КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ТОРГОВЕЛЬНО-ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ

Кафедра комп'ютерних наук та інформаційних систем

ВИПУСКНИЙ КВАЛІФІКАЦІЙНИЙ ПРОЕКТ

на тему:

«Розробка програмно-апаратного забезпечення системи водопостачання на платформі «Arduino»»

Студента 4 курсу, 9 групи,
спеціальності
122 «Комп'ютерні науки»

_____
підпис студента

Кравченко Ростислав Русланович

Науковий керівник
кандидат технічних наук,
доцент

_____
підпис керівника

Демідов Павло Георгійович

Гарант освітньої програми
кандидат технічних наук, доцент

_____
підпис керівника

Демідов Павло Георгійович

Київ 2020

Київський національний торговельно-економічний університет

Факультет інформаційних технологій
Кафедра комп'ютерних наук та інформаційних систем
Спеціальність 122 «Комп'ютерні науки»

Затверджую
Зав. кафедри _____Пурський О.І.
«20» грудня 2019р.

Завдання
на випускний кваліфікаційний проект студенту

Кравченко Ростиславу Руслановичу
(прізвище, ім'я, по батькові)

1. Тема випускного кваліфікаційного проекту
«Розробка програмно-апаратного забезпечення системи водопостачання на платформі«Arduino»»
Затверджена наказом ректора від *«04» грудня 2019 р. № 4111*
2. Строк здачі студентом закінченої проекту _____ *року*
3. Цільова установка та вихідні дані до проекту
Мета проекту: розробити програмно-апаратне забезпечення системи водопостачання на платформі «Arduino».
Об'єкт дослідження: процеси організації водопостачання в тепличних господарствах
Предмет дослідження: математичне програмне та апаратне забезпечення системи водопостачання на платформі «Arduino»
4. Перелік графічного матеріалу: *Загальний вигляд пульта управління мікрокліматом, Загальний вигляд блоку керування мікрокліматом, Схема обслуговування теплиці, Загальний вигляд інтелектуального модуля вводу-виводу WISE-7118Z, Схема управління мікрокліматом теплиці, Загальний*

вигляд кліматичного комп'ютера, платформа Arduino, Датчик BME280, Покроковий енкодер, Датчик вологості грунту Arduino Модуль реле 4 канали для Arduino, Дисплей LCD 2004 HD44780 Arduino PIC STM, Сервопривод, Структурна схема меню системи управління мікрокліматом теплиці, Функціональна схема системи управління мікрокліматом теплиці

5. Консультанти по проекту із зазначенням розділів, за якими здійснюється консультування:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Демідов П.Г.	15.12.2019 р.	15.12.2019 р.
2	Демідов П.Г.	15.12.2019 р.	15.12.2019 р.
3	Демідов П.Г.	15.12.2019 р.	15.12.2019 р.

6. Зміст випускного кваліфікаційного проекту (перелік питань за кожним розділом)

ВСТУП

РОЗДІЛ 1. ТЕОРЕТИЧНІ АСПЕКТИ УПРАВЛІННЯ ВОДОПОСТАЧАННЯ В СУЧАСНИХ УМОВАХ

1.1. Поняття, основні елементи та класифікація систем водопостачання

1.2. Автоматизація процесів сучасних теплиць

1.3. Огляд сучасних мікроконтролерів клімату теплиць

РОЗДІЛ 2. ПРОЕКТУВАННЯ ПРОГРАМНО-АПАРАТНОЇ СИСТЕМИ ВОДОПОСТАЧАННЯ НА ПЛАТФОРМІ «ARDUINO»

2.1. Постановка задачі

2.2. Проектування системи управління мікрокліматом теплиці (СУМТ)

2.3. Проектування структурної схеми СУМТ

2.4. Проектування функціональної схеми СУМТ

РОЗДІЛ 3. РОЗРОБКА ПРОГРАМНО-АПАРАТНОЇ СИСТЕМИ ВОДОПОСТАЧАННЯ НА ПЛАТФОРМІ «ARDUINO»

3.1. Характеристика інтегрованого середовища розробки програмного

забезпечення для платформи «Arduino»

3.2. Розробка алгоритмів функціювання СУМТ

3.3. Розробка програмного забезпечення СУМТ

3.4. Технологія використання розробленої СУМТ

ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

7. Календарний план виконання проекту

№ Пор.	Назва етапів випускного кваліфікаційного проекту	Строк виконання етапів проекту	
		За планом	фактично
1	2	3	4
1	*Вибір теми випускного кваліфікаційного проекту*	*01.10.2019*	*01.10.2019*
2	*Розробка та затвердження завдання на випускний кваліфікаційний проект*	*15.12.2019*	*15.12.2019*
3	*Вступ*	*03.02.2020*	*03.02.2020*
4	*РОЗДІЛ 1. Теоретичні аспекти управління водопостачанням в сучасних умовах*	*28.02.2020*	*28.02.2020*
5	*РОЗДІЛ 2. Проектування програмно-апаратної системи водопостачання на платформі «Arduino»*	*06.04.2020*	*06.04.2020*
6	*РОЗДІЛ 3. Розробка програмно-апаратної системи водопостачання на платформі «Arduino»*	*12.05.2020*	*12.05.2020*
7	*Висновки*	*15.05.2020*	*15.05.2020*
8	*Здача випускного кваліфікаційного проекту на кафедру науковому керівнику*	*20.05.2020*	*20.05.2020*
9	*Попередній захист випускного кваліфікаційного проекту*	*03.06.2020*	*03.06.2020*
11	*Виправлення зауважень, зовнішнє рецензування випускного кваліфікаційного проекту*	*09.06.2020*	*09.06.2020*
12	*Представлення готової зшитої випускного кваліфікаційного проекту на кафедру*	*29.06.2020*	*29.06.2020*
13	*Публічний захист випускного кваліфікаційного проекту*	*За розкладом роботи ЕК*	

8. Дата видачі завдання «15» грудня 2019 р.

9. Керівник випускного кваліфікаційного проекту

Демідов П.Г.
(прізвище, ініціали, підпис)

10. Гарант освітньої програми Демідов П.Г.
(прізвище, ініціали, підпис)

11. Завдання прийняв до виконання студент-дипломник

Кравченко Р.Р.
(прізвище, ініціали, підпис)

12. Відгук керівника випускного кваліфікаційного проекту

____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____

Керівник випускного кваліфікаційного проекту

_________________ ___________ р.

(підпис, дата)

13. Висновок про випускнго кваліфікаційного проект

Випускний кваліфікаційний проект студента ____________________

(прізвище, ініціали)

може бути допущена до захисту в екзаменаційній комісії.

Гарант освітньої програми________________________Демідов П.Г.____

(підпис, прізвище, ініціали)

Завідувач кафедри _______________________________Пурський О.І.____

(підпис, прізвище, ініціали)

«_____»_________________2020 р.

Анотація

У випускному кваліфікаційному проекті здійснено комплексну розробку програмно-апаратного забезпечення системи водопостачання на платформі «Arduino», з метою спрощення управління системою керування теплицею та можливістю детального налаштування водопостачання. Теоретично обґрунтовано основні положення автоматизованих контролерів теплиць та аналогів, роботи і потенціалу мікрокомп'ютерів на прикладі «Arduino», та їх мови програмування «Processing». Розроблено програмну та апаратну частину контролера. Проектування системи управління мікрокліматом теплиці. Створено автоматизовану систему поливу, та виконана її апаратна реалізація.

Ключові слова: автоматизована система, мікрокомп'ютер, програма, «Arduino», системи управління мікрокліматом теплиці, водопостачання, теплиця, система керування.

Annotation

In the final qualification work, a comprehensive development of software and hardware of the water supply system on the platform "Arduino" was carried out, in order to simplify the management of the greenhouse management system and the possibility of detailed configuration of the water supply. The main provisions of automated greenhouse controllers and analogues, operation and potential of microcomputers on the example of "Arduino", and their programming language "Processing" are theoretically substantiated. The software and hardware of the controller are developed. Design of a greenhouse microclimate control system. An automated irrigation system has been created, and its hardware implementation has been completed.

Keywords: automated system, microcomputer, program, "Arduino", greenhouse microclimate control systems, water supply, greenhouse, control system.

ЗМІСТ

ВСТУП

Теплиці є важливою частиною аграрного сектору економіки України, завдяки яким відбувається постійне та внесезоне зрошування рослин. За підрахунками міністерства аграрної політики України 2019 році, теплична продукція забезпечувала 20% внутрішнього ринку овочів в Україні, а в пік сезону овочевих культур досягало відмітки 50%. Процес зрошування рослин досить трудомісткий, та потребує постійного контролю за рослинами і задіяння великої кількості людської сили до даного процесу. Саме тому сьогодні стрімко розвиваються та створюються різні системи контролю мікроклімату теплиці, та водопостачання до рослин, що зумовило **актуальність** обраної теми дослідження.

Мета завдання: розробити програмно-апаратне забезпечення системи водопостачання на платформі «Arduino».

Об'єкт дослідження: процеси організації водопостачання в тепличних господарствах.

Предмет дослідження: математичне програмне та апаратне забезпечення системи водопостачання на платформі «Arduino».

Методи дослідження: теоретичною основою дослідження є загальнонауковий аналітичний метод

Практичне значення. Полягає в тому, что теоретичні та практичні положення і результати, програмно та апаратно реалізовані в системі управління мікрокліматом теплиці.

РОЗДІЛ 1.
ТЕОРЕТИЧНІ АСПЕКТИ УПРАВЛІННЯ ВОДОПОСТАЧАННЯМ В СУЧАСНИХ УМОВАХ

1.1 .Поняття, основні елементи та класифікація систем водопостачання

Сьогодні існує багато методів вирощування рослин, але всі вони залежать від водопостачання. Саме тому водопостачання є невід'ємною частиною рослинництва. Перший метод це традиційне вирощування рослин у відкритому ґрунті, такий метод передбачає вирощування рослин у землі, застосовуючи наявні кліматичні умови і підібрані відповідно до них агротехнології. До агротехнологій входить: використання сезонної ручної та механізованої праці, догляд за посівами за допомогою, переважно, хімічних засобів захисту рослин (гербіциди, інсектициди, протруйники, ретарданти).

