikroupravljačem, koji prikuplja geografske koordinate, obrađuje ih i šalje prema udaljenom poslužitelju putem internetske veze. Za prijenos podataka koristi se HTTP komunikacija, pri čemu se koordinate šalju u JSON formatu. Poslužiteljski dio sustava izrađen je pomoću Flask web okvira, dok je za vizualizaciju lokacije korištena Leaflet biblioteka s OpenStreetMap kartografskom podlogom. Na web sučelju trenutačna pozicija uređaja prikazana je pomoću jasno vidljive oznake na karti, čime je korisniku omogućen jednostavan i pregledan uvid u lokaciju praćenog objekta. Prototip je testiran na otvorenom prostoru, pri čemu su GPS podaci slani prema poslužitelju svakih 10 sekundi. Tijekom testiranja sustav je radio stabilno te nisu zabilježena kašnjenja u prijenosu ni poteškoće u komunikaciji. Prikazani sustav potvrđuje da je korištenjem dostupnih hardverskih i softverskih alata moguće izraditi jednostavno, prilagodljivo i funkcionalno IoT rješenje za daljinsko praćenje lokacije u stvarnom vremenu. Sustav je pogodan za edukacijske svrhe, razvoj prototipova te kao osnova za daljnje nadogradnje, poput spremanja povijesti kretanja, prikaza rute ili praćenja više uređaja istovremeno.

Ključne riječi: *NEO-7M GPS, ESP32, praćenje lokacije, interaktivna karta, internet stvari*

1. Uvod

Jedno od najvećih tehnoloških postignuća čovječanstva, koje omogućuje determinaciju geografskog položaja s iznimno visokom razinom preciznosti, očituje se u suvremenom sustavu globalnog pozicioniranja, poznatijem pod akronimom GPS (engl. *Global Positioning System*) (French, 1996). GPS je

satelitski navigacijski sustav čiji razvoj započinje ranih 1970-ih pod okriljem Ministarstva obrane Sjedinjenih Američkih Država. Iako je primarno razvijan isključivo za vojne svrhe, sustav je naknadno liberaliziran te je danas u potpunosti dostupan i civilnom sektoru (El-Rabbany, 2002). Arhitektura GPS-a objedinjuje složena načela matematike, fizike i inženjerstva, a za njegovo je razumijevanje potrebno analizirati tri osnovne komponente: svemirski, upravljački i korisnički segment.

Svemirski segment čine sateliti koji orbitiraju oko Zemlje na visini od približno 19 300 km i prosječnom brzinom od 11 300 km/h, pri čemu im je za punu orbitalnu revoluciju potrebno oko 12 sati. Konstelacija je prostorno raspoređena na način da je na bilo kojoj lokaciji na Zemlji, u svakom trenutku, osigurana vidljivost i prijam signala s najmanje šest satelita. Drugu komponentu čine upravljački segment, odnosno automatizirane zemaljske stanice raspoređene diljem svijeta koje kontinuirano nadziru rad satelita. Prikupljeni telemetrijski podaci prosljeđuju se u Glavnu kontrolnu stanicu (engl. *Master Control Station*) u zrakoplovnoj bazi Schriever u Coloradu. Ondje se provodi računalna obrada podataka radi izračuna efemerida (predviđenih pozicija satelita) i korekcije vremenskih pogrešaka atomskih satova. Glavna kontrolna stanica aktivira mikropogonske sustave na satelitima radi korekcije njihovih orbita i održavanja pravilne putanje. Treći i posljednji dio sustava čine GPS prijamnici, odnosno korisnički segment, koji pasivno primaju satelitske signale radi određivanja vlastitih prostornih koordinata. Udaljenost do pojedinog satelita određuje se na temelju vremena potrebnog da radiosignal prođe put od satelita do prijamnika. S obzirom na to da satelit i prijamnik generiraju identičnu pseudonasumičnu kodnu sekvencu, prijamnik korelira primljeni signal s lokalno generiranim kodom kako bi utvrdio vremensko kašnjenje, dok se princip Dopplerovog pomaka koristi za izračun brzine kretanja objekta (McNamara, 2004).

Suvremeni GPS pokrenuo je renesansu u područjima navigacije, geodezije i kartografije. Inovativni sustavi i aplikacije temeljeni na ovoj tehnologiji danas se implementiraju u upravljanju transportnim flotama, preciznoj poljoprivredi, gospodarenju prirodnim resursima, pomorskom prometu, istraživanju mineralnih sirovina, sustavima za ublažavanje posljedica katastrofa te kao ključan alat u različitim sigurnosnim protokolima (French, 1996). Vođen tim tehnološkim napretkom, ovaj se rad bavi razvojem i praktičnom implementacijom GPS sustava za praćenje lokacije s interaktivnom kartom. U teorijskom dijelu rada analizirane su Arduino pločice, Raspberry Pi Zero 2 W i ESP32 kao moguće hardverske platforme za realizaciju sustava. Proveden je njihov komparativni prikaz s obzirom na procesorsku snagu, komunikacijske mogućnosti, dostupnost perifernih sučelja, energetsku učinkovitost, cijenu i prikladnost za IoT primjene. Na temelju provedene analize za praktičnu implementaciju sustava odabran je ESP32 mikroupravljač, dok ostale analizirane platforme nisu istodobno ugrađene u izrađeni prototip. Nadalje, opisana je integracija GPS modula, projektiranje sheme povezivanja komponenti te razvoj klijentske aplikacije zadužene za mrežnu komunikaciju. Naposljetku, rad obuhvaća proces izrade web-sučelja, konfiguraciju poslužiteljske aplikacije unutar okvira Flask te implementaciju interaktivnog kartografskog sučelja namijenjenog vizualizaciji i prikazu prikupljenih GPS podataka (Flask, bez dat.; Leaflet, bez dat.).

2. Odabir mikroupravljača

Pri odabiru mikroupravljača za razvoj interneta stvari (engl. *Internet of Things,* IoT) uzimaju se u obzir karakteristike kao što su procesorska snaga, mogućnosti komunikacije, dostupnost periferija, cijena te energetska učinkovitost. Postoji mnogo komercijalno dostupnih rješenja, od kojih su među najpopularnijima platforme Arduino, Raspberry Pi Zero 2 W i ESP32. Svaka od ovih platformi ima svoje specifične prednosti i ograničenja, što ih čini pogodnima za različite primjene. Arduino ploče nude jednostavnost i nisku potrošnju, ali često zahtijevaju dodatne module za bežičnu komunikaciju. Raspberry Pi Zero 2 W, kao kompaktno mini računalo, omogućava obradu složenijih zadataka i pruža veću fleksibilnost, iako je manje pogodan za aplikacije osjetljive na potrošnju energije. S druge strane, ESP32 nudi snažan procesor s integriranim Wi-Fi i Bluetooth modulom, što ga čini idealnim izborom za mnoge IoT aplikacije.

