Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Розробка інформаційної системи GPS-контролю перевезень багажу

Виконав: студент IV курсу, групи СНс-42
спеціальності 122 Комп’ютерні науки
(шифр і назва спеціальності)

(підпис) Церковний В.О
(прізвище та ініціали)

Керівник (підпис) Млинко Б.Б
(прізвище та ініціали)

Нормоконтроль (підпис) Шимчук Г.В.
(прізвище та ініціали)

Завідувач кафедри (підпис) Боднарчук І.О.
(прізвище та ініціали)

Рецензент (підпис) (прізвище та ініціали)

Тернопіль
2021

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«__» _____ 2021 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

Студенту Церковному Владиславу Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка інформаційної системи GPS-контролю перевезень багажу

Керівник роботи Млинко Богдана Богданівна, к.т.н, доцент кафедри КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «2» березня 2021 року № 4/7-171

2. Термін подання студентом завершеної роботи

3. Вихідні дані до роботи вимоги замовника, літературні та інтернет джерела.

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. Розділ 1. Постановка задачі на розробку інформаційної системи. 1.1 Аналітичний огляд відомих рішень. 1.2 Обґрунтування актуальності теми та постановка задачі на розробку.1.3 Висновок до першого розділу. Розділ 2. Структуризація та реалізація технічного проєкту. 2.1 Розробка структурної схеми 2.2 Вибір та обґрунтування вибору елементної бази. 2.3 Розробка функціональної схеми пристрою. 2.4 Розробка алгоритму та написання текстів програми. 2.5 Розробка інструкції з експлуатації електричного пристрою. 2.6 Пошук, виявлення та виправлення несправностей системи. 2.7 Висновок до другого розділу. Розділ 3. Безпека життєдіяльності, основи охорони праці. 3.1 Характеристика життєдіяльності людини у системі "людина – машина – середовище існування". 3.2 Безпечність при виконанні вантажопідйомних робіт та транспортуванні вантажів. 3.3 Особливості безпеки праці під час вантажно-розвантажувальних робіт. 3.4 Висновок до третього розділу. Висновки. Перелік джерел. Додатки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1 Тема роботи; 2 Порівняльна характеристика відомих рішень та розробленого застосунку; 3 Головні недоліки існуючих рішень; 4 Актуальність дослідження, мета дослідження, задачі дослідження; 5 Елементна база; 6 Структурна схема; 7 Функціональна схема; 8 Алгорим роботи системи; 9 Реалізація карти; 10 Розробка маркера; 11 Візуалізація «Google map»; 12 Візуалізація в «Google Earth»; 13 Виявлення та виправлення несправностей; 14 Висновки

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи хорони праці	Гурик О.Я., доцент кафедри МТ		

7. Дата видачі завдання 25 січня 2021 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	26.01.2021	*Виконано*
2.	Підбір джерел про розробку інформаційної системи	08.02.2021-	*Виконано*
	GPS-контролю перевезень багажу	09.02.2021	
3.	Переклад та опрацювання джерел про взаємодію еле-	10.02.2021-	*Виконано*
	ментів, що використовуватимуться в пристрої	11.03.2021	
4.	Виконання дослідження щодо реалізації технічного проєкту.	12.03.2021-16.04.2021	*Виконано*
	Розроблення структурної та функціональної схеми, та		
	розробка алгоритму роботи системи		
5.	Оформлення розділу «Постановка задачі на розробку	17.04.2021-	*Виконано*
	інформаційної системи»	06.06.2021	
6.	Оформлення розділу «Структуризація та реалізація	07.06.2021	*Виконано*
	технічного проекту»		
7.	Виконання завдання до підрозділу «Безпека життєдіяльності»	08.06.2021	*Виконано*
8.	Виконання завдання до підрозділу «Основи охорони	09.06.2021	*Виконано*
	праці»		
9.	Оформлення кваліфікаційної роботи	19.06.2021	*Виконано*
10.	Нормоконтроль		*Виконано*
11.	Перевірка на плагіат	18.06.2021	*Виконано*
12.	Попередній захист кваліфікаційної роботи		*Виконано*
13.	Захист кваліфікаційної роботи	25.06.2021	*Виконано*

Студент _____ Церковний В.О
(підпис) (прізвище та ініціали)

Керівник роботи _____ Млинко Б.Б
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розробка інформаційної системи GPS-контролю перевезень багажу // Кваліфікаційна робота освітнього рівня «Бакалавр» // Церковний Владислав Олександрович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СНс-42 // Тернопіль, 2021 // С. , рис. – 25, табл. – 1, кресл. – 0 , додат. – 4, бібліогр. – 0 .

Ключові слова: мікроконтролер, SD mod, Vibration sensor, Temperature sensor, Humidity sensor, RAM, ROM

Кваліфікаційна робота присвячена розробці інформаційно системи GPS-контролю перевезення багажу. Розроблена інформаційна система дозволить користувачу вирішувати різнотипні проблеми при перевезенні багажів.

В першому розділі кваліфікаційної роботи: розглянуто відомі рішення, наведено їх порівняльну характеристику, обґрунтовано актуальність розробки інформаційної систими GPS – контролю перевезень багажу.

В другому розділі кваліфікаційної роботи: розроблено структурну схему системи, розроблено функціональну схему пристрою та алгоритм, вибрано програмне забезпечення та написано тексти програми, розроблено інструкцію з експлуатації електричного пристрою, розроблено алгоритм дій при виявленні несправностей.

ANNOTATION

Development of information system of GPS-control of luggage transportation // Qualification work of educational level "Bachelor" // Tserkovny Vladislav Alexandrovich // Ternopil National Technical University named after Ivan Pulyuy, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science, group SNs-42 // Ternopil, 2021 // C., fig. – 25, table. – 1, chair. – 0, added. – 5, bibliogr. – 0.

Keywords: microcontroller, SD mod, Vibration sensor, Temperature sensor, Humidity sensor, RAM, ROM.

Qualification work is devoted to the development of information system GPS-control of luggage transportation. The developed information system will allow the user to solve various types of problems when transporting luggage.

In the first section of the qualification work: the known solutions are considered, their comparative characteristics are given, the urgency of development of information system GPS - control of luggage transportation is substantiated.

In the second section of the qualification work: the structural scheme of the system is developed, the functional scheme of the device and algorithm are developed, the software is selected and the program texts are written, the instruction on operation of the electric device is developed, the algorithm of actions at detection of malfunctions is developed.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

МК – мікроконтролер

GPS – система глобального позиціонування

SD mod – модуль карти пам’яті

Vibration sensor – датчик вібрації

Temperature sensor – датчик температури

Humidity sensor – датчик вологості

Log – файл у якому накопичується інформація

Visualizer – середовище візуалізації шляху

OSM – “open street map” карта

PIN – вивід на платі мікроконтролера

RAM – оперативна пам’ять

ROM – сховище даних

GND – заземлення

VCC – живлення

D (0,1…n) – цифрові виводи

A (0,1…n) – аналогові виводи

UART – тип асинхронного приймача-передавача

Host – вузл мережі

Lat – широта (latitude)

Lon – довгота (longitude)

SCK – тактові імпульси (SerialClock)

Delay – затримка обміну даними

АЦП – аналогово-цифровий перетворювач

ЦАП – цифро-аналоговий перетворювач

ЗМІСТ

ВСТУП

Внаслідок розповсюдження замовлень різного роду товару функціями доставки користується велика кількість організацій, компаній та простих користувачів. Доставка товару має досить великий попит на ринку. Завдяки компаніям, які надають функції експрес–доставки, люди замовляють товар кожен день.

За допомогою функцій доставки стає можливість переправлення замовлень на досить великі дистанції, з міста в місто, з одної області в іншу та з одної країни в іншу. Перевезення товару може відбуватись за допомогою автомобільного транспорту, повітряного та за допомогою пароплавів.

Інколи при доставці товару на руки користувачу немає можливості повної перевірки на цілісність замовленого об’єкта та отримавши тільки через деякий час стає замітно подряпини, збої в роботі пристроїв, або замовлення взагалі приїде розбитим, розваленим. Наприклад при перевезенні скляних конструкцій багаж прибуває на свою точку побитим або пошкодженим. При отриманні комп’ютерної техніки, мобільної та інших, можуть спостерігатись збої в роботі цих пристроїв. Також ця система може добре слугувати для перевезення медикаментів.

Для запобігань даних проблем можна використати GPS систему контролю за багажем, яка дасть змогу перевірки цілісності товару при його отриманні. Дана система дає змогу отримувати свій багаж в цілісності або при наявності його пошкодження не приймати його та не оплачувати.

Метою даної кваліфікаційної роботи освітнього рівня «Бакалавр» є:

– провести аналіз відомих рішень та навести їх порівняльну характеристику;

– розробити структурні та функціональні схеми;

– вибрати елементну базу;

– реалізувати автономність пристрою;

– забезпечити онлайн та офлайн моніторинг умов перевезення;

– організувати використання резервного сховища даних;

– забезпечити автономність пристрою.

Практичне значення одержаних результатів.

Розроблений пристрій має практичне використання, та надає користувачу такі переваги: онлайн та офлайн моніторинг умов перевезення, резервне сховище інформації, перезаряджаємий акумулятор.

РОЗДІЛ 1. ПОСТАНОВКА ЗАДАЧІ НА РОЗРОБКУ ІНФОРМАЦІЙНОЇ СИСТЕМИ

1.1 Аналітичний огляд відомих рішень

Сучасна технологія GPS стала невід'ємною частиною сучасного життя. GPS трекинг - це процес, який лежить в основі будь-якої системи відстеження транспортних засобів.

GPS-трекинг належить до глобальної системи позиціонування. Сюди входить мережа з багатьох супутників на орбіті та пристроїв на Землі, які можуть з точністю знаходити людину чи об'єкт на Землі. GPS-трекинг відстежує три окремі набори даних: навігацію, позиціонування та хронометраж.

Ця технологія існує вже давно, спочатку вона була створена для військового використання в 1960-х. У 1983 році GPS став доступним для всіх і з тих пір технологія лише зростала. Сьогодні він використовується для всього, починаючи від точних військових маневрів у зарубіжних країнах і закінчуючи дітьми, які грають у мобільні ігри у вашому районі.

Як працює GPS-відстеження?

GPS вимагає використання багатьох супутників, які обертаються навколо Землі. Ці супутники постійно передають своє місце розташування та статус над нами. Це постійно контролюється «GPS Master Control Station», а також іншими станціями відстеження та моніторингу тут, на землі, для забезпечення точності та належної роботи. Головний контрольний пункт також відповідає за технічне обслуговування та виправлення, якщо щось піде не так [31].

Пристрій GPS на Землі отримує ці сигнали, інтерпретуючи унікальні дані кожної людини. Склавши карту розташування чотирьох і більше супутників щодо пристрою відстеження, він може тріангулювати своє точне положення в тривимірному просторі. Більше супутників часто використовують для перевірки даних та забезпечення більш точних показань місцезнаходження.

Можливо, ви знайомі з деякими найпоширенішими способами використання технології GPS, але є й інші, яких ви, можливо, ніколи не розглядали. GPS є важливою частиною всіх видів операцій, від військових до оперативних, комерційних та особистих потреб.

Військове використання GPS

GPS почав працювати в збройних силах більше 50 років тому і військові продовжують використовувати його для відстеження літаків, пересування військ, навігації на морі і багато чого іншого. GPS-навігація в військах особливо важлива для тих, хто перебуває на незнайомій території або рухається вночі. Зовсім недавно нова технологія GPS дозволила військовим використовувати БПЛА (безпілотні літальні апарати). Ці БПЛА (іноді звані дронами) є технологією, яка рятує життя, тому що вони дозволяють нам бачити і управляти деякими з найнебезпечніших місць на Землі, не завдаючи шкоди нашим військовим і жінкам. Вони можуть управлятися дистанційно і часто використовуються для розвідки, спостереження і розвідки.

GPS-стеження грає важливу роль в пошуково-рятувальних операціях, дозволяючи рятувальникам відстежувати місця, які вони покрили, бачити загальну картину пошукової операції або навіть отримувати дані GPS безпосередньо з пристрою GPS або телефону загубився людини.

Коли пошуково-рятувальна операція шукає тих, хто вижив після великого лиха або намагається знайти зниклого безвісти в слаборозвиненою місцевості, вони використовують мережеву систему. Це гарантує ретельний пошук кожної області, і жодна область не проводитиметься двічі за рахунок інших місць. Багато років тому пошуково-рятувальні роботи проводилися олівцем і папером. Цей метод іноді може бути неточним і збивати з пантелику добровольців. Сьогодні пошуково-рятувальні групи часто оснащуються GPS-трекера, щоб забезпечити більш точну схему пошуку.

GPS відстеження транспортних засобів

GPS-відстеження має ряд комерційних цілей, але, мабуть, одним із найпотужніших є розгортання GPS-пристроїв для відстеження комерційних

флотів. За допомогою GPS на кожному транспортному засобі компанії автопарку можуть відстежувати точне місцезнаходження та стан водія, отримувати потужну інформацію про ефективність руху автопарку та надавати негайну допомогу на дорозі, якщо це необхідно.

GPS є життєво важливою частиною сучасних систем відстеження автопарку для відстеження активності та розташування автомобілів, підвищення безпеки та ефективності. Хоча деякі були стурбовані тим, що ця технологія буде функціонувати як "старший брат" і спричинятиме трудові суперечки з водіями автопарку, вона виявилася гідним включенням у будь-які операції флоту.

Крім того, простота та точність відправлення та маршрутизації продемонстрували, що системи GPS у транспортних засобах зменшують аварії на 38% для малого бізнесу. Це означає безпечніші дороги для всіх та кращу репутацію парку для вашого бренду.

Серед відомих GPS-систем ринку можна виділити такі [3]:

- *BI 310 CICADA* – система призначена для GPS – моніторнигу і контролю за рухом об’єкту по координатам GPS або LBS. Пристрій включає в себе перезаряджаєму батарею об’ємом 3200 мА – відправлення приблизно 1400 позицій на одному заряді. Пристрій є повністю автономним і не вимагає монтажу. Внутрішній акумулятор дозволяє працювати до 4 років при відправці даних 1 раз на добу. За рахунок маленьких розмірів є безліч варіантів для прихованого розміщення в автомобілі або вантажівці. Чи не піддається виявленню діагностичними приладами [4]. Підтримка FOTA (firmware-over-the-air) – можливість віддаленої зміни налаштувань і прошивки обладнання при черговому сеансі зв'язку з сервером. Має відкритий протокол передачі даних і сумісність з усім популярним ПО. Візуальне зображення даного пристрою зображене на рисунку 1.1.

Рисунок 1.1 – Візуальне зображення BI 310 CICADA

- *Lookou*t – система контролю за станом вантажу. Ця система включає в себе:

– датчик температури. Який працює в діапазоні температур від -55 до +125 градусів цельсія, вага - 40 грам, час максимальної роботи до відказу 25 920 годин, відображається звіт роботи датчика;

– датчик вологості. Який використовується для вимірювання вологості, працює в діапазоні вологостівід 0% до 100%, час максимальної роботи датчика 18 340 годин.

- *Triton* – система, для перегляду за переміщеням контейнерів з багажем в реальному часі [5]. Дана система володіє такими характеристиками:

– автономний пристрій (робота від перезаряджаємих акумуляторів);

– можливість отримання інформації про відкриття та закриття дверей, збільшення або зниження температури, підвищення або пониження вологості, включення або вимикання світла, рівень заряду акумуляторної батареї, статус місце положення;

– контроль за місцем знаходження об’єкту, за допомогою відображення інформацію у вигляді SMS повідомлень;

– влаштований акселерометр;

– коли немає покриття і вся інформація з датчиків зберігається у пам’яті пристрою. Система пристрою Triton зображена на рисунку 1.2.

Рисунок 1.2 – Зображення пристрою TRITON

Проаналізувавши відомі рішення, наводжу прорівняльну характеристику у таблиці 1.1.

