Вінницький національний технічний університет
(повне найменування вищого навчального закладу)
Факультет інтелектуальних інформаційних технологій та автоматики
(повне найменування інституту, назва факультету (відділення))
Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

Бакалаврська кваліфікаційна робота на тему:

ПРОГРАМНО-АПАРАТНИЙ МОДУЛЬ АВТОМАТИЧНОЇ ГОДІВНИЧКИ ДЛЯ ТВАРИН

Виконав: студент 4 курсу, групи 1КН-21б
спеціальності 122 – Комп'ютерні науки
(шифр і назва напряму підготовки, спеціальності)
Подолян І. Р.
(прізвище та ініціали)
Керівник: к.т.н., доцент кафедри КН
Озеранський В.С.
(прізвище та ініціали)
«04» червня 2025 р.
Рецензент: к.т.н., доцент кафедри АІІТ
Барабан М.В.
(прізвище та ініціали)
«05» червня 2025 р.

Допущено до захисту
Завідувач кафедри КН
д.т.н., проф. Яровий А.А.
(прізвище та ініціали)
«06» червня 2025 р.

Вінниця ВНТУ – 2025 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра Комп'ютерних наук
Рівень вищої освіти перший (бакалаврський)
Галузь знань – 12 «Інформаційні технології»
Спеціальність – 122 «Комп'ютерні науки»
Освітньо-професійна програма – «Комп'ютерні науки»

ЗАТВЕРДЖУЮ
Завідувач кафедри КН
проф., д.т.н. Яровий А.А.
"05" травня 2025 року

ЗАВДАННЯ
НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Подоляну Іоанну Романовичу

1. Тема роботи: Програмно-апаратний модуль автоматичної годівнички для тварин.
керівник роботи: Озеранський Володимир Сергійович, к.т.н., доцент
затверджені наказом вищого навчального закладу "20" березня 2025 року № 97

2. Строк подання студентом роботи 30 травня 2025 року

3. Вихідні дані до роботи :
мова програмування - об'єктно-орієнтована;
кількість циклів годування - не менше 2раз на добу;
можливість примусового ввімкнення;
максимальна ємність корму – до 1 кг;

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):
вступ, обґрунтування доцільності розробки програмно-апаратного модуля автоматичної годівнички для тварин, розробка апаратної частини модуля автоматичної годівнички для тварин, розробка програмної частини модуля автоматичної годівнички для тварин, список використаних джерел; додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень):
схема загального алгоритму функціонування автоматичної годівнички для тварин; структура модуля автоматичної годівнички для тварин; загальна UML-діаграма класів

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдан прийн
1-3	Озеранський В.С., к.т.н., доцент		

6. Дата видачі завдання 05 травня 2025 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1.	Обгрунтування доцільності розробки програмно-апаратного модуля автоматичної годівнички для тварин	05.05.2025р.	07.05.2025р.	
2.	Розробка апаратної частини модуля автоматичної годівнички для тварин	08.05.2025р.	11.05.2025р.	
3.	Розробка програмної частини модуля автоматичної годівнички для тварин	12.05.2025р.	15.05.2025р.	
4.	Розробка інструкції користувача.	16.05.2025р.	26.05.2025р.	
5.	Висновки та аналіз отриманих результатів	27.05.2025р.	29.05.2025р.	
6.	Оформлення матеріалів до захисту БКР	02.06.2025р.	04.06.2025р.	

Студент _____ (підпис) Подолян І.Р. (прізвище та ініціа

Керівник проєкту (роботи) _____ (підпис) Озеранський І (прізвище та ініціа

АНОТАЦІЯ

УДК 621.374.415

Подолян І.Р. Програмно-апаратний модуль автоматичної годівнички для тварин. Бакалаврська кваліфікаційна робота складається з 68 сторінки формату А4, на яких є 38 рисунків, список використаних джерел містить 21 найменування.

В роботі обґрунтовано доцільність розробки програмно-апаратного модуль автоматичної годівнички для тварин. Проведено аналіз класифікації сучасних технологій автоматичного годування тварин. Описано принцип роботи автоматичної годівниці. Обгрунтовано вибір складових для створення пристрою автоматичної годівниці. Описано бібліотеки, які використовує автоматична годівничка для забезпечення нормального функціонування. Здійснено розробку і описано процес виготовлення програмно-апаратного модуля автоматичної годівниці.

Програмна частина реалізована на базі створеного алгоритма роботи автоматичного пристрою годування тварин. Програмна реалізація виконана на мові програмування Wiring для середовища Arduino IDE.

Ключові слова: Arduino, годівничка, двигун, Arduino IDE, давач, Wiring.

ABSTRACT

Podolian I. Hardware-Software Module of an Automatic Pet Feeder. The bachelor's qualification work consists of 68 A4 pages, including 68 figures and a list of 21 references.

The work justifies the feasibility of developing a hardware-software module for an automatic pet feeder. It includes an analysis of the classification of modern technologies for automatic pet feeding. The operating principle of the automatic feeder is described, and the selection of components for building the device is substantiated. The libraries used by the feeder to ensure proper functionality are also presented.

The development and manufacturing process of the hardware-software module of the automatic feeder is described in detail. The software part is implemented based on a custom-designed algorithm for the operation of the automatic feeding device. The software implementation is written in the Wiring programming language for the Arduino IDE environment.

Keywords: Arduino, feeder, motor, Arduino IDE, sensor, Wiring.

ЗМІСТ

ВСТУП

Актуальність теми. Автоматична годівниця для домашніх улюбленців – корисна річ. Вона призначена для того, щоб нагодувати тварину без господаря або для того, щоб суворо дотримуватися графіку годівлі, якщо, наприклад, того вимагають рекомендації ветеринарного фахівця. Автогодівниці бувають різні, і кожен тип пристрою має свої переваги і недоліки.

Автоматичні годівниці-миски хороші для тих випадків, коли власник планує бути відсутнім вдома один або кілька днів. Як правило, такі пристрої працюють від батарейок, тому домашня тваринка не залишиться без їжі, якщо електроенергію раптом відключать.

Якщо в раціоні вашої кішки чи собаки є вологий корм, треба вибирати автогодівницю з холодоелементом. Але не можна залишати такий корм у разі тривалої відсутності господаря: навіть охолоджений вологий продукт зберігається не більше доби.

Ще один вид автоматичних годівниць – пристрої із таймінгом відкриття відсіку. Зазвичай у них є велика ємність для корму, тому такі годівниці підходять для тих випадків, коли планується тривала відсутність. Але у них є один істотний мінус: при такому зберіганні корму з доступом повітря в продукт потрапляють бактерії, він залежується, і все це може надати їжі неприємного запаху або зробити її непридатною для вживання. Коли корм насипають у миску з упаковки, то він частково перемішується, що призводить до вентиляції і позбавляє від затхлості.

Актуальність роботи полягає в тому, що якщо залишити корм у звичайній годівниці, він ризикує бути з'їденим в першу ж годину. А потім кіт чи собака після переїдання буде погано себе почувати. Тим більше, що поїдання великої кількості сухого корму загрожує серйозними проблемами зі здоров'ям, якщо тваринка після цього п'є мало води. Тому розробка автоматизованої годівниці є дуже важливою задачею.

Метою бакалаврської кваліфікаційної роботи є автоматизація процесу годування домашніх тварин.

Об’єктом дослідження є процес автоматизованого годування домашніх тварин.

Предметом дослідження є програмні та апаратні засоби організації автоматизованого годування домашніх тварин.

Для досягнення поставленої мети необхідно виконати такі задачі:

- обґрунтувати доцільність розробки програмно-апаратного модуля автоматичної годівнички для тварин;
- розробити апаратну частину програмно-апаратного модуля автоматичної годівнички для тварин;
- розробити програмну частину програмно-апаратного модуля автоматичної годівнички для тварин;
- виконати тестування модуля та провести аналіз отриманих результатів.

Результати дослідження впроваджено в роботу ТОВ «ІТІ».

1 ОБҐРУНТУВАННЯ ДОЦІЛЬНОСТІ РОЗРОБКИ ПРОГРАМНО-АПАРАТНОГО МОДУЛЯ АВТОМАТИЧНОЇ ГОДІВНИЧКИ ДЛЯ ТВАРИН

1.1 Аналіз сучасних технологій автоматизованого годування тварин

Автоматичні годівниці для домашніх тварин бувають двох видів: механічні, в яких корм просто висипається з резервуара в міру спустошення миски, та електронні з вбудованим таймером, що дозволяє видавати харчування тваринкам строго по годинах, та вагами, які відміряють необхідну кількість корму. Правильно налаштована електронна годівниця в потрібний час подасть вихованцю звуковий сигнал (можна також записати власний голос) і видасть йому обід. Таким чином, йдучи на роботу, можна не хвилюватися про те, що пухнастий друг виявиться голодним або, навпаки, переїсть. Втім, якщо господар їде у відрядження або у відпустку, все-таки краще попросити друзів відвідувати домашніх тваринок хоча б 1 раз на 2 дні – раптом годівниця вийде з ладу [1].

Робота є актуальною, тому що може вирішити проблеми людей і тварин завдяки розробленій системі, яка більш функціональна та дешевша існуючих, тому вона буде ширше застосовуватися. Сутність бакалаврського дослідження – це розроблення вдосконаленої та універсальної системи, спрямованої на полегшення процесу догляду за домашніми тваринами шляхом автоматизованого годування та поїння, яке може відбуватися дистанційно завдяки керуванню через мобільний застосунок.

Підставою вибору даної теми стали домашні улюбленці, які дуже часто страждають через відсутність часу господарів на якісний догляд. А також недосконалі існуючі системи, які потребують доповнення, а саме: збільшення функціональності пристрою та застосунку, та зниження вартості.

Оскільки технології продовжують розвиватися, домашня автоматизація стає все більш популярною серед домовласників. Завдяки можливості контролювати різні аспекти власного дому за допомогою смартфона або голосової команди, домашня автоматизація стала основним рішенням для тих, хто шукає зручності,

ефективності та розкоші. Серед усіх пристроїв, які може містити система Смарт-будинку або Розумного будинку окреме місце займають годівнички для домашніх улюбленців.

Ми живемо в часи дивовижних технологій, здатних спростити життя не тільки нам, а й нашим улюбленцям. Користь для психіки людини від домашніх тварин складно переоцінити. Якби не вони, ми б просто загрузли в рутині і щоденних клопотах. Ми часто залишаємо собак і кішок на самоті, вирушаючи на роботу або навчання. Щоб улюбленці не відчували себе покинутими і не сумували, для них існує безліч різних інтерактивних іграшок, як керованих дистанційно за допомогою смартфону, так і працюючих у повністю автоматичному режимі [1].

Так, наприклад німецька компанія RelaxoPet почала виготовляти чудесний пристрій RelaxoPet PRO (рис. 1.1), який за допомогою спеціальних звукових коливань здатний заспокоїти собак, кішок, коней і птахів, коли вони починають нервувати через вплив різних зовнішніх факторів (гучних і яскравих феєрверків, проїзду в громадському транспорті і т.п.) [2].

Рисунок 1.1 – Інноваційна система для заспокоєння тварин RelaxoPet PRO

Звуковий тренажер в буквальному сенсі змінює сигнал тривоги у тварин на заспокійливі звуки, чутні тільки їм. Таким чином, новинка допомагає вашим тваринам заспокоїтися і заснути.

На сьогоднішній день одним із старапів є новинка являє собою годівницю для домашнього улюбленця і підходить для використання як кішками, так і собаками. Управляється пристрій за допомогою спеціального додатку, розробленого для операційних систем iOS та Android. Через смартфон можна встановити графік харчування для тварини і розмір одноразової порції їжі, що з'їдається. Далі машина буде поповнювати годівницю строго за встановленими параметрами. Обсяг резервуара для зберігання сухого корму становить близько 20 чашок, а розмір миски з їжею - 4 чашки. Вода зберігається окремо в спеціальному диспенсері і автоматично поповнюється, коли датчики сигналізують про її відсутність. Обсяг такого сховища складає 6л, а одноразово в миску можна набрати 0,8л [2].

Завдяки вбудованій камері можна залишатися на зв'язку зі своїм улюбленцем віддалено, перебуваючи у будь-якій точці світу. А мікрофон дозволить спілкуватися з ним на відстані. Мало того, можна навіть почути, що він відповість [2]. Пристрій має прив'язку до аккаунту користувача на Amazon і дозволить віддалено замовляти сухий корм прямо через додаток (рис. 1.2).

Рисунок 1.2 – Стартап автоматизованої годівнички для тварин

Про дату виходу на світові ринки наразі такої годівнички нічого не відомо. Орієнтовна роздрібна вартість пристрою очікується на рівні 290$ [2].

Автоматична годівниця для котів та невеликих собак Petory Automatic Pet Feeder F01 (рис. 1.3) - це найсучасніший та зручний спосіб контролю харчування та підтримки здоров'я домашнього вихованця. Головні переваги пристрою полягають у тому, що завдяки автоматичному порційному подаванню корму за заданою власником програмою, можна встановити правильний повсякденний режим харчування домашнього улюбленця, не допускаючи перегодовування; позбутися ранніх підйомів, щоб його погодувати; і не перейматися його ситістю, якщо вас часто не буває вдома. Все, що необхідно - це насипати сухий корм у контейнер пристрою та задати програму годування, а про інше подбати електроніка [3].

Рисунок 1.3 – Годівничка Petory Automatic Pet Feeder

Автоматична годівниця для домашніх тварин - ідеальне рішення для тих, хто має напружений графік роботи, або для тих, хто хоче стежити за звичками годування свого вихованця. За допомогою автоматизованої годівниці для домашніх тварин можна налаштувати графік годування для свого вихованця та видавати корм у певний час доби. Можна навіть стежити за харчовими звичками свого вихованця за допомогою смартфона або планшета, переконавшись, що він їсть потрібну кількість їжі [3].

1.2 Принцип роботи автоматичної годівниці для тварин

Типові автогодівниці для домашніх тварин мають два різні відсіки. Верхня частина є перевернутою пластиковою або скляною секцією в якій міститься корм. Дані контейнери бувають різних розмірів і можуть містити до 500 грамів їжі. Друга секція автоматичної годівниці представлена у вигляді миски із пластмаси або кераміки, що містить корм для тварини. Кожна така годівниця проста в установці та поставляється з інструкцією.

Майже всі автоматичні годівниці для тварин використовують програмований таймер, який дозволяє точно визначити, коли і скільки їжі тварина отримуватиме. Автоматична годівниця з таймером містить до 5 кг корму для собак або до 2 кг для кішок і може бути запрограмована, щоб видавати їжу 3 рази на день. Харчування може бути запрограмоване на роздачу від 1/4 до 3 чашок їжі за один раз. Тому, якщо домашня тварина маленького розміру, то можна легко стежити за здоровим раціоном вихованця [4].