2.1. Arduino ploče

Arduino predstavlja otvorenu platformu koja integrira hardver i softver za razvoj prototipova i jednostavnih projekata, namijenjenu širokom spektru korisnika, od entuzijasta i studenata do profesionalnih inženjera (Galadima, 2014). Njegova glavna snaga leži u jednostavnom programiranju mikroupravljača, malog računalnog čipa koji upravlja ulazno-izlaznim signalima i omogućava interakciju s raznovrsnim elektroničkim komponentama poput senzora, motora i LED-a (Hodgson, 2020). Ova platforma koristi intuitivan model ulaza i izlaza (I/O) za povezivanje s vanjskim uređajima, što značajno olakšava razvoj interaktivnih sustava. Nastala je 2005. godine nakon višegodišnjeg istraživanja na Ivrea Interaction Design Institutu u Italiji, s ciljem pojednostavljenja izrade interaktivnih projekata i pružanja obrazovnog alata (Pan & Zhu, 2017). Zahvaljujući svojoj pristupačnosti i jednostavnosti, Arduino je stekao veliku popularnost kako među početnicima, tako i kod stručnjaka koji brzo žele implementirati rješenja iz područja robotike, automatizacije, Interneta stvari i drugih inovativnih tehnologija.

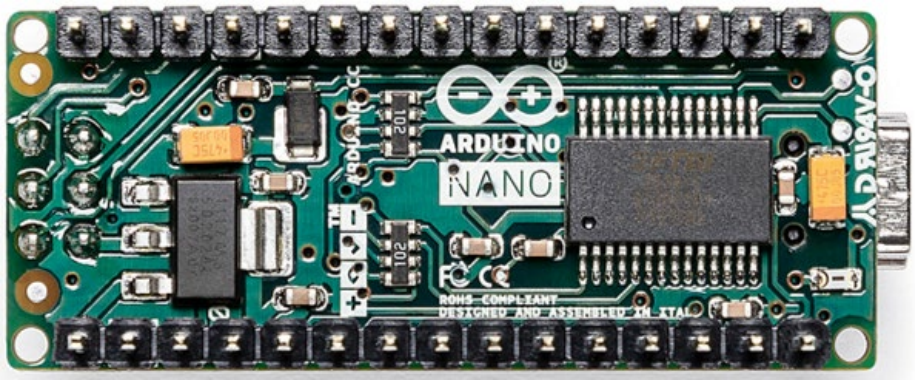
2.1.1. Arduino Nano

Arduino Nano je kompaktna, a ujedno i funkcionalno bogata ploča (slika 1.), temeljena na Atmega328P mikroupravljaču s radnim taktom od 16 MHz. Iako nudi funkcionalnosti slične Arduino Unu, Nano se ističe svojim značajno manjim dimenzijama, što ga čini idealnim za projekte s ograničenim prostorom. Ploča raspolaže s 14 digitalnih ulaza/izlaza, od kojih šest podržava funkcionalnost modulatora širine impulsa (PWM), te ima i 8 analognih ulaza. Osim toga, Arduino Nano posjeduje 2 KB SRAM-a, 32 KB flash memorije i 1 KB EEPROM-a, čime osigurava dovoljno memorijskih resursa za većinu manjih aplikacija. Napajanje ploče moguće je osigurati putem USB kabela ili vanjskog izvora, što dodatno doprinosi fleksibilnosti u raznim projektima (Arduino, bez dat.-b).

2.1.2. Arduino Uno

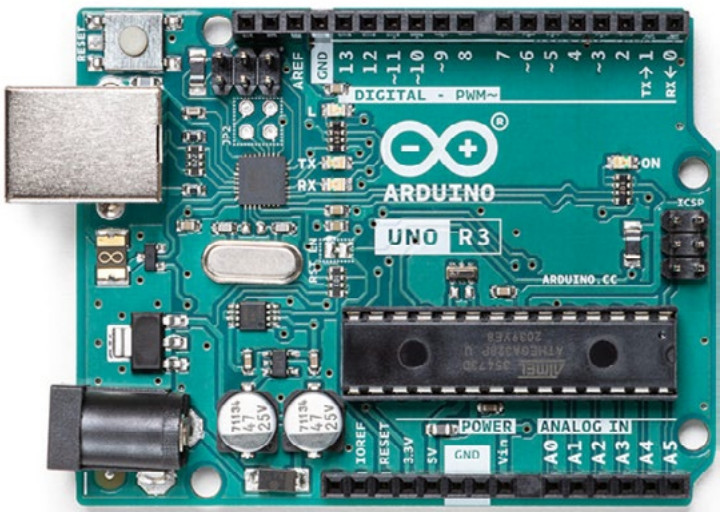
Arduino Uno (slika 2.) je jedna od najpopularnijih Arduino ploča, često preporučena početnicima zbog svoje jednostavnosti i pouzdanosti u realizaciji osnovnih projekata. Ova ploča, temeljena na Atmega328P mikroupravljaču s radnom frekvencijom od 16 MHz, nudi uravnotežene performanse uz 2 KB SRAM-a, 32 KB flash memorije i 1 KB EEPROM-a. Arduino Uno opremljen je s 14 digitalnih ulaza/izlaza, pri čemu je 6 tih pinova sposobni raditi kao PWM izlazi, te s 6 analognih ulaza, omogućujući širok raspon primjena u interaktivnim i senzoriziranim projektima. Ploča se može napajati putem USB kabela ili vanjskog izvora napajanja, čime se povećava fleksibilnost implementacije u različitim okruženjima (Arduino, bez dat.-c).

Slika 1. Prikaz Arduino Nano ploče



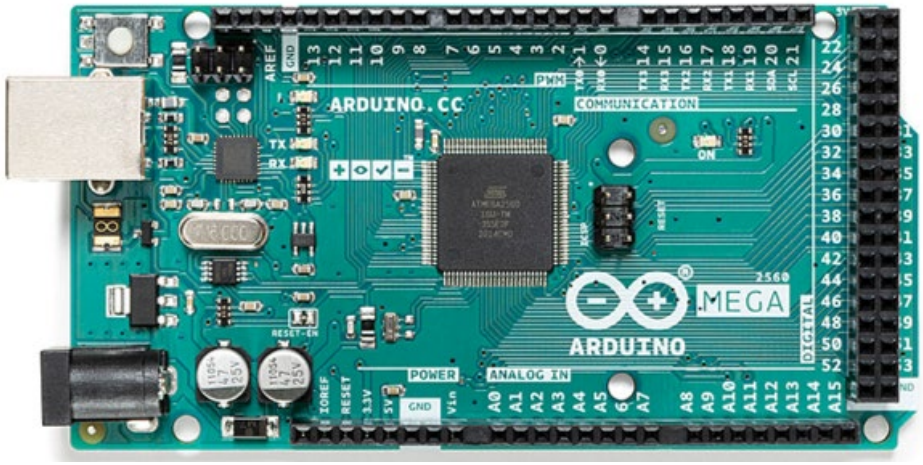
Izvor: https://store.arduino.cc/products/arduino-nano, preuzeto 14. travnja 2025.

Slika 2. Prikaz Arduino Uno ploče



Izvor: https://store.arduino.cc/products/arduino-uno-rev3, preuzeto 15. travnja 2025.

Slika 3. Prikaz Arduino Mega ploče



Izvor: https://store.arduino.cc/products/arduino-mega-2560-rev3, preuzeto 15. travnja 2025.

2.1.3. Arduino Mega

Arduino Mega (slika 3.) predstavlja nadograđenu verziju Arduino Uno ploče, osmišljenu za kompleksnije projekte koji zahtijevaju veći broj ulazno-izlaznih priključaka i veću memoriju. Bazirana na Atmega2560 mikroupravljaču, Mega nudi 54 digitalna ulaza/izlaza, od kojih 15 podržava PWM, te 16 analognih ulaza, što omogućava istovremeni rad s mnogim senzorima i aktuatorima. Uz to, ploča raspolaže s 4 serijska porta, čime se dodatno olakšava komunikacija s vanjskim uređajima i modulima. S 256 KB flash memorije, 8 KB SRAM-a i 4 KB EEPROM-a, Arduino Mega je idealan za aplikacije koje zahtijevaju obradu velikih količina podataka i implementaciju složenih algoritama (Arduino, bez dat.-a).

