ДВНЗ «Донецький національний технічний університет»
Факультет комп'ютерно-інформаційних технологій та автоматизації
(повне найменування інституту, назва факультету)
Кафедра електронної техніки
(повна назва кафедри)

«До захисту допущено»
Завідувач кафедри електронної техніки
_____ Олександр ВОВНА
(підпис) (ініціали, прізвище)
«___» _____ 2021 р.

Випускна кваліфікаційна робота

магістра
(освітньо-кваліфікаційний рівень)

на тему «Розробка та дослідження електронної системи контролю вологості тепличних ґрунтів на базі нечіткої логіки»

Виконав (ла) студент (ка) 2 курсу, групи ЕЛТм–20
(шифр групи)

(шифр і назва напрямку підготовки)

спеціальності 171 Електроніка
(шифр і назва спеціальності)

Бережний Максим Олегович
(прізвище та ініціали) (підпис)

Керівник проф. каф. ЕТ, д.т.н., доц. _____ І.С. ЛАКТІОНОВ
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Рецензент _____ _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цій випускній кваліфікаційній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Покровськ – 2021 р.

ДВНЗ «Донецький національний технічний університет»

Факультет комп'ютерно-інформаційних технологій та автоматизації

(повне найменування інституту, назва факультету)

Кафедра електронної техніки

(повна назва кафедри)

Захист відбувся _____

(дата)

з оцінкою_____

Секретар ДЕК _____

(підпис)

Випускна кваліфікаційна робота

магістра

Тема: «Розробка та дослідження електронної системи контролю вологості тепличних ґрунтів на базі нечіткої логіки»

Спецчастина:_____

Виконавець, студент (ка)
гр. ЕЛТм – 20 Максим БЕРЕЖНИЙ
(підпис, дата, Ім'я та ПРІЗВИЩЕ)

Керівник Іван ЛАКТІОНОВ
(підпис, дата, Ім'я та ПРІЗВИЩЕ)

Консультанти: Олександр ВОВНА
(підпис, дата, Ім'я та ПРІЗВИЩЕ)

Олександр ШТЕПА
(підпис, дата, Ім'я та ПРІЗВИЩЕ)

Іван ЛАКТІОНОВ
(підпис, дата, Ім'я та ПРІЗВИЩЕ)

Нормоконтроль Олександр ВОВНА
(підпис, дата, Ім'я та ПРІЗВИЩЕ)

Покровськ – 2021

Державний вищий навчальний заклад
"Донецький національний технічний університет"

Інститут, факультет Комп'ютерно-інформаційних технологій та автоматизації
Кафедра Електронна техніка
Освітньо-кваліфікаційний рівень магістр
Галузі знань 17 Електроніка та телекомунікації
(шифр і назва)
Спеціальність 171 Електроніка
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри електронної техніки
Олександр ВОВНА
«____» _____2021 року

З А В Д А Н Н Я
НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ МАГІСТРА

Бережний Максим Олегович
(прізвище, ім'я, по батькові)
1. Тема проекту (роботи) Розробка та дослідження електронної системи контролю вологості тепличних ґрунтів на базі нечіткої логіки
керівник проекту (роботи) Лактіонов Іван Сергійович, д.т.н., доц.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від 22.09.2021 року № 570
2. Строк подання студентом проекту (роботи)_____

3. Вихідні дані до проекту (роботи)_____
Технічна документація та матеріали з переддипломної практики

4.Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) аналіз та узагальнення відомих результатів у предметній області досліджень; розробка структурної схеми електронного пристрою; розробка математичної моделі досліджуваних параметрів; розробка алгоритмів керування мікропроцесорної частини електронного пристрою

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1, 3	проф. каф. ЕТ, доц., д.т.н., Лактіонов І.С.		
2	проф. каф. ЕТ., д.т.н., проф. Зорі А.А.		
4	зав. каф. ЕТ, д.т.н., проф., Вовна О.В.		

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Розділ 1. Аналіз відомих результатів розробки систем контролю вологості ґрунту	06.09-26.09	
2	Розділ 2. Розробка структурної схеми та математичної моделі зміни вологості ґрунту	26.09-12.10	
3	Розділ 3. Створення імітаційної моделі системи з урахуванням досліджуваних матеріалів та програмного забезпечення	13.10-07.11	
4	Розділ 4. Розробка дослідного зразка електронної системи контролю вологості ґрунту	08.11-01.12	
5	Додаток А. Охорона праці та безпека під час надзвичайних ситуаціях на підприємстві	27.09-04.10	
6	Додаток Б. Лістинг програми в ППП Mathcad «Математичне моделювання зміни вологості тепличного ґрунту»	13.10-07.11	
7	Додаток В, Початковий код програми після компіляції	08.11-01.12	
8	Додаток Г. Відредагований лістинг програмного коду який використовується в системі	08.11-01.12	

Студент _____ **Максим БЕРЕЖНИЙ**
(підпис) (Ім'я та ПРІЗВИЩЕ)

Керівник роботи _____ **Іван ЛАКТІОНОВ**
(підпис) (Ім'я та ПРІЗВИЩЕ)

ЛИСТ ЗАУВАЖЕНЬ

Посада П.І.Б.	Суть зауваження, оцінка та підпис

АНОТАЦІЯ

Бережний, М.О. Розробка та дослідження електронної системи контролю вологості тепличних ґрунтів на базі нечіткої логіки / Випускна кваліфікаційна роботи на здобуття освітнього ступеня «магістр» зі спеціальності 171 Електроніка – ДВНЗ «ДонНТУ», Покровськ 2021.

