МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
УМАНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ САДІВНИЦТВА
Факультет економіки і підприємництва
Кафедра інформаційних технологій

Допущено до захисту
"_____" _____ 2023 р.
Зав. кафедри, к.т.н., доцент
_____ **Ліщук Р.І.**

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

РОЗРОБКА МІКРОПРОЦЕСОРНОГО ЗАСОБУ ДЛЯ РЕАЛІЗАЦІЇ АДАПТИВНОГО АЛГОРИТМУ ДОЇННЯ

Виконав – студент 4 курсу, групи 41-кн
освітньої програми "Комп'ютерні науки"
спеціальності 122 Комп'ютерні науки
Новіцький Павло Олександрович

Науковий керівник – д.т.н., професор
Кулаков Павло Ігорович

Умань – 2023

ЗМІСТ

ВСТУП 4

РОЗДІЛ 1. АНАЛІТИЧНИЙ ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ 6

1.1 Об’єкт та предмет дослідження 6

1.2 Сучасні тенденції та інновації в галузі автоматизації доїння та їх вплив на розвиток адаптивних алгоритмів. 7

1.3 Рівень автоматизації молочних ферм України 8

1.4 Процес доїння тварини з використанням автоматизованого доїльного апарату та особливості параметрів 9

1.5 Переваги та недоліки автоматизованого доїльного процесу з використанням адаптивного алгоритму доїння 12

1.6 Особливості отримання інформації з датчиків 14

1.7 Постановка задачі 16

РОЗДІЛ 2. ВИБІР ТЕХНОЛОГІЙ ТА АРХІТЕКТУРА ПІДСИСТЕМИ 18

2.1 Технологія розробки Java для мікроконтролерів сімейства AVR з використанням Arduino 18

2.2 Технологія взаємодії мікроконтролера та ПК 22

2.3 Розробка архітектури інформаційної підсистеми 25

РОЗДІЛ 3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 32

3.1 Розробка програмного забезпечення для мікроконтролера 32

3.1.1 Огляд коду 33

3.1.2 Опис методу loop() 34

3.1.3 Опис методу controlPulsators() 37

3.2 Зв'язок між ПЗ мікроконтролера та персонального комп'ютера 41

3.3 Розробка програмного забезпечення для персонального комп’ютера 42

3.3.1 Опис методу main() 44

3.3.2 Огляд методу serialEvent() 46

ВИСНОВОК 48

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 49

Додаток А ..... **Помилка! Закладку не визначено.**

Додаток Б ..... **Помилка! Закладку не визначено.**

ВСТУП

Розвиток сучасного сільського господарства потребує впровадження новітніх технологій і засобів автоматизації для підвищення продуктивності та ефективності виробництва. Ключовим аспектом є доїння корів, де правильне керування процесом доїння може значно вплинути на якість молока, здоров’я тварин і загальну продуктивність ферми.

Метою даної кваліфікаційної роботи є розробка мікропроцесорного засобу для реалізації адаптивного алгоритму доїння, що забезпечує автоматизацію процесу доїння корів. Інструмент має забезпечувати аналіз даних з датчиків, слідкувати за параметрами доїння та надавати зворотній зв’язок для регулювання інтенсивності доїння на основі різних факторів.

Основними завданнями дослідження є:

- Систематизувати та розширити теоретичні та практичні знання в галузі доїння корів та автоматизації сільського господарства.
- Розробка адаптивного алгоритму доїння на основі аналізу даних датчиків та використання мікропроцесорів для керування процесом доїння.
- Визначити оптимальні параметри доїння для забезпечення оптимальної продуктивності та комфорту тварин.

Під час виконання завдання будуть використані такі методи:

- Аналіз наукових даних та літератури: огляд існуючих методів і технологій доїння, а також досліджень, пов'язаних з автоматизацією та адаптивним управлінням процесом доїння.
- Збір даних про параметри доїння, використовуючи сучасні датчики для вимірювання важливих параметрів, таких як об’єм молока, інтенсивність імпульсації, час доїння тощо.

- Розробка програмної частини: розробка програмного коду, включаючи збір даних, алгоритми обробки та аналізу даних, налаштування інших компонентів на основі зібраних даних.
- Математичне моделювання: побудова математичних моделей процесу доїння та адаптивного управління, що дозволяють прогнозувати та оптимізувати параметри доїння за різними факторами.

Інформаційна база дослідження. В ході дослідження було оглянуто:

- «Автоматизація технологічних процесів і системи автоматичного керування» - Барало О. В., Самойленко П. Г., Гранат С. Є., Ковальов В. О.
- «Елементи теорії вимірювального контролю параметрів біотехнічної системи доїння» - Кулаков П.І.

РОЗДІЛ 1. АНАЛІТИЧНИЙ ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Об’єкт та предмет дослідження

Загальний опис галузі «Розробка мікропроцесорного засобу для реалізації адаптивного алгоритму доїння» відноситься до дослідження та розробки спеціалізованого пристрою, що використовує мікропроцесорну технологію для автоматизованого процесу доїння.

Ця галузь поєднує знання в області мікропроцесорних систем, сільськогосподарського виробництва та автоматизації процесів. Основною метою є розробка ефективного та адаптивного алгоритму доїння, який може контролювати та керувати процесом доїння для забезпечення оптимальних умов доїння тварин.

Дослідження зосереджено на вивченні поведінки та фізіології молочних тварин, розробці алгоритмів обробки даних, проектуванні апаратного забезпечення та інтеграції системи з інструментами автоматизації сільського господарства. Результатом досліджень є мікропроцесорний пристрій, що гарантує високу ефективність і точність доїння, а також надаватиме оптимальне використання ресурсів.

Розвиток адаптивного доїння є актуальним процесом і має значний практичний вплив на аграрний сектор. Автоматизація процесу доїння може підвищити ефективність роботи ферми, забезпечити високу якість молока та підвищити продуктивність тварин. Крім того, розробка адаптивного алгоритму доїння сприяє зменшенню витрат праці та забезпечує комфортні умови для тварин.

Актуальність теми обумовлена кількома факторами:

- По-перше, доїння тварин є важливим кроком у молочній промисловості, що впливає на якість і кількість молока. Удосконалення цього процесу призведе до збільшення виробництва

молока, раціонального використання ресурсів та високої якості продукції.

- По-друге, необхідно забезпечити стале виробництво з обмеженими ресурсами, враховуючи зростання населення та попит на молочні продукти. Модифікований блок керування доїнням, вбудований у мікропроцесор, може контролювати процес доїння відповідно до потреб тварин, їх фізіологічного стану та факторів, що зменшують навантаження на обладнання та забезпечують належні умови для кожної тварини.
- По-третє, поточна ситуація робить автоматизацію сільського господарства та впровадження нових пропозицій ще більш важливим. Розвиток мікропроцесорних пристроїв дозволяє фермерам збільшити виробництво і знизити витрати.

1.2 Сучасні тенденції та інновації в галузі автоматизації доїння та їх вплив на розвиток адаптивних алгоритмів.

Сфера автоматизації доїння має велике значення для продуктивності та здоров’я тварин на фермах. Останнім часом з’явилося кілька актуальних тенденцій та інновацій, що впливають на цю галузь і розробку адаптивних алгоритмів. Ось деякі з них:

- Розвиток розпізнавання образів. Сучасні доїльні системи використовують комп’ютерний зір і машинне навчання для розпізнавання таких закономірностей, як положення сосків або мастит у вимені корови. Це допомагає швидко й ефективно виявляти аномалії та реагувати на них.
- Системи моніторингу та аналізу: встановлення датчиків і систем моніторингу в апаратурі збирає велику кількість даних про сільськогосподарських тварин. Потім ці дані можна проаналізувати за допомогою алгоритмів машинного навчання, щоб визначити закономірності та покращити процес доїння.

- Автоматична ідентифікація тварин. Використання таких технологій, як радіочастотна ідентифікація (RFID) або системи комп'ютерного зору, допомагає автоматично ідентифікувати та відстежувати окремих тварин [1]. Це дозволяє вести облік сільськогосподарських тварин, стежити за їх здоров'ям і продуктивністю, а також забезпечити індивідуальний підхід до кожної тварини під час доїння.
- Розробка робототехнічних систем. Роботизовані доїльні системи, такі як автоматизовані доїльні апарати, стають все більш поширеними. Вони автоматизують процес доїння, забезпечуючи стандартизацію та зменшуючи навантаження на фермерів. Такі системи також можна інтегрувати з іншими автоматизованими процесами, щоб забезпечити оптимальний контроль над управлінням запасами.

Ці тенденції в автоматизації доїння безпосередньо впливають на розробку адаптивних алгоритмів. Збір великих обсягів даних і використання комп'ютерного зору та машинного навчання вимагають розробки алгоритмів обробки даних, розпізнавання образів і прийняття рішень. Адаптивні алгоритми дозволяють системам автоматизації доїння швидко адаптуватися до мінливих умов і вивчати зібрані дані для підвищення точності та ефективності процесу.

Загалом сучасні тенденції та інновації у сфері автоматизації доїння сприяють підвищенню продуктивності, здоров'я та добробуту великої рогатої худоби. Розробка адаптивних алгоритмів робить системи автоматизації більш гнучкими, ефективними та адаптованими до унікальних потреб кожної тварини.

1.3 Рівень автоматизації молочних ферм України

Рівень автоматизації молочних ферм в Україні сьогодні різний і залежить від розміру та економічного потенціалу ферми. Зазвичай великі ферми використовують високий рівень автоматизації, щоб зменшити витрати на робочу силу та збільшити ефективність виробництва. Розглядаючи малі та

середні молочні господарства, важливо розуміти, що рівень автоматизації дуже низький через обмежені можливості з точки зору фінансів, розташування, кількості тварин та інших факторів.