2.1.4. Usporedba ploča

Usporedbom svih Arduino ploča (tablica 1.), kao optimalan izbor za ovaj projekt sugerira se Arduino Nano zbog svojih dimenzija koje omogućuju integraciju unutar manjih kućišta, niske potrošnje energije te također zadovoljava sve tehničke zahtjeve projekta.

Tablica 1: Usporedba različitih Arduino ploča

Pločica		Arduino Uno R3	Arduino Nano	Mega2560 Rev3
Mikroupravljač		ATmega328p	ATmega328p	ATmega2560
Vrsta USB-a		USB-B	Mini-B USB	USB-B
Ulazno/izlazni priključci	**Digitalni**	14	14	54
	Analogni ulaz	6	8	16
	Analogni izlaz	0	0	0
	PWM	6	6	15
Komunikacija	**UART**	Da	Da	Da
	I2C	Da	Da	Da
	SPI	Da	Da	Da
	CAN	Ne	Ne	Ne
	Bluetooth	Ne	Ne	Ne
	WiFi	Ne	Ne	Ne
Napajanje	**U/I napon (V)**	5	5	5
	Nominalni ulazni napon	7 - 12	8 - 12	9 - 12
	Struja po U/I priključku (mA)	20	20	20
	Vrsta priključka napajanja	5.5 x 2.5 priključak (Barrel Jack)	GPIO priključak	5.5 x 2.5 priključak (Barrel Jack)
	Baterijsko napajanje	Ne	Ne	Ne
Brzina procesora (MHz)		16	16	16
USB u serijski pretvarač		ATmega16U2 16 MHz	ATmega16U2 16 MHz	ATmega16U2 16 MHz
Memorija	**Flash (KB)**	32	32	256
	SRAM (KB)	2	2	8
	EEPROM (KB)	1	1	4
Dimenzije	**Masa (g)**	25	5	37
	Širina (mm)	53.4	18	53.3
	Dužina (mm)	68.6	45	101.5

Izvor: Radolović, P. (2023).

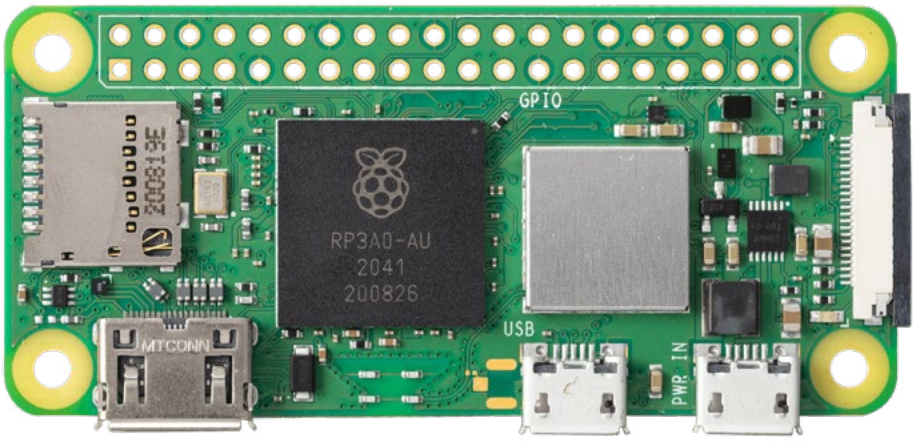
2.2. Raspberry Pi Zero 2 W

Raspberry Pi Zero 2 W (slika 4.) je kompaktno mini računalo osmišljeno za razvoj sofisticiranih prototipova i jednostavnih projekata, namijenjeno širokom spektru korisnika, od hobista do profesionalnih programera. Njegova ključna snaga leži u potpunom operativnom sustavu, obično Linuxu, koji omogućava izvođenje kompleksnih zadataka, obradu podataka i pristup širokom spektru softverskih alata. Uređaj nudi intuitivno upravljanje ulazno-izlaznim signalima, čime se olakšava integracija s raznovrsnim perifernim uređajima poput senzora, kamera i aktuatora. Kao evolucija prethodnih modela, Raspberry Pi Zero 2 W posjeduje poboljšanu procesorsku snagu i ugrađenu bežičnu komunikaciju, čime je idealan za razvoj IoT aplikacija. Raspberry Pi Zero 2 W je bolji sugerirani izbor za ovaj projekt u odnosu na Arduino Nano zbog integriranog Wi-Fi modula i većeg broja GPIO pinova, što omogućava jednostavniju i fleksibilniju integraciju s vanjskim uređajima. Dok Arduino Nano nudi osnovne funkcionalnosti za rad s elektroničkim komponentama, Pi Zero 2 W pruža bolju povezanost, što omogućuje pouzdan prijenos podataka u stvarnom vremenu. Veći broj GPIO pinova također omogućava spajanje većeg broja senzora i aktuatora, čime se projekt može proširiti i prilagoditi specifičnim potrebama bez dodatnih hardverskih ograničenja (Rasperry Pi, bez dat.).

2.3. ESP32

ESP32 (slika 5.) je višenamjenski mikroupravljač koji nudi integrirane Wi-Fi (802.11 b/g/n) i Bluetooth (v4.2 BR/EDR i BLE) module za učinkovitu bežičnu komunikaciju s drugim uređajima, što ga čini optimalnim izborom za IoT aplikacije. Posjeduje dual-core Xtensa LX6 mikroprocesor s radnim taktom do 240 MHz, omogućavajući istovremenu paralelnu obradu GPS podataka i komunikacijskih zadataka u stvarnom vremenu. Ugrađena memorija uključuje 520 KB RAM-a i 4 MB flash memorije, što omogućuje izvršavanje složenijih algoritama i pohranu većih količina podataka (Espressif Systems, bez dat.). Njegove kompaktne dimenzije, pristupačna cijena i niska energetska potrošnja čine ga idealnim izborom za sustave koji zahtijevaju kontinuirano praćenje lokacije, posebice u baterijskim aplikacijama. Dodatna prednost ESP32 platforme leži u širokoj dostupnosti softverskih biblioteka, velikoj podršci zajednice programera otvorenog koda (engl. *open source*) te kompatibilnosti s Arduino razvojnim okruženjem, što znatno pojednostavljuje integraciju i ubrzava razvoj projekata. Fleksibilan raspored GPIO pinova i bogat skup perifernih sučelja poput UART, SPI, I2C, ADC i DAC dodatno olakšavaju povezivanje različitih senzora i modula, omogućujući lako prilagođavanje specifičnim zahtjevima GPS sustava za praćenje. Iako Raspberry Pi Zero 2 W nudi mogućnosti punog operativnog sustava, ESP32 je za ovaj projekt bolji izbor zbog nižih troškova, energetske učinkovitosti, manje složenosti i jednostavne integracije s postojećim Arduino sustavima.

Slika 4. Raspberry Pi Zero 2 W



Izvor: https://www.raspberrypi.com/products/raspberry-pi-zero-2-w/, preuzeto 15. travnja 2025.

Slika 5: ESP32.



Izvor: https://www.mouser.bg/ProductDetail/Espressif-Systems/ESP32-DevKitC-32D?qs=%252BEew9%252B0nqrDsObWEpDx6YQ%3D%3D, preuzeto 17. travnja 2025.