Існують інші види автоматичних годівниць для свійських тварин. До таких відносяться автогодівниці, які використовують силу тяжіння для заповнення миски кормом. Вихованцям у будь-який час доступна їжа з лотка: вихід із відсіку для зберігання не блокується, їжа випадає в міру її споживання. Така автогодівниця сприяє тому, щоб вихованець завжди мав доступ до їжі, проте її використання може сприяти переїданню та бути небезпечним для здоров'я тварин.

Автогодівниця для кішок дозволяє дотримуватися дробового харчування та дозовано видавати їжу за часом. Алгоритм дії апарату простий: засипати в контейнер дозатора корм, виставляти таймер і записати голосове звернення до кішки за наявності цієї функції (рис. 1.4).

Автогодівниця для собак подібна за принципом дії з автогодівницею для котів. Головними відмінностями будуть: більший обсяг корму, що засипається, і механізм дії більше захищений від пошкоджень [4].

Автогодівниця для акваріума призначена для триразового годування протягом тривалого часу. Автогодівниця для риб дозволить зберегти корм від впливу води,

повітря та сонячного світла. Від господаря потрібно засипати корм, налаштувати таймер і перевірити працездатність батарейок або інших джерел енергії, що подають.

Рисунок 1.4 – Зовнішній вигляд автоматичної годівнички

Автогодівниця для птахів призначена як для домашніх тварин, так і для виробництва на фермах. Господарю чи фермеру необхідно засипати корм у спеціальний відсік і щільно закрити кришку, щоб забезпечити збереження корму [4].

Головні критерії, яким має відповідати автогодівниця такі.

Годівниця повинна кріпитися у зручному місці для господаря та вихованця, тому варто звернути увагу на складання та кріплення.

Налаштування автогодівниці повинно бути просте у використанні і не бентежити власника.

Цей аксесуар повинен бути якісним. Тварини схильні ламати, кусати і дряпати годівниці, тому важливо переконатися у міцності матеріалу. Контейнери та миски повинні кріпитися міцно. Пристрій не повинен мати проводок, що стирчать, або деталей, які зможуть нашкодити вихованцю.

Автогодівниця повинна відповідати за збереження корму. Засипана їжа не повинна злипатися або псуватися. Корм повинен бути захищений від вологості та сонячних променів [4].

Будь-яка автогодівниця повинна передбачати відповідний для тварини режим роботи, щоб не перегодувати вихованця.

У сукупності всіх факторів господар знайде найкращу автогодівницю для свого вихованця.

Привчити вихованця до автогодівниці досить просто, якщо робити це правильно (рис. 1.5).

Рисунок 1.5 – Зовнішній вигляд автоматичної годівнички для котів

По-перше, годівницю необхідно поставити в те місце, де до цього у домашньої тварини стояли миски з їжею. Це допоможе улюбленцю швидше розібратися з використанням нового атрибуту [5].

По-друге, налаштувати таймер годівниці на той час, коли вихованцю слід приймати корм, або на той час, коли домашній улюбленець звик вживати їжу. Перевірити наповнюваність контейнерів та працездатність механізму.

По-третє, після всіх приготувань покликати домашню тварину і показати, як працює механізм автогодівниці.

По-четверте, дати улюбленцю самому спробувати скористатися автогодівницею.

Після успішного навчання використанню автогодівниці, можна без остраху залишати домашніх улюбленців самих у будинку навіть на тривалий час. Пристрій бере на себе відповідальність за регулярне годування тварин, що значно спрощує життя власникам і створює комфортні умови для самих тварин.

Варто зазначити, що не всі домашні улюбленці потребують періоду адаптації до нової системи. Наприклад, акваріумні риби зовсім не помітять, хто саме їх годує — господар чи автоматизований механізм. Для них головне, щоб корм подавався регулярно й у потрібній кількості. Така ж ситуація і з деякими птахами або гризунами, які не мають емоційної прив’язаності до господаря та до процесу годування [5].

Запровадження автоматичної годівниці в побуті позитивно впливає як на домашніх тварин, так і на їхніх власників. Улюбленець завжди отримує їжу вчасно, незалежно від зайнятості чи місцезнаходження господаря. Тварина залишається нагодованою, спокійною та задоволеною, що позитивно впливає на її здоров’я та поведінку.

Для власника ж це означає менше стресу та більше свободи. Тепер не потрібно поспішати додому з роботи чи відмовлятися від подорожей. Більше не виникає потреби просити знайомих, родичів або сусідів зайти погодувати кішку, собаку, рибок або папугу. Автоматична система годування бере всі ці обов’язки на себе, забезпечуючи стабільність та комфорт як для тварини, так і для її господаря.

Використання автогодівниці привчить тварин до дисципліни та допоможе правильно вибудувати режим харчування.

1.3 Висновки

В даному розділі було розкрито поняття автоматизованого годування тварин. Проаналізовані аналоги для автоматизованого годування тварин, які є на сучасному ринку. Описано принцип дії автоматизованих годівничок для тварин. Виявлено їх переваги та недоліки.

2 РОЗРОБКА АПАРАТНОЇ ЧАСТИНИ ПРОГРАМНО-АПАРАТНОГО МОДУЛЯ АВТОМАТИЧНОЇ ГОДІВНИЧКИ ДЛЯ ТВАРИН

2.1 Вимоги до обладнання та вибір елементів для автоматичної годівниці

Для реалізації автоматичної годівниці знадобиться ряд компонентів для електричної частини:

- мікроконтролер Arduino;
- двигун з прохідним валом;
- драйвер двигуна;
- енкодер;
- кнопка;
- контролер TP4056
- акумулятори 18650 3,7В.

Для механічної частини годівнички необхідні такі компоненти:

- поліпропіленова трубка діаметром 110мм;
- поліпропіленова трубка діаметром 50мм;
- поліпропіленовий перехідник на 900;
- металева трубка діаметром 7мм;
- роліпропіленова пластинка товщиною 1,5мм;
- термоклей.

Розглянемо детальніше ці компоненти.

Arduino Nano — це повнофункціональний мініатюрний пристрій на базі мікроконтролера ATmega328 (Arduino Nano 3.0) або ATmega168 (Arduino Nano 2.x), адаптований для використання з макетної плати. За функціональністю пристрій схожий на Arduino Duemilanove, і відрізняється від нього розмірами, відсутністю роз'єму живлення, а також іншим типом (Mini-B) USB-кабелю. Arduino Nano (рис. 2.1) розроблено і випускається фірмою Gravitech [6].

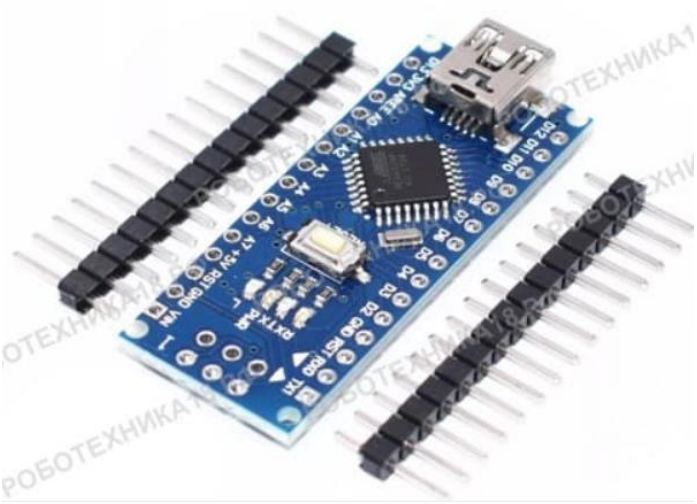

Рисунок 2.1 – Зовнішній вигляд контролера Arduino Nano

Головна відмінність цієї мініатюрної плати полягає у відсутності гнізда для зовнішнього джерела живлення, натомість використовуються V_{IN}. Коли йдеться про створення мініатюрного пристрою, розмір Arduino Nano v3 ATmega328 / ATmega168 відіграє вирішальну роль при виборі платформи.

Технічні характеристики контролера Arduino Nano (рис. 2.2):

- мікроконтролер: ATmega328;
- тактова частота: 16 МГц;
- напруга логічних рівнів: 5 В;
- вхідна напруга живлення: 7-12 В;
- портів введення-виведення загального призначення: 22;
- максимальний струм з піна введення-виведення: 40 мА;
- максимальний вихідний струм піна 3.3V: 50 мА;
- максимальний вихідний струм піна 5V: 800 мА;
- портів з підтримкою ШІМ: 6;
- портів, підключених до АЦП: 8;
- розрядність АЦП: 10 біт;
- Flash-пам'ять: 32 КБ;
- EEPROM-пам'ять: 1 КБ;
- SRAM-пам'ять: 2 КБ;
- габарити: 18×45 мм.

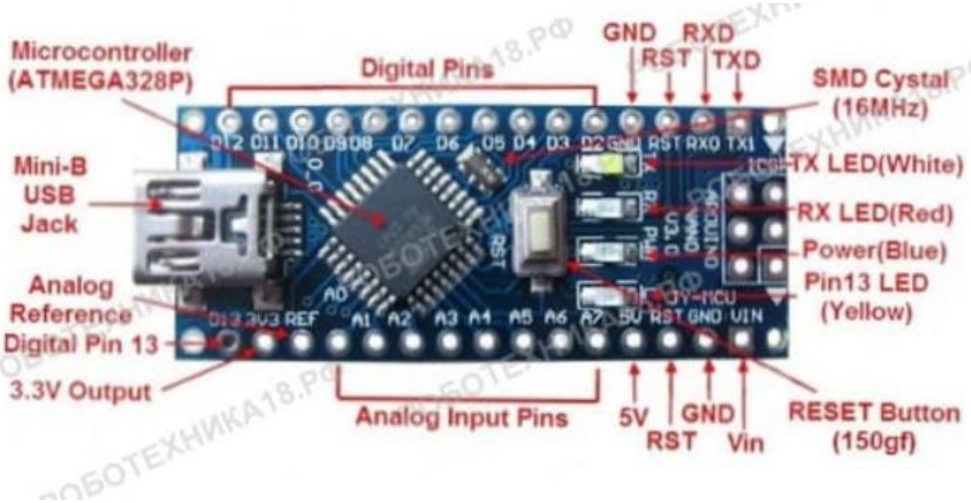

Рисунок 2.2 – Структура контролера Arduino Nano

Для керування автоматичною годівницею потрібен певний діапазон регулювання швидкості, точність підтримки швидкості обертання приводу. Звідси випливає, що необхідним є застосування двигуна постійного струму (ДПС) з редуктором.

Невеликий колекторний двигун постійного струму з редуктором DC3V-6V (рис. 2.3). Застосовується в аматорській робототехніці, має гарний крутний момент. Входить до комплекту більшості платформ-шасі для самостійного складання роботів на колесах. Редуктор має пластикові шестерні, що полегшує конструкцію, але потрібно враховувати це при розрахунку навантажень [7].

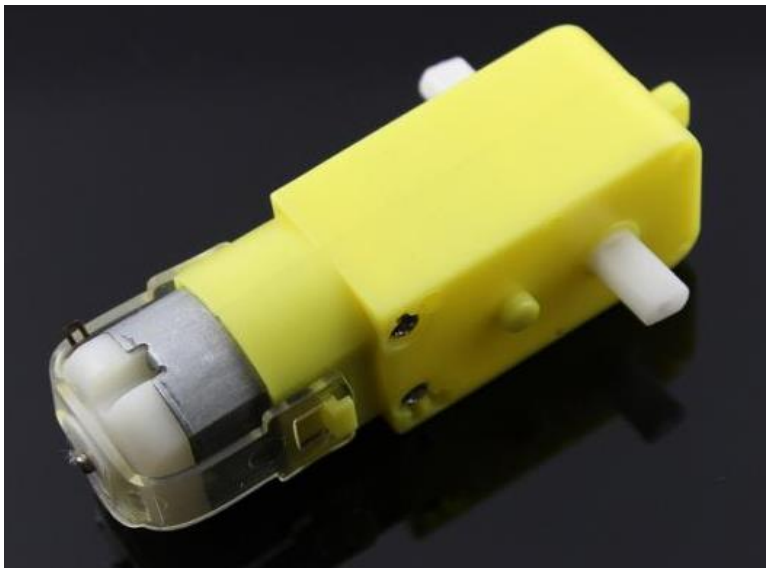

Рисунок 2.3 – Колекторний двигун постійного струму з редуктором

Основні характеристики:

- напруга живлення: 3-12В (рекомендована 3-6В);
- струм холостого ходу: 70 мА (при напрузі 3В);
- струм під навантаженням (максимальний): 250 мА (при напрузі 3В);
- момент, що крутить: 800 г/см (при напрузі 3В);
- швидкість обертання без навантаження 150-180 об/хв (при напрузі 6В);
- передаточна кількість редуктора: 1:48;
- діаметр осі: 5мм;
- габарити: 64х20х20 мм.

Драйвер двигунів MX1508 — це аналог драйвера двигунів L298N. Пристрій створений на основі мікросхеми MX1508, а принцип роботи драйвера досить простий: всередині пристрою знаходиться два транзистора, а при зміні полярності напруги відбувається зміна напрямку обертання двигунів. За допомогою такого драйвера можна створювати мобільних роботів, автономні автомобілі на основі мікроконтролерів, а також інші роботизовані пристрої з механічними модулями.

Модуль двигунів MX1508 (рис 2.4) має стандартний інтерфейс, за допомогою якого легко та швидко можна підключитися до пристроїв, створених на основі Arduino або інших мікроконтролерів. Живлення модуля здійснюється за допомогою керуючого пристрою або від зовнішньої батареї. Дана плата також дозволяє можливість ШІМ-управління для плавної зміни швидкості обертання двигуна. MX1508 може управляти двома двигунами постійного струму або 4-дротяними двофазними кроковими двигунами, в тому числі дозволяє регулювати швидкість обертання чи міняти напрям обертання, здійснювати реверс [8].

Характеристики:

Мікросхема: MX1508;

Напруга живлення модуля: 2-10 В;

Напруга вхідного сигналу: 1.8-7 В;

Робочий струм: 1.5 А, піковий до 2.5 А;

Розміри: 25х21х6 мм;

Вага: 3 г.

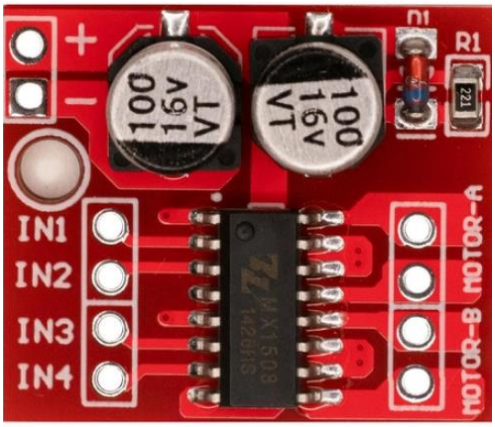

Рисунок 2.4 – Драйвер двигунів MX1508

Драйвер двигуна L298N для Arduino (рис. 2.5) використовується для управління двома малопотужними колекторними двигунами постійного струму або малопотужним 4-х дротяним двофазним кроковим двигуном. Практичне застосування: управління двигунами невеликих колісних роботів або двигунами пересувних іграшок.

