Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет __Комп'ютерної інженерії та управління_____
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки_____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____другий (магістерський)_____
(рівень вищої освіти)

_____Методи модельно-орієнтованого проектування для системи
_____інформування пасажирів міського транспорту_____
(тема)

Виконав: студент 2 курсу, групи СКСм-20-1
_____Криклій Є.В.___
(прізвище, ініціали)
Спеціальність 123 Комп'ютерна інженерія
Тип програми ____освітньо-професійна_____
(освітньо-професійна або освітньо-наукова)
Освітня програма_____
___Спеціалізовані комп'ютерні системи____
(повна назва освітньої програми)
Керівник роботи _____доц. Рахліс Д.Ю.___
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Чумаченко С.В.
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Автоматизації проектування обчислювальної техніки
Рівень вищої освіти другий (магістерський)
Спеціальність 123 – Комп'ютерна інженерія
Тип програми Освітньо-професійна
Освітня програма Спеціалізовані комп'ютерні системи
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри
_____
(підпис)
«_4_» _листопада_ 2021 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Криклію Єгору Вікторовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Методи модельно-орієнтованого проектування для системи інформування пасажирів міського транспорту
затверджена наказом по університету від 04 11 2021 р. № 1635 Ст.
2. Термін подання студентом роботи до екзаменаційної комісії 27 12 2021 р.
3. Вихідні дані до роботи
Arduino Uno, GPS модуль Ublox NEO-6M, Matlab, Simulink
4. Перелік питань, що потрібно опрацювати в роботі
Вступ
1 Огляд систем інформування пасажирів міського транспорту
2 Дослідження методів модельно-орієнтованого проектування систем керування за допомогою Matlab/Simulink
4 Створення моделі системи інформування пасажирів міського транспорту
3 Вибір компонентів інформування пасажирів міського транспорту
4 Реалізація запропонованої моделі та перевірка працездатності шляхом моделювання
Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів)

20 слайдів

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Видача теми роботи, її узгодження та затвердження.	01.09.2021 – 15.09.2021	
2.	Аналіз проблемної області, постановка задачі, вибір засобів реалізації	16.09.2021 – 25.09.2021	
3.	Аналіз методів модельно-орієнтованого	26.09.2021 – 15.10.2021	
4.	Розробка моделі проектування	16.10.2021 – 26.10.2021	
5.	Моделювання системи у середовищі Matlab	27.10.2021 – 15.11.2021	
6.	Реалізація моделі	16.11.2021 – 20.11.2021	
7.	Верифікація моделі	21.11.2021 – 10.12.2021	
8.	Оформлення пояснювальної записки	11.12.2021 – 22.12.2021	
9.	Захист проекту	27.12.2021	

Дата видачі завдання 01.09.2021

Студент _____
(підпис)

Керівник роботи _____ доц. каф. АПОТ Рахліс Д.Ю.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка містить: 64 сторінки, 65 рисунків, 5 таблиць, 3 лістинга, 15 посилань та 3 додатки.

МОДЕЛЬНО-ОРІЄНТОВАНЕ ПРОЕКТУВАННЯ, MATLAB, SIMULINK-МОДЕЛЬ, ARDUINO, GPS-МОДУЛЬ, IOT SERVICE.

Метою даної атестаційної роботи є розробка моделі системи інформування пасажирів громадського транспорту згідно з парадигмами модельно-орієнтованого проектування. В роботі було досліджено існуючі методи розв'язання поставленої задачі та проаналізовано основні положення модельно-орієнтованого проектування. В системі Matlab/Simulink було створено модель системи інформування пасажирів громадського транспорту. Для апаратної реалізації було обрано апаратну платформу, а семе мікроконтролер Arduino, GPS модуль Ublox NEO-6M, Ethernet-модуль W5100, Wi-Fi модуль ESP8266 та рідко-кристалічний екран LCD 16*2.

Проведені експериментальні дослідження розробленного алгоритму шляхом створення Simulink-моделі системи та подальшого її моделювання.

ABSTRACT

The explanatory note contains: 64 pages, 65 figures, 5 tables, 3 listings, 15 references and 3 applications.

MODEL-BASED DESIGN, MATLAB, SIMULINK-MODEL, ARDUINO, GPS- MODULE, IOT SERVICE.

The purpose of this master`s work is to develop a model of a public transport passenger information system in accordance with the paradigms of model-based design. The work investigated the existing methods for solving the problem and analyzed the main methods of model-based design. In the Matlab/Simulink, a model of a public transport passenger information system was created. For the hardware implementation, a hardware platform was chosen, namely an Arduino microcontroller, a GPS-module Ublox NEO-6M, a Ethernet module W5100, an Wi-Fi module ESP8266 and a LCD screen 16*2.

Experimental studies of the developed algorithm were carried out by creating a Simulink-model of the system and its further modeling.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

LCD – рідкокристалічний дисплей;
IoT – інтернет речей;
GPRS – загальний сервіс пакетної радіопередачі;
GPS – система глобального позиціювання;
API – прикладний програмний інтерфейс;
MAC – управління доступом до посередників;
UART – універсальний асинхронний приймач/передавач;
МОП – Модельно-орієнтоване проектування;
СІП – Система інформування пасажирів;
NMEA – National Marine Electronics Association (національна асоціація морської електроніки);
GPS – Global Positioning System (система глобального позиціювання).

ВСТУП

На сьогоднішній день цифрові системи настільки складні, що традиційний процес їх перевірки вже мало придатний. Коли на пізніх стадіях проекту виявляються проблеми, то зазвичай для їх вирішення потрібні складні та витратні за часом зміни проекту, які призводять до дорогої модифікації апаратних засобів. Крім того складність алгоритмів не дає змогу перевірити його працездатність на всіх робочих режимах. Тому і необхідні нові методи, до яких і відноситься модельно-орієнтоване проектування (МОП) [1]. Цей підхід дає змогу перевірити коректність роботи системи на ранніх стадіях, коли ще можливо виправити помилки швидко та з мінімальними затратами.

Об’єктом дослідження є системи інформування пасажирів (СІП) міського транспорту. Предметом дослідження є методи модельно-орієнтованого проектування складних систем. Такі система працюють в реальному режимі часу та можуть мати різноманітні функції [2]. На сьогодні у світі існує багато прикладів реалізації систем інформування пасажирів. Але головним недоліком таких систем є той факт, що вони запрограмовані на визначений набір функцій, який, найчастіше, не має цінності для пасажирів. Надлишкова інформація навпаки відволікає людину від головного, особливо, коли мова йде про людей похилого віку чи гостей міста. Також одним з недоліків готових рішень є те, що виробники даних пристрої зберігають в таємниці алгоритми, за допомогою яких працює їх пристрої, що ускладнює їх модернізацію та модифікацію. Для того, щоб позбутися цих недоліків було вирішено запропонувати пристрій, що може бути запрограмований за допомогою відкритого середовища розробки.

Таким чином, науково-практична задача кваліфікаційної роботи полягає у розробці та верифікації моделі системи інформування пасажирів міського транспорту з використанням середовища Matlab.

1 МОДЕЛЬНО-ОРІЄНТОВАНЕ ПРОЕКТУВАННЯ СИСТЕМИ ІНФОРМУВАННЯ ПАСАЖИРІВ

1.1 Сучасний стан та існуючі рішення систем інформування пасажирів міського транспорту

Система інформації про пасажирів або система відображення інформації про пасажирів − це автоматизована система для надання користувачам громадського транспорту інформації про характер і стан послуг міського транспорту за допомогою візуальних, голосових або інших засобів масової інформації [3]. Вони також відомі як інформаційні системи для клієнтів і операційні інформаційні системи. Відмінність між системами полягає у різних типах інформації, що надається, а саме:

– статична інформація або інформація про розклад, яка змінюється тільки час від часу і зазвичай використовується для планування поїздки до відправлення;

– інформація в реальному часі, отримана з систем автоматичного визначення місцезнаходження транспортного засобу, яка постійно змінюється в результаті реальних подій і зазвичай використовується в ході поїздки.

Статична інформація традиційно надавалася в друкованій формі через карти маршрутної мережі та буклети з розкладом на станціях громадського транспорту. Однак в сучасному світі статична інформація поступається з інформації в реальному часі. Надання пасажирам інформації в режимі реального часу дає їм змогу міняти маршрути в залежності від затримок на ті

чи іншій лінії. Це допомагає стимулювати більш широке використання громадського транспорту, що для багатьох країн є політичною мірою.

Інформація в реальному часі надається пасажирам різними способами, включаючи додатки для мобільних телефонів, покажчики на рівні платформи та автоматизовані системи оповіщення. Вона може включати як прогнози часу прибуття і відправлення, так і інформацію про характер і причини збоїв.

Інформація може бути доставлена через будь-які електронні носії, в тому числі: додаток для мобільного телефону; світлодіодні дисплеї і екрани всередині станцій; електронні паперові дисплеї і екрани на автобусних зупинках; через сайт; телефон.

У кожного підходу є свої сильні і слабкі сторони. Наприклад, інформація може швидко оновлюватися на веб-сайтах і в соціальних мережах, але деякі пасажири не можуть отримати до них доступ або вважають за краще поговорити з кимось, хто може дати додаткові запевнення і відповісти на питання. Перші такі «розумні зупинки» були введені в Чехії, а точніше в Празі. Зупинки в Сінгапурі мають сотні додаткових функцій. За допомогою великих цифрових екранів кожен може отримати корисну інформацію про час прибуття автобуса, переміщатися по карті міста, дізнаватися погоду або навіть місцеві новини. Розумні зупинки також дозволяють підключатися до Інтернету через Wi-Fi, використовувати зарядний пристрій для телефону і навіть завантажувати електронні книги на свій смартфон, відсканувавши QR-код. Крім іншого, для зручності і натхнення жителів зупинки обладнані садами на даху та парковкою для велосипедів (рис. 1.1).

Рисунок 1.1 – «Розумна» зупинка у Сінгапурі

В Україні у м. Київ перва «розумна» зупинка зявилась у 2016 році. Вона є автономною, бо отримує енергопостачання від сонячної батареї, та обладнана Wi-Fi і в ній можна підзарядити телефон.

Рисунок 1.2 – «Розумна» зупинка у Києві

Багато таких зупинок було обладнано у інших крупних містах України (Харків, Полтава, Маріуполь, Рівне та ін.). У м. Харків ще у 2018 році було підписно меморандум про співпрацю, який дозволить місту взяти участь у проекті Smart City UA. Зараз у місті працюють над проектом «Зупинка» для

онлайн-відстеження руху громадського транспорту. За допомогою GPS-трекерів, які встановлені на всьому громадському транспорті, харків'яни зможуть побачити, де зараз знаходиться громадський транспорт і через скільки прибуде на зупинку. Але ця концепція є зручною тільки для певної категорії громадян. Ті категорії пасажирів, що не користуються мобільними додатками в силу різних причин, не зможуть мати доступ до «високих технологій». Інформація про час прибуття того чи іншого транспорту на зупинках має бути зрозумілою всім.

Не дивлячись на те, що інтелектуальні інформаційні системи для зупинок міського транспорту з'явилися в усьому світі, вони до сих пір залишаються дуже складними і дорогими.

Найбільш популярний напрям сучасних досліджень в цій області пов'язано з мікроконтролерами. Наприклад, в статті [4] автори запропонували систему, яка відстежує поточне місце розташування всіх автобусів і оцінює час їх прибуття на різні зупинки на відповідних маршрутах. Оцінки оновлюються кожного разу, коли автобус відправляє оновлення. Він передає цю інформацію пасажирам за допомогою терміналів з дисплеями на автобусних зупинках, веб-інтерфейсу та SMS. Широта, довгота і швидкість автобуса періодично передаються на центральний сервер через GPRS. Функції, пов'язані з позиціонуванням і зв'язком з сервером, обробляються сценаріями Python, запущеними в модулі GPS / GPRS.

У статті [5] описується система, яка використовує ARDUINO UNO, чотири ІК-датчика наближення і модуль GPS. Два ІК-датчика знаходяться на шляху входу, а ще два − на шляху виходу. Інфрачервоний датчик наближення використовується для підрахунку кількості людей в автобусі. Модуль GPS використовується для визначення широти і довготи поточного місцезнаходження автобуса. Інформація від датчиків і модуля GPS відправляється в контролер, і через модуль IoT інформація відображається на веб-сторінці. Для пасажирів всередині автобуса на РК-дисплеї відображається інформація про поточну зупинку, наступну зупинку і

приблизний час до наступної зупинки. А для людей, які перебувають поза автобуса, інформація про поточне місцезнаходження автобуса і кількості пасажирів всередині автобуса буде надана через веб-сторінку. Головний недолік цієї системи в тому, що інформація буде надаватися стороннім пасажирам тільки в тому випадку, якщо вони мають доступ до веб-сторінці.

У цій статті [6] пропонується система стеження за транспортними засобами з використанням технологій GPS і GSM. Приймач GPS і модем GSM використовують Arduino MEGA2560. В транспорті може бути встановлено GSM-пристрій з SIM-картою. Через модем GMS SMS буде отримано і відправлено на Arduino MEGA2560 в автомобілі. При отриманні повідомлення Arduino MEGA2560 перевіряє пароль і запит. Якщо все збігається, він виконає запит, необхідний власником, відправивши посилання з довготою і широтою через Google Maps, яка показує розташування транспортного засобу. У статті [7] пропонується механізм безпеки автобусів на основі IoT, який призначений для підрахунку входу / виходу студентів з автобуса. Технологія відстеження шкільного автобуса на базі Android можлива, і це альтернативний спосіб спостереження та відстеження дітей, коли вони їдуть в школу і додому. Система, запропонована в статті [8], визначала довготу і широту, координати розташування рухомого автобуса і оцінювала швидкість його руху. Інформація про час прибуття автобусу передавалась на ПК в командному центрі автобусного оператора за допомогою бездротового датчика і для користувачів через мобільний додаток.

Огляд літератури показав актуальність обраного об’єкту дослідження та необхідність створення такої системи інформування пасажирів міського транспорту, яка б використовували сучасні технології, але б не була занадто дорогою.

1.2 Модельно-орієнтоване проектування

Для тестування прикладного програмного забезпечення на всіх етапах створення систем застосовують підхід модельно-орієнтоване проектування (МОП), який заснований на використанні моделей проектованої системи і моделі контрольованого процесу. Ці моделі є специфікацією проектованої системи, яка актуалізується під час виконання розробки. Дана специфікація дозволяє виконати аналіз коректності і можливості виконання вимог за допомогою моделювання.

Застосування підходу МОП забезпечує підвищення надійності створюваних програмних, апаратних і програмно-апаратних систем [9]. Появі цього підходу передували етапи становлення загальної теорії систем, створення CASE (Computer-Aided Software Engineering) − сукупність засобів автоматизації розробки програмного забезпечення, розробка мови UML (Unified Modeling Language) в 1990-х рр., що забезпечує проектування систем на різних рівнях абстракцій і поява в 2001 р. MDA (Model Driven Architecture).

Використання ЕОМ практично витіснило аналітичні методи дослідження за рахунок широкого впровадження в практику обчислювальних методів, а потім і зовсім призвело до зміни базового дослідницького підходу на модельно-орієнтоване проектування. Замість фізичних прототипів і текстових специфікацій в модельно-орієнтованому проектуванні застосовується модель (рис. 1.3).

Рисунок 1.3 – Концепція модельно-орієнтованого проектування

МОП, заснований на застосуванні концепції «in-the-loop testing», має декілька варіантів в залежності від того, що виступає в якості об'єкта тестування. Якщо в якості об'єкта тестування виступає модель, то такий підхід носить назву «model-in-the-loop» (MiL), якщо програмне забезпечення – «software-in-the-loop» (SiL), прототип пристрою – «processor-in-the-loop» (PiL), а якщо готовий пристрій – «hardware-in-the-loop» (HiL).

Кожен з рівнів конфігурації скорочує час розробки, яке починається з побудови математичної моделі і закінчується запуском програмного забезпечення на автономній вбудованій платформі.

МОП починається з етапу MiL, при цьому реалізується імітаційна модель системи управління і керованої моделі об'єкта (процесу) управління. Основною метою MiL є тестування і перевірка математичної моделі. На етапі SiL імітаційна модель, що синтезована на попередньому етапі, замінюється виконуваним кодом, запущеним на тій же апаратній платформі, де використовуються обчислення з фіксованою комою. У середовищі Matlab для програмного рівня тестування (SiL) створюється блок S-функції, що дозволяє перейти до використання в Simulink моделі, виробляти підбір і налаштування

необхідних параметрів моделі. При цьому відбувається спадкування: виконання згенерованого і відкомпільованого С-коду на даному етапі виконується в імітаційній Simulink моделі.

Наступний рівень (PiL) визначає найбільш відповідальний в практичному застосуванні рівень тестування. Цей етап призначений для оцінки параметрів ефективності розроблюваної системи управління на вже конкретному, обраному для проекту цільовому процесорі. Проводиться завантаження згенерованого програмного коду на цільову відлагоджувальну плату. Імітаційна модель, запущена на комп’ютері, обмінюється даними з завантаженим на апаратну платформу програмним забезпеченням, як правило, за допомогою послідовних ліній зв'язку. Таким чином, забезпечується обмін тестовими сигналами, що генеруються в імітаційної моделі, і програмним забезпеченням, встановленим в процесорі тестової плати. Тестування програмного забезпечення в реальному часі не може бути виконано на етапі PiL, цей крок виконується на етапі HiL. Переваги PiL полягають в низькій вартості в порівнянні з HIL; можливістю проводити довільні випробування; у високій точності при тестуванні вбудованих алгоритмів; у відсутності обмежень за складністю моделі обладнання (в порівнянні з HiL).