Наступним методом отримання рослинної іжі (особливо зелені, огірків, помідорів) є так званий метод гідропоніки визначення якої дає Вільям Тексье «метод вирощування рослин без грунту воді, що містить розчинені живильні речовини», метод передбачає заміну грунту на спеціальний субстрат (кокосові мати), використання капельного поливу до кожної рослинки, штучного забезпечення поживними мікроелементами, а також відповідним тепловим та світловим режимом на період вегетації.[1] Як модифікація цього методу є використання плаваючих по басейну з поживним розчином лотків з рослинами, коріння яких знаходяться у розчині, а зелена маса на поверхні. Усі сучасні теплиці будуються переважно для гідропонного вирощування рослинної їжі. Вирощування залежить не від клітатичних умов, а фактично від наявності усіх складових розвитку рослини. В останні роки ведуться успішні наукові пошуки в напрямку заміни хімічних розчинів для гідропонних рослин на органічні. Так, дослідження показують, що використання води в якій живуть риби або інші жителі водяної фауни, цей метод є дуже перспективним у цього напрямку. Отже розглянемо такий метод як аквапоніка, яка в свою чергу спирається на природні відносини між

водними тваринами і рослинами, що вельми сприятливо для збереження довкілля. Система являє собою дві ємності розташовані один над одним. У нижній ємності мешкають риби, а у верхній ємності — ростуть рослини [2-3,6]. Вода до рослин подається від риб за допомогою заглибного насоса (помпи). Продукти життєдіяльності риб містять поживні речовини для рослин, але є токсичними для самих риб. Рослини поглинають ці речовини, що забезпечує їм необхідне харчування, і тим самим, очищають воду для риб (при цьому рослини та риби ростуть більш активно). Очищена вода повертається назад до риб, потім цикл повторюється. Ґрунтом для рослин у даному випадку використовується керамзит або гравій [4-6]. Оскільки рослини і керамзит виконують роль біологічного фільтра, у зв'язку з цим можна збільшити кількість риб в ємності без ризику їх захворювання або отруєння продуктами життєдіяльності. Вода додається лише в міру поглинання рослинами, випаровування в повітря або видалення біомаси з системи. Відходи життєдіяльності риб є натуральним добривом для овочів або квітів. Значно підвищується врожайність і прискорюється дозрівання плодів. У помідорах, вирощених на аквапоніці, вміст нітратів зазвичай менше в п'ять — десять разів, ніж у найкращих ґрунтових, а смак і аромат нічим не поступається. Сьогодні велика кількість країн вже використовують аквапоніку в промислових масштабах.

Четвертий метод, що стрімко набирає оберти в світі рослинництва це аеропоніка один із напрямків гідропоніки, метод вирощування рослин без ґрунту та субстрату у вологому повітряному середовищі завдяки періодичному обприскуванню коренів поживними розчинами. Метод цієї повітряної культури рослин 1910 вперше було розроблено В. Арциховським, який описав власний метод фізіологічних досліджень кореневих систем за допомогою розбризкування різних речовин у повітрі[7-9]. В. Картером описано технологію вирощування рослин в парах води, а 1957 Ф. В. Вент винайшов спосіб вирощування рослин за допомогою повітря, який він назвав «аеропоніка» [10]. Суть методу аеропоніки полягає в тому, що коріння рослин,

що знаходяться в повітрі, періодично (кожні 10–20 хв.) зрошують дрібними краплями живильного розчину у формі, подібній до тумана, за допомогою спеціальних форсунок. Переваги аеропоніки: витрата мінімальної кількості живильного розчину, зменшення маси установок для вирощування рослин. Аеропоніка застосовується в теплицях, оранжереях, космічних кораблях тощо. У виноградарстві метод перебуває на стадії науково-дослідних розробок. З 2006 аеропоніка використовується у світовому масштабі; найчастіше в Японії. У виробничих умовах в Україні аеропоніка ще не знайшла широкого промислового застосування [7-11].

Отже розглянувши існуючи сучасні методи водопостачання для рослин, ми обираємо метод гідропоніки, для свого мікроконтролера, оскільки він має переваги в якості та швидкості вирощування рослин над звичайним поливом ґрунту, та не має потреб в складній побудові резервуарів для рослин, як наприклад в аеропоніці та аквапоніці.

1.2. Автоматизація процесів сучасних теплиць.

«Розумна» теплиця – власна автоматизована система, яка відслідковує різні показники датчиків і на основі аналізу цієї інформації керує умовами зрощення та розвитку рослин, котрі потребують певного діапазону температур, вологості ґрунту і повітря, світла та складу повітря.

Основні фактори, для контролю «розумною» теплицею:

- тепло, для попередження замерзання чи перегріву рослин. У кожної рослини є свій сприятливий діапазон температур, для вдалого зрощування, при якому рослина росте та плодоносить, так щоб колір та розмір плоду, був високого ґатунку;
- світло, щоб за допомогою правильної регуляції забезпечити кращий ріст рослини та анулювати недостачу світла, а в жаркі дні корегувати вологість рослини, або зменшити потік світлових променів;
- вода, оскільки у нашої теплиці є дах, котрий не пропускає осади, то життєвонеобхідний полив рослин, а також необхідний контроль об’єму подачі води, час подачі;

- повітря – це те без чого рослини просто не можуть існувати, також в залежності від періоду доби рослини потребують певну кількість кисню, або вуглекислого газу, або зменшення чи відсутність повітряного потоку.

Коли власники теплиць починають використовувати допоміжні системи контролю виникає питання автоматизації процесів теплиці, на яке спунукає бажання контролювання всіх процесів одночасно. Отже розглянемо можливі види автоматизації систем:

- зрошення – полив води відбувається за регулярним графіком, або коли вологість грунту нижче потрібної;
- кількість CO2 – система контролю кількості CO2 в повітрі;
- вентиляція – за допомаогою відкриття, або закриття фрамуг, або контролю вентиляторів та інших вентиляційних приладів;
- освітлення – можливість керування освітлювальними приладами, або затемнення вікон суцільним щільним полотном ;
- опалення – за допомогою різних опалювальних систем, котрі базуються на воді, парі, електриці, та інших джерелах тепла;
- дозування добрив – корисні речовини автоматично розподіляютьсяпо всій системі зрошення;
- знищення шкідників –системі які автоматично розпиляють піретрин, перетроїди;

Ми будемо включати всі системи автоматизації, до нашого контролера управління мікрокліматом теплиці на програмному рівні, також більшість з них буде реалізована і в апаратному екземплярі.

1.3. Огляд сучасних мікроконтролерів клімату теплиць

Управління технологічним процесом - складне завдання, від правильного вирішення якого залежить кінцевий результат - стабільні врожаї

та висока якість. І на перший план виходять системи, що дозволяють комплексно автоматизувати це складне виробництво.

Сучасна теплиця, як об'єкт управління, характеризується незадовільною динамікою і нестабільністю параметрів, що випливають з особливостей технології виробництва. У той же час агротехнічні норми потребують високої точності стабілізації температури (+/-1 градус), своєчасної її зміни в залежності від рівня фотосинтетично активного опромінення, фази розвитку рослин і часу доби. Всі ці обставини потребують високих вимог до функціонування та технічного вдосконалення обладнання апаратного забезпечення[13].

Автоматизація систем управління мікрокліматом дозволяє: економити 15-25% тепла, покращує умови праці персоналу, підвищувати загальну культуру виробництва, забезпечити чіткі межі регулювання мікрокліматичних умов теплиці, точно забезпечити подачу поживних речовин рослинам, тим самим збільшуючи їх врожайність.

З метою забезпечення високої продуктивності тепличних господарств необхідно підтримувати цілу низку параметрів на певному рівні або у певних межах. До основних параметрів відносять: обігрів повітря в середині теплиці, обігрів ґрунту, концентрація вмісту вуглекислого газу в повітрі, циркуляція повітря по теплиці, вентиляція, вологість, освітленість.

Ринок обладнання пропонує широкий вибір фірм і приладів які займаються автоматизацією кліматичних показників у теплиці і автоматизацією цього процесу. Серед них такі як: ТОВ «ФИТО», компанія «ICP DAS», компанія «ОВЕН», «EKF» і т.д.

Приведемо коротку характеристику різного обладнання для забезпечення заданих параметрів у теплиці.

EKF - одна з провідних компаній електротехнічної галузі, що працюють в середньому ціновому сегменті, що займається випуском повного асортименту високоякісної низьковольтної продукції. Апаратний доробок для регулювання мікроклімату (зображений на рисунку 1.1)[13].

Рис. 1.1. Загальний вигляд пульта управління мікрокліматом

Блок управління мікрокліматом призначений для контролю основних параметрів теплиці, оранжереї, гроубокса і т.п. Може контролювати освітлення, полив, температуру, вентиляцію і т.д.

Пульт управління мікрокліматом в автоматичному режимі підтримує задані температурні і вентиляційні режими:

- керує вентиляторами або нагрівачами;
- реалізує будь-які режими освітлення;
- задає програми для поливальних і повітряних насосів.

Основні переваги використання даної системи:

- гнучкість системи;
- простота установки;
- простота настройки та експлуатації;
- вологозахищений корпус;
- можливість використання резервного живлення для збереження налаштувань системи у випадку відключення – електроживлення.

Як бачимо пульт управління має низку переваг серед яких: можливість підбору блоку за необхідними параметрами.

EKF пропонує також 8-ми канальний блок керування мікрокліматом в теплиці на базі мікроконтролера DS1820, загальний вигляд (зображено на рисунку 1.2)

Рис. 1.2. Загальний вигляд блоку керування мікрокліматом

Пристрій дозволяє регулювати температуру і вологість повітря та ґрунту в теплиці, підігрівати воду, вмикати і вимикати насоси гідропонних установок, проводити полив та вентиляцію в теплиці. Управління навантаженнями відбувається за допомогою таймера - включення навантаження в заданому інтервалі часу, а також контролювати мікроклімат через установку температури (режим термостата).

На рисунку 1.3 можна наочно побачити, як функціонує блок керування мікрокліматом в цілому. До блоку підключається насос, який закачує воду у бак з резервуару для поливу. У баці вода нагрівається до заданого значення і по мірі необхідності витрачається для поливу. Можливість обігріву реалізована за допомогою електрообігрівачів, які також можливо підключити для забезпечення температурного режиму повітря і ґрунту.

Рис. 1.3. Схема обслуговування теплиці

Для зручності підключення датчиків температури, вологості, силових виходів, а також інтерфейсу RS232 (COM-порт ПК) в блоці автоматичного управління передбачені відповідні клемні роз'єми. Живлення подається через спеціальний роз'єм від адаптера напругою 9-12 В. Технічна характеристика блоку керування мікрокліматом[16-18]:

- Кількість каналів керування: 8.
- Режими управління по таймеру: включення навантаження в певному проміжку часу; управління навантаженнями по певних днях тижня, дням у місяці, або за обраними місяцях.
- Режими управління по температурі (термостатування): управління як охолоджувачем; управління як нагрівачем.
- Вбудований будильник зі звуком і світлом (підсвічування дисплея): входи для підключення датчиків.
- Цифровий вхід для підключення датчиків: до 32 датчиків.
- Кількість аналогових входів: 2.
- Енергонезалежні годинник реального часу (повний календар з урахуванням високосних років) до 2099
- Збереження усіх налаштувань в енергонезалежній пам'яті; подовження правильної роботи програми в разі тимчасового відключення від мережі.

- Виходи: вісім оптично ізольованих сімісторних двоамперних каскадів з можливістю підключення силових сімісторов для управління більш потужними навантаженнями (більше 2 А).
- Віддалене управління термостатом через COM-порт комп'ютера за допомогою спеціально розробленого програмного забезпечення.

Кількість параметрів, які контролює цей модуль дозволяє використовувати його у невеликих теплиць. У своєму роді його можна назвати одним із кращих. Помірна вартість і простота в експлуатації є позитивною стороною. До недоліків варто віднести складність у програмуванні. Програмується модуль на мовах як високого так і низького рівня, проте необхідно враховувати синтаксис мови.

Для полегшення процесу автоматизації компанія «ICP DAS» пропонує використовувати інтелектуальний модуль вводу-виводу WISE-7118Z. Загальний вигляд інтелектуального модуля наведено на рисунку 1.4.