Пояснювальна записка: 126 стор. (з них основного тексту 72 стор.), 39 рис., 15 табл., 32 посилання.

В роботі було проведено дослідження, аналіз та розробка системи контролю вологості тепличних ґрунтів на базі нечіткої логіки. В процесі виконання роботи було охоплено повний цикл розробки системи від аналізу вже існуючих розробок до розробки готового макету власного проекту. Першим кроком був аналіз існуючих, робочих систем з моніторингу і керування вологістю тепличних ґрунтів, виокремлення їхніх переваг та недоліків. Другий крок - з проведеного аналізу і недоліків систем було створено алгоритм роботи власної, нової системи і її структурна схема. Третій крок – дослідження залежності довжини кореня та його занурення у ґрунт від віку рослини, а також, на основі отриманих даних, створення математичної моделі поглинання води тепличним ґрунтом і розрахунок часу роботи поливного насосу. Четвертий крок – створення бази правил нечіткої логіки за допомогою пакету програмного забезпечення MATLAB&SIMULINK та розширення FuzzyLogicToollbox а також створення імітаційної моделі для тестування справності роботи бази правил в розширенні Simulink. П’ятий крок – за допомогою сайту онлайн конвертації було перетворено початковий файл з базою нечітких правил в програмний код який можна використовувати для подальшої розробки системи. Шостим кроком є створення і випробовування імітаційної моделі системи в програмному забезпеченні Proteus. Сьомим і останнім кроком розробки системи було створення реальної моделі системи та її тестування в реальних умовах. Розроблена система має здатність для подальшого дослідження, модернізації та розширення свого функціоналу.

Ключові слова:

автоматична система, моніторинг, керування, нечітка логіка, вологість ґрунту, математична модель, імітаційна модель, вхідний та вихідний сигнал, програмне забезпечення, фаза росту.

Перелік публікацій:

- Обґрунтування плану експериментальних досліджень комп'ютеризованої системи моніторингу та керування зволоженням тепличних культур / [Лактіонов І. С., Лебедєв В. А., Лактіонова Г. А., Бережний М. О.]. – Наукові праці ДонНТУ. Серія “Обчислвальна техніка та автоматизація”. – Покровськ. 2019. - Випуск №1(32). – 103-113 с.
- Комп’ютеризована система комплексного моніторингу й керування мікрокліматом промислових теплиць на базі нечіткої логіки / [Лактіонов І. С., Вовна О. В., Бережний М. О., Лебедєв В. А.]. – К. : Вістник КрНУ імені Михайла Остроградського. Випуск 3/2019 (116). – 120 с.
- Перспективні напрямки сучасної електроніки, інформаційних і комп’ютерних систем: тези доповідей на IV Всеукр. наук.-практ. конф. MEICS-2019. – К. : м. Дніпро 2019 – 77 с.
- Програмно-апаратне забезпечення комп'ютерно-інтегрованої системи контролю і керування вологістю тепличного ґрунту на базі нечіткої логіки / [Лактіонов І.С., Вовна О.В., Бережний М.О.]. – Науково-виробничий журнал "Електромеханічні і енергозберігаючі системи" . – Кременчук. 2020. - В редакції

ABSTRACT

Berezhnyi, M.O. Development and research of electronic system of humidity control of greenhouse soils on the basis of fuzzy logic / Graduation qualification work for obtaining an educational degree 'Master' in specialty 171 Electronics. – SHEI 'DonNTU', Pokrovsk, 2021.

Explanatory note: 126 pages (of which the main text is 72 pages), 39 figures, 15 tables, 32 references.

The study, analysis and development of a system for controlling the humidity of greenhouse soils on the basis of fuzzy logic was carried out. In the process of work, the full cycle of system development was covered from the analysis of existing developments to the development of a ready-made layout of your own project. The first step was to analyze the existing, working systems for monitoring and managing the humidity of greenhouse soils, highlighting their advantages and disadvantages. The second step - from the analysis and shortcomings of systems the algorithm of work of own, new system and its structural scheme was created. The third step is to study the dependence of the root length and its immersion in the soil on the age of the plant, as well as, based on the obtained data, to create a mathematical model of water absorption by greenhouse soil and calculate the operating time of the irrigation pump. The fourth step is to create a fuzzy logic rule database using the MATLAB & SIMULINK software package and the FuzzyLogicToollbox extension, as well as to create a simulation model to test the operation of the rule database in the Simulink extension. Step Five - The online conversion site has converted the source file with a fuzzy rule database into program code that can be used to further develop the system. The sixth step is to create and test a simulation model of the system in Proteus software. The seventh and final step in developing the system was to create a real model of the system and test it in real conditions. The developed system has the ability to further research, upgrade and expand its functionality.