Таблиця 1.1 – Порівняльна характеристика відомих рішень

	Triton	BI 310 CICADA	Lookout
Датчик температури	+	-	+
Датчик вологості	-	-	+
Датчик вібрації	-	-	-
Автономне живлення	+	+	-
Доступ до мережі	-	+	-
Онлайн перепрошивка	-	+	-
Резервне сховище	+	-	-
Онлайн моніторинг	-	-	-

1.2 Обґрунтування актуальності теми та постановка задачі на розробку

Оскільки відомі рішення володіють такими недоліками як: відсутність потрібних давачів для моніторингу умов перевезення, онлайн та офлайн перегляд шляху, онлайн перепрошивки, резервного сховища даних, розробка є актуальною.

В пристрої буде реалізований моніторниг умов перевезення багажу, таких як: перегляд температури, вологості та вібрації в реальному часі. Оскільки при перевезенні скляних конструкцій багаж часто прибуває на свою точку побитим або пошкодженим. Також при отриманні комп'ютерної техніки, мобільної та інших, можуть спостерігатись збої в роботі цих пристроїв. Ця система буде добре слугувати для перевезення медикаментів, вона буде незамінною при використанні у фірмах або підприємствах які виготовляють таке обладнання як: медичне, скляне, телефони, сервізи, телевізори, та іншого роду крихку техніку. Все це можна використовувати на таких видах транспорту як автомобілі, пароплави та на повітряний транспорт.

За допомогою пристрою, який буде встановлений біля вашого багажу, перегляд якості перевезення багажу та контроль за якістю виконання роботи персоналу стане набагато простішим. Керівництво матиме змогу визначати за якої причини товар прибув у не належному стані.

Розробка є корисною для людей, які замовляють товар різного роду з інтернету, за допомогою таких компаній як «Нова Пошта», «Укр Пошта» та інших. За рахунок відсутності аналогів, система знайде своє місце на ринку.

Для досягнення мети системи контролю перевезень контролю багажів, необхідно розв'язати наступні задачі:

- вибрати елементну базу;
- розробити структурну та функціональну схеми;
- реалізувати автономність пристрою;
- забезпечити онлайн та офлайн моніторинг умов перевезення;

– організувати використання резервного сховища даних;
– забезпечити автономність пристрою.

1.3 Висновок до першого розділу

Розглянуто переваги та недоліки відомих рішень, наведено їх порівняльну характеристику, обґрунтовано актуальність розробки інформаційної системи GPS – контролю перевезень багажу. Поставлено ряд задач, які потребують вирішення для досягнення мети.

РОЗДІЛ 2. СТРУКТУРИЗАЦІЯ ТА РЕАЛІЗАЦІЯ ТЕХНІЧНОГО ПРОЄКТУ

2.1 Розробка структурної схеми

Розроблення структурної схеми виконано на основі аналізу та поставлених задач. Оскільки темою кваліфікаційної роботи є розробка інформаційної системи GPS контролю за багажем, система повинна містити в собі: функції моніторингу шляху вантажу, візуалізацію даних про про температуру, вологість та порушення умов вібрації. Пристрій буде використовувати автономне джерело живлення. Структурну схему пристрою можна побачити на рисунку 2.1.

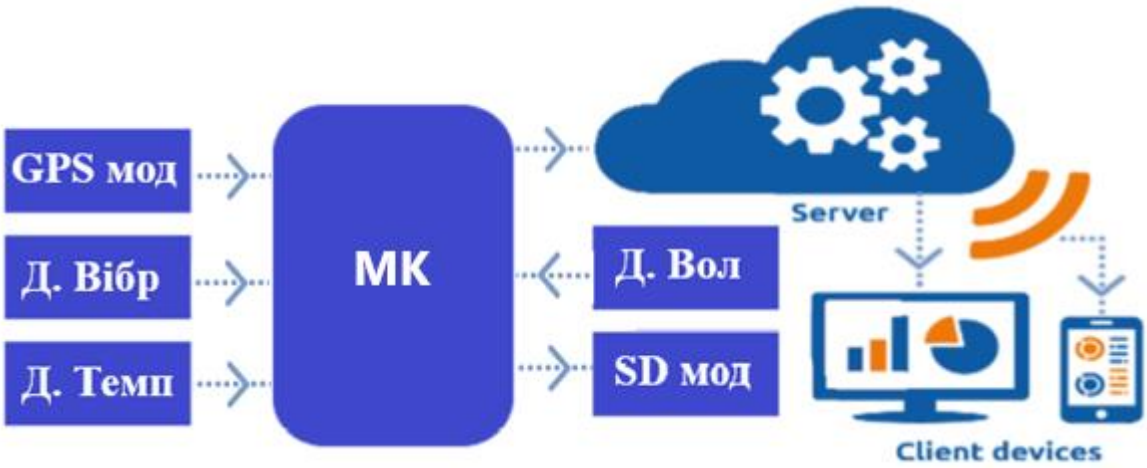


Рисунок 2.1 – Структурна схема пристрою

Пристрій складатиметься з наступних функціональних вузлів:

– мікроконтролер (МК) NodeMcuESP8266 – відповідає за управління між модулями, з'єднання з інтернет мережею, та відправки «get» запитів для візуалізації інформації [6];

– GPS модуль (GPS mod) – відповідає за зв'язок з супутником, що дозволяє визначити положення об'єктів в просторі, відповідно до всесвітньої системи координат (GPS/Голонанс) [7];

– Vibration sensor – інформує про стан вібрації об’єкта [11];

– Temperature sensor – відповідає за визначення темпреатури об’єкта;

– Humidity sensor– – відповідає за визначення вологості об’єкта [8];

– модуль пам’яті (SD mod) – відповідає за збільшення об’єму пам’яті, зберігання GPS координат, та збереження інформації з датчиків [10];

– Visualizer – онлайн утиліта, яка дозволяє візуалізуватиGPS координати.

Пристрій буде взаємодіяти з веб сторінкою, яка здійснюється за допомогою скриптів. За допомогою веб ресурсу, можна побачити карту та маркер в якому відображається інформація про стан об’єкту та його місце знаходження. Також рішення міститиме в собі роботу з інтернет платформами, які відповідають за візуалізацію даних GPS шляху, у вигляді GPS треку. Для резерву данні координат та інформація з датчиків записується на карту пам’яті у вигляді «.txt» файлів [29].

Функціонування пристрою. Пристрій працюватиме на основі двох функцій. Функції першого режиму дозолять переглядати поточне місце знаходження багажу за допомгою веб сайту, за допомогою маркера. В маркері відображатиметься: дата/місяць/рік, година/хвилина/секунда, та інформація з датчиків вібрації, температури та вологості. Все це можна буде переглядати в реальному часі на веб-сторінці, візуалізованій за допомогою карт вільного доступу «Open street map» [22].

Другий режим, представить собою фунції резервного зберігання данних, у файлах формату «txt», у випадку зникнення інтернету або інших виникших проблем. Для того щоб дані не втрачались, вони будуть записуватись у два файла на карту пам’яті [21].

Перший файл «Log» відповідає за збереження GPS координат у форматі дати та часу записаних координат, які потім можна конвертувати в GPS трек за допомогою веб сервісів [23].

Другий файл «Info» відповідатиме за збереження інформації з датчиків, у форматі дати/часу та інформацію з датчиків.

2.2 Вибір та обґрунтування вибору елементної бази

Система буде розроблятися на основі плати NodeMCU.

В якості мікропроцесора в системі управління використовуватиметься плата NodeMCU з мікроконтролером ESP-8266. NodeMCU - за допомогою цієї платформи є можливість скористатися вже вбудованою прошивкою і за допомогою модуля по інтерфейсу SDIO 2.0, SPI, UART та вести передачу даних. Плата з мікроконтролером зображенана рисунку 2.2.

Рисунок 2.2 – Зовнішній вигляд мікроконтролера

В мікроконтролер вбудований інтерпретатор скриптової мови Lua, яким задається поведінка плати [14]. Користувачі у яких стоїть програма-GPS трекера, отримують інформацію про переміщення об’єкта.

Технічні характеристики WiFi модуля ESP8266:

- стандартиWiFi: 802.11 b / g / n;
- вбудований температурний датчик;
- підтримувані типи шифрування: WEP, WPA, WPA2;
- вбудований TCP / IP стек протоколу;
- підтримувані режими роботи: клієнт (STA), точка доступу (AP), клієнт та точка доступу (STA + AP);
- напруга живлення: 1.7 ... 3.6 В;
- вбудований PLL і система управління живленням;

- струм: до 250 мА в залежностівід режиму роботи;
- PCB антена
- доступні GPIO;
- процесор 32-bit з низьким енергоспоживанням;
- споживання в режимі очікування DTIM3 0.86 mA, Deepsleep 10 uA;
- STBC, 1x1 MIMO, 2x1 MIMO;
- час "прокидання" і передачі пакета <2 ms;
- A-MPDU & A-MSDU aggregation;
- зовнішня Flash пам'ять розміром 512 кб. Частина використовується для даних, частина – для коду;
- RAM даних - 80 кб, RAM інструкцій - 32 кб.

Умовне позначення мікроконтролера зображенена рисунку 2.3.

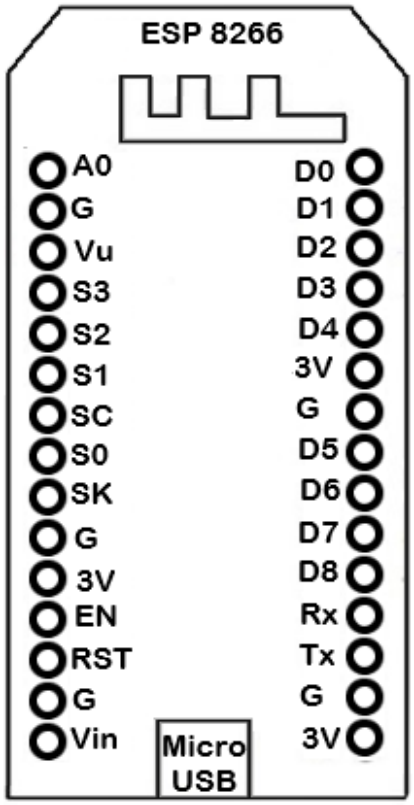


Рисунок 2.3 – Умовне позначення мікроконтролера ESP 8266

На платі оснащений світлодіод який свідчить про готовність пристрою до роботи. Два спалаха світлодіода свідчать про те, що запис розпочато, один

спалах свідчить про те, що запис призупинено. Структурна схема плати зображена рисунку 2.4.

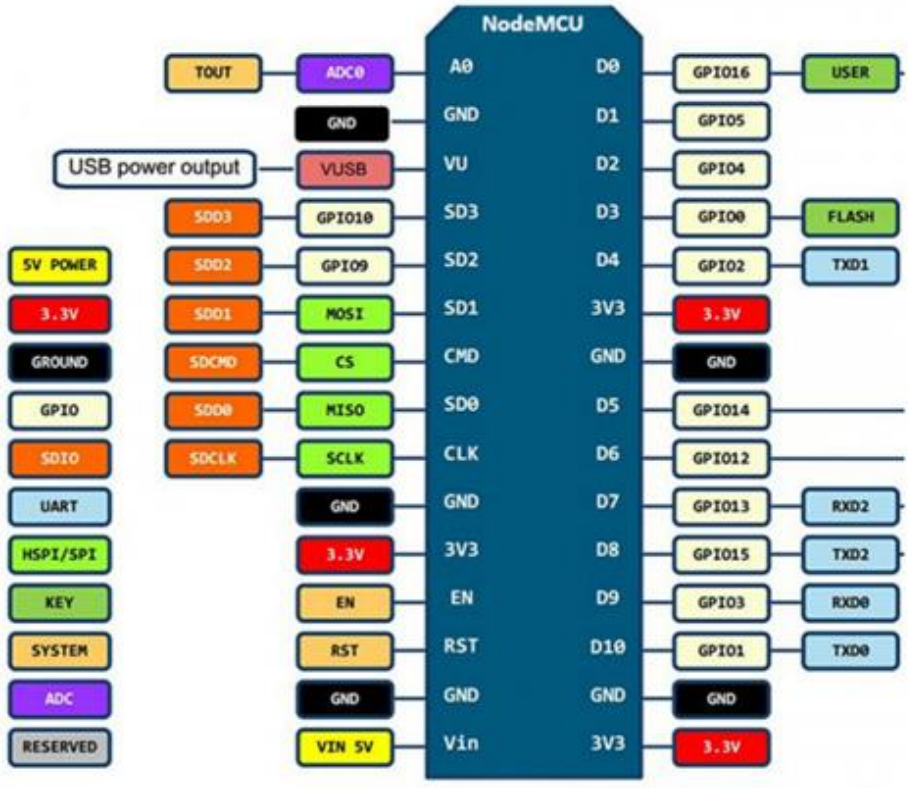


Рисунок 2.4 – Структурна схема плати

Виходи спеціального призначення:

– D1-D10 - виходи з широтно-імпульсною модуляцією;
– D1, D2- виходи для інтерфейсу I²C / TWI;
– D5-D8 - виходи для інтерфейсу SPI;
– D9, D10 - UART;
– A0 - АЦП.

GPS модуль GY-NEO6MV2. В якості GPS модуля вибрано GY-NEO6MV2. GPS модуль GY-NEO6MV2 модуль побудований на базі чіпа UbloxNEO-6M, який є одним з найбільш точних і надійних [2]. Він забезпечує неперевершену точність позиціонування у будь – якому з ваших проектів за допомогою системи глобального позиціонування - GPS. Може бути використаний в

пристроях на Arduino, AVR, PIC, ARM. У комплекті йде антена. Схема підключеня GPS модуля зображенана рисунку 2.5.

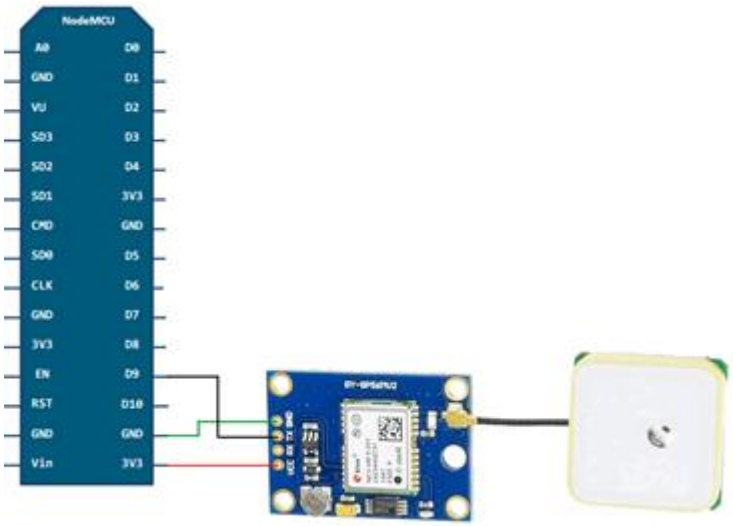


Рисунок 2.5 – Схема підключення GPS модуля

Характеристики:

– сумісність GPS модуля: APM2 (ArduPilotMega 2) і APM2.5 (ArduPilotMega 2.5);

– пам'ять EEPROM для зберігання налаштувань: є, вбудована;

– інтерфейс: RS232 TTL;

– швидкість передачіданих за замовчуванням: 9600 біт / с;

– напруга живлення: 3 - 5 В;

– температурний режим роботи: від -40 до 85 ° С;

– розмір модуля: 37 х 27 мм;

– розмірантени: 27 х 27 мм;

– вага модуля з антеною: близько 16;

Призначення виводів:

– GND – мінус живлення, призначений для з’єднання з виводом GND на платі;

– TX – лінія передачіданих (від модуля), з’єднаний з виводом D9 на платі;

– RX – лінія прийомуданих (модулем);

– VCC – плюс живлення (5В), з’єднаний з виводом 3v3 на платі.

Умовне позначення модуля зображенняна рисунку 2.6.