Рис. 1.4. Загальний вигляд інтелектуального модуля вводу-виводу WISE-7118Z

Даний модуль має 10 каналів аналогового вводу та 6 дискретного виводу. Користувач може конфігурувати канали на різні діапазони струму і напруги, а також різні типи термопар для вимірювання параметрів

мікроклімату, наприклад, температуру або вологість. За допомогою дискретних виходів можна керувати обігрівом, кватиркою вентиляцією, поливом, системою випарного охолодження, освітленням і т.д. Інтелектуальний модуль WISE-7118Z буде «невпинно» контролювати стан мікроклімату теплиці, а в залежності від зміни параметрів повітря, ґрунту, освітленості виробляти відповідне управління.

Крім того інтелектуальний модуль WISE підтримує протокол передачі даних Modbus TCP Slave. Це дозволяє об'єднати такі системи в єдиний диспетчерський центр, де використовуючи SCADA систему, користувач може управляти всім процесом і отримувати актуальну інформацію про стан мікроклімату в кожній теплиці. На рисунку 1.5 зображено схему підключення інтелектуального модуля до обладнання теплиці[14].

Рис. 1.5. Схема управління мікрокліматом теплиці

Переваги використання WISE:

- для створення конфігурації управління в модулі WISE використовується IF-THENELSE логіка. Користувач може

використовувати 36 правил IF-THEN-ELSE. Створивши правила і завантаживши їх у модуль, вони будуть виконуватися з настанням відповідної події. Всі необхідні дії конфігурації виробляються в Webбраузері за кілька клацань миші;

- підтримка Modbus TCP для інтеграції в SCADA систему;
- WISE-7118Z підтримує технологію живлення POE, що звільняє від необхідності прокладання кабелів ліній для живлення.

Фірма «ФИТО» пропонує системи управління мікрокліматом теплиць серії FC. Зокрема загальний вигляд кліматичного комп'ютера цієї фірми відображено на рисунку 1.6.[15].

Рис. 1.6. Загальний вигляд кліматичного комп'ютера

Це оптимальне рішення для будь-якого виду теплиць різних за розміром як плівкових, так і скляних. Для управління основними параметрами теплиці обладнуються, так званими, виконавчими системами: системою обігрівання, вентиляції, освітленості, системою «підгодівлі» CO2. Суворе дотримання основних параметрів мікроклімату - це запорука високої врожайності і стійкості рослин до захворювання. Але ніщо не стоїть на місці, у тому числі і технологія вирощування, і на сьогоднішній день передові агрономи-технологи приділяють велику увагу розширеному набору показників клімату, який

включає в себе температуру листа, вологість листа, розподіл температури повітря по вертикальному зрізу теплиці, швидкість руху повітря . Компанією «ФИТО» розроблено клімат-комп'ютери, які підтримують не тільки основні показники мікроклімату, але і дозволяють контролювати згадуваний розширений набір показників. Архітектура клімат-комп'ютерів дозволяє повністю в автоматичному режимі керувати типами виконавчих систем теплиці із суворим дотриманням заданого агрономом режиму. Зростаючі ціни на енергоносії зобов'язують не тільки дбати про підтримання клімату, а й про ефективне витрачання ресурсів, будь то включення системи освітлення або опалення, подача CO2 або активне відкриття фрамуг. У зв'язку з цим функціональні можливості систем управління дозволяють створювати «стратегію управління», де агроном може в залежності від фази росту рослин та /(або) економічної доцільності вибрати пріоритетне завдання економії енерговитрат або максимального дотримання технології.

Принцип роботи кліматичного комп’ютера. Ядром системи є промисловий контролер управління, розроблений фахівцями фірми «Фито» спеціально для теплиць. Завдяки сучасній елементній базі з американських і японських комплектуючих контролери мають високий показник безперебійної і надійної роботи. Крім контролера, система управління мікрокліматом включає в себе підсистему вимірювальних датчиків, встановлених всередині теплиці.

При необхідності система може бути автоматично інтегрована з котельні. Для цього є спеціальний модуль, який по інтерфейсу FIDUFACE передає дані в котельню для управління виробленням тепла, CO2 і електроенергії. Стежити за процесом мікроклімату, а також вносити завдання в зручній формі можна з ПК. Також є доступною функція віддаленого адміністрування системи через Інтернет.

Функціональні можливості:

- вимірювання параметрів клімату в декількох зонах;

- повний автоматичний контроль систем опалення, вентиляції зашторювання, CO2, освячування;
- створення оптимальної "стратегії управління";
- інтеграція в систему управління котельні (FIDUFACE) ;
- зручний інтерфейс;
- функція економії енергетичних ресурсів;
- віддалений моніторинг і аналіз з ПК.

Таким чином проаналізувавши всі вище перераховані засоби для підтримання мікроклімату в теплиці, можна зазначити, що вибір на сьогоднішній день є досить різноманітним і користувач може вільно вибирати параметри, які повинні його задовольняти. Всі засоби мають у дуже великий потенціал.

Пульт управління мікрокліматом компанії «EKF» варто застосувати у невеликих теплиць. Хоча пульт може виконувати такі дії як: контроль освітленості, полив, регулювання температури у теплиці і забезпечення її вентиляцію, але кількість параметрів можна сказати є середньою в порівнянні з його аналогами.

Ще одне творіння цієї фірми - це блок керування мікрокліматом на мікроконтролері DS1820. Серед переваг, як і у попередника, можна віднести малі розміри, простота в експлуатації, регулювання різними параметрами в залежності від підключених датчиків. Кількість підключених датчиків сягає 32, що при правильному їх підборі можливо повність автоматизувати процес регулювання мікроклімату на одному модулі. Вісім виходів дозволяють підключити пристрої для управління більш потужними навантаженнями. Можливість підключення до комп’ютера дає змогу користувачу відразу, в реальному часі, відслідковувати всі параметри і швидко реагувати у разі необхідності. Блок можна використовувати і для малих, і для великих теплиць.

Найбільш досконалим у своєму роді, з нашої точки зору, є інтелектуальний модуль WISE7118Z. Не зважаючи на його малі розміри, він

може контролювати достатню кількість параметрів, але з додатковими можливостями. А саме: постійний контроль за мікрокліматом у теплиці, можливість використання режиму передачі Modbus TCP, що дозволяє збирати дані про стан у теплиці і відображати це на моніторі користувача. Це досить корисно, коли контроль ведеться за кількома теплицями одночасно. Хоча кількість датчиків, які можна під'єднати до контролера тільки 10, але з використанням комутаторів, таких як RSM-408 і NS-205PSE можна збільшити кількість модулів підключення. Дані подаються на сервер і обробляються у SCADA системі. Передбачено також логічне управління подіями, що додає мікрокнтролеру додаткові переваги.

Останнім серед розглянутих прикладів був кліматичний комп'ютер фірми «ФИТО». Не зважаючи на його великі розміри, він є дуже потужним у плані параметрів, які контролюються. Він добре підходить як для великих так і для малих теплиць, контролює основні параметри, має додаткові можливості контролю температури листка, вологості листка, контролює розподіл температури по теплиці і т.д. Кліматичний комп'ютер є економічно вигідним, а його архітектура зводить до мінімуму процент втручання людини у роботу системи.

З нашої точки зору, закордонні виробники випускають кращу і більш надійну продукцію, а тому, як наслідок, її ціна і якісь є вищою ніж інших вітчизняних виробників.

Якщо кошторис не обмежується, то більш раціональним буде застосування кліматичного комп'ютера фірми «ФИТО», або пристрою на інтелектуальному модулі WISE-7118Z. Ці два представника призначені як для великих, так і для середніх теплиць, а тому в повній мірі задовольнять всі основні вимоги до автоматизації мікроклімату в теплиці.

В свою чергу пульт управління мікрокліматом компанії «EKF» і блок керування мікрокліматом на мікроконтролері DS1820 краще застосовувати для малих теплиць і теплиць середнього розміру. Цьому також сприяє помірна ціна на дані вироби.

РОЗДІЛ 2.
ПРОЕКТУВАННЯ ПРОГРАМНО-АПАРАТНОЇ СИСТЕМИ ВОДОПОСТАЧАННЯ НА ПЛАТФОРМІ «ARDUINO»

2.1. Постановка задачі

Отже розглянувши класифікації та основні елементи водопостачання в сучасних теплицях, ми прийняли рішення, що у нас зрощування рослин, буде відбуватися за методом гідропоніки, та зрощування у ґрунті. Вичленивши всі основні процеси необхідні для автоматизації наш список складає такі процеси:

- зрошення;
- кількість CO2;
- вентиляція;
- освітлення;
- опалення;

Для зрошення, необхідно створити можливість періодичного поливу, а саме за таймером реального часу, з можливістю керувати часом та кількістю подачі води. Також будуть встановленні датчики вологості, для контролю вологи у ґрунті, або контролю рівня води у резервуарі з рослинами.

Для кількості CO2, необхідно встановити датчики повітря, які будуть записувати данні в пам'ять приладу

Для вентиляція, необхідно розробити можливість встановлення серво привіда, який буде здійснювати відкриття та закриття вікон у теплиці.

Для освітлення, необхідний контроль над елементом освітлювання, та можливість його включення і виключення за таймером.

Для опалення, буде встановлено контролер включення та виключення нагріваючого пристрою.

Необхідно розробити зручний вивід інформації, та створити меню навігації по приладу. Зі зручним механічним управлінням.

2.2. Проектування системи управління мікрокліматом теплиці (СУМТ)

Враховуючи всі вище вказані задачі, нам необхідно визначити які ми використовуємо модулі та елементи для створення системи управління мікрокліматом теплиці.

Для виконання дипломного проекту, я буду використовувати плату Arduino на платформі Nano (рис. 2.1). Плата Arduino складається з мікроконтролера Atmel AVR (ATmega328P), а також елементів обв'язки для програмування та інтеграції з іншими схемами. Arduino Nano розроблено і випускається фірмою Gravitech.

Рис.2.1. Платформа Arduino

Живлення Arduino Nano може бути реалізоване через кабель Mini-B USB, від зовнішнього джерела живлення з нестабілізованою напругою 6-20В (через вихід 30) або зі стабілізованою напругою 5В (через вихід 27). Пристрій автоматично вибирає джерело живлення з найбільшою напругою.

Напруга на мікросхему FTDI FT232RL подається тільки в разі живлення Arduino Nano через USB. Тому при живлені пристрою від інших зовнішніх

джерел (не USB), вихід 3.3В (формований мікросхемою FTDI) буде неактивний.

Обсяг пам'яті програм мікроконтролера ATmega328 становить 32 КБ (з яких 2 КБ відведені під завантажувач). ATmega328 має 2 КБ оперативної пам'яті SRAM і 1 КБ EEPROM (бібліотека яка використовується для обробки даних платформую).

З використанням функцій pinMode (), digitalWrite () і digitalRead () кожен з 14 цифрових виходів Arduino Nano може працювати в якості входу або виходу. Робоча напруга висновків - 5В. Максимальний струм, який може віддавати або споживати один висновок, становить 40 мА. Всі висновки пов'язані з внутрішніми підтягуються резисторами (за замовчуванням відключеними) номіналом 20-50 кОм.