Поширеною автоматизованою системою на молочній фермі є автоматизована система доїння, яка забезпечує ефективне, економне та швидке доїння тварин. Ці системи включають системи автоматизації доїльних апаратів, системи контролю якості молока, системи доставки молока, системи годування, обладнання для стерилізації та очищення апаратури.

Слід зазначити, що автоматизація молочних ферм в Україні є поступовим процесом, оскільки дороге обладнання вимагає значні витрати. Однак усвідомлення переваг автоматизації та посилення конкуренції в молочній промисловості стимулюють інтерес фермерів до впровадження автоматизованих систем.

1.4 Процес доїння тварини з використанням автоматизованого доїльного апарату та особливості параметрів

Доїльний апарат — це пристрій, що призначений для автоматизованого процесу доїння тварини[2]. Загалом доїльний апарат складається з основних компонентів, таких як колектору(системи вакууму), пульсатору, доїльних стаканів, двигуна, молокопроводу та системи керування.

Рис. 1.1 – Доїльна установка DeLaval [3]

Основним принципом роботи автоматизованого апарату є створення вакууму, завдяки якому відсмоктується молоко з вимені тварини. Вакуумна система створює вакуум і підтримує його на необхідному рівні, використовуючи для цього вакуумний насос, що створює негативний тиск. Далі починає видоюватись молоко, яке потрапляє до спеціального резервуару завдяки молокопроводу [1-4].

Молокопровід — це система труб і клапанів, що дозволяє транспортувати молоко від корови до збірного резервуару. На шляху до резервуару воно проходить через систему фільтрації та охолодження, тим самим забезпечує необхідну обробку молока.

Система управління відповідає за автоматизацію процесу доїння і контроль його параметрів. Вона має електронні сенсори, контролери і програмне забезпечення, яке може контролювати режим доїння худоби, налаштовувати вакуум, вести облік та слідкувати за якістю молока і забезпечувати безпеку та захист апарату.

Для використання доїльного апарату[2;5], тварину необхідно спочатку оглянути і підготувати до доїння. Якщо вим’я в належному стані, до нього під’єднується відповідна апаратура і запускається доїльна машина. Доїльний апарат стимулює вироблення молока, створюючи пульсуючий масаж вимені за допомогою вакууму. Зазвичай доїння проводиться звичайним для тварини способом, що робить його максимально комфортним для ефективного процесу.

Після запуску агрегату молоко надходить у спеціальні доїльні стакани, які з’єднані молокопроводом. Вони мають власний клапан, який відкривається при створенні вакууму, дозволяючи молоку надходити в молокопровід. Після того, як тварину видоїли, доїльний апарат автоматично від’єднується від вимені та очікує підключення до наступної тварини.

Молоко, що потрапляє в молокопровід, на шляху до місця зберігання проходить через спеціальні фільтри, де видаляється бруд і домішки.

Система управління встановлюється для контролю та регулювання швидкістю доїння, вакууму та контролю якості молока. Система має інтерфейс, який дозволяє оператору регулювати налаштування доїльного апарату, контролювати продуктивність і процеси очищення та обслуговування системи.

Забезпечення ефективного доїння є дуже важливим для тварини та включає в себе такі функції, як контроль параметрів вакууму, регулювання інтенсивності масажу вимені, відключення апарату в разі певних умов тощо. Ці властивості забезпечують високу якість доїння та уникають багатьох проблем, пов’язаних з травмуванням та навантаженням на тварину.

У процесі роботи доїльний апарат фіксує кількість і якість зібраного молока. Відбувається це завдяки спеціальним датчикам, які передаються об'єм молока, концентрацію в ньому білків і жирів, а також виявляють наявність домішок чи інших аномалій. Інформація про молоко передається в комп’ютерну систему для подальшого аналізу, обробки або виконання певних дій.

Крім того, доїльні апарати мають функцію очищення та дезінфекції. Після доїння тварини машина автоматично очищається та дезінфікується перед підключенням до наступної тварини. Це забезпечує гігієнічні стандарти та запобігає можливому зараженню молока чи передачі інфекцій.

Загалом доїльні апарати – це складні технологічні мікропроцесорні системи, які поєднують у собі механічні, електронні та програмні компоненти. Ці системи створені для забезпечення максимальної ефективності, економічності, комфорту тварин, якості зібраного молока та обліку. Вони допомагають підвищити продуктивність і автоматизувати ферми.

Оптимізація взаємодії доїльного апарату, людини і тварини, створення доїльного апарату та інших технологічних пристроїв, що забезпечують

ефективну стимуляцію рефлексу доїння та підвищення продуктивності тварин, є складним науково-технічним завданням, яке ще до кінця не вирішене.

Отже, доїльні апарати є найважливішою частиною автоматизованої системи доїння в сучасному світі. Вони гарантують ефективне та економічне доїння тварин, збір молока з мінімальними затратами праці, часу та високою якістю продукції.

1.5 Переваги та недоліки автоматизованого доїльного процесу з використанням адаптивного алгоритму доїння

Процес автоматизованого доїння за адаптивним алгоритмом, як і будь-який інший процес, має свої переваги та недоліки.

Переваги автоматизації:

- Ефективність: автоматизоване доїння може значно підвищити продуктивність процесу доїння. Процес, який зазвичай виконується вручну, можна виконати швидше та ефективніше за допомогою автоматизованого обладнання.
- Висока якість доїння: адаптивний алгоритм доїння дозволяє адаптувати процес доїння до індивідуальних особливостей кожної корови. Це дозволяє доїльному обладнанню точно контролювати тиск і рух під час доїння, що максимізує збір молока та запобігає дискомфорту, який тварини можуть відчувати під час ручного доїння.
- Комфорт тварин: автоматичне доїння може бути менш стресовим для тварин, оскільки корови можуть почувати себе комфортно та відносно безпечно під час доїння, так як всі доїльні машини обладнані спеціальними силіконовими насадками. Крім того, тварини можуть пристосуватися до регулярного графіка доїння, що призводить до більш стабільного розподілу молока.

- Економія часу та праці: автоматизоване доїння вимагає менше ручної праці. Це дозволяє фермерам ефективніше використовувати свій час і ресурси, зосереджуючись на інших аспектах управління фермою.
- Економія: незважаючи на те, що доїльні апарати капіталомісткі, вони дозволяють істотно знизити трудовитрати. Для роботи з доїльним апаратом потрібен тільки оператор. Молоко збирається автоматично, автоматично потрапляє до резервуару, і машина також автоматично промивається, тому можна заощадити ресурси в майбутньому.
- Серверна система: автоматизований процес доїння автоматично вводить до бази даних дані про доїння, стан тварин, стан машини та інші важливі дані, необхідні для роботи молочного фермерського підприємства. Не потрібно нічого записувати вручну, все автоматизовано. Цей процес полегшить управління фермою, одна людина може керувати доїнням, резервуарами, відвантаженням молока через комп'ютер, ноутбук або, навіть, телефон.

Недоліки автоматизації:

- Високі витрати: впровадження автоматизованого процесу доїння з використанням адаптивного алгоритму доїння може бути дорогим. Вартість обладнання та його встановлення можуть стати значними фінансовими витратами для фермера. Ціна обладнання залежить від потреб і обсягу виробництва, тому це може бути проблемою для невеликих молочних ферм.
- Вимоги до технічного обслуговування: автоматизоване доїльне обладнання вимагає регулярного технічного обслуговування та огляду. Необхідно забезпечити коректну роботу системи, своєчасне виявлення та усунення можливих збоїв, що може бути дорогим і вимагати додаткового персоналу.
- Вплив на зайнятість: автоматизація процесу доїння може вплинути на кількість робочих місць, оскільки деякі ручні роботи можуть бути

замінені автоматизованими пристроями. Це може мати негативний вплив через зменшення працівників.

Таким чином, автоматизований процес доїння з використанням адаптивного алгоритму доїння має багато переваг, таких як підвищення продуктивності, висока якість доїння, комфорт тварин, економія часу та праці, економічність та високий рівень управління фермою. Однак, це також вимагає значних витрат і обслуговування. Враховуючи ці фактори, фермери повинні ретельно обміркувати переваги та недоліки автоматизації доїння, перш ніж прийняти рішення про впровадження на свою ферму.

1.6 Особливості отримання інформації з датчиків

Отримання інформації з датчиків за допомогою мікропроцесорів є важливою складовою багатьох сучасних систем. Датчики можуть бути різних типів, наприклад, датчики температури, датчики освітленості, акселерометри, гіроскопи, датчики тиску, датчики вологості, датчики відстані та ін. Особливості отримання інформації залежать від типу датчика і способу його взаємодії з ним.

Як правило, датчики підключаються до мікропроцесора за допомогою спеціального інтерфейсу даних, такого як I2C (інтегрована схема), SPI (послідовний периферійний інтерфейс) або UART (універсальний асинхронний приймач/передавач) [6]. Ці інтерфейси дозволяють передавати дані між мікропроцесором і датчиками. Оскільки кожен тип датчика може мати власні характеристики, важливо вивчити документацію для конкретного датчика та використовувати відповідний інтерфейс для зв’язку.