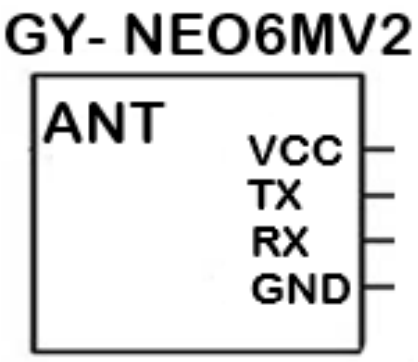


Рисунок 2.6 – Умовне позначення GY-NEO6MV2

Адаптер SD карти. В якості адаптера карти буде використано SD cardreaderarduino [18]. Універсальний адаптер являє собою звичайну плату, на якій можна побачити слот для карти, резистори і регулятор напруги. Зовнішній вигляд модуля SD карти відображений на рисунку 2.7.

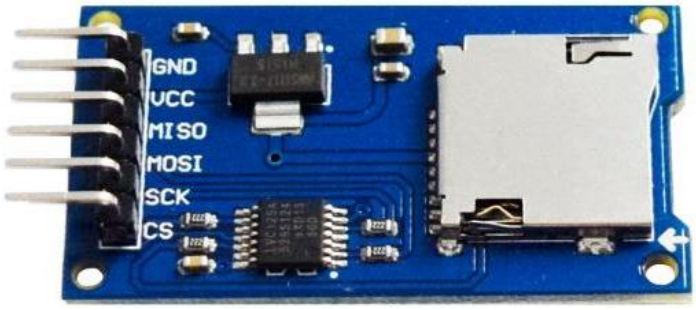


Рисунок 2.7 – Зовнішній вигляд модуля SD карти

Він володіє наступними технічними характеристиками:

− діапазон робочихнапруг 4.5-5В;

− підтримка SD карти до 2 Гб;

− струм 80 мА;

− файлова система FAT 16.

Модуль SD-карти реалізує такі функції як зберігання, читання і запис інформації на карту, які потрібні для нормального функціонування пристрою на базі мікроконтролера [23].Умовне позначення Micro SD адаптера відображене на рисунку 2.8.

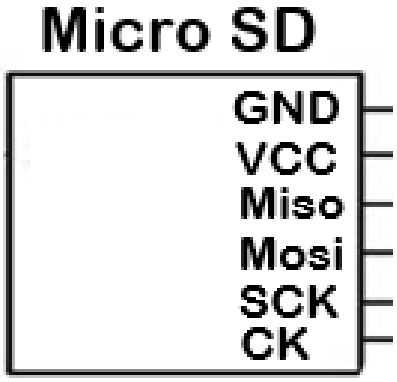


Рисунок 2.8 – Умовне позначення Micro SD адаптера

Призначення виводів:

– GND – мінус живлення, призначений для з’єднання з виводом GND на платі.

– VCC - плюс живлення (5В), з’єднаний з виводом VU на платі.

– MISO (MasterInSlaveOut) - лінія для передачі даних відвідомого пристрою (Slave) до ведучого (Master), з’єднаний з виводом D6 на платі.

– MOSI (MasterOutSlaveIn) - лінія для передачі даних відпровідного пристрою (Master) до веденим (Slave), з’єднаний з виводом D7 на платі.

– SCK (SerialClock)-тактові імпульси, що генеруються провідним пристроєм (Master) для синхронізації процесу передачі даних, з’єднаний з виводом D5, на платі.

– CS - З'єднується з землею через резистор для вимірювання струму через ключ MOSFET, з’єднаний з виводом D8 на платі [27].

Датчик температури та вологості. В якості датчика температури та вологості, використовується цифровий датчик DHT22. Зовнішній вигляд DHT 22 відображений на рисунку 2.9.

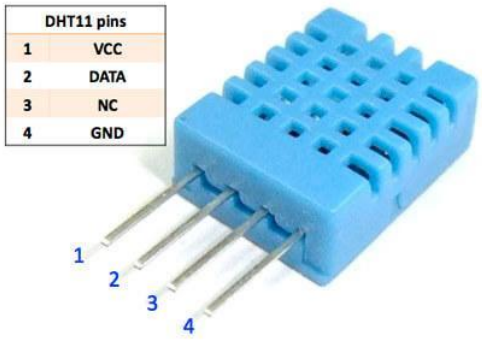


Рисунок 2.9 – Зовнішній вигляд DHT22

Технічні характеристики датчика:

– живлення: DC 3,5 - 5,5 В;

– струм живлення;

– в режимі вимірювання 0.3mA ;

– в режимі очікування 60μA ;

– визначення вологості 20-80% з точністю 5%;

– визначення температури 0-50 ° С з точністю 2%;

– частота опитування не більше 1 Гц;

– розміри 15,5'12'5,5 мм Датчик вібрації;

Датчик має 4 виведення стандарту 2,54 мм:

– VCC (живлення 3-5 В);

– DATA (виведення даних);

– GND (земля).

Між виводами живлення і виведення даних необхідно розмістити резистор. Рекомендований номінал 10 кОм, якщо відстань від датчика до мікрокотролера невелика, для відстані більше 20 метрів рекомендується резистор номіналом 5,1 кОм [15].

Також рекомендується конденсатор (фільтр по живленню між VCC і GND).

Протокол обміну – одно провідний, за структурою дуже схожий на DS18B20, але з відмінностями:

– кожен DS18B20 має персональний ідентифікатор, що дає можливість підключення декількох таких датчиків до одного піну Arduino. Однак у DHT такої можливості немає - один датчик буде використовувати строго один цифровий пін. Умовне позначення DHT 22 відображене на рисунку 2.10.

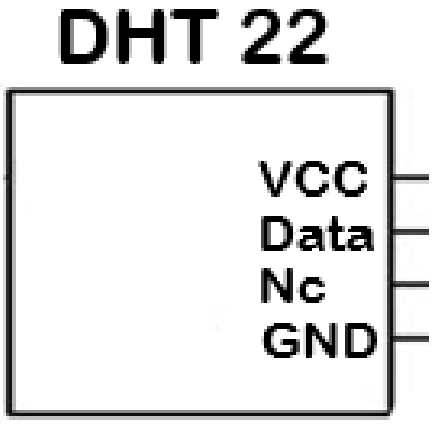


Рисунок 2.10 – Умовне позначення DHT 22

Датчик вібрації KS-037 801S. Датчик нахилу, вібрації KS-037 801S, модуль може використовуватися в нескладних проектах на мікроконтролерах, де потрібно стежити за рівнем нахилу або вібрації [14]. Зовнішній вигляд датчика вібрації відображений на рисунку 2.11.

Рисунок 2.11 – Зовнішній вигляд KS-037 801S

Цей модуль ідеально підходить для зондування вібрації, як правило, в роботах проекту та ситуаціях з коливанням об’єкта. Вивід аналогового сигналу забезпечує аналоговий вихід на основі рівня вібрації, доступного датчику.

Висновок цифрового сигналу виводить «HIGH», коли рівень перевищує рівень тригера і «LOW», коли він знаходиться нижче [19]. Умовне позначення датчика вібрації відображене на рисунку 2.12.

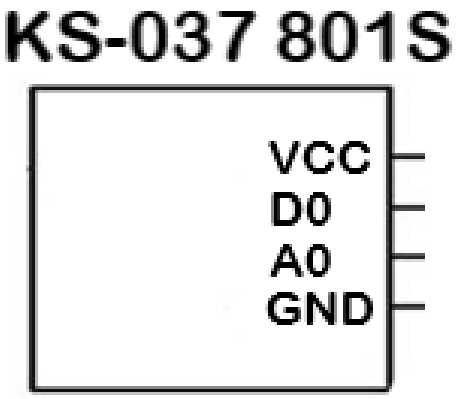


Рисунок 2.12 – Умовне позначення датчика вібрації

Рівень тригера можна регулювати за допомогою бортового потенціометра. Працюючи з низьким входом 3,3 - 5 В, модуль виводитиме високі значення при виявленні вібрації і низьких значень, не виявляючи. Індикатор даних встановлюється на плату, яка вимикається при спрацьовуванні датчика [25]. Крім того, отвір для кріплення забезпечує певний спосіб монтажу пристрою до будь-яких проектів.

2.3 Розробка функціональної схеми пристрою

Функціональна схема являється одним з основних проєктних документів, що визначають функціональну структуру та обсяг системи яка, проєктується та її окремих модулів. Схема призначена для відображення детальної структури пристрою, його основних блоків, вузлів, частин із вказанням зв’язків між ними.

З функціональної схеми має бути зрозуміло, навіщо використовуються вказані блоки і як вони працюють в основних режимах роботи, як підключаюься і взаємодіють їхні складові частини. Функціональна схема системи зображена на рисунку 2.13.

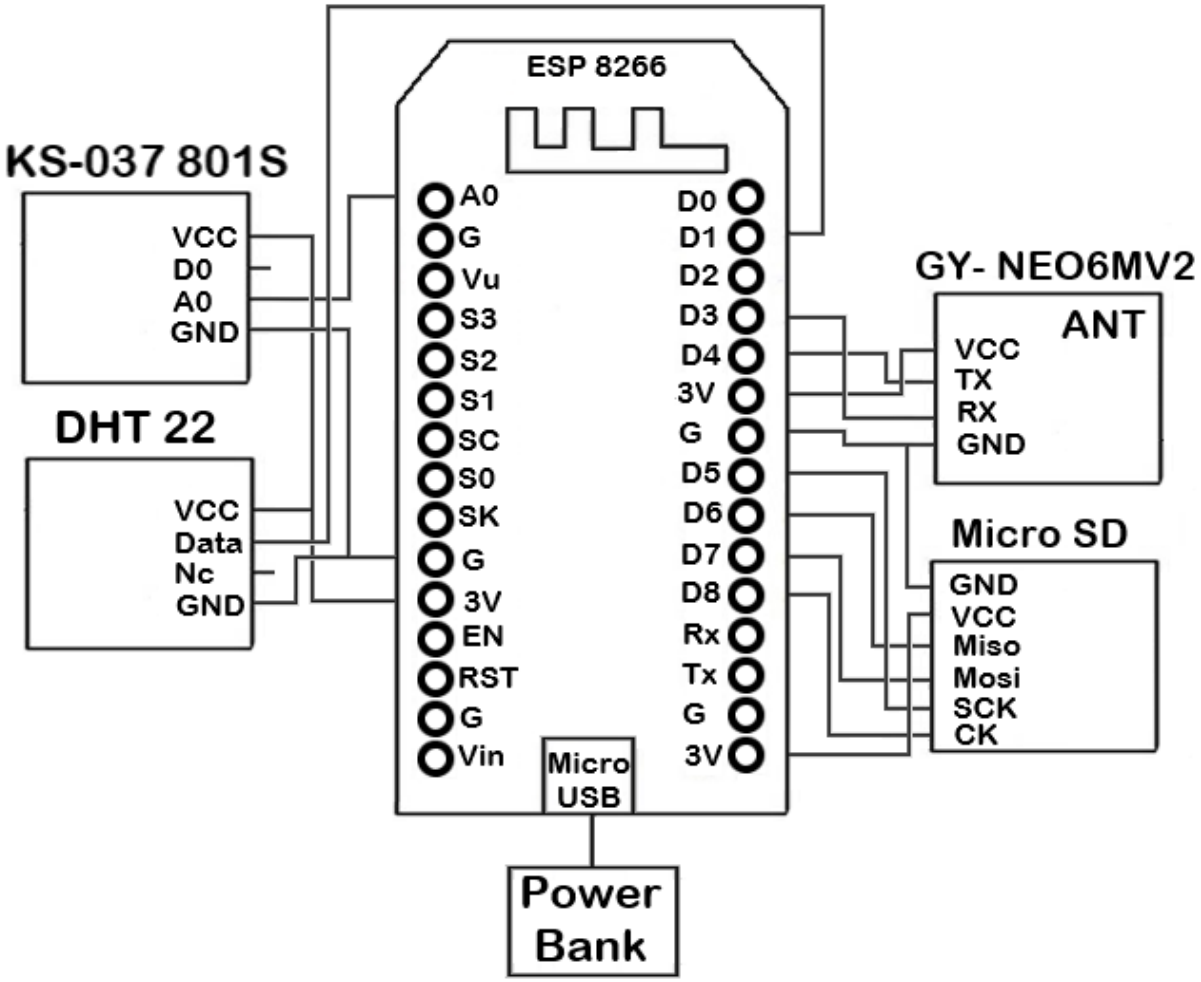


Рисунок 2.13 – Функціональна схема пристрою

Позначення елементів на схемі:

– D1 – датчик температури та вологості DHT 22

– D2/A0 – датчик вібрації KS-037 801S

– D3, D4 – gps модуль GY-NEO6MV2

– D5, D6, D7, D8 – модуль SD карти.

2.4 Розробка алгоритму роботи системи та написання текстів програми

При розробці програми для мікроконтролера, весь код програми був розбитий на окремі функціональні блоки, кожен з яких відповідає за певний

етап процесу роботи системи [4]. Можна виділити наступні функціональні блоки, які наведені на рисунку 2.14:

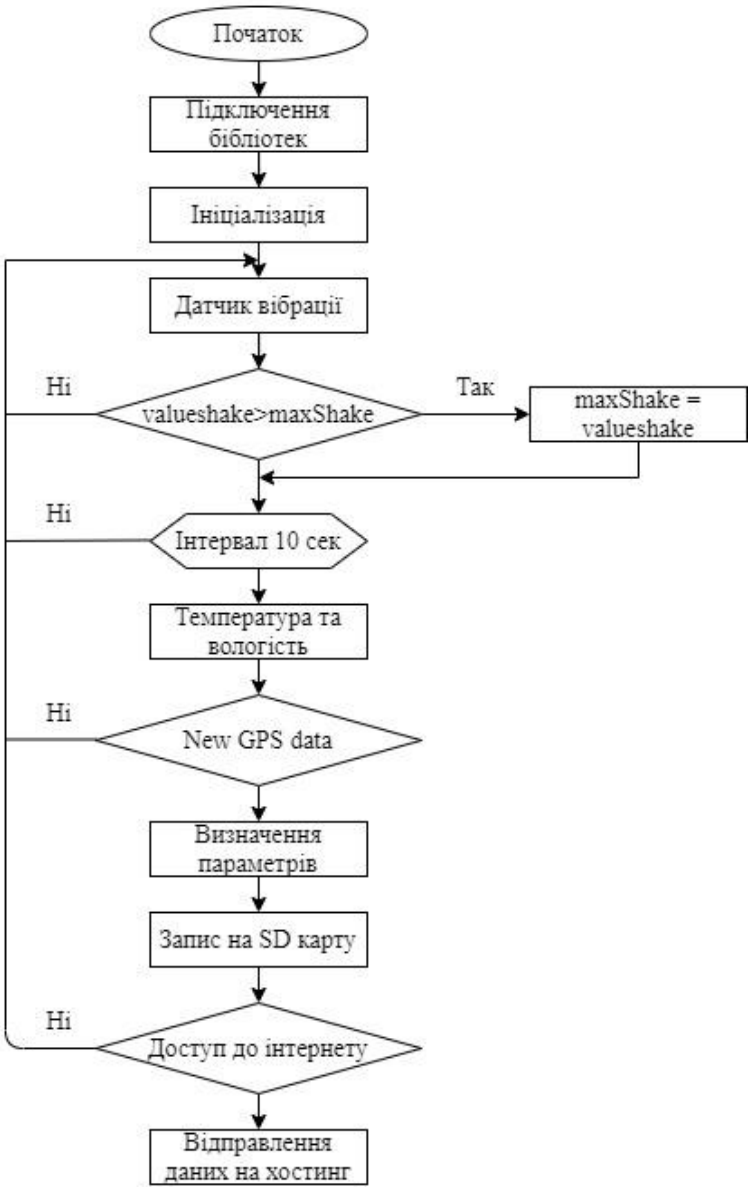


Рисунок 2.14 – Алгоритм роботи системи

- Вологості та температури – зчитування значень вологості tempDHT = dht2.readTemperature( ); humDHT = dht2.readHumidity( ).
- Вібрації – зчитування значення вібрації valueshake з аналового виводу, задання його значення у вигляді відсотків, та присвоєння йому максимального значення maxShake [14].