3. Odabir GPS modula

GPS moduli predstavljaju inovativnu tehnologiju koja omogućuje precizno određivanje položaja i brzine, te se stoga koriste u širokom spektru aplikacija, od navigacijskih sustava u automobilima i zrakoplovima do sustava za praćenje dostavnih vozila. U tablici 2. prikazano je nekoliko odabranih najpopularnijih GPS modula prilagođenih za implementaciju s mikroupravljačima i usporedba njihovih osnovnih karakteristika.. Ovi moduli primaju signale iz mreže satelita koji kruže oko Zemlje, koristeći princip trilateracije kako bi izračunali geografske koordinate i nadmorsku visinu. Na taj način, GPS moduli omogućuju korisnicima pristup informacijama potrebnim za navigaciju i planiranje rute, a njihov rad oslanja se na matematičke algoritme koji analiziraju signale primljene od najmanje četiri aktivna satelita. Prilikom integracije GPS modula s mikroupravljačima, važno je odabrati rješenje koje odgovara specifičnim zahtjevima projekta, bilo da se radi o sustavu za praćenje, navigaciji ili različitim IoT aplikacijama.

GPS modul NEO-7M (slika 6.) pokazao se idealnim izborom za ovaj projekt. Njegovo poboljšano vrijeme akvizicije signala i povećana osjetljivost omogućuju brzo i zadovoljavajuće određivanje položaja, čak i u uvjetima s ograničenim pogledom na nebo, kao što su urbana područja. Uz to, pruža visoku preciznost uz prihvatljivu cijenu (u-blox, 2014). Drugi modeli poput NEO-M9N mogu biti prikladni za projekte koji zahtijevaju višu razinu preciznosti i brzine osvježavanja, ali uz viši trošak implementacije.

Tablica 2. Usporedba GPS modula

GPS modul	Brzina osvježavanja	Točnost	Značajke	Cjenovni rang
NEO-6M	1 Hz	±2.5 m	Jednostavna integracija, osnovna funkcionalnost	Niska
NEO-7M	5 Hz	±2 m	Brži dohvat signala od NEO-6M, poboljšana osjetljivost i stabilnije performanse	Srednja
NEO-M8N	10 Hz	±1.8 m	Podrška za više satelitskih sustava (GPS, GLONASS, Galileo)	Viša
NEO-M9N	25 Hz	±1.5 m	Multi-band tehnologija, bolja preciznost	Visoka

Izvor: autorska sinteza.

Slika 6. NEO-7M GPS modul



Izvor: autori.

4. Shema spajanja

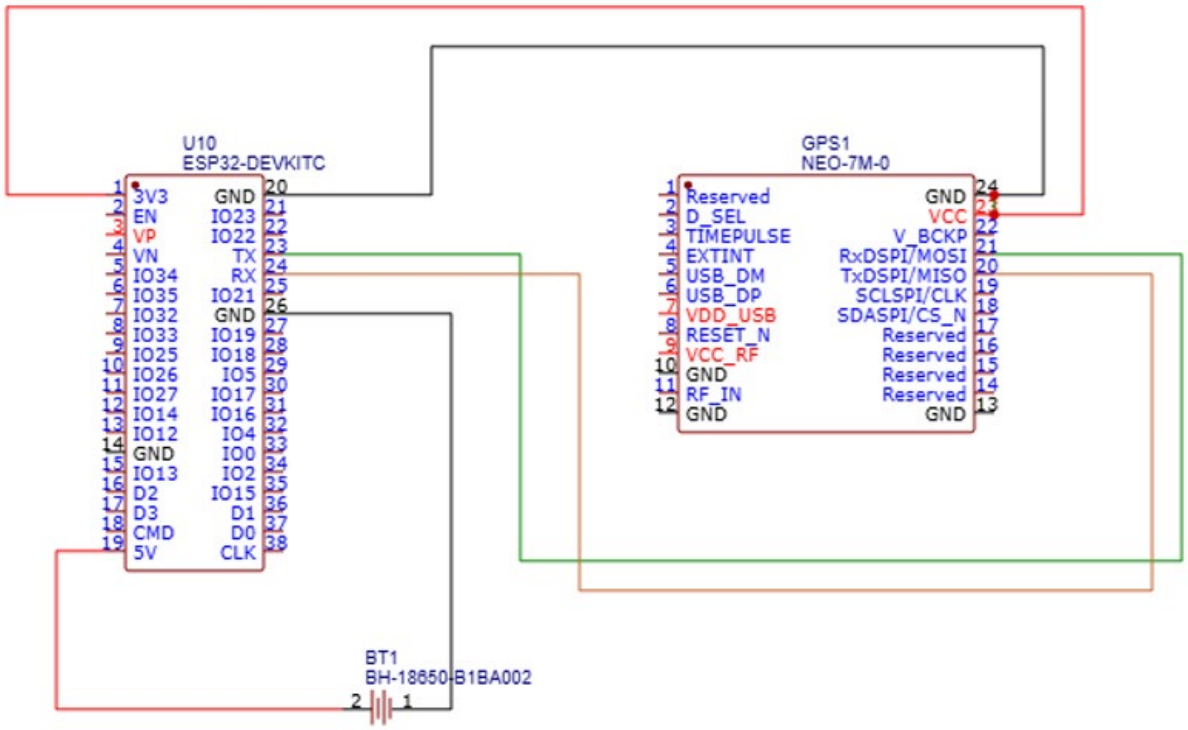
NEO-7M GPS modul komunicira putem UART (engl. *Universal Asynchronous Receiver-Transmitter*) protokola, što znači da za povezivanje s ESP32 mikroupravljačem treba koristiti UART pinove za primanje (RX) i slanje (TX) podataka. Potrebno je obratiti pažnju na pravilno povezivanje pinova između uređaja: TX pin GPS modula povezuje se na RX pin mikroupravljača, dok se RX pin GPS modula povezuje na TX pin mikroupravljača. Osim UART komunikacije, modul zahtijeva odgovarajuće napajanje u rasponu od 3.3 V do 5 V te zajedničko uzemljenje (GND). Tablica 3. detaljno prikazuje ove veze, a potom je predočen shematski prikaz spajanja komponenti (slika 7.). Za dodatnu preciznost vremenske sinkronizacije, NEO-7M modul ima opcionalni PPS (engl. *pulse-per-second*) pin, koji se također može spojiti na jedan od GPIO pinova ESP32, no to nije nužno za osnovni rad GPS sustava.

Tablica 3. Povezivanje GPS modula na ESP32 mikroupravljač

NEO-7M pin	ESP32 pin	Opis
VCC	3V3 (ili 5 V)	Napajanje GPS modula
GND	GND	Zajedničko uzemljenje sustava
TX	GPIO16 (RX2)	Slanje GPS podataka prema ESP32
RX	GPIO17 (TX2)	Primanje konfiguracijskih naredbi s ESP32 (opcionalno)
PPS	GPIO4 (opcionalno)	Puls u sekundi za preciznu vremensku sinkronizaciju

Izvor: autori.

Slika 7. Shema spajanja



Izvor: autori.