Keywords:

automatic system, monitoring, control, fuzzy logic, soil moisture, mathematical model,

simulation model, input and output signal, software, growth phase.

List of publications:

- Substantiation of the plan of experimental researches of the computerized system of monitoring and management of humidification of hothouse cultures / [Laktionov IS, Lebedev VA, Laktionova GA, Berezhny MO]. - Scientific works of DonNTU. Computer and Automation Series. - Pokrovsk. 2019. - Issue №1 (32). - 103-113 p.

- Computerized system of complex monitoring and microclimate control of industrial greenhouses on the basis of fuzzy logic / [Laktionov IS, Vovna OV, Berezhny MO, Lebedev VA]. - Kyiv: Bulletin of Mykhailo Ostrogradskyi KrNU. Issue 3/2019 (116). - 120 s.

- Perspective directions of modern electronics, information and computer systems: abstracts of reports on IV All-Ukrainian. scientific-practical conf. MEICS-2019. - K.: Dnipro2019 - 77 c.

- Prohramno-aparatne zabezpechennia kompiuterno-intehrovanoi systemy kontroliu i keruvannia volohistiu teplychnoho gruntu na bazi nechitkoi lohiky / [Laktionov I.S., Vovna O.V., Berezhnyi M.O.]. – Naukovo-vyrobnychyi zhurnal "Elektromekhanichni i enerhozberihaiuchi systemy" . – Kremenchuk. 2020. - V redaktsii.

ЗМІСТ

ВСТУП

В сільському господарстві дуже активно використовується тепличний спосіб вирощування рослин, як для вирощування розсади так і для отримання зрілих овочів. Тепличне вирощування дозволяє розпочати вирощування рослин набагато раніше ніж вирощування у відкритому середовищі.

Теплиця представляється у вигляді конструкції зі скляним або пластиковим дахом та стінами з такого ж матеріалу. Вона нагрівається тому що сонячне випромінення або система обігріву в самій теплиці, зігріває рослини, ґрунт та інші речі що знаходяться всередині. Іншими словами, теплиця це структура, як правило, з скла або прозорого пластику, яка забезпечує захист та контрольоване середовище для вирощування рослин в приміщенні.

Постановка проблеми

Вода – один з найважливіших елементів який потрібен для життєдіяльності, росту та родючості рослин. Об’єм води та частота поливу змінюється в залежності від температури, довжини світлового дня, фази розвитку рослини та виду ґрунту. Як відомо більшість садівників використовує ручний спосіб поливу. Ця система неефективна через те, що під час ручного способу поливу вірогідність надмірного зволоження, а саме великого відсотку вологи в ґрунті (вологості ґрунту) дуже велика. Надмірна вологість ґрунту може призвести до затоплення рослин або до загнивання частин рослини. Недостатня вологість призводить до заповільненого розвитку рослини, її плодів або навіть до повного засихання та загибелі рослини.

Для того щоб вирішити цю проблему необхідно використовувати автоматичну систему поливу. Необхідна така система, котра б була здатна зчитувати основні параметри що впливають на відсоток вологості ґрунту в реальному часі та не допускала б перерегулювання. Дана система повинна опитувати датчики, та в залежності від вхідних параметрів обирати найліпший

варіант для поливу ґрунту.

Такими параметрами є:

- Фаза росту рослини;
- Вологість ґрунту;
- Тип ґрунту.

Методологія

Основним методом регулювання вологості ґрунту в теплиці буде використання крапельного поливу. В якості розвитку та оновлення основного методу буде створено систему автоматичного моніторингу та варіативного керування зволоженням ґрунту в теплиці. До системи на вході підключається датчик вимірювання вологості ґрунту, на виході підключається поливний насос. Окрім цього в програмному забезпеченні буде вписано математичну модель залежності зволоження ґрунту від дійсного значення вологості а також залежність розвитку рослини від її фази росту. Система зчитує інформацію з датчика, враховує дійсне значення вологості і росту рослини та подає вихідний, керуючий, сигнал на насос. Усе керування обчислюється згідно правил, раніше введених у систему.

Наукова новизна

Було виконано роботу з розробки електронної системи контролю вологості тепличних ґрунтів на базі нечіткої логіки. Основне відрізнення від досліджених існуючих системи моніторингу та керування вологістю ґрунтів є використання бази правил нечіткої логіки та враховує потреби рослин у воді в залежності від віку рослини. Дана розробка по-перше покращує якість рослини через врахування необхідної кількості вологи яка потрібна рослині у різний час росту, та по-друге дозволяє зменшити перерегулювання роботи насосу в залежності від потреб рослини і тим самим зменшити надлишкової витрати води.

