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 інформаційної підсистеми для доїльної установки з молокопроводом

Виконав – студент 4 курсу, групи 41-кн
освітньої програми "Комп'ютерні науки"
спеціальності 122 Комп'ютерні науки
Бондар Євгеній Ігорович
Науковий керівник – д.т.н., професор
Кулаков Павло Ігорович

Умань – 2023

Зміст

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ..... 7

ВСТУП……..8

РОЗДІЛ 1. АНАЛІТИЧНИЙ ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ..... 10

1.1. Об’єкт та предмет дослідження 10

1.2. Постановка задачі..... 13

РОЗДІЛ 2. ВИБІР ТЕХНОЛОГІЙ ТА АРХІТЕКТУРА ПІДСИСТЕМИ 15

2.1. Вибір технологій розробки..... 15

2.2. Архітектура підсистеми..... 27

РОЗДІЛ 3. РОЗРОБКА ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ДЛЯ ДОЇЛЬНОЇ УСТАНОВКИ З МОЛОКОПРОВОДОМ 31

3.1. Розробка структурної схеми інформаційної підсистеми для доїльної установки з молокопроводом 31

3.2. Розробка програмного забезпечення інформаційної підсистеми для доїльної установки з молокопроводом……...34

3.3. Розробка програмного забезпечення для мікроконтролера інтерфейсного блока..... 40

ВИСНОВКИ..... 43

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 45

ДОДАТОК А..... 46

ДОДАТОК Б 55

Перелік умовних позначень, скорочень і термінів

БЗПД	Блок забезпечення процесу доїння
БЗОД	Блок забезпечення обміну даними
ЗВКМ	Засіб вимірювання кількості молока
ЗТ	Зчитувач транспондерів
БУДМ	Блок управління дозатором молока
ЛПМ	Лічильник порцій молока
АСУ	Автоматизована система управління.
БД	База даних.
ПЗ	Програмне забезпечення.
ПД	Програмний додаток.
ПЗП	Постійний запам’ятовуючий пристрій.
RTOS	Операційна система реального часу

Вступ

Молочне скотарство є складною та трудомісткою складовою системи сільськогосподарського виробництва. Технологічні процеси виробництва молока базуються на низці взаємопов'язаних процесів та операцій, які формують систему і технологію виробництва продукції тваринництва. На більшості українських молочних ферм рівень інформатизації та автоматизації в середньому не перевищує 70-75%, що збільшує собівартість молочної сировини. Враховуючи сучасний стан вітчизняного виробництва молока з недостатнім рівнем автоматизації та інформатизації, ступінь підвищення продуктивності праці за рахунок традиційних методів вирощування, годівлі, обліку та доїння досягла свого максимального значення. Як наслідок, в більшості випадків використовується 60-70% потенційної продуктивності тварин. Це пов'язано з широким використанням застарілого обладнання та неналежним рівнем організації і технічного забезпечення тваринницьких ферм.

Актуальність

Технологічний процес виробництва молока є складною біотехнічною системою, що включає техніку, технологію, групи тварин і бригади операторів, а також організацію, інформатизацію та управління технологічними процесами. Аналіз зарубіжного досвіду показує, що впровадження системної інформатизації та централізованого управління тваринницькою фермою вимагає змін в організаційно-технічних і технологічних формах управління. Тому перехід молочного скотарства на більш ефективну технологічну базу та його інформатизація є важливим і актуальним завданням. Наразі відсутня у промисловому виробництві інформаційна підсистема, яка базується на сучасних інформаційних технологіях та забезпечує отримання та представлення у необхідному вигляді параметрів найбільш поширеної в Україні доїльної установки - доїльної установки з молокопроводом та лічильниками молока для кожного оператора.

Об’єкт дослідження – процес та концепція побудови інформаційної підсистеми для доїльної установки з молокопроводом.

Предмет дослідження – методи інформатизації тваринницьких ферм обладнаних доїльною установкою з молокопроводом.

Мета дослідження

Метою та завданням бакалаврської кваліфікаційної роботи є концептуальна розробка інформаційної підсистеми параметрів функціонування доїльної установки з молокопроводом на основі сучасної технології «Internet of things», структурної схеми цієї підсистеми та структурних схем її компонентів, розробка програмного забезпечення для інформаційної підсистеми та програмного забезпечення мікроконтролерів, які входять до складу окремих компонентів підсистеми.

Під час виконання кваліфікаційної роботи бакалавра був проведений аналіз наукової та технічної літератури, технічної документації, міжнародних стандартів у галузі інформатизації та автоматизації технологічних процесів, а також проведено розробку структурної схеми інформаційної підсистеми параметрів функціонування доїльної установки з молокопроводом, розробку структурних схем її окремих компонентів та програмних модулів.

РОЗДІЛ 1. АНАЛІТИЧНИЙ ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Об’єкт та предмет дослідження

При прив’язному утриманні дійних тварин застосовуються доїльні установки з молокопроводом, які мають не більше вісьми стійлових ліній. На рис. 1.1 наведена типова структурна схема інформаційної підсистеми параметрів функціонування доїльних установок з молокопроводом, які мають у своєму складі автоматичну або автоматизовану систему ідентифікації тварин [1].

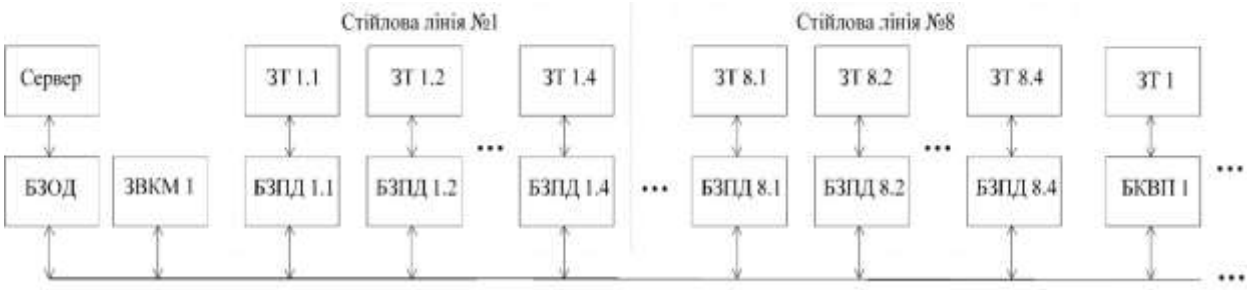


Рисунок 1.1 – Структурна схема інформаційної підсистеми параметрів функціонування доїльних установок з молокопроводом, які мають у своєму складі автоматичну або автоматизовану систему ідентифікації тварин

Контроль переважної кількості параметрів функціонування у наведеній підсистемі здійснюється за допомогою блока забезпечення процесу доїння (БЗПД) під час доїння. Після закінчення доїння тварин, результати вимірювання та контролю параметрів функціонування, номер транспондера та номер тварини, передаються до сервера підсистеми за посередництва блока забезпечення обміну даними (БЗОД), за допомогою засобу вимірювання кількості молока (ЗВКМ) вимзначається кількість отриманого молока, також може контролюватися вага тварин. У такий підсистемі блоки забезпечення процесу доїння обладнані зчитувачами транспондерів (ЗТ).

На рис. 1.2 наведено типовий варіант структурної схеми інформаційної підсистеми параметрів функціонування доїльних установок з молокопроводом

та ручною системою ідентифікації тварин. У цій системі відсутні зчитувачі транспондерів, номер тварини вводиться вручну до блока забезпечення процесу доїння за допомогою клавіатури (К). Якщо номер тварини або номер місця не буде введений доярем перед початком доїння, то робота блока забезпечення процесу доїння буде заблокована. В усьому іншому робота цієї системи аналогічна попередньо розглянутій.

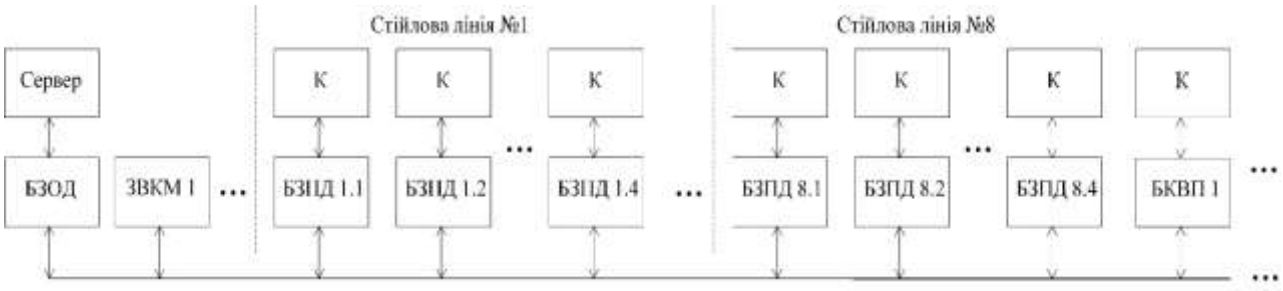


Рисунок 1.2 – Структурна схема інформаційної підсистеми параметрів функціонування доїльних установок з молокопроводом та ручною системою ідентифікацією тварин

На рис. 1.3 наведено типову структурну схему варіанта інформаційної підсистеми параметрів функціонування доїльних установок з молокопроводом, у якому ідентифікація тварин не здійснюється.