- Модуль GPS – отримання нових данних «newdata», за допомгою їх зчитуваня «readgps», створення стрічок для візуалізації інформації довготи «lat», широти «lon». Отримавши інформацію «gps.get_position(&lat, &lon);».
- Визначення параметрів -дати «date», часу «time», хвилини «minute», секунди «seconds», отримуючи дату та час «gps.get_datetime(&date,&t_time);» і довготи та широти, отримавши інформацію «gps.get_position(&lat, &lon);».
- Алгоритм з картою пам’яті та хостингом. Ініціалізація SD картки, якщо вона працює, створюються файли, в які за допомгою стрічок записуються дані про довготу, широту, дату, час, температуру, вологість, вібрацію, використовуючи свій, порядковий номер в списку «id».

При написанні коду для мікроконтолера використовується середовище Arduino IDE. Arduino - це інструмент для проектування електронних пристроїв (електронний конструктор), що тісніше взаємодіє з фізичним середовищем, ніж звичайні персональні комп’ютери, які насправді не виходять за рамки віртуальності [11].

Це платформа, призначена для "фізичних обчислень" з відкритим кодом, побудована на простій друкованій платі з сучасним середовищем для написання програмного забезпечення. Arduino використовується для створення електронних пристроїв з можливістю прийому сигналів від різних цифрових та аналогових датчиків, які можуть бути підключені до нього, та управління різними виконавчими механізмами. Конструкції пристроїв на базі Arduino можуть працювати самостійно або взаємодіяти з програмним забезпеченням на комп’ютері (наприклад, Flash, Processing, MaxMSP). Середовище розробки програмного забезпечення з відкритим кодом доступне для безкоштовного завантаження [2].

При написанні текстів для веб сторінки використовувались скрипти мови PHP. Index.php – в даний скрипт входить, створення сторінки, здання на дій мапи «Open street map», задання її розмірів. Створення маркера та інформації яка буде міститись в ньому. Ініціалізація змінних, які зчитуються з карти пам’яті в контролері [3].

Структура GET запиту для передачі даних для відображення маркера та запису даних про переміщення вантажу у файл «data.txt»

2.5 Розробка інструкції з експлуатації електричного пристрою

Для того, щоб система виконувала свої функції, перш за все до ESP 8266 потрібно приєднати такі елементи: Gps модуль, SD модуль, датчик DHT 22 та датчик SW 420. Підключення протрібно проводити відповідно до функціональної схеми, зображеної на рисунку 3.1. Потім потрібно перевірити правильність з'єднання [11]. Після того, як вище перераховані елементи системи з'єднані між собою та приєднані до ESP 8266, система готова до роботи. При подачі живлення світлодіод на платі починає миготіти, це свідчить про готовність системи до роботи [18]. Після цього на GPS модулі починає миготіти світлодіод, який повідомляє про те, що дані координат з супутників почали записуватись. Після кожного вмикання системи будуть записуватись нові треки координати у вигляді Log1, Log2. При запуску роботи системи на SD карті створюється два файли: LOG.txt, та Info.txt. Мікроконтролер використовує SD карту для запису GPS координат у файлі, у вигляді: id(номера) (date) дати/місяця/року, (time) години/хвилини/секунди, та данних координат (Latitude) широти та (Longitude) довготи. Також на SD карту записується файл, з інформацією з датчиків у вигляді: temp (температура), hum (вологість), shake (вібрація) Структура вмісту у файлу, відображена на рисунку 2.15.

```
Id,Date,Time,Latitude,Longitude
1,21/05,17:33:21,49.57431,25.64140
2,21/05,17:33:26,49.57432,25.64135
```

Рисунок 2.15 – Структура вмісту у файлі

Координати записані у файл можна візуалізувати за допомогою інтернет платформи «GPSVizualizer», виконавши наступну послідовність дій. Висмикнути карту пам'яті, вставити її у смартфон або комп'ютер та перейти на

сайт візуалізатора. Візуальний вигляд GPSVisualizer можна побачити на рисунку 2.16 [23].

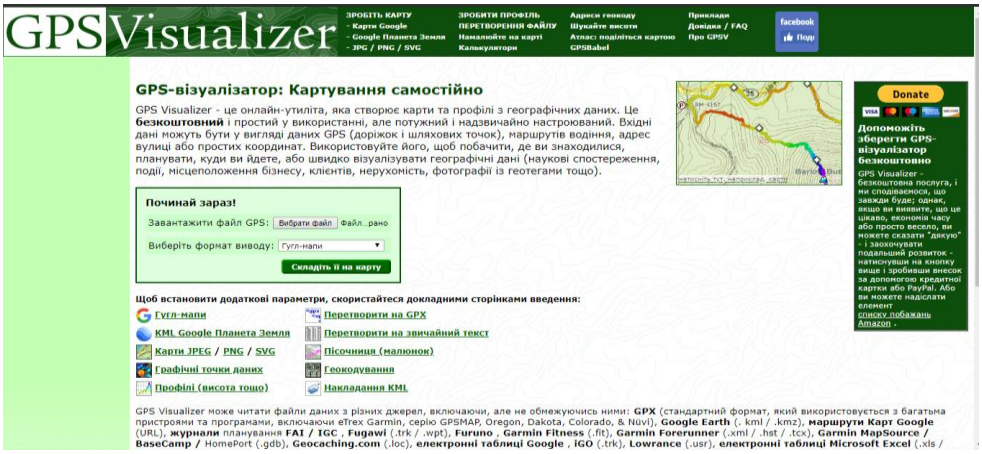


Рисунок 2.16 – Візуальний вигляд GPS Visualizer

Після цього вибрати файл «LOG.txt» який знаходиться на карточці пам'яті та вибрати формати виводу координат. Та натиснути на кнопку «Складіть її на карту». На рисунку 2.17 зображений формат виводу за допомогою програми Googlemaps (гугл карти) [21].

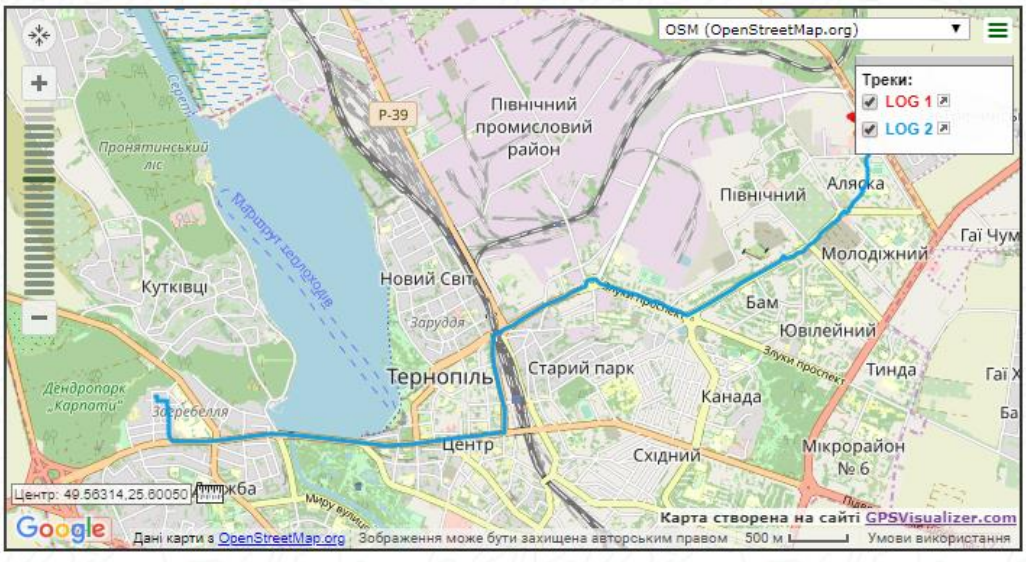


Рисунок 2.17 – Приклад візуалізації на гугл карті

Також можна за допомогою програми «Гугл Планета Земля». Вибравши формат «Гугл Планета Земля», та натиснувши на кнопку «Складіть її на карту».

Після цього з’явиться вікно з файлом. Вигляд вікна з файлом зображений на рисунку 2.18.

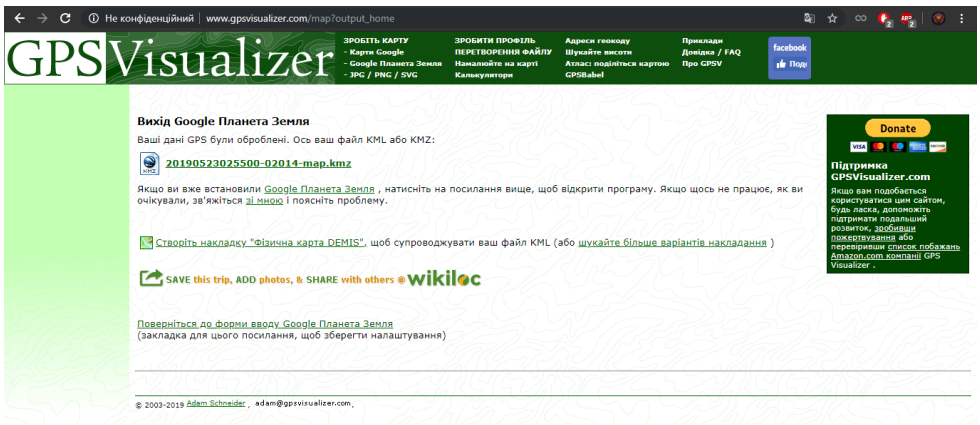


Рисунок 2.18 – Візуальний вигляд вікна з файлом.

Натиснувши на нього він завантажиться. Відкрити це файл можна за умови що у вас установлена програма «Гугл Планета Земля». Якщо програма установлена тоді натиснувши не цей файл, відкриється програма та відобразиться пройдений системою шлях. Приклад візуалізації за допомогою програми «Гугл планета Земля» зображений на рисунку 2.19.

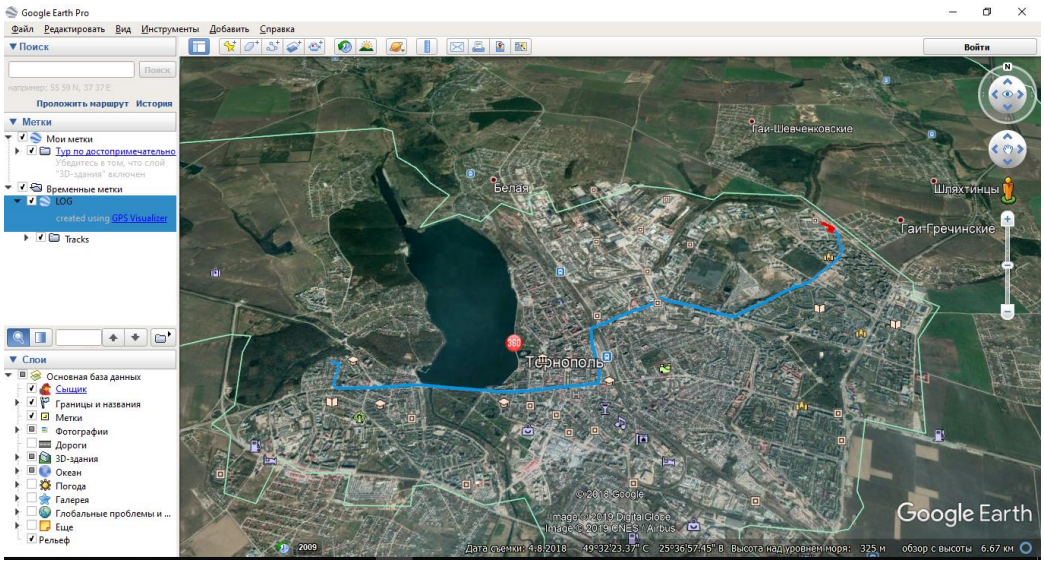


Рисунок 2.19 – Приклад візуалізації в програмі «Google Earth»

На борту транспорту який перевозить вантаж повинен бути встановлений WiFi роутер з доступом до інтернету [5]. Це дозволяє переглядати дані про місце перебування багажу та інформацію датчиків в реальному часі на гугл карті, за допомогою міток на сайті [7]. Візуальний вигляд маркера зображений на рисунку 2.20.

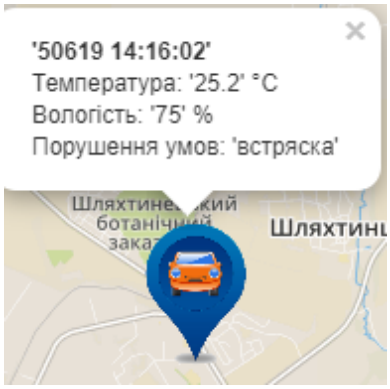


Рисунок 2.20 – Візуальний вигляд маркера

Маркер на карті відображає поточне місце знаходження багажу, в якому відображається: число, місяць, рік, години, хвилини, секунди, інформацію про температуру, вологість, та вібрацію. Візуальний вигляд карти зображений на рисунку 2.21.

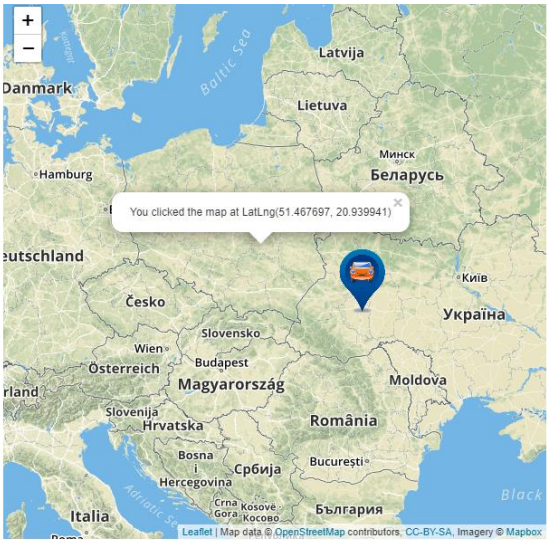


Рисунок 2.21 – Візуальний вигляд карти

При натиску на іншу зону карти буде відображений коментар з координатами вказаними курсором [9]. Візуальний вигляд коментаря зображений на рисунку 2.22.

Рисунок 2.22 – Візуальний вигляд коментаря

2.6 Пошук, виявлення та виправлення несправностей

У системі GPS моніторингу за багажем можливі такі види несправностей.

1) При входженні на сайт не завантажується веб сторінка:

– перевірити з’єднання з інтернетом. Якщо інтернет не працює, слід спробувати перезавантажити інтернет, або дзвонити до провайдера та повідобляти про цю проблему;

– перевірити чи працює сервер. Якщо він не працює, слід його перезавантажити. Якщо це не допомогло, слід перейти на хостинг, та перевірити з’єднання з ним. У випадку, в якому не заходить на хостинг, слід писати в службу підтримки.

2) На веб сторінці данні відображаються не коректно:

– перевірити з’єднання всіх елементів (датчиків, модулів);

– перезавантажити систему;

– перевантажити код.

3) Дані на веб сторінці застарілі або їх взагалі немає:

– перевірити працездатність системи;

– перевірити з’єднання з інтернетом, та перевірити вказані в прошивці данні про ім’я та пароль мережі;

– перезавантажити систему;

– перезавантажити код.

4) Не відображається, або не коректні значення інформації з датчиків:

– перевірити працездатнійсть датчика. Перевірити можна за допомогою тестера, навівши його на виходи датчиків. У випадку, в якому датчик вийшов з ладу слід його замінити;

– перевірити працездатність плати. Перевірити працездатність плати також за допомогою тестера, переглянувши роботу його виводів. У випадку, якщо виводи не працездатні, слід замінити плату, або пересвстановити датчики на інші виводи, переставивши їх та задавши іхні значення в код мікроконтролера.

5) Дані з мікроконтролера не передаються:

– перевірити працездатність мікроконтролера, перевіривши напругу на його виводах тестером;

– перевірити систему автономного живлення, якщо індикатор ні ній погас, слід перезарядити пристрій. Якщо індикатор світиться але система не завантажується, слід перервірити працездатість USB порта. Якщо індикатор світиться, або не світиться і не дає напругу, слід замінити систему живлення.