Останнім етапом розробки вбудованої системи є HiL. На даному етапі виконується емуляція датчиків і виконавчих механізмів в режимі реального часу для цільової платформи до того, як до контролера будуть підключені реальні датчики і виконавчі механізми. Процес емуляції використовується для взаємодії моделі обладнання та інтегрованої системи при випробуванні всіх їх на одній платформі.

1.3 Matlab як середовище проектування та верифікації

Труднощі традиційного підходу до проектування цифрових систем полягають в тому, що на етапі складання вимог і специфікацій

використовуються текстові документи недосконалі при ітеративному підході, фізичні прототипи на етапі проектування також недосконалі, складні і дорогі; на етапі реалізації при ручній розробці застосування інших програмних засобів і людські помилки роблять процес ненадійним, а на етапі перевірки традиційне тестування призводить до виявлення помилок лише в кінцевій стадії розробки. Компанія MathWorks пропонує новий підхід для проектування − модельно-орієнтоване проектування, що реалізовується в середовищі MATLAB/Simulink спільно з пакетом Simulink Coder [10]. Цей метод об'єднує в безперервний робочий процес різні етапи розробки системи, такі як формування специфікацій і системних вимог, імітаційне моделювання, розробка системи, налагодження та тестування помилок.

Необхідно відзначити, що отриманий за допомогою МОП програмний код може бути використаний для вирішення завдань в додатках реального часу, робить більш ефективним сам процес моделювання, дозволяє виробляти швидке прототипування, оптимізувати апаратно-програмне тестування розроблюваних систем.

Крім пакету Simulink Coder є також блоки Embedded Coder в бібліотеці Simulink, що дозволяють працювати з периферією конкретного контролера. Для багатьох контролерів в складі Embedded Coder є структури, які генерують специфічний для даного контролера код управління його периферійними пристроями: аналого-цифровими і цифро-аналоговими перетворювачами, широтно-імпульсними модуляторами, таймерами, контролерами переривань, портами введення-виведення і т.п. До цих блоках, також відносяться низькорівневі операції з пам'яттю, які можуть бути реалізовані у вигляді блоків Simulink. За допомогою спеціальних блоків Simulink і налаштувань моделі можлива також інтеграція з операційною системою реального часу. Embedded Coder надає також можливість проводити перевірку відкомпільованого коду в режимі Processor-in-the-loop (PiL). Для здійснення PiL режиму до складу мікроконтролера, до якого завантажується об'єктний код, додаються програми, які дозволяють вести

обмін даними між моделлю Simulink на головному комп'ютері і мікроконтролером в при виконанні ним керуючої програми в реальному часі. Для цього можуть бути задіяні різні інтерфейси взаємодії цих пристроїв, такі як: Serial, Ethernet, SHM, JTAG. Структурна схема реалізації режиму Processor-in-the-loop представлена на рис. 1.4.

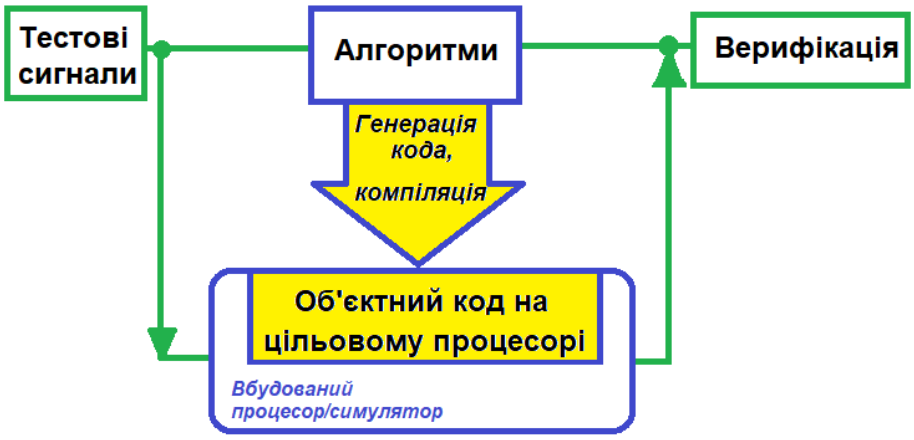


Рисунок 1.4 – Структурна схема реалізації PiL режиму

Підтримка Arduino в Embedded Coder реалізована за допомогою Target Support Package, який можна завантажити з File Exchange: https://www.mathworks.com/matlabcentral/fileexchange/30277-embedded-coder-target-for-arduino.

В командній строчці Matlab треба виконати команду: *supportPackageInstaller*. Запуститься інсталятор (рис. 1.5), який дозволить вам вибрати цільову платформу і автоматично завантажить і встановить всі необхідні файли.

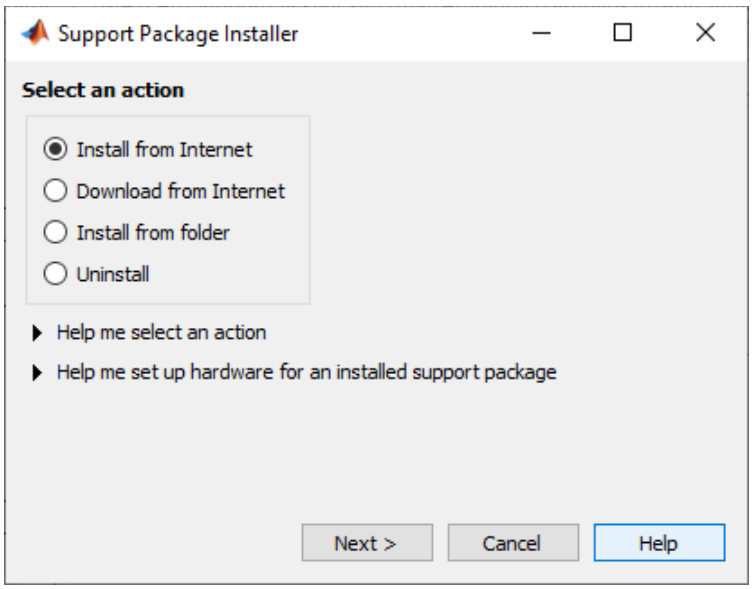


Рисунок 1.5 – Інсталятор

Обравши Arduino у правому вікні відобразяться існуючі пакети для роботи з цим мікроконтролером: MATLAB Support Package for Arduino Hardware https://www.mathworks.com/hardware-support/arduino-matlab.html та Simulink Support Package for Arduino Hardware https://www.mathworks.com/hardware-support/arduino-simulink.html (рис. 1.6).

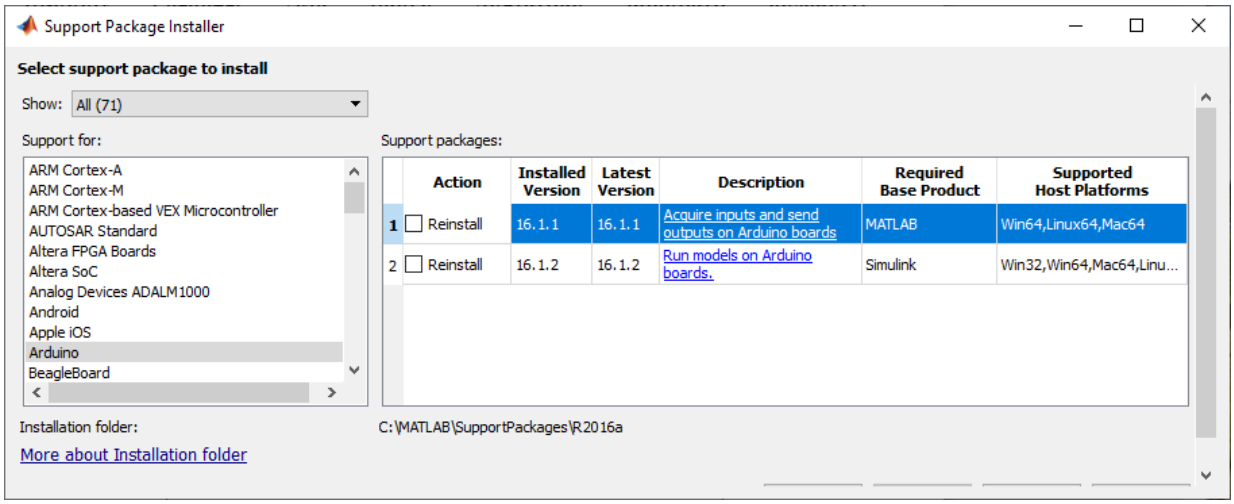


Рисунок 1.6 – Пакети для Arduino

Розглянемо простий приклад створення моделі Simulink (рис. 1.7).

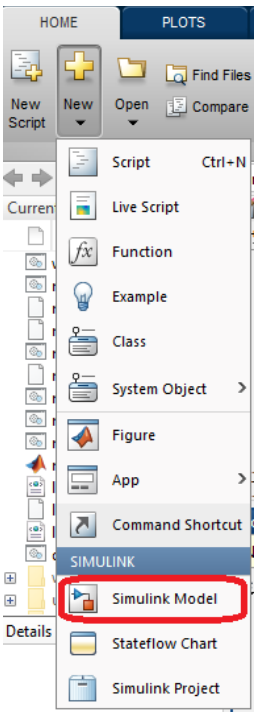


Рисунок 1.7 – Створення Simulink моделі

Натиснувши на кнопку  відкриється список бібліотек Simulink, де потрібно вибрати Simulink Support Package for Arduino hardware (рис. 1.8).

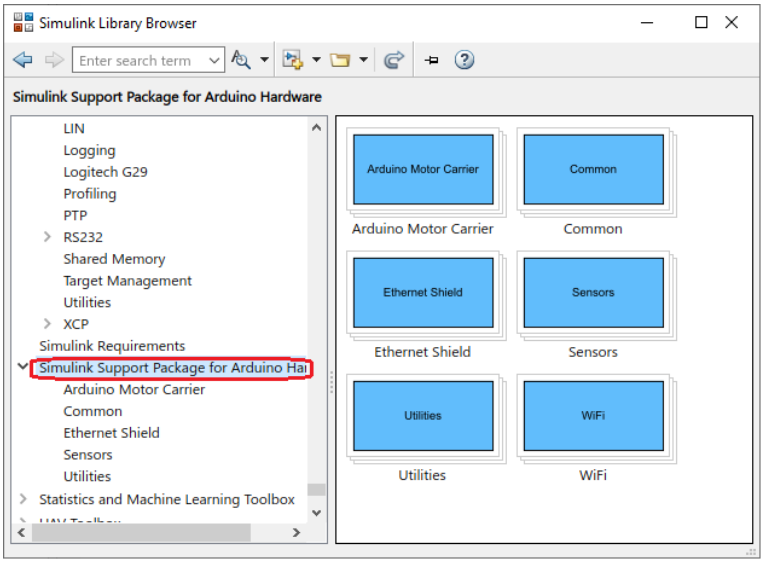


Рисунок 1.8 – Бібліотека Arduino для Matlab

Найпростіша модель для перевірки працездатності мікроконтролера Arduino представлена на рисунку 1.9.

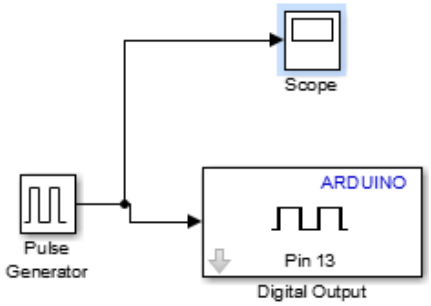


Рисунок 1.9 – Simulink модель

Для коректної роботи моделі треба зробити налаштування усіх компонентів, а саме:

- для генератора сигналів обрано період 2 с. та ширину імпульсу 1 с. (рис. 1.10);

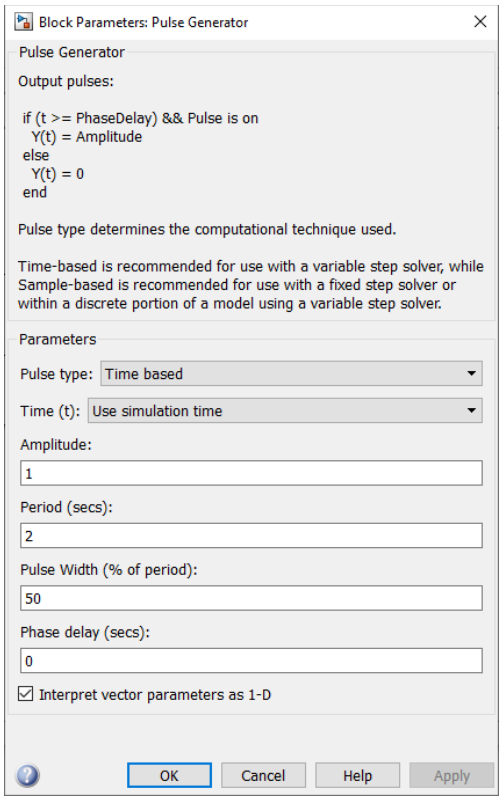


Рисунок 1.10 – Параметри генератора сигналу

- цифровий вихід № 13 відповідаю світлодіоду на платі Arduino Uno (рис. 1.11).

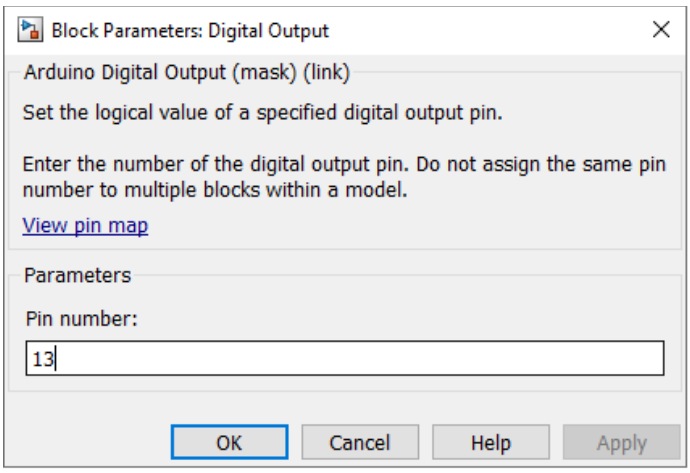


Рисунок 1.11 – Налаштування піна для виходу

Для того, щоб побачити результати треба відкрити блок Scope та запустити моделювання (рис. 1.12). На рисунку видно, що на виході сигнал постійно змінює своє значення.

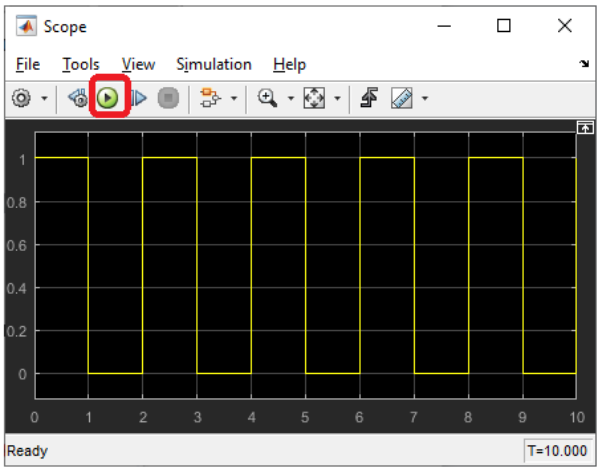


Рисунок 1.12 – Результати моделювання

Matlab дозволяє запрограмувати Arduino. За допомогою USB кабелю (type A − type B) з'єднаємо плату Arduino Uno та персональний комп'ютер, на якому виконується проектування (рис. 1.13).

Рисунок 1.13 – Підключення Arduino до комп'ютера

Після цього переходимо до меню Tools –> Options та обираємо апаратну платформу Arduino Uno (рис. 1.14).

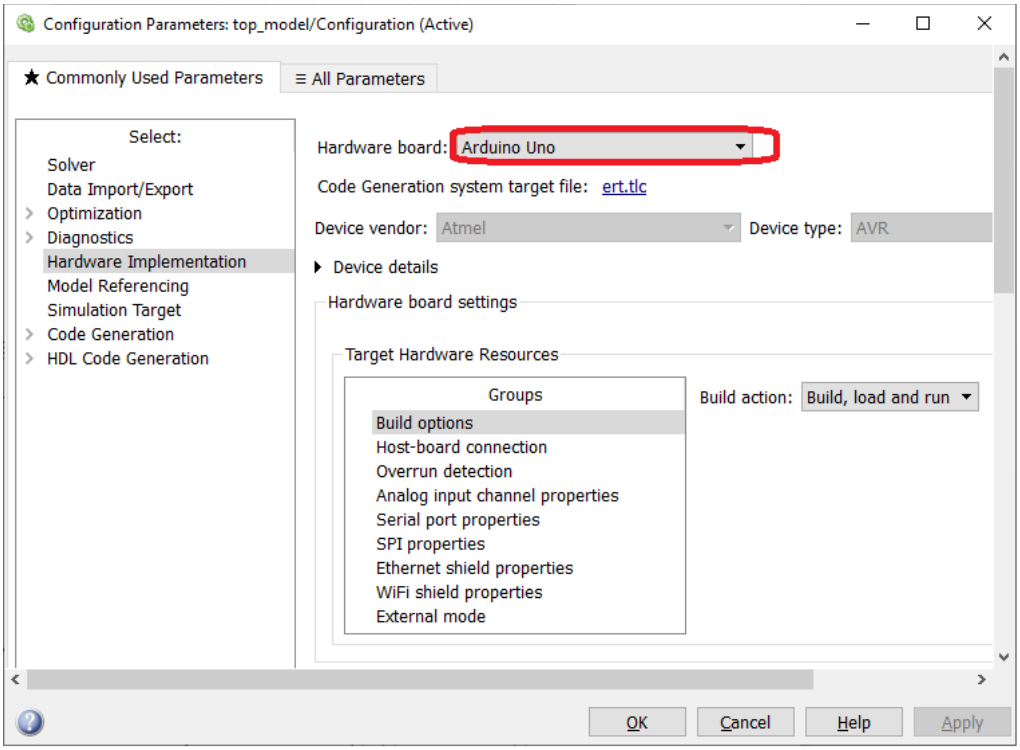


Рисунок 1.14 – Апаратна платформа

Для програмування плати на панелі обираємо кнопку  (рис. 1.15).