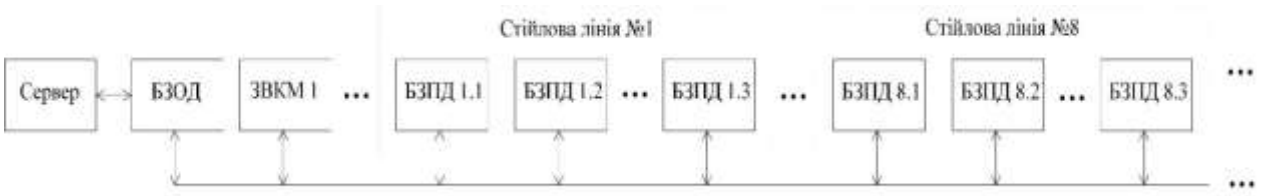


Рисунок 1.3 – Структурна схема інформаційної підсистеми параметрів функціонування доїльних установок з молокопроводом без ідентифікації тварин

Індивідуальні параметри тварин у такій системі не контролюються, за її допомогою можливе отримання інформації тільки про параметри груп тварин.

У наведеній на рис. 1.3 структурній схемі інформаційної підсистеми параметрів функціонування доїльних установок з молокопроводом може бути отримана інформація про середні значення деяких параметрів молока у кожній лінії, загальну кількість отриманого молока, загальний удій групи тварин у лінії, тривалість роботи установки, тривалість роботи кожної лінії, середній удій тварин окремих ліній та всього стада, середній час доїння тварин окремих ліній та всього стада, кількість повторних під'єднань доїльних апаратів у кожній лінії, кількість холостих доїнь у кожній лінії, тривалість холостих доїнь у кожній лінії, кількість відпадань доїльного апарата у кожній лінії. Уся необхідна для визначення цих параметрів інформація отримується за допомогою блока забезпечення процесу доїння, після закінчення доїння інформація передається до сервера підсистеми. Можливий варіант, коли у такій системі проводиться контроль ваги тварин, у цьому випадку, при зважуванні тварин здійснюється їх ручна ідентифікація.

На рис. 1.4 наведено найбільш розповсюджений варіант структурної схеми інформаційної підсистеми параметрів функціонування доїльних установок з молокопроводом, у якому не використовуються блоки забезпечення процесу доїння.

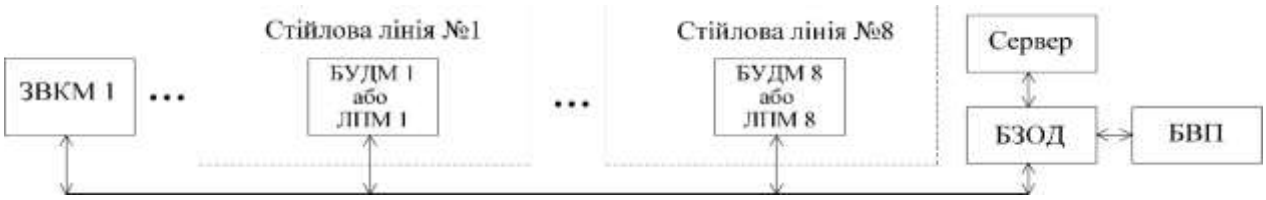


Рисунок 1.4 – Структурна схема інформаційної підсистеми параметрів функціонування доїльних установок з молокопроводом з відсутністю блоків забезпечення процесу доїння

Індивідуальні параметри тварин у такій системі не визначаються, можливе отримання інформації тільки про параметри певних груп тварин [2]. У цій інформаційній підсистемі параметрів функціонування доїльних установок з

молокопроводом отримується інформація про кількість порцій молока сформованих дозатором, загальну кількість отриманого молока, загальний удій групи тварин у лінії, тривалість роботи установки, тривалість роботи кожної лінії, середній удій тварин окремих ліній та всього стада, середній час доїння тварин окремих ліній та всього стада. Вимірювання та контроль цих параметрів здійснюється за допомогою блоків управління дозатором молока (БУДМ) або лічильників порцій молока (ЛПМ), які передають отриману інформацію до сервера підсистеми. У наведеній на рис. 1.4 підсистемі можливий контроль ваги тварин. Він здійснюється так само, як і у попередньо розглянутій підсистемі, структурна схема якої наведена на рис. 1.3.

Слід відзначити, що на доїльних установках з молокопроводом здебільше використовують ручну ідентифікацію, або не здійснюють ідентифікацію тварин взагалі. Спосіб ідентифікації нерухомих тварин, який застосований у системах на рис. 1.1 та рис. 1.2, збільшує трудомісткість роботи дояра. У [3] наведено систему автоматичної ідентифікації, у якому номера тварин та відповідні номера їх стійлових місць знаходяться у спеціалізованій базі даних, номер тварини автоматично візуалізується на індикаторі БЗПД, а дояр повинен перевірити його вірність та підтвердити або скорегувати його [4]. При зміні місця або групи, при виведенні тварини в сухостій виникає необхідність у корегуванні бази даних, що на практиці не завжди можна вчасно забезпечити. Окрім того, перевірка відповідності номера стійла номеру тварини та його корегування збільшує трудомісткість роботи дояра. Для забезпечення максимальної продуктивності доїльної установки з молокопроводом дояр повинен дотримуватись регламенту роботи, тобто доїння повинно відбуватись у певній послідовності у відповідності з номерами стійлових місць.

1.2 Постановка задачі

Ціллю бакалаврської кваліфікаційної роботи є розробка інформаційної підсистеми для доїльної установки з молокопроводом та

програмне забезпечення для мікроконтролерів, під управлінням яких працюють деякі її складові. Завданням розробки є створення на концептуальному рівні підсистеми, яка призначена для функціонування у складі інформаційної системи тваринницької ферми та є інноваційним рішенням для доїльної установки з молокопроводом, у якому використовується сучасна технологія «Internet of things» та реалізована концепція «Ферма в смартфоні».

Інформаційна підсистема, що розробляється, дозволяє користувачу у будь-який момент часу, за допомогою смартфона, персонального комп’ютера або планшетного комп’ютера отримувати інформацію про характеристики технологічного процесу виробництва молока на доїльній установці з молокопроводом, контролювати події, які відбуваються на доїльній установці, виявляти аварійні ситуації та факти порушень регламенту роботи персоналом, здійснювати облік удою та зоотехнічних параметрів тварин.

РОЗДІЛ 2. ВИБІР ТЕХНОЛОГІЙ ТА АРХІТЕКТУРА ПІДСИСТЕМИ

2.1 Вибір технології розробки

Syncthing. Підсистема, що розробляється, складається з певної кількості мікропроцесорних засобів, які забезпечують отримання інформації про визначені параметри технологічного процесу виробництва молока, а саме: зоотехнічних параметрів тварин, події на фермі під час доїння, параметри вакуумної установки, параметри промивання, та інші специфічні параметри. Усі ці данні надсилаються на сервер підсистеми, де зберігаються у визначеному форматі. Для відображення цієї інформації на смартфоні, персональному комп’ютері або планшетному комп’ютері користувача, необхідно забезпечити синхронізацію даних. Таким чином, для використання інформаційної підсистеми, на комп’ютерах та смартфонах користувача повинно бути належним чином встановлено та налаштовано спеціалізоване безкоштовне програмне забезпечення Syncthing [5].

Вищевказане програмне забезпечення дозволяє синхронізувати файли між декількома пристроями, підтримує збереження послідовних версій файлу за багатьма алгоритмам, може працювати в локальній мережі та мережі Internet, передавання усіх даних відбувається по захищених каналах з використанням протоколу обміну ключами з прямою секретністю. Синхронізація відбувається за датою зміни файлу, існує підтримка синхронізації на рівні блоків, тобто при невеликих змінах у файлі будуть синхронізовані тільки блоки, що змінилися, а не увесь файл. Особливістю програмного забезпечення є те, що жодна частина даних ніколи не зберігається ніде, крім налаштованих комп’ютерів. Не існує центрального сервера, доступ до якого можна отримати, усі комунікації захищені протоколом TLS, використаний механізм шифрування включає пряму секретність, яка виключає можливість доступу до даних. Кожен пристрій ідентифікується за криптографічним сертифікатом, тільки допущені пристрої можуть приєднатися до кластеру. Програмне забезпечення для інформаційної

підсистеми повинно працювати на операційних системах Mac OS X, Windows, Linux, FreeBSD і Solaris.

Особливості технології програмного забезпечення Syncthing полягають в наступному. Файли розбиті на блоки, блоки, що утворюють файл, мають однаковий розмір (крім останнього у файлі, який може бути меншим). Розмір блоку залежить від розміру файлу та варіюється від 128 KB до 16 MB. Кожен файл розрізається на кілька таких блоків, і для кожного блоку обчислюється хеш SHA256. Це призводить до списку блоків, що містить зсув, розмір і хеш усіх блоків у файлі.

Щоб оновити файл, Syncthing порівнює список блоків поточної версії файлу зі списком блоків потрібної версії файлу. Потім він намагається знайти джерело для кожного блоку, який відрізняється. Це може бути локально, якщо інший файл уже має блок із таким самим хешем, або це може бути з іншого пристрою в кластері. У першому випадку блок просто копіюється на диск, у другому – запитується по мережі з іншого пристрою.

Коли блок копіюється або отримується з іншого пристрою, його хеш SHA256 обчислюється та порівнюється з очікуваним значенням. Якщо він збігається з блоком, записується до тимчасової копії файлу, інакше він відкидається, і Syncthing намагається знайти інше джерело для блоку.