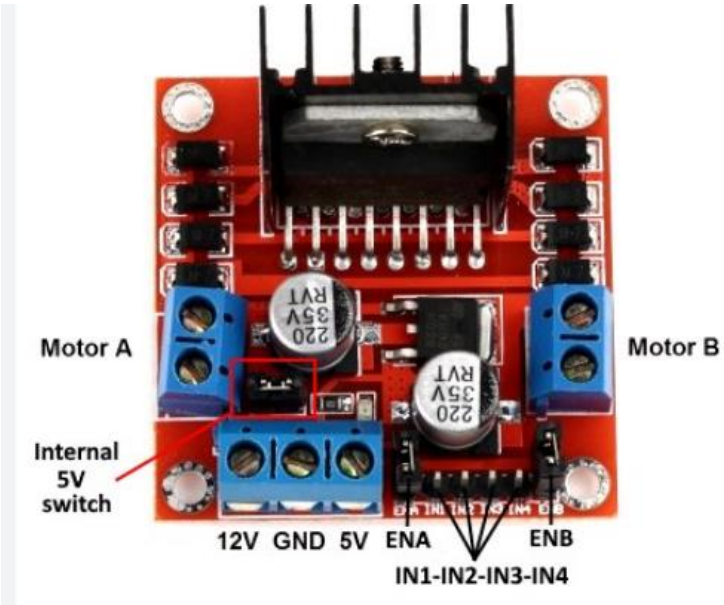

Рисунок 2.5 – Драйвер двигунів L298

Для закріплення модуля на плоскій поверхні в платі передбачено один монтажний отвір.

Управління драйвером L298N здійснюється або від контролера Arduino, або від іншого мікропроцесорного керуючого пристрою. Мікросхема модуля L298N

працює за принципом Н-моста і використовуються для зміни полярності живлення мотора, що дає можливість реверсу.

Живлення драйвера здійснюється або від контролера Arduino, або іншого мікропроцесорного керуючого пристрою, або зовнішнього джерела живлення (блоку живлення, батареї). Напруга живлення становить 2 - 9 В постійного струму.

Характеристики:

- мікросхема: L298N;
- особливість: Н-міст, можливість реверсу;
- напруга живлення: 2 - 9 В постійного струму;
- керуючий сигнал: 1,8 - 7 В постійного струму;
- максимальний споживаний струм двигунів: до 1,5 А;
- розміри: 25 х 21 х 6 мм;
- вага: 3 г.

Оптичний енкодер H206 (рис. 2.6) дуже зручний в підключенні до Arduino, тому що має на борту всю розв'язку, що дозволяє живити його від 3,3 до 5 вольт. Завдяки наявності у схемі тригера Шмітта, вихідний сигнал повністю «цифровий». Також є світлодіод для індикації розмикання лінії. Він буде корисний при налаштуванні взаємного положення енкодера та крильчатки осі двигуна [9].

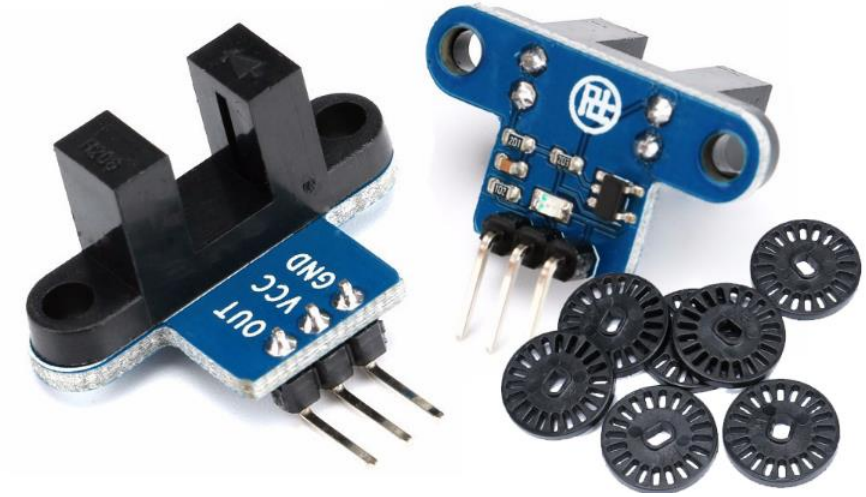

Рисунок 2.6 – Оптичний енкодер H206

Кнопки для Arduino (рис. 2.7) складаються з невеликої текстолітової плати, на якій розпаяні: тактова кнопка 12*12 мм, притягуючий резистор і 3 піна для

підключення до Arduino. Також в комплект входять ковпачки для кнопок різних кольорів. Всього в комплекті 5 модулів кнопок та 5 різнокольорових ковпачків [10].

Рисунок 2.7 – Кнопки для Arduino

Контролер заряду літій-їонних акумуляторів TP4056 (рис. 2.8) - містить контролер заряду Li-Ion акумуляторів TP4056. Мікросхема має індикацію процесу заряду та сама відключає акумулятор при досягненні напруги на ньому 4,2В. У момент заряду світиться червоний світлодіод, коли батарея буде повністю заряджена, засвітиться зелений світлодіод, червоний при цьому згасне.

Процес заряджання акумулятора ідентичний зарядці мобільного телефону, яскраві світлодіоди сповіщають про закінчення заряджання акумулятора.

Модуль підходить для зарядки літій-іонних і літій-полімерних (Li-Ion, Li-Po) акумуляторів на 3,7В. Будь то акумулятори від мобільного телефону або батареї типорозмір 18650, які застосовуються в батареях для ноутбуків [11].

Подати напругу на пристрій можна двома способами: через роз'єм, міні USB, або шляхом паяння дротів, минаючи роз'єм міні USB.

Характеристики:

- вхідна напруга: 4.5В-5.5В;
- напруга повного заряду: 4.2В;
- струм заряду: 1А;
- вхідний роз'єм: mini USB (+ місця для підпаювання проводів);
- робоча температура: -10..+85;
- розміри модуля: 5 х 19 х 25 мм.

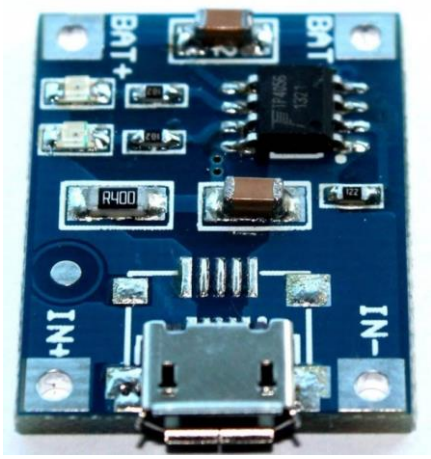

Рисунок 2.8 – Контролер заряду літій-їонних акумуляторів TP4056

Акумулятор 18650 Li-Ion LiitoKala (рис. 2.9) призначений для забезпечення працездатності портативних пристроїв різного призначення з високим постійним споживанням струму, зокрема в електроінструментах, потужних ліхтарях, електронних сигаретах, дронах тощо [12].

Ці акумулятори мають тривалий термін роботи та зберігають свої властивості в екстремальних умовах із кількістю циклів перезаряджання понад 500. Акумулятор 18650 Li-ion завдяки високій щільності струму забезпечує гарантовану стабільну напругу впродовж циклу розряджання, одночасно має низький саморозряд 5—7%. Без "ефекту пам'яті", це дає можливість заряджати його без попереднього розряджання.

Рисунок 2.9 – Літій-їонний акумулятор 18650

Для автоматичної годівнички ще знадобиться тримач на 2 акумулятора типорозміру 18650 (рис. 2.10) та макетна плата (рис. 2.11) для розміщення всіх електронних компонентів [13].

Рисунок 2.10 – тримач на 2 акумулятора типорозміру 18650

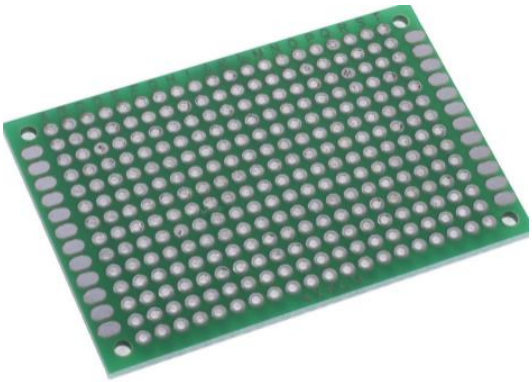

Рисунок 2.11 – Макетна плата

Для виготовлення корпусу годівнички необхідні пластикові комплектуючі, показані на рисунку 2.12

Рисунок 2.12 – Пластикові деталі корпусу годівнички

2.3 Розробка схеми автоматичної годівнички на платформі Arduino

Монтажна схема автоматичної годівниці на Arduino — це детальний план з'єднання всіх апаратних компонентів системи, який забезпечує її повноцінну та стабільну роботу відповідно до закладеної логіки. Вона включає схематичне зображення підключення центрального мікроконтролера (Arduino) з виконавчими пристроями, модулями зв’язку, датчиками та іншими периферійними елементами.

Завдяки цій схемі реалізується чітка послідовність дій: подача електроживлення, зчитування показників з датчиків, прийняття рішень згідно з алгоритмом і подальше керування виконавчими механізмами, такими як електромотори. Важливу роль відіграє правильне підключення елементів до відповідних цифрових або аналогових входів/виходів плати Arduino, адже це впливає на точність виконання команд і стабільність функціонування пристрою.

Крім основного контролера та двигуна, у монтажну схему входять:

- кнопка для локального керування;
- енкодер для контролю кількості обертів двигуна для подачі корму;
- драйвер двигуна для управління двигуном постійного струму;
- контролер заряду для акумулятора 18650.

Правильне складання монтажної схеми дозволяє уникнути коротких замикань, перевантажень у ланцюгах та помилок у керуванні. Саме тому на початковому етапі розробки пристрою схема є фундаментом для побудови працездатної автоматизованої системи годування.

Для подальшої роботи розробимо монтажну схему автоматичної годівнички для тварин на базі плати Arduino.

Відмінність між монтажною схемою та електричною принциповою схемою полягає в їхньому призначенні, способі подання інформації та рівні деталізації. Електрична принципова схема відображає логіку роботи електронного пристрою — вона показує, як елементи з’єднані з точки зору електричних зв’язків, але без прив’язки до реального розташування компонентів. Така схема містить умовні позначення електронних елементів (резисторів, конденсаторів, мікросхем,

контактів), а також вказує, які пін-коди задіяні, як передається сигнал, і як проходить струм по колу.

Натомість монтажна схема (іноді її ще називають схемою з’єднань або фізичною схемою) показує реальне розташування компонентів на макетній платі або друкованій платі (PCB), а також способи з’єднання елементів між собою — проводами, доріжками, конекторами. Вона більше орієнтована на зручність складання пристрою й допомагає під час практичної реалізації проекту.

Монтажна схема автоматичної годівниці наведена на рисунку 2.13.

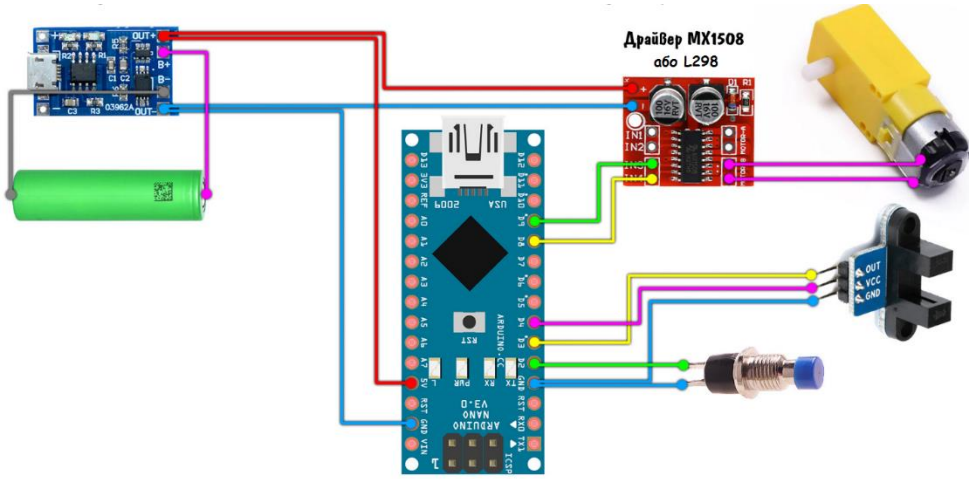

Рисунок 2.13 – Монтажна схема автоматичної годівниці

Можна створити схему автоматичної годівниці без використання акумулятора 18650 та контролера заряду TP4056. Схема такої годівниці подана на рис. 2.14

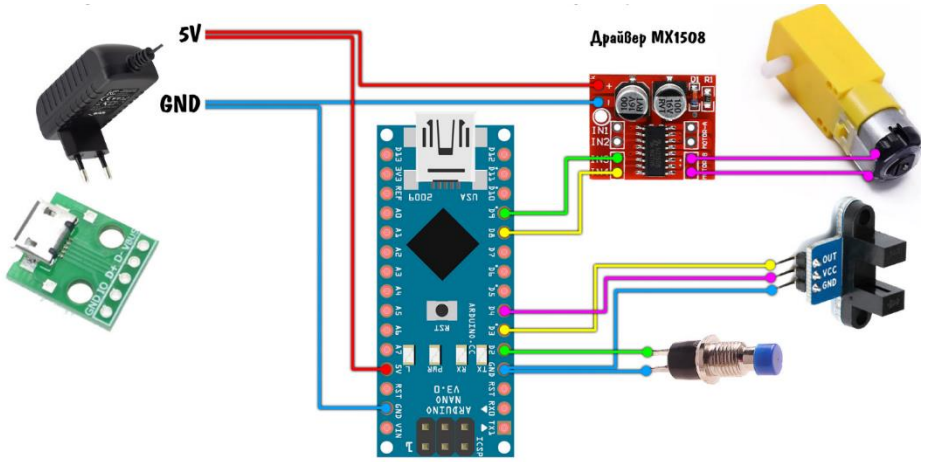

Рисунок 2.14 – Монтажна схема автоматичної годівниці з зовнішнім блоком живлення

Отже після зборки механічної частини годівниці та приєднання до неї електричних компонентів маємо пристрій, як показано на рисунках 2.15-2.19