2.7 Висновок до другого розділу

В процесі виконання другого розділу:

1) реалізовано поставлені задачі, розроблено структурну та функціональні схеми;

2) представлено інструкції з експлуатації електричного пристрою та алгоритм роботи при виявлені несправностей пристрою.

РОЗДІЛ 3. БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

3.1 Характеристика життєдіяльності людини у системі “людина – машина середовище існування”

В історичному аспекті розвитку трудової діяльності людини можна виділити три основні стадії праці: ручна, механізована та автоматизована [20].

Протягом тривалого часу, майже до початку нашого століття, функції людини стосовно техніки залишались в основному енергетичними, тобто для керування технікою людина користувалась своєю м'язовою силою. Ця праця характеризується складними руховими процесами, які вимагають значних затрат фізичної сили, високої координації рухів, спритності. Узгодження людини з технікою зводилось лише до врахування анатомічних та фізіологічних особливостей.

З появою на початку XX ст. нових видів діяльності виникла потреба врахування психологічних можливостей людини, таких як швидкість реакції, особливості пам'яті та уваги, емоційний стан. З широким впровадженням автоматичних систем керування, комплексної механізації та автоматизації виробничихпроцесів з'явилися зміни у фаховій структурі праці, пов'язані з появою операторської діяльності.

Особливості діяльності людини-оператора

Оператором стали називати людину, яка керувала елементами автоматики та обчислювальної техніки, іншими технічними системами. Під "людиною-оператором" в ергономіці розуміють людину, котра виконує трудову діяльність, основу якої становить взаємодія з предметом праці, машиною та зовнішнім середовищем завдяки інформаційній моделі та органам керування.

Операторська діяльність значно змінила працю людини. Збільшилась напруга у праці тому, що перед оператором постає завдання керувати все більшою кількістю об'єктів та параметрів. Людина має справу не з прямим спостереженням, а з інформаційним відображенням. Зростають вимоги до

точності, швидкості та надійності дій людини, до швидкості психологічних процесів. Трудова діяльність супроводжується значними витратами нервово-емоційної та розумової енергії [1].

Комп'ютеризація та роботизація, з одного боку, розширили можливості людини, а з іншого значним чином змінили вимоги до її діяльності. Вже не потрібна примітивна праця з виконанням монотонних фізичних операцій, з шаблонною розумовою діяльністю. Збільшилась потреба у творчій висококваліфікованій праці. Ускладнилась проблема узгодження умов праці, конструкції машин з психологічними та фізіологічними можливостями людини. Людина стала невід'ємною і найважливішою складовою частиною системи ЛМС.

Для того, щоб керувати технологічним процесом, спостерігати та контролювати роботу, оператору необхідні дані, котрі характеризують як хід процесу, так і відповідні органи керування. При керуванні процесом оператору доводиться переробляти великий обсяг інформації [12]. При цьому він зазнає нервового перенапруження. Для розв'язання проблем психологічного характеру конструктори намагаються пристосувати машину до людини так, шоб забезпечити найсприятливіший режим роботи.

Механічні зміни керованого об'єкта автоматизовуються за допомогою датчиків. Сигнали від датчиків перетворюються і подаються до приладів, за котрими спостерігає людина. Вона сприймає показання приладів, розшифровує їх, приймає рішення, виконує відповідні дії. Сигнал, що виникає внаслідок дій людини, перетворюється і надходить до керованого об'єкта, змінюючи його стан.

Основною формою діяльністю людини-оператора є використання та опрацювання інформації. При розробці інформаційної системи GPS-контролю перевезень багажів, сигнали від датчиків передаються на інформаційну панель. Людина сприймає інформацію, переробляє і через пульт керування впливає на систему.

3.2 Безпечність при використанні вантажопідйомних робіт та транспортуванні вантажів

Для безпечного виконання вантажно-розвантажувальних робіт дотримувати відповідних вимог правил, які затверджені українським законодавством в сфері безпеки та охорони праці [13].

Правила охорони праці під час вантажно-розвантажувальних робіт, затверджені наказом Міністерства енергетики та вугільної промисловості України 19 січня 2015 року № 21, поширюються на всіх суб'єктів господарювання (роботодавців та працівників) незалежно від форм власності та організаційно-правової форми, які в процесі своєї діяльності виконують [24]:

– вантажно-розвантажувальні роботи;

– навантаження (розвантаження);

– вивантаження.

Організація безпечного виконання вантажно-розвантажувальних робіт. Для організації безпечного виконання вантажно-розвантажувальних робіт на підприємстві, роботодавцеві дотримувати наступних вимог:

– створити службу охорони праці;

– організувати опрацювання і затвердити нормативні акти про охорону праці, що діють на підприємстві;

– розробити та затвердити інструкції з охорони праці; забезпечити проведення попереднього та періодичних медичних оглядів; розробити і затвердити перелік робіт з підвищеною небезпекою;

– організовувати проведення атестації робочих місць за умовами праці;

– одержати дозвіл на виконання робіт підвищеної небезпеки та на експлуатацію машин, механізмів, устаткування підвищеної небезпеки;

– забезпечити працівників спецодягом, спеціальним взуттям та іншими засобими індивідуального захисту.

Вантажно-розвантажувальні роботи можна виконувати на підприємстві у випадку дотримання й інших вимог щодо безпеки та охорони праці [26]:

– розслідування та облік нещасних випадків, професійних захворювань і аварій на виробництві здійснюються відповідно до вимог законодавства;

– забороняється залучення жінок до робіт, визначених у “Переліку важких робіт та робіт із шкідливими і небезпечними умовами праці, на яких забороняється застосування праці жінок”;

– піднімання та переміщення важких речей жінками необхідно здійснювати з дотриманням граничних норм підіймання і переміщення важких речей жінками, що встановлені нормативними документами;

– забороняється залучення неповнолітніх до робіт, визначених законодавством;

– підіймання та переміщення важких речей неповнолітніми необхідно здійснювати з дотриманням вимог відповідних нормативних актів [13];

– засоби індивідуального захисту працівників повинні відповідати вимогам законодавств, працівники повинні бути забезпечені спецодягом, спецвзуттям та іншими ЗІЗ. Не допускаються до роботи працівники без відповідних ЗІЗ [12];

– навчання і перевірка знань з питань охорони праці посадових осіб та працівників повинні проводитися відповідно до вимог “Типового положення про порядок проведення навчання і перевірки знань з питань охорони праці”, від 15 лютого 2005 р.

Вимоги щодо створення безпечних умов праці.

Створення безпечних умов праці при вантажно-розвантажувальних роботах відбувається з урахування наступних вимог:

– роботодавець забезпечує відповідний санітарно-гігієнічний стан виробничих приміщень;

– під час організації та ведення технологічних процесів, пов’язаних із застосуванням шкідливих речовин, дотримувати правила, що затверджені нормативними актами з цього питання;

– створити відповідний мікроклімат у виробничих приміщеннях;

– природне та штучне освітлення на робочих місцях відповідати вимогам законодавства; до експлуатації допускаються справне устаткування, механізми або пристрої, що відповідають вимогам безпеки [16];

– роботи щодо очищення цистерн виконувати не менше ніж три працівники, один із яких перебуває в цистерні, а двоє працівників, що спостерігають за виконанням роботи, повинні перебувати поза цистерною.

– між спостерігачами і працівником у цистерні повинен бути встановлений переговірний зв’язок або визначені сигнали, які передаються за допомогою страхувального каната (мотузки), що забезпечує підйом працівника нагору на його вимогу;

– навантаження (розвантаження) балонів з горючими газами, а також легкозаймистих рідин, речовин і матеріалів, здатних накопичувати на своїй поверхні заряди статичної електрики, та в спецвзутті, підбитому металевими (крім латунних) цвяхами або підковами [26].

При організації роботи інформаційної системи GPS-контролю перевезень багажів, дотримані всі правила та вимоги безпечності при виконанні вантожопідйомних робіт та транспортуванні вантажів.

3.3 Особливості безпеки праці під час вантажно-розвантажувальних робіт

Механізація найбільш важких та трудомістких робіт, до яких, в першу чергу, належать вантажно-розвантажувальні роботи, є одним з найважливіших завдань охорони праці. Разом з тим, на сьогодні ще досить значною є частка вантажно-розвантажувальних робіт, що виконуються вручну [1]. Аналіз виробничого травматизму, пов'язаного з виконанням вантажно-розвантажувальних робіт, свідчить, що найбільш високий його рівень - там, де такі роботи виконуються вручну. Тому максимальна механізація таких робіт не лише полегшує працю працівників, але й робить її більш безпечною [12].

Безпека під час виконання вантажно-розвантажувальних робіт залежить від групи, класу та категорії вантажу [17]. В залежності від небезпеки, яка виникає під час навантажування, транспортування та розвантажування, всі вантажі поділяються на чотири групи:

– малонебезпечні (будматеріали, продукти харчування);

– небезпечні за своїми розмірами;

– пилові та гарячі (цемент, крейда, вапно, асфальт, бітум);

– небезпечні за своїми властивостями (пожежо- та вибухонебезпечні, отруйні, токсичні, радіоактивні речовини).

Під час виконання вантажно-розвантажувальних робіт з вантажами третьої та четвертої груп використовувати засоби індивідуального захисту.

Вантажі, які є небезпечними за своїми властивостями відповідно до ГОСТ 19433-81 поділяються на дев'ять класів: 1 - вибухові речовини; 2 - стиснені, зріджені та розчинені гази під тиском; 3 - легкозаймисті рідини, суміші рідин, які виділяють легкозаймисті пари, температура спалаху яких становить 61 °С і нижче; 4 - легкозаймисті речовини та матеріали, які здатні займатися внаслідок тертя, нагрівання, поглинання вологи, самочинних хімічних перетворень; 5 - окиснювальні речовини, які легко виділяють кисень; 6 - отруйні та інфекційні речовини; 7 - радіоактивні речовини; 8 - їдкі та корозійно активні речовини; 9 - речовини з відносно низькою небезпекою, під час перевезення та зберігання яких необхідно дотримуватись певних вимог безпеки [24].

На упаковці з небезпечними вантажами, крім стандартного маркування, нанести знак небезпеки. Цей знак має форму квадрата, окантованого чорною рамкою, що повернений на кут і поділений на два однакових трикутники. У верхньому трикутнику наносять символ небезпеки, а у нижньому роблять напис про небезпечність вантажу та номер класу [24]. Знаки можна побачити на рисунку 3.1

Рисунок 3.1 - Приклади знаків небезпеки на упаковках з небезпечними вантажами

Майданчики для проведення вантажно-розвантажувальних робіт: мати рівне та тверде покриття з ухилом не більше ніж 5°, а також природне та штучне освітлення. У місцях виконання вантажно-розвантажувальних робіт необхідно встановити знаки безпеки, відповідно до ГОСТу 12.4.026-76.

До роботи з підіймально-транспортними механізмами та пристроями допускаються особи, не молодші 18 років, які пройшли медичний огляд і спеціальне навчання, склали іспит кваліфікаційній комісії та одержали посвідчення [26].

Для забезпечення безпеки при розробці інформаційнох системи GPS-контролю пеевезень багажів дотриманно встановлені правила складування вантажів. Так, кошики з бутлями агресивних речовин розміщують у складах лише в один ряд, барабани з карбідом кальцію - не більше двох ярусів. Якщо немає відповідних застережень, то вантажі у стандартній тарі зазвичай складають у штабелі. Відношення висоти штабеля до довжини найменшої сторони тари, що штабелюється, не повинно бути більше ніж 6 - для нерозбірної тари, та 4 - для розбірної тари. Ширина штабеля не менша, ніж його висота. Навантаження на нижню тару перевищує допустимі значення [13].

3.4 Висновок до третього розділу

В третьому розділіквалiфікаційної роботи розглянуто питання Безпеки життєдіяльності, основи охорони праці на підприємстві.

Основною формою діяльністю людини-оператора є використання та опрацювання інформації. При розробці інформаційної системи GPS-контролю перевезень багажів, сигнали від датчиків передаються на інформаційну панель. Людина сприймає інформацію, переробляє і через пульт керування впливає на систему.

При організації роботи інформаційної системи GPS-контролю перевезень багажів, дотримані всі правила та вимоги безпечності при виконанні вантожопідйомних робіт та транспортуванні вантажів.

Для забезпечення безпеки при розробці інформаційнох системи GPS-контролю пеевезень багажів дотриманно встановлені правила складування вантажів.

Отже, поняття «охорона праці» визначено статтею 1 Закону України «Про охорону праці». Охорона праці — це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів і засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі праці.

Головною метою охорони праці є створення на кожному ро-бочому місці безпечних умов праці, безпечної експлуатації облад-нання, зменшення або повна нейтралізація дії шкідливих і небез-печних виробничих факторів на організм людини і, як наслідок, зниження виробничого травматизму та професійних захворювань.

ВИСНОВКИ

В першому розділі кваліфікаційної роботи освітнього рівня «Бакалавр»:

– Проведено аналіз відомих рішень та їх порівняльна характеристику.

– Сформульовано мету розробки приладу та поставлено задачі для досягнення мети.

В другому розділі кваліфікаційної роботи:

– Розроблено структурну та функціональну схеми.

– Розглянуто елементну базу, та обґрунтовано вибір кожного елементу.

– Спроєктовано та описано алгоритм роботи програми.

– Протестовано роботу пристрою, розроблено інструкцію з експлуатації електричного пристрою.

– Описано алгоритм дій при виявленні несправностей системи.

У розділі «Безпека життєдіяльності, основи хорони праці». Висвітлено такі питання, а саме:

− Характеристика життєдіяльності людини у системі “людина – машина середовище існування”.

− Безпечність при використанні вантажопідйомних робіт та транспортуванні вантажів.

− Особливості безпеки праці під час вантажно-розвантажувальних робіт.

ПЕРЕЛІК ДЖЕРЕЛ

1. Закон України «Про охорону праці» від 14.10.1992 № 2694-XII із змінами та доповненнями від 04.02.2021 № 1213-IX // Відомості Верховної Ради України. 1992.

2. Анісімов А.В. Інформаційні системи та бази даних: навчальний посібник для студентів факультету комп'ютерних наук та кібернетики / А.В. Анісімов, П.П. Кулябко. – Київ: Основа, 2017. – 110 с.

3. Баженов В.А. Інформатика. Комп'ютерна техніка. Комп'ютерні технології: підручник / В.А. Баженов, П.С. Венгерський, В.С. Гарвона. – Київ: Каравела, 2018. – 592 с.

4. Бережна О.Б. Інформатика та комп'ютерна техніка. 1 частина: навч. посібник / О.Б. Бережна. – Харків: ХНЕУ, 2018. – 164 с.

5. Бичков О.С. Основи сучасного програмування: підручник / О.С. Бичков. – Київ: Основа, 2017. – 285 с.

6. Протоколи з'єднання з IoT [Електронний ресурс] // Режим доступу: http://www.rtsoft.ru/press/articles/detail.php?ID=2718.

7. Вінник В.Ю. Алгоритмічні мови та основи програмування: навчальний посібник / В.Ю. Вінник. – Житомир: ЖДТУ, 2018. – 328 с.

8. Вільна енциклопедія Вікіпедія [Електронний ресурс] // Режим доступу: http: //uk.wikipedia.org 16.

9. Воронін А.М. Інформаційні системи прийняття рішень: навчальний посібник / А. М Воронін, А.С. Климова. − К.: НАУ-друк, 2017. − 136с.

10. Григорович В.Г. Методичні вказівки до виконання лабораторних і практичних завдань з курсу «Алгоритмізація та програмування». Частина 1: навч. посібник / В.Г. Григорович. – Дрогобич: Редакційно-видавничий відділ Дрогобицького державного педагогічного університету імені Івана Франка, 2018. – 540 с.

11. Глинський Я.М. Інформатика. Практикум з інформаційних технологій: навч. посібник / Я.М. Глинський. – Тернопіль: Підручники і

посібники, 2015. – 304 с.