Існує два методи, за допомогою яких Syncthing виявляє зміни: шляхом регулярного повного сканування та сповіщень, отриманих від файлової системи («спостерігач»). За замовчуванням спостерігач увімкнено, і повне сканування виконується раз на годину. Цю поведінку можна змінити для кожної папки. Збільшення повного інтервалу сканування використовує менше ресурсів і корисно, наприклад, для великих папок, які рідко змінюються. Щоб переконатися, що не всі папки скануються повторно одночасно, фактичний інтервал сканування є випадковим часом між 3/4 і 5/4 даного інтервалу сканування. Навіть якщо «спостерігач» увімкнено, рекомендується залишати

ввімкненим регулярне повне сканування, оскільки воно може не зафіксувати деякі зміни.

Під час повторного сканування наявні файли перевіряються на наявність змін у їхньому часі модифікації, розмірі чи бітах дозволу. Якщо на основі цих атрибутів виявляється зміна, файл «переробляється», тобто для файлу розраховується новий список блоків. Неможливо дізнатися, які частини файлу змінилися, не прочитавши файл і не обчисливши нові хеші SHA256 для кожного блоку.

«Спостерігач» не планує сканування одразу після виявлення зміни. Він накопичує зміни протягом 10 секунд, а видалені файли додатково затримуються на 1 хвилину. Тому очікується, що між внесенням змін і їх появою на іншому пристрої буде невелика затримка. Зміни, які були виявлені та хешовані, передаються на інші пристрої після кожного повторного сканування.

Syncthing відстежує кілька версій кожного файлу: версію, яка зараз є на диску, називається локальною версією, версії, оголошені всіма іншими підключеними пристроями, і «найкращу» (зазвичай останню) версію файлу. Ця версія називається глобальною версією, і кожен пристрій намагається її оновлювати. Ця інформація зберігається в індексованій базі даних, яка зберігається в каталозі конфігурації та називається index-vx.y.z.db. Коли з інших пристроїв надходять нові дані, Syncthing перераховує, яка версія для кожного файлу має бути глобальною версією, і порівнює її з поточною локальною версією. Якщо вони відрізняються, Syncthing має синхронізувати файл. Списки блоків порівнюються для створення списку необхідних блоків, які потім запитуються з мережі або копіюються локально.

Syncthing розпізнає конфлікти. Якщо файл було змінено на двох пристроях одночасно, а вміст фактично відрізняється, один із файлів буде перейменовано на <filename>.sync-conflict-<date>-<time>-<modifiedBy>.<ext>. Файл із давнішим часом модифікації буде позначено як конфліктний файл і, таким чином, буде перейменовано. Якщо модифікації однакові, файл, що

надходить із пристрою, який має більше значення перших 63 бітів для свого ідентифікатора пристрою, буде позначено як конфліктуючий файл. Якщо конфлікт виникає між модифікацією та видаленням файлу, змінений файл завжди виграє та відроджується без перейменування на пристрої, де його було видалено. Файли <filename>.sync-conflict-<date>-<time>-<modifiedBy>.<ext> після створення розглядаються як звичайні файли, тому вони поширюються між пристроями. Так зроблено тому що конфлікт виявляється та вирішується на одному пристрої, створюючи файл конфлікту синхронізації, але він так само сильно конфліктує всюди, і не відомо, який із конфліктуючих файлів є «найкращим» від користувача.

Syncthing працює з регістром шляхів, тобто file.txt і FILE.txt позначають дві незалежні речі. Отже, він ніколи не розглядає обидвафайли як такі, ніби вони якось пов’язані для синхронізації їхнього вмісту. Деякі операційні системи (наприклад, певною мірою Windows, Mac і Android) припускають протилежне, розглядаючи їх як той самий файл (або каталог) і ніколи не можуть мати обидві назви одночасно. Таким чином, якщо віддалений пристрій надає спільний доступ до файлу, який суперечить існуючому локальному файлу, його неможливо синхронізувати з такою системою, і Syncthing повідомить про це як конфлікт регістрів. Подібним чином, якщо два віддалені пристрої мають різні варіанти регістру для вашого локального пристрою, лише один із них буде вилучено, а інший позначено відповідним повідомленням про помилку. Щоб розв’язати таку конфліктну ситуацію регістру, потрібно вибрати узгоджене ім’я файлу та вручну застосувати його для всіх задіяних пристроїв, не чутливих до регістру.

Syncthing ніколи не записує безпосередньо у файл призначення. Натомість усі зміни вносяться до тимчасової копії, яка потім переміщується на стару версію. Якщо під час копіювання або синхронізації виникає помилка, наприклад необхідний блок недоступний, тимчасовий файл зберігається протягом доби. Це робиться для того, щоб уникнути непотрібного запиту даних

через мережу. Тимчасові файли називаються .syncthing.original-filename.ext.tmp або, у Windows, ~syncthing~original-filename.ext.tmp, де original-filename.ext — ім’я цільового файлу. Зазвичай тимчасовий файл приховано. Якщо назва тимчасового файлу буде надто довгою через додавання префікса та додаткового розширення, тимчасові файли називаються .syncthing.<hash>.tmp або, у Windows, ~syncthing~<hash>.tmp, де <hash> це хеш SHA-256 оригінального імені файлу.

IAR Embedded Workbench. Для створення програмного забезпечення для мікроконтролерів архітектури AVR виробництва компанії Microchip, під управлінням яких працюють складові інформаційної підсистеми, використовується мова програмування C++ з пакету програм IAR Embedded Workbench for AVR. IAR Embedded Workbench є повним набором інструментів для розробки вбудованого програмного забезпечення, що надає розробнику повністю інтегроване середовище розробки, яке включає компілятор, асемблер, лінкер і налагоджувач [6].

Компілятор IAR C/C++ для AVR — це сучасний компілятор, який пропонує стандарт мов C і C++, а також їх спеціальні розширення. Асемблер IAR для AVR — це потужний асемблер макросів із універсальним набором директив і операторів, асемблер має вбудований препроцесор мови C. Асемблер IAR для AVR використовує ту саму мнемоніку та синтаксис операнда, що і асемблер Microchip Technology AVR, що спрощує міграцію існуючого коду.

IAR XLINK Linker — це потужний, гнучкий програмний інструмент для використання в розробці додатків для вбудованих систем. Він однаково добре підходить для зв’язування невеликих, однофайлових, абсолютних асемблерних програм, як і для зв’язування великих, переміщуваних вхідних, багатомодульних, C/C++ або змішаних C/C++ і асемблерних програм. Для обробки бібліотек включені бібліотечні інструменти XAR і XLIB.

Компілятор IAR C/C++ для AVR підтримує:

- C, найпоширеніша мова програмування високого рівня в галузі вбудованих систем. Є можливість створювати автономні програми, які відповідають цим стандартам:

- Стандарт C — також відомий як C99.

- Стандарт C89 — також відомий як C94, C90, C89 і ANSI C. Цей стандарт потрібен, якщо ввімкнено MISRA C.

- C++, сучасна об'єктно-орієнтована мова програмування з повнофункціональною бібліотекою, яка добре підходить для модульного програмування. Можна використовувати будь-який із цих стандартів: Embedded C++ (EC++) — підмножина стандарту програмування C++, який призначений для програмування вбудованих систем; IAR Extended Embedded C++ (EEC++)— EC++ із додатковими функціями, такими як повна підтримка шаблонів, множинне успадкування, підтримка простору імен, нові оператори приведення, а також стандартна бібліотека шаблонів. Кожну з підтримуваних мов можна використовувати в суворому чи послабленому режимі або в послабленому з увімкненими розширеннями IAR. Суворий режим дотримується стандарту, тоді як послаблений режим допускає деякі загальні відхилення від стандарту. Також можна реалізувати частини програми або всю програму на мові асемблера.

Компілятор IAR C/C++ для AVR підтримує всі пристрої на основі стандарту Microchip Technology AVR. Інсталяція продукту IAR містить попередньо налаштовані файли для підтримки різних пристроїв. Якщо потрібні додаткові файли для підтримки пристрою, їх можна створити за допомогою одного з наданих шаблонів. Стандартні периферійні пристрої визначаються у файлах заголовків вводу-виводу для конкретного пристрою з розширенням імені файлу h. Пакет продукту містить файли вводу-виводу для багатьох пристроїв, доступних на момент випуску продукту. Якщо потрібні додаткові файли заголовків вводу/виводу, їх можна створити за допомогою одного з наданих шаблонів. Каталоги avr\config і avr\src\template містять готові файли

конфігурації лінкера для всіх підтримуваних пристроїв. Файли мають розширення імені xcl і містять інформацію, необхідну лінкеру.

Налагоджувач обробляє кілька специфічних для пристрою вимог, таких як визначення доступних областей пам’яті, периферійних регістрів і їх груп, використовуючи файли опису пристрою. Ці файли розташовані в каталозі avr\config і мають розширення ddf. Периферійні регістри та їх групи можуть бути визначені в окремих файлах (розширення імені файлу sfr), які в такому випадку включені до файлу ddf.

Як правило, вбудоване програмне забезпечення, написане для спеціального мікроконтролера, розроблено як нескінченний цикл, який очікує, що відбудуться якісь зовнішні події. Програмне забезпечення знаходиться в ПЗП і виконується після скидання. Необхідно враховувати кілька апаратних і програмних факторів, коли створюється програмне забезпечення такого типу.