Рисунок 2.15 – Механізм подачі корму – вид зверху

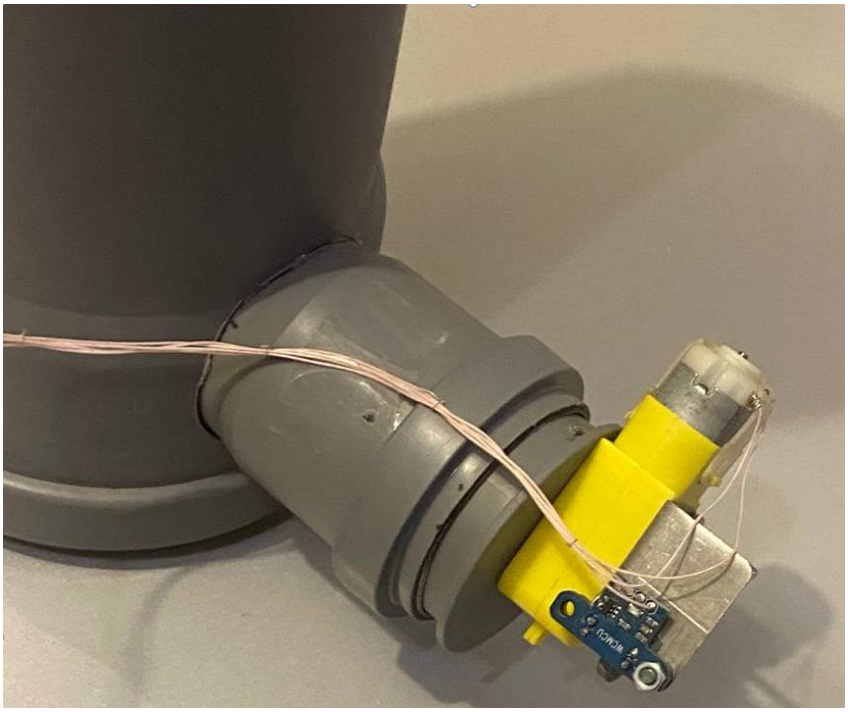

Рисунок 2.16 – Блок зі встановленим двигуном годівнички

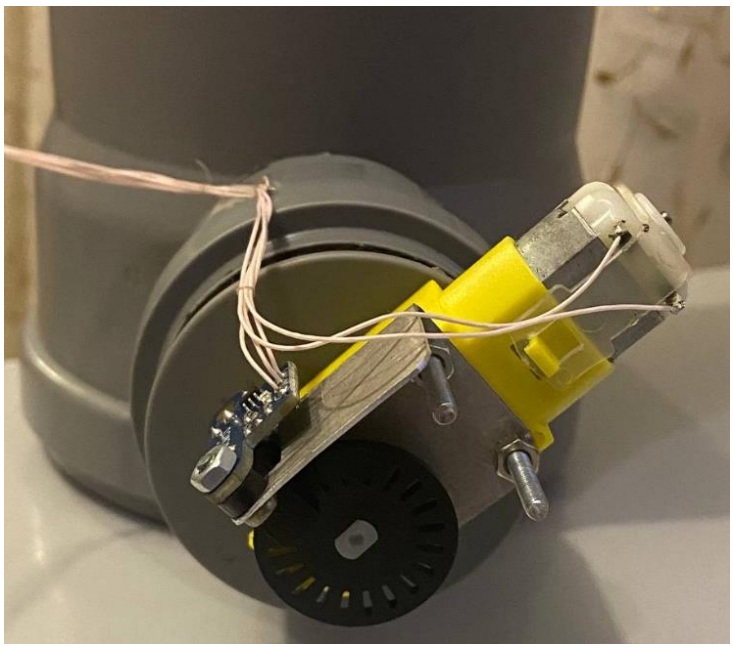

Рисунок 2.17 – Розміщення оптичного енкодера на прохідній осі двигуна

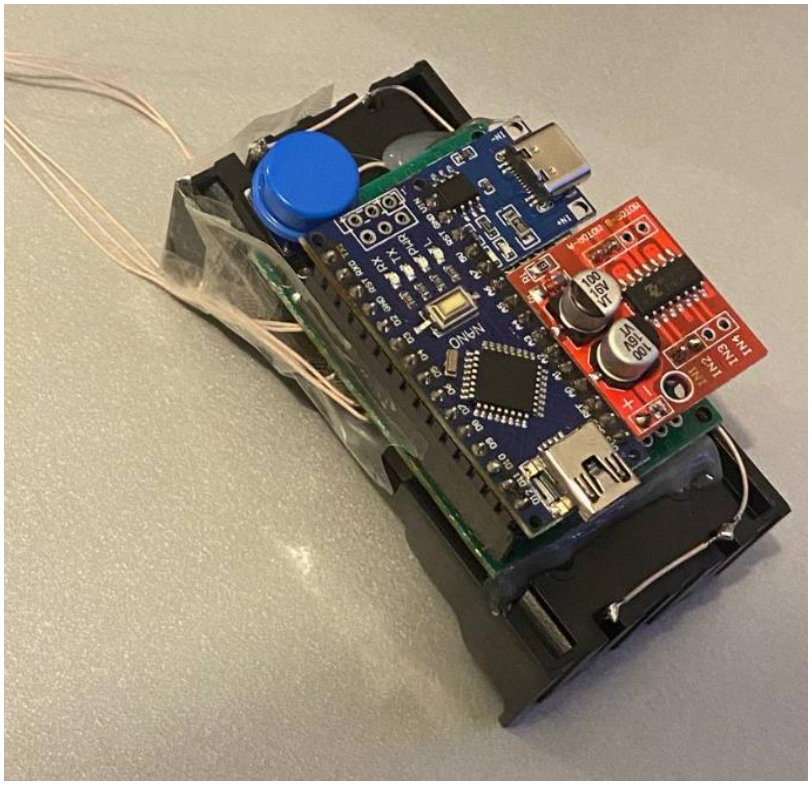

Рисунок 2.18 – Готовий макет Блоку електронного управління годівниці

Рисунок 2.19 – Зовнішній вигляд автоматичної годівниці на платформі Arduino

Після того як ми успішно завершили збирання електронної та механічної частин програмно-апаратного модуля автоматичної годівнички для тварин, наступним етапом є розробка та завантаження відповідного програмного забезпечення у мікроконтролер Arduino. Саме програмна частина відповідає за логіку функціонування пристрою, послідовність його дій та реакцію на сигнали датчиків і зовнішні умови. Без правильно написаного коду навіть найякісніша апаратна реалізація залишиться просто набором деталей, не здатним виконувати жодну функцію.

Крім того, у процесі програмування важливо реалізувати захист від помилок, таких як блокування роботи у випадку втрати сигналу, зависання двигуна або перевищення тривалості роботи. Також варто передбачити можливість ручного

керування пристроєм, наприклад, через кнопки, щоб можна було протестувати механізм без очікування на встановлений час спрацювання.

2.4 Розробка алгоритму роботи автоматичної годівнички для тварин

Отже, з урахуванням усього вищесказаного, розробимо алгоритм роботи автоматичної годівнички, який схематично зображено на рисунку 2.20.

Рисунок 2.20 – Схема алгоритму роботи автоматичної годівнички

Після подачі живлення на пристрій, плата Arduino автоматично ініціалізується та виконує початкову перевірку працездатності основних компонентів системи. Зокрема, здійснюється тестовий запит до контролера двигуна для перевірки зв'язку та готовності шнекового механізму до виконання подальших команд. Якщо зв'язок

встановлено успішно, система переходить до режиму очікування наступного моменту годування.

У програмному коді мікроконтролера заздалегідь задано інтервали годування – наприклад, три рази на добу з рівномірним інтервалом у 8 годин. Для цього використовується вбудований таймер або модуль реального часу (RTC), який дозволяє точно відстежувати поточний час навіть при тривалому знеструмленні пристрою.

У визначений час система активує двигун, який приводить у дію шнековий механізм. Цей механізм обертається протягом заданого інтервалу часу, подаючи порцію корму для тварини у ємність під годівничкою. Після завершення обертання двигун вимикається, і пристрій знову переходить у режим очікування до наступного циклу.

Таким чином, автоматична годівничка працює у циклічному режимі, самостійно забезпечуючи своєчасну подачу корму, що значно полегшує догляд за домашніми улюбленцями та підвищує комфорт їх утримання.

2.5 Висновки

В даному розділі було детально описано принцип роботи автоматичної годівниці для тварин. Детально описано вибір складових для створення пристрою автоматичної годівниці. Розроблено алгоритм роботи автоматичної годівнички для тварин.

3 РОЗРОБКА ПРОГРАМНОЇ ЧАСТИНИ ПРОГРАМНО-АПАРАТНОГО МОДУЛЯ АВТОМАТИЧНОЇ ГОДІВНИЧКИ ДЛЯ ТВАРИН

3.1 Обгрунтування вибору середовища розробки

Для апаратної реалізації автоматичної годівниці було обрано контролер Arduino Nano. А отже і програмування даного контролера ми будемо виконувати за допомогою спеціалізованого середовища Arduino IDE.

Перш ніж почати писати програму та програмувати контролер, потрібно виконати деяку попередню підготовку. На персональному комп'ютері потрібно установити драйвер для віртуального COM-порту (послідовний інтерфейс) і середовище програмування / розробки для Arduino.

Спочатку потрібно встановити драйвер для мікросхеми перетворювача USB–UART. Якщо використовується оригінальна плата Arduino, або високоякісний клон – необхідно встановити драйвер мікросхеми FT232RL, хоча дану операцію можна і не проводити, оскільки цей драйвер вже входить до комплекту інсталятора середовища розробки Arduino. У випадку використання сумісних плат, необхідно встановити драйвер перетворювача мікросхеми, що встановлена в ній (здебільшого використовуються мікросхеми CH340, CP2102 або PL2303). Мікроконтролер перетворювача інтерфейсів запрограмований таким чином, що в менеджері пристроїв плата позначена як віртуальний COM-порт (віртуальний послідовний інтерфейс). При підключенні плати комп'ютер отримує додатковий COM-інтерфейс. ОС Windows не робить різниці між фізичним та віртуальним COM-інтерфейсом. Новий інтерфейс отримує наступне вільне ім'я, наприклад, COM7.

Тепер встановимо середовище програмування Arduino, яке базується на програмі Processing. Processing – це просте інтегроване середовище розробки, розроблене спеціально для користувача, який не глибоко володіє програмуванням, але, тим не менше, потребує написання власної програми. Останню стабільну версію середовища можна завантажити з офіційного сайту проєкту https://www.arduino.cc/ у вигляді встановлювача, або ж архіву з розпакованою версією, яка не потребує встановлення (портативна версія). Важливо регулярно виконувати оновлення

програмного забезпечення, оскільки в середовище постійно додаються нові бібліотеки, драйвери а також виправляються наявні помилки.

Інтегрована середовище розробки (IDE – Itegrated Development Environment) Arduino – це кроссплатформенний додаток на Java, що включає в себе редактор коду, компілятор і модуль передачі прошивки в плату.

Середовище розробки спроєктоване для програмування новачками, які не знайомі близько з розробкою програмного забезпечення. Мова програмування аналогічна використовуваному в проєкті Wiring. Строго кажучи, це C/C ++, доповнена деякими бібліотеками та з певними обмеженнями (наприклад недоступні можливості багатопоточного програмування та відсутня об'єктна модель C++). Програми обробляються з допомогою препроцесора, а потім компілюються за допомогою вільно розповсюджуваного компілятора AVR-GCC.

3.2 Установка середовища Arduino IDE

Після встановлення або копіювання з архіву програми Arduino можна запустити файл Arduino.exe. Для зручності виклику програми можна створити ярлик на робочому столі. В Arduino-IDE знаходяться різні інструменти і настройки , які полегшують спілкування з програмою Arduino (рис. 3.1).

Розглянемо докладніше середовище розробки Arduino IDE. З меню можна викликати всі функції Arduino. Під головним меню знаходиться панель інструментів, де розташовані такі команди:

- перевірка – виконується перевірка на наявність помилок в програмному коді;
- завантаження – формування файлу, який буде переданий в плату мікроконтролера та його завантаження;
- новий – створення нового файлу Arduino;
- відкрити – відкриття тексту програми;
- зберегти – збереження тексту програми;
- монітор COM порту – відкриття вбудованого командного терміналу.

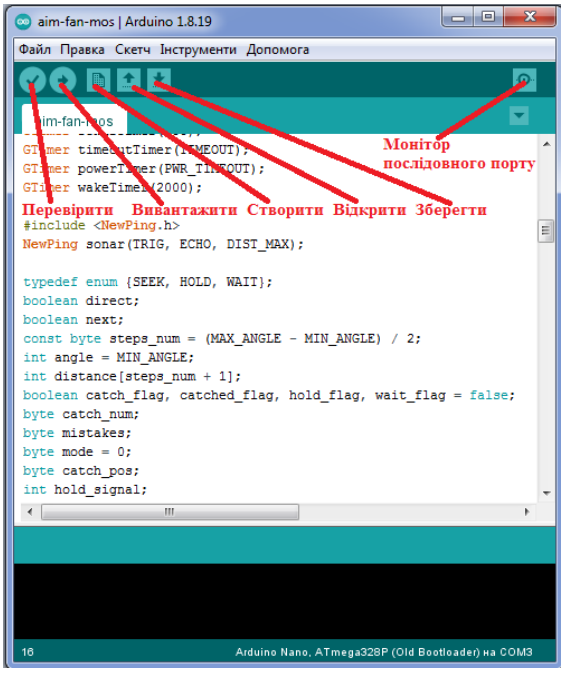

Рисунок 3.1 - Середовище розробки Arduino

До початку роботи необхідно задати вихідні установки. Для цього потрібно вибрати потрібну модель плати Arduino і використовуваний інтерфейс. У даному випадку вказана плата Arduino Nano. Якщо потрібна інша, то слід вибрати відповідну плату зі списку (рис. 3.2).

