

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, І ТЕРМІНІВ.....	8
ВСТУП.....	11
1 АНАЛІЗ ФУНКЦІОНАЛЬНИХ ВИМОГ ТА ТЕХНОЛОГІЙ РЕАЛІЗАЦІЇ СИСТЕМИ І ПОСТАНОВКА ЗАВДАННЯ.....	13
1.1 Дослідження предмета, цілей і особливостей розробки системи управління діяльністю хлібопекарського підприємства.....	13
1.2 Огляд існуючих систем управління та технологій їх реалізації.....	14
1.3. Класифікація контролерів.....	15
1.4 Вибір мікроконтролера.....	18
1.5 Мова програмування C/C++.....	19
1.6. Об'єктно-орієнтоване програмування в C++.....	22
1.7. Особливості мови програмування C++ в середовищі Arduino IDE.....	22
1.8. Постановка завдання.....	23
2 ВИБІР АРХІТЕКТУРНИХ РІШЕНЬ ТА ПРОГРАМНИХ ЗАСОБІВ РОЗРОБКИ СИСТЕМИ.....	25
2.1 Визначення завдань прийняття рішень вибору технології в умовах багатокритеріальності.....	25
2.2 Теоретичні відомості про апаратні особливості реалізації.....	26
2.3 Особливості будови та застосування Arduino.....	30
2.4 Середовище розробки програмного забезпечення.....	31
3 АНАЛІЗ ТА МОДЕЛЮВАННЯ МАЙБУТНЬОЇ СИСТЕМИ МЕТОДОМ ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ.....	33
3.1 Загальна інформація.....	33
3.2 Контроль динаміки зміни маси вулика.....	34
3.3 Визначення внутрішньої вологості повітря.....	34
3.4 Визначення внутрішньої температури повітря.....	35
3.5 Фіксація руху поряд на території бджільництва.....	36
3.6 Формування звіту для власника.....	37
4 ПРОЕКТУВАННЯ ТА ПРАКТИЧНА РЕАЛІЗАЦІЯ СИСТЕМИ УПРАВЛІННЯ ТА МОНІТОРИНГУ ЗА ПАСІКОЮ.....	38
4.1. Проектування та реалізація підключення тензодатчиків.....	38
4.2. Реалізація моніторингу за вологістю та температурою.....	40
4.3 Підключення LCD дисплею для виводу інформації.....	42
4.4 Реалізація системи охорони та сповіщення.....	43
4.4 Реалізація системи стільникового зв'язку.....	45
4.5 Прототип системи BeeFarm.....	46
4.6 Виявлення недоліків системи BeeFarm.....	47

ВИСНОВКИ.....	49
ПЕРЕЛІК ПОСИЛАНЬ.....	51

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, І ТЕРМІНІВ

Скорочення

АЗ	– апаратне забезпечення;
БД	– база даних;
ПЗ	– програмне забезпечення;
ЦП	– центральний процесор;
ООП	– об’єктно орієнтоване програмування;
ОПР	– особа що приймає рішення;
ОС	– операційна система;
ПК	– персональний комп’ютер;
СУБД	– система управління базами даних.
CEO	– Chief Executive Officer (головний виконавчий директор);
DOM	– Document Object Model (програмний інтерфейс для доступу до документів);
GSM	– Global System for Mobile Communications (глобальна система мобільного зв’язку);
IDE	– Integrated Development Environment (інтегроване середовище розробки);
LCD	– liquid crystal display (рідкокристалічний дисплей);
LED	– light-emitting diode (світлодіод);
MIT	– ліцензія Массачусетського технологічного інституту;
MVC	– Model-View-Controller модель-представлення-контролер);
MySQL	– вільна реляційна система управління базами даних;
HamI	– Hyper abstraction markup language (мова розмітки для спрощеної генерації HTML);
JS	– JavaScript (язык програмування);
JSON	– JavaScript Object Notation (текстовий формат обміну даними);
SPA	– односторінкові додатки (single page application);
SVG	– Scalable Vector Graphics (масштабуєма векторна графіка);
URL	– Uniform Resource Locator (визначник місцезнаходження ресурсу).

Терміни

Принципова схема – схема, що визначає повний склад елементів і зв’язків між ними і, як правило, дає детальне уявлення про принципи роботи виробу (установки);

Модуль – уніфікований функціональний вузол, виконаний у вигляді самостійного виробу, установки тощо.

Послідовний порт – комунікаційний інтерфейс, через який дані передаються послідовно, по одному біту за такт.

Інкапсуляція – це принцип, згідно з яким будь-який клас повинен розглядатися як чорний ящик - користувач класу повинен бачити і використовувати тільки інтерфейсну частину класу (т. Е. Список декларованих властивостей і методів класу) і не вникати в його внутрішню реалізацію;

Інтерфейс – призначений для взаємодії класу з іншою програмою;

Спадкування – це породження нового класу-нащадка від вже існуючого класу-батька. Клас-батько називають також супер-класом, а клас-нащадок – підкласом. Спадкування відбувається з передачею всіх або деяких полів і методів від класу-батька до класу-нащадка;

Об'єкт – сутність володіє певним поведінкою і способом подання, т. Е. Відноситься до класу (кажуть, що об'єкт – це екземпляр класу, якщо він до нього ставиться);

Веб-додаток – клієнт-серверний додаток, в якому клієнт взаємодіє з сервером за допомогою браузера, а за сервер відповідає – веб-сервер;

Поліморфізм – це явище, при якому функції (методу) з одним і тим же ім'ям відповідає різний програмний код (поліморфний код) в залежності від того, в якому контексті він викликається (об'єктами якого класу або з якими параметрами);

Абстракція даних – це виділення істотних характеристик об'єкта, які відрізняють його від інших об'єктів, чітко визначаючи його межі;

Arduino (Ардуіно) – апаратна обчислювальна платформа для аматорського конструювання, основними компонентами якої є плата мікроконтролера з елементами вводу/виводу та середовище розробки (Вікіпедія);

C++ (Сі-плюс-плюс) – мова програмування високого;

JavaScript Object Notation – форма представлення даних, в який самі дані представляються у вигляді об'єкта мови JavaScript, виду «Ключ»:«значення»;

Node.js – програмна платформа, заснована на движку V8 (здійснює трансляцію JavaScript в машинний код), що перетворює JavaScript з вузькоспеціалізованою мови в мову загального призначення;

TypeScript – мова програмування, представлена Microsoft в 2012 році і позиціонується як засіб розробки веб-додатків, що розширює можливості JavaScript;

Full-stack – це фахівець, який здатний брати активну участь у всіх етапах розробки веб-додатків, починаючи від серверної логіки і її реалізації за допомогою різних технологій і фреймворків, і закінчуючи клієнтським кодом, які працюють безпосередньо в браузері;

Open-source – програмне забезпечення, користувачі якого мають права («свободи») на його необмежену установку, запуск, вільне використання, вивчення, поширення і зміна (вдосконалення), а також поширення копій і результатів зміни;

ВСТУП

Важливою складовою технологічного розвитку сьогодні є не тільки винайдення нових засобів та технологій в цілому. Це впершу чергу вдосконалення способів використання вже існуючих технологій та засобів, а також їх інтеграція в усі сфери людської діяльності. Економія часу та зусиль людини – є головною ціллю автоматизації (або механізації) будь-якого процесу.

За словами одного відомого професора Стенфордського університету Дональда Ервіна Кнута, винахідника декількох програмних технологій та ідеолога цієї галузі – «Найкращий спосіб розібратись в чомусь до кінця – це навчитись цьому комп'ютер».

Так, об'єктом моїх досліджень стало сільське господарство. А саме – така непроста галузь як бджільництво. «Навчитись» комп'ютер цієї непростой дисципліни – задача не з простих і потребує в першу чергу здобути глибокі базові знання про особливості даного роду діяльності.

Цей вид сільського господарства особливий тим – що потребує постійного нагляду та контролю зі сторони великої кількості співробітників, які вимагатимуть вищої оплати праці тільки через сам факт бути укушеним однією (або декількома) із комах. Це, як правило, дуже дорого для власників подібних підприємств.

Через це більшість із них, а особливо виїзні пасіки, які потребують ще й охорони – під постійною небезпекою, через неможливість отримувати об'єктивну інформацію про діяльність бджільництва. Звичайно ж бджоли дуже організовані комахи, цикли праці яких – чіткі та послідовні. Це дає в теорії людині можливість прогнозувати їх поведінку.

Стає очевидно – це відкриває безліч можливостей та мінімізацію будь-яких ризиків (окрім непередбачуваних випадків). Але, звичайно ж, можливості людини достить обмежені – контролювати кожен вулик (яких може налічуватись до 500 на одну пасіку!) задача близька до неможливої.

Інформаційні технології вже давно міцно ввійшли в наше життя і в багатьох сферах вже стали незамінними помічниками нам, людям. Більшість апаратних рішень та програмного забезпечення доступні у відкритому вигляді, ціни на радіодеталі та обладнання з кожним днем стають все нижчими – цьому сприяє постійний технологічний прогрес.

Враховуючи ці фактори – можна сміливо стверджувати, що зараз саме час зараз запроваджувати подібні системи в дрібне фермерське та сільське

господарство, адже зараз всі умови як ніколи раніше сприятливі – як для розробників подібний рішень, так і для цільової аудиторії.

Я вирішив розробити програмно-апаратний комплекс, який носитиме назву «BeeFarm». Так званім серцем цієї системи буде платформа Arduino, яка як ніяка інша підійде для наших потреб. В результаті готовий продукт буде компактным та головне не дорогим засобом контролю, моніторингу, аналізу та відслідковування динаміки. Призваний стати незамінним помічником будь-якого бджоляра.

1 АНАЛІЗ ФУНКЦІОНАЛЬНИХ ВИМОГ ТА ТЕХНОЛОГІЙ РЕАЛІЗАЦІЇ СИСТЕМИ І ПОСТАНОВКА ЗАВДАННЯ

Ціллю даного проекту буде розробка системи управління та моніторингу з використанням передових технологій, зрушити з мертвого місця інтеграцію інформаційних технологій для таких потенційних користувачів, як фермери та люди які займаються дрібним сільським господарством.

Дана система покликана зацікавити дану аудиторію та змусити їх повірити в те, що сучасні технології можуть спростити їхнє життя та зекономити час при невеликих капіталовкладеннях.

Проаналізувати ринок на наявність подібних пропозицій, з'ясувати які основні елементи необхідні для повноцінної роботи системи та вибрати ту технологію, котра задовольнить всі майбутні вимоги.

Здійснити вибір платформи для реалізації управління за сукупністю декількох показників з урахуванням обмежень, що відображають індивідуальні переваги та дають більш наглядне та статистичне уявлення про те що відбувається на вашій ділянці.

Будемо створювати повноцінний апаратну систему моніторингу та контролю на базі мікропроцесора. Нам знадобиться безліч радіо-деталей, платформа для реалізації досить складної системи та взаємодії із всіма необхідними нам елементами.

1.1 Дослідження предмета, цілей і особливостей розробки системи управління діяльністю хлібопекарського підприємства

Важливою складовою технологічного розвитку сьогодні є не тільки винайдення нових засобів та технологій в цілому. Це впершу чергу вдосконалення використання вже існуючих технологій а також їх інтеграція в усі сфери людської діяльності.

Так, об'єктом моїх досліджень стало сільське господарство. А саме – така непроста галузь як бджільництво.

Цей вид сільського господарства особливий тим – що потребує постійного нагляду, контролю, а виїзди пасіки – ще й охорони. Бджоли дуже організовані комахи, цикли праці яких – чіткі та послідовні. Це дає людині можливість прогнозувати їх поведінку.

Стає очевидно – це відкриває безліч можливостей та мінімізацію будь-яких ризиків (окрім непередбачуваних випадків). Але, звичайно ж, можливості

людини достить обмежені – контролювати кожен вулик (яких може налічуватись до 500 на одну пасіку!) задача близька до неможливої.

Інформаційні технології вже давно міцно ввійшли в наше життя і в багатьох сферах вже стали незамінними помічниками нам, людям. Тому, як нам мене, саме час зараз запроваджувати подібні системи в дрібне фермерське та сільське господарство.

Я вирішив розробити програмно-апаратний комплекс BeeFarm. Так званім серцем цієї системи буде платформа Arduino, яка як ніяка інша підійде для наших потреб. В результаті готовий продукт буде компактним та головне не дорогим засобом контролю, моніторингу, аналізу та відслідковування динаміки. Призваний стати незамінним помічником будь-якого бджоляра.

1.2 Огляд існуючих систем управління та технологій їх реалізації

- Arduino (Ардуіно) – апаратна обчислювальна платформа для аматорського конструювання, основними компонентами якої є плата мікроконтролера з елементами вводу/виводу та середовище розробки Processing/Wiring на мові програмування, що є підмножиною C/C++. Arduino може використовуватися як для створення автономних інтерактивних об'єктів, так і підключатися до програмного забезпечення, яке виконується на комп'ютері (наприклад: Processing, Adobe Flash, Max/MSP, Pure Data, SuperCollider). Інформація про плату (рисунок друкованої плати, специфікації елементів, програмне забезпечення) знаходяться у відкритому доступі і можуть бути використані тими, хто воліє створювати плати власноруч.

- Плата Arduino складається з мікроконтролера Atmel AVR, а також елементів обв'язки для програмування та інтеграції з іншими пристроями. На багатьох платах наявний лінійний стабілізатор напруги +5В або +3,3В. Тактування здійснюється на частоті 16 або 8 МГц кварцовим резонатором. У мікроконтролер записаний завантажувач (bootloader), тому зовнішній програматор не потрібен.

- На концептуальному рівні усі плати програмуються через RS-232 (послідовне з'єднання), але реалізація даного способу різниться від версії до версії. Новіші плати програмуються через USB, що можливо завдяки мікросхемі конвертера USB-to-Serial FTDI FT232R. У версії платформи Arduino Uno як конвертер використовується контролер Atmega8 у SMD-корпусі. Дане рішення дозволяє програмувати конвертер таким чином, щоб

платформа відразу розпізнавалася як миша, джойстик чи інший пристрій за вибором розробника зі всіма необхідними додатковими сигналами керування. У деяких варіантах, таких як Arduino Mini або неофіційній Boarduino, для програмування потрібно підключити до контролера окрему плату USB-to-Serial або кабель.

- Плати Arduino дозволяють використовувати значну кількість виводів мікроконтролера як входні/вихідні контакти у зовнішніх схемах. Наприклад, у платі Decimila доступно 14 цифрових входів/виходів, 6 із яких можуть генерувати ШІМ сигнал, і 6 аналогових входів. Ці сигнали доступні на платі через контактні площадки або штирові розніми. Також існує багато різних зовнішніх плат розширення, які називаються «shields» («щити»), які приєднуються до плати Arduino через штирові роз'єми.

1.3. Класифікація контролерів

Щодо апаратних засобів які задовільняють наші вимоги – вибір зводиться до трьох позицій. В першу чергу мова йде про Arduino, звичайно ж Raspberry PI та найскладніший спосіб це створення власної платформи замість готових комплексних рішень на базі мікро-процесора ATmega328.