Окрім цього було удосконалено метод моніторингу вимірювання вологості ґрунту за рахунок прогнозування вологості на різних глибинах при вимірюванні під поверхнею ґрунту.

Практична значимість

Розробка електронної системи дозволяє економно та ефективно вирощувати рослини від першого дня посадки в ґрунт до кінцевого збору врожаю. Система забезпечує найоптимальніші умови вологості ґрунту в залежності від доби росту рослини та її фази росту. Окрім цього система забезпечує зменшення перерегулювання роботи поливного насосу за допомогою бази правил нечіткої логіки. Розробка системи виконувалася опираючись на дослідженні вирощування розсади помідора. В подальшому система може бути розширена та модернізована за рахунок програмного вибору типу рослини, ґрунту, масштабу теплиці та потужності насосу. Буде створено базу даних рослин, ґрунтів, насосів і т. ін., де користувач зможе обирати необхідну йому комбінацію.

РОЗДІЛ 1
АНАЛІЗ ВІДОМИХ РЕЗУЛЬТАТІВ РОЗРОБКИ

1.1 Ціль розробки

Дана система базується на двох основних факторах, це вологість ґрунту та фаза росту рослини. В різні фази росту – рослині необхідна різна вологість ґрунту, окрім цього необхідно враховувати ріст самої рослини, а саме її кореня. Від самого моменту посадки насіння в ґрунт, рослина розвивається та її корінь все більше занурюється та займає все більшу поверхню у ґрунті. Тому постає питання в надходженні необхідної кількості вологи на певну глибину в залежності від розвитку самої рослини. Для цього необхідно було створити математичну модель просочування, насичення вологи у ґрунті та модель приблизного співвідношення глибини на якій знаходиться корінь від розвитку рослини.

Основна ціль наукової роботи – це створення та дослідження автоматичної системи поливу за допомогою нечіткої логіки (Fuzzy logic) яка зможе, в залежності від фази розвитку, забезпечити рослину необхідною кількістю вологи на певній глибині кореня.

1.2 Аналіз існуючих розробок

Розробка систем розумного зрощування ґрунтів це одна з найбільш затребуваних галузей розвитку сільського господарства. З початку 2018-го року в цю сферу було інвестовано більше двадцяти шести мільйонів доларів, основна частина яких припала на США.

Основна ціль створення систем розумного зрощування – це збільшення ефективності та економічності виробництва за рахунок економії ресурсів води та електроенергії. Системи допомагають аграріям контролювати витрати води за рахунок аналізу вологості ґрунту, стану росту та розвитку рослини. Серед

найбільш перспективних розробок у цій галузі є розробки: Hortau, Smart Farm Systems, HydroPoint Data Systems.

- Система моніторингу вологи ґрунту Hortau [1]

Дана розробка є творінням учених-агрономів та сертифікованих фахівців по зрощуванню. Система надає фермерам інформацію про стан ґрунту на полях в режимі реального часу, та передає до мобільного веб-додатку. На основі отриманої інформації можна приймати рішення про застосування зрошення і добрив. Інформація зчитується за допомогою датчиків натягу ґрунту

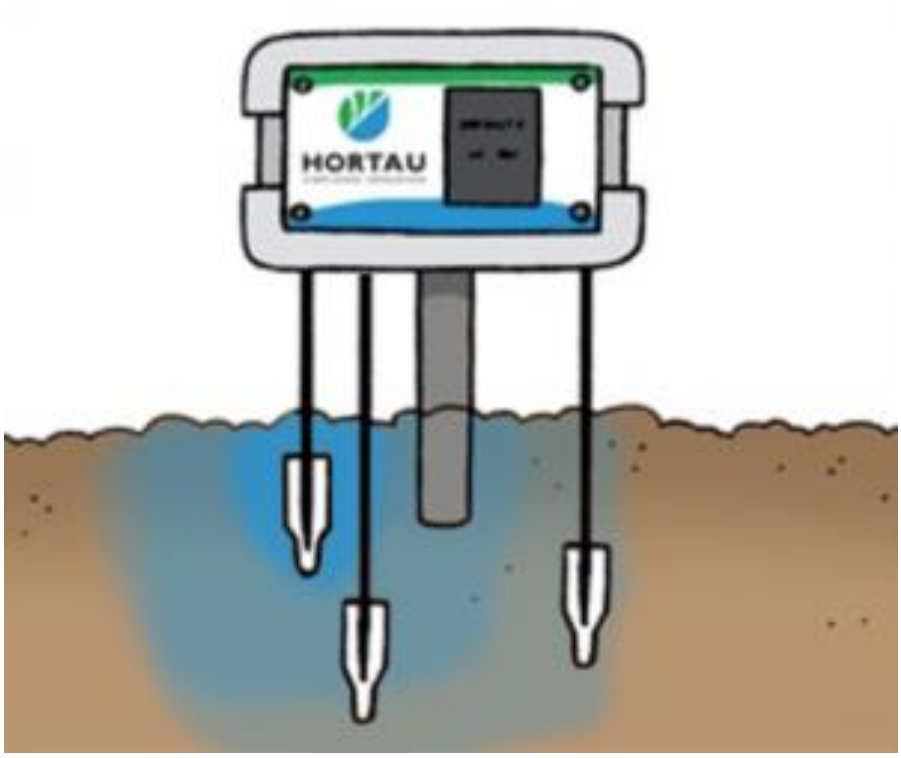


Рисунок 1.1 – Структурно функціональна схема відображення моніторингу стану ґрунту системою Хортау

Натяг ґрунту вимірює силу всмоктування, необхідну для поглинання води з ґрунту. Ця сила однакова незалежно від типу ґрунту.