Використано датчик абсолютного тиску BME280 (рис.2.2). Виготовлений відомим, німецьким виробником Bosch. BME280 може вимірювати не тільки тиск, але температуру і вологість повітря. Передає дані по I2C або SPI. Працює по I2C. На верхній стороні плати розташований сам датчик BME280 (металевий корпус з отвором), а зі зворотного боку знаходяться ldo стабілізатор на 3.3В і мікросхема перетворення логічних рівнів, для використання GY-BME280 спільно з контролерами, в тому числі і Ардуіно, що живиться від 5В.

Для роботи датчика з Ардуіно потрібно завантажити бібліотеку BME280I2C.h. Бібліотека містить безліч прикладів скетчів

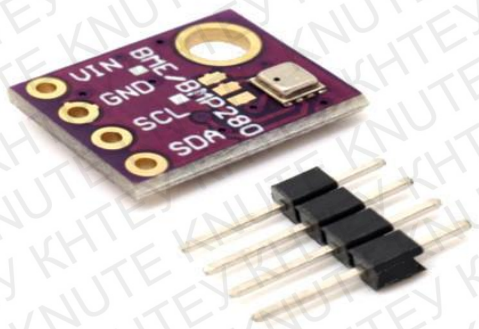

Рис.2.2. Датчик BME280

Для майбутньої навігації в нашому меню використаємо імпульсний енкодер (Рис2.3). Імпульсний (покроковий) енкодер відноситься до типа енкодерів, які призначені для вказівки напряму руху, або кутового переміщення зовнішнього механізму. Покроковий (також іменований інкрементний або інкрементальний) енкодер формує імпульси, кількість яких відповідає повороту вала на певний кут. Цей тип енкодерів, на відміну від абсолютних, не формує код положення вала, коли вал знаходиться в спокої.

Покроковий енкодер пов'язаний з рахунковим пристроєм, це необхідно для підрахунку імпульсів і перетворення їх в міру переміщення вала.

Рис.2.3. Покроковий енкодер

Наша система має потребу роботи в реальному часі та необхідно встановлювати таймер за реальним поточним часом, для цього використаємо: модуль годиника реального часу DS1302 RTC (Real Time Clock). Даний модуль спроектований на основі мікросхеми DS1302, яка володіє 31 байтом статичного ОЗУ. Чіп DS1302 є поліпшеним аналогом DS1202. Основними відмінностями мікросхем є наявність додаткових 7 байт ОЗУ і два виходи для підключення живлення на платі з мікросхемою DS1302. Модуль здатний відображати інформацію про реальну поточну дату, та визначати кількість діб в місяці, включаючи дні для високосного рока. Модуль підтримує як 24-годинний, так і 12-годинний формати. Інформація може передаватися або

прийматися по 1 байту або в пакеті з 31 байта. Для роботи даного датчика використана бібліотека iarduino_RTC для RTC DS1302, DS1307, DS3231.

Системи управління мікрокліматом теплиці, може виконувати полив за методом гідропоніки, або за звичайним традиційним методом вирощування в грунті, при традиційному вирощувані необхідно відслідковувати рівень вологості грунту, і в цьому нам допоможе: датчик вологості грунту Arduino (рис.2.4) призначений для визначення вологості грунту, в яку він занурений. Він дозволяє дізнатися про недостатнє або надмірне поливання рослин. Підключення даного модуля до контролера дозволяє автоматизувати процес поливу ваших рослин, городу або плантації. Модуль складається з двох частин: контактного щупа YL-69 і датчика YL-38, в комплекті йдуть дроти для підключення. Між двома електродами щупа YL-69 створюється невелика напруга. Якщо ґрунт сухий, опір великий і струм буде менше. Якщо земля волога - опір менше, струм - трохи більше. За підсумковим аналоговому сигналу можна судити про ступінь вологості. Щуп YL-69 з'єднаний з датчиком YL-38 по двох проводах. Крім контактів з'єднання з щупом, датчик YL-38 має чотири контакти для підключення до контролера. Для роботи з даним контролером було використано бібліотеку Software Serial library.

Рис.2.4. Датчик вологості грунту Arduino

Arduino Датчик рівня води призначений для визначення рівня води в різних ємностях, де недоступний візуальний контроль, з метою контролю рівня води в ємності. Конструкції датчиків рівня води можуть бути різними - поплавкові, занурені, врізні. Даний датчик води - занурений. Чим більше занурення датчика в воду, тим менше опір між двома сусідніми проводами. Датчик має три контакти для підключення до контролера. Для роботи з даним датчиком було використано бібліотеку Water sensor.

Оскільки робота нашої помпи подачі води, не є постійною, та для її роботи потрібно 12V, ми використовуємо реле(рис2.5), яке буде подавати струм на помпу. Модуль реле з комутуючих реле 5В. Управляється безпосередньо з виходів мікроконтролера. Максимальний струм навантаження 10А при напрузі 250В. 5В TTL керуючий вхід, ток для спрацьовування реле: 15-20мА. Для роботи з даним приладом було використано бібліотеку iarduino_I2C_Relay.

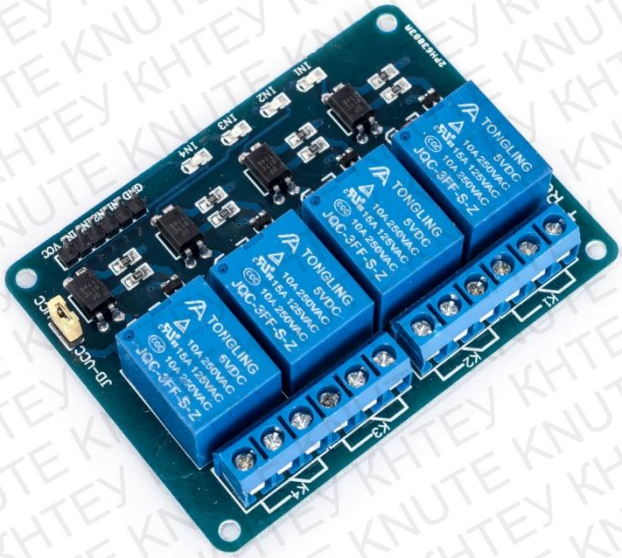

Рис.2.5. Модуль реле 4 канали для Arduino

Нашим путівником по керуваню сисстеми контролю мікрокліматом теплиці виступить: дисплей LCD 2004 HD44780 Arduino PIC STM (рис.2.6). Символьний рідкокристалічний дисплей з підсвічуванням для підключення до мікроконтролеру

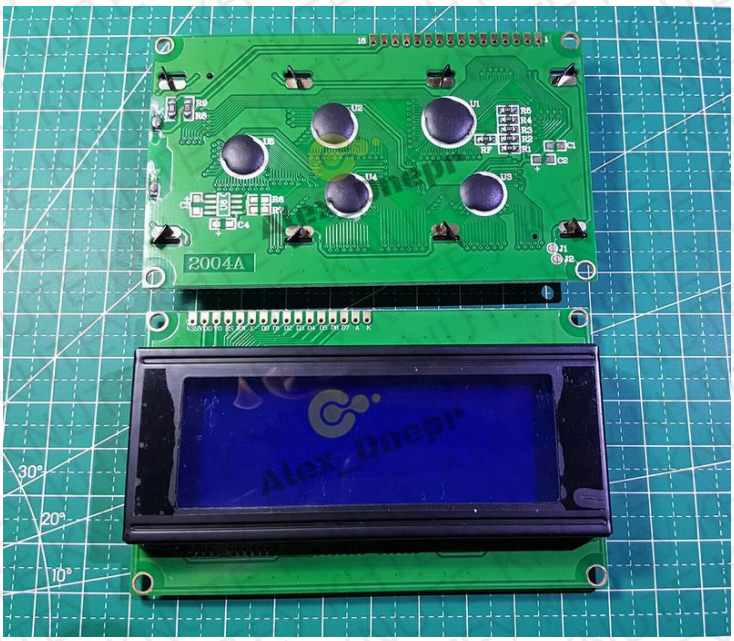

Рис.2.6. Дисплей LCD 2004 HD44780 Arduino PIC STM

Для здійснення механічно руху певних елементів теплиці, таких як: відкриття та закриття затінючої тканини, відкриття та закриття вікон та інших речей. Використовуємо сервопривод(рис.2.7) - механізм, який має в своєму пристрої спеціальний датчик, за яким відслідковуються певні значення, блок управління, двигун. Завданням пристрою є контроль і підтримання параметрів під час роботи, в залежності від сигналу, що передається в окремий момент часу. Для роботи з даним приладом було використано бібліотеку Servo.

Рис.2.7. Сервопривод

2.3. Проектування структурної схеми СУМТ

Для керування всією системою управління мікрокліматом теплиці, побудуємо структурну схему меню нашої системи.

Оскільки ми визначили елементи нашої системи, а саме: датчики температури, датчики вологості грунту, кількості води в ємності з водою для поливу, помпи для води, які підключаються через реле. Розглянемо структурну схему(рис.2.8)