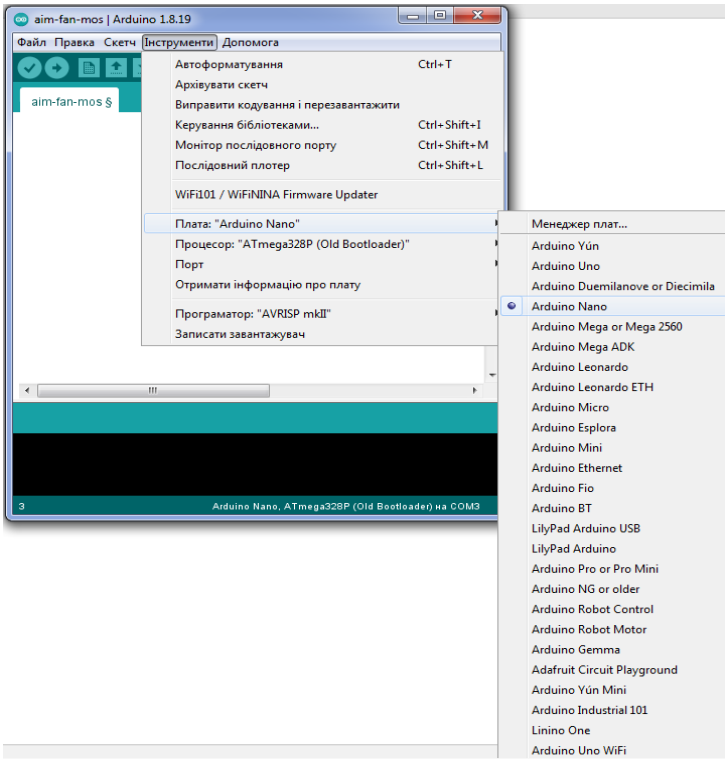

Рисунок 3.2 - Вибір плати мікроконтролера Arduino

3.3 Вибір мови програмування

Для програмування контролера Arduino було обрано мову Wiring, яка є спрощеною та адаптованою версією мови C/C++ і спеціально розроблена для використання з платформою Arduino. Основною перевагою цієї мови є її простота, зрозумілий синтаксис і висока ефективність для роботи з мікроконтролерами, навіть для користувачів без глибокого досвіду в програмуванні. Arduino IDE забезпечує зручне середовище для написання, компіляції та завантаження програм, а також містить велику кількість готових бібліотек, які суттєво спрощують роботу з різноманітними сенсорами, модулями та виконавчими пристроями. Саме тому мова Wiring була обрана як оптимальний варіант для реалізації логіки функціонування автоматичної годівнички, забезпечуючи надійність, гнучкість та простоту в реалізації алгоритмів керування.

Програму для функціонування пристрою смарт-підсвітлення для монітора написано на мові C++.

C++ було розроблено як універсальну мову зі статичними типами даних, ефективністю та переносимістю мови C.

C++ розроблена так, щоб безпосередньо та всебічно підтримувати безліч стилів програмування (процедурне програмування, абстракцію даних, об'єктно-орієнтоване програмування та узагальнене програмування).

Вона розроблена так, щоб давати програмісту свободу вибору, навіть якщо це дає можливість вибирати неправильно.

C розроблена так, щоб максимально зберегти сумісність із C, тим самим уможливлюючи легкий перехід від програмування на C.

Уникає таких особливостей, які залежать від платформи чи не є універсальними.

Не накладає жодного надлишкового навантаження на програму, яка не використовує жодних можливостей.

Розроблено так, щоб не вимагати надто ускладненого середовища програмування.

C++ вважається однією з дуже швидких мов програмування.

Переваги C++ над C:

- велика безпека;
- можливість писати узагальнений код за допомогою шаблонів;
- можливість використовувати об'єктно-орієнтований підхід;
- управління ресурсами за допомогою RAII;
- спрощення коду за рахунок перевантаження функцій та операторів;
- простіша обробка помилок за рахунок винятків.

Недоліки:

- довга компіляція;
- більший обсяг згенерованого машинного коду.

3.4 Встановлення драйвера CH340 Arduino Nano

Для початку - мікросхема CH340G використовується для передачі даних USB -> UART та в повній функціональності замінила мікросхему ATmega16U2 зберігаючи надійність та швидкість при передачі даних.

На платах перетворювач CH340 для Arduino має такий вигляд, як на рисунку 3.3.

Рисунок 3.3 – Мікросхема перетворювача CH340 для Arduino

Для інсталювання драйвера операційної системі Windows 10, Windows 8, Windows 7, треба перейти за посиланням - "Завантажити драйвер V3.5 03-2019". --- ZIP-архів.

Після того як закінчилось завантаження файлу, необхідного його відкрити та натиснути на кнопку "INSTALL (рис. 3.4)

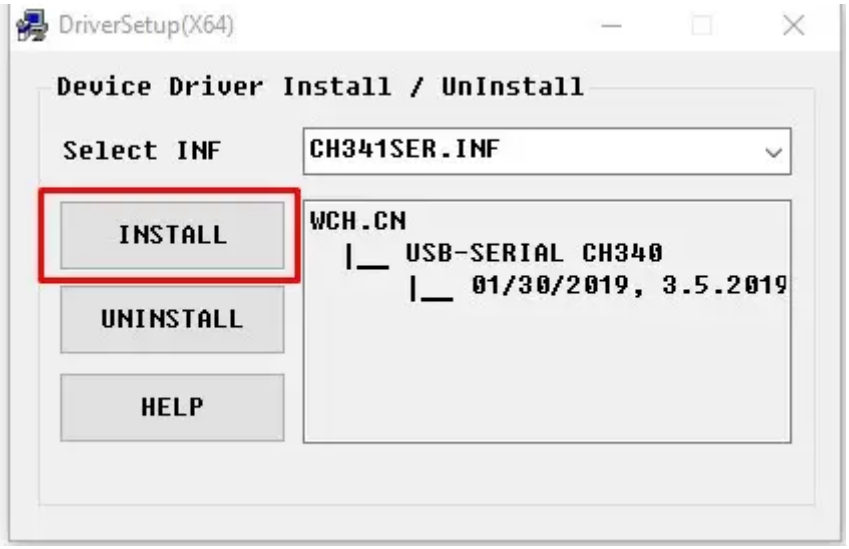

Рисунок 3.4 – Інсталювання дрейвера перетворювача CH340 для Arduino

Треба зачекати кілька секунд і драйвер буде встановлений, про що свідчить дане сповіщення (рис. 3.5)

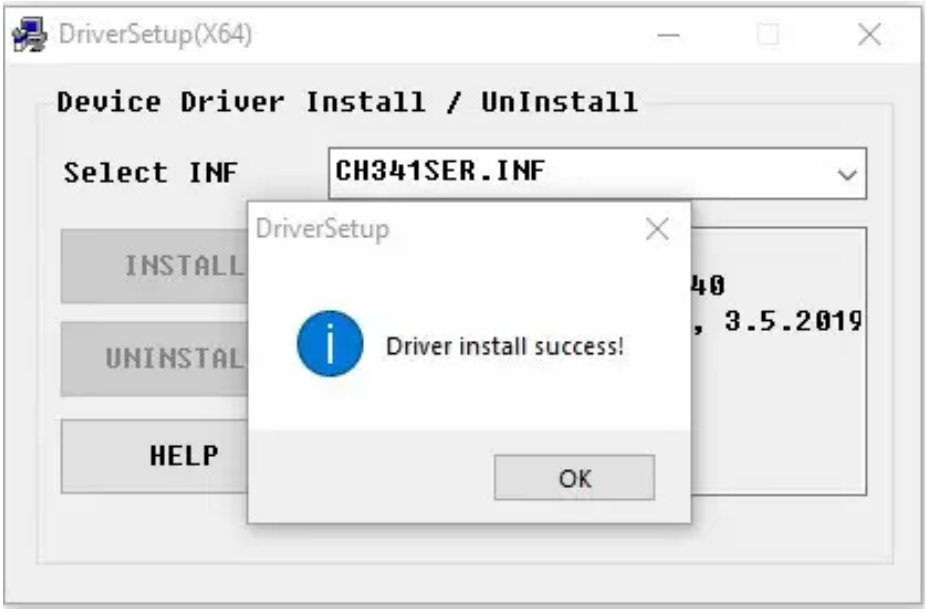

Рисунок 3.5 – Успішне встановлення дрейвера перетворювача CH340 для Arduino

Після підключення плати драйвер зробить емуляцію COM порта та починає передачу даних між комп'ютером та платою Arduino (рис. 3.6).

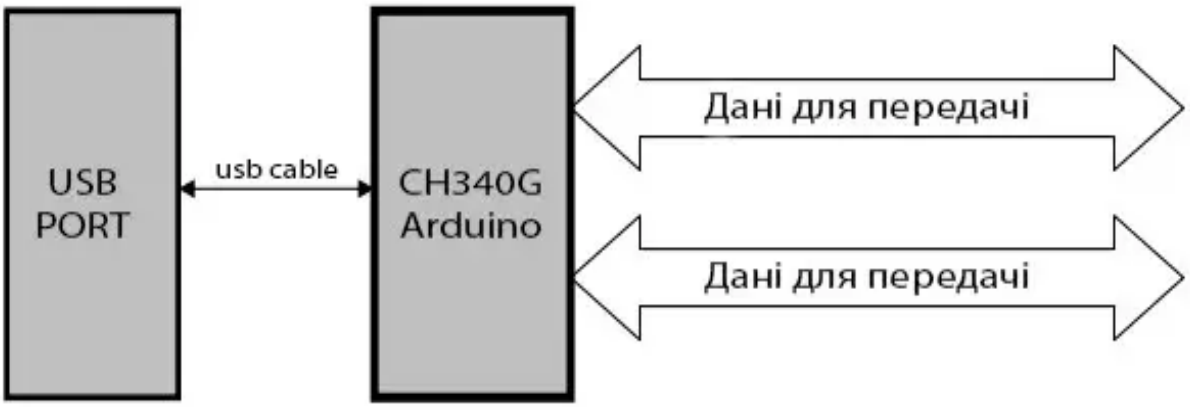

Рисунок 3.6 – Схема роботи дрейвера перетворювача CH340 для Arduino

Який саме номер займає встановлений COM-порт, відображається в Диспетчері пристроїв (рис. 3.7).

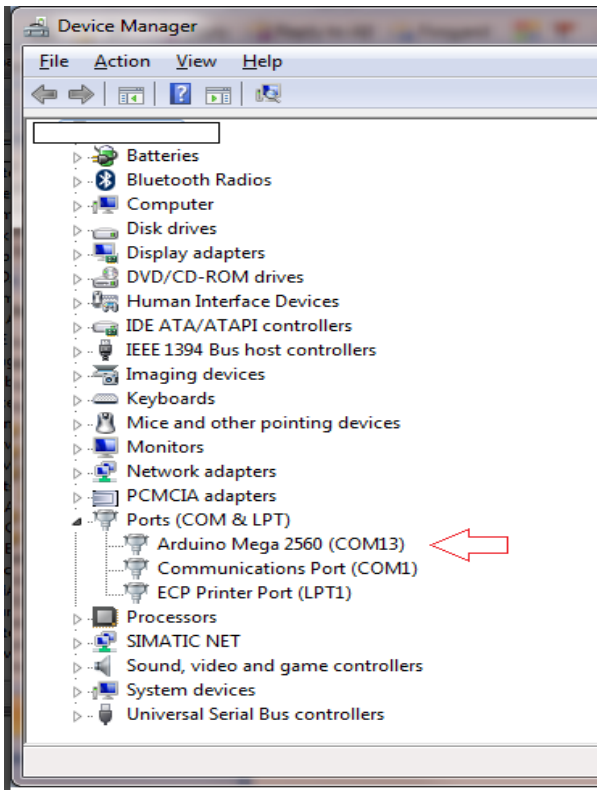

Рисунок 3.7 – Відображення плати Arduino в Диспетчері пристроїв

Отже, даний драйвер та бібліотека CH340 інсталювалися вірно і можливо починати програмувати плати Arduino. Але після підключення плати та запуску середовища програмування Arduino IDE необхідно змінити COM порт (рис. 3.8).

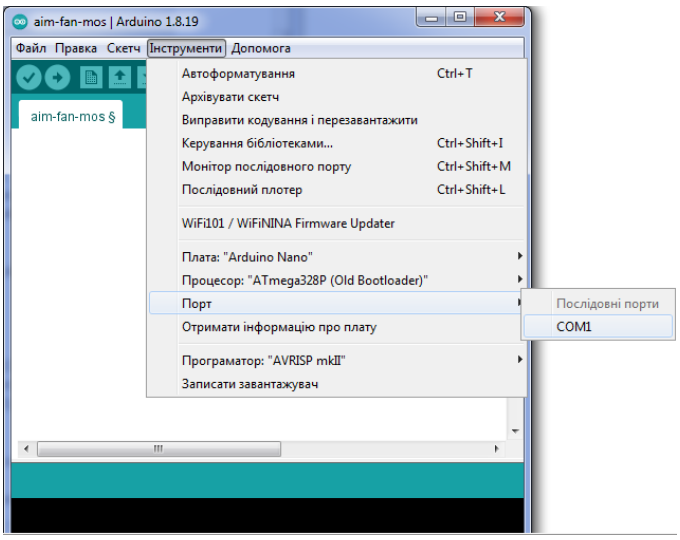

Рисунок 3.8 – Налаштування COM порта в середовищі Arduino IDE

3.5 Програмування контролера Arduino

Розглянемо склад основних компонентів редактора Arduino IDE

1. Головне меню.

Меню редактора включає в себе наступні основні елементи: файл, правка, скетч, сервіс і довідка. Розглянемо докладніше кожен з них (рис. 3.9).

У пункті Файл можна знайти команди, що відповідають за створення нової програми, читання старої, збереження її змін, а також команди для завантаження програми на мікроконтролер.

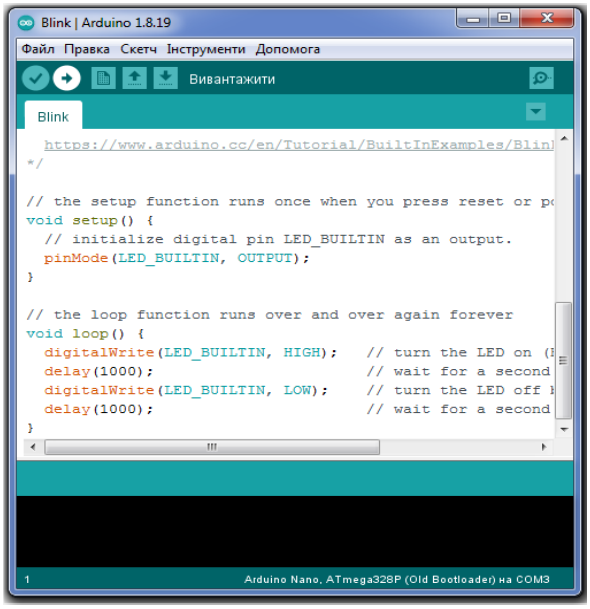

Рисунок 3.9 – Зовнішній вигляд редактора Arduino IDE

Створити - створити нову програму (скетч).

Відкрити - відкрити існуючу програму.

Папка зі скетчами - відкрити програму із заданої папки.

Приклади - відкрити приклад програми; Закрити - закрити поточне вікно.

Зберегти - зберегти зміни в раніше збережену програму.

Зберегти як - зберегти нову програму, із зазначенням імені.

Завантажити - завантажити програму в Arduino.

Завантажити за допомогою програматора - завантажити програму за допомогою програматора.

Налаштування друку – налаштування принтера.

Друк - вивід на друк коду програми.

Налаштування – налаштування редактора.