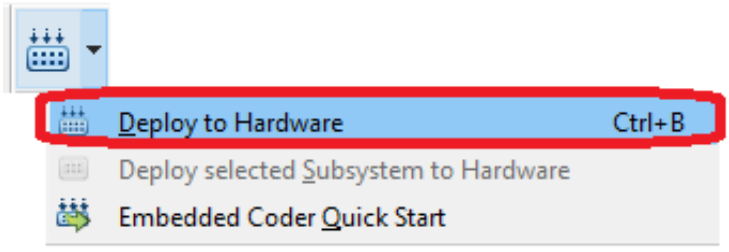


Рисунок 1.15 – Програмування Arduino через Simulink

Тепер ми можемо спостерігати за роботою Arduino, а саме за мерехтінням світлодіоду з періодом 2 секунди (рис. 1.16).

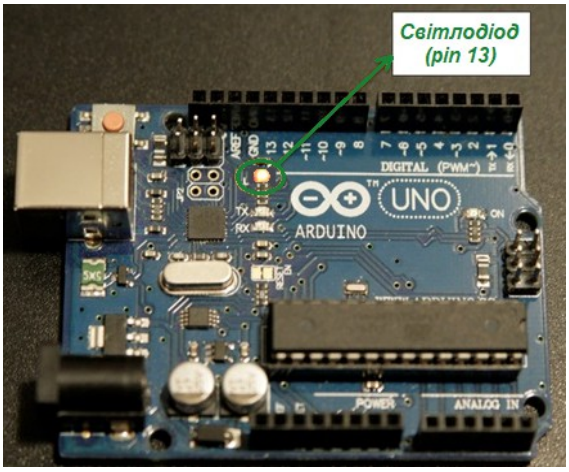


Рисунок 1.16 – Результат програмування плати

1.4 Мета та задачі проекту

При МОП систем реального часу, до яких відноситься система інформування пасажирів міського транспорту, ядром розробки є програмна модель. Така модель дозволить тестувати і досліджувати алгоритм функціонування, реалізований в пристрої, в реальному часі.

Тому, *мета проекту* – розробка моделі системи інформування пасажирів громадського транспорту згідно з парадигмами модельно-орієнтованого проектування.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

- проаналізувати існуючі моделі СІП міського транспорту;
- проаналізувати основні положення модельно-орієнтованого проектування;
- запропонувати модель СІП з урахуванням методів модельно-орієнтованого проектування;

– побудувати модель СІП в середовище Simulink для верифікації проекту;

– обґрунтувати вибір апаратної платформи реалізації запропонованої системи.

2 МОДЕЛЬ СИСТЕМИ ІНФОРМУВАННЯ ПАСАЖИРІВ

2.1 Загальна модель системи інформування пасажирів міського транспорту

Запропонована система інформування пасажирів може бути встановлена на зупинці будь якого міського транспорту. Наприклад, як це показано на рисунку 2.1.

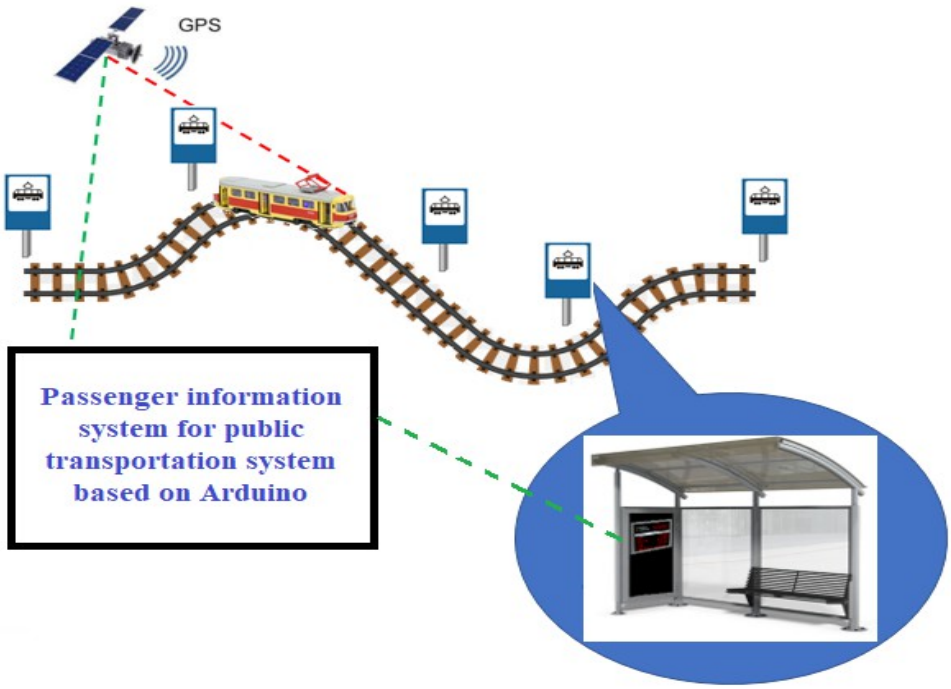


Рисунок 2.1 – Місце використання СІП

Щоб відстежувати точне місцезнаходження транспорту, ми використовуємо модуль GPS і модуль ESP, який вбудований і розміщений в трамваї, тролейбусі, автобусі, тощо. Дані (місце розташування) будуть відправлені на хмарний сервер IoT через Wi-Fi. Додатковий модуль Ethernet, підключений до Arduino Uno, буде обробляти інформацію для розрахунку часу прибуття. Дисплей на зупинці транспорту покаже очікуваний час прибуття транспорту.

Розглянемо модель системи інформування пасажирів (рис. 2.2).

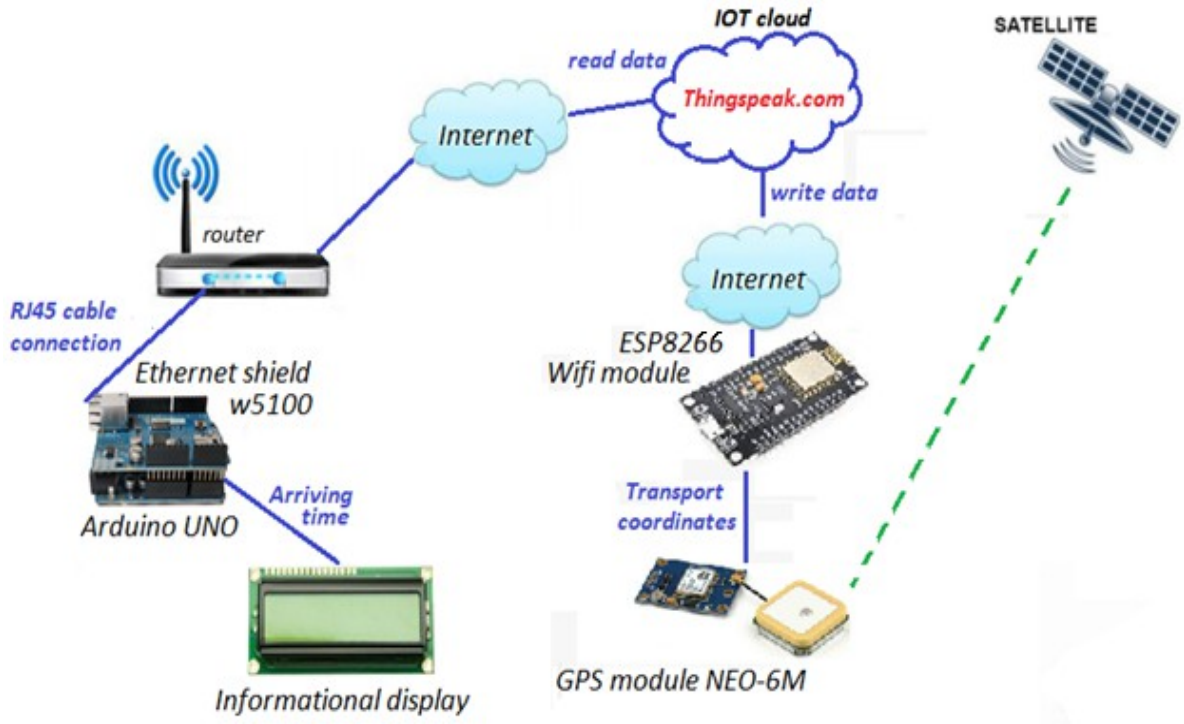


Рисунок 2.2 – Модель системи інформування пасажирів міського транспорту

На рисунку 2.2 присутні наступні апаратні компоненти:

- Arduino Uno Rev 3;
- Ethernet shield W5100;
- Liquid crystal display LCD 16*2;
- Wi-fi module ESP8266;
- GPS-module NEO-6M.

Щоб знайти час прибуття транспорту, ми використовуємо формулу відстані Хаверсіна (1). Формула Хаверсіна використовується для розрахунку географічного відстані на Землі [11]. Якщо у вас є два різних значення широти і довготи для двох різних точок на Землі, то за допомогою формули Хаверсіна ви можете легко обчислити відстань по дузі великого кола (найкоротша відстань між двома точками на поверхні сфери). Термін Хаверсін був введений професором Джеймсом Інманом у 1835 році. Хаверсін – дуже популярна і часто використовувана формула при розробці

геоінформаційних систем:

$$
\begin{aligned}
dlon &= lon2 - lon1 \\
dlat &= lat2 - lat1 \\
a &= (sin(dlat/2))^2 + cos(lat1) * cos(lat2) * (sin(dlon/2))^2, \\
c &= 2 * atan2(sqrt(a), sqrt(1-a)), \\
d &= R * c, \text{ where } R=6,371 \text{ km} - \text{is the radius of the Earth}
\end{aligned}
\tag{1}
$$

Поточні координати транспорту позначені як *lat2* − широта та *lon2* − довгота. Координати зупинки позначені як *lat1* − широта, *lon1* − широта. Отже, застосувавши цю формулу, ми можемо отримати приблизний час прибуття транспорту.

2.2 Хмарний сервіс інтернет речей ThingSpeak.com

Для зберігання даних запропоновано використовувати хмарний IoT сервіс. Хмара IoT − це величезна мережа, яка підтримує пристрої та додатки IoT [12]. Вона включає в себе базову інфраструктуру, сервери і сховище, необхідні для операцій і обробки в реальному часі. В наші дні таких сервісів дуже багато. Наприклад, Thingworx, Microsoft Azure, платформа Google, IBM Watson, AWS, Cisco IoT Cloud Connect, Salesforce, Kaa, Oracle, ThingSpeak. У всіх є свої переваги і недоліки. Для нашої системи був обраний IoT ThingSpeak.com.

Особливості Thingspeak: збір даних в приватних каналах; інтеграція додатків; планування заходів; Matlab аналітика і візуалізація.

До переваг цього сервісу можна віднести:

- безкоштовний хостинг для каналів;
- легка візуалізація;
- додаткові функції для Ruby, Node.js і Python.

До недоліків цього сервісу можна віднести:

– обмежене завантаження даних для API;

– ThingSpeak API може стати перешкодою для новачків.

ThingSpeak − це місце, де ми збираємося зберігати дані, зібрані нашої будь яким чином, і де ми можемо бачити дані, які ми зібрали. Ядро платформи ThingSpeak − це канали, по яких користувачі відправляють інформацію для зберігання і візуалізації. Кожен канал містить, крім назви і опису, вісім полів для інформації будь-якого типу, три поля для географічних координат (широта, довгота, висота), поле для посилання на сайт, де використовується збережена інформація, посилання на канал YouTube (рис. 2.3).

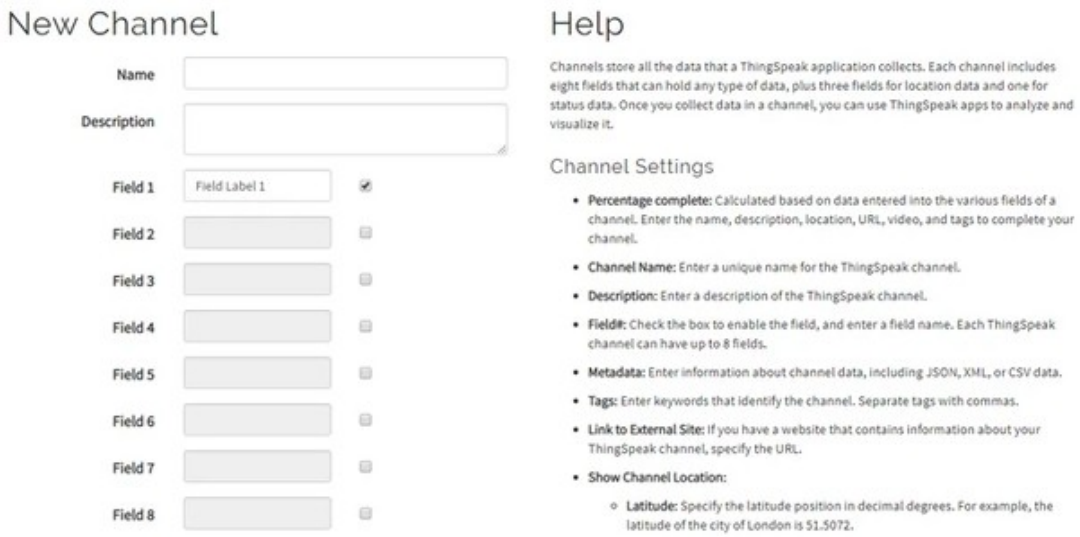


Рисунок 2.3 – Створення нового каналу даних

Зареєструвавши канал в ThingSpeak, ви можете відразу ж відправляти туди інформацію, обробляти її і отримувати до неї доступ. Канали можуть працювати з інформацією в форматах JSON, XML і CSV.

Щоб використовувати хмару IoT, нам потрібно відвідати ThingSpeak.com і зареєструватися. Це займе всього хвилину, а облікові записи користувачів безкоштовні. Якщо у вас є обліковий запис користувача, вам необхідно створити канал. У каналах ThingSpeak зберігаються дані (рис. 2.4).

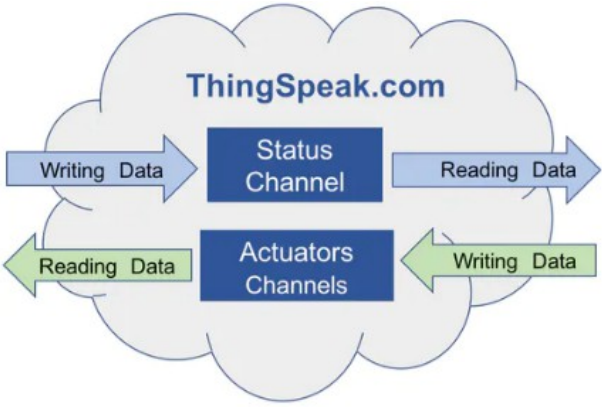


Рисунок 2.4 – ThingSpeaks IoT хмара

Щоб використовувати канали ThingSpeaks в середовищі Arduino IDE, нам потрібно вибрати Sketch / Include Library / Manage Libraries. У списку требі обрати бібліотеку ThingSpeak і натисніть кнопку «Встановити» (рис. 2.5).

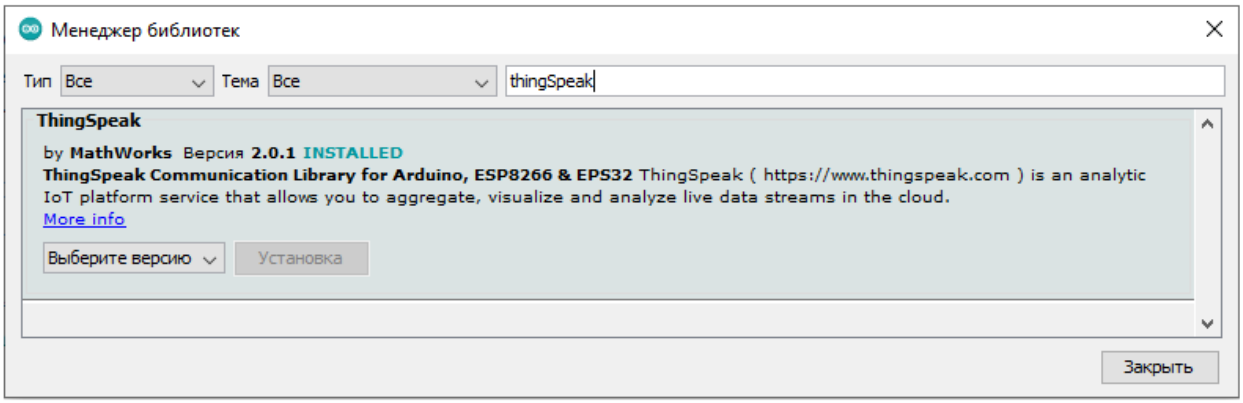


Рисунок 2.5 – Бібліотека ThingSpeaks для Ардуіно

2.3 Simulink-модель системи інформування пасажирів міського транспорту

Як уже зазначалося, пакет розширення Simulink системи MATLAB є ядром інтерактивного програмного комплексу, призначеного для математичного моделювання лінійних та нелінійних динамічних систем та пристроїв, представлених своєю функціональною блок-схемою, що називається S-моделлю, або просто моделлю. Для побудови функціональної

блок-схеми Simulink має велику бібліотеку блокових компонентів і зручний редактор блок-схем. Він заснований на графічному інтерфейсі користувача і є типовим засобом візуально-орієнтованого програмування.