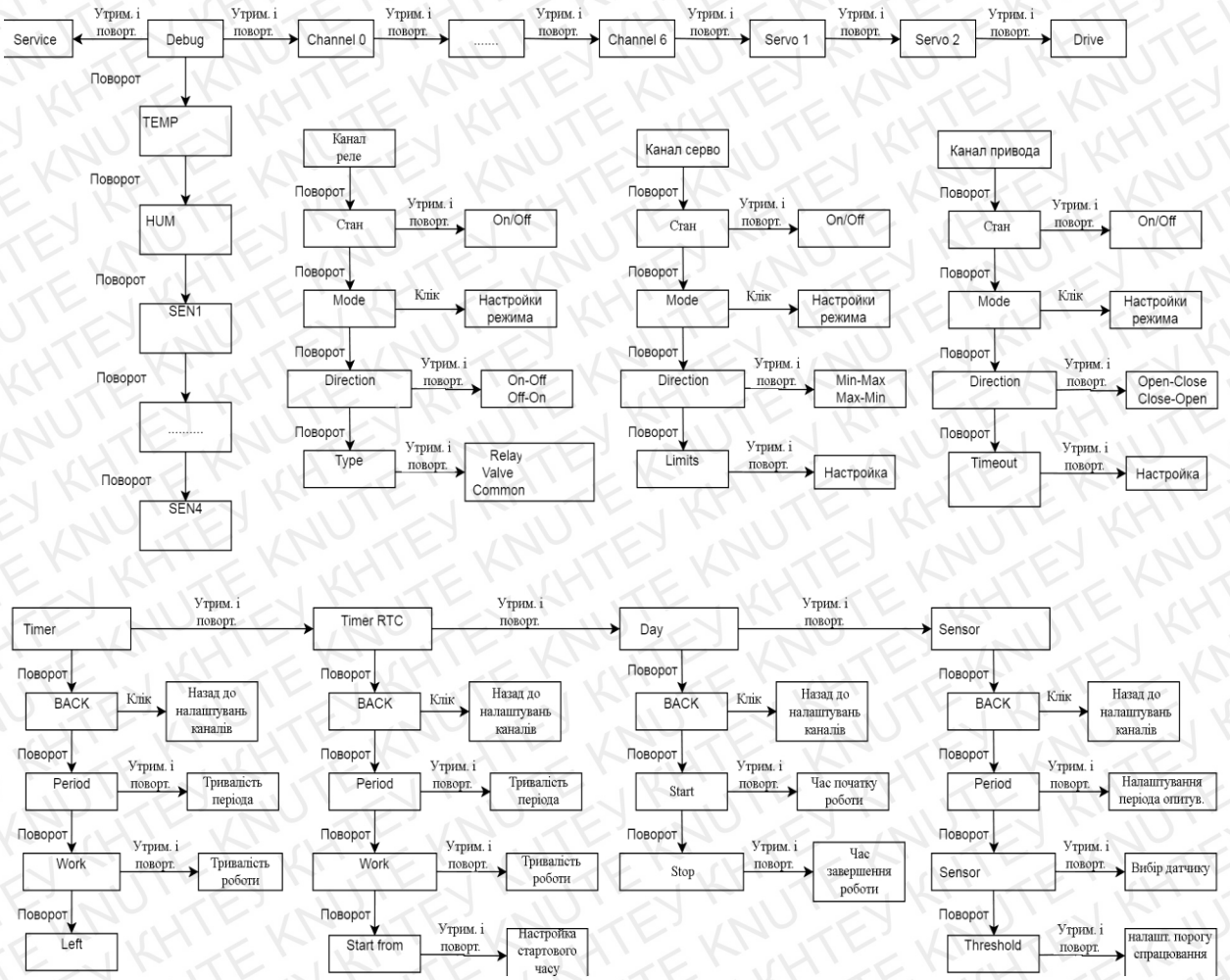

Рис.2.8. Структурна схема меню системи управління мікрокліматом теплиці

Починаємо ми з відділа Debug, в яке ми входимо натиском на енкодер, з якого ми можемо відслідкувати всі наші данні про стан мікроклімату, температуру та показнки наших датчиків, переміщеня відбувається завдяки прокручувані ендокера. Для виходу з меню ми здійснюємо утримування. Відділ Debug показує нам стан всіх компонентів системи, позиції реле, сервопривода та показники всіх датчиків поточний час, та аптайм – час роботи прилада з моменту включення, данна функція створенна для контролю відключення приладу.

Наступний ряд меню, че наші канали підключення до реле, а вже з них до водяних помп, ми маємо 6 каналів, оскільки ми можемо підключити реле з 6 каналами до нашого контролера Arduino.

Кожен канал (Chanel) може мати пвний режим роботи, таких як: «Timer», «Timer RTC», «Day», «Sensor».

Timer - простий періодичний таймер: задаються періоди Period і час Work в форматі ГГ: ХХ: СС. З періодом Period відбувається обрану дію і виконується протягом періоду Work. Наприклад, Period встановлено 1 годину, Work - 10 секунд. Щогодини буде відбуватися дія протягом 10 секунд, тобто якщо обраний канал реле, то реле включиться і вимкнеться через 10 секунд, потім знову включиться через годину і вимкнеться через 10 секунд і так далі. Як канал поводиться на ділянці Work задається в параметрі Direction, тобто це може бути вкл викл і викл вкл (реле), направо наліво і наліво направо (серво) і відкрити закрити і закрити відкрити (лінійний привід) . Даний режим не має прив'язки до реального часу, перезавантаження системи скидає поточний таймер. Застосування: полив в гідропонних системах, провітрювання без датчика.

Timer RTC - періодичний таймер, на відміну від попереднього володіє прив'язкою до реального часу, має налаштування Period включення і тривалості Work (в секундах), яка буде відбуватися, і Start hour - початкового години, з якого починається відлік періоду (для періодів понад 2 годин). Наприклад, період 15 хвилин, робота 10 секунд: кожні 15 хвилин буде проводитися дію тривалістю 10 секунд. Прив'язка до реального часу працює наступним чином: дія буде відбуватися з обраним періодом від початку години, тобто якщо обраний 15 хвилинний, то дія буде в 0, 15, 30 і 45 хвилин кожної години. Якщо обраний Period більше години (від двох і більше) то можна вибрати годину Start from, від якого піде відлік. Всі періоди кратні 24 годинах, тому робота починається в одні і ті ж години кожного дня. Наприклад: Period 8 годин, початковий годину 0. Дія буде виконано в 0, 8 і 16 годин кожного дня. Якщо поставити початковий годину (Start hour) 3 години,

то дія буде виконано в 3, 11 і 19 годин кожного дня. При скиданні харчування наступна дія буде здійснено найближчим часом. Застосування: полив в гідропонних системах, провітрювання без датчика.

Day - простий таймер на одну дію з прив'язкою до реального часу, має налаштування On (час в форматі ГГ: ХХ: СС) - час, з якого дія активно, і Off (час в форматі ГГ: ХХ: СС) - час, з якого дія не активно. Також є 7 днів тижня Days, з понеділка по неділю. При перезавантаженні дію повернеться в потрібне положення згідно з поточним часу. Приклад: таймер налаштований на 6 і 20 годин (Start і Stop). Відповідне поточного каналу та параметру Direction дію буде активно з 6 до 20 годин, і неактивно з 20 до 6 години ранку наступного дня. Застосування: ідеальний режим для освітлення і рідкісного поливу.

Sensor - дія на основі датчика. З періодом опитування Period опитується обраний датчик під назвою Sensor і при перевищенні порогового значення maxV і виконується дія відповідно до обраного каналу (реле, або серво, або привід). Дія «відключиться» при досягненні величиною порога minV, таким чином реалізований гистерезис. Period опитування задається в секундах або хвилинах (в міру збільшення). Датчик вибирається зі списку: Air t. - температура повітря, Air h. - вологість повітря і 4 аналогових датчика (вологості грунту) з SENS_1 по SENS_4. Граничне значення (minV і maxV) задається з 0 до 255 з кроком 1 до значення 50 і з кроком 10 починаючи від 50 (датчики вологості грунту мають діапазон значень 0-255). Наприклад, обраний датчик температури повітря, період опитування 1 годину і граничне значення 25. Щогодини система перевіряє температуру, при перевищенні 25 градусів буде виконано відповідне каналу дію (наприклад включити реле, відкрити вікно). Через годину буде знову проведено перевірку. Застосування: відкриття, закриття стулок по температурі, полив по вологості грунту, управління вентилятором, зволожувачем (реле), або заслінками (серво) по температурі, або вологості.

Наступні відділи це Servo, вони для контролю положенню та стану сервоприводу. Тут можна задати ліміти руху сервопривода та точно также як і в відділі Chanel, обрати режим роботи.

Також існує віділ Service, з якого ми починаємо першу налаштування нашого контролера, а саме встановлюємо час, та можливість видалення певних пунктів меню, як наприклад Chanel, так як ми маємо 6 відділів даного меню, але у нас може бути підключенно тільки 4 прилада, тобто у нас не має потреби в Chanel 5 та в Chanel 6 тому ми маємо можливість видалити їх в данному відділі, або налаштувати швидкий доступ до певних відділів.

2.4. Проектування функціональної схеми СУМТ

Розглянемо створену функціональну схему (рис.2.9) системи управління мікрокліматом теплиці.

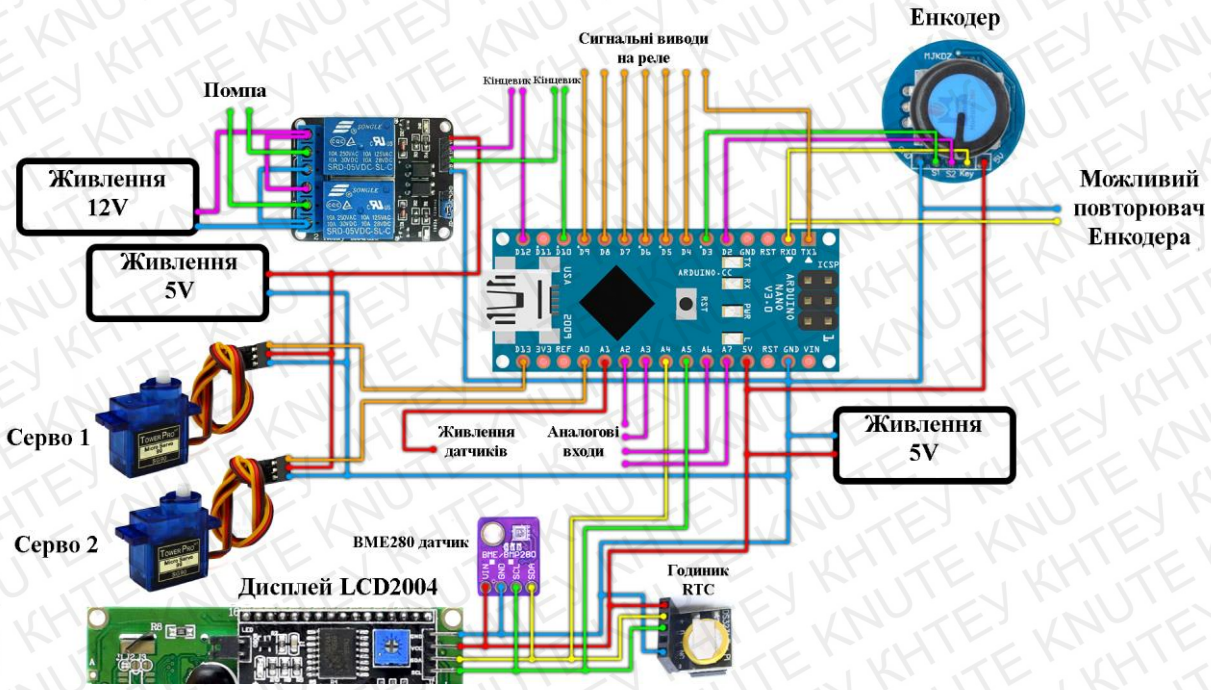

Рис.2.9 Функціональна схема системи управління мікрокліматом теплиці

Перше і найголовніше це живлення нашого контролера, а саме нам потрібно 5V:

- для живлення нашого контролера Arduino;
- для живлення годиника реального часу;
- для живлення енкодера;

- для живлення дисплею;
- для живлення датчика BME280;
- для живлення інших підключених датчиків;
- для живлення сервоприводів;
- для живлення реле;

Для роботи водяної помпи нам необхідно живлення 12V, яке буде підключатися до кінцевиків реле. Живлення забезпечне завдяки, блокам живлення відповідного номіналу.

РОЗДІЛ 3.
РОЗРОБКА ПРОГРАМНО-АПАРАТНОЇ СИСТЕМИ ВОДОПОСТАЧАННЯ НА ПЛАТФОРМІ «ARDUINO»

3.1. Характеристика інтегрованого середовища розробки програмного забезпечення для платформи «Arduino»

Середовище розробки Arduino представляє собою текстовий редактор програмного коду, область повідомлень, вікно виведення тексту (консоль), панель інструментів і кілька меню. Для завантаження програм і зв'язку середовище розробки підключається до апаратної частини Arduino.

Меню редактора включає в себе наступні основні елементи: файл, правка, скетч, інструменти і довідка. Розглянемо докладніше кожен з них.