Вихід - вихід з Arduino IDE.

Пункт меню Правка містить команди, пов'язані з редагуванням тексту програми, включаючи копіювання, вставку, настройку відступів і пошук за ключовим словом.

У розділі Скетч розміщуються команди для контролю за процесом компіляції програми.

Перевірити / Компілювати - компілювати програму.

Показати папку скетчів - відкрити системну папку з програмами.

Додати файл - додати до проекту файл з даними або програмою.

Імпортувати бібліотеку - підключити до програми бібліотеку зі списку встановлених.

Пункт меню Сервіс включає в себе допоміжні функції для роботи з самим мікроконтролером.

Автоформатування - автоматична розстановка відступів, переносів рядків і т.п. Архівувати скетч - архівація папки з програмою, і збереження архіву в вказане місце.

Виправити - кодування і перезавантажити.

Монітор порту - відкрити вікно для обміну даними з мікроконтролером.

Плата - вибір поточної плати (в даному випадку Arduino Uno).

Послідовний порт - вибір порту, до якого підключений.

Нарешті, меню Довідка містить докладний опис всіх функцій самого редактора Arduino IDE, а також всілякі команди і прийоми роботи з платформою Arduino.

2. Меню іконок:

Перевірити / Компілювати програму.

Завантажити програму в Arduino.

Створити нову програму.

Відкрити існуючу програму.

Зберегти програму.

Монітор послідовного порту.

3. Вкладки

Кожна програма для Arduino може складатися з декількох файлів. Для перемикання між цими файлами служить система вкладок в редакторі. Там же, можна створити нову вкладку, і асоціювати з нею файл в папці з проектом.

4. Вікно програми

Безпосередньо, текст програми створюється і редагується в головному вікні редактора. По суті, вікно редактора являє собою типовий текстовий редактор, з підсвічуванням конструкцій коду.

5. Вікно повідомлень

У самому низу редактора Arduino IDE є невелике вікно, що служить для виведення повідомлень про проблеми, які виникають в процесі компіляції програми, або під час завантаження програми в мікроконтролер.

Програмування Arduino здійснюється на мові C ++. Нижче приведені основні конструкції і умовності мови, необхідні для успішного проходження даного курсу.

1. Кожен вираз закінчується символом; крапка з комою. наприклад:

a = b + c;

Тіло функцій і складових операторів (if, else, for, while) відокремлюється фігурними дужками(Аналогічно BeginEnd в мові Pascal). наприклад:

```
if ( a>0 )
{
   b = a+1;
}
```

Рядки відокремлюються звичайними подвійними лапками ".

Приклад: println ("some text");

Символи відокремлюються одинарними лапками: symbol = 'a';

Підключення бібліотек здійснюється за допомогою конструкції:

#include <math.h>

Кометарі в програмі починаються з символів // два прямих слеша. Приклад:

// це моя програма

Оголошення змінної в мові с ++ здійснюється за допомогою конструкції виду:

тип_змінної ім'я_змінної;

Наприклад:

int x, y; // оголошені дві змінні x і y, які мають цілий тип

Цілі числа:

byte від 0 до 255;

int від 32 768 до 32 767;

word від 0 до 65535;

long від 2 147 483 648 до 2 147 483 647.

Дробові числа:

float від 3.4028235E + 38 до 3.4028235E + 38;

double еквівалетно float в поточній версії Arduino.

Масиви

Масиви в C ++ задаються конструкцією типу:

тип_елемента імя_масивц [розмір];

Наприклад:

int numbers [10]; // задає масив з десяти цілих чисел

Рядки і символи:

char символ;

Рядки в C ++ є масиви з елементами типу char.

Наприклад:

char my_str [10]; // рядок з десяти символів

Інші типи:

void порожній тип;

boolean false або true (хибність або істина).

Оператори порівняння:

== рівність

! = Нерівність

< менше

<= Менше, або дорівнює

> більше

> = Більше, або дорівнює

Умови

```
if (a> 0) { ...
   команди, що виконуються в разі істинності умови
} else { ...
   команди, що виконуються в іншому випадку
}
```

Цикли

```
for (k = 0; k <3; k = k + 1) { ...
   команди, що виконуються на кожному кроці циклу
}
```

У дужках послідовно вказується:

- початкове значення ітератора k = 0;

- умова продовження циклу k <3 (поки ітератор менше трьох);

- дію над ітератором під час кожного кроку k = k + 1 (збільшуємо на одиницю на кожному кроці).

Функції;

- тип_функції імя_функції (аргументи)

{ команди, що виконуються в рамках функції return результат_функції;

}

тип_функції - тип значення.

Наприклад, стандартна функція sin має тип значення, що повертається float.

- ім’я_функції - будь-який рядок, що починається з букви, і містить тільки літери і символи підкреслення. аргументи - перелік аргументів, які функція використовує для своїх дій.

- результат_функції - пременися або число, що визначає значення, що повертається функції.

Нижче наведено приклад функції, що підсумовує два цілих числа.

int sum (int a, int b) { int result; result = a + b; return result; }

Мінімальна програма, яка може запустити на Arduino складається всього з двох функцій:

void setup () {

}

void loop () { }

Перша функція setup викликається тільки один раз, після перезапуску Arduino. Друга loop, викликається нескінченне число разів під час роботи Arduino.

Для зв'язку з платою Arduino можна використовувати спеціальну програму моніторингу послідовного порту (Serial Monitor), вбудовану в програмне забезпечення Arduino . Монітор послідовної шини відображає дані, що посилаються в плату Arduino через віртуальний послідовний порт за допомогою USB. Для відправки даних вибирається швидкість передачі зі списку, відповідна значенню Serial.begin в скетчі. Потім необхідно ввести текст і натиснути кнопку Send або Enter.

Плата Arduino Nano має один послідовний порт (також відомий як UART або USART): Serial, що використовується для зв'язку з комп'ютером через USB. Він пов'язаний з цифровими виводами 0 (RX) і 1 (TX). Таким чином, під час роботи

послідовного порту, виводи 0 і 1 не можуть використовуватися в якості цифрових входів або виходів.

Для забезпечення зв'язку плати Arduino з комп'ютером або іншими пристроями використовується клас Serial. Клас - це абстрактний тип даних. За допомогою класу описується деяка сутність (її характеристики і можливі дії). Наприклад, клас може описувати змінні, параметри і функції послідовного порту. Клас Serial містить близько 20 функцій. Розглянемо деякі.

Функції для роботи з послідовним портом плати Arduino:

1. if (Serial)

Параметри – ні.

Значення, що повертаються – boolean: повертає true, якщо вказаний послідовний порт готовий до роботи.

Опис: дозволяє перевірити готовність певного послідовного порту.

2. Serial.begin(speed) Serial.begin(speed, config)

Параметри:

speed: швидкість в бітах на секунду (бодах) - long

config: задає кількість біт даних, перевірку парності и стопові біти:

SERIAL_5N1

SERIAL_6N1

SERIAL_7N1

SERIAL_8N1 (за замовченням)

SERIAL_5N2

SERIAL_8E2

SERIAL_7O2

Опис: задає швидкість передачі даних по послідовному інтерфейсу в бітах в секунду (бодах). Для взаємодії з комп'ютером слід використовувати одну з попередньо встановлених швидкостей обміну: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 або 115200.

Другий аргумент цієї функції необов'язковий. Він дозволяє налаштувати кількість біт даних, перевірку парності і степових біти. За замовчуванням, посилка складається з 8 біт даних, без перевірки парності, з одним стоповим бітом.

3. Serial.end ()

Параметри – ні.

Значення, що повертаються – немає.

Опис: функція розриває послідовний зв'язок, після чого виводи RX і TX знову можна використовувати як виводи загального призначення. Для відновлення послідовного з'єднання необхідно використовувати функцію

Serial.begin ().

4. Serial.available()

Параметри – ні.

Значення, що повертаються: кількість байт, доступних для зчитування.

Опис: повертає кількість байт (символів) доступних для зчитування з буфера послідовного порту. Під символами розуміються дані, які вже прийняті і зберігаються в послідовному приймальному буфері (який може зберігати максимум 64 байта).

5. Serial.read()

Параметри – ні.

Значення, що повертаються: перший байт прийнятих даних (або -1, якщо таких нема) – int.

Опис: зчитує дані, що надходять по послідовному інтерфейсу.

6. Serial.print(val)

Serial.print(val, format)

Параметри: val: значення, яке необхідно вивести - будь-який тип даних; format: визначає систему числення (для цілочисельних типів), а також кількість десяткових знаків після коми (для чисел з плаваючою точкою).

Значення, що повертаються: кількість виведених байт. Зчитування цього значення не є обов'язковим.

Опис: функція виводить через послідовний порт заданий ASCII текст у вигляді, зрозумілому для людини. Ця команда може мати кілька різних форм.

При виведенні числа кожній його цифрі відповідає один ASCII-символ. Дробові числа теж виводяться у вигляді ASCII-цифр, при цьому після коми за замовчуванням залишається два десяткових знака. Байти виводяться у вигляді окремих символів, а символи і рядки виводяться без змін – «як є».

Приклад:

Serial.print(78) - виведе "78"

Serial.print(1.23456) - виведе "1.23"

Serial.print('N') - виведе "N"

Serial.print("Hello world.") - виведе "Hello world."

Другий параметр необов'язковий. Він задає формат виведення; цей параметр може набувати таких значень: BIN (двійкова система з основою 2), OCT (восьмерична система з основою 8), DEC (десяткова система з основою 10), HEX (шістнадцятирічна система з основою 16). Для числа з плаваючою точкою цей параметр визначає кількість десяткових знаків після коми.

Приклад:

Serial.print (78, BIN) - виведе "1001110"

Serial.print (78, OCT) - виведе "116"

Serial.print (78, DEC) - виведе "78" Serial.print (78, HEX) - виведе "4E"

Serial.println (1.23456, 0) - виведе "1"

Serial.println (1.23456, 2) - виведе "1.23"

Serial.println (1.23456, 4) - виведе "1.2346"

Отже, далі завантажуємо та розпаковуємо середовище Arduino IDE.

Підключаємо мікрокомп'ютер Arduino до роз’єму USB. Драйвер (послідовний інтерфейс CH340) встановлюється автоматично. Якщо цього не сталося, то в папці Arduino IDE є тека Drivers з усім необхідним.

Запускаємо Arduino IDE та відкриваємо файл з прошивкою (рис. 3.10).

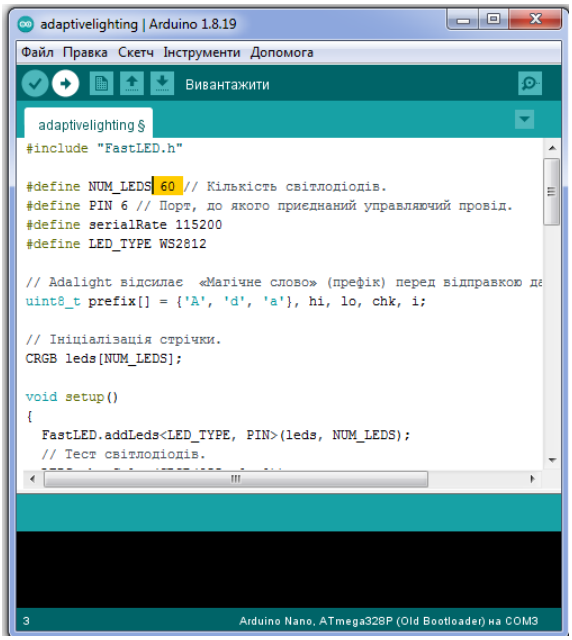

Рисунок 3.10 – Скетч для управління годівничкою

Завантажуємо програму в Arduino. Програма сама проінформує, коли завантаження буде завершено (тривалість - кілька секунд).

Тепер потрібно відключити контролер Arduino Nano від шини USB та підключити заново. Світлодіодна стрічка загориться послідовно червоним, зеленим та синім кольором – Arduino Nano активувався та готовий до роботи.

Розглянемо детальніше кожний фрагменти програми.

Блок оголошення початкових даних має вигляд:

#define FEED_PERIOD 8 // період годування в Годинах

#define FEED_SPEED 10 // швидкість обертання шнека (ум.од.)

#define MIN_SPEED 150 // мін. швидкість двигуна (зменшує час розгону)

#define INVERSE_MOTOR 0 // 0/1 інвертувати двигун

#define INVERSE_BUTTON 1 // 0 - норм. Розімкнута, 1 - норм. замкнута кнопка

#define CLEAR_TIME 400 // час заднього руху, мс (ліквідація стопора)

#define WAIT_MODE 1 // 0 - сон, 1 - затримка

Блок ініціювання пінів для зовнішніх пристроїв має вигляд:

// піни

```
#define BTN_PIN 2
#define ENC_DO 3
#define ENC_VCC 4
#define MOTOR_DIR 8
#define MOTOR_PWM 9
```

Після ввімкнення живлення відбувається початкове налаштування всіх елементів:

```
void setup() {
  if (EEPROM.read(1000) != 50) {  // перший запуск
#if (WAIT_MODE == 0)
    realPeriod = calibrateWDT();  // калібровка
#endif
    EEPROM.write(1000, 50);
    EEPROM.put(0, feedAmount);
    EEPROM.put(2, realPeriod);
  }
  EEPROM.get(0, feedAmount);
#if (WAIT_MODE == 0)
  EEPROM.get(2, realPeriod);
  sleedAmount = (float)FEED_PERIOD * 3600 / realPeriod * 1000;  // скільки разів спати по 8 сек
#endif

  pinMode(ENC_VCC, OUTPUT);
  pinMode(MOTOR_DIR, OUTPUT);
  pinMode(MOTOR_PWM, OUTPUT);
  pinMode(BTN_PIN, INPUT_PULLUP);
  digitalWrite(ENC_VCC, HIGH);
```

```
  TCCR1A = 0b00000001; // 8bit
  TCCR1B = 0b00001010; // x8 fast pwm
  attachInterrupt(0, isrHandler, INVERSE_BUTTON ? RISING : FALLING);
  delay(300);
  isrState = false;
 }
```

Блок опрацювання реакції пристрою на натиснення кнопки має ось такий вигляд:

```
 void loop() {
  if (isrState) {
   delay(1000);
   if (digitalRead(BTN_PIN) ^ INVERSE_BUTTON) {       // якщо короткий клік
    isrState = false;                                 //
   } else {                                          // якщо кнопка нажата
    digitalWrite(ENC_VCC, HIGH);                      // вмикаємо енкодер
    while (!digitalRead(BTN_PIN) ^ INVERSE_BUTTON) {  // поки кнопка нажата
     feedRoutine();                                   // обертаємо двигун
    }
    runMotor(0, 0);                                   // стоп
    feedAmount = encCounter;                          // запам’ятовуємо скільки обертів зробили
    EEPROM.put(0, feedAmount);                        // записали в епром
    encCounter = 0;                                   // скидання лічильника
    motorSpeed = MIN_SPEED;                           // швидкість в мінімум
    digitalWrite(ENC_VCC, LOW);                       // вимикаємо енкодер
   }
  }
```

Якщо не відбувалось ніяких натиснень, то в роботу включається блок спрацьовування по годиннику:

```
      if (!isrState) {                    // прокинулись по таймеру чи клікнули по кнопці
        digitalWrite(ENC_VCC, HIGH);      // вмикаємо енкодер
        while (encCounter < feedAmount) {   // поки не наберемо необхідну кількість обертів
          feedRoutine();
        }
        encCounter = 0;                   // скидання лічильника
        motorSpeed = MIN_SPEED;           // швидкість в мінімум
        runMotor(0, 0);                   // стоп
        digitalWrite(ENC_VCC, LOW);       // вимакаємо енкодер
      }
      isrState = false;
    #if (WAIT_MODE == 0)
      // ЗДОРОВЫЙ СОН
      for (int i = 0; i < sleedAmount; i++) {   // спимо по 8 сек
        LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
        if (isrState) break;              // якщо було натиснення прокидаємося і ідемо на початок loop
      }
    #else
      // Очікування
      uint32_t tmr = millis();
      uint32_t waitPrd = (uint32_t)FEED_PERIOD * 3600 * 1000;
      while (millis() - tmr < waitPrd) {
        if (isrState) break;              // якщо було натиснення – виходимо і йдемо на початок loop
      }
    #endif
    }
```

Якщо при подачі корму його частинки попадають під витки шнекового механізму і блокують його роботу, то щоб не намагатися обертати двигун далі, контроллер має зробити дві коротких спроби ввімкнути задній хід, тим самим звільнивши шнековий вал від стопору. Якщо цього не зробити, така ситуація може призвести до виходу з ладу або двигуна, або редуктора, який знаходиться всередині двигуна, або контроллера двигуна внаслідок його перегріву, або шнекового механізму (в даному макеті його витки виконані з м’якого пластику, який за допомогою термоклею кріпиться на металічний вал, як показано на рис. 3.12).