Вбудовані системи, як правило, містять різні типи пам’яті, такі як вбудована RAM, зовнішня DRAM або SRAM, ROM, EEPROM або флеш-пам’ять. Розробник вбудованого програмного забезпечення повинен розуміти особливості різних типів пам’яті. Наприклад, вбудована оперативна пам’ять часто швидша за інші типи пам’яті, і змінні, до яких часто звертаються, у критичних за часом програмах виграють від розміщення тут. Навпаки, доступ до деяких конфігураційних даних може здійснюватися рідко, але вони повинні зберігати своє значення після вимкнення живлення, тому їх слід зберігати в EEPROM або флеш-пам’яті. Для ефективного використання пам’яті компілятор надає декілька механізмів керування розміщенням функцій і об’єктів даних у пам’яті. Якщо до мікроконтролера підключено зовнішні пристрої, може знадобитися ініціалізувати і керувати інтерфейсом, наприклад, за допомогою вибору контактів мікросхеми, а також виявляти і обробляти зовнішні сигнали переривання. Як правило, це потрібно ініціалізувати та контролювати під час виконання. Звичайним способом зробити це є використання спеціальних

функціональних регістрів. Зазвичай вони доступні за спеціальними адресами та містять біти, які керують конфігурацією чіпа.

У вбудованих системах використання переривань є методом негайної обробки зовнішніх подій; наприклад, визначення того, що була натиснута кнопка. Загалом, коли в коді виникає переривання, мікроконтролер негайно припиняє виконання коду, який він виконує, і замість цього починає виконувати програму переривання. Компілятор надає різні примітиви для керування апаратними та програмними перериваннями.

У всіх вбудованих системах код запуску системи виконується для ініціалізації системи — апаратного та програмного забезпечення — перед викликом основної функції програми. Розробник вбудованого програмного забезпечення повинен переконатися, що код запуску розташований за адресами спеціальної пам’яті або доступ до нього можна отримати за допомогою покажчика з векторної таблиці. Це означає, що код запуску та початкова векторна таблиця повинні бути розміщені в енергонезалежній пам’яті, такій як ROM, EPROM або flash. Програма C/C++ також потребує ініціалізації всіх глобальних змінних. Ця ініціалізація обробляється лінкером і кодом запуску системи разом.

У багатьох випадках вбудована програма є єдиним програмним забезпеченням, що працює в системі. Однак використання RTOS має деякі переваги. Наприклад, на час виконання завдань з високим пріоритетом не впливають інші частини програми, які виконуються в завданнях з нижчим пріоритетом. Зазвичай це робить програму більш детермінованою та може зменшити енергоспоживання за рахунок ефективного використання процесорного ядра і переведення його у стан нижчого енергоспоживання під час простою. Використання RTOS може полегшити читання та підтримку програми, а в багатьох випадках також зменшити її розмір. Код програми можна чітко розділити на завдання, які справді не залежать одне від одного. Це полегшує командну роботу, оскільки роботу з розробки можна легко розділити

на окремі завдання, які виконуються одним розробником або групою розробників. Використання RTOS зменшує залежність від апаратного забезпечення та створює чистий інтерфейс програми, що полегшує перенесення програми на інше цільове обладнання. Програмне забезпечення вбудованої програми зазвичай реалізовано як цикл, який або керується перериваннями, або використовує опитування для контролю зовнішньої взаємодії або внутрішніх подій. Для системи, керованої перериваннями, переривання зазвичай ініціалізуються на початку основної функції. У системі з поведінкою в режимі реального часу, там де швидкість реагування є критичною, може знадобитися багатозадачна система. Це означає, що програмне забезпечення має бути доповнено операційною системою реального часу. У цьому випадку RTOS і різні завдання також повинні бути ініціалізовані на початку основної функції.

Як правило, виконання вбудованої програми ніколи не повинно закінчуватися, тобто необхідно визначити правильну кінцеву поведінку. Щоб завершити програму контрольованим способом, необхідно викликати одну з функцій стандартної бібліотеки C exit, _Exit, або abort, або повернутися з main. Якщо повернення відбувається з основного циклу, виконується функція виходу, що означає, що викликаються деструктори C++ для статичних і глобальних змінних (тільки для C++), а всі відкриті файли закриваються. Звичайно, у разі неправильної логіки програми програма може здійснити неконтрольоване та ненормальне завершення роботи — системний збій.

Оптимізатор компілятора виконує, серед іншого, усунення мертвого коду, розповсюдження констант, вбудовування, усунення загальних підвиразів і зниження точності. Він також виконує оптимізацію циклу, наприклад усунення змінної індукції. Є можливість вибрати між кількома рівнями оптимізації, а для найвищого рівня можна вибрати між різними цілями оптимізації. Більшість оптимізацій зроблять додаток меншим і швидшим. Однак, коли це не так, компілятор використовує вибрану ціль оптимізації, щоб вирішити, як виконати оптимізацію. Рівень і ціль оптимізації можна вказати для всієї програми, для

окремих файлів і для окремих функцій. Крім того, можна вимкнути деякі окремі оптимізації, наприклад вбудовані функції.

У типовій програмі дані можна зберігати в пам’яті трьома різними способами:

- Автоматичні змінні. Усі змінні, які є локальними для функції, за винятком оголошених статичних, зберігаються або в регістрах, або в стеку. Ці змінні можна використовувати, поки функція виконується. Коли функція повертається до свого виклику, простір пам’яті більше не дійсний.

- Глобальні змінні, статичні змінні модуля та локальні змінні, оголошені статичними. У цьому випадку пам’ять виділяється раз і назавжди. Слово статичний у цьому контексті означає, що обсяг пам’яті, виділеної для цього типу змінних, не змінюється під час роботи програми.

- Динамічно розподілені дані. Програма може розміщувати дані в комплексі, де дані залишаються дійсними. Цей тип пам’яті корисний, коли кількість об’єктів невідома до виконання програми. Існують потенційні ризики, пов’язані з використанням динамічно розподілених даних у системах з обмеженим обсягом пам’яті або системах, які працюватимуть протягом тривалого часу.

Компілятор використовує різні типи пам’яті для доступу до даних, які розміщені в різних областях пам’яті. Існують різні методи доступу до областей пам’яті, і вони мають різну швидкодію, коли йдеться про простір коду, швидкість виконання та використання реєстру. Методи доступу варіюються від загальних, які можуть отримати доступ до повного простору пам’яті, до простих методів, які можуть отримати доступ до обмежених областей пам’яті. Кожен тип пам'яті відповідає одному методу доступу до пам'яті. Компілятор може створити код, який може ефективно отримувати доступ до даних. Наприклад, пам'ять, доступ до якої здійснюється за допомогою методу доступу до ближньої пам'яті, називається пам'яттю ближнього типу або просто ближньою пам'яттю. Вибравши модель пам’яті, обирається тип пам’яті за

замовчуванням, який використовуватиме програма. Проте можна вказати для окремих змінних або покажчиків різні типи пам’яті. Це дає змогу створити програму, яка може містити велику кількість даних, і в той же час переконатися, що змінні, які часто використовуються, розміщуються в пам’яті, до якої можна ефективно отримати доступ. Компілятор надає набір розширених ключових слів, які можна використовувати як атрибути пам'яті даних. Ці ключові слова дозволяють перевизначати тип пам’яті за замовчуванням для окремих об’єктів даних, що означає, що є можливість розміщувати об’єкти даних в інших областях пам’яті, крім пам’яті за замовчуванням. Це також означає, що можна точно налаштувати метод доступу для кожного окремого об’єкта даних, що призводить до меншого розміру коду. Змінні, визначені всередині функції, і не оголошені статичними, за стандартом С називаються автоматичними змінними. Деякі з цих змінних розміщуються в регістрах процесора; решта поміщаються в стек. З семантичної точки зору це еквівалентно. Основні відмінності полягають у тому, що доступ до регістрів є швидшим і що потрібно менше пам’яті порівняно з тим, коли змінні розташовані в стеку. Автоматичні змінні можуть активні до тих пір, поки виконується функція; коли функція повертається, пам'ять, виділена в стеку, звільняється.

Коли в коді виникає переривання, мікроконтролер негайно припиняє виконання коду, який він виконує, і замість цього починає виконувати програму переривання. Важливо, щоб середовище перерваної функції було відновлено після обробки переривання (це включає значення регістрів процесора та регістр стану процесора). Це дає можливість продовжити виконання вихідного коду після виконання коду, який обробив переривання. Мікроконтролер AVR підтримує багато джерел переривань. Для кожного джерела переривання можна записати програму переривання. Кожна процедура переривання пов'язана з номером вектора, який вказано в документації до мікроконтролера AVR від виробника мікросхеми. Якщо необхідно обробляти

кілька різних переривань за допомогою однієї процедури переривання, можна вказати кілька векторів переривань.

Для мікроконтролера AVR таблиця векторів переривань завжди починається з адреси 0x0 і розміщується в сегменті INTVEC. Вектор переривань є зміщенням у таблиці векторів переривань. Таблиця векторів переривань містить покажчики на процедури переривання, включаючи програму скидання. Пристрій AT90S80515 має 13 векторів переривань і один вектор скидання. З цієї причини необхідно вказати 14 векторів переривань, кожен з двох байтів. Якщо у визначенні функції переривання вказано вектор, таблиця векторів переривань процесора заповнюється. Також можна визначити функцію переривання без вектора. Це корисно, якщо програма здатна заповнювати або змінювати таблицю векторів переривань під час виконання. Функція моніторингу спричиняє відключення переривань під час виконання функції. При вході в функцію регістр стану зберігається, а переривання відключаються. При виході з функції вихідний регістр стану відновлюється, і таким чином також відновлюється стан переривання, який існував до виклику функції.