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

- створити - створити нову програму (скетч);
- відкрити - відкрити існуючу програму;
- папка зі скетчами - відкрити програму із заданої папки;
- приклади - відкрити приклад програми;
- закрити – закрити поточне вікно.
- зберегти - зберегти зміни в попередньо збереженій програмі;
- зберегти як - зберегти нову програму, із зазначенням імені;

- завантажити - завантажити програму в Arduino;
- завантажити за допомогою програматора - завантажити програму за допомогою програматора;
- налаштування друку - налаштування принтера;
- друк - виведення на друк коду програми;
- налаштування – налаштування редактора;
- вихід - вихід з Arduino IDE.

В меню «Правка» розташовані команди для роботи з кодом вашої програми. Часто використовувані команди зручні наявністю комбінацій для швидкого доступу за допомогою клавіатури. Зручними функціями є можливість копіювання для форумів і в html форматі, що дозволяють ділитися вашими скетчами, зберігаючи наочність розмітки у вигляді BB кодів або html розмітки відповідно[19].

У меню «Скетч» розміщуються команди для контролю за процесом компіляції програми.

- перевірити / компілювати - компілювати програму;
- показати папку скетчів - відкрити системну папку з програмами;
- додати файл - додати до проекту файл з даними або програмою;
- імпортувати бібліотеку - підключити до програми бібліотеку зі списку встановлених.

Окремо хотілося б зупинитися на імпорті бібліотек. Arduino IDE містить безліч попередньо бібліотек. Бібліотеки додають додаткову функціональність скетчам, наприклад, при роботі з апаратною частиною або при обробці даних. Одна або кілька директив #include будуть розміщені на початку коду скетчу з подальшою компіляцією бібліотек і разом зі скетчем. Завантаження бібліотек вимагає додаткового місця в пам’яті Arduino. Для встановлення сторонніх бібліотек можна скористатися командою «Імпортувати бібліотеку».

Меню «Довідка» містить докладний опис всіх функцій самого редактора Arduino IDE, а також команди і прийоми роботи з платформою Arduino.

Пункт меню «Інструменти» включає в себе допоміжні функції для роботи з самим мікро контролером.

- автоформатування - автоматична розстановка відступів, переносів рядків і т.п .;
- архівувати скетч - архівація папки з програмою, і збереження архіву у вказане місце;
- виправити кодування і перезавантажити;
- монітор порту - відкрити вікно для обміну даними з мікро контролером;
- плата - вибір поточної плати;
- послідовний порт - вибір порту, до якого підключений пристрій;
- програматор - вибір програм;
- записати завантажувач - запис программизагрузчіка в мікроконтролер.

Кожна програма для Arduino може складатися з декількох файлів. Для перемикання між цими файлами служить система вкладок у редакторі. Там же, можна створити нову вкладку, і асоціювати з нею файл в папці з проектом [19].

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

У самому низу редактора Arduino IDE є невелике вікно, що служить для виведення повідомлень про проблеми, які виникають в процесі.

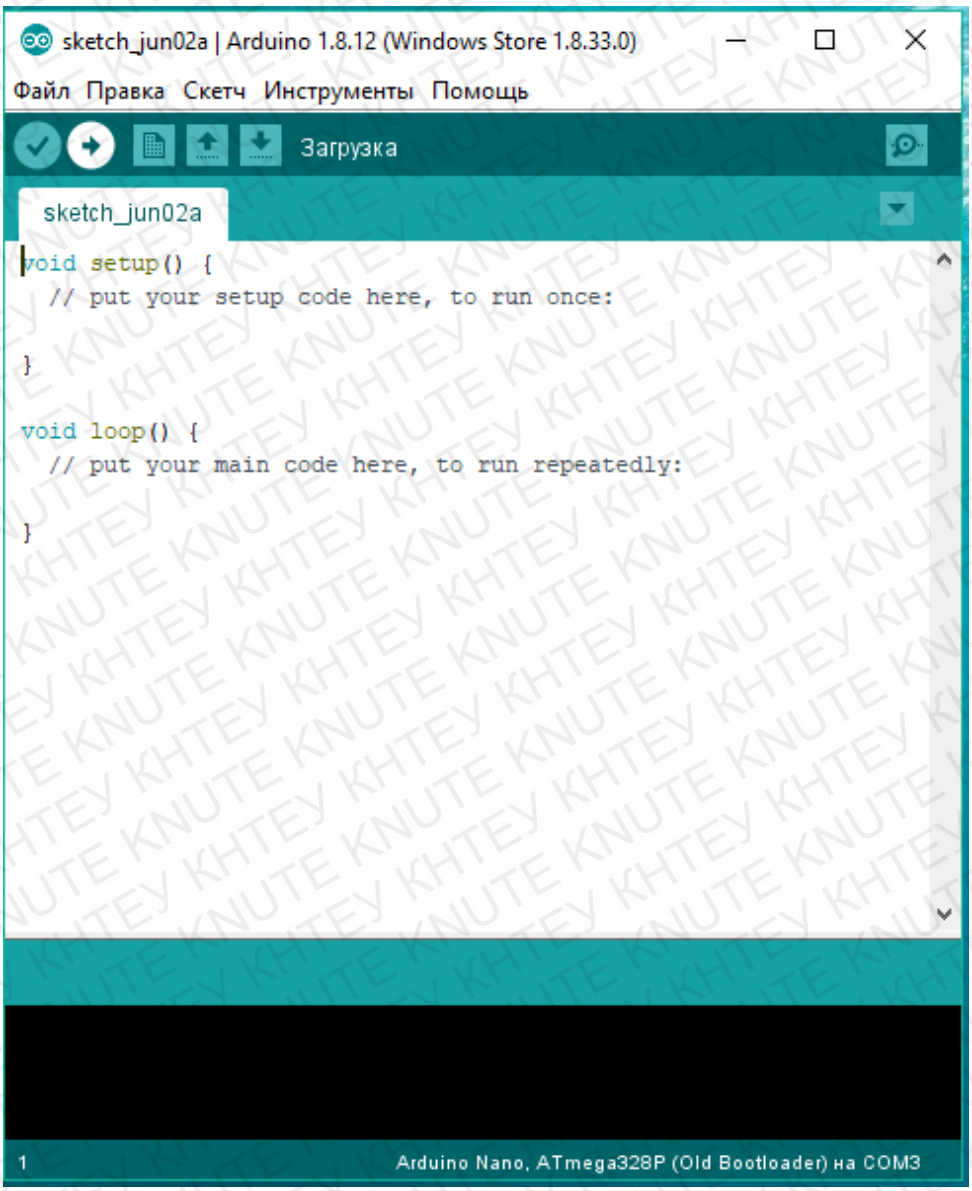

Рис.3.1. Інтерфейс Arduino IDE

3.2. Розробка алгоритмів функціонування СУМТ

Алгоритми це основа будь-якої програми, адже нам потрібно «пояснити» контролеру, чого ми від нього очікуємо, саме для цього і пишеться програма. Програма, написана за допомогою певної мови програмування - це переклад в письмовий код очікувань оператора. Оскільки будь-яка програма виконується послідовно рядок за рядком, за винятком переривань, тому ми повинні розуміти, що відбувається на кожному кроці виконання програми, щоб отримати бажаний результат.

Тому ми створемо алгоритм роботи Arduino.

Рис.3.2. Алгоритм роботи Arduino

3.3 Розробка програмного забезпечення СУМТ

Спочатку ми задаємо назви та посиланя на виходи плати Arduino (рис3.3)

```
#define SW          0
#define RELAY_0     1
#define DT          2
#define CLK         3
#define RELAY_1     4
#define RELAY_2     5
#define RELAY_3     6
#define RELAY_4     7
#define RELAY_5     8
#define RELAY_6     9
#define DRV_SIGNAL1 10
#define DRV_PWM     11
#define DRV_SIGNAL2 12
#define SERVO_0     13
#define SERVO_1     A0
#define SENS_VCC    A1
#define SENS_1      A2
#define SENS_2      A3
#define SENS_3      A6
#define SENS_4      A7
#define CO2_RX      A1
```

Рис.3.3. Виходи (Pin) плати Arduino

Далі задаємо процеси роботи кожного вихода (рис3.4)

```
#define USE_PID 1
#endif

#if (SCHEDULE_NUM > 2)
#define SCHEDULE_NUM 2
#endif

#if (PID_AUTOTUNE == 1)
#define SMOOTH_SERVO 0
#endif


#if (SERVO1_RELAY == 1 && SERVO2_RELAY == 1)

#define SMOOTH_SERVO 0
#endif


#if (DHT_SENS2 == 1)
#define USE_BME 0
#endif


#define SCHEDULE_MAX 30


#define EEPR_KEY_ADDR 1022
#define EEPR_KEY 121

#define EEPR_CH 0
#define EEPR_PID 320
#define EEPR_DAWN 460
#define EEPR_SETTINGS 502
#define EEPR_PLOT_D 524
#define EEPR_PLOT_H 704
```

Рис.3.4. Процеси роботи

Оскільки ми використовуємо і динамічні і статичні бібліотеки для роботи наших датчиків, або контролерів, тому створюємо посилання на бібліотеки для кожного процеса, або на файл проекту або на бібліотеки в теці програминої розробки (рис.3.5)

```
#include "encMinim.h"
encMinim enc(CLK, DT, SW, ENC_REVERSE, ENCODER_

#if (SERVO1_RELAY == 0 || SERVO2_RELAY == 0)
#if (SMOOTH_SERVO == 1)
#include <ServoSmooth.h>
#else
#include <Servo.h>
#endif
#endif

#if (SERVO1_RELAY == 0)
#if (SMOOTH_SERVO == 1)
ServoSmooth servo1;
#else
Servo servo1;
#endif
#endif

#if (SERVO2_RELAY == 0)
#if (SMOOTH_SERVO == 1)
ServoSmooth servo2;
#else
Servo servo2;
#endif
#endif

#include <microWire.h>
#include <microLiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(LCD_ADDR, 20, 4);

#include <EEPROM.h>
```

Рис.3.5. Посилання на бібліотеки

Потім створюємо змінні роботи каналів, та граничні елементи роботи елементів (рис.3.6).

```
int8_t lastScreen = 0;


#if (SCHEDULE_NUM > 0)
struct scheduleStruct {
  byte pidChannel = 0;
  int startDay = 1;
  int endDay = 1;
  byte pointAmount = 15;
  int setpoints[SCHEDULE_MAX];
};
scheduleStruct setSchedule, activeSchedule;
#define loadSchedule(x) scheduleStruct(EEPROM.get((x) * EEPR_SHED_STEP + EEPR_SHED, activeSc


const char *schedulePageNames[] = {
  "Channel",
  "Start",
  "End",
  "Amount",
};
#endif

#if (USE_CO2 == 1 && CO2_CALIB == 0)
uint16_t _tx_delay;
uint8_t *_tx_pin_reg;
uint8_t _tx_pin_mask;
#endif

// dawn
#define DAWN_SET_AMOUNT 6
#if (USE_DAWN == 1)
struct dawnStruct {
  int8_t start = 0;
  int8_t stop = 0;
```

Рис.3.6. Зміні каналів та відділів

Потім створюємо режими роботи, та віділи меню, і задаємо посилання на створену схему меню (рис3.7)

```
void incr(boolean* val, int incr) {
  if (incr == 1) *val = true;
  else *val = false;
}
void incr(int* val, int incr, int limit) {
  *val += incr;
  if (*val > limit) *val = limit;
  if (*val < 0) *val = 0;
}
void incr(uint32_t* val, int incr, int limit) {
  if (incr < 0 && *val < -incr) *val = 0;
  else *val += incr;
  if (*val > limit) *val = limit;
  if (*val < 0) *val = 0;
}
void incr(int8_t* val, int incr, int limit) {
  *val += incr;
  if (*val > limit) *val = limit;
  if (*val < 0) *val = 0;
}
void incr(float* val, float incr, float limit) {
  *val += incr;
  if (*val > limit) *val = limit;
  if (*val < 0) *val = 0;
}
void incr(byte* val, int incr, int limit) {
  int value = (int) * val;
  value += incr;
  if (value > limit) value = limit;
  if (value < 0) value = 0;
  *val = (byte)value;
}
void incrInt(int* val, int incr, int limit) {
  *val += incr;
  if (*val > limit) *val = limit;
```

Рис.3.7. Меню та функції його управління

Робимо файл з роботою кожного датчику використовуючи статичні бібліотеки роботи датчиків з відкритого доступу, а саме для: датчик BME280, датчик вологості грунту Arduino, сервопривод, помпа подачі води та інші.