Переваги вимірювання вологості ґрунту за рахунок натягу ґрунту з використанням датчика Хортау є:

- Натяг однаковий для будь-якого типу ґрунту;

- Калібрування не потрібно;
- Пряме вимірювання наявності води;
- Висока точність вимірювання.

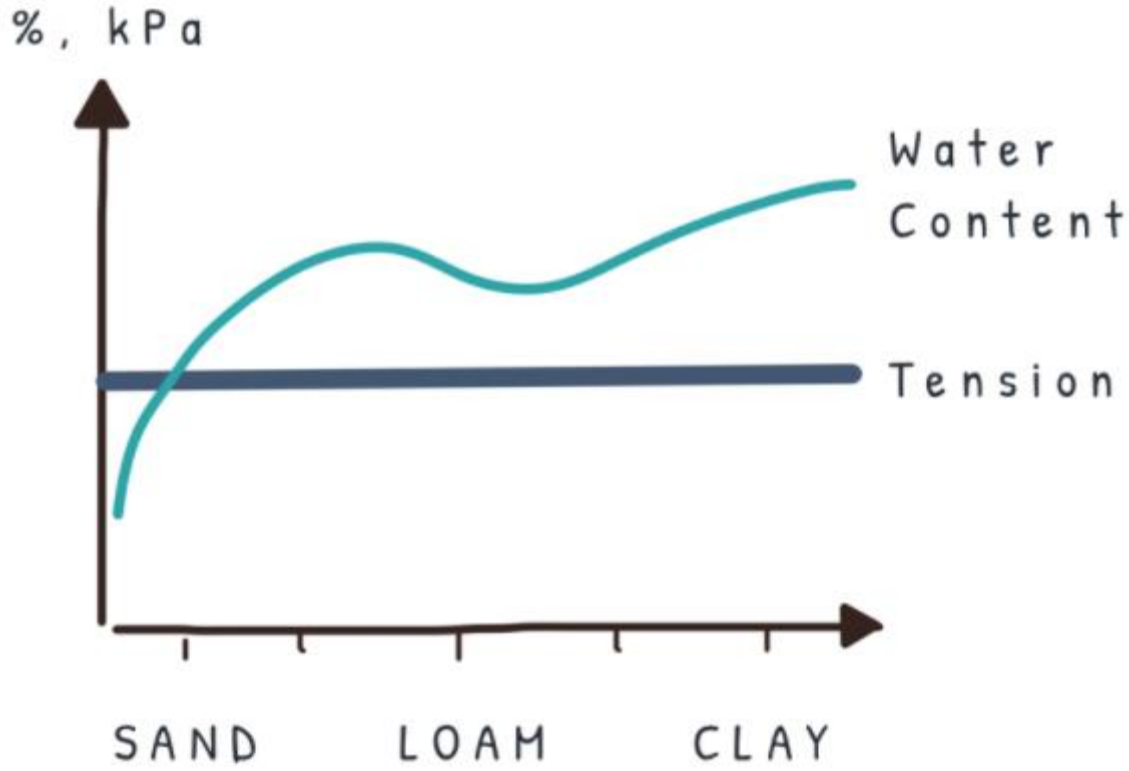


Рисунок 1.2 – Графік залежності зміни вологості ґрунту від виду ґрунту та сили натягу

- Автоматичний консультант по зрощенню Tule [2]

Розробники системи Tule позиціонують її як автоматичного консультанта по зрощенню 24/7. Система надає рекомендації по зрощенню для конкретної ділянки які засновані на попередньо поставлених фермером цілях. Основними функціями розробки є:

- Передбачення загрози врожайності та якості рослин;
- Моніторинг та можливість керування великою кількістю акрів з меншими витратами часту та робочої сили;
- Рекомендації по зрощенню для конкретної ділянки досліджуваної території;
- Масштабність, один датчик вимірює площу до 10 акрів;