Інтерфейс вводу/виводу низького рівня використовується для зв'язку між програмою, що налагоджується, і самим налагоджувачем. Інтерфейс простий: C-SPY розмістить точки зупину на певних мітках асемблера в програмі. Коли код, розташований на спеціальних мітках, починає виконуватись, C-SPY отримає сповіщення та зможе виконати дію.

Компілятор надає кілька попередньо визначених функцій, які дозволяють прямий доступ до низькорівневих операцій процесора без використання мови асемблера. Ці функції відомі як внутрішні функції. Вони можуть бути дуже корисними, наприклад, у критичних за часом процедурах. Внутрішня функція виглядає як звичайний виклик функції, але насправді це вбудована функція, яку розпізнає компілятор. Внутрішні функції компілюються у вбудований код або як одну інструкцію, або як коротку послідовність інструкцій. Перевага

внутрішньої функції порівняно з використанням вбудованого асемблера полягає в тому, що компілятор має всю необхідну інформацію для належного інтерфейсу послідовності з розподілом регістрів і змінними. Компілятор також знає, як оптимізувати функції з такими послідовностями; те, що компілятор не може зробити з вбудованими послідовностями асемблера. Результатом є те, що отримується потрібну послідовність, належним чином інтегровану у код, і що компілятор може оптимізувати результат.

Компілятор пропонує функції C і широкий набір розширень, починаючи від особливостей, розроблених спеціально для ефективного програмування у галузі вбудованих систем, до послаблення деяких незначних проблем стандартів. Це огляд доступних розширень: Розширення мови IAR C; Розширені ключові слова; Прагматичні директиви. Директива #pragma визначена стандартом C і є механізмом для контрольованого використання розширень від виробника, щоб переконатися, що вихідний код залишається переносним. Компілятор надає набір попередньо визначених прагма-директив, які можна використовувати для контролю поведінку компілятора, наприклад, як він розподіляє пам'ять, чи дозволяє він розширені ключові слова та чи виводить попереджувальні повідомлення. Більшість прагма-директив попередньо обробляються, що означає, що макроси замінюються в прагма-директиві. Директиви pragma завжди включені в компіляторі. Для деяких із них також існує відповідне розширення мови C/C++; Розширення препроцесора. Препроцесор компілятора відповідає стандарту C. Компілятор також надає доступ до кількох пов’язаних з препроцесором розширень; Внутрішні функції. Внутрішні функції забезпечують прямий доступ до низькорівневих операцій процесора і можуть бути дуже корисними, наприклад, у критичних за часом процедурах. Внутрішні функції компілюються у вбудований код у вигляді окремої інструкції або у вигляді короткої послідовності інструкцій.

2.2. Архітектура підсистеми

Розроблена в процесі виконання кваліфікаційної бакалаврської роботи архітектура інформаційної підсистеми для доїльної установки з молокопроводом наведена на рис. 1.

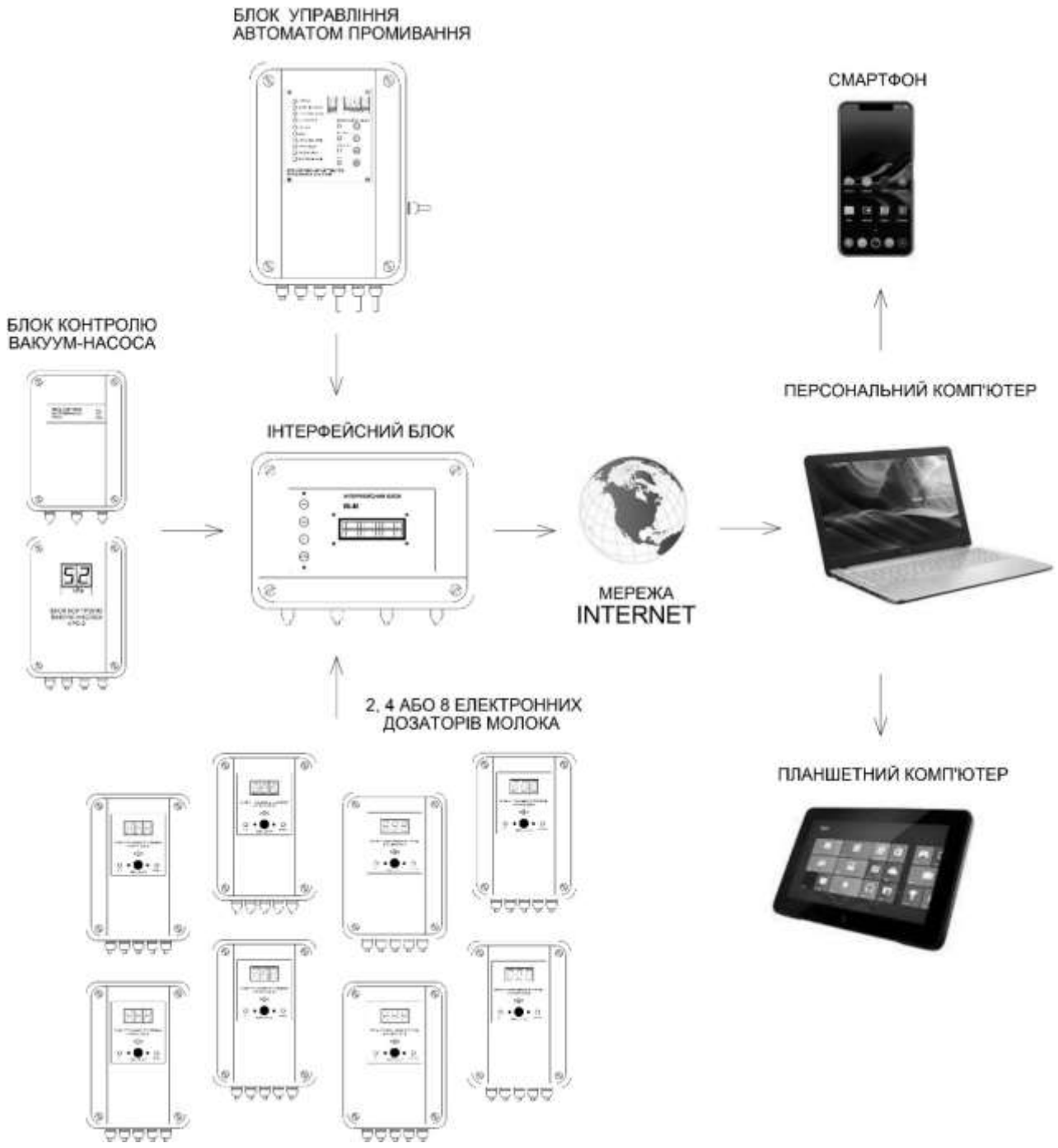


Рисунок 2.1 – Архітектура інформаційної підсистеми для доїльної установки з молокопроводом

Інформаційна підсистема складається з наступних компонентів:

- електронні дозатори молока, призначені для управління виконавчою системою дозатора молока, підрахунку кількості сформованих дозатором порцій молока, відображення та збереження інформації про кількість отриманих окремими доярами порцій молока, фіксації дати та часу зняття показань, а також для передавання даних про результат удою до інтерфейсного блоку;

- блока контролю вакуум-насоса, призначеного для фіксації дати та часу моментів ввімкнення та вимкнення вакуум-насоса та передавання цієї інформації до інтерфейсного блоку (опційно);

- блока контролю вакуум-насоса, призначеного для фіксації дати та часу моментів ввімкнення та вимкнення вакуум-насоса, вимірювання миттєвого та середнього значення вакуумметричного тиску у вакуумпроводі доїльної установки та передавання цієї інформації до інтерфейсного блоку (опційно);

- блока управління автоматом промивання, який використовується на доїльних установках з молокопроводом та призначений для автоматичного управління процесами переддоїльного промивання, післядоїльного промивання та дезінфекції, контролю параметрів технологічного процесу промивання та передавання інформації про параметри технологічного процесу промивання до інтерфейсного блоку;

- інтерфейсного блоку, який використовується на доїльних установках з молокопроводом у складі інформаційної підсистеми, що розробляється, та призначений для приймання даних про події, що відбуваються на доїльній установці (інформація про події надходить від електронних дозаторів молока, блока управління автоматом промивання, блока контролю вакуум-насоса), для фіксації дати та часу цих подій, обробки отриманих даних, представлення їх у необхідному вигляді та передавання цих даних користувачу через мережу Інтернет.

- спеціалізованого програмного забезпечення, яке встановлене на інтегрованому в інтерфейсний блок одноплатному промисловому комп’ютері, що здійснює обробку, представлення у необхідному вигляді та запис на карту пам’яті даних, які надходять від інших складових компонентів підсистеми, здійснює синхронізацію даних про події на доїльній установці, які знаходяться на смартфонах або комп’ютерах користувача, з даними, які знаходяться на карті пам’яті одноплатного промислового комп’ютера;

- спеціалізованого програмного забезпечення для мікроконтролерів складових елементів підсистеми (інтерфейсного блоку, електронних дозаторів молока, блока управління автоматом промивання, блока контролю вакуум-насоса), яке написано на мові програмування С++ з пакету програм IAR Embedded Workbench for AVR.