Почнемо з досить популярної, надійної та дешевої платформи Arduino та її видів, особливостей (рис 1.1).



Рисунок 1.1 – Плата Arduino.

Стандартний і найпоширеніший розмір. Коли говорять «Ардуіно» («звичайна Ардуіно») - зазвичай все відразу уявляють саме такі плати.

Найперші плати були в цьому цьому форм-факторі, відповідно саме він пережив найбільше реінкорнацій (USB-версії в хронологічному порядку виходу): Extreme, NG, Diecimila, Duemilanove, Uno, Leonardo.

Ви не повірите, але відчутна різниця для користувача спостерігається тільки в Леонардо

Зараз на оф. сайті подано доступну до покупки тільки Leonardo і Uno, проте інтернет завалений варіантами Duemilanove наша CraftDuino саме її різновид і не дарма – все що потрібно середньому користувачеві було втілено ще в Ardino Extreme, з тих пір змінилося вкрай мало.

Всі ці плати мають однакову кількість входів-виходів, зібраних на однакові роз'єми (для підключення перефіїри і ШІлд), програмуються по USB, і мають мікроконтролер ATmega на борту. На ранніх версіях стояв ATmega8, потім стали ставити ATmega168, потім ATmega328.

На «вісімці» тільки 3 ШІМ виходу, 8Кб під скетч 1Кб оперативки, але для багатьох додатків вистачає У ATmega168 вже 6 ШІМ каналів і 16Кб під ваші потреби, а у 328-й 32Кб під програми і вже 2Кб оперативки. До речі не вся флеш-пам'ять доступна користувачу, частина її займає бутлоадер.

На всіх платах до UNO стояв чіп-перетворювач USB-UART FT232, що дозволяє встромляти плату прямо в USB і програмувати без програматора. При встромляє в системі з'являвся віртуальний COM-порт, який і використовується середовищем розробки Ардуіно для програмування.

Що ж до нашого другого кандидата – Raspberry Pi. Безумовно викликає куди більше симпаті у досвідчених розробників та користується великою повагою. Пропонує куди більший спектр можливостей на відміну від Arduino, яка до всього ще й обмежена своєю внутрішньою пам'яттю.

Raspberry Pi – одноплатний комп'ютер, тобто різні частини комп'ютера, які зазвичай розташовуються на окремих платах, тут представлені на одній.

Перший проект майбутньої Raspberry Pi був представлений в 2011 р, промислове виробництво почалося в 2012 р назві продукту об'єднані Raspberry – малина і Pi – число Пі. Зображення малини стало логотипом проекту.

Спочатку проект створювався як освітній, Raspberry Pi відмінно підходить для вивчення основ електроніки - все видно, є і ціна не велика. На сайті проекту є вагомий освітній розділ з докладними інструкціями і методиками. Raspberry Pi Foundation постійно акцентує увагу на навчальних

можливостях комп'ютера, в їх офіційному твіттері ви знайдете безліч юних щасливих володарів пристрої та їх досягнення.

Одноплатні комп'ютери зазвичай невеликого розміру, енергоефективні та мають відносно невелику вартість. Все це відноситься і до Raspberry Pi. Багато користувачів використовують Raspberry Pi як сервер, адже вона споживає мало енергії і абсолютно безшумно. Завдяки невеликому розміру її можна вбудувати в різні корпуси і використовувати, наприклад, як мозок для робота. Тому проект «Цікава робототехніка» і звернувся до розгляду цього пристрою. В цілому з Raspberry Pi можна зробити основну масу всього того, що ви робите на звичайному настільному комп'ютері, але з деякою специфікою, в основному пов'язаної з несумісністю з багатьма операційними системами.

Виглядає Raspberry Pi досить компактно, розмір плати схожий з кредитною картою (рис. 1.2).



Рисунок 1.2 – Одноплатний комп'ютер Raspberry Pi.

RaspberryPi – один з небагатьох продуктів у своїй ніші, основна маса інших одноплатних комп'ютерів нагадує Arduino. Серед інших одноплатних комп'ютерів Raspberry Pi виділяється і наявністю масштабного співтовариства, де ви зможете знайти відповіді на каверзні запитання і знайдете аудиторію, щоб поділитися своїми ідеями та розробками.

Сьогодні Raspberry Pi популярний не тільки в навчанні, але і серед дорослих і цілком серйозних дядечок, які створюють іноді такі ось

божевільні проекти – всього якихось 64 Raspberry Pi, трохи LEGO-деталей і супер-комп'ютер готовий!

1.4 Вибір мікроконтроллера

Попри універсальність Raspberry вибір було зроблено на перевагу Arduino за рахунок їх бюджетності, що дозволяє зробити цінцевий продукт більш привабливим.

На UNO вирішили замінити хардварний перетворювача USB-UART, на мікроконтролер Atmega8U2 (в більш пізніх ревізіях 16U2) – в нього залита спеціальна прошивка, що робить рівно те ж що і FT232.

Що це дало? Піднялася швидкість прошивки – тепер замість 10секунд треба чекати приблизно 3 секунди. А головне, в цей МК-конвертор можна залити свою прошивку, і перетворити Ардуіно в мишку, клавіатуру або міді пристрій – напевно комусь це дуже треба (рис. 1.3).

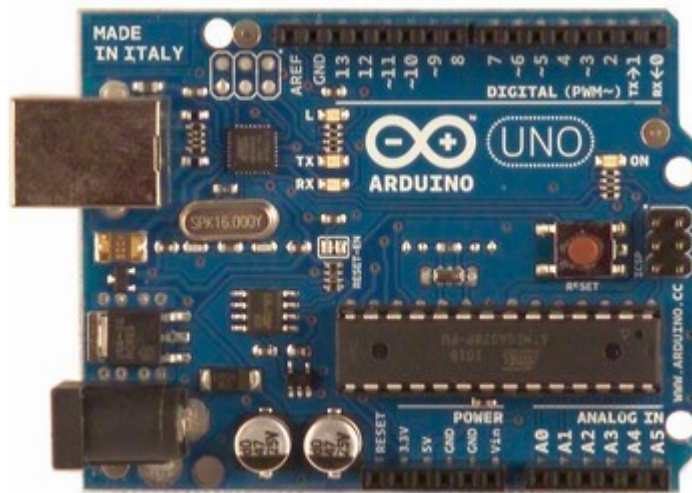


Рисунок 1.3 – Arduino Uno.

Тільки робиться це якось не дуже по-ардуіновські, і прикладів поки вкрай мало.

Так, тут ще варто зробити застереження – ця Atmega8U2 / 16U2 насправді робить не рівно те ж, що FT232, вона не реалізує дуже зручного – BitBang-a, так що перетворити плату в програматор таким нехитрим чином вже не вийде. Але все ж можна.

Arduino Uno може бути живиться від USB або від зовнішнього джерела живлення – тип джерела вибирається автоматично.

В якості зовнішнього джерела живлення (НЕ USB) може використовуватися мережевий AC / DC-адаптер або акумулятор / батарея. Штекер адаптера (діаметр – 2.1мм, центральний контакт – позитивний) необхідно вставити у відповідний роз'єм живлення на платі. У разі живлення від акумулятора / батареї, її проводу необхідно під'єднати до висновків Gnd і Vin роз'єму POWER.

Напруга зовнішнього джерела живлення може бути в межах від 6 до 20 В. Однак, зменшення напруги живлення нижче 7В призводить до зменшення напруги на виводі 5V, що може стати причиною нестабільної роботи пристрою. Використання напруги більше 12В може призводити до перегріву стабілізатора напруги і виходу плати з ладу. З огляду на це, настрійно рекомендується використовувати, як правило джерело живлення з напругою в діапазоні від 7 до 12В (рис. 1.4).

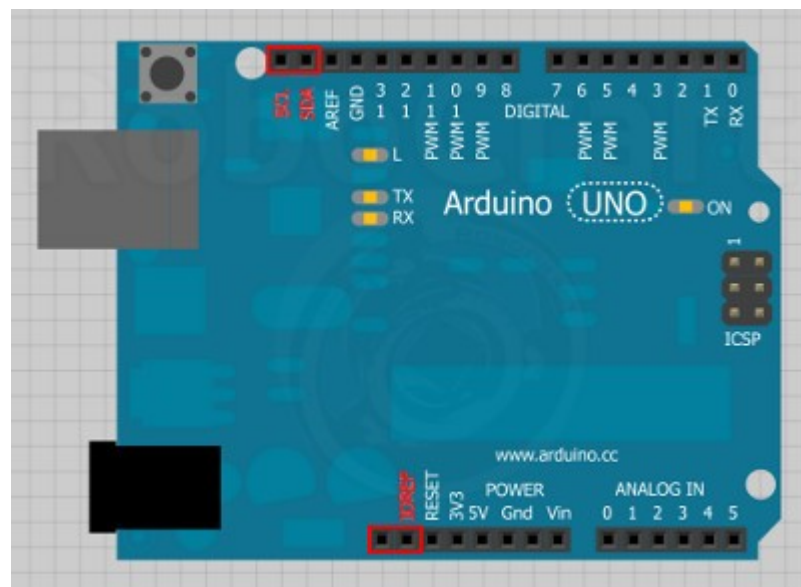


Рисунок 1.4 – Схема підключення Arduino Uno

1.5 Мова програмування C/C++

Як уже згадувалось вище – обрана нами платформа програмується з допомогою сімейства об'єктно-орієнтованих мов C/C++.

Мова Сі – це універсальна мова програмування, для якого характерні економічність вирази, сучасний набір операторів і типів даних. Мова Сі не є ні мовою "дуже високого рівня", ні "великим" мовою, і не призначається для

деякої спеціальної області застосування, але відсутність обмежень і спільність мови роблять його для багатьох завдань більш зручним і ефективним, ніж мови, імовірно більш потужні. Операційна система, компіром з мови Сі і по суті всі прикладні програми системи "ДЕМОС" написані на Сі. Мова Сі не пов'язаний з будь-якими або певними апаратними засобами або системами, і на ньому легко писати програми, які можна пропускати без змін на будь-який ЕОМ, що має Сі-компілятор.

Мова Сі є універсальною мовою програмування. Він спочатку з'явився в операційній системі UNIX, і розвивався як основна мова систем, сумісних з ОС UNIX. Сама мова, проте, не пов'язаний з якоюсь однією операційною системою або машиною; і хоча його називають мовою системного програмування, так як він зручний для написання операційних систем, він може використовуватися для написання будь-яких великих обчислювальних програм, програм для обробки текстів і баз даних.

Мова Сі - це мова щодо "низького рівня". Це означає, що Сі має справу з об'єктами того ж виду, що й більшість ЕОМ, а саме, з символами, числами і адресами.

Вони можуть об'єднуватися і пересилатися за допомогою звичайних арифметичних і логічних операцій, здійснюваних реальними ЕОМ.

У мові Сі відсутні операції, що мають справу непосредственно зі складовими об'єктами, такими як рядки символів, множини, списки або з масивами, які розглядаються як ціле. Тут, наприклад, немає ніякого аналога операціях PL/1, оперують з масивами і рядками. Мова не надає ніяких інших можливостей розподілу пам'яті, окрім статичного визначення та механізму стеків, що забезпечується локальними змінних функцій. Сам по собі мова Сі не забезпечує ніяких можливостей вводу-виводу. Всі ці механізми високого рівня повинні забезпечуватися явно викликаються функціями.

Аналогічно, мова Сі пропонує тільки прості, послідовні конструкції управління: перевірки, цикли, групування і підпрограми, але не мультипрограмування, паралельного операції, синхронізацію або співпрограми.

На відміну від бібліотек та фреймворків, інструмент зазвичай виконує завдання на клієнтському коді. Зазвичай використовуються інструменти включаючи в себе менеджери файлів і інструменти для збірки, активатори, модулі та інструменти для скаффолдингу.

Інструменти – універсальні виконавці завдань. Універсальні виконавці завдання – це інструменти, які використовуються для автоматизації певних завдань, що повторюються.

Ci ++ (англ. C ++) – компільований строго типізований мова програмування загального призначення. Підтримує різні парадигми програмування: процедурну, узагальнену, функціональну; найбільшу увагу приділено підтримці об'єктно-орієнтованого програмування.

Розробка мови почалася в 1979 році. Метою створення C ++ було доповнення C можливостями, зручними для масштабної розробки ПО, зі збереженням гнучкості, швидкості і Портательная C. Разом з тим творці C ++ прагнули зберегти сумісність з C: синтаксис першого заснований на синтаксисі останнього, і більшість програм на C працюватимуть і як C ++. Спочатку новий мова називалася "C з класами", але потім ім'я було змінено на C ++ – це повинно було підкреслити як його походження від C, так і його перевага над останнім.

Перший випуск C ++ для комерційного використання відбувся в 1985 році, разом з публікацією книги "The C ++ Programming Language", яка на довгий час стала його неофіційним стандартом. У 1989 році вийшла друга версія мови в супроводі книги "The Annotated C ++ Reference Manual".

У 1990-х роках мова стала одним з найбільш широко використовуваних мов програмування загального призначення. Першим офіційним стандартом мови став ISO / IEC 14882: 1998, більш відомий як C ++ 98. У 2003 році була прийнята його доповнена версія, C ++ 03, а в 2005 році був опублікований "Library Technical Report 1" (скорочено TR1) – документ, що описує розширення стандартної бібліотеки. TR1 не є стандартом, але більшість актуальних компіляторів C ++ підтримує його. Нарешті, в 2011 році була прийнята поточний стандарт, C ++ 11.

Нововведеннями C ++ в порівнянні з C є:

- підтримка об'єктно-орієнтованого програмування через класи. C ++ надає всі чотири можливості ООП – абстракцію, інкапсуляцію, успадкування (в тому числі і множинне) і поліморфізм.
- підтримка узагальненого програмування через шаблони функцій і класів;
- Стандартна бібліотека C ++ складається зі стандартної бібліотеки C (з деякими модифікаціями) і бібліотеки шаблонів (Standard Template Library, STL), яка надає широкий набір узагальнених контейнерів і алгоритмів;
- додаткові типи даних;
- обробка виключень;
- віртуальні функції;
- простору імен;

- вбудовуються (inline) функції;
- перевантаження (overloading) операторів;
- перевантаження імен функцій;
- посилення і оператори управління вільно розподіленою пам'яттю.

1.6. Об'єктно-орієнтоване програмування в C++

Згідно вільної енциклопедії «Вікіпедія» – Об'єктно-орієнтоване програмування – це метод програмування, заснований на поданні програми у вигляді сукупності взаємодіючих об'єктів, кожен з яких є екземпляром певного класу, а класи є членами певної ієрархії наслідування. Програмісти спочатку пишуть клас, а на його основі при виконанні програми створюються конкретні об'єкти (екземпляри класів). На основі класів можна створювати нові, які розширюють базовий клас і таким чином створюється ієрархія класів.