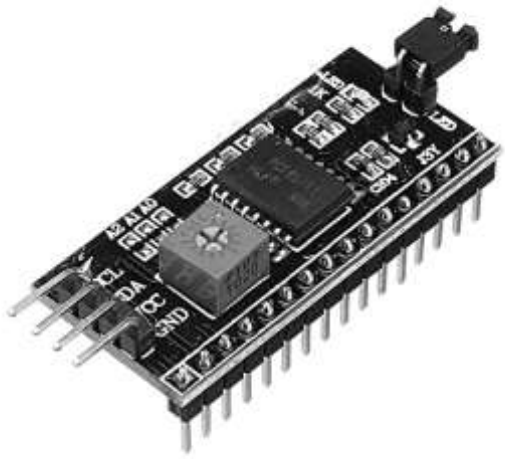

Рис. 1.2 – Інтегрована схема I2C

При отриманні даних від датчиків за допомогою мікропроцесорів зазвичай використовуються спеціальні протоколи зв'язку. Наприклад, для I2C і SPI існують спеціальні команди та регістри, за допомогою яких можна налаштувати датчик і отримати дані. Мікропроцесор подає команди на датчик і приймає дані відповідно до своїх можливостей. Кожен тип датчика може мати різні команди та регістри, тому важливо прочитати документацію датчика, щоб правильно з ним взаємодіяти.

На додаток до протоколів зв’язку існують бібліотеки та фреймворки, які спрощують отримання даних датчиків за допомогою мікропроцесорів. Наприклад, для платформи Arduino існує безліч бібліотек для різних типів датчиків, що полегшує взаємодію з ними. Такі бібліотеки часто надають функції для ініціалізації датчика, зчитування даних і встановлення параметрів.

Дані, отримані від датчиків, можуть оброблятися безпосередньо на мікропроцесорі або передаватись на зовнішні пристрої для подальшої обробки. Залежно від програми можна виконувати різні дії з отриманими даними, наприклад зберігати їх у пам’яті, надсилати через мережу або використовувати для керування іншими системами.

Загалом, отримання інформації з датчиків залежить від конкретних типів датчиків, мікропроцесора та технології, що використовується під час розробки програмного забезпечення. Важливо ознайомитися з документацією для кожного компонента та використовувати відповідні методи зв’язку для успішного обміну даними та отримання необхідних даних.

1.7 Постановка задачі

Метою дослідження є розробка мікропроцесорного засобу для реалізації адаптивного алгоритму доїння.

Поставлена задача створити мікропроцесорний пристрій, який забезпечить реалізацію адаптивного алгоритму доїння в умовах виробництва. Адаптивний алгоритм доїння є ключовим елементом у вдосконаленні технології доїння тварин, особливо на молочних фермах.

Основне призначення алгоритму адаптивного доїння – автоматизація процесу доїння та оптимізація його ефективності. Доїльні засоби повинні бути адаптовані до індивідуальних особливостей кожної тварини, її фізіологічного стану та реакції на процес доїння. Це забезпечує оптимальну якість молока, запобігає травмуванню тварини та забезпечує максимальні надої.

Реалізація адаптивного алгоритму доїння потребує спеціалізованого мікропроцесора, здатного виконувати обчислення та керувати доїльним обладнанням. Цей інструмент повинен мати достатню обчислювальну потужність, надійність і зв'язок з іншими системами. Він має забезпечувати надійне та точне виконання алгоритму доїння в режимі реального часу, його легко налаштовувати та розширювати для використання в різних середовищах.

Основними елементами алгоритму адаптивного доїння є зчитування та аналіз даних про фізіологічний стан тварини, керування доїльним обладнанням, моніторинг параметрів доїння та збір даних для подальшого аналізу.

Мікропроцесор повинен забезпечувати взаємодію всіх вузлів, координувати їх роботу і забезпечувати високу якість доїння.

В рамках цього дослідження буде розроблено мікропроцесорний скрипт, який міститиме алгоритми реалізації ефективної системи доїння. При розробці інструменту буде врахована специфіка виробничого середовища та вимоги до надійності та швидкості передачі інформації та її обробки.

РОЗДІЛ 2. ВИБІР ТЕХНОЛОГІЙ ТА АРХІТЕКТУРА ПІДСИСТЕМИ

2.1 Технологія розробки Java для мікроконтролерів сімейства AVR з використанням Arduino

Arduino [7] — це інтерфейс між мовою програмування високого рівня(Java) та мікроконтролерами сімейства AVR, які зазвичай розробляються з використанням мови нижчого рівня, такої як C, C++ або ж асемблер. Дозволяючи розробникам використовувати Java для програмування мікроконтролерів, ця технологія спрощує процес розробки та надає для роботи інструменти вищого рівня.

Одним із ключових елементів технології розробки програмного засобу для мікроконтролерів та персональних комп'ютерів є система Arduino, яка є відкритою електронною платформою проектування на основі мікроконтролерів Atmel AVR. Arduino надає зручне та просте у використанні середовище розробки (IDE), яке дозволяє писати програми Arduino на основі C/C++ і завантажувати їх на плату Arduino. Однак для розширення функціональності мікроконтролеру, використовується мова Java, якою створюються програмні застосунки для комп'ютера, який взаємодіє з мікропроцесором. Java - це можливість створювати абсолютно нові застосунки, які зможуть відразу працювати з базою даних, персональним комп'ютером тощо. Для розробки програм мовою Java для мікроконтролерів AVR за допомогою Arduino існує кілька бібліотек і фреймворків, які дозволяють взаємодіяти між мовою та мікроконтролерами. Однією з таких бібліотек є RXTX, яка забезпечує послідовний зв'язок між комп'ютером і мікроконтролером через порт RS-232 або USB. RXTX забезпечує інтерфейс мови Java для читання та запису даних у мікроконтролер.

Ще один фреймворк для розробки Java-додатків для Arduino — Firmata. Firmata дозволяє використовувати Arduino як периферійний контролер, яким можна керувати за допомогою програми на мові Java. Це дозволяє розробникам

використовувати мову програмування високого рівня, яка є об’єктно-орієнтованою, і зручно взаємодіяти з мікроконтролером.

Зокрема, популярний серед художників і дизайнерів фреймворк Processing. Цей фреймворк можна використовувати для створення Java-програм для комп’ютерів, що будуть взаємодіяти з Arduino. Processing надає безліч готових функцій і бібліотек для візуалізації даних, що дозволяє створювати складні програми графічного інтерфейсу для мікроконтролерів AVR.

За допомогою цих інструментів і бібліотек програміст може писати додатки мовою Java, які працюють з мікроконтролерами AVR. Використовуючи Arduino як посередника, можна отримати доступ до цифрового та аналогового вводу-виводу, зчитувати дані з датчиків, керувати мікропроцесорами та іншими периферійними пристроями, а також виконувати різноманітні програмні логічні дії.

Хоча розробка мовою Java для мікроконтролерів AVR за допомогою Arduino відкриває нові можливості для розробників, використання мови вищого рівня може вплинути на продуктивність програми та продуктивність порівняно з рідними мовами програмування, такими як C або асемблер. Крім того, в залежності від проекту можуть бути обмеження в обсязі пам'яті або швидкості роботи мікроконтролера AVR, що також слід враховувати при виборі технології розробки.

Мова програмування Arduino створена максимально схожою на C/C++. Однак Arduino IDE має деякі особливості та функції, які полегшують роботу з мікроконтролерами. Основними елементами програми Arduino є:

- Функція setup(): Ця функція викликається один раз під час запуску мікроконтролера. Використовується для встановлення контактів, ініціалізації змінних та встановлення інших параметрів.
- Функція loop(): ця функція виконується безперервно після функції setup(). Вона містить основний програмний код, який виконується циклічно. В

функції зазвичай розписана основна логіка програми, наприклад зчитування даних з датчиків, обробка цих даних, управління мікропроцесором тощо.

- Піни (порти): Arduino має набір цифрових і аналогових контактів введення/виведення, які можна використовувати для зчитування вхідного сигналу або встановлення значення вихідного сигналу. Наприклад, можна використовувати цифрові контакти для керування світлодіодами, реле або кнопками зчитування. Аналогові виходи можна використовувати для читання аналогових даних із датчиків.
- Бібліотеки: Arduino поставляється з набором стандартних бібліотек, які полегшують взаємодію з різними пристроями та модулями, такими як LCD-дисплеї, сенсори, серводвигуни тощо. Ці бібліотеки містять готові функції, які полегшують програмування та роботу з цими пристроями.
- Послідовний зв'язок: Arduino може взаємодіяти з комп'ютером або іншими пристроями через порти зв'язку, такі як USB або RS-232. Використовуючи функцію послідовного порту, можна надсилати й отримувати дані через це з’єднання, дозволяючи взаємодіяти із зовнішніми програмами або налагоджувати програму.
- Функції таймера та переривання: Arduino підтримує таймери та переривання для виконання певних дій у фоновому режимі або відповідно до певних подій. Це дозволяє створювати програми, які реагують на зміну умов, введення інформації або працювати на основі таймерів.

Після написання програми на Arduino її можна завантажити на мікроконтролер через USB-порт або інший засіб зв’язку, який дозволяє програмувати контролер. Після завантаження програма почне працювати на мікроконтролері.

Arduino спрощує програмування мікроконтролерів, зменшує потребу в глибокому розумінні апаратного забезпечення, забезпечуючи при цьому

гнучкість і можливість реалізації широкого спектру проєктів. Він став популярним у багатьох сферах, включаючи робототехніку, IoT тощо.