РОЗДІЛ 3. РОЗРОБКА ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ДЛЯ ДОЇЛЬНОЇ УСТАНОВКИ З МОЛОКОПРОВОДОМ

3.1. Розробка структурної схеми інформаційної підсистеми для доїльної установки з молокопроводом

Розроблена у процесі виконання кваліфікаційної бакалаврської роботи структурна схема інформаційної підсистеми для доїльної установки з молокопроводом наведена на рис. 3.1.

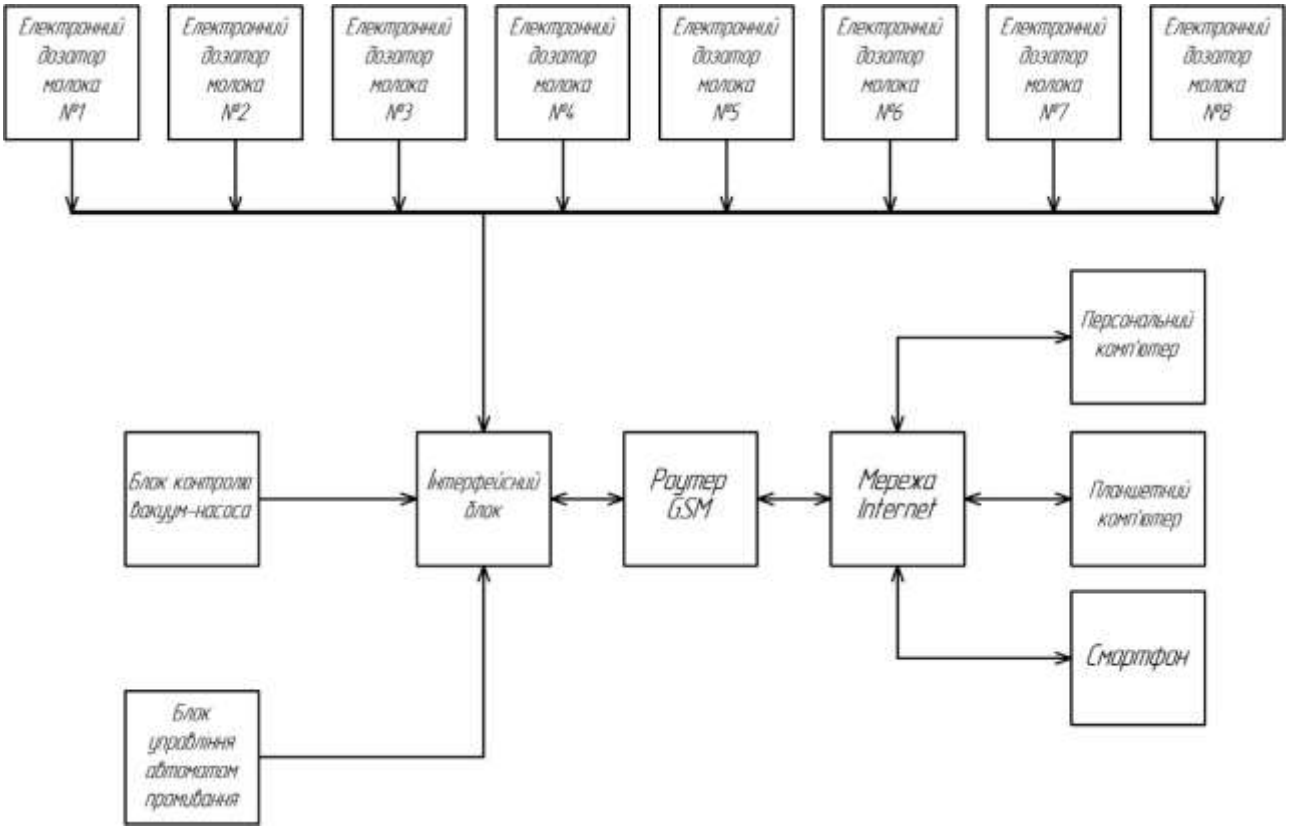


Рисунок 3.1 – Структурна схема інформаційної підсистеми для доїльної установки з молокопроводом

Принцип роботи інформаційної підсистеми для доїльної установки з молокопроводом полягає в наступному [7]. Інформаційна підсистема, яка забезпечує отримання, аналіз та представлення у необхідному вигляді інформацію про параметри функціонування доїльної установки з молокопроводом - це рішення у якому використовується сучасна технологія «Internet of things» (Інтернет речей) та реалізована концепція «Ферма в смартфоні». Підсистема призначена для віддаленого контролю параметрів технологічного процесу виробництва коров'ячого молока на доїльних установках з молокопроводом (УДМ і аналогічні). Уся інформація про усі події з моменту інсталяції підсистеми зберігається у спеціальній базі даних.

Підсистема дозволяє користувачу у будь-який момент часу (за допомогою смартфона, персонального комп'ютера або планшета) отримувати інформацію про параметри технологічного процесу виробництва коров'ячого молока, контролювати події, які відбуваються на доїльній установці, здійснювати облік удою. Особливо важливою її функцією є здатність виявляти аварійні ситуації та факти порушень регламенту роботи персоналом ферми, а саме:

- доволі часто зустрічаються факти нехтування персоналом ферм своїми службовими обов'язками при проведенні промивання доїльної установки - переривання промивання з метою залишення робочого місця; виконання менш тривалої програми «Переддоїльне промивання» замість програм «Промивання» та «Дезінфекція». Такі порушення значно впливають на якість промивання і відповідно, на якість молока.

- якщо на фермі виникли проблеми з водопостачанням і протягом 10 хвилин при виконанні будь-якої із програм промивання в баку автомата промивання відсутній достатній рівень води, промивання аварійно переривається і інформація про цю подію фіксується у базі даних підсистеми.

- якщо на фермі виникли проблеми з нагрівачем води (вийшов з ладу бойлер, встановлена недостатня температура води, не надходить гаряча вода

тощо), то факт наявності такої проблеми встановлюється шляхом контролю температури миючого розчину під час виконання програм промивання.

- інтерфейсний блок призначений для цілодобової роботи, і відповідно, для цілодобового контролю подій на фермі. Якщо відбулося вимкнення інтерфейсного блоку (для приховування порушення регламенту роботи або з інших причин), то після відновлення його функціонування ця подія буде зафіксована.

- один і той самий час початку доїння є дуже важливим для забезпечення високої продуктивності тварин. Шляхом контролю часу увімкнення та вимкнення вакуумної установки можна контролювати час початку та закінчення доїння.

- шляхом аналізу тривалості виконання програм промивання та фіксації фактів перепрограмування автомата промивання можна встановити факт несанкціонованих змін у налаштуваннях автомата промивання.

- шляхом аналізу послідовності подій, які відбуваються на фермі, можна встановити факт відхилення від регламенту роботи персоналом ферми.

- шляхом контролю значення вакумметричного тиску можна попередити виникнення маститу у тварин та систематичне відпадання доїльних стаканів.

- на основі аналізу послідовності подій, які відбуваються на фермі, у певних випадках можна здійснювати віддалену діагностику доїльного обладнання.

- на основі аналізу тривалості доїння можна зробити висновок про якість підготовки тварин та про виконання персоналом ферми регламенту доїльних операцій.

Пісистема може складатися з наступних компонентів: електронний дозатор молока (2, 4 або 8 дозаторів); блок контролю вакуум-насоса; блок управління автоматом промивання; інтерфейсний блок.

Спеціалізоване програмне забезпечення, яке встановлене на інтегрованому в інтерфейсний блок одноплатному промисловому комп’ютері, що здійснює

обробку, представлення у необхідному вигляді та запис на карту пам’яті даних, які надходять від інших складових компонентів системи, здійснює синхронізацію даних про події на доїльній установці, які знаходяться на смартфонах або комп’ютерах користувача, з даними, які знаходяться на карті пам’яті одноплатного промислового комп’ютера.

3.2. Розробка програмного забезпечення інформаційної підсистеми для доїльної установки з молокопроводом

Для використання інформаційної підсистеми на комп’ютерах та смартфонах користувача повинно бути належним чином встановлено та налаштовано спеціалізоване безкоштовне програмне забезпечення Syncthing, яке поставляється у складі системи. Вищевказане програмне забезпечення дозволяє синхронізувати файли між декількома пристроями, підтримує збереження послідовних версій файлу за багатьма алгоритмам, може працювати в локальній мережі та мережі Internet, передавання усіх даних відбувається по захищених каналах з використанням протоколу обміну ключами з прямою секретністю. Синхронізація відбувається за датою зміни файлу, існує підтримка синхронізації на рівні блоків, тобто при невеликих змінах у файлі будуть синхронізовані тільки блоки, що змінилися, а не увесь файл. Особливістю програмного забезпечення є те, що жодна частина даних ніколи не зберігається ніде, крім налаштованих комп’ютерів. Не існує центрального сервера, доступ до якого можна отримати, усі комунікації захищені протоколом TLS, використаний механізм шифрування включає пряму секретність, яка виключає можливість доступу до даних.

Кожен пристрій ідентифікується за криптографічним сертифікатом, тільки допущені пристрої можуть приєднатися до кластеру. Програмне забезпечення для інформаційної підсистеми повинно працювати на операційних системах Mac OS X, Windows, Linux, FreeBSD і Solaris [8].