На думку Алана Кея, розробника мови Smalltalk, якого вважають одним з «батьків-засновників» ООП, об'єктно-орієнтований підхід полягає в наступному наборі основних принципів:

- Все є об'єктами.
- Всі дії та розрахунки виконуються шляхом взаємодії (обміну даними) між об'єктами, при якій один об'єкт потребує, щоб інший об'єкт виконав деяку дію. Об'єкти взаємодіють, надсилаючи і отримуючи повідомлення. Повідомлення – це запит на виконання дії, доповнений набором аргументів, які можуть знадобитися при виконанні дії.
- Кожен об'єкт має незалежну пам'ять, яка складається з інших об'єктів.
- Кожен об'єкт є представником (екземпляром, примірником) класу, який виражає загальні властивості об'єктів.
- У класі задається поведінка (функціональність) об'єкта. Таким чином усі об'єкти, які є екземплярами одного класу, можуть виконувати одні й ті ж самі дії.
- Класи організовані у єдину деревоподібну структуру з загальним корінням, яка називається ієрархією успадкування. Пам'ять та поведінка, зв'язані з екземплярами деякого класу, автоматично доступні будь-якому класу, розташованому нижче в ієрархічному дереві.

Таким чином, програма являє собою набір об'єктів, що мають стан та поведінку. Об'єкти взаємодіють використовуючи повідомлення. Будується ієрархія об'єктів: програма в цілому – це об'єкт, для виконання своїх функцій вона звертається до об'єктів що містяться у ньому, які у свою чергу виконують запит шляхом звернення до інших об'єктів програми. Звісно, щоб уникнути безкінечної рекурсії у зверненнях, на якомусь етапі об'єкт трансформує запит у повідомлення до стандартних системних об'єктів, що даються мовою та середовищем програмування. Стійкість та керованість системи забезпечуються за рахунок чіткого розподілення відповідальності об'єктів (за кожен дію відповідає певний об'єкт), однозначного означення інтерфейсів міжоб'єктної взаємодії та повної ізольованості внутрішньої структури об'єкта від зовнішнього середовища (інкапсуляції).

ООП виникло в результаті розвитку ідеології процедурного програмування, де дані і підпрограми (процедури, функції) їх обробки формально не пов'язані. Для подальшого розвитку об'єктно-орієнтованого програмування велике значення мають поняття події(так зване подієво-орієнтоване програмування) і компоненти (компонентне програмування, КОП).

Формування КОП від ООП відбулося, так само як формування модульного від процедурного програмування: процедури сформувалися в модулі - незалежні частини коду до рівня збірки програми, так об'єкти сформувалися в компоненти – незалежні частини коду до рівня виконання програми. Взаємодія об'єктів відбувається за допомогою повідомлень. Результатом подальшого розвитку ООП, мабуть, буде агентно-орієнтоване програмування, де агенти – незалежні частини коду на рівні виконання. Взаємодія агентів відбувається за допомогою зміни середовища, в якій вони знаходяться.

Об'єктно-орієнтоване програмування (ООП) – підхід до програмування, при якому основними концепціями є поняття об'єктів та класів.

Об'єкт – сутність володіє певним поведінкою і способом подання, т. Е. Відноситься до класу (кажуть, що об'єкт – це екземпляр класу, якщо він до нього ставиться).

Клас можна порівняти з кресленням, згідно з яким створюються об'єкти. Зазвичай класи розробляють таким чином, щоб їх об'єкти відповідали об'єктам предметної області розв'язуваної задачі.

Опис класу починається зі службового слова `class`, слідом за яким вказується ім'я класу. Потім в блоці слід опис класу і після закриває дужки блоку ставиться крапка з комою.

Опис класу складається з полів і методів.

Поля (або властивості, в рамках C ++ це можна вважати синонімом) описують те, які дані зможуть зберігати екземпляри класу (тобто об'єкти). Конкретні значення зберігаються вже всередині об'єктів. Поля оголошуються в тілі класу.

До полів всередині класу можна звертатися безпосередньо за іменами полів.

Методи класу – це функції, які зможуть застосовуватися до екземплярів класу. Грубо кажучи, метод – це функція оголошена всередині класу і призначена для роботи з його об'єктами.

Методи оголошуються в тілі класу. Описуватися можуть там же, але можуть і за межами класу (всередині класу в такому випадку досить представити прототип методу, а за межами класу визначати метод поставивши перед його ім'ям – ім'я класу і оператор).

1.7. Особливості мови програмування C++ в середовищі Arduino IDE

Мова програмування пристроїв Ардуіно заснований на C / C ++ і скомпанований з бібліотекою AVR Libc і дозволяє використовувати будь-які її функції. Разом з тим він простий в освоєнні, і на даний момент Arduino - це, мабуть, найзручніший спосіб програмування пристроїв на мікроконтролерах.

Функція `setup()` викликається, коли стартує скетч. Використовується для ініціалізації змінних, визначення режимів роботи висновків, запуску використовуваних бібліотек і т.д. Функція `setup` запускає тільки один раз, після кожної подачі живлення або скидання плати Arduino.

Після виклику функції `setup()`, яка ініціалізує і встановлює початкові значення, функція `loop()` робить те саме, що означає її назва, і крутиться в циклі, дозволяючи вашій програмі здійснювати обчислення і реагувати на них. Використовується для активного управління платою Arduino (рис. 1.5).

```
void setup()
{
}

void loop()
{
}
```

Рисунок 1.5 – Пустик «скетч» Arduino IDE

Для підключення сторонніх бібліотек в ваш скетч використовується «`#include`». Це дає доступ до великої кількості стандартних бібліотек C (бібліотекою називають групи попередньо написаних функцій), і бібліотек написаних спеціально для Arduino.

Зручною директивою, яка, дозволяє дати ім'я константі перед тим як програма буде скомпільована є «`#define`». Певні цією директивою константи не займають програмної пам'яті, оскільки компілятор замінює всі звернення до них їх значеннями на етапі компіляції, відповідно вони служать виключно для зручності програміста і поліпшення читаності тексту програми.

Варто згадати про деяке небажаному ефекті, який може мати місце при використанні директиви `#define`. Наприклад, якщо ім'я константи, заданий за допомогою директиви `#define` включити в ім'я іншої константи або змінної, то воно буде замінено на своє значення.

У загальному випадку рекомендується використовувати вираз «`const`» для визначення констант замість «`#define`»

1.8. Постановка завдання

Метою роботи є створення программно-апаратної платформи для приватної виїздної пасіки. Система повинна бути спроможна здатного представляти текстову і графічну інформацію користувачеві, здійснювати доступ до бази даних і файлів для отримання інформації, що цікавить.

Опираючись на дослідження проведені в сфері бджолярства – було встановлено, що вкрай важливими показниками для бджіл температурні умови та вологість в середині вулика. Не менш важливим показником для власника буде наявність тензо-датчиків як платформи. Їх інтеграція даватиме можливість відслідковувати динаміку зміни маси вулику на протязі дня, тижня та місяця. Такі статистичні дані безумовно дадуть можливість глибокого аналізу ефективності вашого господарства, кількості меду – який приносять бджоли за день або можливо скільки розходують – це може бути сигналом що їм не вистачає їжі.

Також BeeFarm – це система, яка буде оснащена датчиком відкриття кришки, датчиками руху – це необхідна міра безпеки, яка допоможе запобігти зловмисницьким нападам або ж вторгненню якихось тварин. Про такі засоби всього кілька років годі було мріяти малим підприємцям – розробка та інтеграція подібних рішень окупляла б себе дуже довго.

Ще однієї вимогою є об'єднання великої пасіки в єдину структуру. Кожен вулик буде оснащено необхідним набором сенсорів, компактним

мікроконтролером та джерелом живлення. Завдяки радіо-модулю кожен з подібних вуликів передаватиме всю необхідну інформацію в батьківську станцію з особистим ідентифікатором. А центральна плата формуватиме звіт (наприклад двічі в день) та відправлятиме власнику у зручному та компактному вигляді статистичної інформації з допомогою GSM-модуля зв'язку.

Крім цього в центральній-управляючій системі повинен бути реалізований зрозумілий інтерфейс, який би міг надати інформацію про стан системи та давати доступ до калібровки сенсорів кожного вулика.

Не менш важливими чинниками будуть: відмово-стійкість системи, здатність працювати за різних температурних режимів, а також як один із основних критеріїв – готовий виріб повинен залишатись доступним рішенням, а не розкішшю. Тому базова версія системи включатиме лише необхідний функціонал, але з можливістю вдосконалення за бажання замовника.

2 ВИБІР АРХІТЕКТУРНИХ РІШЕНЬ ТА ПРОГРАМНИХ ЗАСОБІВ РОЗРОБКИ СИСТЕМИ

2.1 Визначення завдань прийняття рішень вибору технології в умовах багатокритеріальності

Задача вибору і обґрунтування технології, на базі якої буде розроблена система управління бджолою пасікою, відноситься до завдань прийняття рішень в умовах багатокритеріальності.

Багатокритеріальна задача вибору формулюється в наступному вигляді. Дано безліч допустимих альтернатив, кожна з яких оцінюється безліччю критеріїв. Потрібно визначити найкращу альтернативу. При її вирішенні основна складність полягає в неоднозначності вибору найкращого рішення. Для її усунення використовуються дві групи методів. У методах першої групи прагнуть скоротити число критеріїв, для чого вводять додаткові припущення, що відносяться до процедури ранжирування критеріїв і порівняння альтернатив. У методах другої групи прагнуть скоротити число альтернатив у вихідній множині, виключивши свідомо погані альтернативи.

До методів першої групи відносяться метод згортки, метод головного критерію, метод граничних критеріїв, метод відстані. Слід зазначити, що суворе обґрунтування цих методів відсутнє і їх застосування визначається умовами завдання і перевагою особи яка приймає рішення.

Завдання багатокритеріальної оптимізації розглянемо в наступній постановці. Нехай задано безліч можливих (допустимих) рішень, позначимо його через D і безліч критеріїв їх оцінки (критеріїв) $f_1(x), f_2(x) \dots f_m(x)$, визначених на всіх елементах множини. Потрібно знайти рішення $x \in D$, для якого значення окремих критеріїв максимально можливі.

Зовні це завдання нагадує гру m осіб (гравців), в якій множина D – це безліч випадків в грі, а функції $f_i(x), i=1, m$ – це функції виграшу гравців.

Однак є різниця, яка полягає в тому, що в грі гравці прагнуть окремо, індивідуально максимізувати свій виграш за рахунок вибору своєї стратегії, а в задачі багатокритеріальної оптимізації вибрати найкращий результат який необхідний особі, що приймає рішення (ОПР), тобто в ігровому завданні кожен гравець прагне досягти свого максимального результату, а в багатокритеріальному завданні рішення приймає одна людина, яка приймає рішення.

Розглянемо методи розв'язання задачі, які часто використовували на практиці.

2.2 Теоретичні відомості про апаратні особливості реалізації

Незважаючи на те, що схемотехніка розглядає всі пристрої як "чорні ящики", тобто, ігноруючи фізичні процеси, що визначають принцип їх роботи, для введення в схемотехніку, вважаю за необхідне приділяти увагу розгляду принципів роботи окремих компонентів, їх елементарних поєднань, проте, робити це буду без зайвого "фанатизму".

Справа в тому, що крім досить складних радіоелементів, до яких відносяться, наприклад, всілякі мікросхеми, внутрішній устрій яких ми розглядати не будемо (початківцям це складно) схемотехніка використовує різні дискретні елементи – діоди, резистори, стабілітрони тощо., Знання основних принципів роботи яких може виявитися корисним.

До речі, оскільки будь-яке, навіть найскладніший пристрій, може бути приведено до досить простим комбінацій електричних (електронних) компонентів настійно рекомендую початківцям схемотехніки приділити належну увагу питанням функціонування саме елементарних вузлів.

Мушу зауважити, що схемотехніка – напромак досить складний, вимагає спеціальних знань в цілому ряді суміжних областей, однак, початківцям може вистачити елементарного володіння основними законами електротехніки – Ома і Кірхгофа.

Найменше претендуючи на короткий обіг поставленої переді мною задачі та мого способу її реалізації.

Резистори – елементи, без яких уявити схемотехніку неможливо. Резистори є найбільш поширеними елементами радіоелектронної апаратури та застосовують для регулювання струму в електричних ланцюгах.

Опір резистора – його основна характеристика. Основною одиницею електричного опору є ом (Ом) (рис. 2.1).

На принципових схемах поряд з умовним позначенням резистора проставляють значення його опору. Опір менш кілоомах записують як число без одиниць виміру; опору від одного кілоомах і вище, але менш одного мегаомах, висловлюють в кілоомах і поряд з цифрою ставлять букву "к"; опору від одного мегаомах і вище записують як число, додаючи поруч букву "М".



Рисунок 2.1 – Резистори з різним опором та маркуванням.

Значення опору зазвичай вказано на поверхні резисторів. Для маркування малогабаритних резисторів використовують буквено-цифровий код або код кольору, що складається з кольорових смужок.

Змінний резистор служить для плавного регулювання сили струму і напруги (рис. 2.2).

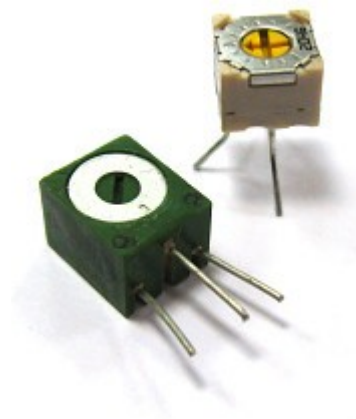


Рисунок 2.2 – Змінні резистори.

Змінні резистори ділять на регулювальні і підлаштовані. Резистори, за допомогою яких здійснюють різні регулювання зміною їх опору, називають змінними резисторами або потенціометрами. Резистори, опір яких змінюють тільки в процесі налагодження (настройки) пристрої, називають підлаштування.

Змінні резистори мають три висновки, один з яких пов'язаний з рухомим контактом, що ковзає по поверхні провідного шару. Движок регулювального резистора переміщують рукою за допомогою повороту виступає ручки, підлаштування – викруткою, вставленої в проріз.

Опір між будь-яким крайнім висновком змінного резистора і рухомим контактом залежить від положення регулятора.

Напівпровідниковий діод – це електронний прилад, виконаний на основі напівпровідникового кристала.

Варто зауважити, що технологій виготовлення діодів досить багато, але розгляд принципу роботи напівпровідникового діода на молекулярно - електронному рівні метою даної статті не є.

Справа в тому, що для більшості практичних цілей достатньо знати основні параметри, призначення, загальні принципи дії різних типів діодів, схеми підключення.

Області застосування напівпровідникових діодів вельми різноманітні, нижче я їх конспективно перерахую, а питання застосування найбільш поширених типів напівпровідникових діодів детально розгляну на відповідних сторінках.

Випрямляючі діоди мають високий опір при зворотному включенні і низьким - при прямому, тобто добре проводять струм тільки в одному напрямку.

Високочастотні і імпульсні діоди мають схожий принцип дії з попереднім типом напівпровідникових приладів, однак, за рахунок малої власної ємності можуть працювати на високих частотах, що, власне, впливає з їх назви.

Світлодіоди (LED діоди) перетворюють електричну енергію в світлову, широко використовуються як індикатори і освітлювальні пристрої (рис. 2.3).