Для програмування самого мікроконтролера з Arduino використовується спеціальна мова Arduino, базована на C/C++, тому вона й має так багато збіжностей. А власне написання коду відбувається в спеціальному додатку, що пропонує Arduino – Arduino IDE. Arduino IDE(Integrated Development Environment) – це середовище розробки, спеціально призначене для програмування мікроконтролерів компанії Arduino. Середовище надає простий і зручний спосіб написання коду, компіляції та завантаження програм на мікроконтролер. Основні компоненти середовища розробки Arduino:

- Одним із найважливіших компонентів Arduino IDE є вбудований текстовий редактор коду, в якому можна створювати свої програми для мікроконтролерів Arduino. Редактор підтримує функції виділення синтаксису, автодоповнення та перевірку синтаксису. Також, є функція створення власних плагінів для редактору. В ньому можна створювати нові файли, відкривати і редагувати існуючі скетчі і зберігати їх на комп'ютері.
- Наступним компонентом є компілятор. Arduino включає компілятор, який перетворює код на мові Arduino в машинний код, зрозумілий мікроконтролеру. Також в ньому є вбудований завантажувач, який дозволяє передавати скомпільований код на мікроконтролер для його виконання. Для цього необхідно підключити мікроконтролер до ПК, використовуючи USB-порт, та натиснути на кнопку компіляції та завантаження. Таким чином, необхідність в використанні стороннього софту для програматора відпадає.
- Arduino IDE має велику кількість вбудованих бібліотек, які містять готові функції та класи для роботи з різними компонентами та модулями, такими як дисплеї, датчики, двигуни тощо. Бібліотеки можна легко

підключити до проєкту та використовувати готові функції для спрощення програмування.

- Монітор портів: Arduino IDE має вбудований монітор портів, який дозволяє переглядати та налагоджувати дані, що надсилаються між мікроконтролером і комп'ютером через послідовний порт. Використовувати монітори портів можна для моніторингу вихідних даних програми, налагодження помилок і взаємодії з програмою на мікроконтролері за допомогою текстового введення-виведення.
- Встановлення бібліотек і плат: Arduino IDE спрощує встановлення нових бібліотек і плат. Шукати бібліотеки в онлайн-сховищі інструменту та інсталювати їх в інтегроване середовище можна лише одним кліком миші.
- Вбудована підтримка документації: Arduino IDE поставляється з вбудованою документацією, в якій можна знайти описи функцій, класів, методів та іншу інформацію про Arduino API. Зручний для вивчення та використання доступних функцій.

Arduino IDE — це безкоштовне та просте у використанні середовище розробки, яке дозволяє навчитися програмувати мікроконтролери Arduino та створювати різноманітні проєкти. Він підтримує широкий спектр платформ Arduino, має активну спільноту користувачів і велику кількість навчальних матеріалів.

2.2 Технологія взаємодії мікроконтролера та ПК

Взаємодія мікроконтролера за допомогою Arduino з персональним комп'ютером (ПК) за допомогою мови програмування Java включає кілька основних етапів[7-9]:

1) Підключення та налаштування підключення: мікроконтролер підключається до комп'ютера через послідовний порт (наприклад, USB). Комп'ютер має

бути налаштований за допомогою програмного забезпечення для взаємодії з цим портом. Для Java можна використовувати такі бібліотеки, як RXTX або jSerialComm, щоб забезпечити доступ до COM-порту.

2) Передача даних: за допомогою програми на мікроконтролері та програми на ПК дані можуть передаватися в обох напрямках. На мікроконтролері програма використовує послідовну бібліотеку Arduino для читання та запису даних через послідовний порт. На ПК з Java можна використовувати методи, надані відповідними бібліотеками, для читання та запису даних через COM-порт.
3) Протокол передачі даних: для забезпечення структурованого зв’язку можна використовувати спеціальний протокол передачі даних. Наприклад, можна використовувати текстовий протокол, такий як JSON або XML, для передачі й аналізу даних між мікроконтролером і комп’ютером. Це дозволяє точно визначити формат даних і спрощує обробку даних з обох сторін.
4) Обробка запитів: У програмі для ПК можна визначити логіку обробки запитів, що надходять від мікроконтролера. Наприклад, можна надсилати команди зі свого комп’ютера на мікроконтролер, і мікроконтролер виконуватиме відповідні дії на основі цих команд. Після обробки запиту на мікроконтролері відповідь може бути відправлена на комп'ютер для подальшої обробки.

Загалом взаємодія між мікроконтролером і ПК полягає в надсиланні даних через послідовний порт, використанні відповідних протоколів для структурування даних і обробці запитів з обох сторін. Це дозволяє створювати програми керування мікроконтролером на ПК та обмінюватися з ним даними.

На основі взаємодії мікроконтролера та персонального комп’ютера можна розробити спеціальне програмне забезпечення, яке буде працювати на комп’ютері, за допомогою якого можна в реальному часі стежити за процесами, що відбуваються в процесі доїння, керувати цим процесом, відслідковувати графіки імпульсації тощо. А з використанням Arduino, розробка додатку для

комп’ютера є ще простішою, адже можна використовувати мову програмування Java.

Java — це потужна, універсальна мова програмування, яка зазвичай використовується для розробки програмного забезпечення для настільних комп’ютерів, веб-додатків, мобільних додатків і багатьох інших галузей.

Основні особливості мови Java[8]:

- Об’єктно-орієнтованість: Java є об’єктно-орієнтованою мовою, це означає, що все в ній є об’єктом. Вона підтримує базові концепції ООП, такі як успадкування, поліморфізм, інкапсуляція та абстракція.
- Незалежність від платформи: код Java компілюється в байт-код, який можна виконати на будь-якій платформі, де встановлена віртуальна машина Java. Це робить Java портативною мовою, оскільки розробники можуть написати код один раз і працювати з ним в різних операційних системах, таких як Windows, macOS і Linux.
- Багатопотоковість: Java має вбудовану підтримку багатопоточності, яка дозволяє створювати програми, які можуть виконувати багато завдань одночасно. Це особливо корисно під час розробки програм, які працюють у паралельних середовищах, таких як мобільні додатки, сервери тощо.
- Велика бібліотека стандартів: Java включає в себе велику бібліотеку стандартів, яка надає готові рішення для багатьох завдань, таких як робота з мережею, введення/виведення даних, обробка інформації, маніпулювання базами даних, створювати графічний інтерфейс користувача тощо.
- Підтримка бази даних: Java пропонує широку підтримку роботи з базами даних. Вона надає API для підключення до реляційних баз даних, таких як MySQL, Oracle, PostgreSQL, і для роботи з базами даних NoSQL, такими як MongoDB. Завдяки цьому зручно створювати програми, що працюють зі структурованими даними.

Java також має багато інших функцій, таких як автоматичне керування пам’яттю, безпека, підтримка мережевих протоколів тощо. Вона має велику спільноту розробників і багато інструментів, таких як інтегровані середовища розробки (наприклад, Eclipse, IntelliJ IDEA) і фреймворки (наприклад, Spring), які полегшують розробку програмного забезпечення на Java.

Одним із найкращих інтегрованих середовищ Java є IntelliJ IDEA. IntelliJ IDEA — це потужне середовище розробки (IDE) для Java та інших мов, яке забезпечує аналіз коду, автодоповнення, перевірку синтаксису, рефакторинг і багатомовну підтримку. Воно забезпечує зручний інтерфейс, підвищує продуктивність розробника, підтримує інтеграцію з іншими інструментами розробки, допомагає зменшити кількість помилок і швидше створювати якісний код.

2.3 Розробка архітектури інформаційної підсистеми

Розробка архітектури інформаційної підсистеми для реалізації адаптивного алгоритму доїння включає в себе розробку алгоритму для мікропроцесорного засобу, який забезпечуватиме ефективне управління процесом доїння. Даний алгоритм повинен автоматизувати процес доїння тварин з використанням принципів, які допомогли б адаптувати доїння для потреб кожної окремої тварини. Розробка архітектури передбачає визначення оптимальної структури засобу, зокрема, вибір необхідних компонентів, інтерфейсів та взаємозв’язків елементів. При розробці враховуються особливості фізіології тварин та необхідні параметри, що дозволяють досягти оптимальних результатів доїння. Основною метою розробки є створення ефективного та надійного засобу, що забезпечуватиме оптимальні умови доїння та підвищить продуктивність молочного скоту.

В рамках розробки архітектури інформаційної підсистеми мікропроцесорний засіб виконує ключові функції, а саме: керує процесом

доїння на основі зібраних даних про тварину, таких як інтенсивність доїння, стан молочних залоз, розташування сосків та інших параметрів, за допомогою датчиків, які встановлюються на доїльних апаратах.

Мікропроцесорний засіб має використовувати адаптивний алгоритм доїння, що розробляє оптимальний план доїння для кожної тварини індивідуально на основі зібраних даних. Алгоритм враховує індивідуальні особливості корови, її фізіологічні потреба та забезпечує максимальний комфорт та ефективність процесу доїння.

Крім того, відслідковувати доїння можна як через персональний комп’ютер, який з’єднаний з мікропроцесором та відображає на екрані всі необхідні дані, так і через сам інтерфейс користувача, який дозволяє налаштовувати параметри алгоритму, відстежувати стан доїльних апаратів, отримувати інформацію про продуктивність кожної тварини тощо.

Прикладом універсального автоматизованого блока доїння може слугувати «Доїльний блок «БД-05»[10]. Цей блок є одним з пристроїв, які використовуються в сільському господарстві для автоматизації процесу доїння корів. Пристрій був розроблений для забезпечення ефективності, комфорту та безпеки під час доїння тварин.