5. Postavljanje klijenta

Klijentski dio sustava odnosi se na hardversko-programsku cjelinu koja prikuplja GPS podatke i šalje ih prema udaljenom poslužitelju. Budući da sustav mora omogućiti praćenje lokacije izvan dosega lokalne Wi-Fi mreže, posebna pozornost posvećena je načinu povezivanja na mrežu i pouzdanom slanju podataka prema web aplikaciji. U nastavku je opisano uspostavljanje mrežne veze, osnovna programska struktura klijentskog uređaja te način na koji se GPS podaci pripremaju i šalju poslužitelju. Time se prikazuje kako klijentski dio sustava povezuje fizičko očitanje lokacije s kasnijom vizualizacijom na interaktivnoj karti.

5.1. Povezivanje na mrežu

Za povezivanje sustava s internetom koristi se Qualcomm MDM9600 4G LTE modem, koji omogućuje brzu mobilnu internetsku vezu putem LTE mreže (Qualcomm, 2009). Modem je u originalnom izdanju prilagođen za USB, ali za potrebe ovog projekta jedan od autora ga je modificirao tako da se direktno napaja iz izvora od 5 V. Internetska mreža uspostavljena na ovaj način služi kao pristupna točka za ESP32 mikroupravljač, koji koristi integrirani Wi-Fi modul za povezivanje s modemom. Ovakva konfiguracija omogućuje ESP32 modulu slanje GPS podataka prikupljene pomoću NEO-7M GPS modula prema udaljenom poslužitelju putem mobilne mreže, čime je osigurana pouzdana komunikacija bez potrebe za lokalnom Wi-Fi infrastrukturom. Stabilnost i brzina ove veze ključni su za sustave za praćenje lokacije u stvarnom vremenu, osobito u okruženjima gdje je pristup klasičnoj Wi-Fi mreži ograničen ili nepostojeći.

5.2. Programski kod

U ovom se popoglavlju razmatra programska implementacija sustava za praćenje temeljenog na ESP32 mikroupravljaču. Ova implementacija obuhvaća uspostavljanje serijske komunikacije, dekodiranje GPS podataka, povezivanje s internetskom infrastrukturom pomoću Qualcomm MDM9600 4G LTE modema, te prijenos podataka prema udaljenom poslužitelju radi obrade podataka i prikaza lokacije na web sučelju. U nastavku se detaljno analiziraju funkcionalni blokovi programskog

koda, pružajući uvid u njihovu ulogu unutar sustava. Za potrebe inicijalne konfiguracije sustava uključene su sljedeće biblioteke:

```
#include <WiFi.h>
#include <TinyGPSPlus.h>
#include <HardwareSerial.h>
#include <HTTPClient.h>
```

Biblioteka „WiFi.h" omogućuje uspostavu internetske veze putem ugrađenog Wi-Fi modula ESP32 mikroupravljača, dok biblioteka „TinyGPSPlus.h" pruža funkcionalnosti za dekodiranje NMEA (engl. *National Marine Electronics Association*) podataka koje generira GPS modul (Hart, bez dat.). „HardwareSerial.h" omogućava serijsku komunikaciju između ESP32 mikroupravljača i GPS modula, dok „HTTPClient.h" omogućuje HTTP komunikaciju i slanje podataka na udaljeni poslužitelj. Deklaraciju globalnih varijabli za Wi-Fi konfiguraciju (ssid, password), konfiguraciju javne adrese poslužitelja (server_ip, server_port, endpointRoute), te pokretanje serijske komunikacije s GPS modulom pomoću biblioteke „HardwareSerial.h", potrebno je napraviti na sljedeći način:

```
const char* ssid = <Vaš SSID>;
const char* password = <Vaša zaporka>;

const char* server_ip = <Vaš ip>;
const uint16_t server_port = <Vaš port>;
const char* endpointRoute = </Vaša ruta>;

TinyGPSPlus gps;
HardwareSerial GPS_Serial(1);
```

Deklaracija varijabli za mjerenje vremena od posljednjeg slanja koordinata (lastSentTime), koordinata (latitude i longitude), te provjera šalje li GPS modul koordinate mikroupravljaču (gpsDataAvailable):

```
unsigned long lastSentTime = 0;
double latitude = 0.0;
double longitude = 0.0;
bool gpsDataAvailable = false;
```

U setup() funkciji za inicijalizaciju konfiguriraju se serijske veze i uspostavlja Wi-Fi konekcija, ESP32 se prvo povezuje na Wi-Fi mrežu koristeći zadane podatke, a zatim pokreće serijsku vezu brzinom od 9600 bps.

```
void setup() {
  Serial.begin(115200);
  GPS_Serial.begin(9600, SERIAL_8N1, 16, 17);

  WiFi.begin(ssid, password);
  Serial.print("Povezivanje na WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nSpojeno na WiFi");
}
```

U glavnoj funkciji, odnosno loop() funkciji kontinuirano se čitaju podaci s GPS modula putem UART veze. Funkcija najprije započinje provjerom serijskog međuspremnika (engl. *serial buffer*) putem uvjeta GPS_Serial.available() > 0. Sve dok u međuspremniku postoje bajtovi koje je GPS modul poslao, petlja se izvršava i čita znak po znak pomoću GPS_Serial.read(). Ova faza osigurava kontinuiran prihvat sirovih NMEA rečenica s hardverskog modula bez gubitka podataka, kako slijedi:

```
  while (GPS_Serial.available() > 0) {
    if (gps.encode(GPS_Serial.read())) {
      . . .
```

Pročitani bajtovi prosljeđuju se metodi gps.encode(), koja pripada biblioteci za obradu GPS podataka (TinyGPSPlus). Ova funkcija analizira tekstualne nizove i vraća istinitu vrijednost tek kada uspješno parsira kompletnu NMEA rečenicu. Unutar tog uvjeta, sustav odmah provjerava valjanost koordinata naredbom gps.location.isValid():

```
      if (gps.location.isValid()) {
        latitude = gps.location.lat();
        longitude = gps.location.lng();
        gpsDataAvailable = true;
        . . .
```

Ako modul još uvijek nema stabilnu satelitsku vezu (engl. *GPS fix*), postavlja se zastavica gpsDataAvailable = false. Kada se potvrdi valjanost

signala, iz objekta gps.location izdvajaju se trenutačne vrijednosti geografske širine (lat()) i duljine (lng()). Te se vrijednosti pohranjuju u globalne varijable latitude i longitude, a kontrolna zastavica gpsDataAvailable postavlja se na true, što signalizira da sustav raspolaže važećim prostornim podacima spremnima za daljnju obradu. Umjesto korištenja blokirajuće funkcije delay(), koja bi prekinula rad mikroupravljača i uzrokovala gubitak GPS podataka, implementiran je nesinkroni vremenski brojač. Pomoću izraza millis() - lastSentTime > 10 000 provjerava se je li od posljednjeg uspješnog slanja prošlo više od 10 sekundi:

```
if (millis() - lastSentTime > 10000) {
  lastSentTime = millis();
...
```

ESP32 redovito prima i obrađuje podatke s GPS modula. Ukoliko su podaci ispravni, pripremaju se u JSON formatu te se šalju putem HTTP POST zahtjeva na poslužitelj svakih 10 sekundi, te provjerava jesu li podaci uspješno obrađeni od poslužitelja. Ako je uvjet ispunjen, varijabla lastSentTime ažurira se na trenutačnu vrijednost, a proces prelazi u fazu mrežnog prijenosa. Prije samog slanja, podaci se strukturiraju u format prikladan za web-poslužitelje. Koordinate se pretvaraju u tekstualni niz (string) s preciznošću od 6 decimalnih mjesta te se ručno uokviruju u standardni JSON paket s ključevima „latitude“ i „longitude“:

```
String jsonPayload = "{\"latitude\":
" + String(latitude, 6) + ", \"longitude\": " +
String(longitude, 6) + "}";
```