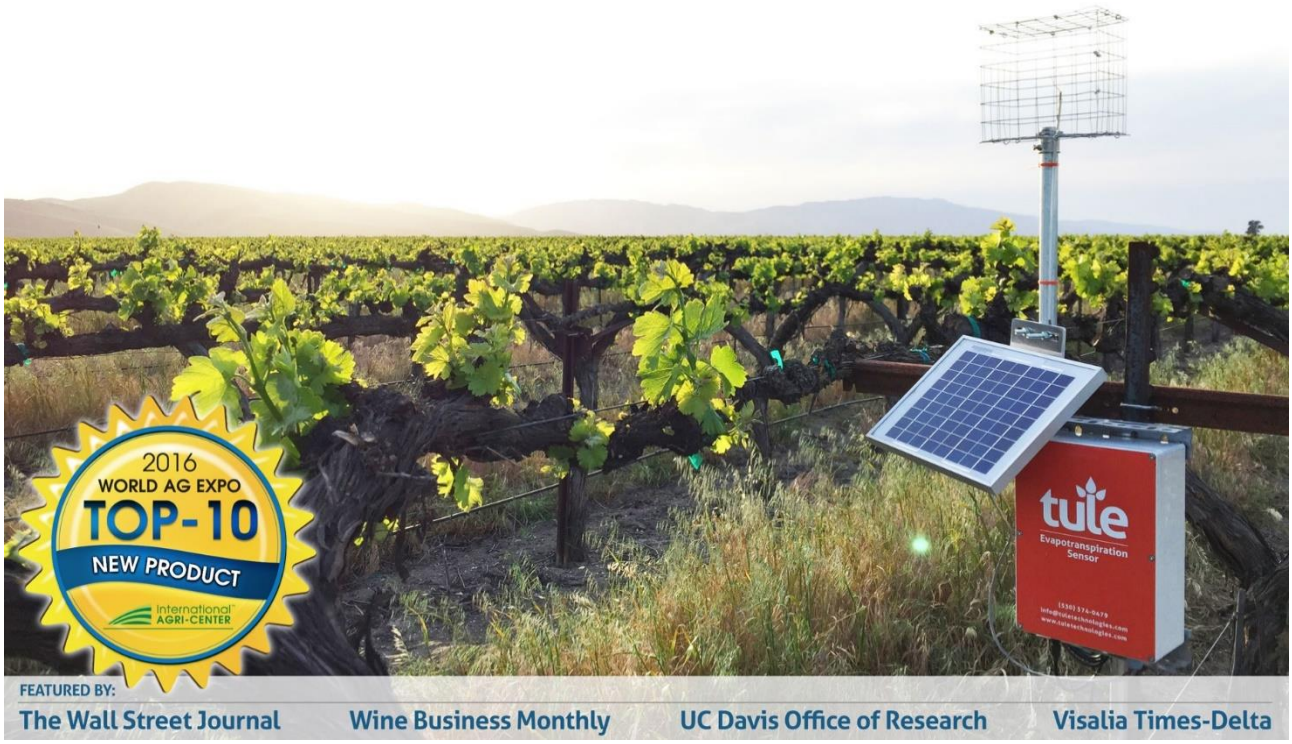


Рисунок 1.3 – Відображення однієї з багатьох можливих метеорологічних станцій Tule у робочому середовищі