Для створення системи у Simulink, треба запустити MATLAB та натиснути на відповідну кнопку на панелі задач (рис. 2.6).

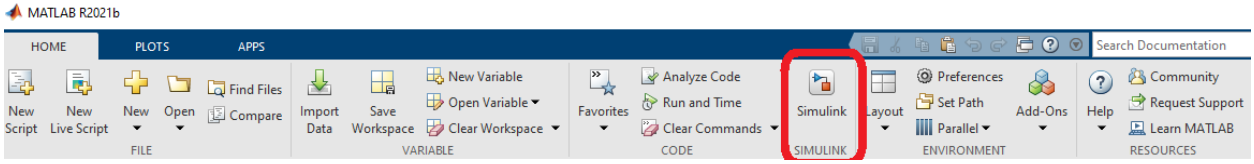


Рисунок 2.6 – Запуск Simulink

Використовуючи компоненти користувач за допомогою миші переносить потрібні блоки на робочий стіл Simulink і з'єднує лініями входи та виходи блоків (рис. 2.7).

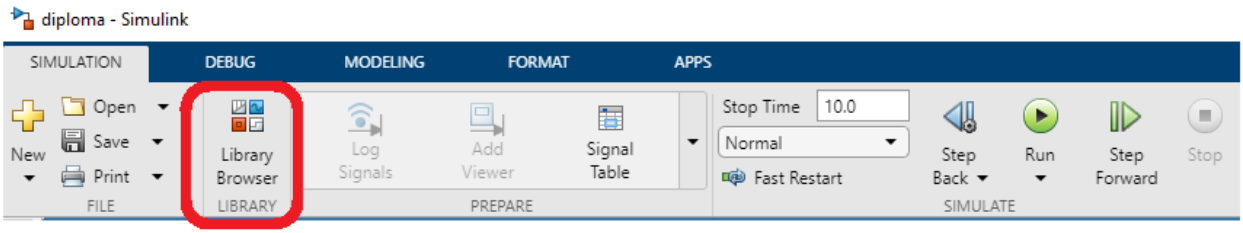


Рисунок 2.7 – Бібліотека компонентів

Таким чином, створюється діаграма (блок-схема) системи чи пристрою, тобто модель.

На рисунку 2.8 представлено модель системи інформування пасажирів міського транспорту, що побудована у системі Simulink. Спеціалізовані блоки для роботи з Arduino були взяті з бібліотеки Simulink Support Package for Arduino Hardware.

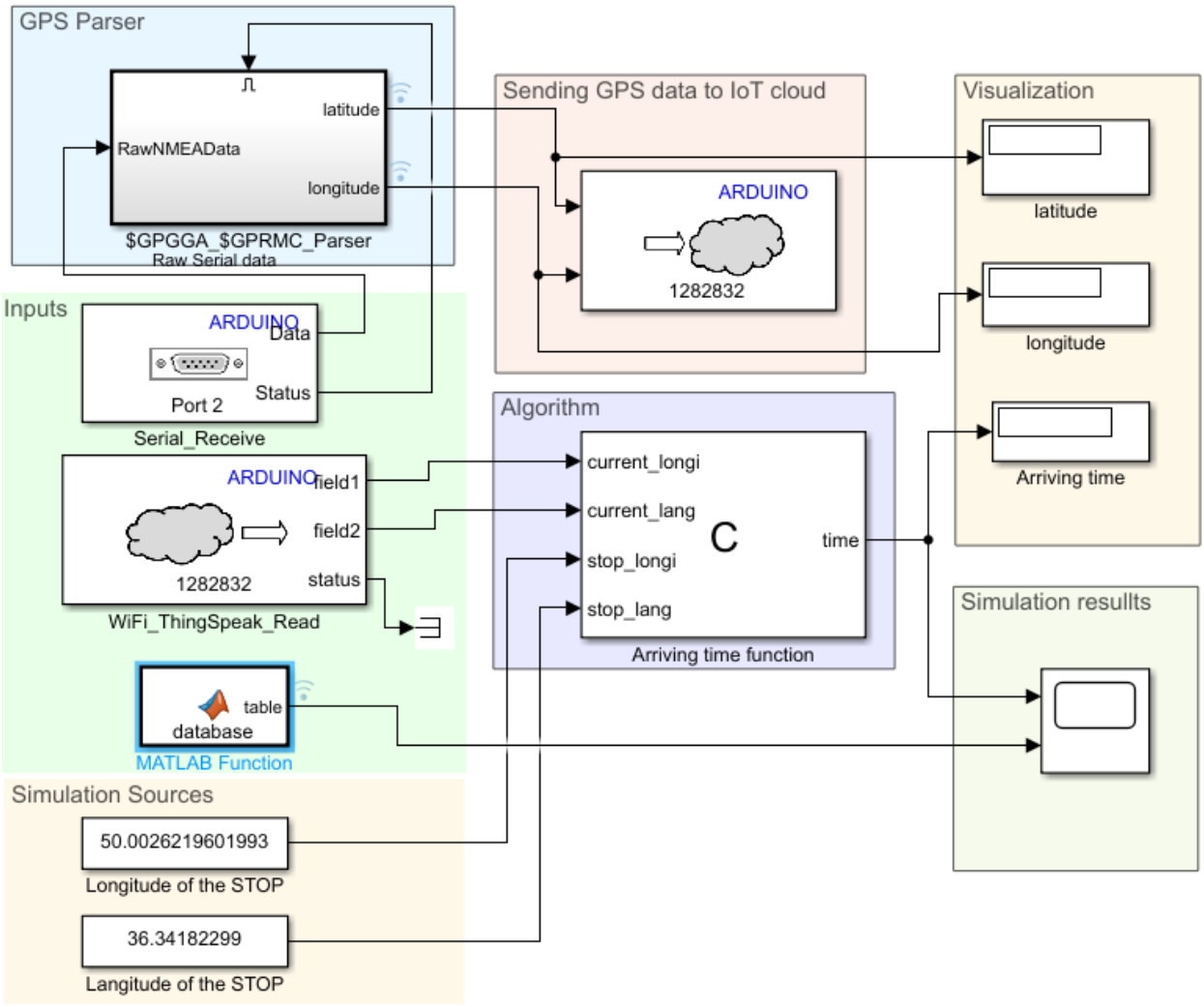
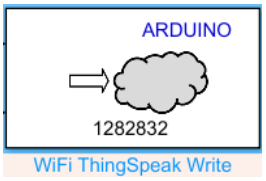
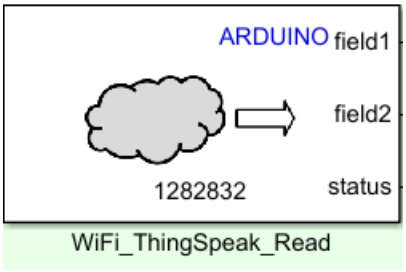
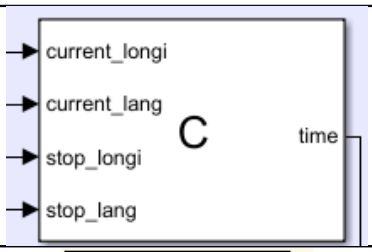
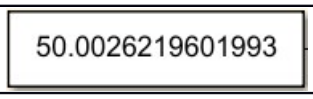


Рисунок 2.8 – Simulink модель СІП

Опис використаних блоків представлено у таблиці 2.1.

Таблиця 2.1 – Блоки Simulink моделі

Блок	Опис
	Блок S-функції “$GPGGA_$GPRMC_Parcer”– самостійна програма, написана мовою С. Блок описує алгоритм декодування NMEA повідомлень, що надходять до мікроконтролера Arduino через модуль GPS U-BLOX NEO-6M.
	Блок ThingSpeak Write записує дані до полів field1 та field2 каналу ThingSpeak 1282832.

	Блок ThingSpeak Read зчитує дані з каналу ThingSpeak. Кількість каналів даних на виході блоку залежить від кількості полів, які ви вводите у параметр Fields для читання. Параметр Fields для читання встановлено [1 2] визначає два вихідні порти з іменами портів field1 та field2. Якщо дані, отримані з каналу, є нульовими, блок виводить 0. Вихід status – відповідь сервера ThingSpeak для кожного запиту на читання.
	Блок завдання S функцій – блок, що призначений для обчислення часу прибуття транспорту на зупинку (мова С).
	Matlab-функція, що зчитує дані з координатими всіх зупинок маршруту та виводить їх на мапу.
	Віртуальний дисплей Display – пристрій для подання цифрової інформації.
	Будує графіки досліджуваних сигналів. Позволяє спостерігати за змінами сигналів у процесі моделювання.
	Джерело постійного впливу Constant визначає константу або вектор констант.
	Terminator – заглушка для невикористовуваних виходів.

В якості параметрів блока ThingSpeak Read виступають (рис. 2.9):

– Channel ID: 1282832;

– Channel access: Public;

– Fields to read: [1 2];

– Sample time: 10 sec.

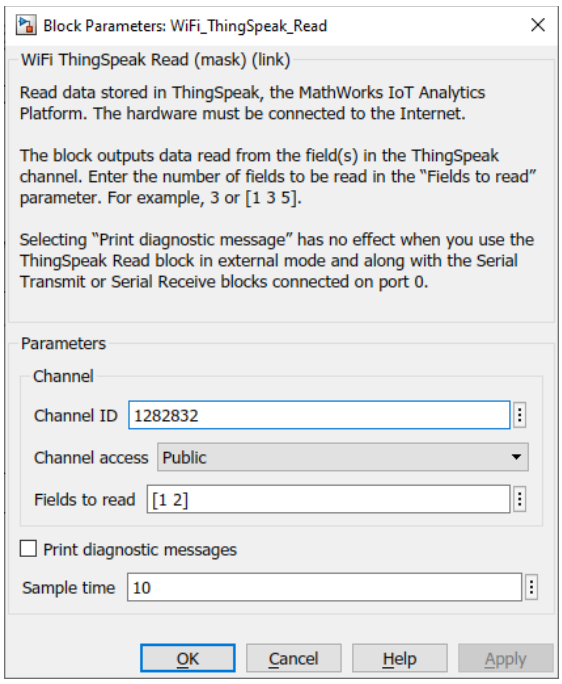


Рисунок 2.9 − Параметри блок aThingSpeak Read

В якості параметрів блока ThingSpeak Read виступають (рис. 2.10):

– Channel ID: 1282832;

– Write API key: AME6XG0SJ7LJU0SA;

– Fields to write: [1 2];

– Update interval (seconds): 10.

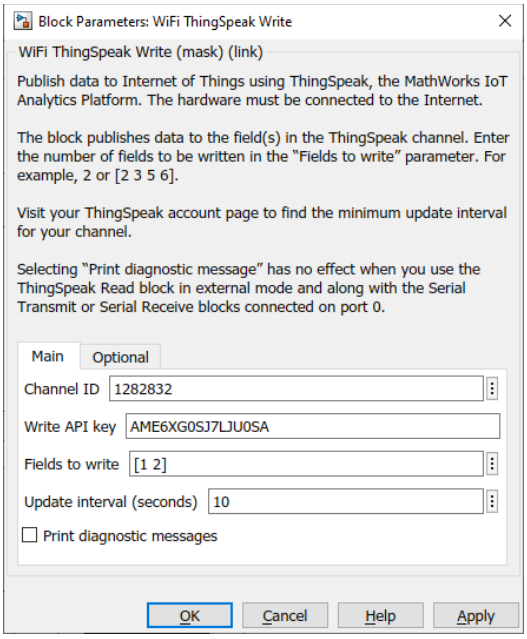


Рисунок 2.10 − Параметри блок aThingSpeak Write

Блок “$GPGGA_$GPRMC_Parcer” виконує декодування NMEA повідомлень, за допомогою яких GPS-модуль спілкуються з Arduino (рис. 2.11).

Рисунок 2.11 – NMEA-протокол

NMEA (National Marine Electronics Association) − це стандарт передачі даних обладнання навігації, зв'язку та інших інформаційних мереж [13]. NMEA повідомлення мають простий і зрозумілий формат, що дозволяє забезпечити обмін даними між GPS приймачами та навігаційними програмами. Дані передаються в послідовності, яка називається реченням. Кожне речення містить інформацію, таку як широта, довгота, швидкість і час, у вигляді символів ASCII. Речення може містити не більше 80 символів (рис. 2.12).

StartChar	SentenceName	Separator	$Data_1$	Separator	$Data_2$	Separator	...	$Data_N$	CheckChar	Checksum	EndChar

NMEA Header	Value	Description
StartChar	$	ASCII for 36
SentenceName	For example, GPGGA	NMEA Sentence Identifier
Separator	,	ASCII for 44
Data_1 to Data_N	For example, 083445.00,1256.60109,N	Data fields, such as latitude and longitude
CheckChar	*	ASCII for 42
CheckSum	For example, 7E	Hexadecimal number representing 8-bit exclusive OR of all characters between $ and *
EndChar	CR	Carriage return

Рисунок 2.12 – Формат речення NMEA

GPS-плата U-BLOX NEO-6M, що використана у цьому проекті, підтримує шість типів речень NMEA: $GPGSV, $GPGLL, $GPRMC, $GPVTG, $GPGGA та $GPGSA.

Для декодування $GPGGA (рис. 2.13) та $GPRMC (рис. 2.14) речень розроблено синтаксичний аналізатор.

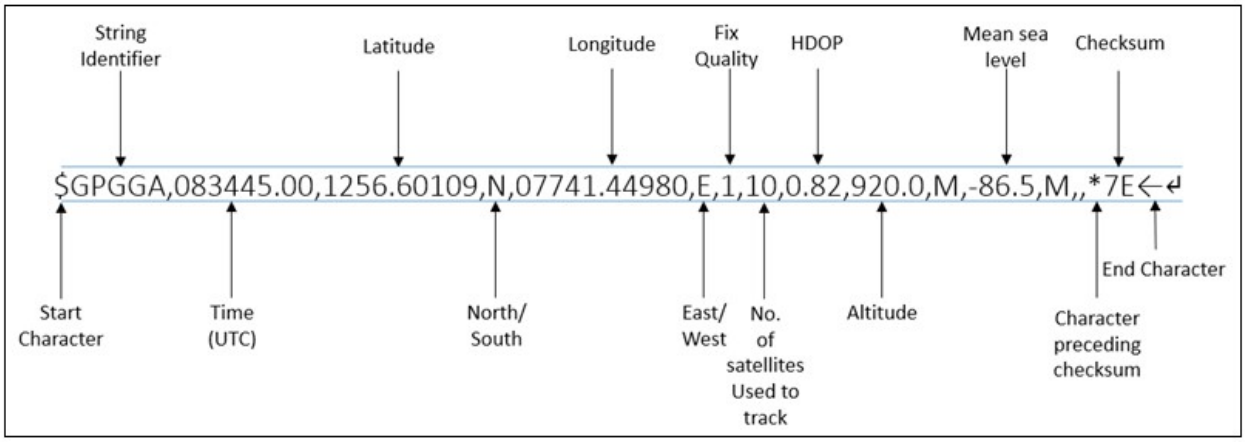


Рисунок 2.13 – Формат речення $GPGGA

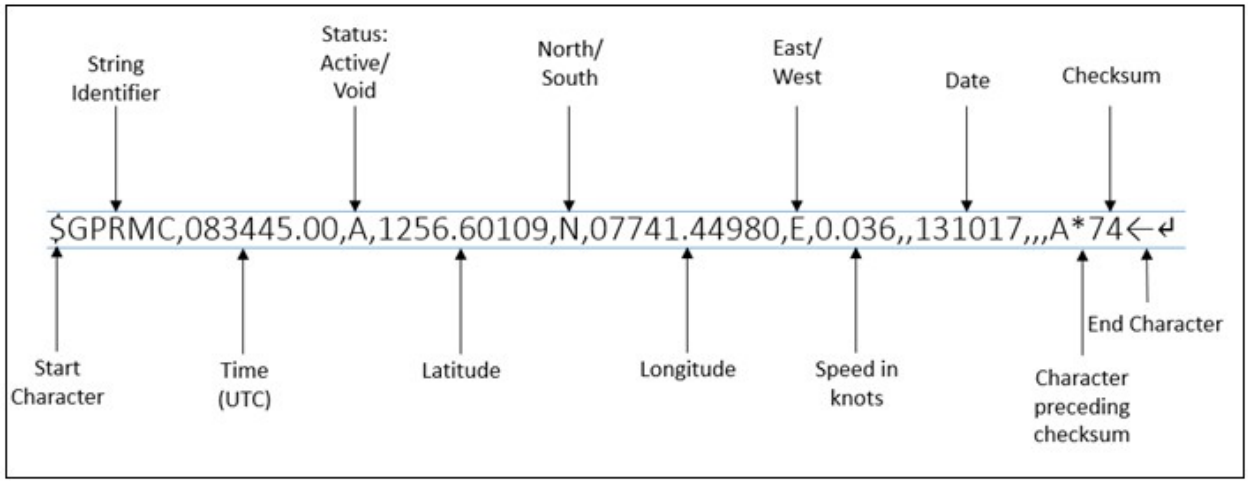


Рисунок 2.14 – Формат речення $ GPRMC

GPS-плата надсилає дані на послідовний порт 2 плати Arduino, отриманні дані зберігаються у змінних Matlab workspace.

Алгоритм роботи аналізатора можна представити наступним чином (рис. 2.15).