12. Грибан В.Г. Охорона праці: навч. посібник / В.Г. Грибан, О.В. Негодченко. – Київ: Центр учбової літератури, 2016. 280 с.

13. Геврик Є.О. Охорона праці: навч. посібн. / Є.О. Геврик. – Київ: Ніка-Центр, 2016. 280 с.

14. Годун В.М. Інформаційні системи і технології: навчальний посібник / В.М. Годун, Н.С. Орленко, М. А. Сендзюк. – Київ: КНЕУ, 2017. – 267 с.

15. Дибкова Л.М. Інформатика і комп'ютерна техніка: навч. посібник / Л.М. Дибкова. – Київ: Академвидавництво, 2016. – 463 с.

16. Жидецький В.Ц. Охорона праці користувачів комп'ютерів: навчальний посібник / В.Ц. Жидецький. – Львів: Афіша, 2017. 176 с.

17. Жернаков В. В. Трудове право: підручник / В.В. Жернаков, С.М. Прилипко, О.М. Ярошенко. – Харків: Право, 2015. 496 с.

18. Інформаційні технології: навчально-методичний посібник / [М.І. Жалдак, О.А. Хомік, І.В. Володько та ін.]. Київ: Основа, 2015. – 194 с.

19. Інформаційні системи в сучасному бізнесі: навчальний посібник / [В.С. Пономаренко, І.О. Золотарьова, Р.К. Бутова та ін.]. Харків: ХНЕУ, 2016. – 484 с.

20. Іванов Ю.Ф. Трудове право України: навч. посібник / Ю.Ф. Іванов, М.В. Іванова. – Київ: Алерта, 2020. 442 с.

21. Інформаційні системи і технології в економіці: навчальний посібник / [В.С. Пономаренко, Р.К., Бутова І.В. Журавльова та ін.].– Київ: Академія, 2016. – 542с.

22. Пасічник О. Г. Основи веб-дизайну: підручник / О.Г. Пасічник. – Київ: Основа, 2017. – 336 с.

23. Ковалюк Т.В. Алгоритмізація та програмування: підручник / Т.В. Ковалюк. – Львів : Магнолія, 2016. – 400 с.

24. Керб Л.П. Основи охорони праці: навч.-метод. посібник для самост. вивч. дисц. / Л.П. Керб. – Київ: КНЕУ, 2015. 252 с.

25. Кравець П.О. Об’єктно-орієнтоване програмування: навч. посібник / П.О. Кравець. – Львів: Видавництво Львівської політехніки, 2017. – 624 с.

26. Основи охорони праці: підручник / [К.Н. Ткачук, М.О. Халімовський, В.В. Зацарний та ін.]. – Київ: Основа, 2016. 448 с.

27. Пономарьова Г.Ф. Нові педагогічні технології: навчально-методичний посібник / Г.Ф. Пономарьова, О.О. Бабакіна, С.Б. Бєляєв. – Харків: ХНЕУ, 2015. – 282с.

28. Мережевий протокол MQTT [Електронний ресурс] // Режим доступу: https://1sheeld.com/mqtt-protocol/.

29. IOTA [Електронний ресурс]. – Режим доступу https://www.iota.org/.

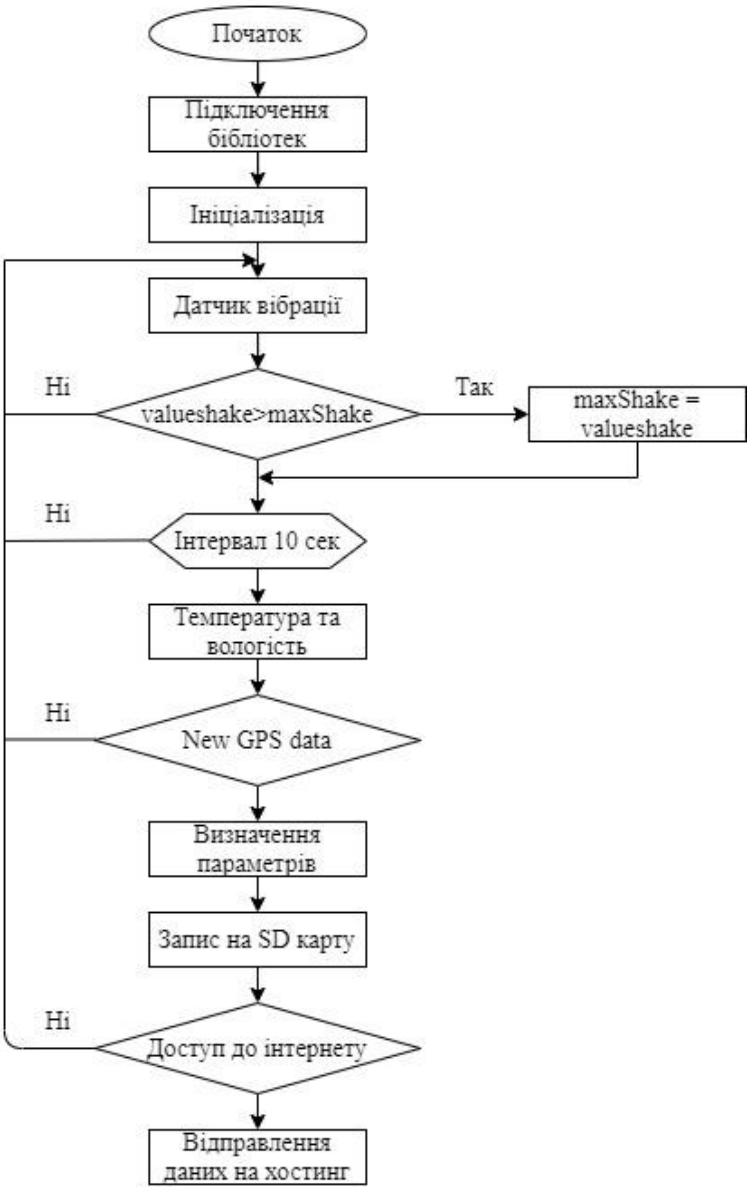
30. Мережевий протокол MQTT [Електронний ресурс] / Режим доступу до ресурсу https://1sheeld.com/mqtt-protocol/.

31. What is Gps-tracking [Електронний ресурс] / Режим доступу до ресурсу https://www.azuga.com/blog/what-is-gps-tracking-and-how-does-it-work.

ДОДАТКИ

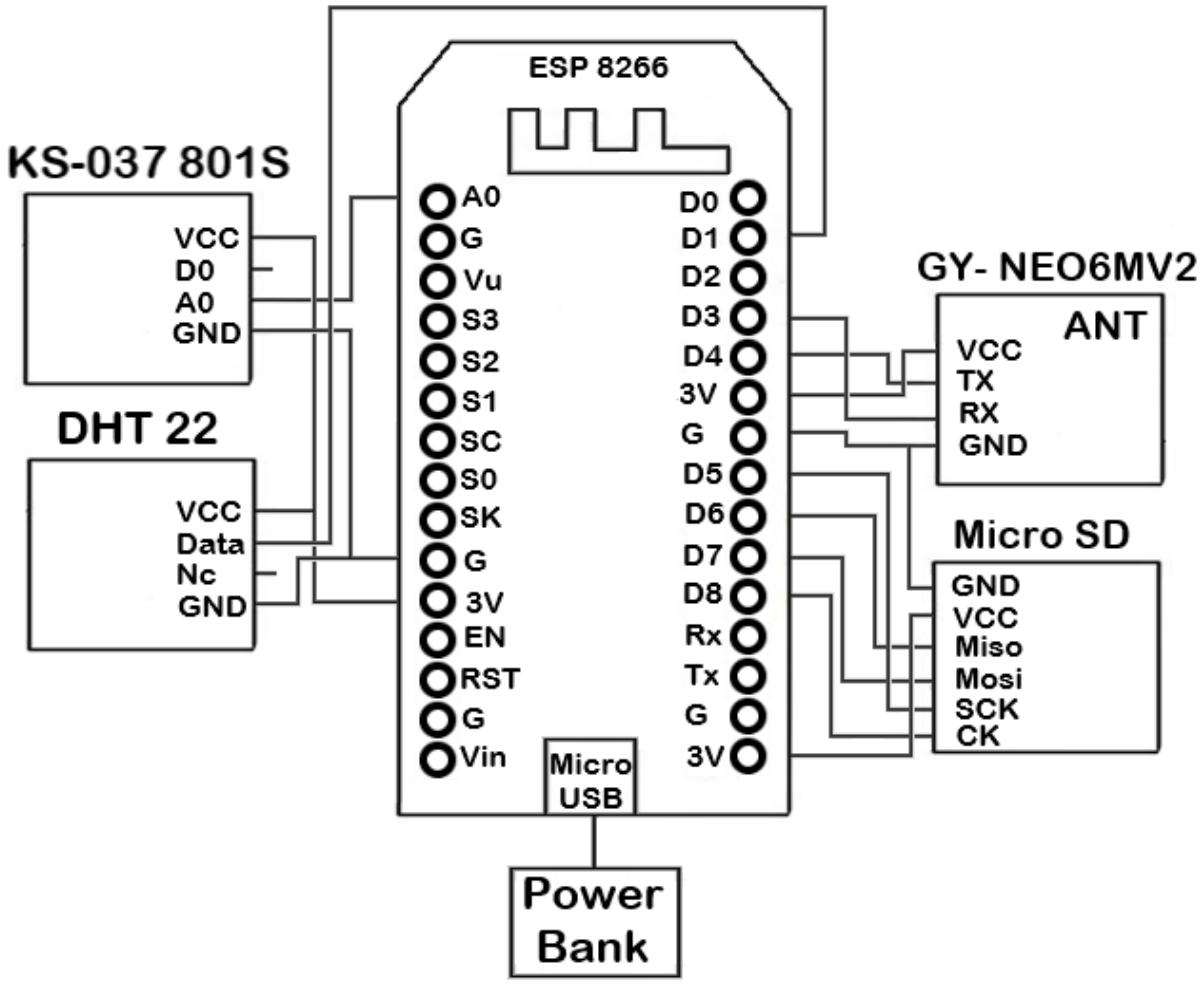
Додаток А

Алгоритм роботи системи



Додаток Б

Функціональна схема пристрою



Додаток В

Порівняльна характеристика відомих рішень

	Triton	BI 310 CICADA	Lookout	Моє рішення
Датчик температури	+	-	+	+
Датчик вологості	-	-	+	+
Датчик вібрації	-	-	-	+
Автономне живлення	+	+	-	+
Доступ до мережі	-	+	-	+
Онлайн перепрошивка	-	+	-	+
Резервне сховище	+	-	-	+
Онлайн моніторинг	-	-	-	

Додаток Г

Лістинг коду для мікроконтролера

Підключення бібліотек

```
#include <ESP8266WiFi.h> //Білбіотека для роботи з WIFI
#include <ESP8266mDNS.h> // Бібліотека для задання IP адресу
#include <WiFiUdp.h>  // Бібліотека для створення комунікації
#include <ArduinoOTA.h> // Библіотека для OTA-прошивки
#include <SD.h> //Бібліотека для роботи з картою пам’яті
#include <SoftwareSerial.h> // Бібляотека для послідовних портів
#include <TinyGPS.h> // Бібоіотека для роботи з Gps координатами
```

Ініціалізація портів

```
DHT dht2(D1, DHT11);
#define CS_PIN  D8
```

Підключення до інтернет мережі

```
const char* ssid = "lab101"; //Имя точки доступа WIFI
const char* password = "101pass101"; //пароль точки доступа WIFI
const char* host = "gps.lab-101.org.ua";  // хостинг
```

Ініціалізація змінних

```
bool newdata = false; /нові данні з GPS модуля
unsigned long start; / початок запису довготи
long lat, lon; / задання довжини довготи та широти
unsigned long t_time, date; / використання довжини часу та дати
String longtitude; / створення стрічки довготи
String latitude; / створення стрічки широти
String dayString; / створення стрічки дня
String monthString; / створення стрічки місяця
String entry; /стрічка запису
String hourString; /стрічка години
String minuteString; / стрічка хвилини
String secondsString; / стрічка секунди
int type  = 0;
int id   = 0;
int sense_Pin = 0;
int valueshake = 0;
String s_date = "";
String s_time = "";
String s_lat  = "";
String s_lon  = "";
String tempDHT = "";
String humDHT  = "";
int maxShake = 0;
boolean SDcardWorked = true;
```

Цикл датчика вібрації

```
valueshake = analogRead(sense_Pin); - зчитування вібрації з аналового виводу.
valueshake = valueshake;
valueshake = map(valueshake, 0, 1023, 0, 100); //задання вигляду у відсотках
```

```
if (valueshake > maxShake) maxShake = valueshake; // якщо значення вібрації більше за максимальне значення, то привтоїти це значення максимальним.
                    Зчитування з датчика DHT 22
tempDHT = dht2.readTemperature( );
humDHT = dht2.readHumidity( );
                          Нові данні GPS
newdata = readgps();//
1) отримання позиції, дати та часу, виведення: довготи, широти, дати, часу
2) створення стрічок
                          Карта пам'яті
if (SDcardWorked) {
File dataFile = SD.open("log.txt", FILE_WRITE);//створення файлу для запису
 // якщо файл був відкритий правильно, запишіть у нього дані
        if (dataFile) {
if (id == 0) dataFile.println("Id,Date,Time,Latitude,Longitude");

          String entry = String(id) + "," + dayString + "/" + monthString + "," + hourString + ":" + minuteString + ":" + secondsString + "," + latitude + "," + longtitude;
dataFile.println(entry);
Serial.println (entry);
//      закриває      файл      після      його      використання
dataFile.close();
        }
        // якщо файл не може бути відкритий, дані не будуть записані.
else {
          Serial.println("Не вдалося відкрити файл log.txt");
        }
        //--------------------------
        dataFile = SD.open("info.txt", FILE_WRITE);
        // якщо файл був відкритий правильно, запишіть у нього дані
if (dataFile) {
          if (id == 0) dataFile.println("Id,Date,Time,temp,hum, shake");
          id++;
          entry = String(id) + "," + dayString + "/" + monthString + "," + hourString + ":" + minuteString + ":" + secondsString + "," + String(tempDHT) + "," + String(humDHT) + "," + String(maxShake) + "%";
          dataFile.println(entry);
          Serial.println (entry);
          // закриває файл після його використання
          dataFile.close();
        }
        // якщо файл не може бути відкритий, дані не будуть записані.
```

```
 else {
          Serial.println("Не вдалося відкрити файл info.txt");
        }
                        З'єднання з хостингом
     WiFiClient client;
          if (client.connect(host, 80))
          {
            id++;

       Ініціалізація змінних
            s_date = String(date);
            s_time = hourString + ":" + minuteString + ":" +
secondsString;
            s_lat = latitude;
            s_lon = longtitude;

     Створення Get запитів
            client.print( "GET /gps/get.php?");       // папка з
файлом get.php
            client.print("type=1");
            client.print("&id=" + String(id));
            client.print("&date=" + s_date);
            client.print("&time=" + s_time);
            client.print("&lat=" + s_lat);
            client.print("&lon=" + s_lon);
            client.print("&temp=" + tempDHT);
            client.print("&hum=" + humDHT);
            client.print("&shake=" + String(maxShake) + '%');
            client.println( " HTTP/1.1");
            client.print( "Host:" );
            client.println(host);
            client.println( "Connection: close" );
            client.println();
            client.println();
          }
                            Цикл з readgps
bool readgps()
{
  while (gpsSerial.available())
  {
    int b = gpsSerial.read();
    //в бібліотеці TinyGPS міститься помилка: не оброблюються
данні с \r і \n    if ('\r' != b)
    {
      if (gps.encode(b))
        return true;
    }
  }
  return false;
}
```