Рис. 2.1 – Зовнішній вигляд доїльного блоку «Блок доїння «БД-05»

«БД-05» має такі основні компоненти:

- Доїльний апарат: цей елемент складається з датчиків ізоляції, реєстратора молока, вакуумного насоса та всмоктувального пристрою. Він розміщується на вимені корови під час доїння та забезпечує відцентровий негативний тиск для збору молока. Ізоляційні датчики контролюють якість зв’язку між доїльним апаратом і вим’ям, щоб запобігти втраті вакууму та забезпечити ефективне доїння.
- Система контролю: БД-05 оснащений комп'ютерною системою управління, яка контролює хід процесу доїння. Ця система має датчики для вимірювання кількості молока, вакууму та часу доїння. Використовуючи ці дані, він регулює параметри процесу доїння, забезпечуючи корові оптимальні умови та максимальне споживання молока.
- Молочний блок: цей компонент включає систему збору та транспортування молока. Його можна підключити до доїльного апарату та автоматично закривати молокопроводи після доїння. Молочний агрегат також може мати вбудовану систему фільтрації та охолодження молока для забезпечення його якості та тривалого збереження.
- Дані та моніторинг: БД-05 можна підключити до комп’ютерної мережі, дозволяючи контролювати та збирати дані про процес доїння. Він дозволяє фермерам аналізувати ефективність процесу доїння, контролювати виробництво молока, виявляти будь-які аномалії та проблеми, які виникають під час доїння.

Доїльний апарат «БД-05» - сучасний автоматизований пристрій, який дозволяє підвищити ефективність доїння, покращити якість молока та комфорт для корів. Він забезпечує ефективний контроль і автоматизацію процесу доїння, заощаджуючи час і зусилля фермерів, підвищуючи продуктивність і добробут тварин.

В ході розробки архітектури інформаційної підсистеми було створено схему майбутнього мікропроцесорного засобу для реалізації адаптивного алгоритму доїння(рис. 2.2). Схема передбачає всі необхідні елементи для автоматизованої роботи пристрою та взаємодії з ним, а саме: матрична клавіатура, світлові індикатори, електричні кола, зв'язок з персональним комп'ютером.

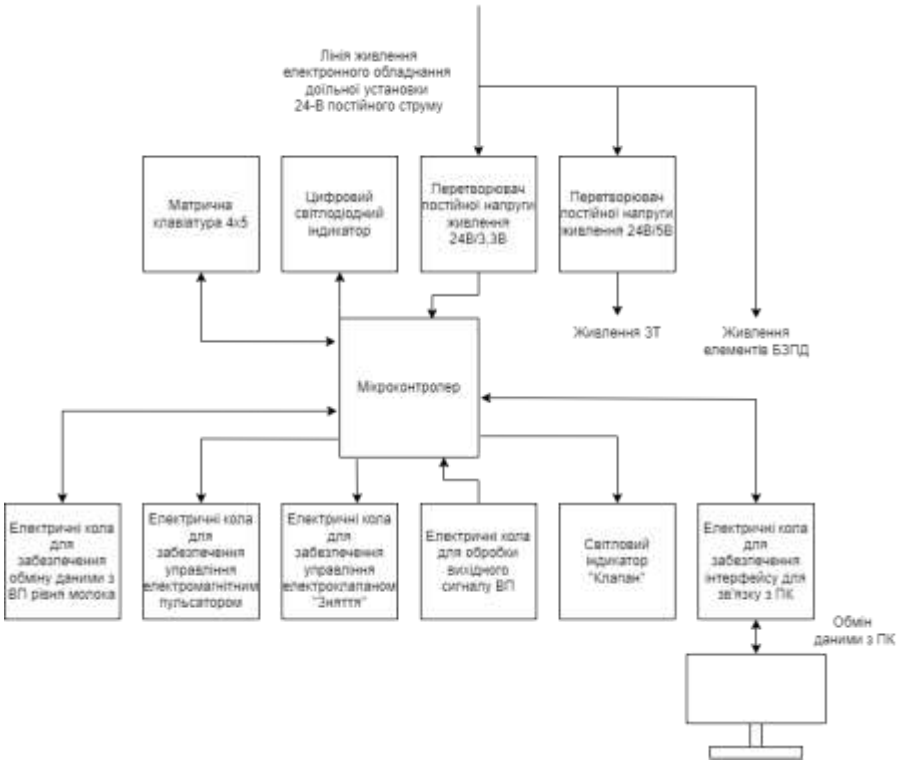

Рис. 2.2 – Схема мікропроцесорного засобу для реалізації адаптивного алгоритму доїння

Найважливішим елементом доїльного апарату є пульсатор через те, що він і забезпечує доїння тварини, створюючи вакуум. Призначенням пульсатора є створення пульсуючого вакууму з необхідними параметрами доїльних склянок доїльного апарату з метою видоювання тварини. Пульсатор складається з двох електроклапанів з незалежним керуванням. Управляються вони завдяки транзисторних ключів. На рис. 2.3 відображені стандартні циклограми керуючих сигналів для транзисторних ключів, що формуються завдяки роботі мікроконтролеру, і відповідні циклограми зміни вакууму в доїльних стаканах. Керуючі сигнали для транзисторних ключів зсунуті між

собою на 180 градусів по фазі для забезпечення нормальної роботи доїльної установки.

Виникнення переднього та заднього фронту при збільшенні та зменшенні вакууму в дійкових стаканах обумовлено перехідними процесами, що відбуваються в системі, і інерцією електромагнітних клапанів.

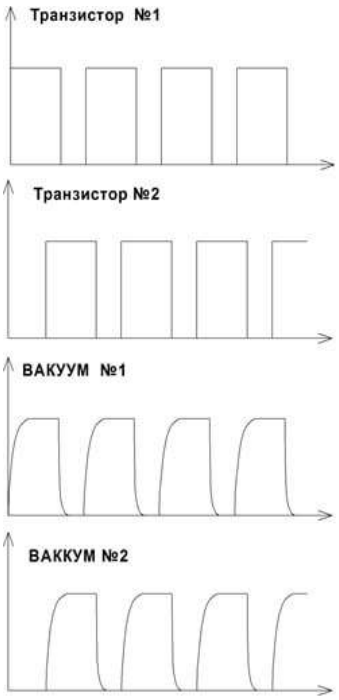

Рис 2.3 – Часові діаграми роботи пульсатора

На рис. 2.4, що наведений нижче, можна побачити циклограму пульсуючого вакууму в доїльних склянках та назви часових інтервалів відповідно до ISO. На циклограмі виділяються такі часові інтервали:

- фаза доїння – період часу, протягом якого відкритий транзистор керування електромагнітним клапаном.
- фаза спокою – час, протягом якого закритий транзистор керування електромагнітним клапаном.
- фаза А - інтервал часу, в якому вакуум зростає до максимального значення.
- фаза В - проміжок часу, в який значення вакууму є максимальним.

- фаза C - період часу, за який вакуум падає до мінімального значення.
- фаза D – інтервал часу, в якому значення вакууму є мінімальним.
- фаза A+B - інтервал часу, в якому вакуум зростає до максимального значення і підтримує максимальне значення.
- фаза C+D - інтервал часу, в якому вакуум падає до мінімального значення і підтримує мінімальне значення.

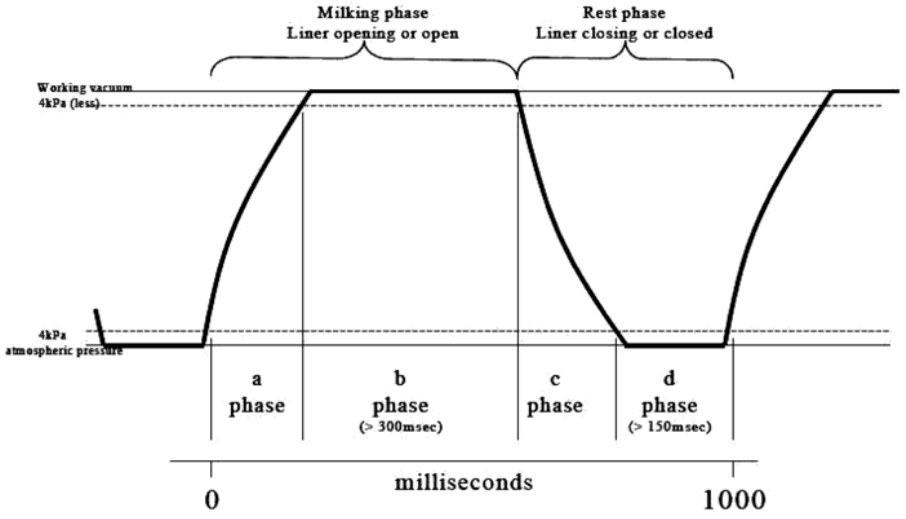

Рис. 2.4 – Цикл пульсації

Електромагнітний пульсатор має два режими роботи: швидкий та ручний. У швидкому режимі вмикається перший пульсатор, який працює 0.67 секунди, після чого в нього відбувається фаза відпочинку, що триває 0.33 секунди, другий пульсатор працює по такому ж принципу, але запускається він через 0.5 секунди після запуску першого для того, щоб доїння відбувалось рівномірно. Ручний режим, в свою чергу, має схожий стиль, але фаза роботи в нього триває 0.6 секунд, а відпочинку – 0.4 с. Другий пульсатор в ручному режимі також запускається через 0.5 секунди після запуску першого. Режим роботи швидкого та ручного пульсаторів можна побачити на рис. 2.5.