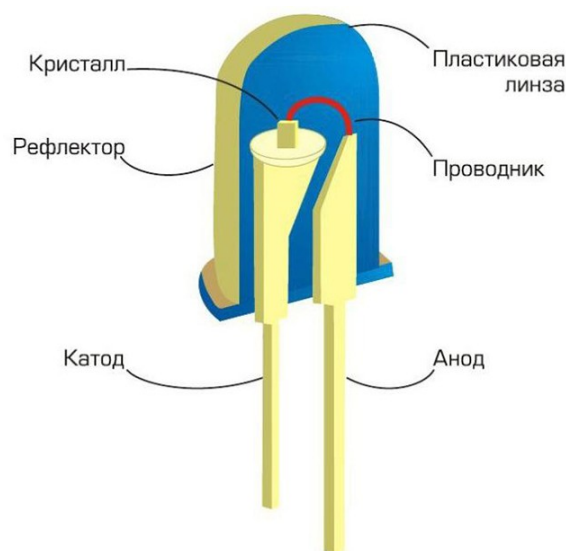


Рисунок 2.3 – Будова світлодіода.

Фотодіоди перетворюють оптичне випромінювання в електричний заряд. Можуть використовуватися як джерела електроенергії (сонячні батареї), крім того, спільно зі світлодіодами застосовуються в пультах дистанційного управління, а також можуть забезпечувати гальванічну розв'язку в електронних схемах.

Варикапи володіють залежністю своєї ємності від прикладеної напруги. Є свого роду електронно керованими конденсаторами змінної ємності.

Транзистори. Транзистор був винайдений в 50-х роках минулого століття, його поява справила справжній фурор – досить сказати, що його винахідники отримали Нобелівську премію (рис.2.4).

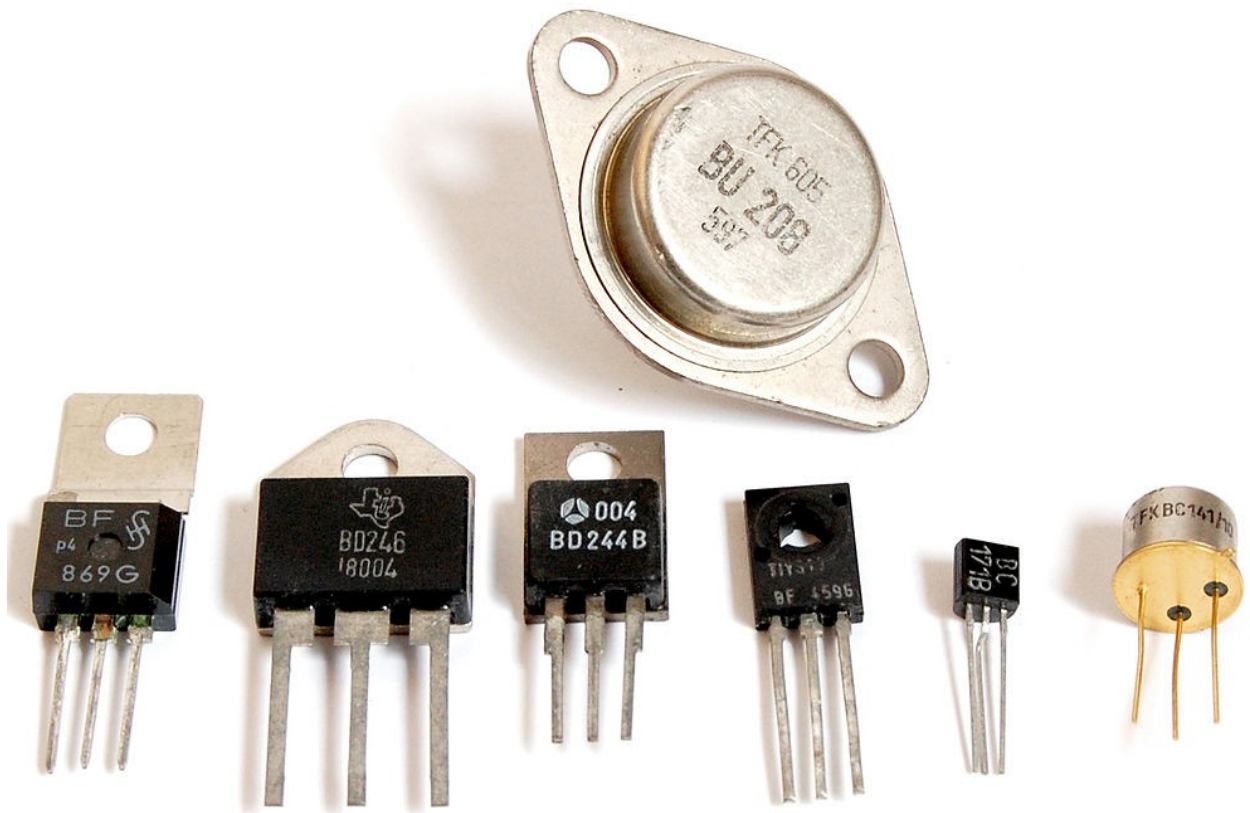


Рисунок 2.4 – Види транзисторів.

Біполярний транзистор складається з трьох шарів напівпровідника, званих "база" (Б), "колектор" (К), "емітер" (Е). Струм, що протікає через перехід база - емітер (I_B) викликає зміни опору зони емітер - колектор, відповідно змінюється струм колектора I_K , причому його значення більше ніж базового. Це основний принцип роботи біполярного транзистора, його практичне використання розглянемо пізніше.

Оскільки матеріал транзистора напівпровідник, то струм може протікати тільки в одному напрямку, який визначається типом переходу.

Відповідно до цих визначається полярність підключення (тип провідності) транзистора (пряма – р-п-р, зворотна – п-р-п. Ось, власне, вся теорія, яка Вам спочатку необхідна.

2.3 Особливості будови та застосування Arduino

Коли хто-небудь говорить про Ардуіно, він, швидше за все, має на увазі електронну монтажну плату, яка називається Arduino Uno.

Це плата з мікро контролером. За допомогою неї можна створювати і управляти абсолютно різними проектами: датчик руху, датчик температури, можна створити робота, який принесе вам тапочки. Все обмежується тільки фантазією людини, і фінансовими засобами. Наприклад, якщо створювати систему «розумний» будинок за допомогою Ардуіно, треба буде витратитися. Ардуіно можна порівняти з материнською платою комп'ютера, яка також має процесор, що знаходиться під кулером. Материнська плата з процесором виконує ті ж функції, що і Arduino Uno. До неї підключаються інші пристрої: дисплей, дисководи, жорсткі диски, і вся решта периферія. Також можна підключити багато різних пристроїв (рис. 2.5).

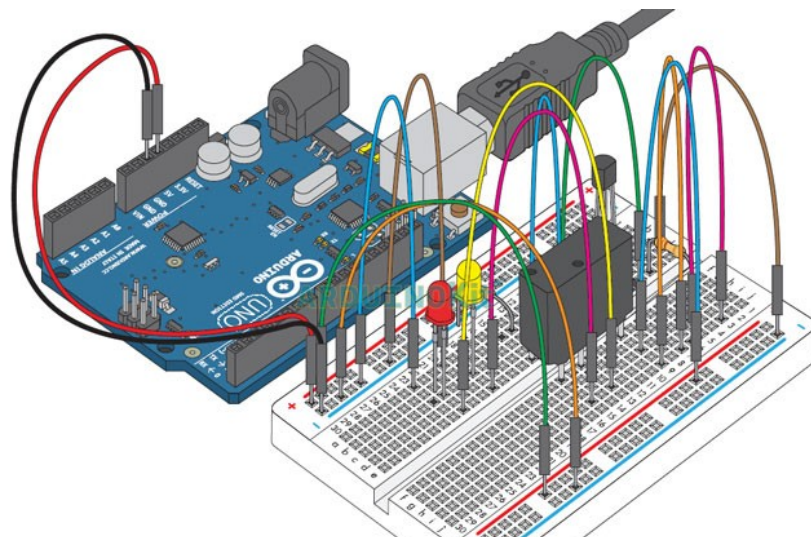


Рисунок 2.5 – Arduino підключена до макетної плати.

Унікальність Ардуіно полягає в тому, що тепер навіть недосвідчений радіоаматор зможе створювати різні складні цифрові пристрої, не особливо при цьому вдаючись в теорію. Раніше, наприклад, для цього були потрібні програматори, глибокі знання цифрової електроніки, програмування. Ардуіно створювалося, щоб практично будь-який користувач зміг створити своє цікаве і корисне пристрій.

Звичайно, плідна робота вимагає деяких знань: схемотехніки, електроніки, і навіть основ програмування на мові C++. Однак, повірте, це нескладно, якщо все докладно розглянути і вивчити. Показана вище плата не є єдиною платою в цьому сімействі. Є безліч плат з іншими мікроконтроллерами, іншими характеристиками для різних цілей. Однак Ардуіно UNO є найбільш популярною, і краще за всіх підходить для перших проектів. Це не оригінальна плата. Оригінальні коштують дуже дорого. Їх надсилають з Італії в картонних коробках, а не в поліетиленових пакетах. Але китайські клони також цілком працездатні, і зможуть замінити оригінальні плати за цілком адекватну ціну, що нас і особливо цікавить в рамках нашого непростого проекту.

2.4 Середовище розробки програмного забезпечення

В якості середовища розробки програмного забезпечення було обрано Arduino IDE 1.8 (рис. 2.6). Це програмне забезпечення з свого роду інтерпритатором для нашого мікроконтроллера який переводить мову програмування c++ в зрозумілу логіку для нашого процесора.

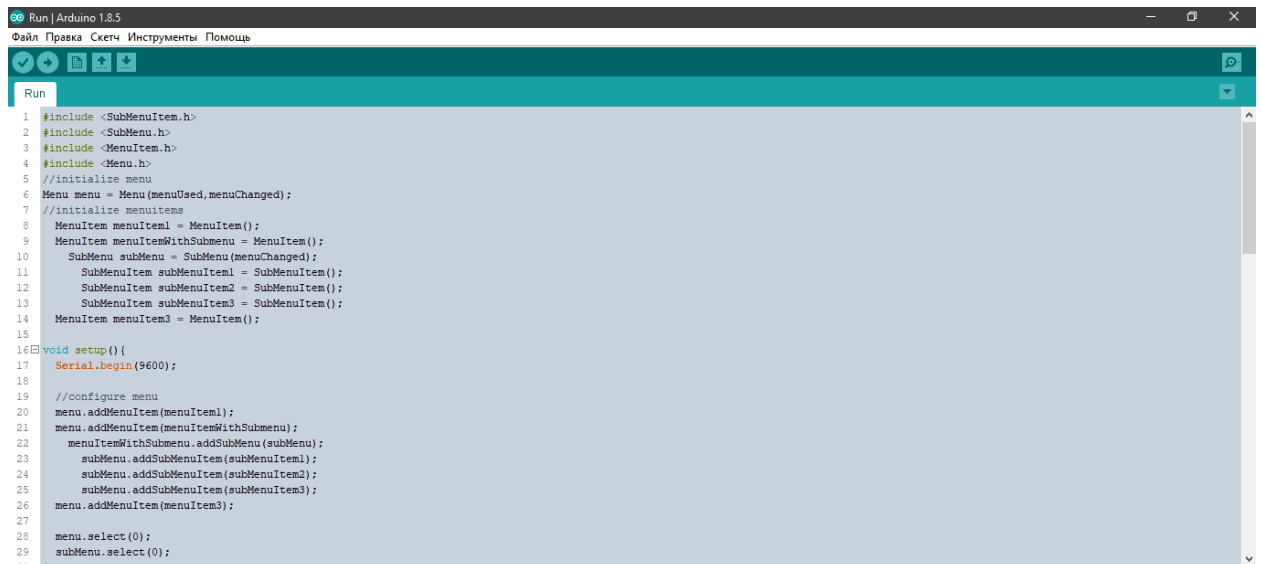


Рисунок 2.6 – Головне вікно Arduino IDE.

Дане середовище дає можливість безпосереднього редагування програмного коду та його подальшої компіляції. Також перед користувачем є можливість імпортування бібліотек, які значно спрощують інтеграцію будь-яких пристроїв, але нажаль сильно перенавантажують систему в цілому – тому найкращим виходом, в даному випадку буде дослідження кожної

бібліотеки, та витяг з неї тієї частини яка є необхідною складовою логіки роботи вашої системи (проте це нік не стосується фундаментальних бібліотек, які Ардуіно задіює на повну).

В режимі діагностики в нас є можливість вивожу необхідної нам інформації через COM-порт та дає отримувати дані з цифрових та аналогових виходів плати на екран.

Плата підключається до комп'ютера шляхом інтерфейсу USB, що на етапі діагностики та тестування може бути також джерелом живлення.

3 АНАЛІЗ ТА МОДЕЛЮВАННЯ МАЙБУТНЬОЇ СИСТЕМИ МЕТОДОМ ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

3.1 Загальна інформація

Оскільки обрана мною галузь на сьогоднішній день є досить незайнятою нішею це дає великий перелік переваг проте несе в собі і безліч труднощів.

Важливим моментом – є врахувати тонкості взаємодії людини з природою. Адже не завжди інтеграція будь-яких технологій є доречною коли мова йде про співпрацю із живими організмами, які можуть гостро реагувати на будь які магнітні коливання або випромінювання радіосигналів. Це може негативно впливати на їхню ефективність а то й куди гірше – завдавати їм шкоди.

Ставлення медоносних бджіл до випромінювань і полів різного походження є об'єктом сотень досліджень.

Все випромінювання і поля, які нас оточують, мають естестественное або штучне походження. Бджоли, як і інші живі істоти, схильні до їх дії, котре може по-різному впливати на них. Перш ніж оцінити ступінь цього впливу, класифікуємо види випромінювань і полів на основі їх походження:

- 1) Випромінювання і поля природного походження: гравітаційне поле Землі;
- 2) магнітне поле Землі;
- 3) постійне і змінне електричні поля атмосфери; природна радіація (радіоактивність) Землі; світлове випромінювання Сонця.
- 4) випромінювання і поля штучного походження:
- 5) електромагнітні поля (випромінювання) передавачів радіохвильового діапазону (радіостанції, телебачення, зв'язок, радіолокаційні станції);
- 6) електрополя високовольтних ліній електропередачі (ЛЕП); низькочастотні електрополя, створювані технічними пристроями (генераторами)» акустичні поля (звукові випромінювання); ультразвукові випромінювання.

А тепер розглянемо, як ставляться медоносні бджоли до випромінювань і полів природного походження і який вплив ці поля надають на них. Окрему увагу також приділимо питання про безпосереднє впливі електрики на бджіл.

В результаті глибокого дослідження було виявлено що ці організми досить гостро реагують на коливання які відрізняють від звичних для них природних коливань землі. Це може робити їх більш агресивними та не здатними давати гідне потомство.

Цей фактор необхідно буде врахувати при монтажі обладнання та старанній ізоляції всіх радіо-компонентів які видаватимуть будь-які коливання.