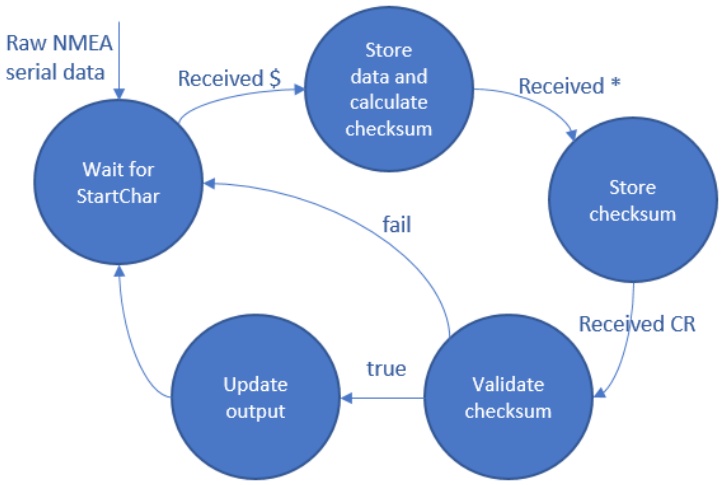


Рисунок 2.15 – Алгоритм “$GPGGA_$GPRMC_Parcer”

Отримана апаратно-програмна модель дозволяє відпрацьовувати алгоритми керування, налаштовувати параметри та досліджувати різні режими роботи системи інформування пасажирів громадського транспорту у реальному часі за допомогою пакету Simulink програмного середовища Matlab та функціональних можливостей платформи Arduino Uno.

Код представлено у додатку Б (лістинг Б.1).

3 РЕАЛІЗАЦІЇ МОДЕЛІ СИСТЕМИ ІНФОРМУВАННЯ ПАСАЖИРІВ МІСЬКОГО ТРАНСПОРТУ НА БАЗІ ARDUINO

3.1 Технологічна платформа реалізації

3.1.1 Плата Arduino Uno Rev3

Для нашого проекту використовується мікроконтролер Arduino Uno R3 [13]. Модель R3 є третьою та останньою версією Arduino Uno – плати на базі мікроконтролера ATmega328 (рис. 3.1).

Рисунок 3.1 – Плата Arduino Uno

ATmega328 має 32 КБ пам’яті (0,5 КБ з яких зайнято загружчиком). Він також має 2 КБ SRAM та 1 КБ EEPROM (які можна читати і записувати за допомогою бібліотеки EEPROM). Він має 20 цифрових входів / виходів (з яких 6 можуть використовуватися як виходи ШІМ, а 6 можуть використовуватися як аналогові входи), інтерфейс USB, роз'єм живлення, заголовок внутрисхемного системного програмування (ICSP) і кнопку скидання. Він підключається до комп'ютера за допомогою кабелю USB. Vin − це вхідна напруга для плати Arduino, коли вона використовує зовнішнє джерело живлення (на відміну від 5 вольт від USB-з'єднання або іншого регульованого джерела живлення). Плата мікроконтролера може отримувати живлення або від роз'єму постійного струму (7 - 12 В), роз'єму USB (5 В), або від контакту Vin плати (7 - 12 В). Подача напруги через контакти 5 В або

3,3 В обходить регулятор і може пошкодити вашу плату. Тому не рекомендується цього робити. Максимальний споживаний струм складає 50 мА. Плата Arduino заснована на мікросхемі мікроконтролера AVR, і коли плата нічого не підключено споживає близько 80 мА струму при 5 вольт. Тактова частота Arduino становить 16 МГц, тому він може виконувати ту чи іншу задачу швидше, ніж інший процесор або контролер. Чіп AVR постійно працює на частоті 16 МГц, незалежно від того, що робить код, він ніколи не зупиняється, тому його поточне споживання в основному не залежить від коду, який він виконує. Плата Arduino підтримує також інтерфеси I2C і SPI. Програмне забезпечення Arduino включає необхідні для цього бібліотеки [14].

Модуль Arduino UNO має порти для зв'язку з комп'ютером, з іншого платою UNO або з іншими мікроконтролерами. Для цього на платі є інтерфейс UART з логічними рівнями TTL (5 В), підключений до контактів 0 (RX) і 1 (TX). Для побудови електронного прототипу може використовуватися макетна плата (рис. 3.2).

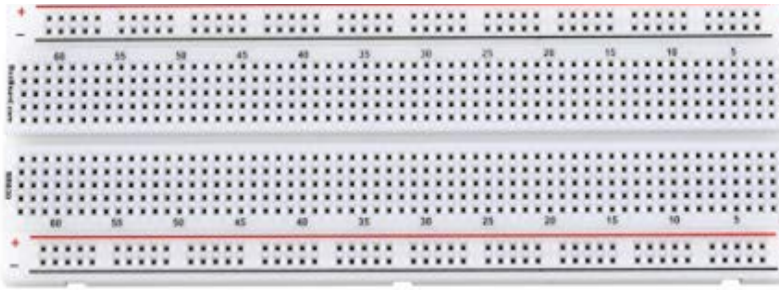


Рисунок 3.2 – Макет для проектів Arduino

Основні переваги:

– може обслуговувати одну схему з декількома модулями;

– можна з'єднати їх один з одним і уникнути плутанини з великою кількістю проводів;

– проста у використанні;

– можна додати кнопки, резистори, тощо.

Модуль Ethernet W5100

Модуль Etherrnet W5100 дозволяє легко перетворити контролер в мережеве пристрій − перетворити Arduino в простий веб-сервер або використовувати Інтернет для читання/запису його цифрових і аналогових виходів/входів. Він безпосередньо підтримується офіційною бібліотекою Ethernet Arduino. Також він підтримує читання і запис на міні-SD-карту (TF-карту), роз'єм для якої знаходиться на платі. Модуль має масштабовану конструкцію. Його можливо не тільки підключити безпосередньо до плати Arduino, але і встановити на нього інші модулі. Варто відзначити, що на платі встановлені резистори на 51 Ом, згідно даташіту до мікросхеми W5100, що призводить до значного відведення тепла і необхідності використання додаткового радіатора відведення тепла. Модуль впевнено працює з Ethernet 10/100 Мбіт/с, але його робота з деякими маршрутизаторами на 1000 Мбіт/с не гарантовано. Модуль Ethernet W5100 представлений на рисунку 3.3.

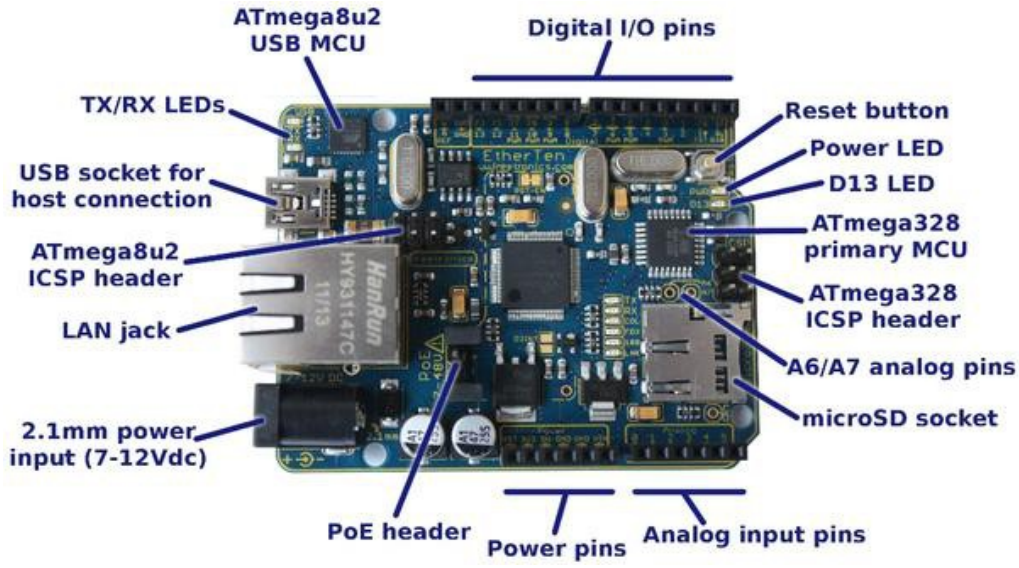


Рисунок 3.3 – Модуль Ethernet W5100

Характеристики модуля Ethernet W5100:

- інтерфейсний чіп: W5100;
- мережевий інтерфейс: Ethernet 10/100 Мбіт/с;
- напруга живлення 5В;

– сумісний з модулями Arduino;
– карта micro-SD для зберігання файлів;
– підтримка протоколів TCP/IP.

Мається на увазі, що для проектів, пов'язаних з підключенням Arduino до мережі, потрібно мати хоча б загальні знання мережевих технологій. Сьогодні Ethernet є ключовою і найбільш поширеною технологією для організації локальних мереж поряд з WiFi. У стандартній моделі OSI він знаходиться на канальному і фізичному рівнях, визначаючи підрівні управління доступом до середовища передачі і управління логічним каналом. Творцем Ethernet була компанія Xerox, її інженер Роберт Метклаф створив технологію як інструмент для підключення багатьох комп'ютерів до загальних ресурсів в локальній мережі. Технологія стала офіційним стандартом в 1982 році після появи специфікації IEEE802.3. Проста конфігурація Ethernet показана на рисунку 3.4.

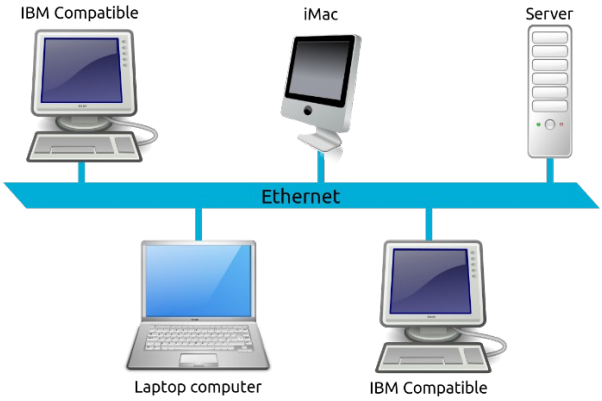


Рисунок 3.4 – Просте підключення по інтерфейсу Ethernet

Мікроконтролер ESP8266 NodeMCU

Плата ESP8266 NodeMCU поставляється з модулем ESP-12E з 32-бітовим мікропроцесором LX106 RISC Tensilica Xtensa. Цей мікропроцесор працює на тактовій частоті від 80 МГц до 160 МГц. ESP8266 NodeMCU має 128 КБ ОЗУ і 4 МБ флеш-пам'яті для зберігання даних і програм. Він інтегрований з функціями Wi-Fi/Bluetooth і Deep Sleep Operating, що ідеально

підходить для проектів IoT. Давайте розглянемо особливості та характеристики ESP8266 NodeMCU:

– вбудована антена;
– SRAM: 64кб;
– 13 контактів GPIO, 10 каналів ШІМ, I2C, ADC і 1-wire;
– проста у використанні;
– робоча напруга: 3,3 В;
– Вхід напруги: від 7 до 12 Вольт;
– SPI: 1;
– мережеві проекти;
– стандарти пристроїв IoT;
– програмується за допомогою Arduino IDE або мови IUA;
– невеликий модуль ідеально підходить для проектів IoT.

Виводи (піни) ESP8266 NodeMCU представлені на рисунку 3.5.

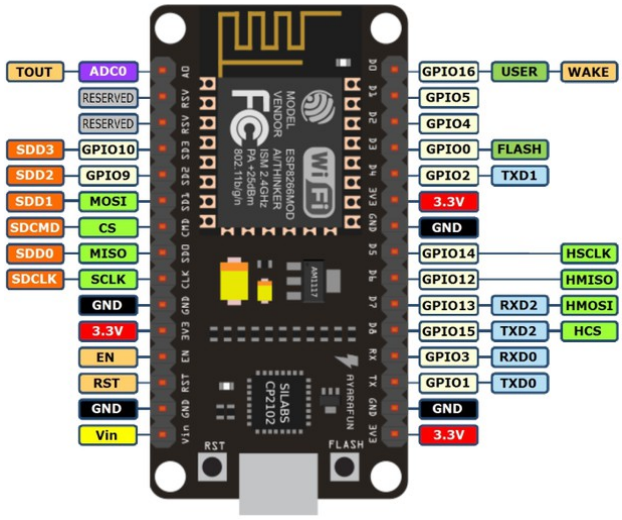


Рисунок 3.5 – Піни модуля ESP8266 NodeMCU

Для установки і програмування ESP8266 NodeMCU на Arduino IDE потрібне програмне забезпечення Arduino IDE, де необхідно вибрати налаштування (Preferences) в меню File і ввести скопійований URL (http://arduino.esp8266.com/stable/package_esp8266com_index.json) в розділ

Additional Board Manager URLs. Потім натисніть ОК (рис. 3.6).

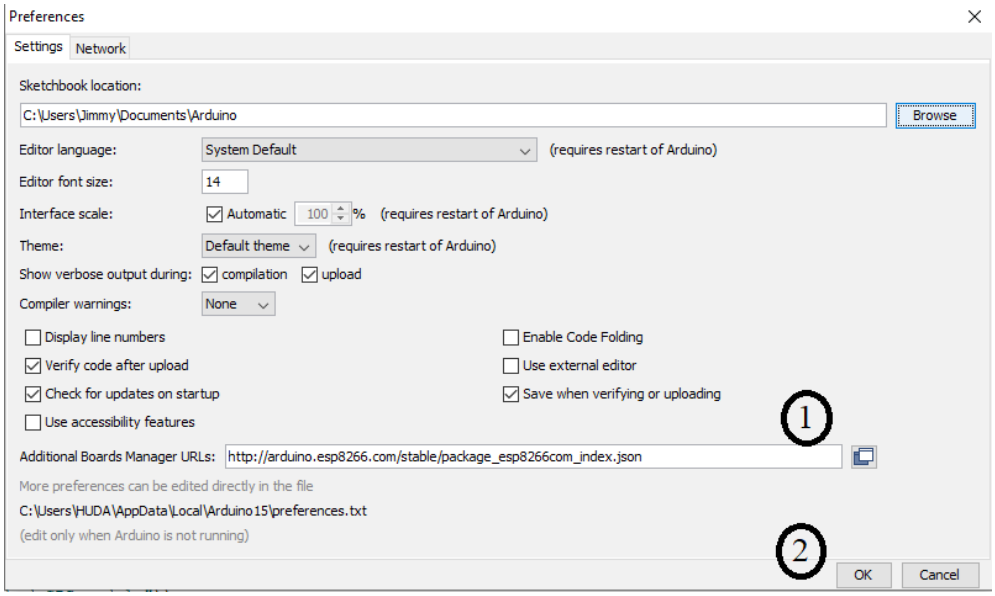


Рисунок 3.6 – Налаштування Arduino

Встановіть плату ESP8266, скориставшись пошуком в диспетчері плат в меню «Інструменти» (рис. 3.7).

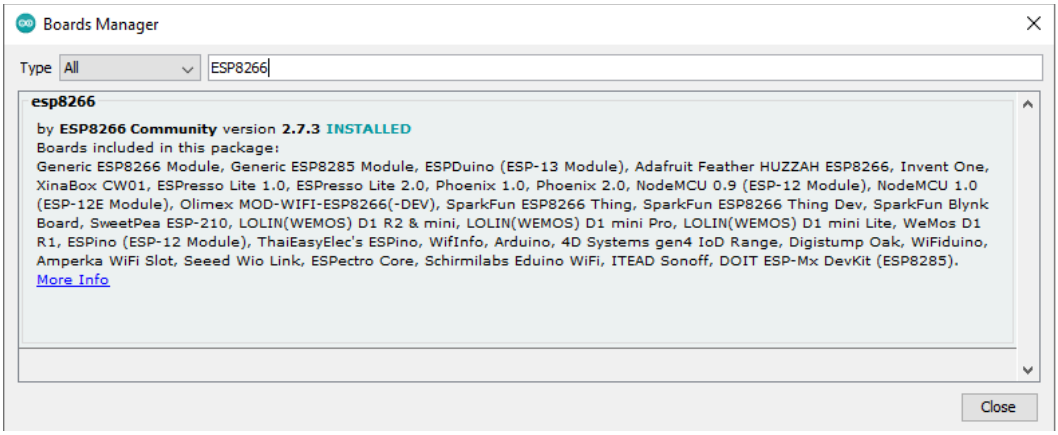


Рисунок 3.7 – Менеджер додаткових модулів

Після виконання всіх попередніх кроків ви побачите ESP8266 Board і режим NodeMCU (рис. 3.8).

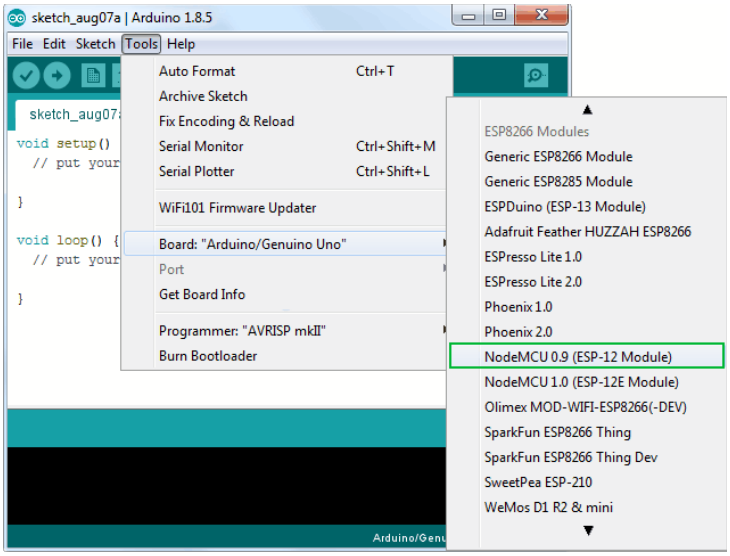


Рисунок 3.8 – Вибір необхідного мікроконтролеру

Тепер можливо писати код і завантажувати його в скетч.