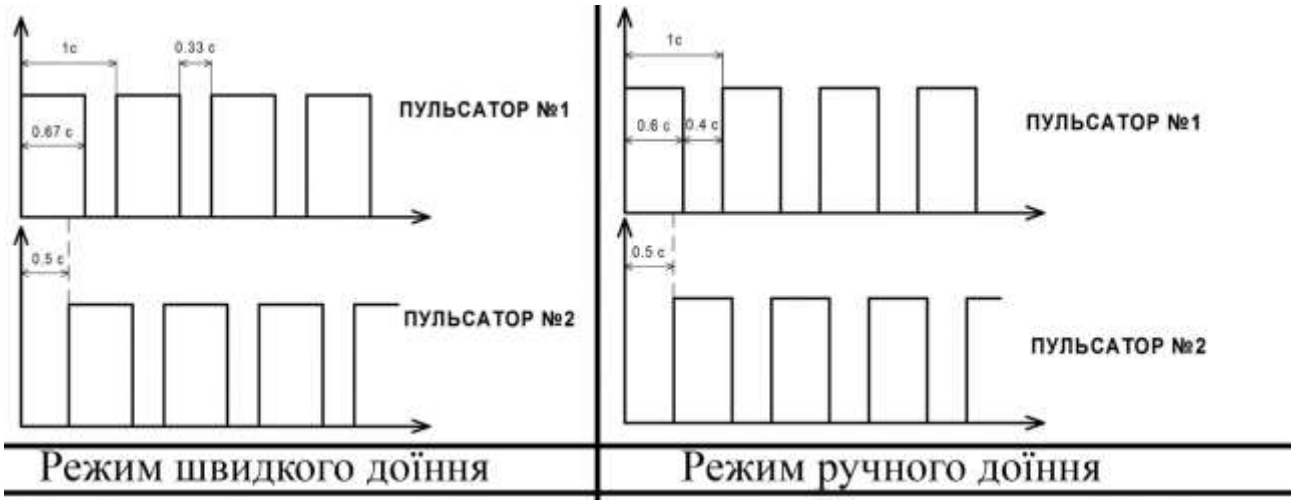

Рис. 2.5 – Режими роботи пульсатора

На основі ілюстрацій та їх опису вище розроблено план, за яким відбувається доїння:

1) Під’єднується та запускається апаратура до тварини;
2) Вмикається фаза швидкого доїння;
3) Перевіряється інтенсивність молоковіддачі, якщо вона дорівнює 600 грам за хвилину, то відбувається перехід до ручного режиму доїння.
4) Ручний режим доїння;
5) Перевірка інтенсивності молоковіддачі. Якщо інтенсивність 200 грам за хвилину і менше, доїльний апарат відключається та знімається оператором.

Отже, у рамках розробки архітектури інформаційної підсистеми для реалізації адаптивного алгоритму доїння були враховані основі вимоги та параметри, що дозволяють створити ефективний та надійний засіб для доїння тварини та підвищення продуктивності молочного скоту.

РОЗДІЛ 3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Розробка програмного забезпечення для мікроконтролера

Для розробки програмного забезпечення мікроконтролера було розроблене математичне формулювання основних аспектів задачі. Отже, математична формалізація:

- Параметри фізіологічного стану тварини: для досягнення адаптивності алгоритму доїння необхідно враховувати параметри, що характеризують фізіологічний стан тварини. До цих параметрів слід віднести такі показники: частота серцевих скорочень, температура тіла, зовнішній огляд тварини, дата останнього доїння та інші.
- Стан доїльної системи: для ефективного контролю та управління доїнням необхідно враховувати стан доїльної машини. Цей пункт повинен містити такі показники, як тиск, час доїння, потужність насоса та ін.
- Алгоритм доїння: для визначення оптимальних параметрів доїння необхідно розробити математичний алгоритм, який враховує отримані дані про фізіологічний стан тварини та стан доїльної системи. Цей алгоритм може базуватися на різних методах, таких як статистичний аналіз, адаптивні алгоритми керування.
- Режим доїння: режим доїння є особливо важливою частиною дослідження. В даному випадку розроблено альтернативний варіант режиму доїння: при включені доїльного апарату запускається швидкий

режим доїння, через деякий час, коли інтенсивність молоковіддачі досягає 600 гр/хв, вмикається ручний режим доїння. Режим доїння повинен забезпечувати оптимальну ефективність і якість доїння.

Математична формалізація задачі полягає у визначенні математичних величин, функцій і залежностей, що відображають зв’язки між параметрами фізіологічного стану тварини, станом доїльного апарату та алгоритмом доїння.

Отже, на основі цих даних було розроблено програмний код для мікроконтролера на мові C++ з використанням Arduino. Сам код містить необхідні поля та методи для реалізації адаптивного алгоритму доїння. Код наведений в додатку А.

3.1.1 Огляд коду

Код має наступні змінні та константи:

- pulsatorPin1 та pulsatorPin2 – піни, до яких підключені пульсатори;
- prevMillis, currentMillis – змінні для відстеження часу молоковіддачі;
- pulseDuration, restDuration – тривалість імпульсу та паузи в автоматичному режимі;
- manualPulseDuration, manualRestDuration – тривалість імпульсу та паузи в ручному режимі доїння;
- flowrate – змінна, що зберігає значення інтенсивності молоковіддачі;
- fastMode, manualMode – значення цих змінних вказують на поточний режим роботи доїльного апарату.

Основною функцією є controlPulsators(), яка керує роботою електромагнітного пульсатору залежно від встановленого режиму. В автоматичному режимі пульсатори увімкнені через певний інтервал часу, який визначається pulseDuration та restDuration. У ручному режимі пульсатори ввімкнені відповідно за manualPulseDuration та manualRestDuration.

Основний код реалізований в методах setup() та loop() і виконує наступні дії:

- Ініціалізує піни для пульсаторів та зв’язок для передачі даних на персональний комп’ютер через Serial;
- У безкінечному циклі loop() викликається функція controlPulsators() для керування пульсаторами.
- У залежності від поточного режиму роботи(fastMode або manualMode), вимірює інтенсивність молоковіддачі та виконує відповідні дії.

Як можна було зрозуміти, програма має два режими роботи: швидкий та ручний. Також, постійно вимірюється інтенсивність молоковіддачі, завдяки якій працює перехід між режимами та вимкнення апаратури. Швидкий режим: у цьому режимі програма керує пульсаторами залежно від заданого інтервалу – це 0.67 с для роботи, та 0.33 – для відпочинку. Якщо інтенсивність молоковіддачі дорівнює 600 гр/хв, система переходить у ручний режим. Ручний режим: програма керує пульсаторами так, як і в швидкому, але інтервал – 0.6 та 0.4 с відповідно.

3.1.2 Опис методу loop()

Loop() є основним методом програми. При запуску доїльного апарата, мікроконтролер відразу розпочинає роботу з методу setup(), а далі loop(). В методі setup() відбувається базове налаштування, а в loop() – описується алгоритм роботи. Метод наведений нижче:

```
void loop() {
  controlPulsators();

  if (fastMode) {
    // Вимірювання інтенсивності молоковіддачі
    flowRate = analogRead(A0);
    float litersPerMinute = flowRate * 0.2;  //
Калібровка значення молоковіддачі
    Serial.print("Інтенсивність молоковіддачі: ");
    Serial.print(litersPerMinute);
```

```
    Serial.println(" гр/хв");

    // Перевірка інтенсивності молоковіддачі для переходу
в ручний режим
    if (litersPerMinute >= 600) {
      fastMode = false;
      manualMode = true;
      Serial.println("Відбувся перехід до ручного режиму
роботи");
    }
  } else if (manualMode) {
    // Вимірювання інтенсивності молоковіддачі
    flowRate = analogRead(A0);
    float litersPerMinute = flowRate * 0.2;  //
Калібровка значення молоковіддачі
    Serial.print("Інтенсивність молоковіддачі: ");
    Serial.print(litersPerMinute);
    Serial.println(" гр/хв");

    // Перевірка інтенсивності молоковіддачі для
відключення електромагнітного пульсатора
    if (litersPerMinute <= 200) {
      fastMode = false;
      manualMode = false;
      digitalWrite(pulsatorPin1, LOW);  // Вимкнути
перший пульсатор
      digitalWrite(pulsatorPin2, LOW);  // Вимкнути
другий пульсатор
      Serial.println("Інтенсивність молоковіддачі <= 200.
Відключення електромагнітного пульсатору.");
```

```
    }
  }
  delay(10);
}
```

Отже, відразу коли програма переходить до методу loop() запускається метод controlPulsators(), який регулює пульсатори в залежності від режиму. Далі відбувається перевірка, якщо доїльний апарат працює в швидкому режимі(fastMode), то опрацьовується код if для істини, якщо ж ручний режим(manualMode) – для хибності.

Код, якщо швидкий режим: вимірюється інтенсивність молоковіддачі, в наступному ж рядку вона множиться на 0.2. Множення відбувається через те, що за замовчуванням значення, яке отримується з функції analogRead(A0), надається в діапазоні від 0 до 1023. Цей діапазон відповідає вхідній напрузі аналогового піна A0, який може бути від 0 до 5 вольт. Отже, необхідним є переведення в літри на хвилину, це відбувається завдяки калібруванню та множенням результату на 0.2. Число 0.2 використовується як коефіцієнт перетворення, який необхідно налаштовувати відповідно до характеристик сенсора і системи молоковіддачі. Після зчитування даних молоковіддачі відбувається відправка цих даних на персональний комп’ютер за допомогою функції print класу Serial. Пізніше йде перевірка інтенсивності молоковіддачі, якщо вона дорівнює або більше 600 гр/хв, відбувається перехід до ручного режиму доїння встановленням флажка false для fastMode та true для manualMode, і після чого відправляється інформація про перехід до іншого режиму на ПК.