Istovremeno, kôd dinamički spaja dijelove tekstualnih nizova (IP adresu poslužitelja, port i rutu) kako bi formirao kompletan odredišni URL za HTTP protokol:

```
String url = "http://";
url += server_ip;
url += ":";
url += String(server_port);
url += endpointRoute;

HTTPClient http;
http.begin(url);
http.addHeader("Content-Type",
"application/json");

int httpResponseCode = http.
POST(jsonPayload);

http.end();
...
```

U posljednjoj fazi instancira se objekt HTTPClient te se pokreće veza prema generiranom URL-u. Zaglavlje zahtjeva definira se kao application/json, nakon čega se metoda http.POST(jsonPayload) brine za fizičko slanje paketa preko mreže. Sustav prihvaća povratni kôd poslužitelja (httpResponseCode); ako je kôd veći od nule (npr. HTTP 200), u serijski monitor ispisuje se uspješan odgovor servera, dok se u suprotnom ispisuje dijagnostička pogreška. Na samom kraju, veza se formalno zatvara naredbom http.end(). Potpuni kod funkcije loop() nalazi se u Dodatku.

6. Postavljanje web sučelja

Web sučelje predstavlja završni dio sustava jer korisniku omogućuje pregled lokacijskih podataka u razumljivom i vizualno jasnom obliku. Nakon što ESP32 mikroupravljač prikupi GPS koordinate i pošalje ih prema poslužitelju, potrebno ih je obraditi, privremeno pohraniti i prikazati na interaktivnoj karti. Zbog toga je web dio sustava podijeljen na poslužiteljsku aplikaciju, izrađenu pomoću Flask okvira (Pallets, bez dat.), i korisnički dio, izrađen pomoću Leaflet biblioteke za prikaz karte (Leaflet, bez dat.).

U ovom poglavlju opisuje se način na koji Flask aplikacija prima koordinate od ESP32 mikroupravljača i omogućuje njihovo dohvaćanje putem web sučelja. Nakon toga prikazuje se implementacija

interaktivne karte, na kojoj se trenutna lokacija uređaja označava vidljivim markerom. Takva struktura omogućuje jasno razdvajanje obrade podataka na strani poslužitelja i vizualizacije podataka na strani korisnika.

6.1. Postavljanje Flask aplikacije

Za prijenos GPS koordinata na web sučelje korišten je Flask, fleksibilan Python web okvir (engl. *framework*), koji omogućuje jednostavan razvoj pozadinskih procesa za web aplikacije (Leaflet, bez dat.). Flask aplikacija sastoji se od ključne funkcije koja omogućuju prijem koordinata sa ESP32 mikroupravljača te njihov prijenos web aplikaciji. Postavlja se na način:

```
def receive_coordinates():
    global latest_coordinates
    data = request.get_json()

    if not data or 'latitude' not in data or
'longitude' not in data:
        return jsonify({"success": False,
"message": "Invalid data"}), 400

    try:
        latest_coordinates['latitude'] =
float(data['latitude'])
        latest_coordinates['longitude'] =
float(data['longitude'])
        logging.info(f"Updated coordinates:
{latest_coordinates}")
        return jsonify({"success": True,
"message": "Coordinates updated"}), 200
    except ValueError as e:
        logging.error(f"Error parsing
coordinates: {e}")
        return jsonify({"success": False,
"message": "Invalid coordinate format"}), 400
```

Funkcija receive_coordinates() služi kao krajnja točka (engl. *endpoint*) za prijem GPS koordinata u JSON formatu koje šalje ESP32 mikroupravljač. Flask aplikacija prvo provjerava valjanost primljenih podataka te zatim ažurira pohranjene koordinate. Za svaki uspješni prijem koordinata, aplikacija šalje potvrdu o uspjehu, dok u slučaju greške šalje odgovarajuću poruku o neispravnosti podataka

6.2. Postavljanje interaktivne mape

Kako bi se korisnicima omogućila jednostavna vizualizacija prikupljenih GPS podataka, razvijeno je interaktivno web sučelje koje koristi „Leaflet" biblioteku za prikaz kartografskih podloga i dinamičkih oznaka za trenutnu lokaciju uređaja. Navedena biblioteka je odabrana zbog svoje jednostavnosti, fleksibilnosti i visokih performansi u prikazu karata temeljenih na podacima „OpenStreetMapa". Vizualno su jasno istaknuti interaktivni elementi poput gumba za automatsko centriranje i lociranje korisnika. Crvena točka jasno označava trenutnu poziciju uređaja, omogućavajući korisniku brz vizualni pregled trenutnog statusa uređaja bez potrebe za dodatnim interakcijama ili kompleksnim radnjama.

HTML struktura web sučelja temelji se na standardnoj web aplikaciji, jedino što je potrebno je dodati „Leaflet" kartu dodavanjem sljedećeg retka unutar <head> bloka:

```
<link rel="stylesheet" href="https://unpkg.
com/leaflet@1.7.1/dist/leaflet.css" />
```

U prvom dijelu implementacije web sučelja, „Leaflet" biblioteka se inicijalizira standardnim JavaScript kodom, s početnim koordinatama (45.1, 15.2) te početnom razinom zumiranja od 8:

```
var map = L.map('map', {
    zoomControl: true,
    dragging: true,
    scrollWheelZoom: true
}).setView([45.1, 15.2], 8);

L.tileLayer('https://{s}.tile.openstreetmap.org/
{z}/{x}/{y}.png', {
    attribution: '&copy; <a href="https://
www.openstreetmap.org/
copyright">OpenStreetMap</a> contributors'
}).addTo(map);
```

Prikazivanje koordinata GPS modula u stvarnom vremenu ostvareno je pomoću JavaScript funkcije koja dohvaća trenutne koordinate s Flask

poslužitelja svakih 500 milisekundi. Lokacija uređaja označena je jasno vidljivom crvenom točkom na karti, a za ažuriranje lokacije koristi se sljedeći dio koda:

```
function updateESPLocation() {
  fetch('/coordinates')
    .then(response => response.json())
    .then(data => {
      if (data.success) {
        const { latitude, longitude } = data.
coordinates;

        if (espLocationMarker) {
          map.
removeLayer(espLocationMarker);
        }

        espLocationMarker =
L.circleMarker([latitude, longitude], {
          radius: 8,
          fillColor: "#ff0000",
          color: "#ff0000",
          weight: 1,
          opacity: 1,
          fillOpacity: 0.8
        }).addTo(map);

        if (autoCenter) {
          map.setView([latitude, longitude],
18, { animate: true });
        }
}

setInterval(updateESPLocation, 500);
```

Koordinate uređaja ESP32 se dohvaćaju putem HTTP GET zahtjeva poslanog Flask aplikaciji. Nakon primitka koordinata, JavaScript provjerava postojeću lokaciju uređaja te, ako je razlika veća od određenog praga, osvježava poziciju crvene točke na karti.