Додаток Д

Лістинг коду для веб-сторінки

```
http://gps.lab-
101.org.ua/gps/get.php?type=5&id=5&date=12.05&time=12:55&lat=49.55
589&lon=25.60556&temp=25.5&hum=80&shake=zzz

http://gps.lab-101.org.ua/gps/index.php     -     веб     сторінка
відображення маркера.
L.marker([lat, lon], {icon: new L.Icon({
        iconUrl:'http://gps.lab-101.org.ua/gps/marker1.png',
//https://latitude.to/img/gmapz/pin-default.png
         iconSize: [74, 74],
        iconAnchor: [32, 64],
        popupAnchor: [1, -66]
В скрипт також входить створення та задання розмірів мапи.

<!DOCTYPE     HTML     PUBLIC     "-//W3C//DTD     HTML     4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<script>
(function(){
 (function(){
function e(g){
this.t={};
this.tick=function(h,k,f){
his.t[h]=[void 0!=f?f:(new Date).getTime(),k];
if(void 0==f)try{window.console.timeStamp("CSI/"+h)}
catch(m){}};this.tick("start",null,g)}var a;
if(window.performance)
var d=(a=window.performance.timing)&&a.responseStart;
var l=0<d?new e(d):new e;window.jstiming={Timer:e,load:l};
if(a){var    b=a.navigationStart;0<b&&d>=b&&(window.jstiming.srt=d-
b)}
if(a){var c=window.jstiming.load;
0<b&&d>=b&&(c.tick("_wtsrt",void
0,b),c.tick("wtsrt_","_wtsrt",d),c.tick("tbsd_","wtsrt_"))}
try{a=null,window.chrome&&window.chrome.csi&&(a=Math.floor(window.
chrome.csi().pageT),c&&0<b&&(c.tick("_tbnd",void
0,window.chrome.csi().startE),c.tick("tbnd_","_tbnd",b))),null==a&
&window.gtbExternal&&(a=window.gtbExternal.pageT()),null==a&&windo
w.external&&(a=window.external.pageT,c&&0<b&&(c.tick("_tbnd",void
0,window.external.startE),c.tick("tbnd_","_tbnd",b))),a&&(window.j
stiming.pt=a)}
catch(g){}})();}).call(window);
</script>
<script
src="https://translate.googleusercontent.com/translate/releases/tw
sfe_w_20190520_RC03/r/js/translate_c.js">
</script>
<meta http-equiv="X-Translated-By" content="Google">
```

```
 <linkhref=https://leafletjs.com/examples/quick-start/example.html
hreflang=en rel="alternate machine-translated-from">
 <base                    href=https://leafletjs.com/examples/quick-
start/example.html />
  <title>Позиція вантажу</title>
 <meta         http-equiv="Content-Type"         content="text/html;
charset=windows-1251">
 <meta    name=viewport    content="width=device-width,    initial-
scale=1.0">
 <link                                                rel=stylesheet
href=https://unpkg.com/leaflet@1.5.1/dist/leaflet.css
integrity=sha512-
xwE/Az9zrjBIphAcBb3F6JVqxf46+CDLwfLMHloNu6KEQCAWi6HcDUbeOfBIptF7tc
CzusKFjFw2yuvEpDL9wQ== crossorigin="" />
 <script        src=https://unpkg.com/leaflet@1.5.1/dist/leaflet.js
integrity=sha512-
GffPMF3RvMeYyc1LWMHtK8EbPv0iNZ8/oTtHPx9/cc2ILxQ+u905qIwdpULaqDkyBK
gOaB57QTMg7ztg8Jm2Og== crossorigin="">
                 Отримання данних з карти пам’яті
<?php
     $type     =0;
     $id  =0;
     $date     =0;
     $time     =0;
     $latitude      =49.55589;
     $lontitide     =25.60556;
     $temp     =0;
     $hum      =0;
     $shake    =0;
     $file = 'track_new.txt';
     if (file_exists($file)){
     $current = file_get_contents($file);
     list($date,   $time,   $latitude,   $lontitide,   $temp,   $hum,
$shake) = explode("&", $current);
     if($shake=="TRUE") $text_val="встряска";
     else $text_val="немає";
     } else {$time="Файлу даних немає!";}
     ?>
     </script>
 </head>
 <body>
 <div id=mapid style="width: 600px; height: 600px;"></div>
 <script>
    var lat ='<?php echo $latitude;?>';
     var lon   ='<?php echo $lontitide;?>';
     var mymap = L.map('mapid').setView([lat, lon], 13);
     L.tileLayer('https://api.tiles.mapbox.com/v4/{id}/{z}/{x}/{y}
.png?access_token=pk.eyJ1IjoibWFwYm94IiwiYSI6ImNpejY4NXVycTA2emYyc
XBndHRqcmZ3N3gifQ.rJcFIG214AriISLbB6B5aw', {
maxZoom: 18,
attribution:'Mapdata&copy;
 <a         href="https://www.openstreetmap.org/">OpenStreetMap</a>
contributors, ' +
```

```
                '<a   href="https://creativecommons.org/licenses/by-
sa/2.0/">CC-BY-SA</a>, ' +
                'Imagery                    ©                    <a
href="https://www.mapbox.com/">Mapbox</a>',
            id: 'mapbox.streets'
        }).addTo(mymap);
        L.marker([lat, lon], {icon: new L.Icon({
           iconUrl:       'http://gps.lab-101.org.ua/gps/marker1.png',
//https://latitude.to/img/gmapz/pin-default.png
            iconSize: [74, 74],
           iconAnchor: [32, 64],
           popupAnchor: [1, -66]
         })
       }).addTo(mymap)
         .bindPopup("<b>'<?php    echo    $date."    ".$time;?>'</b><br
/>Температура: '<?php echo $temp;?>' °C <br /> Вологість: '<?php
echo $hum;?>' % <br /> Порушення умов: '<?php echo $text_val;?>'
").openPopup();  // відкритий попуп
        //L.marker([49.55589, 25.60556]).addTo(mymap)
        //   .bindPopup("<b>Hello    world!</b><br    />I    am    a
popup.").openPopup();
        //    .bindPopup('<p  class="title"><strong>Ternopil</strong>,
Ukraine</p>');  // закритий попуп
        var popup = L.popup();
        function onMapClick(e) {
            popup
                .setLatLng(e.latlng)
                .setContent("You   clicked   the   map   at   "   +
e.latlng.toString())
                .openOn(mymap);
        }
        mymap.on('click', onMapClick);
</script></body>
<script>_addload(function(){_setupIW('com');_csi('en','ru','https:
//leafletjs.com/examples/quick-
start/example.html');});</script></html>
```

Get.php – даний скрипт відповідає за надсилання Get запитів.

```
<?php
        $type     =0;
        $id  =0;
        $date     =0;
        $time     =0;
        $lat      =0;
        $lon      =0;
        $temp     =0;
        $hum      =0;
        $shake    =0;

if ( !empty( $_GET['id'] ))
{
        $id =     $_GET['id'];
        $date =   $_GET['date'];
```

```
     $time =   $_GET['time'];
     $lat =    $_GET['lat'];
     $lon =    $_GET['lon'];
     $temp =   $_GET['temp'];
     $hum =    $_GET['hum'];
     $shake =$_GET['shake'];

     echo "1)".$id."<br>";
     echo "2)".$date."<br>";
     echo "3)".$time."<br>";
     echo "4)".$lat."<br>";
     echo "5)".$lon."<br>";
     echo "6)".$temp."<br>";
     echo "7)".$hum."<br>";
     echo "8)".$shake."<br>";
//gps.lab101.org.ua/gps/get.php?type=5&id=5&date=12.05&time=12:55&
lat=12.457&lon=30.777&temp=25.5&hum=80&shake=false
}
//$_GET=array();
//$_POST=array();
// запис даних для відслідковування маршуту google earth
$file = 'data.txt';
$current = file_get_contents($file);
$current .= "\n";
$current .= $id;  $current .= ",";  $current .= $date;  $current
.= ",";
$current .= $time;  $current .= ",";  $current .= $lat;
$current .= ",";  $current .= $lon;
file_put_contents($file, $current);
// запис даних для маркера
$current="";
$file = 'track_new.txt';
$current .= $date;
$current .= "&";
$current .= $time;
$current .= "&";
$current .= $lat;
$current .= "&";
$current .= $lon;
$current .= "&";
$current .= $temp;
$current .= "&";
$current .= $hum;
$current .= "&";
$current .= $shake;
file_put_contents($file, $current);
?>
```

Додаток Е

Лістинг коду бібліотек

Датчик вібрації

```
int sense_Pin = 0;
int value = 0;
void setup() {
  Serial.begin (9600);
}

void loop() {
  Serial.print("MOISTURE LEVEL : ");
   value= analogRead(sense_Pin);
   value= value/10;
   Serial.println(value);

}
```

Tiny gps

```
/*
TinyGPS - a small GPS library for Arduino providing basic NMEA
parsing
Based on work by and "distance_to" and "course_to" courtesy of
Maarten Lamers.
Suggestion to add satellites(), course_to(), and cardinal(), by
Matt Monson.
Precision improvements suggested by Wayne Holder.
Copyright (C) 2008-2013 Mikal Hart
All rights reserved.

This library is free software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free
Software
Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-
1301  USA
*/

#include "TinyGPS.h"

#define _GPRMC_TERM   "GPRMC"
#define _GPGGA_TERM   "GPGGA"

TinyGPS::TinyGPS()
```

```
:  _time(GPS_INVALID_TIME)
  ,  _date(GPS_INVALID_DATE)
  ,  _latitude(GPS_INVALID_ANGLE)
  ,  _longitude(GPS_INVALID_ANGLE)
  ,  _altitude(GPS_INVALID_ALTITUDE)
  ,  _speed(GPS_INVALID_SPEED)
  ,  _course(GPS_INVALID_ANGLE)
  ,  _hdop(GPS_INVALID_HDOP)
  ,  _numsats(GPS_INVALID_SATELLITES)
  ,  _last_time_fix(GPS_INVALID_FIX_TIME)
  ,  _last_position_fix(GPS_INVALID_FIX_TIME)
  ,  _parity(0)
  ,  _is_checksum_term(false)
  ,  _sentence_type(_GPS_SENTENCE_OTHER)
  ,  _term_number(0)
  ,  _term_offset(0)
  ,  _gps_data_good(false)
#ifndef _GPS_NO_STATS
  ,  _encoded_characters(0)
  ,  _good_sentences(0)
  ,  _failed_checksum(0)
#endif
{
  _term[0] = '\0';
}

//
// public methods
//

bool TinyGPS::encode(char c)
{
  bool valid_sentence = false;

#ifndef _GPS_NO_STATS
  ++_encoded_characters;
#endif
  switch(c)
  {
  case ',': // term terminators
    _parity ^= c;
  case '\r':
  case '\n':
  case '*':
    if (_term_offset < sizeof(_term))
    {
      _term[_term_offset] = 0;
      valid_sentence = term_complete();
    }
    ++_term_number;
    _term_offset = 0;
    _is_checksum_term = c == '*';
```

```
 return valid_sentence;

  case '$': // sentence begin
    _term_number = _term_offset = 0;
    _parity = 0;
    _sentence_type = _GPS_SENTENCE_OTHER;
    _is_checksum_term = false;
    _gps_data_good = false;
    return valid_sentence;
  }

  // ordinary characters
  if (_term_offset < sizeof(_term) - 1)
    _term[_term_offset++] = c;
  if (!_is_checksum_term)
    _parity ^= c;

  return valid_sentence;
}

#ifndef _GPS_NO_STATS
void   TinyGPS::stats(unsigned   long   *chars,   unsigned   short
*sentences, unsigned short *failed_cs)
{
  if (chars) *chars = _encoded_characters;
  if (sentences) *sentences = _good_sentences;
  if (failed_cs) *failed_cs = _failed_checksum;
}
#endif

//
// internal utilities
//
int TinyGPS::from_hex(char a)
{
  if (a >= 'A' && a <= 'F')
    return a - 'A' + 10;
  else if (a >= 'a' && a <= 'f')
    return a - 'a' + 10;
  else
    return a - '0';
}

unsigned long TinyGPS::parse_decimal()
{
  char *p = _term;
  bool isneg = *p == '-';
  if (isneg) ++p;
  unsigned long ret = 100UL * gpsatol(p);
  while (gpsisdigit(*p)) ++p;
  if (*p == '.')
  {
```

```
 if (gpsisdigit(p[1]))
    {
      ret += 10 * (p[1] - '0');
      if (gpsisdigit(p[2]))
        ret += p[2] - '0';
    }
  }
  return isneg ? -ret : ret;
}

// Parse a string in the form ddmm.mmmmmmm...
unsigned long TinyGPS::parse_degrees()
{
  char *p;
  unsigned long left_of_decimal = gpsatol(_term);
  unsigned long hundred1000ths_of_minute = (left_of_decimal %
100UL) * 100000UL;
  for (p=_term; gpsisdigit(*p); ++p);
  if (*p == '.')
  {
    unsigned long mult = 10000;
    while (gpsisdigit(*++p))
    {
      hundred1000ths_of_minute += mult * (*p - '0');
      mult /= 10;
    }
  }
  return (left_of_decimal / 100) * 1000000 +
(hundred1000ths_of_minute + 3) / 6;
}

#define COMBINE(sentence_type, term_number)
(((unsigned)(sentence_type) << 5) | term_number)

// Processes a just-completed term
// Returns true if new sentence has just passed checksum test and
is validated
bool TinyGPS::term_complete()
{
  if (_is_checksum_term)
  {
    byte checksum = 16 * from_hex(_term[0]) + from_hex(_term[1]);
    if (checksum == _parity)
    {
      if (_gps_data_good)
      {
#ifndef _GPS_NO_STATS
        ++_good_sentences;
#endif
        _last_time_fix = _new_time_fix;
        _last_position_fix = _new_position_fix;
```

```
 switch(_sentence_type)
        {
        case _GPS_SENTENCE_GPRMC:
          _time      = _new_time;
          _date      = _new_date;
          _latitude  = _new_latitude;
          _longitude = _new_longitude;
          _speed     = _new_speed;
          _course    = _new_course;
          break;
        case _GPS_SENTENCE_GPGGA:
          _altitude  = _new_altitude;
          _time      = _new_time;
          _latitude  = _new_latitude;
          _longitude = _new_longitude;
          _numsats   = _new_numsats;
          _hdop      = _new_hdop;
          break;
        }

        return true;
      }
    }

#ifndef _GPS_NO_STATS
    else
      ++_failed_checksum;
#endif
    return false;
  }

  // the first term determines the sentence type
  if (_term_number == 0)
  {
    if (!gpsstrcmp(_term, _GPRMC_TERM))
      _sentence_type = _GPS_SENTENCE_GPRMC;
    else if (!gpsstrcmp(_term, _GPGGA_TERM))
      _sentence_type = _GPS_SENTENCE_GPGGA;
    else
      _sentence_type = _GPS_SENTENCE_OTHER;
    return false;
  }

  if (_sentence_type != _GPS_SENTENCE_OTHER && _term[0])
    switch(COMBINE(_sentence_type, _term_number))
  {
    case  COMBINE(_GPS_SENTENCE_GPRMC,  1):  //  Time  in  both
sentences
    case COMBINE(_GPS_SENTENCE_GPGGA, 1):
      _new_time = parse_decimal();
      _new_time_fix = millis();
      break;
```

```
case COMBINE(_GPS_SENTENCE_GPRMC, 2): // GPRMC validity
      _gps_data_good = _term[0] == 'A';
      break;
    case COMBINE(_GPS_SENTENCE_GPRMC, 3): // Latitude
    case COMBINE(_GPS_SENTENCE_GPGGA, 2):
      _new_latitude = parse_degrees();
      _new_position_fix = millis();
      break;
    case COMBINE(_GPS_SENTENCE_GPRMC, 4): // N/S
    case COMBINE(_GPS_SENTENCE_GPGGA, 3):
      if (_term[0] == 'S')
        _new_latitude = -_new_latitude;
      break;
    case COMBINE(_GPS_SENTENCE_GPRMC, 5): // Longitude
    case COMBINE(_GPS_SENTENCE_GPGGA, 4):
      _new_longitude = parse_degrees();
      break;
    case COMBINE(_GPS_SENTENCE_GPRMC, 6): // E/W
    case COMBINE(_GPS_SENTENCE_GPGGA, 5):
      if (_term[0] == 'W')
        _new_longitude = -_new_longitude;
      break;
    case COMBINE(_GPS_SENTENCE_GPRMC, 7): // Speed (GPRMC)
      _new_speed = parse_decimal();
      break;
    case COMBINE(_GPS_SENTENCE_GPRMC, 8): // Course (GPRMC)
      _new_course = parse_decimal();
      break;
    case COMBINE(_GPS_SENTENCE_GPRMC, 9): // Date (GPRMC)
      _new_date = gpsatol(_term);
      break;
    case COMBINE(_GPS_SENTENCE_GPGGA, 6): // Fix data (GPGGA)
      _gps_data_good = _term[0] > '0';
      break;
    case  COMBINE(_GPS_SENTENCE_GPGGA,  7):  //  Satellites  used
(GPGGA)
      _new_numsats = (unsigned char)atoi(_term);
      break;
    case COMBINE(_GPS_SENTENCE_GPGGA, 8): // HDOP
      _new_hdop = parse_decimal();
      break;
    case COMBINE(_GPS_SENTENCE_GPGGA, 9): // Altitude (GPGGA)
      _new_altitude = parse_decimal();
      break;
  }

  return false;
}

long TinyGPS::gpsatol(const char *str)
{
  long ret = 0;
```

```
while (gpsisdigit(*str))
    ret = 10 * ret + *str++ - '0';
  return ret;
}

int TinyGPS::gpsstrcmp(const char *str1, const char *str2)
{
  while (*str1 && *str1 == *str2)
    ++str1, ++str2;
  return *str1;
}

/* static */
float TinyGPS::distance_between (float lat1, float long1, float
lat2, float long2)
{
  // returns distance in meters between two positions, both
specified
  // as signed decimal-degrees latitude and longitude. Uses great-
circle
  // distance computation for hypothetical sphere of radius
6372795 meters.
  // Because Earth is no exact sphere, rounding errors may be up
to 0.5%.
  // Courtesy of Maarten Lamers
  float delta = radians(long1-long2);
  float sdlong = sin(delta);
  float cdlong = cos(delta);
  lat1 = radians(lat1);
  lat2 = radians(lat2);
  float slat1 = sin(lat1);
  float clat1 = cos(lat1);
  float slat2 = sin(lat2);
  float clat2 = cos(lat2);
  delta = (clat1 * slat2) - (slat1 * clat2 * cdlong);
  delta = sq(delta);
  delta += sq(clat2 * sdlong);
  delta = sqrt(delta);
  float denom = (slat1 * slat2) + (clat1 * clat2 * cdlong);
  delta = atan2(delta, denom);
  return delta * 6372795;
}

float TinyGPS::course_to (float lat1, float long1, float lat2,
float long2)
{
  // returns course in degrees (North=0, West=270) from position 1
to position 2,
  // both specified as signed decimal-degrees latitude and
longitude.
  // Because Earth is no exact sphere, calculated course may be
off by a tiny fraction.
```

```
// Courtesy of Maarten Lamers
  float dlon = radians(long2-long1);
  lat1 = radians(lat1);
  lat2 = radians(lat2);
  float a1 = sin(dlon) * cos(lat2);
  float a2 = sin(lat1) * cos(lat2) * cos(dlon);
  a2 = cos(lat1) * sin(lat2) - a2;
  a2 = atan2(a1, a2);
  if (a2 < 0.0)
  {
    a2 += TWO_PI;
  }
  return degrees(a2);
}

const char *TinyGPS::cardinal (float course)
{
  static const char* directions[] = {"N", "NNE", "NE", "ENE", "E",
"ESE", "SE", "SSE", "S", "SSW", "SW", "WSW", "W", "WNW", "NW",
"NNW"};

  int direction = (int)((course + 11.25f) / 22.5f);
  return directions[direction % 16];
}

// lat/long in MILLIONTHs of a degree and age of fix in
milliseconds
// (note: versions 12 and earlier gave this value in 100,000ths of
a degree.
void TinyGPS::get_position(long *latitude, long *longitude,
unsigned long *fix_age)
{
  if (latitude) *latitude = _latitude;
  if (longitude) *longitude = _longitude;
  if (fix_age) *fix_age = _last_position_fix ==
GPS_INVALID_FIX_TIME ?
   GPS_INVALID_AGE : millis() - _last_position_fix;
}

// date as ddmmyy, time as hhmmsscc, and age in milliseconds
void TinyGPS::get_datetime(unsigned long *date, unsigned long
*time, unsigned long *age)
{
  if (date) *date = _date;
  if (time) *time = _time;
  if (age) *age = _last_time_fix == GPS_INVALID_FIX_TIME ?
   GPS_INVALID_AGE : millis() - _last_time_fix;
}

void TinyGPS::f_get_position(float *latitude, float *longitude,
unsigned long *fix_age)
{
```

```
long lat, lon;
  get_position(&lat, &lon, fix_age);
  *latitude = lat == GPS_INVALID_ANGLE ? GPS_INVALID_F_ANGLE :
(lat / 1000000.0);
  *longitude = lat == GPS_INVALID_ANGLE ? GPS_INVALID_F_ANGLE :
(lon / 1000000.0);
}

void TinyGPS::crack_datetime(int *year, byte *month, byte *day,
  byte *hour, byte *minute, byte *second, byte *hundredths,
unsigned long *age)
{
  unsigned long date, time;
  get_datetime(&date, &time, age);
  if (year)
  {
    *year = date % 100;
    *year += *year > 80 ? 1900 : 2000;
  }
  if (month) *month = (date / 100) % 100;
  if (day) *day = date / 10000;
  if (hour) *hour = time / 1000000;
  if (minute) *minute = (time / 10000) % 100;
  if (second) *second = (time / 100) % 100;
  if (hundredths) *hundredths = time % 100;
}

float TinyGPS::f_altitude()
{
  return _altitude == GPS_INVALID_ALTITUDE ?
GPS_INVALID_F_ALTITUDE : _altitude / 100.0;
}

float TinyGPS::f_course()
{
  return _course == GPS_INVALID_ANGLE ? GPS_INVALID_F_ANGLE :
_course / 100.0;
}

float TinyGPS::f_speed_knots()
{
  return _speed == GPS_INVALID_SPEED ? GPS_INVALID_F_SPEED :
_speed / 100.0;
}

float TinyGPS::f_speed_mph()
{
  float sk = f_speed_knots();
  return sk == GPS_INVALID_F_SPEED ? GPS_INVALID_F_SPEED :
_GPS_MPH_PER_KNOT * sk;
}
```

```
float TinyGPS::f_speed_mps()
{
  float sk = f_speed_knots();
  return  sk  ==  GPS_INVALID_F_SPEED  ?  GPS_INVALID_F_SPEED  :
_GPS_MPS_PER_KNOT * sk;
}

float TinyGPS::f_speed_kmph()
{
  float sk = f_speed_knots();
  return  sk  ==  GPS_INVALID_F_SPEED  ?  GPS_INVALID_F_SPEED  :
_GPS_KMPH_PER_KNOT * sk;
}

const float TinyGPS::GPS_INVALID_F_ANGLE = 1000.0;
const float TinyGPS::GPS_INVALID_F_ALTITUDE = 1000000.0;
const float TinyGPS::GPS_INVALID_F_SPEED = -1.0;
```