В результаті консультацій із спеціалістами з догляду та розведення бджіл, спільними зусиллями було сформульовано ряд вимог та основних базових функцій системи, які б робили це пристосування незамінним помічником господаря бджільництва не залежно від об'ємів його власності.

3.2 Контроль динаміки зміни маси вулика

Однією із фундаментальних функцій нашої системи буде засіб вимірювання маси кожного вулика та порівняння його із попереднім значенням. Дані виміри будуть проводитись декілька разів на добу. Час встановлюватиметься оптимальний для мінімізації погрешностей з кількістю осіб в середині вулика так кількості меду.

Попри оманливу тривіальність даної задачі вона дає можливість аналізу великої кількості факторів.

Пасічник, отримуючи різницю замірів до світанку (близько 4:00) та близько півночі (00:00) має можливість оцінити ефективність в той чи інший день певного вулика.

Також моніторинг на протязі дня дозволяє, у випадку втечі бджолиної сім'ї, зафіксувати цей момент та зреагувати. Адже навіть сьогодні подібні випадки є досить великою фінансовою та часозатратною неприємністю. Тому можливість її мінімізувати для фахівців – багато значить.

3.3 Визначення внутрішньої вологості повітря

Вміст водяної пари в різних зонах гнізда залежить від рівня повітрообміну між внутріульевой простором і зовнішнім середовищем. Для інтенсифікації цього процесу даху вуликів постачають вентиляційними вікнами. Якщо надрамочное простір щільно закрити поліетиленовою плівкою, то через кілька хвилин на її внутрішній стороні почнеться утворення конденсату. Отже, вологовміст повітря досягає повного насичення (100%). Абсолютна вологість повітря може в цьому випадку коливатися в широких межах в залежності від зовнішньої температури, кількості бджіл в

одиниці об'єму вулика і їх активності. При зовнішній температурі 20°C і нормальної льотної діяльності бджіл, сім'ї середньої величини абсолютна вологість в його верхній частині і зоні, протилежної летковому отвору, досягає 26-33 г / м³.

Коливання вологості в бджолиному житло можуть корелювати в певні періоди життєдіяльності сім'ї з вмістом вуглекислого газу і кисню. Це залежить значною мірою від зовнішньої температури, сезону та фізіологічного стану бджіл. У період високої активності сім'ї, як правило, такого зв'язку не простежується. У зимові ж місяці зміни вологості супроводжують значні зміни концентрації кисню і вуглекислого газу. Наприклад, від початку до кінця зими абсолютна вологість повітря в надрамочного просторі в вуликах сімей, зимуючих під відкритим небом, збільшується приблизно в півтора рази. Відповідно до цього концентрація вуглекислоти підвищується в середньому в п'ять разів, а вміст кисню зменшується в три рази.

Таким чином, гідрорежим бджолиного житла формується під впливом фізичних факторів середовища і життєдіяльності бджіл. З факторів зовнішнього середовища найбільший вплив робить вологість зовнішнього повітря, що залежить від температури. Взимку в дні з ясною морозною погодою, наприклад при -20°C вміст водяної пари поза вулика – 0,4-0,6 г / м³. Приплив такого повітря в гніздо, температура в якому не опускається взимку нижче $24-28^{\circ}\text{C}$, веде до зниження в ньому відносної і абсолютної вологості, так як зі збільшенням різниці температур між зовнішнім середовищем і внутрігніздової простором інтенсивність повітрообміну зростає.

3.4 Визначення внутрішньої температури повітря

Кожен бджоляр повинен знати температурний режим бджолиної сім'ї, розміщеної у вулику і вміти регулювати температуру у вулику в будь-який час року.

Найбільш сприятлива температура для розвитку розплоду $+35^{\circ}\text{C}$. Таку температуру бджоли підтримують на більшій частині стільників з розплодом. Температура знижується до $+34^{\circ}\text{C}$ лише на їх краях. Розплід виводиться без дефектів і в тому випадку, якщо він розвивався при температурі $+32^{\circ}\text{C}$, однак час його розвитку збільшується. З розплоду, що розвивається при температурі $+28 - +31^{\circ}\text{C}$ виведені дорослі бджоли можуть мати недорозвинені кінцівки - крила, ноги а хоботок. При температурі $+26 - +28^{\circ}\text{C}$

С розвивається лише невелика кількість личинок з недорозвиненими кінцівками. Температура нижче $+26^{\circ}\text{C}$ є недостатньою для розвитку розплоду. Розплоду шкідлива також температура вище нормальної. Якщо тривалий час зберігається температура $+37^{\circ}\text{C}$, то частина розплоду гине, а при температурі $+38^{\circ}\text{C}$ розплід гине повністю (Himmeler).

Температура $+10 - +31^{\circ}\text{C}$ для бджолиного розплоду не є небезпечною, але ембріональний розвиток бджіл затримується до тих пір, поки вони не потраплять в середу з потрібною температурою. Поки точно не встановлено, на який час при цих температурах затримується ембріональний розвиток бджіл.

Бджоли підтримують температуру у вулику в залежності від внутрішньої температури і температури зовнішнього повітря. Температуру регулюють робочі бджоли. Індивідуальна температура тіла бджоли, при якій вона виробляє нормально всі роботи, коливається в межах від $+25$ до $+36^{\circ}\text{C}$ (Kresak). Оптимальна температура для бджіл $+35^{\circ}\text{C}$. При температурі тіла бджоли нижче $+15^{\circ}\text{C}$, вона втрачає здатність літати. При зниженні температури тіла бджоли до $+9^{\circ}\text{C}$, починається холодовий заціпеніння, і тим раніше, ніж бджола голодніше і молодше (Suva). При температурі тіла бджоли $+6^{\circ}\text{C} - +4^{\circ}\text{C}$ холодовий заціпеніння посилюється, а при температурі $+2^{\circ}\text{C}$ бджола зовсім ціпеніє. Стан заціпеніння через низьку температури схоже на параліч. Подібний стан викликається уповільненням процесу дихання, яке може повністю припинитися, тим самим порушуючи підведення кисню. Нестача кисню викликає гальмування діяльності Центральної нервової системи, що в свою чергу призводить до паралічу. Однак якщо заціпенілих бджіл відігріти, то вони приходять до тями. При температурі тіла пчели $+0,5-12^{\circ}\text{C}$, бджола витримує в заціпенілому стані півтора дня (Wegmann).

Бджолина сім'я як ціле, і головним чином, зимовий клуб бджіл, витримує зниження температури краще, ніж окремі особини. Якщо температура зовнішнього повітря знижується нижче $+11^{\circ}\text{C}$, більшість бджіл збирається близько кормових запасів у гнізді. При температурі зовнішнього повітря нижче $+6^{\circ}\text{C}$ бджоли об'єднуються в клуб і щільно притискаються один до одного. У такому стані вони зможуть перенести і суворі зими.

3.5 Фіксація руху поряд на території бджільництва

Нажаль досить поширеним явищем сьогодні є зловмисні дії певних осіб. Цілей тому може бути безліч, але результат завжди один – збитки для власника.

Способів боротьби із подібними випадками може бути теж безліч, але все ж методів їх автоматичної фіксації не так вже й багато. Один із найефективніших – датчики руху. Які в разі виявлення будь-якого руху тіл (більших від собаки) негайно подаватимуть сигнал до нашої платформи, яка прийматиме подальші міри.

3.6 Формування звіту для власника

Попри велику кількість корисних функцій по збору, аналізу та контролю наою системою оточуючого середовища та інших факторів – все ж важливим моментом є своєчасне отримання цієї інформації.

Важливою інтегрованою функцією нашої платформи BeeFarm буде відправка актуальних звітів господареві про стан його пасіки.

Проте також вся необхідна інфомація знаходитиметься в самому пристрою і може бути зчитана із дисплею за допомогою маніпуляторів (клавіш).

4 ПРОЕКТУВАННЯ ТА ПРАКТИЧНА РЕАЛІЗАЦІЯ СИСТЕМИ УПРАВЛІННЯ ТА МОНІТОРИНГУ ЗА ПАСІКОЮ

4.1. Проектування та реалізація підключення тензодатчиків

Функцію зважування вулику буде реалізовано шляхом розміщення платформи під ним. Платформа складатиметься із 4 тензодатчиків які будуть розміщені по краям (рис 4.1).

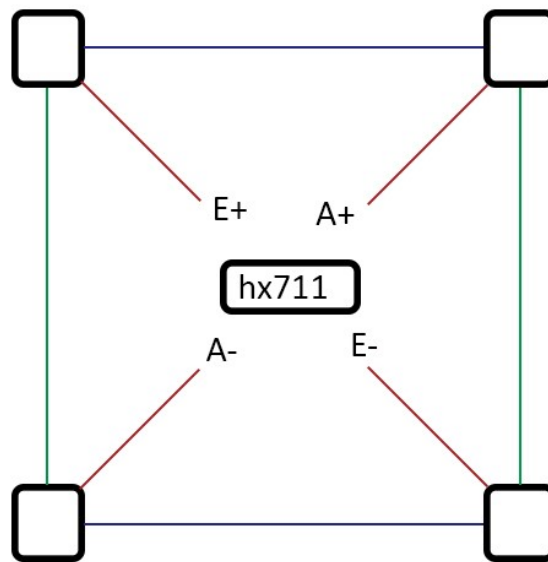


Рисунок 4.1 – Принципова схема з'єднання тензорезисторів.

Тензометрический датчик (рис. 4.2) – датчик, що перетворює величину деформації в зручний для вимірювання сигнал (зазвичай електричний), основний компонент тензометра (приладу для вимірювання деформацій).

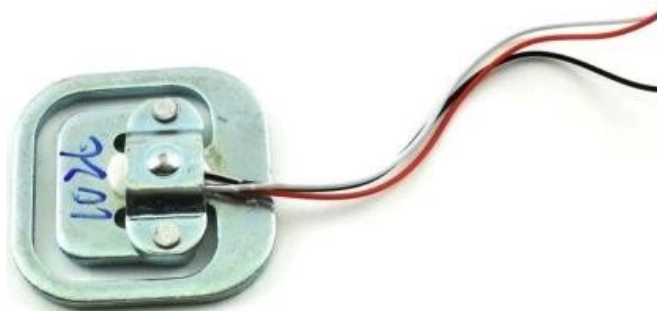


Рисунок 4.2 – зразок тензодатчика який використаний в проекті

Існує безліч способів вимірювання деформацій: тензорезистивного, п'єзоелектричний, оптико-поляризаційний, п'єзорезистивного, волоконно-оптичний, або простим зчитуванням показань з лінійки механічного тензодатчика. Серед електронних тензодатчиків найбільшого поширення набули тензорезистивні датчики.

Дані датчики здатні витримувати навантаження до 50 кілограм кожен. Це дає нам в результаті вантажопід'ємність в 200 кілограм – це вистачить для наших цілей. Для того щоб вони працювали як єдина система – вони будуть з'єднані між собою паралельно та послідовно як показано на рис. 4.2.

Безумовно важливим моментом на даному етапі є якісні з'єднання, адже будь які проблеми із отриманням сигналу призведуть до серйозних погіршень (рис. 4.3).

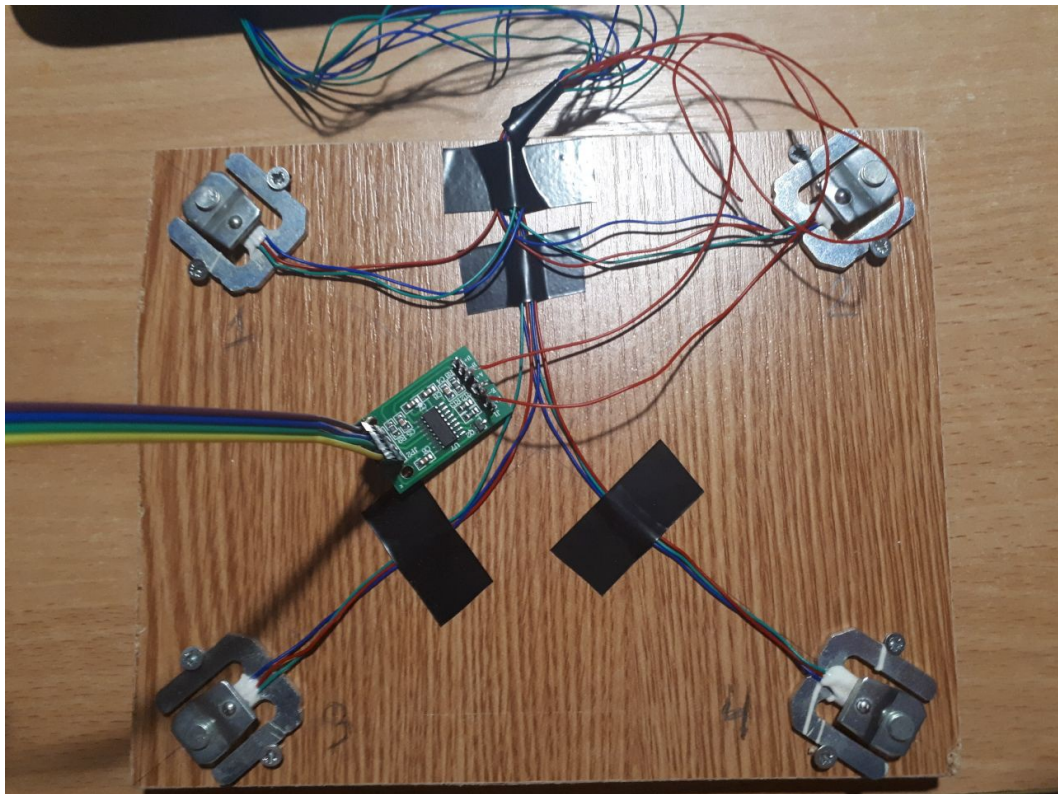


Рисунок 4.3 – Реалізація вагової платформи вулика

Посередником, який декодуватиме сигнал для нашої платформи Arduino буде АЦП hx711. Для його правильного функціонування нам необхідно правильно підключити його (рис. 4.4).