GPS-модуль Ublox NEO-6M

GPS-модуль NEO-6M − це повноцінний GPS-приймач з вбудованою керамічної антеною 25x25x4 мм, що забезпечує широкі можливості пошуку супутників (рис. 3.9).

Рисунок 3.9 – Модуль NEO-6M GPS

За допомогою індикаторів живлення та сигналів можливо контролювати стан модуля. Завдяки резервної батареї даний модуль може зберігати дані при випадковому відключенні основного живлення. Принципова схема модуля представлена на рисунку 3.10.

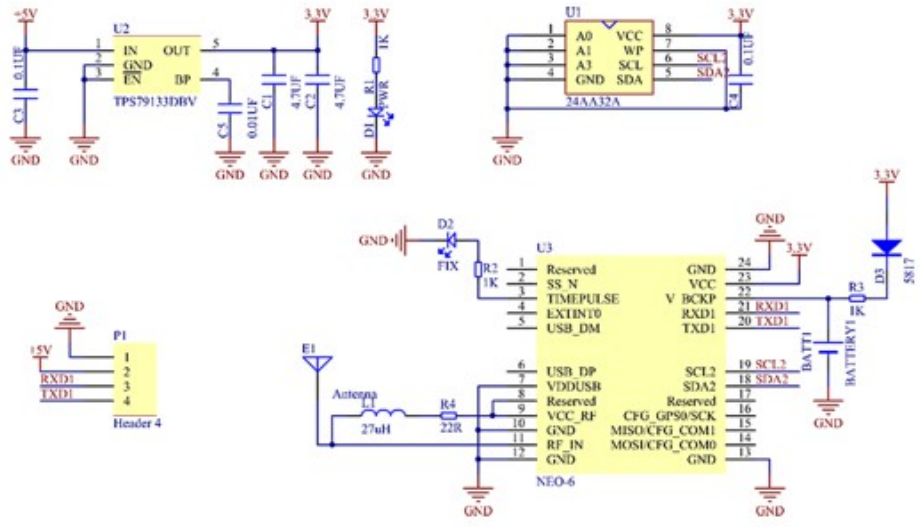


Рисунок 3.10 – Схема GPS модуля

GPS-модуль U-blox NEO-6M має високу чутливість для використання всередині приміщень. Крім того, є одна сумісна з MS621FE акумуляторна батарея для резервного копіювання та EEPROM для зберігання налаштувань конфігурації. Модуль добре працює з входом постійного струму в діапазоні від 3,3 до 5 В. Як зазначено, модулі GPS засновані на движку GPS U-blox NEO-6M. Модуль NEO-6M має інтерфейс UART для послідовного зв'язку, але швидкість передачі даних UART (TTL) за замовчуванням становить 9600 бод. Патч-антени плоскі, зазвичай мають керамічний і металевий корпус і встановлюються на металевій опорній пластині. Їх часто відливають в корпусі. Положення кріплення антени дуже важливо для оптимальної роботи GPS-приймача. При використанні патч-антени вона повинна бути орієнтована паралельно географічної горизонту. Антена повинна мати повний огляд неба, забезпечуючи пряму видимість з максимально можливою кількістю видимих супутників.

Для швидкого тестування за допомогою комп'ютера під керуванням Windows вам просто потрібно встановити послідовну зв'язок з модулем GPS за допомогою одного адаптера USB-UART, такого як модуль PL2303 USB-to-Serial Converter. Налаштування обладнання приведена в таблиці 3.1.

Таблиця 3.1 – Налаштування обладнання

NEO-6M GPS module	*USB-TO-SERIAL converter*
TX	RX
RX	TX
GND	GND
VCC	5 V

Потім завантажити та встановити засіб налагодження / оцінки для ПК з Windows «U-center». Коли модуль GPS працює, зелений індикатор на модулі GPS блимає (червоний − для індикації включення), а цифри, які стосуються часу, широті, довготі і т.п. будуть відображатися в «U-center».

Зверніть увагу, що в прикладах, представлених в бібліотеці TinyGPS + +, передбачається, що швидкість передачі даних для модуля GPS становить 4800 бод, але необхідно змінити її на 9600 для модуля NEO-6M.

Рідкокристалічний дисплей

Рідкокристалічний дисплей (РК) − це дисплей електронного модуля, який використовується в багатьох проектах (рис. 3.11).

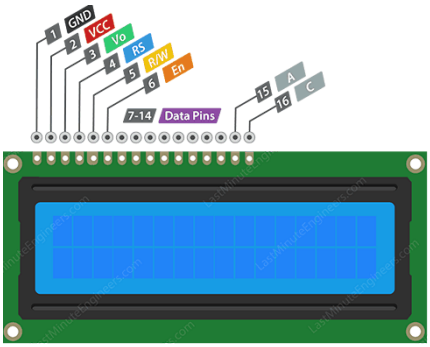


Рисунок 3.11 – LCD екран

Характеристики РК-дисплея:

– робоча потужність від 4,7В до 5,3В;
– відображає буквено-цифрові данні;
– працює в 4-бітному і 8-бітному режимах;

– може відображати будь-який символ;

– споживання при роботі 1 мА без підсвічування;

– доступні кольори підсвічування − зелений і синій.

РК-дисплей 16×2 − це дуже простий модуль, який можна легко запрограмувати, він може містити до 16 символів в рядку, всього 32 символу в обох рядках, розмір кожного відображуваного символу становить матрицю 5x7 пікселів. Він здатний відображати до 224 різних знаків і символів. На РК-дисплеї є 2 типу регістрів: регістр команд або регістр даних. Піни LCD дісплея 16×2 представлені у таблиці 3.2.

Таблиця 3.2 – Распіновка LCD екрану

Пін №	*Им'я*	*Опис*
1	Vss	Ground pin (Green Screen)
2	Vcc	LCD Power +5v
3	Vo	Contrast Control
4	Rs	Register Select
5	Rw	Read Write
6	E	Enable
7	D0	Data Pin 0
8	D1	Data Pin 1
9	D2	Data Pin 2
10	D3	Data Pin 3
11	D4	Data Pin 4
12	D5	Data Pin 5
13	D6	Data Pin 6
14	D7	Data Pin 7
15	A	Anode (LED+) (+5v)
16	C	Cathode (LED-) (GND)

Регістр команд зберігає командні інструкції такі як:

– ініціалізація РК-дисплею;

– очищення перед використанням;

– установка положення курсора;

– управління дисплеєм та ін.

3.2 Збірка моделі системи інформування пасажирів міського транспорту

Підключення модулів ESP8266 NodeMCU та GPS NEO-6M

Для початку роботи з модулем ESP8266 необхідно завантажити та встановити бібліотеку «TinyGPS ++», яка є бібліотекою для аналізу потоків даних, що надаються модулями GPS. Після цього підключимо GPS-модуль NEO-6M до ESP8266 NodeMCU як показано на рисунку 3.12.

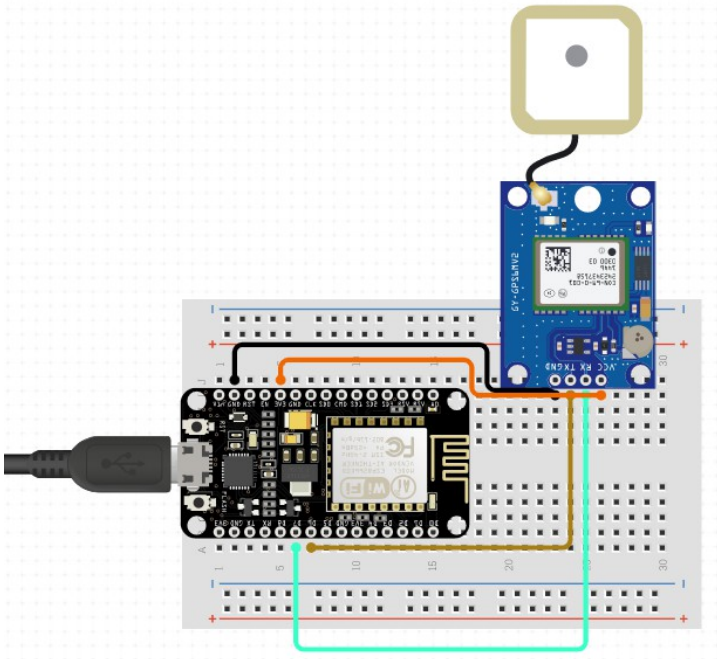


Figure 3.12 – Модулі NEO-6M GPS та ESP8266

Налаштування обладнання показано у таблиці 3.3.

Table 3.3 – Налаштування

NEO-6M GPS module	*ESP8266*
TX	D6
RX	D7
GND	GND
VCC	3.3V

Для перевірки коректності роботи відкриємо будь-який приклад з бібліотеки TinyGPS++ та завантажимо його на плату. Координати, що надходять з послідовного порту GPS, передаються на послідовний монітор у достатньо зрозумілому форматі (рис. 3.13).

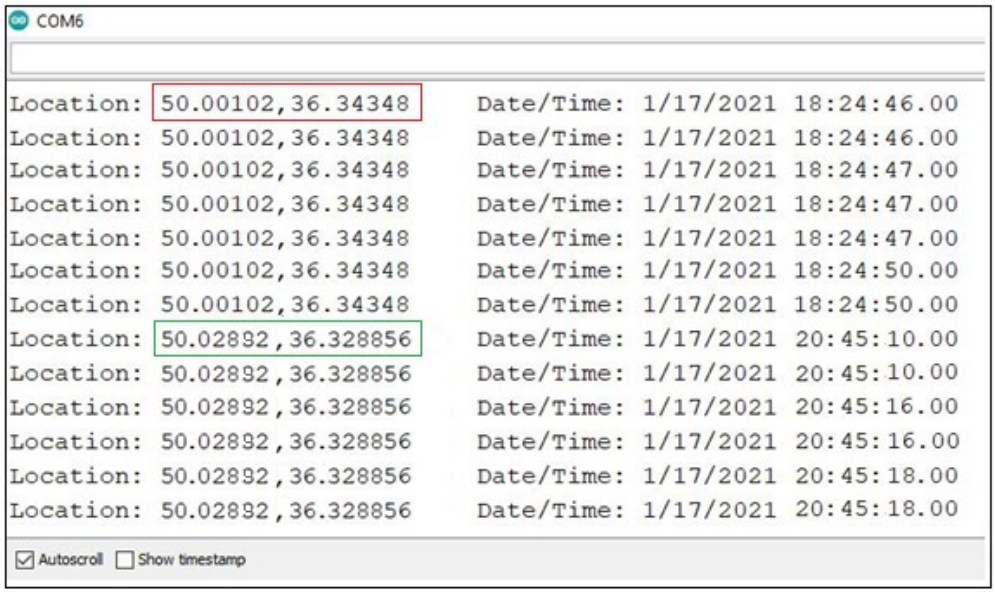


Figure 3.13 – Послідовний монітор середовища Arduino IDE

Відмітимо, приклади, надані в бібліотеці TinyGPS++, передбачають швидкість 4800 бод для модуля GPS, але нам потрібно змінити її на 9600 для модуля NEO-6M. Дані, що передаються модулем GPS NEO-6, відповідають стандарту 0183 Національної асоціації морської електроніки (NMEA), який підтримує односторонню послідовну передачу даних від одного оратора до одного або кількох слухачів.

Налаштування сервісу Thingspeak IoT

Перш за все необхідно зареєструватися на сайті https://thingspeak.com/. Наступний крок – створення свого каналу (рис. 3.14).

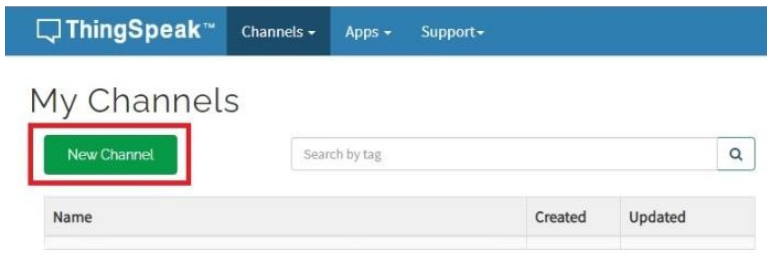


Рисунок 3.14 – Створення нового каналу

Після створення каналу (channel ID: 1282832) на сайті ThingSpeak ми заповнили його необхідною інформацією, а саме назвами змінних, що будуть надходити: field 1 (довгота) та field 2 (широта) (рис. 3.15).

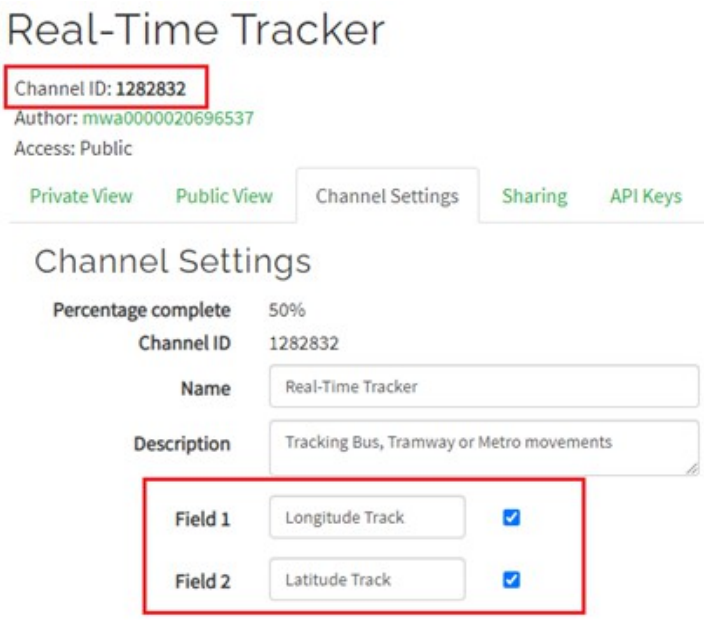


Рисунок 3.15 – Сторінка створеного каналу

Після цього ми перевіряємо сформовані ключі API, які будуть використовуватися для зчитування даних (read API key: *IMHMPHI1XS062OMS*) та для запису (write API key: *AME6XG0SJ7LJU0SA*) (рис. 3.16).

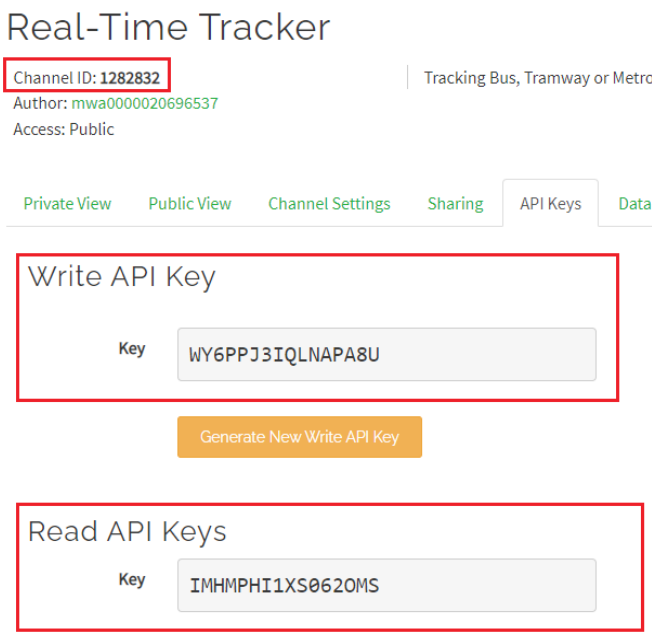


Рисунок 3.16 – API ключі каналу

Запрограмуємо ESP8266 NodeMCU для завантаження даних GPS на ThingSpeak за допомогою Arduino IDE. Підключаємо ESP8266 до Wi-Fi, щоб отримати доступ до Інтернету (рис. 3.17). Використані бібліотеки: TingGPS++, SoftwareSerial, ThingSpeak, ESP8266WiFi.

```
ESP822NodeMCU
#include <TinyGPS++.h>
#include <SoftwareSerial.h>
#include "ThingSpeak.h"
#include <ESP8266WiFi.h>
static const int RXPin = 12, TXPin = 13;
static const uint32_t GPSBaud = 9600;
const char* ssid     = "Jimmy";  Wi-Fi Name
const char* password = "Mind_Your_Own_Business!"; Wi-Fi Password
unsigned long myChannelNumber = 1282832; Channel ID
const char * myWriteAPIKey = "WY6PPJ3IQLNAPA8U"; Write API Key
TinyGPSPlus gps;
WiFiClient  client;
SoftwareSerial ss(RXPin, TXPin);
void setup()
{
  Serial.begin(115200);
  ss.begin(GPSBaud);
  Serial.println(F("DeviceExample.ino"));
  Serial.println(F("A simple demonstration of TinyGPS++ with an attached GPS module"));
  Serial.print(F("Testing TinyGPS++ library v. ")); Serial.println(TinyGPSPlus::libraryVersion());
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
```

Рисунок 3.17 – Фрагмент скетчу для ESP8266 з GPS

На рисунку 3.17 видно, що було внесено всю необхідну інформацію,

таку як Wi-Fi, пароль, ThingSpeak ID та ключ Write API. Повний код представлено на лістингу Б.2.

З’єднання LCD екрану з модулем Ethernet

Подключение Arduino Uno rev3 к Ethernet Shield показано на рисунку 3.18.