7. Rezultati

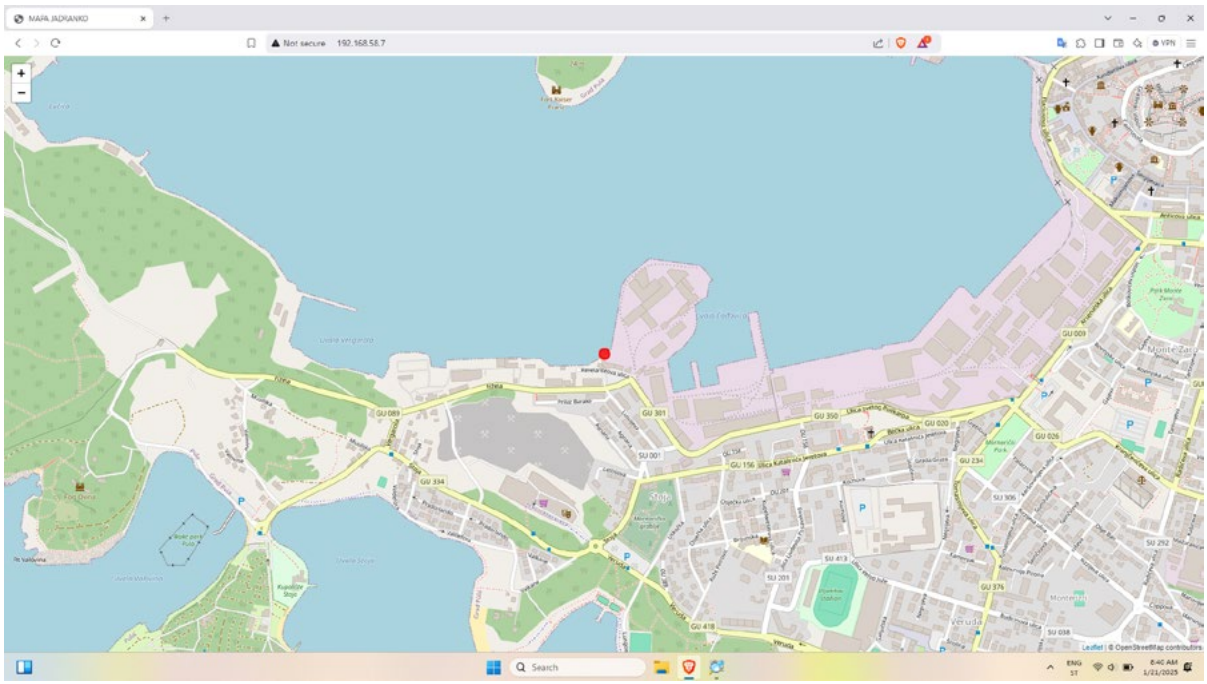
Kao rezultat provedene implementacije izrađen je funkcionalan GPS sustav za praćenje lokacije u stvarnom vremenu, temeljen na povezivanju ESP32 mikroupravljača, NEO-7M GPS modula, mobilne internetske veze i web sučelja s interaktivnom kartom. Sustav omogućuje prikupljanje geografskih koordinata pomoću GPS modula, njihovu obradu na strani ESP32 mikroupravljača te slanje prema Flask poslužitelju putem HTTP komunikacije. Poslužitelj zatim pohranjuje zadnje primljene koordinate i prosljeđuje ih web sučelju, gdje se lokacija uređaja prikazuje na karti.

Na slici 8. prikazan je završni izgled interaktivne karte koja se koristi za vizualizaciju trenutne pozicije GPS uređaja. Karta je izrađena pomoću biblioteke „Leaflet" te koristi „OpenStreetMap" kartografsku podlogu (Leaflet, bez dat.). Trenutna lokacija uređaja označena je crvenom točkom, čime je korisniku omogućen jasan i pregledan prikaz položaja praćenog objekta. Prikaz na karti obuhvaća urbano i obalno područje grada Pule, pri čemu je marker pozicioniran na temelju GPS koordinata koje sustav zaprima od uređaja.

Testiranje sustava provedeno je na otvorenom, u području morskog dobra pod koncesijom tvrtke Calucem d.o.o., gdje je porinut prototip autonomnog plovila Jadranko 1.0. Tijekom cjelokupnog razdoblja ispitivanja sustav je pokazao izrazitu stabilnost u radu. Podaci s GPS uređaja prenošeni su kontinuirano i u realnom vremenu, pri čemu nisu zabilježena nikakva kašnjenja u prijenosu niti problemi u komunikaciji između prototipa i poslužitelja.

Iz prikaza se uočava da sustav uspješno povezuje hardverski dio za prikupljanje podataka s programskim dijelom za njihovu obradu i vizualizaciju. Crvena oznaka na karti omogućuje brzo prepoznavanje trenutne pozicije uređaja, dok mogućnost zumiranja i pomicanja karte doprinosi boljoj preglednosti prostora. Time je ostvarena osnovna funkcionalnost sustava za praćenje lokacije, odnosno prijenos GPS koordinata s mikroupravljača na web aplikaciju i njihov prikaz u obliku razumljivom krajnjem korisniku.

Slika 8. Prikaz lokacije GPS uređaja na interaktivnoj karti



Izvor: autori.

Tijekom testiranja potvrđeno je da se podaci mogu periodično slati s ESP32 mikroupravljača prema poslužitelju te da se na karti prikazuje zadnja dostupna lokacija uređaja. Budući da se koordinate dohvaćaju s poslužitelja u kratkim vremenskim intervalima, web sučelje omogućuje gotovo trenutačan uvid u promjenu položaja. Takav način prikaza posebno je koristan u primjenama u kojima je potrebno pratiti kretanje mobilnih objekata, primjerice vozila, opreme, terenskih uređaja ili drugih pokretnih sustava.

Važno je istaknuti da je prikazani sustav izveden kao funkcionalni prototip. Njegova osnovna namjena nije razvoj komercijalnog rješenja, već demonstracija mogućnosti povezivanja GPS tehnologije, mikroupravljača i web aplikacije u jedinstvenu cjelinu. Unatoč jednostavnoj arhitekturi, sustav pokazuje da se uz relativno dostupne komponente može razviti praktično rješenje za daljinsko praćenje lokacije.

8. Zaključak

U radu je prikazan razvoj jednostavnog GPS sustava za praćenje lokacije s interaktivnom kartom. Sustav se temelji na ESP32 mikroupravljaču, NEO-7M GPS modulu, mobilnoj internetskoj vezi i Flask web aplikaciji. ESP32 prikuplja GPS podatke, obrađuje ih i šalje prema poslužitelju, dok web sučelje omogućuje njihov prikaz na karti. Time je ostvarena cjelovita komunikacija između hardverskog dijela sustava i korisničkog sučelja.

Odabir ESP32 mikroupravljača pokazao se prikladnim zbog ugrađene Wi-Fi komunikacije, malih dimenzija, prihvatljive cijene i dovoljne procesorske snage za obradu GPS podataka i mrežnu komunikaciju. NEO-7M GPS modul odabran je kao uravnoteženo rješenje koje pruža zadovoljavajuću točnost i stabilnost za potrebe prototipnog sustava. Kombinacijom tih komponenti izrađen je sustav koji može prikupljati, slati i prikazivati lokacijske podatke u stvarnom vremenu.

Razvijeno web sučelje s interaktivnom kartom značajno povećava uporabnu vrijednost sustava jer omogućuje jednostavan vizualni prikaz lokacije bez potrebe za ručnim očitavanjem koordinata. Korisnik putem karte može odmah prepoznati položaj uređaja, što sustav čini pogodnim za primjene u edukaciji, prototipiranju IoT rješenja, praćenju mobilnih objekata i demonstraciji rada GPS tehnologije. Iako sustav uspješno ostvaruje osnovnu funkcionalnost, postoje mogućnosti daljnjeg unaprjeđenja. Buduće verzije mogle bi uključivati spremanje povijesti kretanja u bazu podataka, prikaz prijeđene rute, korisničku autentifikaciju,

upozorenja pri izlasku iz definiranog područja, optimizaciju potrošnje energije te izradu kućišta otpornog na vanjske uvjete. Dodatno, sustav bi se mogao proširiti podrškom za više uređaja istovremeno, čime bi se omogućilo praćenje više mobilnih jedinica unutar iste aplikacije.