На початку кожної доби спеціалізоване програмне забезпечення у визначеній папці створює спеціальний log-файл, до якого послідовно, у вигляді

строк, записується інформація про події, які відбуваються на доїльній установці. Файл має розширення .csv, та формується у стандартному форматі для баз даних. Даний формат файлу відкривається стандартними додатками практично будь-якої операційної системи. Назва файлу формується у форматі РРРР_ММ_ДД_log, де РРРР - рік, ММ - номер місяцю, ДД - дата. Так, наприклад, log-файл сформований 18 липня 2023 року буде мати наступну назву: 2023_07_18_log.csv. Усі log-файли знаходяться у єдиній, визначеній під час налаштування папці, та доступні для перегляду у будь-який момент часу.

У таблиці 3.1 наведено перелік подій на доїльній установці з молокопроводом, які фіксуються у інформаційній підсистемі для доїльної установки з молокопроводом, яка розробляється.

3.1. Перелік подій на доїльній установці з молокопроводом, які фіксуються за допомогою інформаційної підсистеми

№	Подія	Код події
Автомат промивання		
1	Увімкнення автомата промивання	Р-00
2	Запуск програми «Переддоїльне промивання»	Р-01
3	Запуск програми «Промивання»	Р-02
4	Запуск програми «Дезінфекція»	Р-03
5	Закінчення програми «Переддоїльне промивання»	Р-04
6	Закінчення програми «Промивання»	Р-05
7	Закінчення програми «Дезінфекція»	Р-06
8	Ручне переривання програми «Переддоїльне промивання»	Р-07
9	Ручне переривання програми «Промивання»	Р-08
10	Ручне переривання програми «Дезінфекція»	Р-09
11	Немає води у баку	Р-10
12	Вхід в режим програмування	Р-11
13	Вихід з режиму програмування	Р-12
14	Температура води у баку, хх - температура в градусах Цельсія	РСхх
Вакуумна установка		
15	Увімкнення вакуумної установки	V-01
16	Вимкнення вакуумної установки	V-00
17	Середнє значення вакуумметричного тиску, хх - величина	V%хх

	вакуумметричного тиску у кПа	
Інтерфейсний блок		
18	Увімкнення інтерфейсного блоку	I-01
19	Помилка передавання даних у підсистемі	ERRx
Електронні дозатори молока		
20	Результат удою, a - номер електронного дозатора молока, b - результат удою	a-bbb
21	Увімкнення електронних дозаторів молока	V-ON

Дані у log-файл записуються послідовно строками, кожній строці відповідає окрема подія. Для кожної події вказується час події, дата, код події, коментар до події. У таблиці 3.2 наведено перелік усіх можливих подій, які фіксуються у інформаційні підсистемі, формат та коди усіх подій (вони відображаються на індикаторі інтерфейсного блоку та у записах log-файлу), та типові приклади коментарів, які формуються у log-файлі та відповідають кожній події.

3.2. Події, які відбуваються на доїльній установці,
їх коди та приклади коментарів у log-файлі

№	**Подія**	**Код події**	**Приклад коментарю у log-файлі**
1	Удій кожного дояра.	a-bbb, a-номер, b - удій	2, Дозатор, 340
2	Увімкнення автомата промивання.	P-00	Автомат промивання, Ввімкнення
3	Запуск програми «Переддоїльне промивання».	P-01	Переддоїльне промивання, Початок
4	Запуск програми «Промивання».	P-02	Промивання, Початок
5	Запуск програми «Дезінфекція».	P-03	Дезінфекція, Початок
6	Закінчення програми «Переддоїльне промивання».	P-04	Переддоїльне промивання, Закінчення
7	Закінчення програми «Промивання».	P-05	Промивання, Закінчення

8	Закінчення програми «Дезінфекція».	P-06	Дезінфекція, Закінчення
9	Ручне переривання програми «Переддоїльне промивання».	P-07	Переддоїльне промивання, Переривання, !!!
10	Ручне переривання програми «Промивання».	P-08	Промивання, Переривання, !!!
11	Ручне переривання програми «Дезінфекція».	P-09	Дезінфекція, Переривання, !!!
12	Відсутність води у баку автомата промивання.	P-10	Автомат промивання, Немає води в баку, !!!

Продовження таблиці 3.2

13	Температура води у баку автомата промивання.	PCxx, xx - температура °C	PC65,Температура води,65
14	Вхід в режим програмування автомата промивання.	P-11	Автомат промивання, Вхід в режим програмування, !!!
15	Вихід з режиму програмування автомата промивання.	P-12	Автомат промивання, Вихід з режиму програмування, !!!
16	Увімкнення електронних дозаторів молока.	V-ON	Дозатори, Ввімкнення
17	Увімкнення вакуумної установки.	V-01	Вакуум-насос,Запуск
18	Вимкнення вакуумної установки.	V-00	Вакуум-насос,Зупинка
19	Середнє значення вакуумметричного тиску протягом роботи вакуумної установки.	V%xx, xx - вакуумметричний тиск, кПа	Величина вакууму (кПа),48
20	Помилка сенсора вакуумметричного тиску.	V%00	Помилка сенсора вакууму, 00, !!!
21	Увімкнення інтерфейсного блоку.	I-01	I-01,Інтерфейсний блок, Ввімкнення, !!!
22	Результат тестування підсистеми.	TEST	Тестування підсистеми,OK
23	Помилка передавання даних в підсистемі.	ERRx	ERROR, Помилка даних, !!!

У таблиці 3.3 в якості прикладу, наведено типовий вміст та аналіз записів log-файлу, який формує інформаційна підсистема, при коректній роботі доїльної установки протягом доби при дворазовому доїнні.

3.3. Типовий вміст та аналіз записів log-файлу при коректній роботі доїльної установки протягом доби при дворазовому доїнні

Послідовні записи у log-файлі	**Аналіз запису**
00:01,18/07/23,TEST,Тестування підсистеми,ОК,	Проведено успішне тестування підсистеми
04:00,18/07/23,TEST,Тестування підсистеми,ОК,	Проведено успішне тестування підсистеми
05:56,18/07/23,V-ON,Дозатори,Ввімкнення,	Увімкнено дозатори молока
05:56,18/07/23,P-00,Автомат промивання,Ввімкнення,	Увімкнено автомат промивання
05:57,18/07/23,V-01,Вакуум-насос,Запуск,	Запуск вакуум-насоса для прочищення молокопроводу за допомогою пижа
05:59,18/07/23,V-00,Вакуум-насос,Зупинка,	Зупинка вакуум-насоса після прочищення молокопроводу за допомогою пижа
05:59,18/07/23,V%22,Величина вакууму (кПа),22,	Середнє значення вакууму протягом часу прочищення молокопроводу за допомогою пижа (будь-яке значення цього параметра після прочищення молокопроводу за допомогою пижа не приймається до уваги)
06:02,18/07/23,V-01,Вакуум-насос,Запуск,	Запуск вакуум-насоса на початку першого добового доїння
08:00,18/07/23,TEST,Тестування підсистеми,ОК,	Проведено успішне тестування підсистеми
09:13,18/07/23,V-00,Вакуум-	Зупинка вакуум-насоса по закінченню

насос,Зупинка,	першого добового доїння, тривалість доїння 3 год. 11 хв.
09:13,18/07/23,V%49,Величина вакууму (кПа),49,	Середнє значення вакууму протягом доїння відповідає нормі (від 46 кПа до 50 кПа)
09:14,18/07/23,1,Дозатор,325,	Удій дояра на дозаторі №1 - 325 літрів
09:14,18/07/23,2,Дозатор,328,	Удій дояра на дозаторі №2 - 328 літрів
09:14,18/07/23,P-02,Промивання,Початок,	Початок виконання програми «Промивання»
09:16,18/07/23,V-01,Вакуум-насос,Запуск,	Запуск вакуум-насоса під час промивання

Продовження таблиці 3.3

09:29,18/07/23,PC54,Температура води,54,	Температура нагрітої води у баку автомата промивання 54 °С - відповідає нормі (не менше 35 °С)
10:04,18/07/23,V-00,Вакуум-насос,Зупинка,	Зупинка вакуум-насоса перед закінченням промивання
10:04,18/07/23,V%36,Величина вакууму (кПа),36,	Середнє значення вакууму протягом промивання відповідає нормі - 36 кПа (не менше 35 кПа)
10:04,18/07/23,P-05,Промивання,Закінчення,	Успішно закінчилося промивання, тривалість промивання 50 хв.
12:00,18/07/23,TEST,Тестування підсистеми,OK,	Проведено успішне тестування підсистеми
16:00,18/07/23,TEST,Тестування підсистеми,OK,	Проведено успішне тестування підсистеми
18:00,18/07/23,TEST,Тестування підсистеми,OK,	Проведено успішне тестування підсистеми
18:02,18/07/23,V-ON,Дозатори,Ввімкнення,	Увімкнено дозатори молока
18:02,18/07/23,P-00,Автомат промивання,Ввімкнення,	Увімкнено автомат промивання
18:03,18/07/23,V-01,Вакуум-насос,Запуск,	Запуск вакуум-насоса для прочищення молокопроводу за допомогою пижа
18:04,18/07/23,V-00,Вакуум-насос,Зупинка,	Зупинка вакуум-насоса після прочищення молокопроводу за допомогою пижа
18:04,18/07/23,V%00,Помилка сенсора вакууму,00,!!!	Після прочищення молокопроводу за допомогою пижа може виникнути повідомлення про помилку сенсора вакууму, у цьому випадку його не потрібно брати до уваги