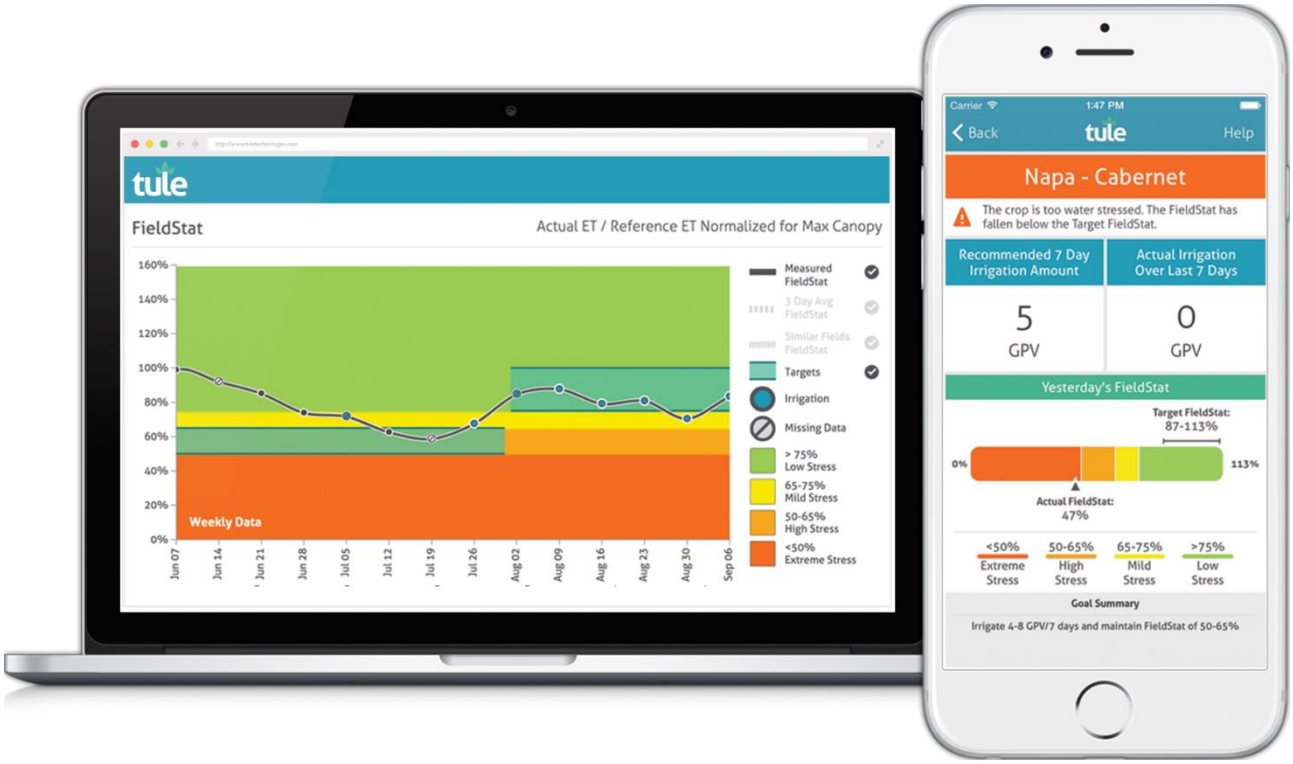


Рисунок 1.4 – Відображення отриманої інформації та її обробка для різного функціоналу у веб та мобільному додатку Tule

- Система автономного управління зрощенням Tevatronic [3]

Розробка оптимізує використання води та знижує затрати на технічне обслуговування. Набір бездротових датчиків, які допомагають фермерам віддалено оцінювати ситуацію на своїх полях і теплицях. Ці дані та оповіщення в режимі реального часу допомагають фермерам швидше реагувати на правильне зрошення і економити ресурси, коли в цьому немає необхідності.

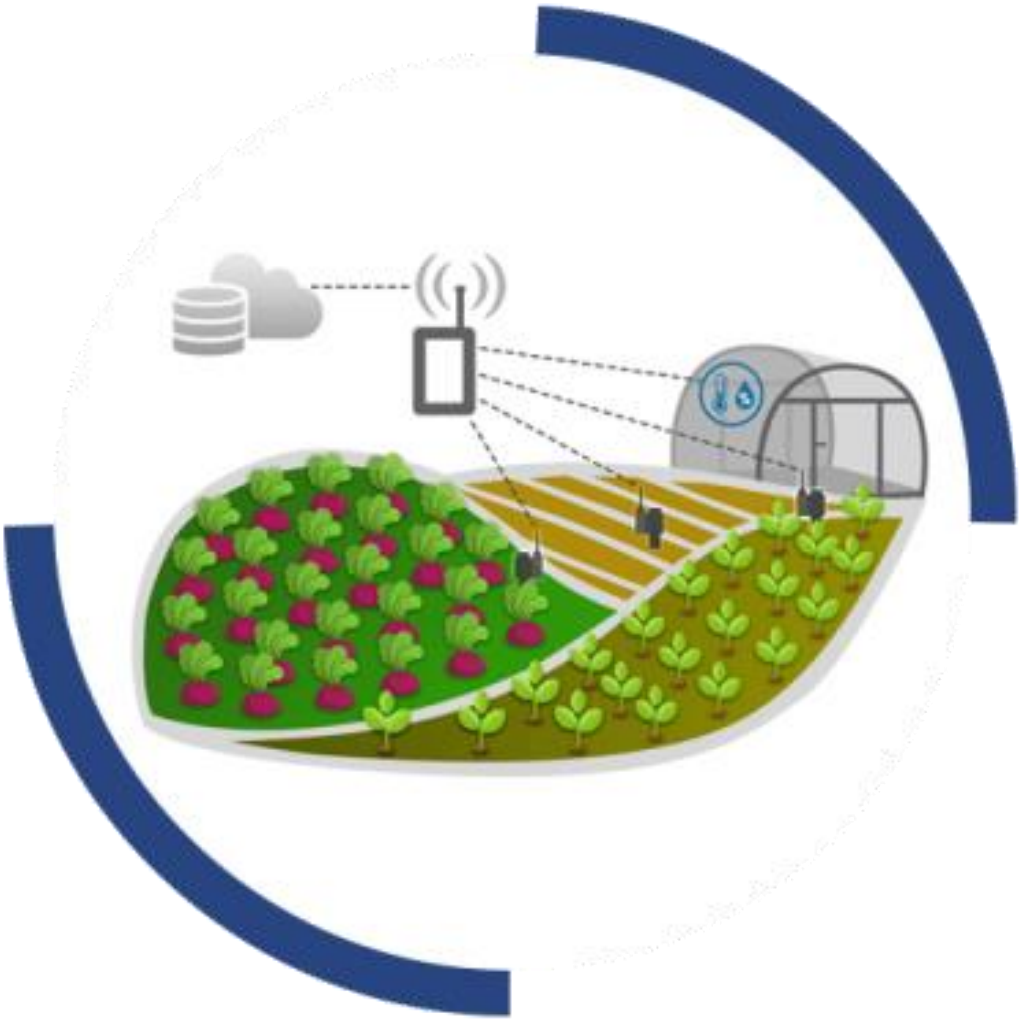


Рисунок 1.5 – Загальний вигляд моніторингу та передачі інформації між модулями системи

Tevatronic розробляє різні бездротові датчики, призначені для використання на відкритому повітрі на полях, в садах і теплицях. На основі даних в режимі реального часу фермери можуть коригувати обсяги і строки

поливу відповідно з вимогами до урожаю.