Zaključno, izrađeni prototip potvrđuje da je pomoću dostupnih hardverskih i softverskih alata moguće razviti funkcionalan i pregledan sustav za GPS praćenje. Sustav uspješno povezuje akviziciju lokacijskih podataka, bežični prijenos i web-vizualizaciju, čime predstavlja dobru osnovu za daljnji razvoj naprednijih IoT rješenja za praćenje i nadzor u stvarnom vremenu.

Zahvala

Autori zahvaljuju tvrtkama Calucem d.o.o. i Infobip d.o.o. na sufinanciranju projekta „Jadranko 1.0: Izrada prototipa robot plovila za sakupljanje plastike i mikroplastike“ u okviru kojeg je nabavljena oprema korištena u ovom radu. Mjerenje prikazano u radu provedeno je neposredno prije završetka navedenog projekta. Također, autori zahvaljuju Eneru Špadi na pomoći pri provedbi mjerenja.

Dodatak

Kod funkcije loop prikazan je u nastavku:

```
void loop() {
  // Čitanje podataka s GPS modula
  while (GPS_Serial.available() > 0) {
    if (gps.encode(GPS_Serial.read())) {
      if (gps.location.isValid()) {
        latitude = gps.location.lat();
        longitude = gps.location.lng();
        gpsDataAvailable = true;
        // Slanje podataka svakih 10 sekundi
        if (millis() - lastSentTime > 10000) {
          lastSentTime = millis();
          // Kreiranje JSON paketa s ključevima latitude i longitude
          String jsonPayload = «{\»latitude\»: « + String(latitude, 6) + «, \»longitude\»: « + String(longitude,
6) + «}»;
          // Kreiranje URL-a za HTTP POST upit
          String url = «http://»;
          url += server_ip;
          url += «:»;
          url += String(server_port);
          url += endpointRoute;
          HTTPClient http;
          http.begin(url);
          http.addHeader(«Content-Type», «application/json»);
          int httpResponseCode = http.POST(jsonPayload);
          if (httpResponseCode > 0) {
            Serial.print(«HTTP Response code: «);
```

```
            Serial.println(httpResponseCode);
            String response = http.getString();
            Serial.println(«Odgovor servera: « + response);
          } else {
            Serial.print(«Greška prilikom slanja POST zahtjeva: «);
            Serial.println(httpResponseCode);
          }
          http.end();
        }
      } else {
        gpsDataAvailable = false;
      }
    }
  }
}
```

Literatura

Arduino. (n. d.-a). Arduino Mega 2560 Rev3. Arduino Store. Preuzeto 30. kolovoza 2024. s https://store.arduino.cc/products/arduino-mega-2560-rev3

Arduino. (n. d.-b). Arduino Nano. Arduino Store. Preuzeto 30. kolovoza 2024. s https://store.arduino.cc/en-hr/products/arduino-nano

Arduino. (n. d.-c). Arduino Uno Rev3. Arduino Store. Preuzeto 30. kolovoza 2024. s https://store.arduino.cc/products/arduino-uno-rev3

Bae, I., & Hong, J. (2023). Survey on the developments of unmanned marine vehicles: Intelligence and cooperation. Sensors, 23(10), 4643. https://doi.org/10.3390/s23104643

El-Rabbany, A. (2002). Introduction to GPS: The Global Positioning System. Artech House

Espressif Systems. (n.d.). ESP32 series datasheet [PDF]. https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

French, G. T. (1996). Understanding the GPS. GeoReasearch

Galadima, A. (2014). Arduino as a learning tool. 2014 11th International Conference on Electronics, Computer and Computation (ICECCO), 1-4. https://doi.org/10.1109/ICECCO.2014.6997577

Hart, M. (n. d.). TinyGPSPlus: A new, customizable Arduino NMEA parsing library [Computer software]. GitHub. Preuzeto 22. svibnja 2026. s https://github.com/mikalhart/TinyGPSPlus

Hodgson, D. (2021). Mechatronics and Physical Computing., 453-477. https://doi.org/10.1016/b978-0-12-815073-3.00020-x

Leaflet. (n. d.). Leaflet: An open-source JavaScript library for mobile-friendly interactive maps. Retrieved January 21, 2025, from https://leafletjs.com/

Pallets. (n. d.). Flask documentation. Preuzeto 21. siječnja 2025. s https://flask.palletsprojects.com/

McNamara, J. (2004). GPS for Dummies. Wiley

Pan, T., & Zhu, Y. (2018). Getting Started with Arduino., 3-16. https://doi.org/10.1007/978-981-10-4418-2_1

Qualcomm. (2009, November 11). Qualcomm now sampling industry's first dual-carrier HSPA+ and multi-mode 3G/LTE chipsets [Press release]. Preuzeto 22. svibnja 2026. s https://www.qualcomm.com/news/releases/2009/11/qualcomm-now-sampling-industrys-first-dual-carrier-hspa-and-multi-mode

Radolović, P. (2023). Arduino C programski jezik u mehatronici. Završni rad, Istarsko veleučilište – Università Istriana di scienze applicate. Dostupno na nacionalnom repozitoriju završnih i diplomskih radova ZIR: https://urn.nsk.hr/urn:nbn:hr:212:271432

Raspberry Pi (n. d.). Raspberry Pi Zero 2 W product brief [PDF]. https://pip.raspberrypi.com/categories/584-raspberry-pi-zero-2-w

u-blox. (2014). NEO-7: u-blox 7 GNSS modules data sheet (UBX-13003830, R07). Preuzeto 22. svibnja 2026. s https://content.u-blox.com/sites/default/files/products/documents/NEO-7_DataSheet_%28UBX-13003830%29.pdf

Xu, G. (2007). GPS: Theory, Algorithms, and Applications. Springer.

GPS tracking system with interactive map

Summary

The paper presents the development of a functional prototype of a GPS location tracking system with data display on an interactive map. The theoretical part of the paper analyzes Arduino boards, the ESP32, and the Raspberry Pi Zero 2 W as potential platforms for system implementation, while the ESP32 microcontroller was selected for the practical realization of the prototype. The system is based on connecting a NEO-7M GPS module to an ESP32 microcontroller, which collects geographic coordinates, processes them and sends them to a remote server via an internet connection. HTTP communication is used for data transfer, with coordinates sent in JSON format. The server part of the system was created using the Flask web framework, while the Leaflet library with OpenStreetMap cartographic support was used for location visualization. On the web interface, the current position of the device is displayed using a clearly visible marker on the map, which provides the user with a simple and clear insight into the location of the monitored object. The prototype was tested in an open area, with GPS data sent to the server every 10 seconds. During testing, the system worked stably and no transmission delays or communication difficulties were recorded. The presented system confirms that using available hardware and software tools it is possible to create a simple, adaptable and functional IoT solution for remote location monitoring in real time. The system is suitable for educational purposes, prototype development and as a basis for further upgrades, such as saving movement history, displaying routes or monitoring multiple devices simultaneously.

Keywords: *NEO-7M GPS, ESP32, location tracking, interactive map, IoT*