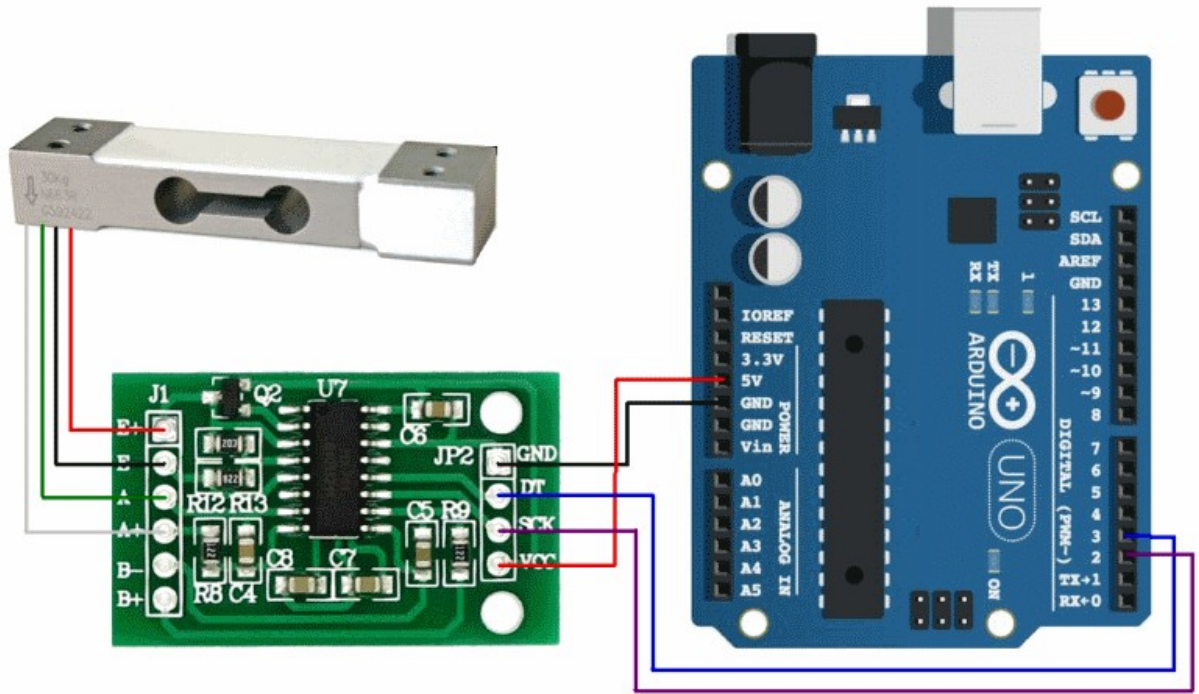


Рисунок 4.4 – Схема підключення hx711 до Arduino.

В якості тестового зразка було зібрано платформу, яка на даному етапі розробки цілком задовільняє наші потреби та надаватиме нам всі необхідні дані.

4.2. Реалізація моніторингу за вологістю та температурою

Як ми визначили на етапі постановки задачі такі показники як вологість та температура є вкрай важливими для бджіл. Відповідно це вносить контроль за даними явищами необхідною функцією нашого пристрою.

Найефективнішим способом отримати потрібні нам показники – підключення модулю Dht22 (рис 4.5).

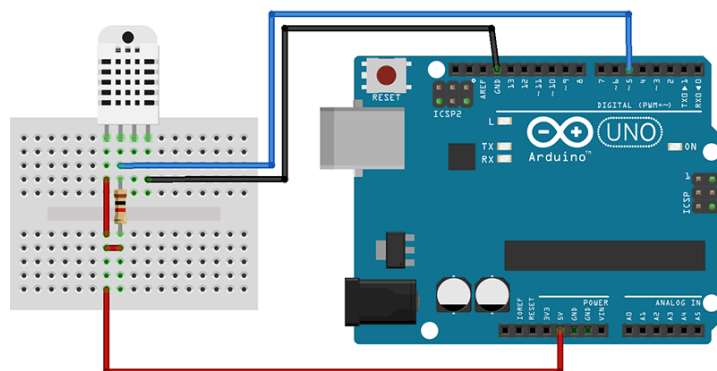


Рисунок 4.5 – Схема підключення датчика вологості до Arduino.



Рисунок 4.6 – Підключення Dht22 до Arduino.

Він володіє достатніми показниками точності та роботи при низьких температурах та не створює додаткових коливань в середині.

В результаті цифрового сигналу на один із конекторів нашої Arduino ми отримуємо дані про вологість та температуру повітря, з якими ми можемо працювати, виводити на екран та відправляти господареві (рис. 4.6).

4.3 Підключення LCD дисплею для виводу інформації

Щоб зробити подання інформації більш зручним та дружельюбним до користувача було інтегровано LCD модуль 1602 для виводу зображення в розширенні 16x2 точок. Його було підключено через змінний резистор (потенціометр) для можливості регулювання контрастності дисплею (рис. 4.7). Це сприятиме читабельності тексту за будь-яких умов. Контакти на цьому дисплеї пронумеровані від 1 до 16. Завдано вони на задній частині плати. Підключення 1602 до Arduino Якщо дисплей 1602 харчується від Arduino через 5-ти вольтової USB-кабель і відповідний пін, для контакту контрасту дисплея (3-й коннектор – Contrast) можна використовувати номінал 2 кОм. Для Back LED + контакту можна використовувати резистор на 100 Ом.

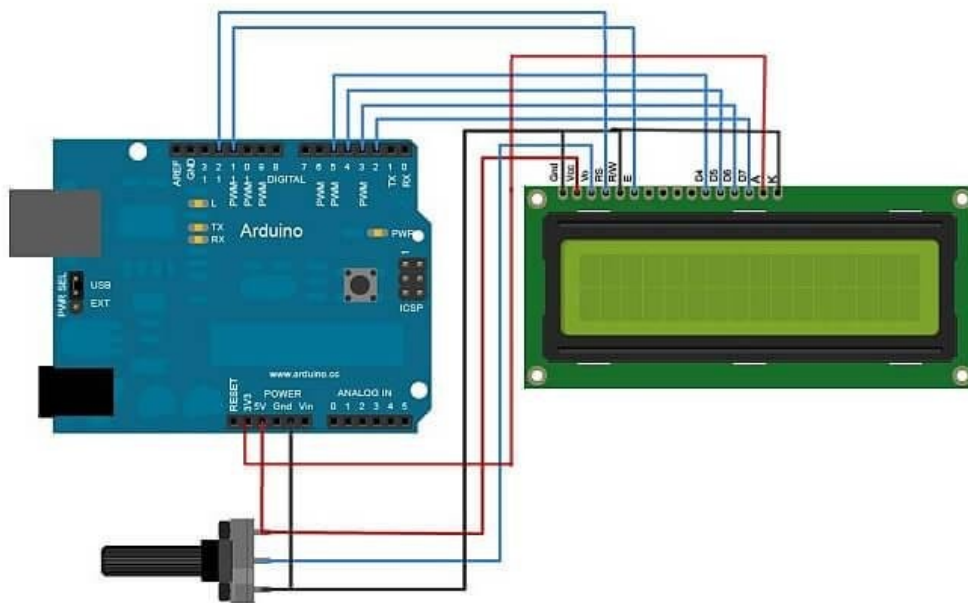


Рисунок 4.7 – Схема підключення LCD 1602 до Arduino.

Можна використовувати і змінний резистор – потенціометр для настройки контрасту вручну.

Енергоефективність в нашому випадку грає далеко не останню роль – тому нам дисплей відображатиме інформації лише на короткий час після нажаття на відповідну клавішу (рис 4.8).

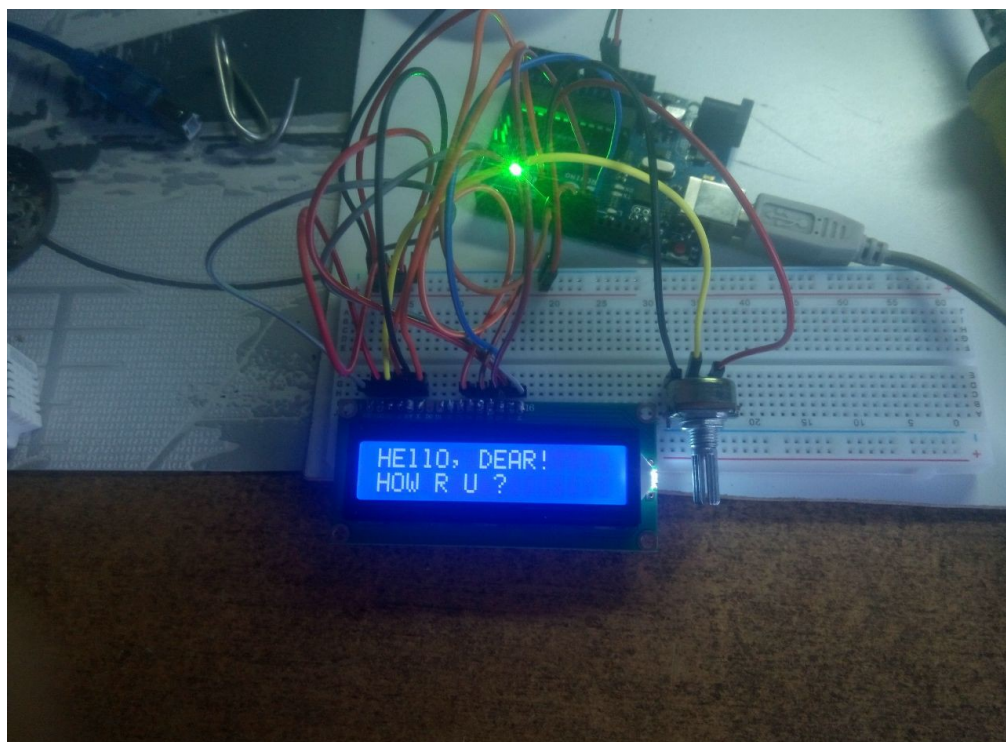


Рисунок 4.8 – Макет підключення LCD 1602 до Arduino.

4.4 Реалізація системи охорони та сповіщення

Систему детекції руху в радіусі досяжності вулику реалізовано завдяки датчику SR-501 в якому встановлений один такий піроелектричний датчик. Зверху піроелектрик оточений півсферою, розбитою на кілька сегментів. Кожен сегмент цієї сфери є лінзою, яка фокусує теплове випромінювання на різні ділянки ПІР-датчика (рис 4.9). Часто в якості лінзи використовують лінзу Френеля.

Такі датчики часто використовуються в охоронних системах і в побуті для виявлення руху в приміщенні. Наприклад, на принципі визначення руху засноване автоматичне включення світла в під'їзді або у ванній. Піроелектричні датчики досить простого влаштувані, недорогі і невибагливі в установці і обслуговуванні.

Принцип роботи датчику руху наступний. Припустимо, що датчик встановлений в вулику. Кожен чутливий елемент отримує постійну дозу випромінювання, а значить і напруга на них має постійне значення.

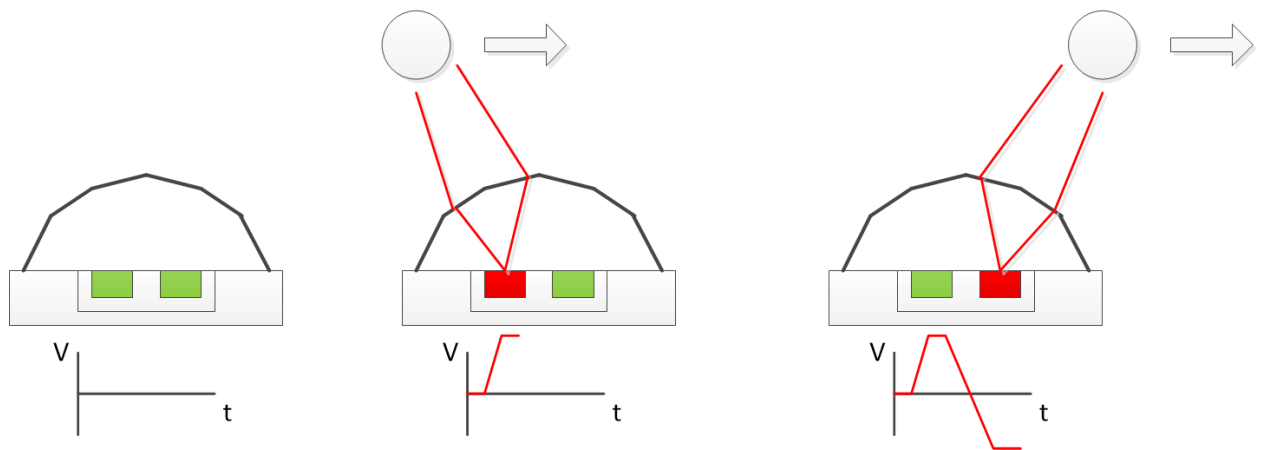


Рисунок 4.9 – Принцип спрацювання датчика руху.

Як тільки в поле радіусу дії заходить будь-яка жива істота, вона потрапляє спочатку в зону огляду першого елемента, що призводить до появи позитивного електричного імпульсу на ньому.

Об'єкт рухається, і його теплове випромінювання через лінзи потрапляє вже на другий PIR-елемент, який генерує негативний імпульс. Електронна схема датчика руху реєструє ці різноспрямовані імпульси і

робить висновки про те, що в поле зору датчика потрапив об'єкт, що рухається. На виході датчика генерується позитивний імпульс.

Підключення до платформи Arduino – досить просте, оскільки й принцип роботи пристрою полягає в зміні різниці імпульсів (рис 4.10).

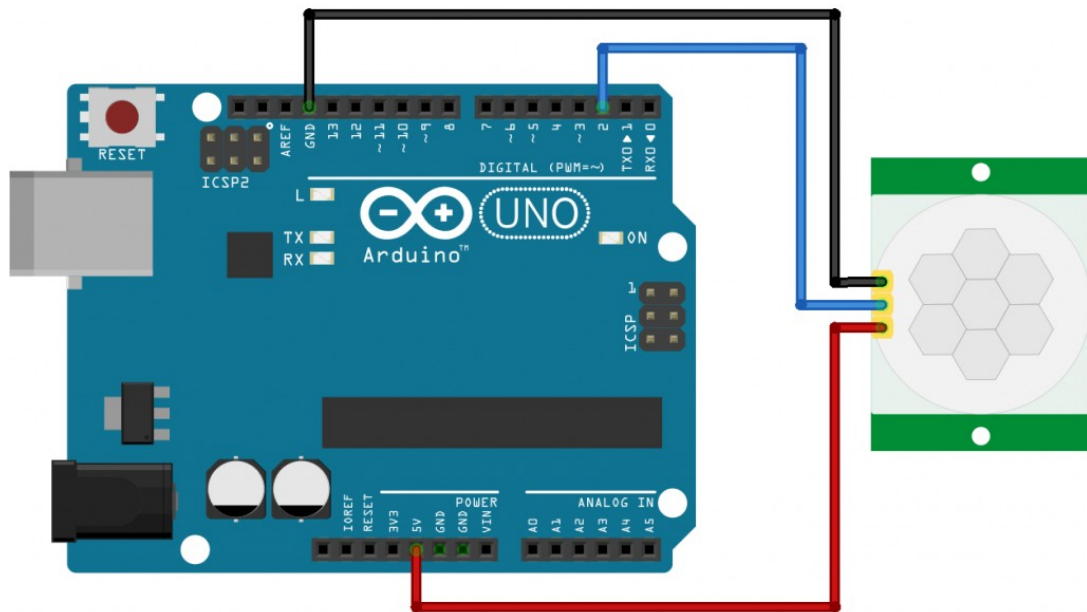


Рисунок 4.10 – Схема підключення датчика руху до Arduino.

4.5 Реалізація системи стільникового зв'язку

Стільниковий зв'язок, мережу рухомого зв'язку - один з видів мобільного радіозв'язку, в основі якого лежить стільникова мережа. Ключова особливість полягає в тому, що загальна зона покриття ділиться на осередки (стільники), що визначаються зонами покриття окремих базових станцій (БС).

Стільники частково перекриваються і разом утворюють мережу. На ідеальній (рівній і без забудови) поверхні зона покриття однієї БС являє собою коло, тому складена з них мережа має вигляд шестикутних осередків (сот).