Розроблюємо кожен режим роботи відділів Chanel та Servo, та водимо на них посилання.

3.4. Технологія використання розробленої СУМТ

В завершені виконання проекту можна чітко зазначити функції нащої СУМТ:

- Періодичний полив через реле індивідуальними помпами, або клапанами;

- Полив на основі показників датчиків вологості грунту;
- Полив в зазначені дні тижня з закінченням по таймеру АБО за показаннями з датчика;
- Управління освітленням черех реле з прив'язкою до часу доби
- Провітрювання серво відкриває заслінку по датчику температури або вологості повітря;
- Зволоження, включення зволожувача по датчику вологості повітря;
- Обігрів включення обігрівача по датчику температури;
- Виконання дій сервоприводом натискання кнопок на пристроях, поворот рукояток, поворот заслінок, переміщення предметів по датчику або таймером.

Основним органом управління є енкодер, рукоятку якого може обертати і натискати вона є кнопкою. При запуску системи ми потрапляємо на настройку каналу 0. Обертаючи рукоятку енкодера можна переміщати курсор вибору стрілочка по пунктах меню. Щоб змінити значення вибраного пункту, потрібно натиснути рукоятку енкодера і повернути її, утримуючи. Утриманий поворот при обраному імені каналу - зміна каналу для налаштування. Гортаємо направо і у нас буде по порядку 7 каналів реле, два серво і лінійний привід. Щоб перейти до налаштування режиму, потрібно навести на нього курсор і клацнути кнопкою, не повертаючи. Відкриється вікно налаштування режиму, вийти з якого можна клікнувши по напису BACK назад.

Утримуючи і обертаючи рукоятку на обраному назві режиму можна змінити режим, всього їх 4. У корені меню вибір каналів гортаючи наліво від каналу 0 буде екран налагодження DEBUG, режим налаштувань SETTINGS та сервісний режим SERVICE. На екрані налагодження показані всі поточні положення реле, приводів і показання з датчиків. Обертаючи рукоятку на екрані налагодження послідовно перегортаються добові графіки показань з датчиків: температура повітря, вологість і показання з аналогових датчиків.

Поділу на графіку мають крок 1.6 години. На екрані сервісу можна управляти будь-яким каналом в ручному режимі, при активному екрані сервісу автоматика не працює, система знаходиться повністю в ручному режимі. Поворотом рукоятки можна вибрати потрібний канал, положення серво або настройку поточного часу, і утриманих поворотом її змінити. Якщо включити систему з затиснутою рукояткою енкодера, відбудеться повне скидання налаштувань каналів і режимів. При утриманні кнопки енкодера більше двох секунд без повороту рукоятки привід пересунеться в протилежний зміст, при повторному натисканні - пересунеться назад. Зроблено для доступу в теплицю, у якій привід управляє дверима.

ВИСНОВКИ

У випускному кваліфікаційному проекті представлено результати теоретичних і прикладних досліджень, що полягають у розробці системи управління мікрокліматом теплиці. В результаті проведених досліджень були отримані такі **висновки**:

1. Впровадження автоматизації систем управління мікрокліматом теплиці, спріятіме збільшенню ефективності вірощенню рослин та Зменшення задіяної робочої сили.
2. Створено структурну та функціональну схему роботи системи управління мікрокліматом теплиці з можлівістю її адаптації під певний формат та потреби з подальшою можливістю її удосконаленнювати та адаптаптувати під потреби та вимоги користувача.
3. Враховуючи всі віщезазначені фактори, було розроблено програмне забезпечення системи управління мікрокліматом теплиці. Та створення апаратного забезпечення роботи системи управління мікрокліматом теплиці.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Hydroponics For Everybody: All About Home Horticulture / William Texier - 7 с.
2. Аквапоніка в Україні: URL: http://rodovid.me/Asya/2013/07/12/ akvaponika---vyraschivanie-ryb-i
3. Барабаш О.Ю. Овочівництво: підручник. Київ: Вища школа, 1994. - 374 с.
4. Білецький П.М. Овочівництво. К.: Вища школа, 1970. - 420 с.
5. Боронецкая О.И. Использование тиляпии (Tilapiinae) в мировой и отечественной аквакультуре. Известия Тимирязевской сельскохозяйственной академии. 2012. № 1. С. 164–173.
6. Wilson A. Lennard, Brian V. Leonard A Comparison of Three Different Hydroponic Sub-systems (gravel bed, floating and nutrient film technique) in an Aquaponic Test System. Aquaculture International. 2006. Vol. 14. Вып. 6. P. 539–550.
7. Мураш И. Г. О воздушной культуре растений в закрытом ґрунте // Физиология растений. 1983. Т. 10. Вып. 5. 350 с.
8. Сич З. Д., Сич І. М. Гармонія овочевої краси та користі. Київ : Арістей, 2005. 192 с.
9. Белогубова Е. Н. и др. Современное овощеводство открытого и закрытого ґрунта. Киев : Киевская правда, 2006. 528 с.
10. Шульгина Л. М. Справочник огородника. Харьков : Фолио, 2006. 350 с
11. Слєпцов Ю. В., Федосій І. О. Органічне овочівництво : в 2 ч. Вінниця : Нілан-ЛТД, 2016–2017.
12. Инфосфера: Информационные структуры, системы и процессы в науке и обществе / Арский Ю.М., Гиляревский Р.С., Туров И.С., Чёрный А.И.– М.: ВИНИТИ, 1996.– 489 с.
13. Профессиональное тепличное оборудование. − Москва: http://www.fito-system.ua/
14. Промгідропоніка. −: http://www.promgidroponica.ua/

15. Промишленні компьютери. − http://www.ipc2u.ua/
16. Бородин И.Ф. Автоматизация технологических процессов; -М.: Агропроиздат. 1986. − 368 с.
17. Марченко А.С. Справочник по автоматизации в животноводстве, - К.: Урожай, 1990. − 450 с.
18. Василенко П. М., Василенко И.И., Автоматизация процессов сельскохозяйственного производства. М., «Колос», 1972 −186 с.
19. Еванс Б. Arduino. Блокнот програміста. [пер. з англ. Голобов В.] - - СПб.: БХВ-Петербург 2007. 40с.