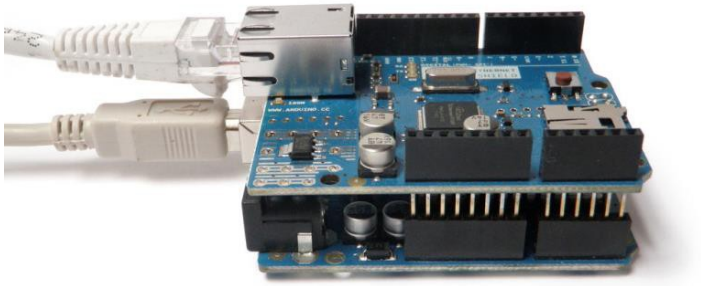


Figure 3.18 − Підключення Ethernet до Arduino Uno Rev3

При підключенні LCD-дисплею до макетної плати, необхідно додати потенціометр для контролю контрастності, як показано на рисунку 3.19.

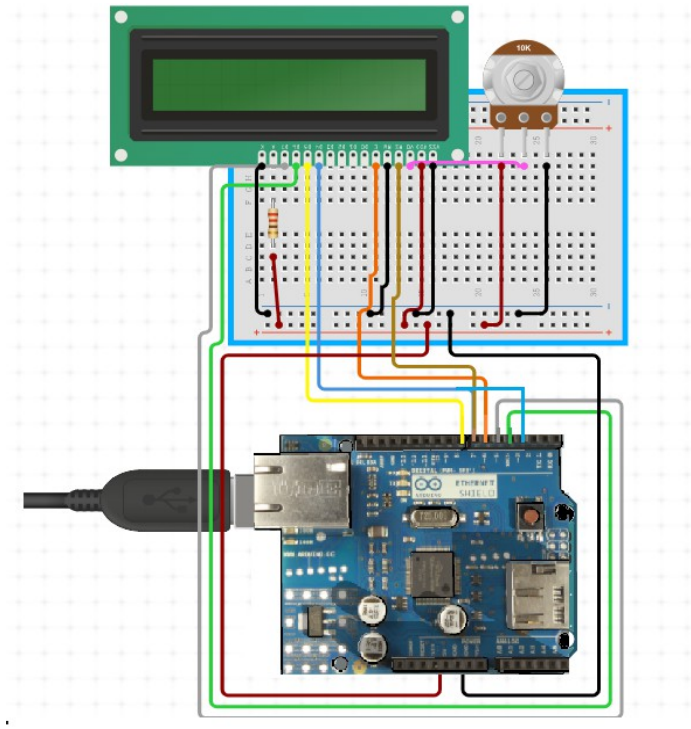


Рисунок 3.19 − Підключення Ethernet+Arduino та LCD-дисплею

У таблиці 3.4 наведені порти для підключення (Arduino Uno Rev3 +

Ethernet shield) та LCD –дисплею.

Table 3.4 – Connection Arduino Uno Rev3 + Ethernet shield with LCD

Pins of LCD		*Arduino Uno + Ethernet Shield*
1	Vss	Ground
2	Vcc	+5V
3	Vo	Potentiometer (signal pin)
4	Rs	Pin No.2
5	Rw	Ground
6	E	Pin No.3
7	D0	Not Used
8	D1	Not Used
9	D2	Not Used
10	D3	Not Used
11	D4	Pin No.4
12	D5	Pin No.5
13	D6	Pin No.6
14	D7	Pin No.7
15	A	+5V
16	C	Ground

Необхідно використовувати наступні бібліотеки Arduino IDE: Ethernet, LiquidCrystal, ThingSpeak. На рисунку 3.20 показано, що MAC-адреса створена випадковим чином.

```
#include <Ethernet.h>
#include <LiquidCrystal.h>
#include "ThingSpeak.h"
#define SECRET_MAC {0xDA, 0xDB, 0xDC, 0xDD, 0xDE, 0xDF}
byte mac[] = SECRET_MAC;
// with the arduino pin number it is connected to
const int rs = 2, en = 3, d4 = 4, d5 = 5, d6 = 6, d7 = 7;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
// Set the static IP address to use if the DHCP fails to assign
IPAddress ip(192, 168, 88, 239);
EthernetClient client;
//---------Channel Details---------//
unsigned long counterChannelNumber = 1282832;            // Channel ID
const char * myCounterReadAPIKey = "IMHMPHI1XS062OMS"; // Read API Key
const int FieldNumber1 = 1;  // The field you wish to read
const int FieldNumber2 = 2;  // The field you wish to read
//-----------------------------//
```

Figure 3.20 – Фрагмент скетчу для Ethernet та LCD

IP-адреса була отримана автоматично через DHCP (рис. 3.21). Повний код представлено на лістингу Б.3.

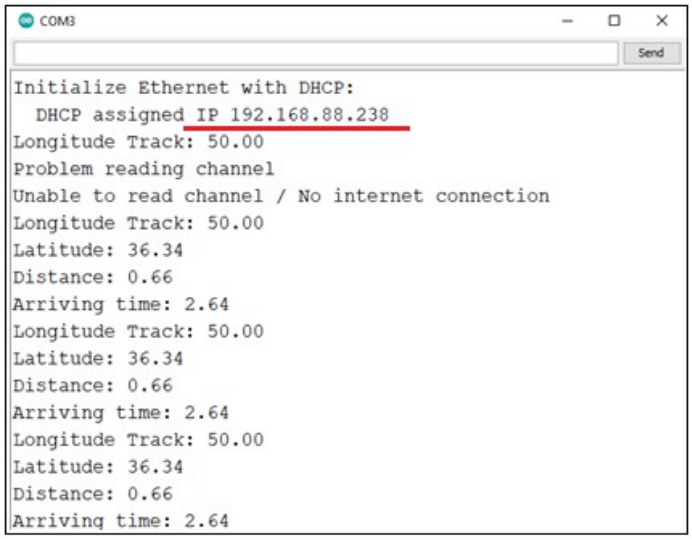


Figure 3.21 – Монітор послідовного порту середовища Arduino IDE

3.3 Перевірка працездатності запропонованої моделі

Перевірку запропонованої моделі можно виконати двома способами: запрограмувавши мікроконтролер або шляхом моделювання в системі Matlab.

Виконавши збірку моделі (рис. 3.22) та налаштувавши IoT сервісу ThingSpeak, можна переходити до фізичного тестування.

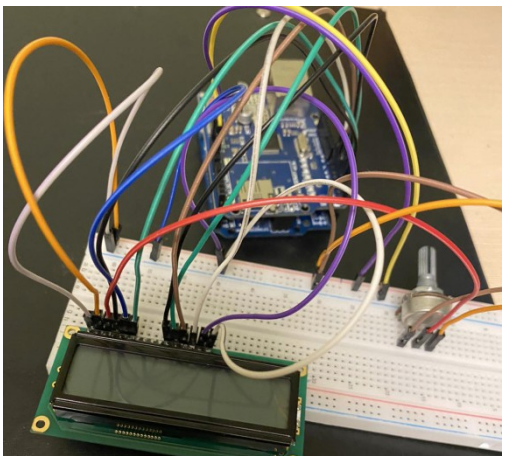


Figure 3.22 – Збірка модуля Ethernet W5100 та плати Arduino Uno

GPS-модуль треба розмістити на будь-якому рухливому об’єкті для прийому сигналу від супутників. Модуль GPS почне блимати, що означає, що є з'єднання з супутником (рис. 3.23).

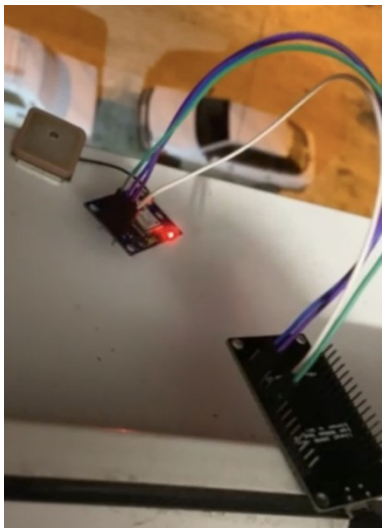


Рисунок 3.23 – Встановлення зв’зяку GPS модуля та супутника

Наступним кроком було перевірити дані (координати), які були надіслані на хмарний сервер IoT. На рисунку 3.24 показано,що модуль ESP8266 оновлює необхідні поля із затримкою 15 секунд.

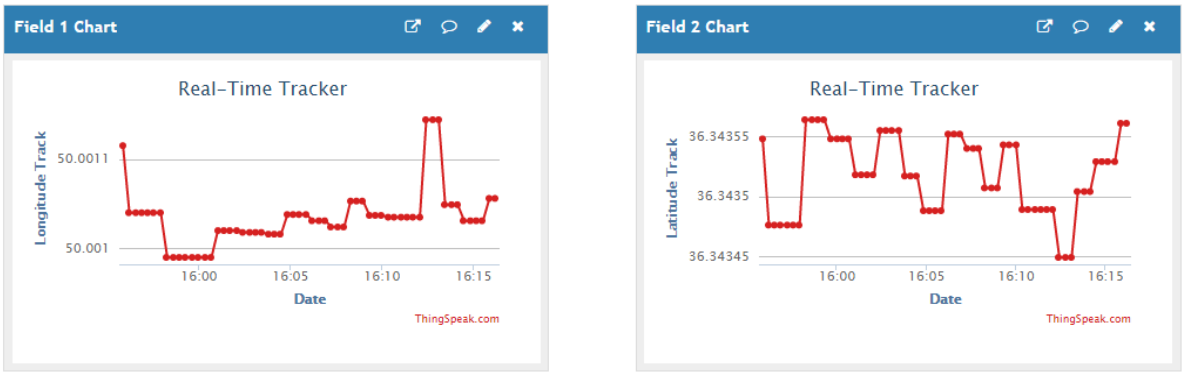


Рисунок 3.24 – Загальнодоступний вид каналу Thingspeak

В той же час, на LCD-дисплеї ми отримаємо повідомлення «Час прибуття:» (рис. 3.25).

Рисунок 3.25 – Початкове повідомлення

Кожні 15 секунд у другому рядку буде оновлюватися параметр часу, що залишився до прибуття транспорту до певної зупинки (рис. 3.26). Змінюючи положення модуля GPS, ми імітуємо рух транспорту по маршруту. Оскільки ми перемістили модуль лише на кілька метрів, час не сильно змінився.

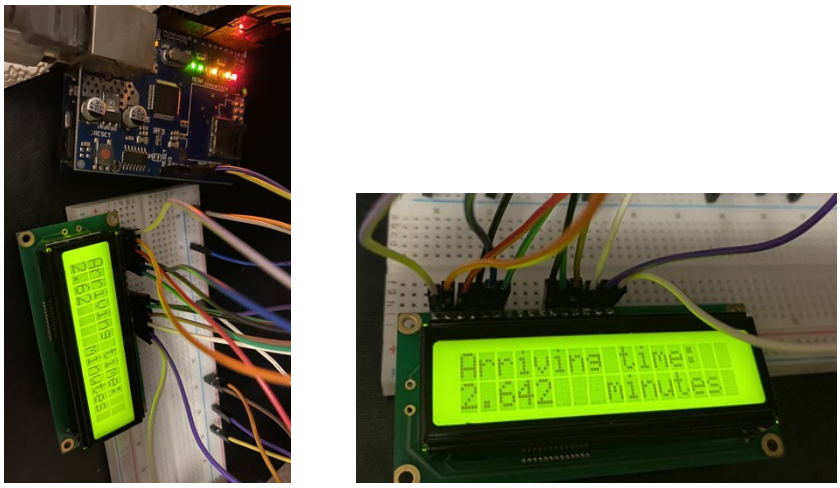


Рисунок 3.26 – Результат роботи системи

Проведені дослідження дали підстави стверджувати, що Matlab як середовище для модельно-орієнтованого проектування, надає багато переваг розробнику подібних систем [15]. Перевірку працездатності моделі можна виконати ще до збірки моделі, з використанням Simulink.

Створена Simulink-модель (рис. 2.8) дозволяє перевірити створену модель системи інформування пасажирів міського транспорту шляхом імітування руху того чи іншого транспорту.

В якості прикладу міського транспорту було взято трамвай № 26 м. Харкова. Маршрут цього трамвая від зупинки «Південно-Східна» до «Лісопарку» має 51 зупинку, координати яких представлено у таблиці 3.5.

Таблиця 3.5 – Координати зупинок трамвая № 26 м. Харкова

ID	ЗУПИНКИ	Координати	
		Latitude	Longitude
1	Південно-Східна	49,93893818	36,42370463
2	Плитковий завод	49,94030263	36,42084921
3	Плиткова (Московський проспект)	49,94228999	36,41483586
4	Метро Індустріальна	49,94707449	36,4001166
5	12 Квітня (Московський проспект)	49,9487903	36,39489018
6	Верстатобудівна	49,95078493	36,38920782
7	Індустріальний проспект (вул. Оскольська)	49,95151171	36,38660936
8	Станція Лосєве	49,95231624	36,38406807
9	Метро Тракторний завод (Московський проспект)	49,95402224	36,37905752
10	Северина Потоцького (вул. Лосівська)	49,95578045	36,37363846
11	Свистуна	49,95788895	36,36718024
12	Метро імені А,С, Масельського	49,95911959	36,36348492
13	Автогенна	49,96127779	36,35684358
14	Немишлянська	49,97361698	36,34638267
15	Краснодарська (проспект Тракторобудівників)	49,97721337	36,344526
16	Кронштадтська	49,98437977	36,34143186
17	Салтівське шосе (проспект Тракторобудівників)	49,98789035	36,34058152
18	Парк Перемоги	49,99307567	36,34050668
19	проспект Ювілейний (проспект Тракторобудівників)	49,99613558	36,34048183
20	Василя Стуса (проспект Тракторобудівників)	50,00020818	36,34075328
21	**Владислава Зубенка (проспект Тракторобудівників)**	**50,00262196**	**36,34182299**
22	Сади	50,00692366	36,34604086
23	Валентинівська (проспект Тракторобудівників)	50,01078965	36,35042033
24	606-й мікрорайон	50,01551692	36,35646128
25	Регіональний центр послуг	50,01821837	36,35947007
26	Володимирська церква	50,0197951	36,3602685
27	531-й мікрорайон	50,0208472	36,35644773
28	Гвардійців-Широнінців (вул. Героїв Праці)	50,0211921	36,34974452
29	Салтівський ринок	50,0222353	36,34207639
30	Метро Героїв Праці	50,02470016	36,33471569
31	Піски	50,02594111	36,33135947
32	Барабашова	50,02787658	36,3250671
33	Сосновий бір	50,02922209	36,31673173
34	Жилярді (вул. Саперна)	50,02947888	36,3033365
35	Саперна	50,02916482	36,3020328
36	Борткевича	50,02400761	36,29731034
37	Журавлівський гідропарк	50,01926108	36,29305005
38	Тахіатаська	50,01380651	36,28930835
39	Луганська	50,01089641	36,28561356

40	Житлово-комунальний технікум	50,00741124	36,28022521
41	Кольцовська	50,00365854	36,2738811
42	Метро Київська	50,00263894	36,27129176

продовження табл. 3.5

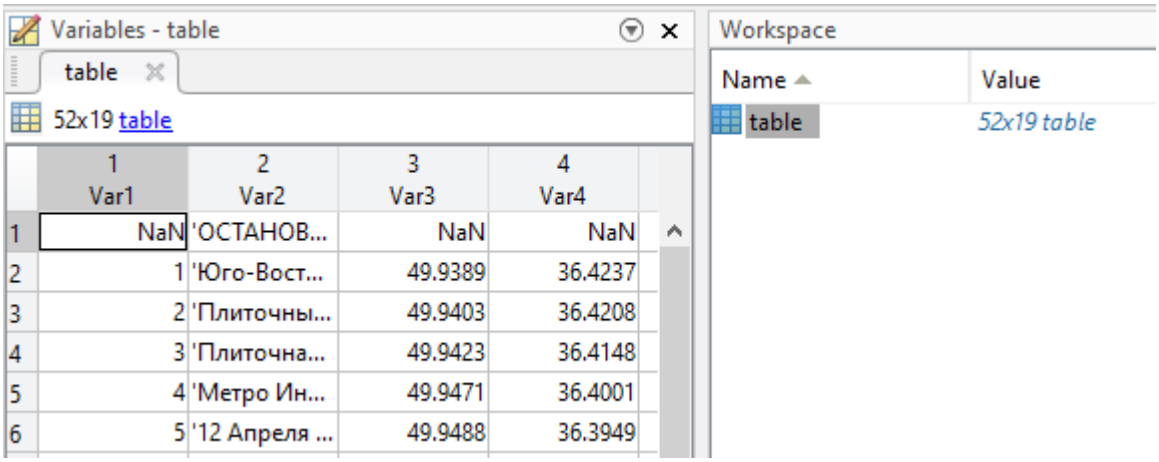
43	Челюскінців	50,00423084	36,26331983
44	Пушкінська	50,01170648	36,25449505
45	Мироносицька	50,01377428	36,24780694
46	Центральний парк культури та відпочинку	50,01663879	36,24902197
47	Дитяча залізниця	50,02069417	36,25274317
48	Сокольники	50,026579	36,25879316
49	Велоцентр	50,03177635	36,26379112
50	Меморіал Слави	50,03177635	36,26379112
51	Лісопарк	50,04032926	36,27212566

Ці данні були збережені у Exel-файлі «Coordinates.xlsx». А блок Matlab-функції в Simulink-моделі виконує візуалізацію маршруту, що розглядається (рис. 3.27).

```
table = readtable('Coordinates.xlsx');
geoplot(table.Var3(2:end),table.Var4(2:end),'r-o');
geobasemap Satellite;
```

Рисунок 3.27 – Код Matlab-функції

Так за допомогою функції readtable координати були завантажені до робочого простіру Matlab в форматі таблиці table (рис. 3.28).