18:05,18/07/23,V-01,Вакуум-насос,Запуск,	Запуск вакуум-насоса на початку другого добового доїння
21:18,18/07/23,V-00,Вакуум-насос,Зупинка,	Зупинка вакуум-насоса по закінченню другого добового доїння, тривалість доїння 3 год. 13 хв.
21:19,18/07/23,1,Дозатор,327,	Удій дояра на дозаторі №1 - 327 літрів
21:19,18/07/23,2,Дозатор,336,	Удій дояра на дозаторі №2 - 336 літрів
21:19,18/07/23,P-02,Промивання,Початок,	Початок виконання програми «Промивання»
21:21,18/07/23,V-01,Вакуум-насос,Запуск,	Запуск вакуум-насоса під час промивання

Продовження таблиці 3.3

21:34,18/07/23,PC55,Температура води,55,	Температура нагрітої води у баку автомата промивання 55 °С - відповідає нормі (не менше 35 °С)
22:09,18/07/23,V-00,Вакуум-насос,Зупинка,	Зупинка вакуум-насоса перед закінченням промивання
22:09,18/07/23,V%35,Величина вакууму (кПа),35,	Середнє значення вакууму протягом промивання відповідає нормі - 35 кПа (не менше 35 кПа)
22:09,18/07/23,P-05,Промивання,Закінчення,	Успішно закінчилося промивання, тривалість промивання 50 хв.

Спеціалізоване програмне забезпечення, робота якого описана вище, встановлене на інтегрованому в інтерфейсний блок одноплатному комп’ютері Raspberry Pi4, працює пд управлінням операційної системи Windows 7/8/10 та створено на мові програмування С++. Лістинг програмного забезпечення наведений у додатку А.

3.3. Розробка програмного забезпечення для мікроконтролера інтерфейсного блока

Спеціалізоване програмне забезпечення, під управлінням якого працює мікроконтролер ATMEGA32A інтерфейсного блоку, створено на мові програмування С++ з пакету програм IAR Embedded Workbench for AVR. Лістинг розробленого програмного забезпечення наведений у додатку Б.

Мікроконтролер інтерфейсного блоку забезпечує управління отриманням даних, які надходять від електронних дозаторів молока, блока управління автоматом промивання, блока контролю вакуум-насоса, фіксацію дати та часу подій, які відбуваються на доїльній установці, попередньої обробки отриманих даних, перетворення інтерфейсу та передавання цих даних через порт GPIO до одноплатного комп'ютера Raspberry Pi4, який входить до складу інтерфейсного блоку. Також мікроконтролер забезпечує управління інтегрованим енергонезалежним годинником реального часу (який живиться за допомогою спеціальної низькотемпературної батареї з номінальним значенням вихідної напруги 3.6 В), рідкокристалічним індикатором (16 символів у 2 строках) з підсвічуванням, забезпечує безперервне вимірювання вихідної напруги батареї, від якої живиться годинник реального часу, та у випадку необхідності її заміни відображає попередження на індикаторі, обробляє сигнали, які надходять від клавіатури. У режимі налаштування мікроконтролер забезпечує програмування наступних параметрів: мова інтерфейсу (українська або англійська); поточний час; поточний місяць; поточний рік; кількість електронних дозаторів молока (2, 4 або 8).

Розподіл функцій портів мікроконтролера ATMEGA32A наведений в таблиці 3.4.

3.4. Розподіл функцій портів мікроконтролера ATMEGA32A

Порт мікроконтролера ATMEGA32A	**Функція порта**
PC3 (25)	GPIO 1
PC2 (24)	GPIO 7
PD7 (21)	GPIO8
PC4(26)	GPIO 12
PC5 (27)	GPIO 16
PC6 (28)	GPIO 20
PC7(29)	GPIO 21
PD3/INT1 (17)	GPIO 23

PD5 (19)	GPIO 24
PD6 (20)	GPIO 25
PD1/TxD (15)	RxD
PA4 (36)	Кнопка «SET»
PA5 (35)	Кнопка «--->»
PA6 (34)	Кнопка «<---»
PA7 (33)	Кнопка «ENTER»
PC0/SCL (22)	RTC SCL
PC1/SDA (23)	RTC SDA
PB0 (1)	LCD DB0
PB1 (2)	LCD DB1

Продовження таблиці 3.4

PB2 (3)	LCD DB2
PB3 (4)	LCD DB3
PB4 (5)	LCD DB4
PB5 (6)	LCD DB5
PB6 (7)	LCD DB6
PB7 (8)	LCD DB7
PA1 (39)	LCD E
GND	LCD R/W
PA2 (38)	LCD RS
PA0/ADC0 (40)	Контроль стану батареї
PD2/INT0 (16)	Контроль напруги живлення
PD0/RxD (14)	Дані

Висновок

Розроблена інформаційна підсистема для доїльної установки з молокопроводом, яка забезпечує отримання та представлення інформації про параметри технологічного процесу виробництва молока та є інноваційним рішенням для доїльної установки з молокопроводом та індивідуальним обліком удою кожного дояра, у якому використовується сучасна технологія «Internet of things» (Інтернет речей) та реалізована концепція «Ферма в смартфоні».

Підсистема призначена для віддаленого контролю параметрів технологічного процесу виробництва коров'ячого молока на доїльних установках з молокопроводом (УДМ-100, УДМ-200 та аналогічні). Уся інформація про усі події з моменту інсталяції системи зберігається у спеціальній базі даних. Інформаційна підсистема дозволяє користувачу у будь-який момент часу (за допомогою смартфона, персонального комп'ютера або планшета) отримувати інформацію про параметри технологічного процесу виробництва коров'ячого молока, контролювати події, які відбуваються на доїльній установці, виявляти аварійні ситуації та факти порушень регламенту роботи персоналом ферми, здійснювати облік удою.

Інформаційна підсистема може складатися з 2, 4 або 8 електронних дозаторів молока, які призначені для управління виконавчою системою

дозатора, підрахунку кількості сформованих дозатором порцій, відображення та збереження інформації про кількість отриманих окремими доярами порцій молока, фіксації дати та часу зняття показань, а також для передавання даних про результат удою до інтерфейсного блоку; блока контролю вакуум-насоса, призначеного для фіксації дати та часу моментів ввімкнення та вимкнення вакуум-насоса, вимірювання миттєвого та середнього значення вакуумметричного тиску у вакуумпроводі доїльної установки та передавання цієї інформації до інтерфейсного блоку; блока управління автоматом промивання, призначеного для автоматичного управління процесами переддоїльного промивання, післядоїльного промивання та дезінфекції, контролю параметрів технологічного процесу промивання та передавання інформації про параметри технологічного процесу промивання до інтерфейсного блоку; інтерфейсного блоку призначеного для приймання даних про події, що відбуваються на доїльній установці, для фіксації дати та часу цих подій, обробки отриманих даних, представлення їх у необхідному вигляді та передавання цих даних користувачу через мережу Інтернет; спеціалізованого програмного забезпечення, яке встановлене на інтегрованому в інтерфейсний блок одноплатному промисловому комп’ютері, що здійснює обробку, представлення у необхідному вигляді та запис на карту пам’яті даних, які надходять від інших складових компонентів підсистеми, здійснює синхронізацію даних про події на доїльній установці, які знаходяться на смартфонах або комп’ютерах користувача, з даними, які знаходяться на карті пам’яті одноплатного комп’ютера Raspberry Pi4.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Кучерук, В. Ю. Огляд інформаційно-вимірювальних систем зоотехнічних параметрів тварин / В. Ю. Кучерук, Є. А. Паламарчук, П. І. Кулаков, Т. В. Гнесь // Вісник Вінницького політехнічного інституту. – 2015. – № 3(120). – С. 15–23.

2. Технології та обладнання для тваринництва ВАТ "Брацлав" / Рекламний проспект. – 2010. – 27 с.

3. Каталог продуктів та послуг ДеЛаваль / 2022. – 37 с.

4. DeMax і DemaTron. Універсал та асистент для доїльного залу. Рекламний проспект / GEA Group. – 2019. – 11 с.

5. Кулаков, П. І. Статистичні моделі тривалості машинного доїння / П. І. Кулаков, Т. В. Гнесь // Обчислювальний інтелект (результати, проблеми, перспективи). – Черкаси, 2015. – С. 301 – 302.

6. Кучерук, В. Ю. Статистичні моделі тривалості машинного доїння / В. Ю. Кучерук, Є. А. Паламарчук, П. І. Кулаков, Т. В. Гнесь // Восточно-европейский журнал передовых технологий. – 2014. – Т. 1, № 3 (67). – С. 4 – 7.

7. Кучерук, В. Ю. Статистична модель тривалості машинного доїння на стійловій доїльній установці / В. Ю. Кучерук, Є. А. Паламарчук, П. І. Кулаков, Т. В. Гнесь // Восточно-европейский журнал передовых технологий. – 2014. – № 2/4 (68). – С. 31–77.

8. Кулаков, П. І. Вимірювальний контроль тривалості роботи стійлової доїльної установки / П. І. Кулаков, О. Л. Бігдай // Матеріали п’ятої науково-практичної конференції «Методи та засоби неруйнівного контролю промислового обладнання». – Івано-Франківськ, 2015, С. 111