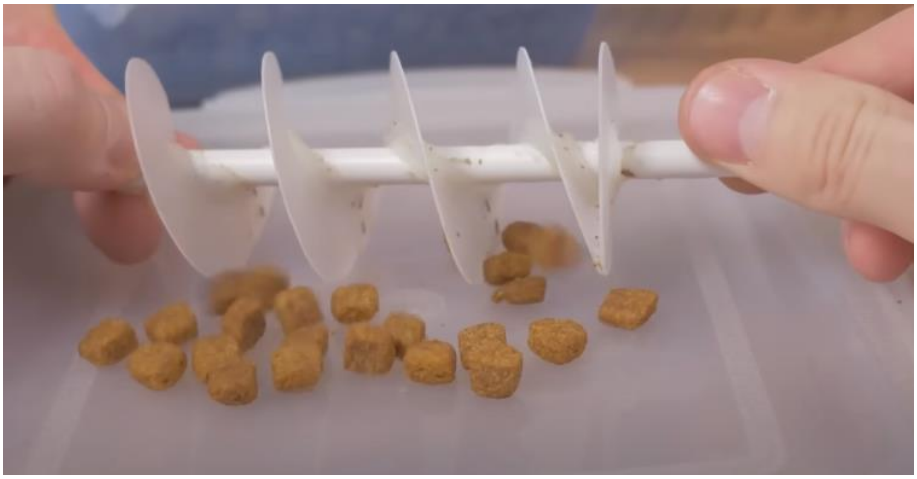

Рисунок 3.12 – Шнековий механізм годівнички

Блок програми, який виконує таку дію має вигляд:

```
void feedRoutine() {
  encTick();                          // опитування енкодера
  static uint32_t tmr;
  if (millis() - tmr >= 30) {               // управління двигуном
    tmr = millis();

    if (curEncSpeed < FEED_SPEED) motorSpeed += 2;  //  збільшити швидкість
    if (curEncSpeed > FEED_SPEED) motorSpeed -= 2;  // зменшити швидкість
    motorSpeed = constrain(motorSpeed, 0, 255);     // обмежити 0-255

    static bool stuckFlag = false;
```

```
    static uint32_t stuckTimeout;
    if (motorSpeed == 255 && curEncSpeed == 0) {     // якщо двигун на повній швидкості, але швидкість 0
      if (millis() - stuckTimeout > 1000) {         // і цей стан більше 1 сек.
        runMotor(1, 255);                          // Задній хід!
        uint32_t stuckTmr = millis();
        while (millis() - stuckTmr < CLEAR_TIME) {  // протягом CLEAR_TIME
          encTick();                               // оптування енкодера
        }
        //delay(CLEAR_TIME);
        motorSpeed = MIN_SPEED;                    // скидаємо швидкість
        stuckTimeout = millis();                   // скидаємо таймаут
      }
```

Якщо все відбувається у штатному режимі, то відпрацьовуємо функції:

```
      stuckTimeout = millis();                     // скидаємо таймаут
    }
    runMotor(0, motorSpeed);                       // обертаємо двигун
  }
}

void encTick() {
  static bool lastState;
  bool curState = digitalRead(ENC_DO);    // опитування
  if (lastState != curState) {            // якщо є зміни
    lastState = curState;
    if (curState) {                       // по фронту
      encCounter += (motorDirection ? -1 : 1);  // запам’ятовуємо поворот
    }
  }
```

```
  static uint32_t tmr;
  if (millis() - tmr >= 300) {            // таймер для розрахунку швидкості
    tmr = millis();
    static int lastPos = 0;
    curEncSpeed = encCounter - lastPos;   // сама швидкість
    lastPos = encCounter;
  }
}

void runMotor(bool dir, byte speed) {
  motorDirection = dir;
  dir = dir ^ INVERSE_MOTOR;            // інверсія, якщо треба
  digitalWrite(MOTOR_DIR, dir);         // напрям
  analogWrite(MOTOR_PWM, (dir) ? (255 - speed) : (speed));  // швидкість
}

void isrHandler() {
  if (!isrState) isrState = true;          // підняти прапорець перериваня, якщо він скинутий
}
```

Блок програми, який дозволяє виконувати зовнішнє управляння годівничкою - натискання кнопки – (рис. 3.7) має вигляд:

```
int calibrateWDT() {
  WDTCSR |= (1 << WDCE) | (1 << WDE);   // дозволяємо зовнішнє управління
  WDTCSR = 0x47;                        // таймаут ~ 2 сек
  asm ("wdr");                          // скидання
  uint16_t startTime = millis();        // зафіксували час
  while (!(WDTCSR & (1 << WDIF)));      // чекаємо таймаут
  uint16_t ms = millis() - startTime;
```

```
  WDTCSR |= (1 << WDCE) | (1 << WDE);   // дозволяємо зовнішнє управління
  WDTCSR = 0;
  return ms * 4;
}
```

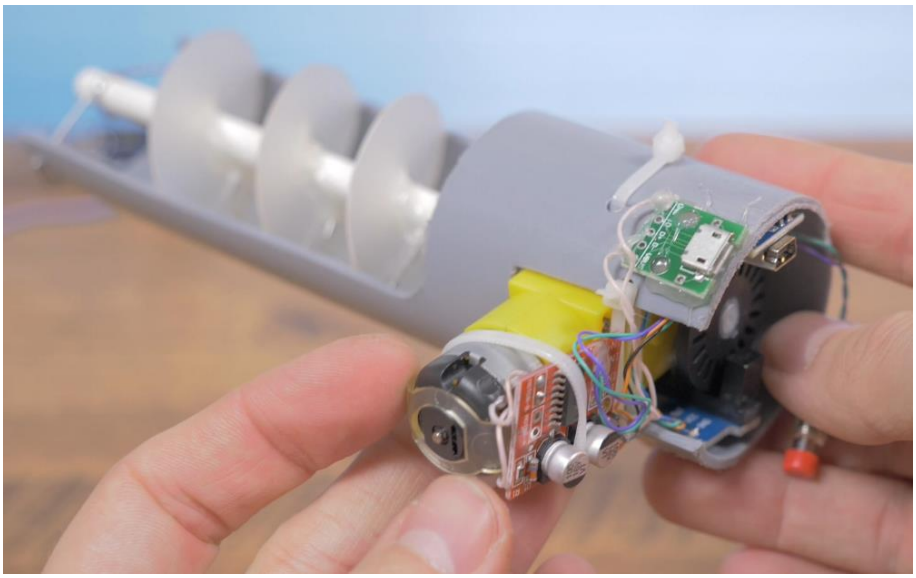

Рисунок 3.13 – Шнековий механізм годівнички з кнопкою

Отже, подаємо живлення до нашої годівниці вставивши акумулятори у тримач. Вмикача не передбачено, тому, що годівниця має бути завжди включеною. При короткому натисканні кнопки на Блоці управління (рис. 2.18) спрацьовує двигун, який активізує шнековий механізм (черв'ячний вал), щоб видати необхідну порцію корму. Об'єм порції (час спрацьовування двигуна) можна змінити, тримаючи натисненою кнопку на блоці управління. Годівничка готова до роботи.

3.6 Висновки

В даному розділі було детально описано розробку програмного коду програмно-апаратного модуля автоматичної годівнички для тварин. Описано процес інсталяції необхідного програмного забезпечення та принцип програмування контроллера Arduino. Протестовано розроблений пристрій автоматичної годівнички для тварин.

ВИСНОВКИ

Всі завдання, поставлені в бакалаврській кваліфікаційній роботі були виконані в повному обсязі. В роботі розкрито поняття автоматизованого годування тварин. Проаналізовані аналоги для автоматизованого годування тварин, які є на сучасному ринку. Описано принцип дії автоматизованих годівничок для тварин. Виявлено їх переваги та недоліки. Детально описано принцип роботи автоматичної годівниці для тварин. Детально описано вибір складових для створення пристрою автоматичної годівниці. Розроблено алгоритм роботи автоматичної годівнички для тварин.

Детально описано розробку програмного коду програмно-апаратного модуля автоматичної годівнички для тварин. Описано процес інсталяції необхідного програмного забезпечення та принцип програмування контроллера Arduino.

Розроблений програмно-апаратний модуль успішно пройшов тестування, яке підтвердило його працездатність у повній мірі.

Отже всі поставлені задачі виконано, мету роботи досягнуто.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Arduino Pro Mini [Електронний ресурс] // doc.arduino.ua – 2021. – Режим доступу до ресурсу: https://doc.arduino.ua/ru/hardware/ProMini.

2. Arduino Nano [Електронний ресурс] // arduino.ua – 2021. – Режим доступу до ресурсу: https://arduino.ua/prod166-arduino-nano-v3-0-avratmega328p-s-raspayannimi-razemami.

3. Unified Modeling Language – Вікіпедія: веб-сайт. - Режим доступу: URL: https://uk.wikipedia.org/wiki/ Unified_Modeling_Language.

4. Microsoft Visual Studio – Вікіпедія: веб-сайт. Режим доступу: URL: https://uk.wikipedia.org/wiki/Microsoft_Visual_Studio

5. Muresan D. D., Parks T. W., Adaptive Principal Components and Image Denoising, in: Image Processing, 2003, Proceedings 2003 IEEE International Conference on Image Processing (ICIP), 14-17 Sept. 2003, V. 1, pp. I-101-104.

6. Офіційний сайт середовища Visual Studio [Електронний ресурс]. Режим доступу до матеріалу: https://visualstudio.microsoft.com/ru/

7. Паралельні обчислення CUDA [Електронний ресурс]. Режим доступу до матеріалу:https://www.nvidia.com.ua/object/cuda-parallel-computing-ru.html.

8. Плата Arduino UNO [Електронний ресурс] // arduino.ua. – 2021. – Режим доступу до ресурсу: https://www.mini-tech.com.ua/arduino-uno.

9. Драйвер двигуна L298N [Електронний ресурс] // arduino.ua. – 2021. – Режим доступу до ресурсу: https://arduino.ua/prod406-draiver-dvyh-dvigatelei-na-l298n.

10. Двигун двоосьовий 1:48 [Електронний ресурс] // arduino.ua. – 2021. – Режим доступу до ресурсу: https://arduino.ua/prod662-motor-s-redyktorom-148-dvyh-osevoi.

11. Arduino. URL: https://vseplus.com/ua/article/cto-takoe-arduino-783 (дата звернення: 29.05.2025).

12. Arduino інтерфейси. URL: https://geekmatic.in.ua/ua/arduino_lesson_112 (дата звернення: 29.05.2025)

13. Основні оператори. URL: http://tinker.uamper.com/docs/programuvannya-arduino.html (дата звернення: 29.05.2025).

14. Платформа Arduino. URL: https://www.duet.edu.ua/uploads/DocS/9st9.pdf (дата звернення: 29.05.2025).

15. IDE (інтегроване середовище розробки). URL: https://apixdrive.com/ua/blog/useful/ide-i-sdk-strashnye-terminy-prostymi-slovami (дата звернення: 29.05.2025).

16. Мікроконтролери і мікропроцесорні пристрої. URL: https://elprivod.nmu . org.ua/ua/interesting/what_is_mp_mc_plc.php (дата звернення: 29.05.2025).

17. Основи мікропроцесорної техніки. URL: https://ela.kpi.ua/bitstream/123456789/27992/1/OMPT_laboratorni.pdf (дата звернення: 29.05.2025).

18. C++ 17 STL. Стандартна бібліотека шаблонів. Яцек Галовіц. 2018. ISBN: 978-5-4461-0680-6

19. Парадигми програмування. Елементи програм C++ [Electronic resource]. – Mode of access : WWW/URL : http://om.univ.kiev.ua/users_upload/15/upload/file/prog_lecture_01.pdf. – Title from the screen.

20. Ситник В. Ф., Писаревська Т. А., Єрьоміна Н. В., Краєва О. С. Основи інформаційних систем: Навчальний посібник Київ, 2005. 308 с.

21. Методичні вказівки до виконання бакалаврських кваліфікаційних робіт для студентів спеціальності 122 «Комп’ютерні науки» [Електронний ресурс] / уклад.: А. А. Яровий, О. В. Сілагін, І. В. Богач. – Вінниця: ВНТУ, 2023. – (PDF, 54 с.).

ДОДАТОК А (обов'язковий)
ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Назва роботи: Програмно-апаратний модуль автоматичної годівнички для тварин

Тип роботи: бакалаврська кваліфікаційна робота
(бакалаврська кваліфікаційна робота / магістерська кваліфікаційна робота)

Підрозділ кафедра комп'ютерних наук
(кафедра, факультет, навчальна група)

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism 16,22 %

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне)

✓ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту

☐ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.