Код, якщо ввімкнений ручний режим: початок аналогічний – відбувається зчитування молоковіддачі та передача її до додатку на персональному комп'ютері. Потім відбувається перевірка інтенсивності молоковіддачі. Якщо вона дорівнює або менше 200 гр/хв, доїльний апарат вимикається,

встановлюючи флажки false для fastMode та manualMode і вимикаючи пульсатори методом digitalWrite().

3.1.3 Опис методу controlPulsators()

Якщо метод loop() є основним методом програми, для забезпечити роботу мікропроцесора, то метод controlPulsators() – є найголовнішим методом з цього коду через те, що він забезпечує алгоритм доїння. Метод controlPulsators():

```
void controlPulsators() {
  currentMillis = millis();

  if (fastMode) {
    // Обчислення, які пульсатори повинні бути увімкнені або вимкнені
    if ((currentMillis - prevMillis) >= pulseDuration) {
      digitalWrite(pulsatorPin1, LOW);  // Вимкнути перший пульсатор
    }
    if ((currentMillis - prevMillis) >= (pulseDuration + 335)) {
      digitalWrite(pulsatorPin2, LOW);  // Вимкнути другий пульсатор
    }
    if ((currentMillis - prevMillis) >= (pulseDuration + restDuration{
      digitalWrite(pulsatorPin1, HIGH);  // Увімкнути перший пульсатор
    }
```

```
    if ((currentMillis - prevMillis) >= (pulseDuration +
restDuration + 335)) {
      digitalWrite(pulsatorPin2, HIGH);  // Увімкнути
другий пульсатор
    }
  } else if (manualMode) {
    // Обчислення, які пульсатори повинні бути увімкнені
або вимкнені
    if ((currentMillis - prevMillis) >=
manualPulseDuration) {
      digitalWrite(pulsatorPin1, LOW);  // Вимкнути
перший пульсатор
    }
    if ((currentMillis - prevMillis) >=
(manualPulseDuration + 300)) {
      digitalWrite(pulsatorPin2, LOW);  // Вимкнути
другий пульсатор
    }
    if ((currentMillis - prevMillis) >=
(manualPulseDuration + manualRestDuration)) {
      digitalWrite(pulsatorPin1, HIGH);  // Увімкнути
перший пульсатор
    }
    if ((currentMillis - prevMillis) >=
(manualPulseDuration + manualRestDuration + 300)) {
      digitalWrite(pulsatorPin2, HIGH);  // Увімкнути
другий пульсатор
    }
  }
```

```
  prevMillis = currentMillis;  // Оновлення попереднього
часу
}
```

Цей метод викликається з методу loop() кожен раз, як проходить повний цикл. В методі є два основних блоки коду: if(fastMode) та else if(manualMode). За допомогою логічної перевірки відбувається перехід до певного блоку(повільного чи ручного доїння).

Блок швидкого доїння встановлюється за замовчуванням в програмі через те, що коли доїльний апарат запускається він повинен працювати в режимі швидкого доїння. Цей блок коду містить в собі 4 логічних оператори if, саме які керують доїнням. За допомогою змінних prevMillis та currentMillis вимірюється час пульсації. Отже, різниця `currentMillis - prevMillis` дорівнює часу пульсації, а змінна pulseDuration та restDuration дорівнюють сталим значенням 670(час роботи пульсатора) та 330(час відпочинку пульсатора) відповідно. Таким чином відбувається керування швидким доїнням з використанням 4 блоків if:

1) Якщо час роботи пульсатора більший або дорівнює 670(pulseDuration), то перший пульсатор вимикається.

```
if ((currentMillis - prevMillis) >= pulseDuration) {
    digitalWrite(pulsatorPin1, LOW);  // Вимкнути
перший пульсатор
}
```

2) Якщо час роботи пульсатора більший або дорівнює 670(pulseDuration) + 325, то вимикається другий пульсатор. 325 – це половина часу роботи пульсатора, використовується дане значення для налагодження роботи пульсаторів: один працює, інший відпочиває.

```
if ((currentMillis - prevMillis) >= (pulseDuration +
335)) {
```

```
    digitalWrite(pulsatorPin2, LOW);  // Вимкнути
другий пульсатор
}
```

3) Якщо час роботи пульсатора більший або дорівнює 670(pulseDuration) + 330(restDuration), то вмикається перший пульсатор. Отже, якщо час роботи пульсатора дорівнює одній секунді, значить перший пульсатор пройшов повний цикл і може знову бути запущеним.

```
if ((currentMillis - prevMillis) >= (pulseDuration +
restDuration{
    digitalWrite(pulsatorPin1, HIGH);  // Увімкнути
перший пульсатор
}
```

4) Якщо час роботи пульсатора більший або дорівнює 670(pulseDuration) + 330(restDuration) + 335(час старту другого пульсатора), то означає, що другий пульсатор пройшов всю фазу і може бути ввімкненим.

```
if ((currentMillis - prevMillis) >= (pulseDuration +
restDuration + 335)) {
    digitalWrite(pulsatorPin2, HIGH);  // Увімкнути
другий пульсатор
}
```

Принцип роботи ручного режиму аналогічний швидкому, але використовує інші значення часу для роботи та відпочинку. В кінці методу йде оновлення часу роботи пульсатора:

```
prevMillis = currentMillis;
```

Отже, таким чином працює алгоритм роботи мікропроцесора для адаптивного доїння. Всі значення змінних повинні бути змінені відповідно до мікропроцесора, який використовується, та аналізу даних, на основі яких відбувається налаштування процесу доїння.

3.2 Зв'язок між ПЗ мікроконтролера та персонального комп'ютера

Для розробки програмного забезпечення безпосередньо для персонального комп’ютера використовують різні технології, однією з найпопулярніших є передача даних за допомогою послідовного порту(COM-порту)[7-9]. COM-порт є одним з типів послідовних портів, які використовуються для зв’язку з різними пристроями, включаючи мікроконтролери, модеми, принтери та інші пристрої. У Windows COM-порти представлені як COM1, COM2, COM3 і так далі, в Linux і Mac вони можуть мати інші назви, наприклад /dev/ttyUSB0. COM-порти дозволяються передавати дані в обидві сторони між ПК і підключеними пристроями, використовуючи послідовний протокол для передачі даних.

Для передачі даних за допомогою COM-порту в Java існує бібліотека RXTX, яка надає можливості для роботи з послідовними портами в програмах Java. Вона дозволяє встановлювати з’єднання з COM-портом і зчитувати або записувати дані через нього. Ця бібліотека забезпечує переносимість між різними операційними системами, що дозволяє розробляти програмне забезпечення для зчитування даних з COM-порту на різних платформах без необхідності вручну налаштовувати драйвери COM-портів для кожної операційної системи.

В коді програмного забезпечення мікроконтролера, який наведено в додатку А, можна побачити наступні рядки:

```
Serial.print("Інтенсивність молоковіддачі: ");
Serial.print(litersPerMinute);
Serial.println(" гр/хв");
```

У цьому фрагменті коду можна побачити як передається повідомлення про інтенсивність молоковіддачі з мікроконтролера на послідовний порт персонального комп’ютера за допомогою функцій Serial.print() та

Serial.println(). За допомогою бібліотеки RXTX, що прослуховує послідовний порт з додатку на ПК, дані можна зчитати безпосередньо в коді на Java за допомогою методу serialEvent()(додаток Б). В цьому методі Java-код аналізує рядки, що надійшли через послідовний порт, та здійснює необхідні дії, наприклад, виводить повідомлення у консоль Java або ж записує дані до бази даних.

Клас Serial в Arduino за своєю функціональністю схожий на клас RXTX в Java, проте відрізняється синтаксисом. Serial використовується для взаємодії з послідовним портом, через який можна передавати дані між Arduino та зовнішнім пристроєм, таким як комп'ютер. Клас Serial містить різноманітні методи для читання та запису даних через персональний порт. Одним з цих методів є print(), використання якого можна побачити вище. Цей метод призначений для виводу даних у послідовний порт зі зміною рядка. Синтаксис методу виглядає наступним чином:

```
Serial.println(data);
```

Тут "data" є значенням, яке передається у послідовний порт. Метод перетворює це значення у рядок та виводить його у COM-порт, після чого він може бути зчитаним з коду Java. Використання методу дозволяє зручно виводити повідомлення у послідовний порт, які можна переглянути та обробити зовнішніми пристроями, що взаємодіють з Arduino через порт. Зчитування даних з порту можна побачити в наступному пункті з розробкою програмного забезпечення для персонального комп'ютера.

3.3 Розробка програмного забезпечення для персонального комп'ютера

Для того, щоб можна було працювати з автоматизованим доїльним апаратом та відслідковувати якісь важливі деталі на персональному комп'ютері, було розроблено додатковий код на мові Java(додаток Б) для

отримання та обробки даних, надісланих із мікроконтролера Arduino через COM-порт.

Код створює клас ArduinoDataReceiver, який реалізує інтерфейс SerialPortEventListener для обробки подій COM-порту.

В основному методі програми підключення до COM-порту мікроконтролера Arduino здійснюється за допомогою бібліотеки gnu.io. Спочатку визначається ідентифікатор порту, пов’язаний із COM-портом «COM1». Якщо порт не зайнятий, то відбувається його відкриття.

Якщо об’єкт CommPort, який повертається під час відкриття порту, є екземпляром класу SerialPort, тоді для об’єкта serialPort встановлюється прослуховувач подій. ArduinoDataReceiver додається як слухач подій і встановлюється сповіщення про дані, які надійшли.