	1 Var1	2 Var2	3 Var3	4 Var4
1	NaN	'ОСТАНОВ...	NaN	NaN
2	1	'Юго-Вост...	49.9389	36.4237
3	2	'Плиточны...	49.9403	36.4208
4	3	'Плиточна...	49.9423	36.4148
5	4	'Метро Ин...	49.9471	36.4001
6	5	'12 Апреля ...	49.9488	36.3949

Рисунок 3.28 – Фрагмент таблиці координат у Matlab

Для графічного відображення маршруту в системі Matlab є відповідні функції geoplot та geobasemap. Параметрами функції geoplot є таблиця з координатами, а саме стовпчик Var2 (широта) та Var3 (довгота). Парметр 'r-o' означає колір лінія на мапі (r – red) та вигляд маркерів (круглий маркер).

Змінити базову карту можна за допомогою функції geobasemap. Параметром функції geobasemap є Satellite, що відповідає супутниковому зображеню високої роздільної здатності (рис. 3.29).

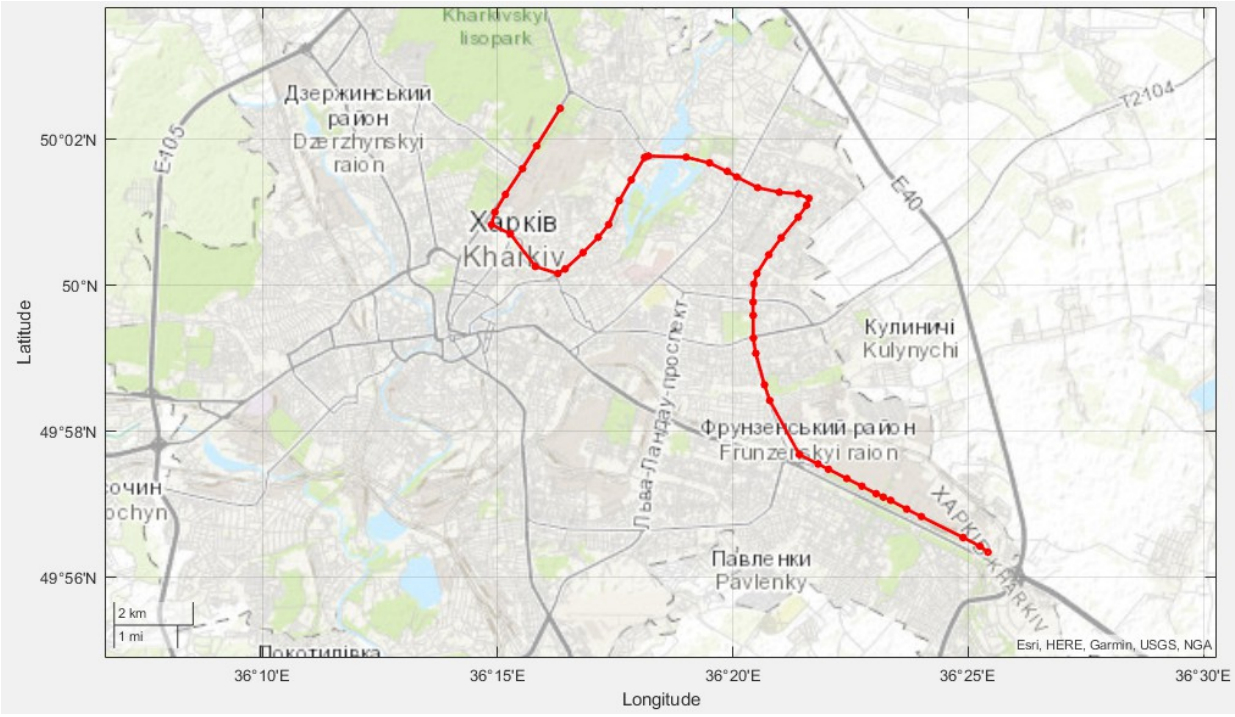


Рисунок 3.29 – Візуалізація маршруту руху трамвая № 26

При моделюванні запропонованної системи в Simulink шляхом задання конкретних координат зупинки міського транспорту (рис. 3.30), де знаходиться пасажир, можна отримати графік, що відображає час прибуття (рис. 3.31).

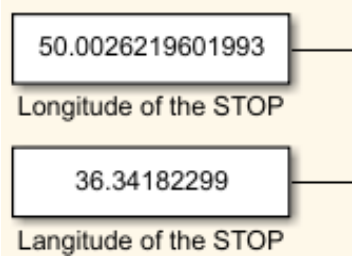


Рисунок 3.30 – Координати зупинки, де встановлено модель СІП

Якщо трамвай рухається з середньою швидкістю 15км/год, то в блоці Display ми бачимо оновлення данних про час прибуття трамваю (рис. 3.31).

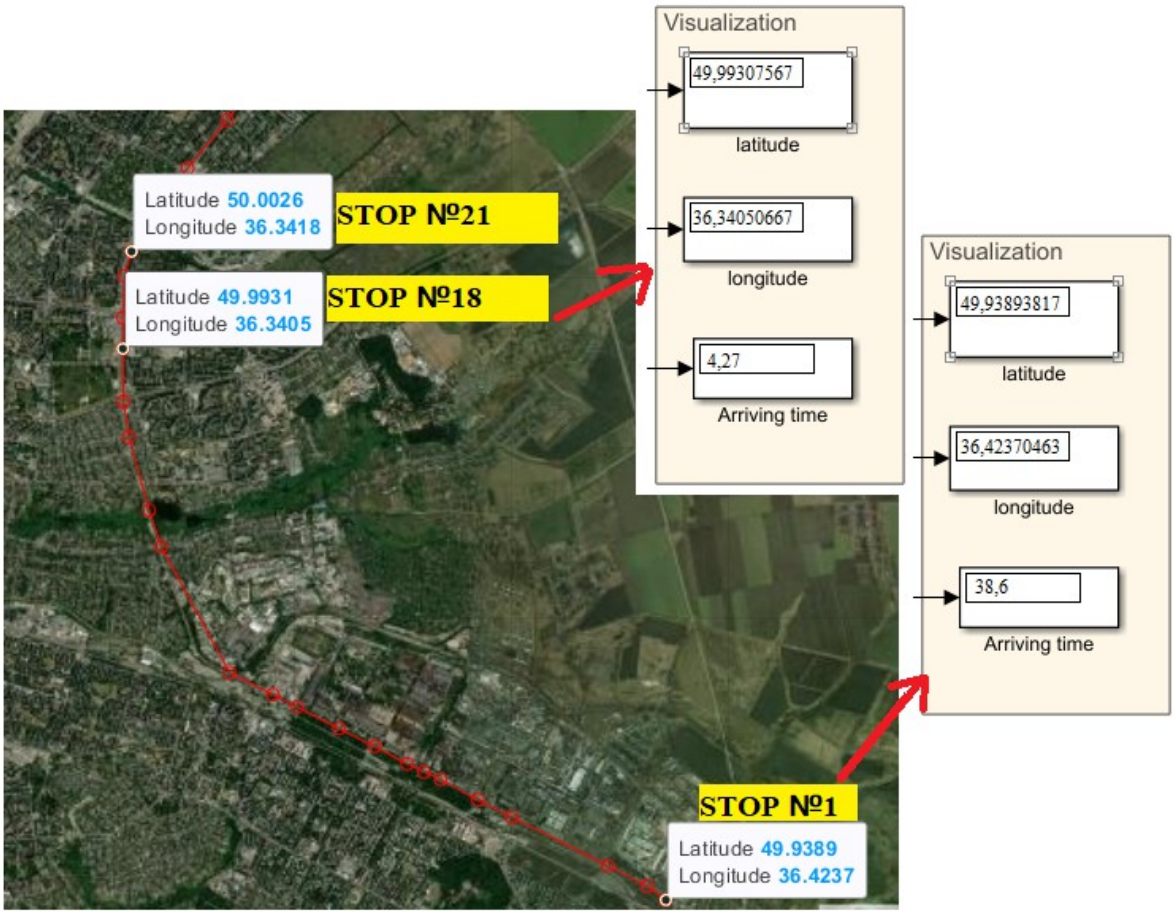


Рисунок 3.31 – Візуалізація данних про час прибуття транспорту

Так наприклад, якщо пасажир знаходиться на зупинці № 21 «Владислава Зубенка», координати якої [50.00262196; 36.34182299], а трамвай рухається з кінцевої зупинки № 1, координати якої [49.9389381788168; 36.4237046339693], то час його прибуття 38,6 хвилини, а якщо трамвай знаходиться вже на зупинці № 18, координати якої [49.99307567; 36.34050668] – то час його прибуття лише 4,27 хвилин. Графік, побудований в результаті моделювання, в системі Simulink підтверджує загальну тенденцію (рис. 3.32).

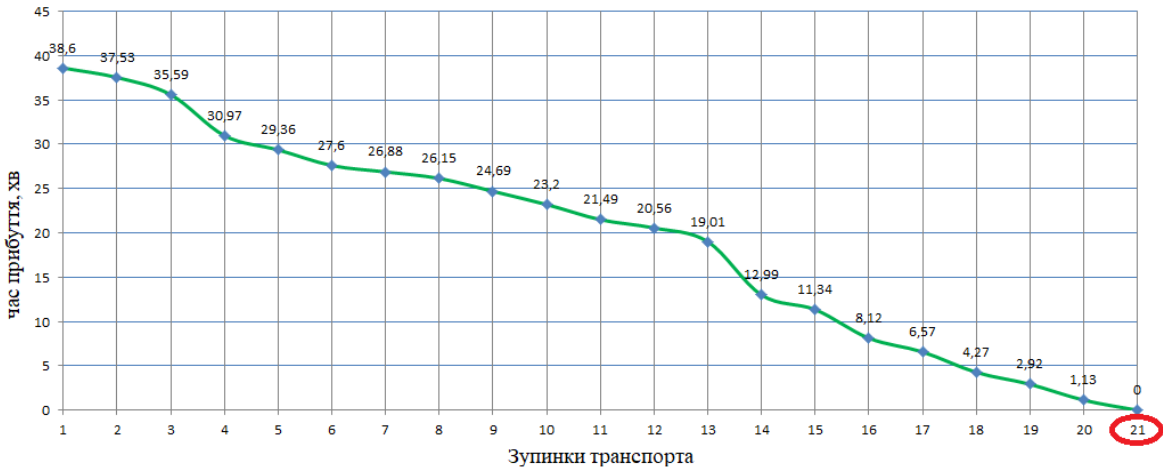


Рисунок 3.32 – Залежність часу прибуття трамвая від його відстані від зупинки № 21

Моделювання роботи системи, створенної у Simulink також підтверджує працездатність моделі.

ВИСНОВКИ

Одним з основних критеріїв якості роботи громадського транспорту з погляду пасажирів є передбачуваність часу поїздки будь-яким маршрутом. Система інформування пасажирів призначена для підвищення якості транспортного обслуговування населення на основі оперативного інформування пасажирів про роботу громадського транспорту.

Для цього встановлюється табло на зупинках громадського транспорту, де дублюється основна інформація. Так пасажири зможуть знати номери маршрутів на цьому напрямку та час прибуття автобуса, трамвая чи тролейбуса. Це дуже зручно, адже у разі аварії на лінії можна викликати таксі чи навпаки – виїхати на громадському транспорті пізно вночі або рано вранці.

Виходячи з актуальних на сьогоднішній день тенденцій (розширення функціональності і зменшення розмірів), щодо розроблюваного пристрою було пред'явлено наступні основні вимоги: забезпечення роботи на базі мікроконтролера, невеликі габарити.

Дослідження існуючих систем показало доцільність використання методів модельно-орієнтованого проектування, як процесу постійної верифікації та валідації при розробці систем. Систематичне використання моделей дозволяє на різних етапах життєвого циклу проекту проводити та повторювати випробування, щоразу переконуючись, що система все ще функціонує відповідно до закладених вимог. Застосування підходу МОП забезпечує підвищення надійності створюваних програмних, апаратних та програмно-апаратних систем.

Модель запропонованої СІП базується на мікроконтролері Arduino, а значить може бути перепрограмована в залежності від локації та потреб тієї чи іншої зупинки громадського транспорту. Принцип роботи системи полягає у тому, що на борт громадського транспорту встановлюється

обладнання з GPS-приймачем. Цей приймач оброблює сигнали зі супутникових навігаційних систем. Данні про координати транспорту передаються до блока управління, який обчислює час його прибуття. Ця інформація виводиться на екран, що встановлений на зупинці громадського транспорту. Для розробки, налагодження і тестування програмного забезпечення системи інформування пасажирів громадського транспорту застосовують такі засоби, як програмні та фізичні імітатори сигналів та інтерфейсів.

Основою для модельно-орієнтованого проектування було обрано пакет прикладних програм MATLAB та його Simulink-додаток, призначений для візуального моделювання динамічних систем.

Наукова новизна визначається використанням підходу модельно-орієнтованого проектування СІП громадського транспорту.

Це дає змогу перевірити працездатність системи не на реальному об’єкті, а імітувати навколишнє середовище системи. Замість фізичних прототипів та текстових специфікацій у модельно-орієнтованому проектуванні застосовується модель, що виконується. Ця модель використовується у всіх етапах розробки. При такому підході можна розробляти і проводити імітаційне моделювання як усієї системи, так і її компонентів. Є можливість автоматичної генерації коду, випробувань у безперервному режимі та верифікації. d

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Jones D. Model-Based Design of control systems: Simulate and test before committing to hardware / Doug Jones, Brian McKay // Industrial embedded systems. – Режим доступу: http://industrial.embedded-computing.com/articles/model-based-design-control-systems-simulate-test-committing-hardware/. – Дата доступу: 24.12.2020. – Загол. з екр.
2. Esfeh Ansari M. Waiting time and headway modelling for urban transit systems – a critical review and proposed approach / Mohammad Ansari Esfeh, S. C. Wirasinghe, Saeid Saidi, Lina Kattan // Transport Reviews. – 2020. – P. 1-23.
3. Harold, C. M. Identifying the information needs of users in public transport/ C. M. S. Harold, H. Kramer//Advances in Human Aspects of Roadand Rail Transportation. − CRC Press. − 2012, P. – 31-34.
4. Ganesh K. Implementation of a real time passenger information system / K. Ganesh, M. Thrivikraman, J. Kuri, H. Dagale, G. Sudhakar, S. Sugata // arXiv preprint arXiv: 1206.0447. – 2012. – URL: https://arxiv.org/abs/1206.0447 .− Access date: 17.12.20. − Screen title.
5. Lavanya R. A Smart Information System for Public Transportation Using IoT / R. Lavanya, K. Sheela Sobana Rani, R. Gayathri, D. Binu // International Journal of Recent Trends in Engineering & Research (IJRTER). – Vol. 03, issue № 4. – April 2017. – p. 222 – 230.
6. Alshamsi H. Real time vehicle tracking using Arduino Mega / Humaid Alshamsi, Veton Këpuska, Hazza Alshamsi// International Journal of Science and Technology (IJST). – Vol.5, No.12. − December, 2016.− P. 624 - 626.
7. Shelke P. Smart tracking system for school buses for ensuring child security using IoT implications and GPS technology / Palvi Shelke, Pravin Dere // International Research Journal of Engineering and Technology (IRJET). – Vol. 6, issue 10. – October 2019. – P. 1235 – 1238.
8. Mohd I. Development of Arduino Based Real Time Bus Tracking and

Monitoring System International / Izzeldin I. Mohd, Chong Yew Kent1, Nazar Elfadil // Journal of Computer Trends and Technology (IJCTT). – Vol. 68, Issue 4. – April 2020. – P. 100-106.

9. Журавлев С. С. Применение модельно-ориентированного проектирования к созданию АСУ ТП опасных промышленных объектов / С. С. Журавлев, С. В. Рудометов, В. В. Окольнишников, С. Р. Шакиров // Вестн. НГУ. Серия: Информационные технологии. − 2018. − Т. 16, № 4.− С. 56-67.

10. Топораш Г. К. Модельно-ориентированное проектирование программного обеспечения для встраиваемых систем в среде Matlab/Simulink / Г. К. Топораш, А. В. Мазур, Д. А. Ковальчук, А. А. Пушкин // Автоматизація технологічних і бізнес-процесів. – 2014. – № 17. – С. 26-29.

11. Dauni P. Implementation of Haversine formula for school location tracking / P. Dauni1, M. D. Firdaus, R. Asfariani, M. Saputra, A. Hidayat, W. Zulfikar // Journal of Physics: Conference Series. – Vol. 1402, Issue 7. – 2019. – P. 1-6. – URL: https://iopscience.iop.org/article/10.1088/1742-6596/1402/7/077028/pdf. − Access date: 17.12.20. − Screen title.

12. Maureira M.A.G. ThingSpeak – an API and Web Service for the Internet of Things / Marcello A. Gómez Maureira, Daan Oldenhof,, Livia Teernstra // Leiden University's computer science institute (LIACS). – 2014. – URL: https://mediatechnology.leiden.edu/images/uploads/docs/wt2014_thingspeak.pdf. − Access date: 17.12.20. − Screen title.

13. NMEA 0183. Standard For Interfacing Marine Electronic Devices. – Intro. 2002-01-01. – Maryland, USA: National Marine Electronics Association, 2001. – 88 с.

14. Bayle J. C Programming for Arduino / Julien Bayle // Packt publishing. – 2013. – P. 478.

15. Тимохин, А.Н. Моделирование систем управления с применением Matlab: учебное пособие / А.Н. Тимохин, Ю.Д. Румянцев. – М.: НИЦ ИНФРА-М, 2016. – 256 с.