Мережа становлять рознесені в просторі приймач, що працюють в одному і тому ж частотному діапазоні, і комутуюче обладнання, що дозволяє визначати поточне місце розташування рухливих абонентів і забезпечувати безперервність зв'язку при переміщенні абонента із зони дії одного приймач в зону дії іншого.

Пристроєм який було обрано для реалізації стількинового з'язку є – SIM900 GSM/GPRS модуль (рис 4.11). Радіоелементи цієї плати є дуже чутливими до перепадів напруги тому він потребує окремого джерела живлення та трансформатора до 3.3 вольт.

Даний пристрій відправлятиме звичайні СМС повідомлення на вказаний номер із інформацією яку для нього формуватиме наша Arduino.

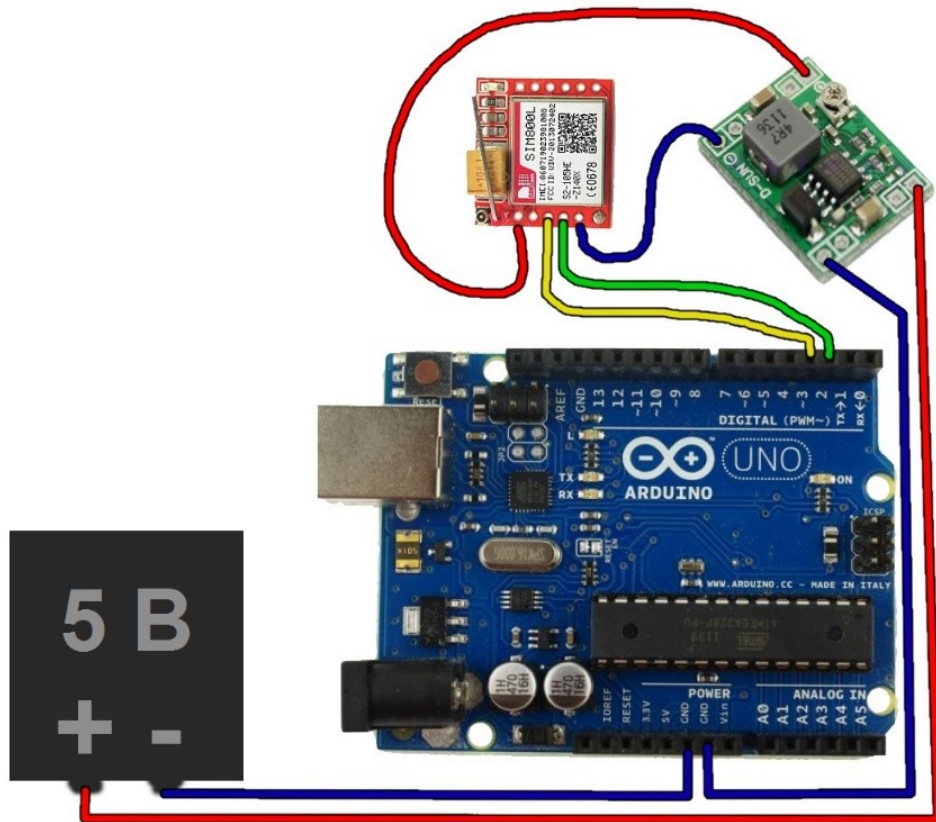


Рисунок 4.11 – Схема підключення модуля стільникового зв'язку до Arduino.

4.5 Прототип системи BeeFarm

На даному етапі цього проекту було протестовано всі модулі системи на працездатність із нашою платформою.

Також було написано безмірна кількість програмного коду, який поясню наші Arduino що саме до неї підключено та в якому вигляді. В результаті було зібрано тестовий стенд який зображено на рисунку 4.12 – він дає можливість виявлень недосконалостей системи та вразливостей.

До макетної плати було підключено:

- 1) засіб відображення інформації – LCD 1602;
- 2) датчик температури та вологості повітря – Dht22;

- 3) датчик руху – SR-501;
- 4) потенціометр;
- 5) модуль стільникового зв'язку – GSM900L;
- 6) індикатор – LED (синього кольору);
- 7) модуль декодування сигналу тензодатчиків – hx711;
- 8) резистори змінного опору (тензодатчики) – ESP8266;
- 9) мікроконтроллер – Arduino Uno;
- 10) Джерело живлення – Крона на 9 вольт.

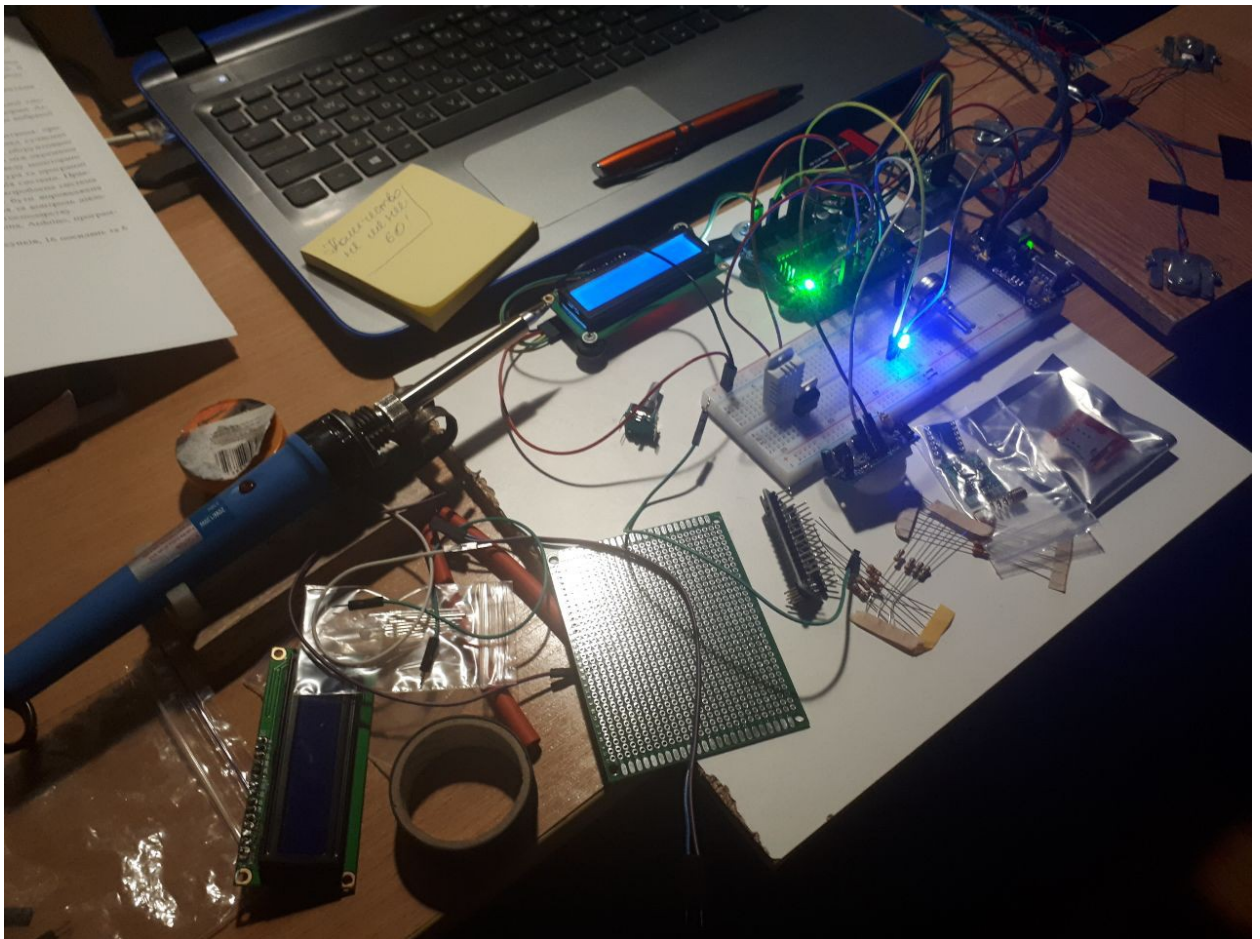


Рисунок 4.12 – Тестовий стенд BeeFarm на базі Arduino.

4.6 Виявлення недоліків системи BeeFarm

Одразу ж після підключення прототипу до джерела живлення – стало зрозуміло, що автономність системи під великою загрозою через перегруженість мікроконтроллера компонентами. Оскільки тривалий час роботи для нас є пріоритетною задачею – необхідно приймати кардинальні міри по вдосконаленню моделі та, можливо відмовою від деяких її компонентів та систем.

Також було виявлено велику погрішність у роботі однієї із основних систем – виміру ваги вулика. В зв'язку з тим що модуль АЦП hx711 є дуже чутливою мікросхемою – процес її калібровки дуже трудоемний процес (рис. 4.13).

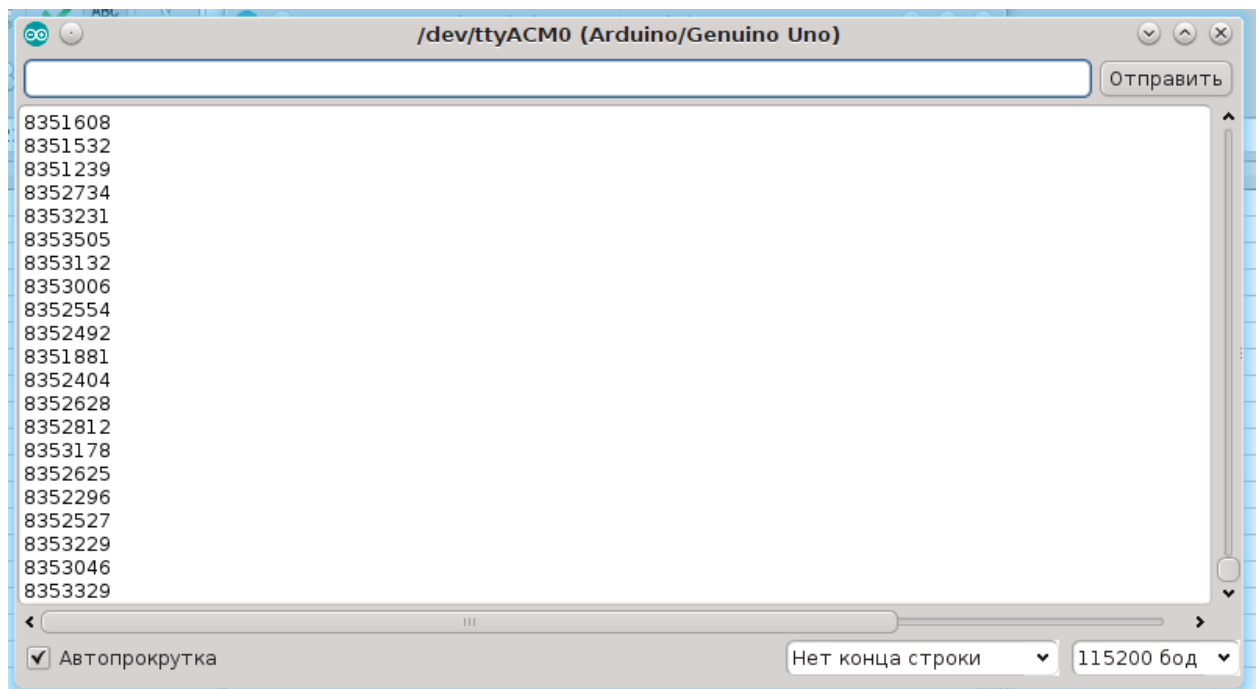
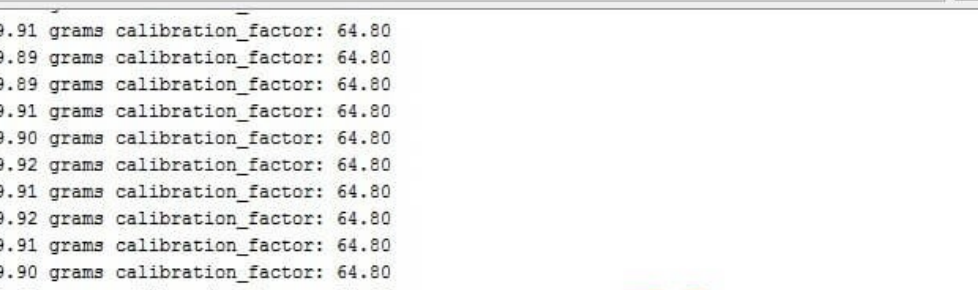


Рисунок 4.13 – Екран калібрування модуля тензодатчиків.

Все ж приємливого результату досягти так і не вдалось, з досить таки банальних причин – дуже велика кількість браку при замовленні дешевих радіодеталей. Після безліч спроб натрапити на якісний товар – було досягнуто результатів які цілком задовільняли наші цілі та вимоги стосовно проекту BeeFarm (рис. 4.14, 4.15).



The screenshot shows the Arduino IDE serial monitor window. The title bar reads "COM3 (Arduino Uno)". The monitor displays a series of weight readings in grams, each followed by a calibration factor of 64.80. The readings are: 69.91, 69.89, 69.89, 69.91, 69.90, 69.92, 69.91, 69.92, 69.91, 69.90, 69.90, 69.92, 69.91, 69.95, 69.93, 69.87, 69.87, 69.86, 69.86, 69.86, 69.84, 69.88, 69.88, 69.85, 69.86, and 69.85. A red arrow points from the text "70 г" to the reading "69.87 grams". The status bar at the bottom shows "Автопрокрутка" (checked), "Не найден конец строки", and "9600 бод".

Reading (grams)	calibration_factor
69.91	64.80
69.89	64.80
69.89	64.80
69.91	64.80
69.90	64.80
69.92	64.80
69.91	64.80
69.92	64.80
69.91	64.80
69.90	64.80
69.90	64.80
69.92	64.80
69.91	64.80
69.95	64.80
69.93	64.80
69.87	64.80
69.87	64.80
69.86	64.80
69.86	64.80
69.86	64.80
69.84	64.80
69.88	64.80
69.88	64.80
69.85	64.80
69.86	64.80
69.85	64.80

Рисунок 4.15 – Екран вирахування калібровочного фактору.

4.6 Заплановані вдосконалення платформи

Згідно відомого прислів'я –ланцюг міцний настільки, наскільки міцна її найслабша ланка. Найслабшою ланкою нашої системи є живлення та автономна робота тому додаткове джерело живлення – необхідне (рис. 4.16).



Рисунок 4.16 – Сонячна панель.

Логічними та першочерговими змінами стосовно даного питання буде, звичайно ж перехід на більш досконалий тип живлення – від акумуляторних батарей які будуть заряджатись на протязі дня від сонячної панелі.

Також було прийнято зміни моделі платформи Arduino з Uno на Nano (рис. 4.17).