ДОДАТОК

Програмний код реалізації

```
#define SW       0
#define RELAY_0   1
#define DT        2
#define CLK       3
#define RELAY_1   4
#define RELAY_2   5
#define RELAY_3   6
#define RELAY_4   7
#define RELAY_5   8
#define RELAY_6   9
#define DRV_SIGNAL1 10
#define DRV_PWM     11
#define DRV_SIGNAL2 12
#define SERVO_0   13
#define SERVO_1   A0
#define SENS_VCC  A1
#define SENS_1    A2
#define SENS_2    A3
#define SENS_3    A6
#define SENS_4    A7
#define CO2_RX    A1


#define DEBUG_ENABLE 0

#if (DEBUG_PID > 0)
#define DEBUG_ENABLE 1
#endif
```

```
#if (DEBUG_ENABLE == 1)
#include <GyverUART.h>
#define DEBUG(x) uart.println(x)
#else
#define DEBUG(x)
#endif


#if (SCHEDULE_NUM > 0)
#define USE_PID 1
#endif

#if (SCHEDULE_NUM > 2)
#define SCHEDULE_NUM 2
#endif

#if (PID_AUTOTUNE == 1)
#define SMOOTH_SERVO 0
#endif


#if (SERVO1_RELAY == 1 && SERVO2_RELAY == 1)

#define SMOOTH_SERVO 0
#endif


#if (DHT_SENS2 == 1)
#define USE_BME 0
```

```
#endif


#define SCHEDULE_MAX 30


#define EEPR_KEY_ADDR 1022
#define EEPR_KEY 121

#define EEPR_CH 0
#define EEPR_PID 320
#define EEPR_DAWN 460
#define EEPR_SETTINGS 502
#define EEPR_PLOT_D 524
#define EEPR_PLOT_H 704
#define EEPR_SHED 884

#define EEPR_CH_STEP 32
#define EEPR_PID_STEP 20
#define EEPR_DAWN_STEP 6
#define EEPR_SETTINGS_STEP 22
#define EEPR_SHED_STEP (6+SCHEDULE_MAX*2)


#include "encMinim.h"
encMinim enc(CLK, DT, SW, ENC_REVERSE, ENCODER_TYPE);

#if (SERVO1_RELAY == 0 || SERVO2_RELAY == 0)
#if (SMOOTH_SERVO == 1)
#include <ServoSmooth.h>
```

```
#else
#include <Servo.h>
#endif
#endif

#if (SERVO1_RELAY == 0)
#if (SMOOTH_SERVO == 1)
ServoSmooth servo1;
#else
Servo servo1;
#endif
#endif

#if (SERVO2_RELAY == 0)
#if (SMOOTH_SERVO == 1)
ServoSmooth servo2;
#else
Servo servo2;
#endif
#endif

#include <microWire.h>
#include <microLiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(LCD_ADDR, 20, 4);

#include <EEPROM.h>

#include <microDS3231.h>
MicroDS3231 rtc;
```

```
// bme
#if (USE_BME == 1)
#include <GyverBME280.h>
GyverBME280 bme;
#endif

#if (DALLAS_SENS1 == 1)
#include <microDS18B20.h>
#if (DALLAS_AMOUNT > 1)
MicroDS18B20 dallas[DALLAS_AMOUNT];
float dallasBuf[DALLAS_AMOUNT];
#else
MicroDS18B20 dallas(SENS_1);
#endif
#endif

#if (DHT_SENS2 == 1)
#include <DHT.h>
DHT dht(SENS_2, DHT_TYPE);
#endif

#if (WDT_ENABLE == 1)
#include <avr/wdt.h>
#endif

int8_t lastScreen = 0;


#if (SCHEDULE_NUM > 0)
struct scheduleStruct {
```

```
  byte pidChannel = 0;
  int startDay = 1;
  int endDay = 1;
  byte pointAmount = 15;
  int setpoints[SCHEDULE_MAX];
 };
 scheduleStruct setSchedule, activeSchedule;
 #define loadSchedule(x) scheduleStruct(EEPROM.get((x) * EEPR_SHED_STEP + EEPR_SHED, activeSchedule))


 const char *schedulePageNames[] = {
  "Channel",
  "Start",
  "End",
  "Amount",
 };
 #endif

 #if (USE_CO2 == 1 && CO2_CALIB == 0)
 uint16_t _tx_delay;
 uint8_t *_tx_pin_reg;
 uint8_t _tx_pin_mask;
 #endif

 // dawn
 #define DAWN_SET_AMOUNT 6
 #if (USE_DAWN == 1)
 struct dawnStruct {
  int8_t start = 0;
```

```
  int8_t stop = 0;
  uint8_t dur1 = 30;
  uint8_t dur2 = 30;
  uint8_t minV = 0;
  uint8_t maxV = 255;
  // 6
 };
 dawnStruct setDawn, activeDawn;
 #define loadDawn(x) dawnStruct(EEPROM.get((x) * EEPR_DAWN_STEP
+ EEPR_DAWN, activeDawn))
 #endif

 // PID
 #define PID_CH_AMOUNT 7
 #define PID_SET_AMOUNT 8
 #if (USE_PID == 1)
 struct PIDstruct {
  int8_t sensor = 0;
  float kP = 0.0;
  float kI = 0.0;
  float kD = 0.0;
  byte dT = 1;
  byte minSignal = 0;
  byte maxSignal = 200;
  float setpoint = 20;
  // 20
 };
 PIDstruct activePID, setPID;
```

```
#define loadPID(x) PIDstruct(EEPROM.get((x) * EEPR_PID_STEP + EEPR_PID, activePID))
#define savePID(x) (EEPROM.put((x) * EEPR_PID_STEP + EEPR_PID, activePID))

float integralSum[PID_CH_AMOUNT];
float prevInput[PID_CH_AMOUNT];
float input[PID_CH_AMOUNT];
int output[PID_CH_AMOUNT];
uint32_t PIDtimers[PID_CH_AMOUNT];
#endif

#if (USE_PID_RELAY == 1)
uint32_t tmr0, tmr1;
bool flag0, flag1;
#endif

// Settings
#define SETTINGS_AMOUNT 8
#if (SMOOTH_SERVO == 1)
#define SETTINGS_AMOUNT 12
#endif
#if (PID_AUTOTUNE == 1)
#define SETTINGS_AMOUNT 19
#endif

#if (PID_AUTOTUNE == 1)
struct {
  bool tuner = false;
  bool restart = true;
```

```
  bool result = false;
  byte channel = 0;
  byte sensor = 0;
  bool manual = false;
  byte steady = 50;
  byte step = 25;
  float window = 0.1;
  byte kickTime = 30;
  byte delay = 20;
  byte period = 1;
 } tunerSettings;

 struct {
  byte status = 0;
  uint32_t cycle = 0;
  float min = 0;
  float max = 0;
  float P = 0;
  float I = 0;
  float D = 0;
  byte value = 0;
  float input = 0;
 } tuner;

 const char *tuneNames[] = {
  "off",
  "wait steady",
  "wait kick",
  "step up",
  "step down",
```

```
};
#endif

struct {
  boolean backlight = 1;
  byte backlTime = 60;
  byte drvSpeed = 125;
  byte srv1_Speed = 40;
  byte srv2_Speed = 40;
  float srv1_Acc = 0.2;
  float srv2_Acc = 0.2;
  int16_t comSensPeriod = 1;
  int8_t plotMode = 0;
  byte minAngle[2] = {0, 0};
  byte maxAngle[2] = {180, 180};
  int16_t driveTimeout = 50;
} settings; //21

// Channels
struct channelsStruct {
  boolean type = 0;
  boolean state = 0;
  boolean direction = true;
  boolean global = false;
  int8_t week = 0;
  int8_t sensor;
  int8_t relayType;
  int8_t mode;
  int8_t startHour = 0;
  int8_t impulsePrd = 1;
```

```
  int16_t threshold = 30;
  int16_t thresholdMax = 30;
  int16_t sensPeriod = 2;
  uint32_t period = 100;
  uint32_t work = 10;
  uint32_t weekOn = 100;
  uint32_t weekOff = 10;
};
// 32
channelsStruct activeChannel, setChannel;

#define loadChannel(x) channelsStruct(EEPROM.get((x) * EEPR_CH_STEP, activeChannel))

uint32_t timerMillis[10];

uint32_t driveTimer;
byte driveState;
boolean lastDriveState;
boolean manualControl;
boolean manualPos;
boolean controlState;

const byte PIDchs[] = {0, 1, 2, 3, 7, 8, 9};
const byte channelToPWM[] = {0, 1, 2, 3, 0, 0, 0, 4, 5, 6};
const byte impulsePrds[] = {1, 5, 10, 15, 20, 30, 1, 2, 3, 4, 6, 8, 12, 1, 2, 3, 4, 5, 6, 7};
const byte relayPins[] = {RELAY_0, RELAY_1, RELAY_2, RELAY_3, RELAY_4, RELAY_5, RELAY_6, SERVO_0, SERVO_1};
```

```
float sensorVals[6];
int8_t realTime[3];
float uptime = 0;
byte servoPosServ[2];
const int PWMperiod = (float)1000 / PWM_RELAY_HZ;
int PWMactive[2];

int pwmVal[7];
boolean drivePidFlag;

boolean channelStates[10];
boolean channelStatesServ[10];
int8_t debugPage;

int8_t arrowPos;
int8_t navDepth;
int8_t currentChannel = 0;
int8_t currentMode;
int8_t thisH[2], thisM[2], thisS[2];
int8_t currentLine;
uint32_t commonTimer, backlTimer, plotTimer;
boolean backlState = true;

int sensMinute[6][15];

boolean serviceFlag;
boolean startPID = false;
boolean timeChanged;
boolean startFlagDawn;
uint32_t settingsTimer;
```

```
    uint32_t driveTout;

    byte thisMode;
    byte curMode;

    const byte daysMonth[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31, 0};
    uint16_t thisDay;

    #if (START_MENU == 1)
    bool startService = false;
    #endif

    #if (USE_PLOTS == 1 || USE_PID == 1 || USE_DAWN == 1)
    // график
    const char row8[] PROGMEM = {0b11111, 0b11111, 0b11111, 0b11111, 0b11111, 0b11111, 0b11111, 0b11111};
    const char row7[] PROGMEM = {0b00000, 0b11111, 0b11111, 0b11111, 0b11111, 0b11111, 0b11111, 0b11111};
    const char row6[] PROGMEM = {0b00000, 0b00000, 0b11111, 0b11111, 0b11111, 0b11111, 0b11111, 0b11111};
    const char row5[] PROGMEM = {0b00000, 0b00000, 0b00000, 0b11111, 0b11111, 0b11111, 0b11111, 0b11111};
    const char row4[] PROGMEM = {0b00000, 0b00000, 0b00000, 0b00000, 0b11111, 0b11111, 0b11111, 0b11111};
    const char row3[] PROGMEM = {0b00000, 0b00000, 0b00000, 0b00000, 0b00000, 0b11111, 0b11111, 0b11111};
    const char row2[] PROGMEM = {0b00000, 0b00000, 0b00000, 0b00000, 0b00000, 0b00000, 0b11111, 0b11111};
    const char row1[] PROGMEM = {0b00000, 0b00000, 0b00000, 0b00000, 0b00000, 0b00000, 0b00000, 0b11111};
```

```
#endif

// co2
#if (USE_CO2 == 1)
#if (CO2_PIN == 1)
#define CO2_PIN_NAME SENS_1
#elif (CO2_PIN == 2)
#define CO2_PIN_NAME SENS_2
#else
#error "Wrong CO2 pin"
#endif

int CO2ppm = 0;
bool CO2_flag = false;
bool CO2_rst = false;

void CO2tick() {
 static uint32_t tmr;
 if (millis() - settingsTimer > 1000) {
  if (digitalRead(CO2_PIN_NAME)) {
   if (!CO2_flag) {
    tmr = millis();
    CO2_flag = true;
   }
  } else {
   if (CO2_flag) {

    if (!CO2_rst) {
     tmr = millis() - tmr;
     CO2ppm = (CO2_MAX / 1000) * (tmr - 2);
```

```
      }
      CO2_flag = false;
      CO2_rst = false;
    }
   }
  }
}
#endif

const char *settingsPageNames[]  = {
  "A-BKL",
  "BKL-time",
  "Drv",
  "Day",
  "Month",
  "Year",
  "Sens prd",
  "Plot prd",
#if (SMOOTH_SERVO == 1)
  "S1 sp",
  "S1 acc",
  "S2 sp",
  "S2 acc",
#endif
#if (PID_AUTOTUNE == 1)
  "Tuner",
  "Result",
  "Channel",
  "Sensor",
  "Manual",
```

```
  "Steady",
  "Step",
  "Window",
  "Kick time",
  "Delay",
  "Period",
#endif
};

#if (USE_PID == 1)
const char *pidNames[] = {
  "P",
  "I",
  "D",
  "Sens",
  "Set",
  "T",
  "Min",
  "Max",
};
#endif
#if (USE_DAWN == 1)
const char *dawnNames[]  = {
  "Start",
  "Dur. up",
  "Stop",
  "Dur. down",
  "Min",
  "Max",
};
```

```
#endif
const char *settingsNames[]  = {
  "Mode",
  "Direction",
  "Type",

  "Mode",
  "Direction",
  "Limits",

  "Mode",
  "Direct.",
  "Timeout",
};

const char *modeNames[]  = {
  "<Timer>",
  "<Timer RTC>",
  "<Week>",
  "<Sensor>",
  "<PID>",
  "<Dawn>",
};

const char *relayNames[]  = {
  "Relay",
  "Valve",
  "Common",
};
```

```
const char *modeSettingsNames[]  = {
  "Period",   // 0
  "Work",     // 1
  "Left",     // 2

  "Period",   // 3
  "Work",     // 4
  "Start hour", // 5

  "",    // 6
  "",     // 7

  "Period",   // 8
  "Sensor",   // 9
  //"Threshold",  // 10
};


const char *directionNames[]  = {
  "Off-On",
  "On-Off",
  "Min-Max",
  "Max-Min",
  "Close-Open",
  "Open-Close",
};

// названия
#define SENS1_NAME "Sen1"
#define SENS2_NAME "Sen2"
```

```
#define SENS3_NAME "Sen3"
#define SENS4_NAME "Sen4"

#if (DALLAS_SENS1 == 1)
#define SENS1_NAME "Dall"
#endif

#if (THERM1 == 1)
#define SENS1_NAME "Tmp1"
#endif
#if (THERM2 == 1)
#define SENS2_NAME "Tmp2"
#endif
#if (THERM3 == 1)
#define SENS3_NAME "Tmp3"
#endif
#if (THERM4 == 1)
#define SENS4_NAME "Tmp4"
#endif

#if (USE_CO2 == 1)
#if (CO2_PIN == 1)
#define SENS1_NAME "CO2"
#elif (CO2_PIN == 2)
#define SENS2_NAME "CO2"
#endif
#endif

const char *sensorNames[]  = {
  "AirT",
```

```
  "AirH",

  SENS1_NAME,
  SENS2_NAME,
  SENS3_NAME,
  SENS4_NAME,
};


const char *plotNames[]  = {
  "Min",
  "Hour",
  "Day",
};



void smartArrow(bool state = true);
```