Vibration Sensor library

```
/*
* Copyright (C) 2008 The Android Open Source Project
*
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
*
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing,
software< /span>
* distributed under the License is distributed on an "AS IS"
BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
* See the License for the specific language governing permissions
and
* limitations under the License.
*/

/* Update by K. Townsend (Adafruit Industries) for lighter
typedefs, and
 * extended sensor support to include color, voltage and current
*/

#ifndef _ADAFRUIT_SENSOR_H
#define _ADAFRUIT_SENSOR_H

#ifndef ARDUINO
 #include <stdint.h>
#elif ARDUINO >= 100
 #include "Arduino.h"
 #include "Print.h"
```

```
#else
 #include "WProgram.h"
#endif

/* Intentionally modeled after sensors.h in the Android API:
 *
https://github.com/android/platform_hardware_libhardware/blob/mast
er/include/hardware/sensors.h */

/* Constants */
#define  SENSORS_GRAVITY_EARTH                             (9.80665F)
/**< Earth's gravity in m/s^2 */
#define  SENSORS_GRAVITY_MOON                               (1.6F)
/**< The moon's gravity in m/s^2 */
#define  SENSORS_GRAVITY_SUN                              (275.0F)
/**< The sun's gravity in m/s^2 */
#define SENSORS_GRAVITY_STANDARD          (SENSORS_GRAVITY_EARTH)
#define   SENSORS_MAGFIELD_EARTH_MAX                       (60.0F)
/**< Maximum magnetic field on Earth's surface */
#define   SENSORS_MAGFIELD_EARTH_MIN                       (30.0F)
/**< Minimum magnetic field on Earth's surface */
#define   SENSORS_PRESSURE_SEALEVELHPA                   (1013.25F)
/**< Average sea level pressure is 1013.25 hPa */
#define  SENSORS_DPS_TO_RADS                          (0.017453293F)
/**< Degrees/s to rad/s multiplier */
#define   SENSORS_GAUSS_TO_MICROTESLA                       (100)
/**< Gauss to micro-Tesla multiplier */

/** Sensor types */
typedef enum
{
  SENSOR_TYPE_ACCELEROMETER         = (1),   /**< Gravity + linear
acceleration */
  SENSOR_TYPE_MAGNETIC_FIELD        = (2),
  SENSOR_TYPE_ORIENTATION           = (3),
  SENSOR_TYPE_GYROSCOPE             = (4),
  SENSOR_TYPE_LIGHT                 = (5),
  SENSOR_TYPE_PRESSURE              = (6),
  SENSOR_TYPE_PROXIMITY             = (8),
  SENSOR_TYPE_GRAVITY               = (9),
  SENSOR_TYPE_LINEAR_ACCELERATION   = (10),  /**< Acceleration not
including gravity */
  SENSOR_TYPE_ROTATION_VECTOR       = (11),
  SENSOR_TYPE_RELATIVE_HUMIDITY     = (12),
  SENSOR_TYPE_AMBIENT_TEMPERATURE   = (13),
  SENSOR_TYPE_VOLTAGE               = (15),
  SENSOR_TYPE_CURRENT               = (16),
  SENSOR_TYPE_COLOR                 = (17)
} sensors_type_t;

/** struct sensors_vec_s is used to return a vector in a common
format. */
```

```
typedef struct {
    union {
        float v[3];
        struct {
            float x;
            float y;
            float z;
        };
        /* Orientation sensors */
        struct {
            float roll;     /**< Rotation around the longitudinal
axis (the plane body, 'X axis'). Roll is positive and increasing
when moving downward. -90°<=roll<=90° */
            float pitch;    /**< Rotation around the lateral axis
(the wing span, 'Y axis'). Pitch is positive and increasing when
moving upwards. -180°<=pitch<=180°) */
            float heading; /**< Angle between the longitudinal
axis (the plane body) and magnetic north, measured clockwise when
viewing from the top of the device. 0-359° */
        };
    };
    int8_t status;
    uint8_t reserved[3];
} sensors_vec_t;

/** struct sensors_color_s is used to return color data in a
common format. */
typedef struct {
    union {
        float c[3];
        /* RGB color space */
        struct {
            float r;       /**< Red component */
            float g;       /**< Green component */
            float b;       /**< Blue component */
        };
    };
    uint32_t rgba;         /**< 24-bit RGBA value */
} sensors_color_t;

/* Sensor event (36 bytes) */
/** struct sensor_event_s is used to provide a single sensor event
in a common format. */
typedef struct
{
    int32_t version;                                   /**< must be
sizeof(struct sensors_event_t) */
    int32_t sensor_id;                               /**< unique sensor
identifier */
    int32_t type;                                  /**< sensor type */
    int32_t reserved0;                             /**< reserved */
```

```
 int32_t timestamp;                                  /**< time is in
milliseconds */
    union
    {
        float           data[4];
        sensors_vec_t    acceleration;            /**< acceleration
values are in meter per second per second (m/s^2) */
        sensors_vec_t   magnetic;               /**< magnetic vector
values are in micro-Tesla (uT) */
        sensors_vec_t    orientation;             /**< orientation
values are in degrees */
        sensors_vec_t    gyro;                      /**< gyroscope
values are in rad/s */
        float            temperature;           /**< temperature is
in degrees centigrade (Celsius) */
        float             distance;               /**< distance in
centimeters */
        float           light;                  /**< light in SI lux
units */
        float             pressure;               /**< pressure in
hectopascal (hPa) */
        float                relative_humidity;      /**< relative
humidity in percent */
        float             current;                /**< current in
milliamps (mA) */
        float             voltage;                /**< voltage in
volts (V) */
        sensors_color_t color;                    /**< color in RGB
component values */
    };
} sensors_event_t;

/* Sensor details (40 bytes) */
/** struct sensor_s is used to describe basic information about a
specific sensor. */
typedef struct
{
    char     name[12];                           /**< sensor name */
    int32_t  version;                             /**< version of the
hardware + driver */
    int32_t  sensor_id;                           /**< unique sensor
identifier */
    int32_t  type;                                /**< this sensor's
type (ex. SENSOR_TYPE_LIGHT) */
    float    max_value;                           /**< maximum value
of this sensor's value in SI units */
    float    min_value;                           /**< minimum value
of this sensor's value in SI units */
    float     resolution;                             /**< smallest
difference between two values reported by this sensor */
    int32_t  min_delay;                           /**< min delay in
microseconds between events. zero = not a constant rate */
```

```
} sensor_t;

class Adafruit_Sensor {
 public:
  // Constructor(s)
  Adafruit_Sensor() {}
  virtual ~Adafruit_Sensor() {}

  // These must be defined by the subclass
  virtual void enableAutoRange(bool enabled) { (void)enabled; /*
suppress unused warning */ };
  virtual bool getEvent(sensors_event_t*) = 0;
  virtual void getSensor(sensor_t*) = 0;

 private:
  bool _autoRange;
};

#endif
```

DHT sensor lib

```
/* DHT library

MIT license
written by Adafruit Industries
*/
#ifndef DHT_H
#define DHT_H

#if ARDUINO >= 100
 #include "Arduino.h"
#else
 #include "WProgram.h"
#endif


// Uncomment to enable printing out nice debug messages.
//#define DHT_DEBUG

// Define where debug output will be printed.
#define DEBUG_PRINTER Serial

// Setup debug printing macros.
#ifdef DHT_DEBUG
  #define DEBUG_PRINT(...) { DEBUG_PRINTER.print(__VA_ARGS__); }
  #define DEBUG_PRINTLN(...) { DEBUG_PRINTER.println(__VA_ARGS__);
}
#else
  #define DEBUG_PRINT(...) {}
  #define DEBUG_PRINTLN(...) {}
#endif

// Define types of sensors.
```

```
#define DHT11 11
#define DHT22 22
#define DHT21 21
#define AM2301 21


class DHT {
  public:
   DHT(uint8_t pin, uint8_t type, uint8_t count=6);
   void begin(void);
   float readTemperature(bool S=false, bool force=false);
   float convertCtoF(float);
   float convertFtoC(float);
   float       computeHeatIndex(float       temperature,       float
percentHumidity, bool isFahrenheit=true);
   float readHumidity(bool force=false);
   boolean read(bool force=false);

 private:
  uint8_t data[5];
  uint8_t _pin, _type;
  #ifdef __AVR
    // Use direct GPIO access on an 8-bit AVR so keep track of the
port and bitmask
    // for the digital pin connected to the DHT.  Other platforms
will use digitalRead.
    uint8_t _bit, _port;
  #endif
  uint32_t _lastreadtime, _maxcycles;
  bool _lastresult;

  uint32_t expectPulse(bool level);

};

class InterruptLock {
  public:
   InterruptLock() {
    noInterrupts();
   }
   ~InterruptLock() {
    interrupts();
   }

};

#endif
```