☐ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

Яровий А.А., зав. каф. КН _____
(прізвище, ініціали, посада) (підпис)

Колесницький О.К., проф. каф КН _____
(прізвище, ініціали, посада) (підпис)

Особа, відповідальна за перевірку _____ Озеранський В.С.
(підпис) (прізвище, ініціали)

З висновком експертної комісії ознайомлений(-на)

Керівник _____ Озеранський В.С.
(підпис) (прізвище, ініціали, посада)

Здобувач _____ Подолян І.Р.
(підпис) (прізвище, ініціали)

ДОДАТОК Б
Інструкція користувача

1. Встановити годівничку на місце її розташування (рис. Б.1).
2. Поставити під кран корму ємність для висипання корму для тварин.
3. Насипати в контейнер для корму необхідний об’єм сухого корму.
4. Установити акумулятори у відсік акумуляторів.
5. Годівничка готова до роботи.
6. Для перевірки можна натиснути кнопку на електронному блоці годівнички. З крану має висипатись певна порція сухого корму.

Рисунок Б.1 – Зовнішній вигляд автоматичної годівниці на платформі Arduino

ДОДАТОК В
Лістинг програми

```
#define FEED_PERIOD 8     // період годування в Годинах
#define FEED_SPEED 10     // швидкість обертання шнека (ум.од.)
#define MIN_SPEED 150     // мін. швидкість двигуна (зменшує час розгону)
#define INVERSE_MOTOR 0   // 0/1 інвертувати двигун
#define INVERSE_BUTTON 1  // 0 - норм. Розімкнута, 1 - норм. замкнута кнопка
#define CLEAR_TIME 400    // час заднього руху, мс (ліквідація стопора)
#define WAIT_MODE 1       // 0 - сон, 1 - затримка

// піни
#define BTN_PIN 2
#define ENC_DO 3
#define ENC_VCC 4
#define MOTOR_DIR 8
#define MOTOR_PWM 9

int realPeriod;
int sleedAmount;
int feedAmount = 100;
int encCounter = 0;
int curEncSpeed = 0;
volatile bool isrState = false;
int motorSpeed = MIN_SPEED;
bool motorDirection = false;  // 0 - вперед, 1 -  назад

#if (WAIT_MODE == 0)
#include "LowPower.h"
#endif
#include <EEPROM.h>

void setup() {
  if (EEPROM.read(1000) != 50) {  // перший запуск
#if (WAIT_MODE == 0)
    realPeriod = calibrateWDT();  // калібровка
#endif
    EEPROM.write(1000, 50);
    EEPROM.put(0, feedAmount);
    EEPROM.put(2, realPeriod);
  }

  EEPROM.get(0, feedAmount);
#if (WAIT_MODE == 0)
```

```
  EEPROM.get(2, realPeriod);
  sleedAmount = (float)FEED_PERIOD * 3600 / realPeriod * 1000;  // скільки разів спати по 8 сек
#endif

  pinMode(ENC_VCC, OUTPUT);
  pinMode(MOTOR_DIR, OUTPUT);
  pinMode(MOTOR_PWM, OUTPUT);
  pinMode(BTN_PIN, INPUT_PULLUP);
  digitalWrite(ENC_VCC, HIGH);

  TCCR1A = 0b00000001; // 8bit
  TCCR1B = 0b00001010; // x8 fast pwm
  attachInterrupt(0, isrHandler, INVERSE_BUTTON ? RISING : FALLING);
  delay(300);
  isrState = false;
}

void loop() {
  if (isrState) {
    delay(1000);
    if (digitalRead(BTN_PIN) ^ INVERSE_BUTTON) {        // якщо короткий клік
      isrState = false;                                 //
    } else {                                            // якщо кнопка натиснута
      digitalWrite(ENC_VCC, HIGH);                      // вмикаємо енкодер
      while (!digitalRead(BTN_PIN) ^ INVERSE_BUTTON) {  // поки кнопка натиснута
        feedRoutine();                                  // обертаємо двигун
      }
      runMotor(0, 0);                                   // стоп
      feedAmount = encCounter;                          // запам’ятовуємо скільки обертів зробили
      EEPROM.put(0, feedAmount);                        // записали в епром
      encCounter = 0;                                   // скидання лічильника
      motorSpeed = MIN_SPEED;                           // швидкість в мінімум
      digitalWrite(ENC_VCC, LOW);                       // вимикаємо енкодер
    }
  }

  if (!isrState) {                  // прокинулись по таймеру чи клікнули по кнопці
    digitalWrite(ENC_VCC, HIGH);    // вмикаємо енкодер
    while (encCounter < feedAmount) {   // поки не наберемо необхідну кількість обертів
      feedRoutine();
    }
    encCounter = 0;                 // скидання лічильника
```

```
    motorSpeed = MIN_SPEED;       // швидкість в мінімум
    runMotor(0, 0);               // стоп
    digitalWrite(ENC_VCC, LOW);   // вимакаємо енкодер
   }
   isrState = false;

#if (WAIT_MODE == 0)
   // ЗДОРОВЫЙ СОН
   for (int i = 0; i < sleedAmount; i++) {   // спимо по 8 сек
    LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
    if (isrState) break;        // якщо було натиснення прокидаємося і ідемо на початок
loop
   }
#else
   // Очікування
   uint32_t tmr = millis();
   uint32_t waitPrd = (uint32_t)FEED_PERIOD * 3600 * 1000;
   while (millis() - tmr < waitPrd) {
    if (isrState) break;        // якщо було натиснення – виходимо і йдемо на початок loop
   }
#endif
}

void feedRoutine() {
   encTick();                              // опитування енкодера
   static uint32_t tmr;
   if (millis() - tmr >= 30) {                 // управління двигуном
    tmr = millis();

    if (curEncSpeed < FEED_SPEED) motorSpeed += 2;  //  збільшити швидкість
    if (curEncSpeed > FEED_SPEED) motorSpeed -= 2;  // зменшити швидкість
    motorSpeed = constrain(motorSpeed, 0, 255);     // обмежити 0-255
    static bool stuckFlag = false;
    static uint32_t stuckTimeout;
    if (motorSpeed == 255 && curEncSpeed == 0) {    // якщо двигун на повній
швидкості, але швидкість 0
      if (millis() - stuckTimeout > 1000) {         // і цей стан більше 1 сек.
       runMotor(1, 255);                          // Задній хід!
       uint32_t stuckTmr = millis();
       while (millis() - stuckTmr < CLEAR_TIME) {  // протягом CLEAR_TIME
        encTick();                              // оптування енкодера
       }
       //delay(CLEAR_TIME);
       motorSpeed = MIN_SPEED;                     // скидаємо швидкість
       stuckTimeout = millis();                   // скидаємо таймаут
```

```
      }
    } else {                                // якщо все в нормі
      stuckTimeout = millis();                    // скидаємо таймаут
    }
    runMotor(0, motorSpeed);                    // обертаємо двигун
  }
}
void encTick() {
  static bool lastState;
  bool curState = digitalRead(ENC_DO);    // опитування
  if (lastState != curState) {            // якщо є зміни
    lastState = curState;
    if (curState) {                     // по фронту
      encCounter += (motorDirection ? -1 : 1);  // запам’ятовуємо поворот
    }
  }
  static uint32_t tmr;
  if (millis() - tmr >= 300) {            // таймер для розрахунку швидкості
    tmr = millis();
    static int lastPos = 0;
    curEncSpeed = encCounter - lastPos;   // сама швидкість
    lastPos = encCounter;
  }
}
void runMotor(bool dir, byte speed) {
  motorDirection = dir;
  dir = dir ^ INVERSE_MOTOR;            // інверсія, якщо треба
  digitalWrite(MOTOR_DIR, dir);         // напрям
  analogWrite(MOTOR_PWM, (dir) ? (255 - speed) : (speed));  // швидкість
}
void isrHandler() {
  if (!isrState) isrState = true;       // підняти прапорець переривання, якщо він скинутий
}
int calibrateWDT() {
  WDTCSR |= (1 << WDCE) | (1 << WDE);   // дозволяємо зовнішнє управління
  WDTCSR = 0x47;                        // таймаут ~ 2 сек
  asm ("wdr");                        // скидання
  uint16_t startTime = millis();        // зафіксували час
  while (!(WDTCSR & (1 << WDIF)));      // чекаємо таймаут
  uint16_t ms = millis() - startTime;
  WDTCSR |= (1 << WDCE) | (1 << WDE);   // дозволяємо зовнішнє управління
  WDTCSR = 0;
  return ms * 4;
}
```

ДОДАТОК Г (обов'язковий)

ГРАФІЧНА ЧАСТИНА

«ПРОГРАМНО-АПАРАТНИЙ МОДУЛЬ АВТОМАТИЧНОЇ ГОДІВНИЧКИ ДЛЯ ТВАРИН»

Виконав: студент 4 курсу, групи 1КН-21б
спеціальності 122 – Комп'ютерні науки
(шифр і назва напряму підготовки, спеціальності)

Подолян І.Р.
(прізвище та ініціали)

Керівник: к.т.н., доцент кафедри КН
Озеранський В.С.
(прізвище та ініціали)

03.06.2025р.

Вінниця ВНТУ - 2025 рік

Рисунок Г.1 – програмно-апаратні аналоги автоматичної годівнички

Рисунок Г.2 − Компоненти для реалізації електричної частини програмно-апаратного модуля автоматичної годівнички для тварин

Рисунок Г.3 − Компоненти для реалізації механічної частини програмно-апаратного модуля автоматичної годівнички для тварин

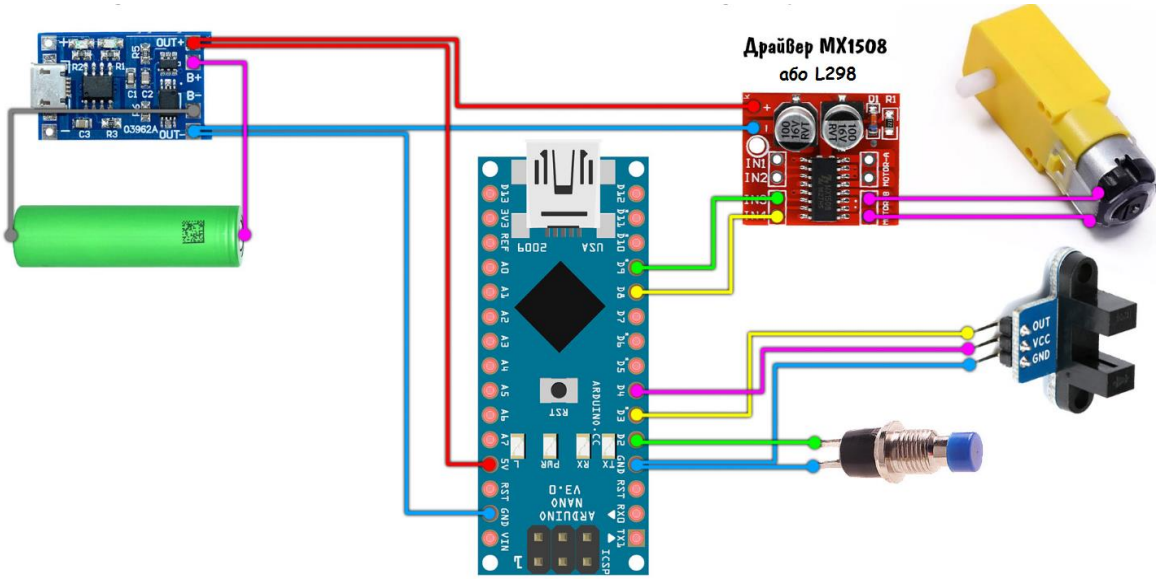

Рисунок Г.4 – Монтажна схема програмно-апаратного модуля автоматичної годівнички для тварин

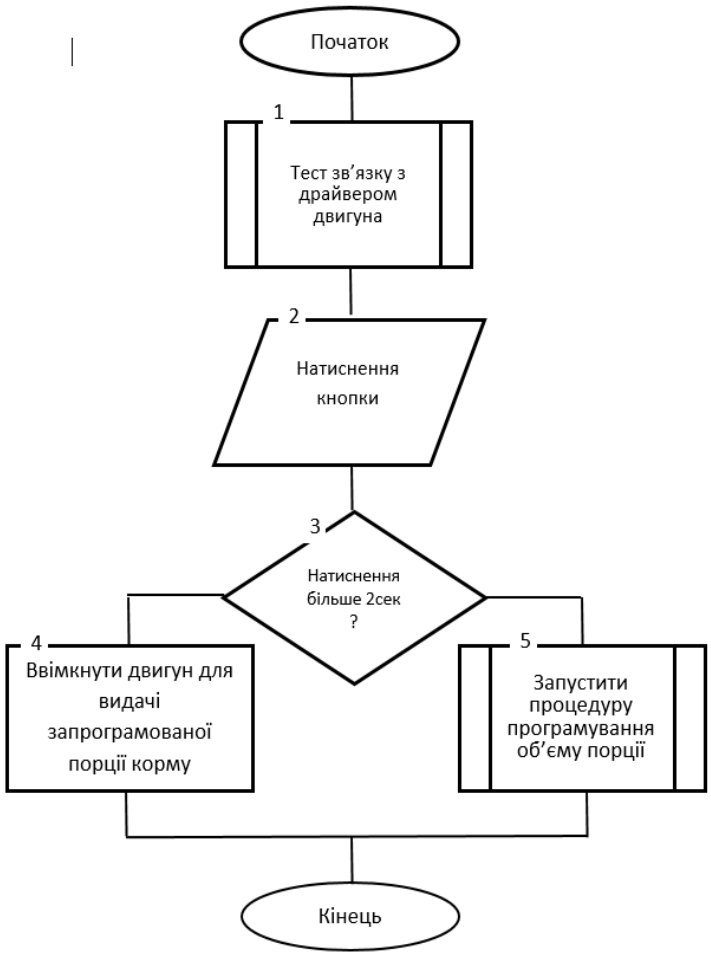

Рисунок Г.5 – Алгоритм роботи програмно-апаратного модуля автоматичної годівнички для тварин

Рисунок Г.6 – Процес створення програмно-апаратного модуля автоматичної годівнички для тварин

Рисунок Г.7 – Зовнішній вигляд програмно-апаратного модуля автоматичної годівнички для тварин

ДОДАТОК Д

Довідка про впровадження

МАЛЕ НАУКОВО-ВИРОБНИЧЕ ПІДПРИЄМСТВО "ТОВ "ІТІ"
Україна, 21021, м.Вінниця, вул. Келецька, 56

№ 53 від "6" 06 2025 р.

ДОВІДКА

Дана Подоляну Іоанну Романовичу в тому, що результати, одержані ним в процесі виконання бакалаврської кваліфікаційної роботи, а саме алгоритм та програмне забезпечення автоматичної годівнички для тварин, планується використати в розробках ТОВ «ІТІ».

Зам. директора ТОВ «ІТІ» Бодяк В.М.