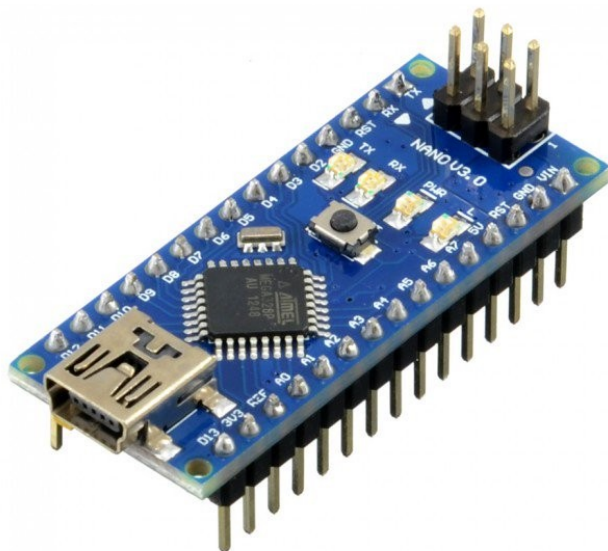


Рисунок 4.17 – плата Arduino Nano.

Важливими перевагами Nano є енергоефективність та компактні розміри. Також приємно радує відсутність додаткової індикації та можливість відключити контролер живлення (рис. 4.18).

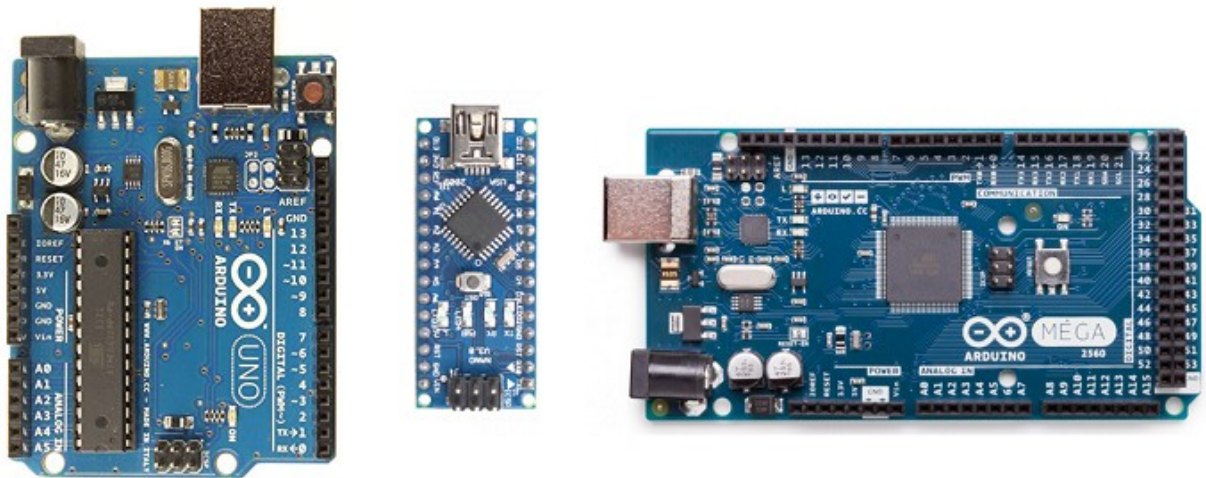


Рисунок 4.18 – Сімейство плат Arduino.

ВИСНОВКИ

В рамках виконання магістерської роботи було здійснено дослідження предмета, цілей і особливостей розробки системи управління діяльністю бджільництва та пасіки на основі застосування сучасної платформи на базі мікро-контролера ATmega328.

Проведено аналіз методів прийняття рішень в умовах багатокритеріальності та обґрунтування вибору технології програмної реалізації системи на основі методу головного критерію.

В роботі проведено вибір архітектурних рішень та програмних засобів реалізації системи управління та здійснена програмна реалізація системи.

Було враховано всі відомі на момент проектування даної системи нюанси, та реалізований прототип, готовий до тестування в сезон бджільництва 2018 року.

Підтримка технологією даних реального часу приваблива і важлива, особливо з урахуванням того, що подібні рішення в Україні досі впроваджено не було. Взаємодія в режимі реального часу набуває все більшого значення для майбутнього, і здатність BeeFarm легко і швидко працювати зі складним набором даних в режимі реального часу також стає все більш цінною.

Система управління реалізує всі визначенні вимоги та функціональні можливості для всіх категорій користувачів системи. Застосування платформи BeeFarm при здійсненні програмної реалізації системи створює фундамент, що дозволяє з легкістю здійснювати модернізацію системи та розширювати її функціональні можливості.

В реалізованій системі звичайно є багато недоліків які будуть усуватись в процесі тестування та розвитку до повноцінного сертифікованого продукту який на своєму останньому етапі повинен стати повноцінною системою для впровадження.

На наступному етапі вкрай важливо заручитись підтримкою та консультацією фахівців в галузі бджільництва, які готові інтегрувати на своїх узгіддях систему BeeFarm та активно підтримувати розвиток даного продукту та займатись збором необхідної інформації.

Для подальшого росту проекту необхідна фінансова підтримка зацікавлених в цьому фізичних або юридичних осіб. Це дасть можливість збільшити кількість персоналу який працюватиме над збором, встановленням та супроводом даної платформи на різних пасічних узгіддях.

Якщо говорити об'єктивно – BeeFarm на даному етапі розвитку є «сирим» та недосконалим рішенням і більшість питань його працездатності стоїть під питанням. Проте враховуючи пройдений шлях – показує себе як вельми перспективна система – яка зможе справитись із поставленою на неї задачею – спростувати життя звичайним дрібним підприємцям.

ПЕРЕЛІК ПОСИЛАНЬ

1. Бджільництво в Україні [Електронний ресурс] – Режим доступу: <http://beekeeping.com.ua>.
2. Arduino в Україні [Електронний ресурс] – Режим доступу: <https://arduino.ua/>
3. Программирование на C и C++ [Електронний ресурс] – Режим доступу: <http://www.c-cpp.ru/books/yazyk-s>.
4. Arduino [Електронний ресурс] – Режим доступу: <https://www.arduino.cc/> ІТ українською – огляд 5 найпопулярніших JavaScript-фреймворків 2017 [Електронний ресурс] – Режим доступу: <http://it-ua.info/news/2017/02/14/oglyad-5-naypopulyarnshih-javascript-freymvorkv-2017.html>
5. Вікіпедія – Ангуляр [Електронний ресурс] – Режим доступу: <https://ru.wikipedia.org/wiki/AngularJS>
6. getInstance – основи використання Gulp для збірки JavaScript-додатків [Електронний ресурс] – Режим доступу: <https://getinstance.info/articles/tools/introduction-to-gulp/>
7. Хакер – atom текстовий редактор [Електронний ресурс] – Режим доступу: <https://xaker.ru/2014/02/27/62122/>
8. DeveloperWorks – розробка простих web-сайтів реального часу на платформі Meteor [Електронний ресурс] – Режим доступу: <https://www.ibm.com/developerworks/ru/library/wa-meteor/index.html>.
9. Хабрахабр – Переваги використання JavaScript фреймворків [Електронний ресурс] – Режим доступу: <https://habrahabr.ru/post/321844/>
10. Методи рішення багатокритеріальних задач вибору [Електронний ресурс] – Режим доступу: <http://um.co.ua/4/4-18/4-180270.html>
11. Студопедія – багатокритерійні завдання і можливі шляхи їхнього рішення [Електронний ресурс] – Режим доступу: <http://studopedia.com.ua/bagatokriteriyni-zavdannya-i-mozhlivi-shlyahi-ih-nogo-rishennya.html>
12. Хабрахабр – як влаштований Meteor зсередини [Електронний ресурс] – Режим доступу: <https://habrahabr.ru/post/205878>.
13. Фізичне проектування бази даних. [Електронний ресурс] – Режим доступу: <http://da.coolreferat.com.ua/nuda/proektuvannya-bagatovimiri-nih-baz-danih/main.html>

14. IT українською – розробка Meteor.js і реактивне програмування [Електронний ресурс] – Режим доступу: <http://it-ua.info/news/2014/08/11/rozrobka-z-meteorjs--reaktivne-programuvannya.html>
15. Bitbucket хмарний GIT репозиторій [Електронний ресурс] – Режим доступу: <http://jak.magey.com.ua/articles/bitbucket-krutij-hmarnij-git-repozitorij.html>

Д О Д А Т К И

ДОДАТОК А

ПРОГРАММНЫЙ КОД МЕНЮ

```
//-----ПЕРЕМЕННЫЕ-----
float calibr_f=0; // Коэффициент калибровки
float known_weight=0; // Известный вес
float weight=0; // Последний зафиксированный вес
float tare=0; // Тарирование в 0
int stb_time = 10000; // время бездействия, через которое система уйдёт в сон (миллисекунды)
//-----БИБЛИОТЕКИ-----
#include <MenuSystem.h> // Меню
#include "LowPower.h" // Энергосбережение
#include "EEPROM.h" // Энергонезависимая память
#include "LCD_1602_RUS.h" // Дисплей
//-----НАСТРОЙКИ-----
LCD_1602_RUS lcd(0x27, 16, 2); // создать дисплей
boolean recogn_flag, sleep_flag = true; // флажки
//-----КНОПКИ-----
byte button = 2; // кнопка "проснуться"
byte calibr_button = 3; // скрытая кнопка калибровки и сброса
byte disp_power = 12; // питание дисплея
//-----ИНИЦИАЛИЗАЦИЯ-----
void setup() {
  pinMode(button, INPUT_PULLUP); // подтягиваем кнопки
  pinMode(calibr_button, INPUT_PULLUP);
  pinMode(disp_power, OUTPUT); // пины питания как выходы
  digitalWrite(disp_power, 1); // подать питание на дисплей
  attachInterrupt(0, wake_up, CHANGE); // подключить прерывание
  lcd.init(); // инициализация дисплея
  lcd.backlight();
  if (!digitalRead(calibr_button)) { // если при запуске нажата кнопка калибровки
    lcd.clear();
    lcd.setCursor(3, 0);
    lcd.print(L"Сервис");
    delay(500);
    reset_timer = millis();
    while (1) { // бесконечный цикл
      if (millis() - reset_timer > 3000) { // если кнопка всё ещё удерживается и прошло 3 секунды
        //сброс показателей тарирования и калибровки
        EEPROM_float_write(0, 0);
      }
    }
  }
}
```

```

EEPROM_float_write(5, 0);
}
lcd.clear();
lcd.setCursor(0, 0);
lcd.print(L"Память очищена");
delay(100);
}
if (digitalRead(calibr_button)) { // если отпустили кнопку, перейти к
калибровке
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print(L"Калибровка");
    break;
}
}
while (1) {
// процесс записи новых данных
    break;
}
}
// при старте системы считать из памяти данные для дальнейшей работы
calibr_f = EEPROM_float_read(0);
tare = EEPROM_float_read(0);

standby_timer = millis(); // обнулить таймер ухода в сон
}
//-----БЕСКОНЕЧНЫЙ ЦИКЛ РАБОТЫ-----
void loop(){
}
//-----ЗАПИСЬ-----
void EEPROM_float_write(int addr, float val) // запись в ЕЕПROM
{
    byte *x = (byte *)&val;
    for(byte i = 0; i < 4; i++) EEPROM.write(i+addr, x[i]);
}
//-----ЧТЕНИЕ-----
float EEPROM_float_read(int addr) // чтение из ЕЕПROM
{
    byte x[4];
    for(byte i = 0; i < 4; i++) x[i] = EEPROM.read(i+addr);
    float *y = (float *)&x;
    return y[0];
}

```

```

//-----ФУНКЦИЯ СНА-----
void good_night() {
    // перед тем как пойти спать, записываем в EEPROM новый полученный вес
    EEPROM_float_write(5, weight);
    sleep_flag = true;
    // вырубить питание со всех дисплеев и датчиков
    digitalWrite(dis_power, 0);
    delay(100);
    // и вот теперь спать
    LowPower.powerDown(SLEEP_FOREVER, ADC_OFF, BOD_OFF);
}
//-----ПРОСЫПАЕМСЯ
ПРЕРЫВАНИЮ-----
void wake_up() {
    // возвращаем питание на дисплей и датчик
    digitalWrite(dis_power, 1);
    standby_timer = millis(); // и обнуляем таймер
}

```

ПО

ДОДАТОК Б

ПРОГРАММНЫЙ КОД КАЛИБРОВКИ

```

#include "HX711.h"
// Скетч выводит в монитор порта Arduino IDE калибровочный коэффициент
// весов для функции set_scale()
// Впишите известный вам вес предмета в граммах в строку с указывающим
// комментарием в нижней части этого скетча (рекомендуется от 500 до 1000
// граммов)
// Загрузите скетч в Arduino
// Убедитесь, что на весах нет груза
// Запустите монитор порта
// В течение 5-и секунд после появления надписи Ready поместите на весы
// предмет, вес которого вам известен
// Запомните значение set_scale - это ваш калибровочный коэффициент
// HX711.DOUT - pin #A1
// HX711.PD_SCK - pin #A0
HX711 scale(A1, A0);
float myScale;
float myRatio;
void setup() {
  Serial.begin(9600);
  scale.set_scale();
  scale.tare();
  Serial.println("Ready");
  delay(5000);
  myScale = scale.get_units(10);
  myRatio = myScale/970;      // В этой строке необходимо заменить число на
// известный вам вес в граммах
  Serial.print("set_scale: ");
  Serial.println(myRatio);
}
void loop() {
}

```

ДОДАТОК В

ПРОГРАММНЫЙ КОД ВИВОДУ МАСИ НА ЕКРАН

```

#include <LiquidCrystal_I2C.h>           // Подключаем библиотеку OLED_I2C
для работы со шрифтами и графикой

#include <HX711.h>                       // Подключаем библиотеку HX711
для работы с АЦП hx711

HX711 scale(A1, A0);

LiquidCrystal_I2C lcd(0x3F,16,2);       // Указываем в какие пины
подключен АЦП hx711 (DT(DOUT) - pin A1, SCK - pin A0)

float myScale;                          // Инициализируем переменную в
которой будем хранить текущий вес

void setup() {
    lcd.init();                         // Обновляем содержимое экрана
    lcd.backlight(); // Включаем подсветку дисплея
    scale.set_scale(24.15);             // Устанавливаем калибровочный
коэффициент
    scale.tare();                       // Сбрасываем весы на 0
    myScale = round(scale.get_units(10));
    // Запоминаем среднее значение без груза, но с учетом тары
    lcd.setCursor(0, 0);               // Устанавливаем курсор в начало 1 строки
    lcd.print("0.00");                 // Выводим на дисплей
}

void loop() {
    scale.power_up();                  // Отключаем энергосберегающий
режим hx711

    if (round(scale.get_units(10)) != myScale) { // Условие выполняется, если
округленный до граммов вес отличается от последнего

        myScale = scale.get_units(20);
        lcd.setCursor(0, 0);
    }
}

```

```
if (myScale > 0.5) // Если вес не ничтожно мал
    lcd.print(myScale); // то выводим его
else // иначе
    lcd.print("0.00"); // выводим нули
    myScale = round(myScale); // Запоминаем округленный
текущий вес (для сравнения)
}
scale.power_down(); // Включаем энергосберегающий
режим hx711
delay(500); // Пауза на половину сек.
}
```