Отримання даних із COM-порту здійснюється в методі serialEvent, який реалізує обробник подій SerialPortEvent. Отримавши подію DATA_AVAILABLE, він перевіряє наявність даних для читання. Якщо рядок починається з фрази «Відбувся перехід до ручного режиму роботи», то на графіку вибирається фаза режиму і виводиться повідомлення про зміну режиму доїння. Якщо рядок починається з фрази «Інтенсивність молоковіддачі: », то значення інтенсивності доїння береться з терміну і виводиться повідомлення про поточну інтенсивність доїння.

Код також містить метод setReader, який налаштовує зовнішній об’єкт BufferedReader для читання даних із COM-порту. Він використовується для забезпечення зв’язку між ArduinoDataReceiver і кодом, який завантажений на мікропроцесор, де дані виводяться на монітор порту.

3.3.1 Опис методу main()

Метод main() – це основний метод будь-якої програми, розробленої на мові Java, з нього починається запуск алгоритму, він описує виклик інших необхідних методів по порядку. В даній програмі, в цьому методі описується встановлення з'єднання з COM-портом, детальніше з яким можна ознайомитись в пункті 3.2, де йдеться про передачу даних між мікропроцесором та персональним комп'ютером. Метод main():

```
public static void main(String[] args) {
    try {
    // Встановлення з'єднання з COM-портом
мікроконтролера Arduino
    CommPortIdentifier portIdentifier =
CommPortIdentifier.getPortIdentifier("COM1");
    if (portIdentifier.isCurrentlyOwned()) {
        System.err.println("COM-порт вже
використовується");
    }else {
        CommPort commPort =
portIdentifier.open(ArduinoDataReceiver.class.getName(),
2000);
         if (commPort instanceof SerialPort) {
             SerialPort serialPort = (SerialPort)
commPort;
serialPort.addEventListener(new ArduinoDataReceiver());
serialPort.notifyOnDataAvailable(true);
            InputStream inputStream =
serialPort.getInputStream();
            InputStreamReader inputStreamReader = new
InputStreamReader(inputStream);
```

```
            BufferedReader bufferedReader = new
BufferedReader(inputStreamReader);
            ArduinoDataReceiver dataReceiver = new
ArduinoDataReceiver();
dataReceiver.setReader(bufferedReader);
          }
      }
      } catch (Exception e) {
          e.printStackTrace();
      }
    }
```

Метод починається відразу з обробки помилок, які можуть виникнути в ході роботи програми, за допомогою блоку try-catch. Try – це блок коду, в якому потенційно можуть виникнути помилки в роботі, наприклад може бути зайнятий порт. Catch – в свою чергу повідомляє про помилку, яка виникла в блоці try. Якщо ж код виконаний без перешкод, то catch пропускається. Отже, порядок роботи блоку try:

1) Спроба визначити ідентифікатор порту COM1 за допомогою CommPortIdentifier.getPortIdentifier("COM1").
2) Перевірка, чи порт вже використовується за допомогою portIdentifier.isCurrentlyOwned(). Якщо порт вже використовується, виводиться повідомлення про помилку.
3) Якщо порт не використовується, виконується відкриття порту за допомогою portIdentifier.open(ArduinoDataReceiver.class.getName(), 2000). 2000 вказує таймаут для відкриття порту (2 секунди).
4) Перевірка, чи порт є екземпляром класу SerialPort за допомогою commPort instanceof SerialPort. Якщо це так, виконуються наступні дії.
5) Створення екземпляра SerialPort під назвою serialPort для подальшої роботи з ним.

6) Додавання об'єкту ArduinoDataReceiver як прослуховувача подій порту serialPort за допомогою serialPort.addEventListener(new ArduinoDataReceiver()). Це дозволяє отримувати сповіщення про доступні дані, які надходять в порт з мікроконтролера.
7) Встановлення прапорця notifyOnDataAvailable на true для того, щоб порт serialPort повідомляв про наявність даних.
8) Отримання потоку вводу inputStream з порту serialPort за допомогою serialPort.getInputStream().
9) Створення InputStreamReader inputStreamReader на основі inputStream для зручного читання символів з потоку.
10) Створення BufferedReader bufferedReader на основі inputStreamReader для зручного читання рядків з потоку.
11) Створення об'єкту ArduinoDataReceiver під назвою dataReceiver.
12) Встановлення створеному об'єкту dataReceiver раніше створеного bufferedReader за допомогою dataReceiver.setReader(bufferedReader).

3.3.2 Огляд методу serialEvent()

SerialEvent() є основним методом цього додатку, тому що в ньому відбувається виведенням даних, для програми ж головним методом є за замовчуванням main(), в якому відбувається налаштування та перехід до відповідних методів. Отже, метод serialEvent():

```
@Override
    public void serialEvent(SerialPortEvent event) {
        if (event.getEventType() ==
SerialPortEvent.DATA_AVAILABLE) {
            try {
                String line = reader.readLine();
```

```
                if (line.startsWith("Відбувся перехід до
ручного режиму роботи")) {
                    String phase = line.substring(30);
                    System.out.println("Режим доїння
змінено. Теперішній режим: " + phase);
                } else if (line.startsWith("Інтенсивність
молоковіддачі: ")) {
                    String flowRate = line.substring(29);
                    System.out.println("Інтенсивність
молоковіддачі: " + flowRate + " гр/хв");
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
```

Метод має анотацію @Override, це означає те, що цей метод перевизначений так як він є методом інтерфейсу SerialPortEventListener та відповідає за прослуховування повідомлень, що мають прийти до додатку. На початку методу відбувається перевірка події event, яка виникає на серійному порті. Умова «event.getEventType() == SerialPortEvent.DATA_AVAILABLE» перевіряє, чи тип події є DATA_AVAILALE, що вказує на наявність нових даних для читання з порту. Якщо умова виконується і виникає подія з новими доступними даними, тоді виконується наступний блок коду. А саме відбувається зчитування нових даних. Все це відбувається в блоці try – для обробки помилок, якщо щось піде не так в ході зчитування інформації, запуститься код catch, який виведе текст про помилку в консоль. Отже, в блоці try зчитується рядок, надійшовший з мікроконтролера, якщо текст починається з «Відбувся перехід до ручного режиму роботи», то цей рядок відразу обрізається на 30 символів(саме тих, які перевірялись) та виводиться

інформація про те, що режим доїння було змінено. Якщо ж текст починається з фрази «Інтенсивність молоковіддачі: », то вже відкидається 29 символів і виводиться в консоль теперішній рівень молоковіддачі.

ВИСНОВОК

В ході кваліфікаційної роботи було вирішено завдання автоматизації процесу доїння тварин за допомогою розробленого мікропроцесорного засобу. Важливість цього завдання полягає у підвищенні продуктивності та ефективності доїння, а також у забезпеченні оптимальних умов утримання тварин.

Результати дослідження включають програму, розроблену для мікроконтролера Arduino, яка керує пульсаторами та вимірює інтенсивність молоковіддачі. Реалізовано два режими роботи: автоматичний і ручний. Під час роботи програма періодично вимикає і вмикає пульсатори відповідно до заданих інтервалів. Програма також вимірює інтенсивність молоковіддачі за допомогою датчика та передає дані на комп’ютер через серійний порт. Якщо

інтенсивність перевищує встановлений поріг, система автоматично переходить в ручний режим і працює з новими інтервалами.

На персональному комп’ютері реалізована програма на мові Java для отримання даних від мікроконтролера Arduino. Вона підключається до COM-порту мікроконтролера і отримує дані про зміну режимів та інтенсивності доїння. Отримані дані відображаються на консолі, в подальшому вони відправляються до бази даних, після чого обробляються для покращення виробництва.

Розроблений адаптивний алгоритм доїння дозволяє підтримувати оптимальну інтенсивність молоковіддачі, що сприяє підвищенню продуктивності та комфорту тварин.

Таким чином, в результаті проведеної роботи розроблено повноцінний програмний код мікропроцесорного засобу для реалізації адаптивного алгоритму доїння, що дозволяє автоматично керувати режимом роботи пульсаторів та забезпечувати оптимальну інтенсивність доїння. Розроблений засіб можна рекомендувати для використання в фермерських господарствах з метою поліпшення якості доїння та підвищення ефективності виробництва.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Домдузис К. Radio–frequency identification (RFID) applications: A brief introduction / K. Domdouzis, B. Kumar, C. Anumba // Adv. Engineering Informatics. – 2007
2. BS ISO 5707:2007. Milking machine installations. Construction and performance. – 30 March 2007
3. DeLaval / Tetra Laval Group. – Режим доступу : www.delaval.com.
4. Датчик інтенсивності молоковіддачі переносного доїльного апарата для стійлового молокопроводу / В. Ю. Кучерук, П. І. Кулаков, Є. А. Паламарчук, Т. В. Гнесь // Вимірювальна та обчислювальна техніка в технологічних процесах. – 2013

5. Brandt M. Invited review: Technical solutions for analysis of milk constituents and abnormal milk / M. Brandt , A. Haeussermann, E. Hartung // Journal of Dairy Science. – 2010.
6. Fundamentals of Sensors for Engineering and Science / Patrick F. Dunn – 2017
7. Arduino Starter Kit Manual: A Complete Beginners Guide to the Arduino / Earthshine Electronics
8. Java: A Beginner's Guide, Eighth Edition / Herbert Schildt – 2018
9. The c++ programming language \ Bjarne Stroustrup – 2013
10. Монографія «Елементи теорії вимірювального контролю параметрів біотехнічної системи доїння» - Кулаков П.І.