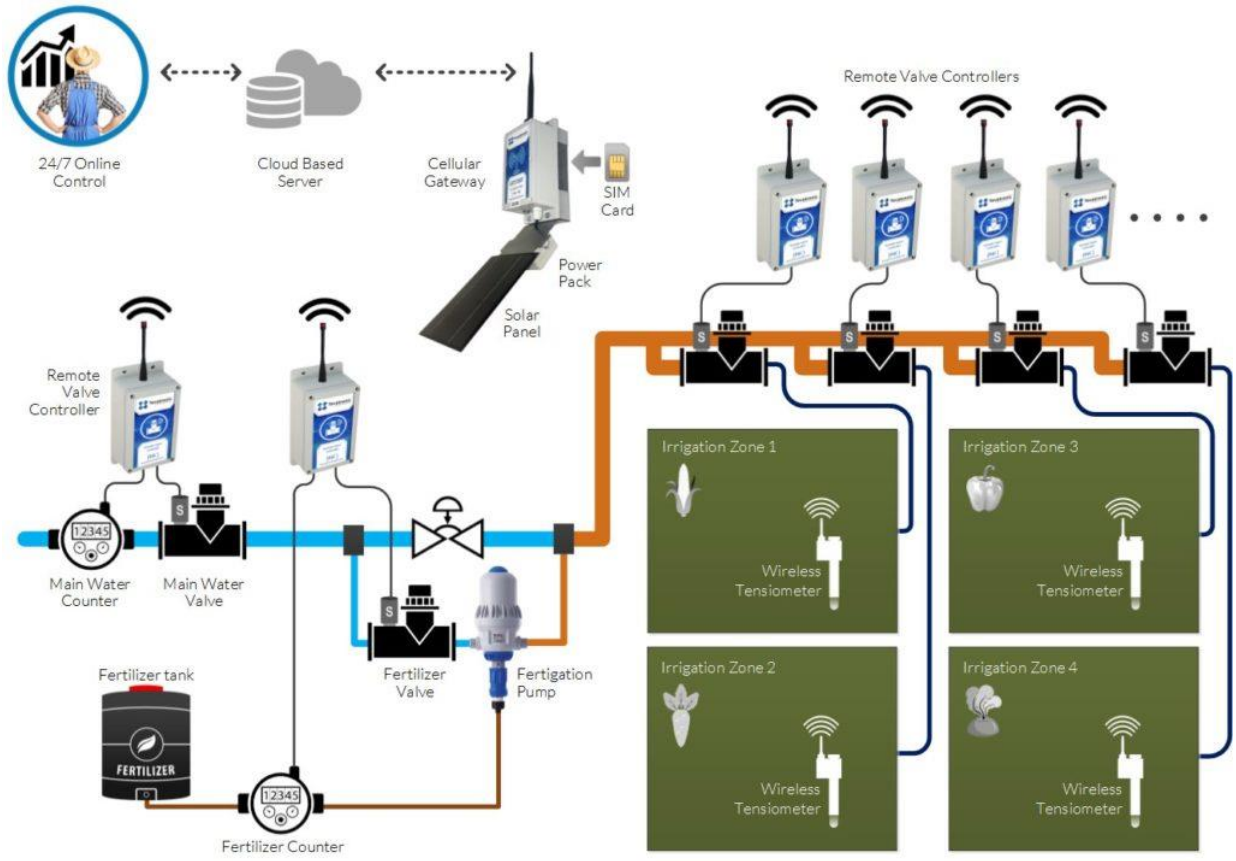


Рисунок 1.6 – Архітектура системи Tevatronic

Отримана інформація через кабельне з’єднання або за допомогою бездротового з’єднання з датчиків передається до веб-сайту та на додаток смартфону де може відображатися у вигляді графіків чи таблиць. Програма аналізує отриману інформацію та інформує користувача якщо показники з датчиків є вище або нижче виставлених меж необхідних для ефективного росту рослини.

Розробка здатна не тільки проводити моніторинг стану ґрунту і проводити аналіз отриманої з датчиків інформації, а ще й має зворотній зв’язок у керуванні зрощуванням ґрунту. Система може відкривати та перекривати гідравлічний клапан подачі води у систему зрощування. В залежності від масштабу та площі яку моніторить система може бути більше одного клапану подачі води, даними клапанами можна керувати як разом так і окремо.

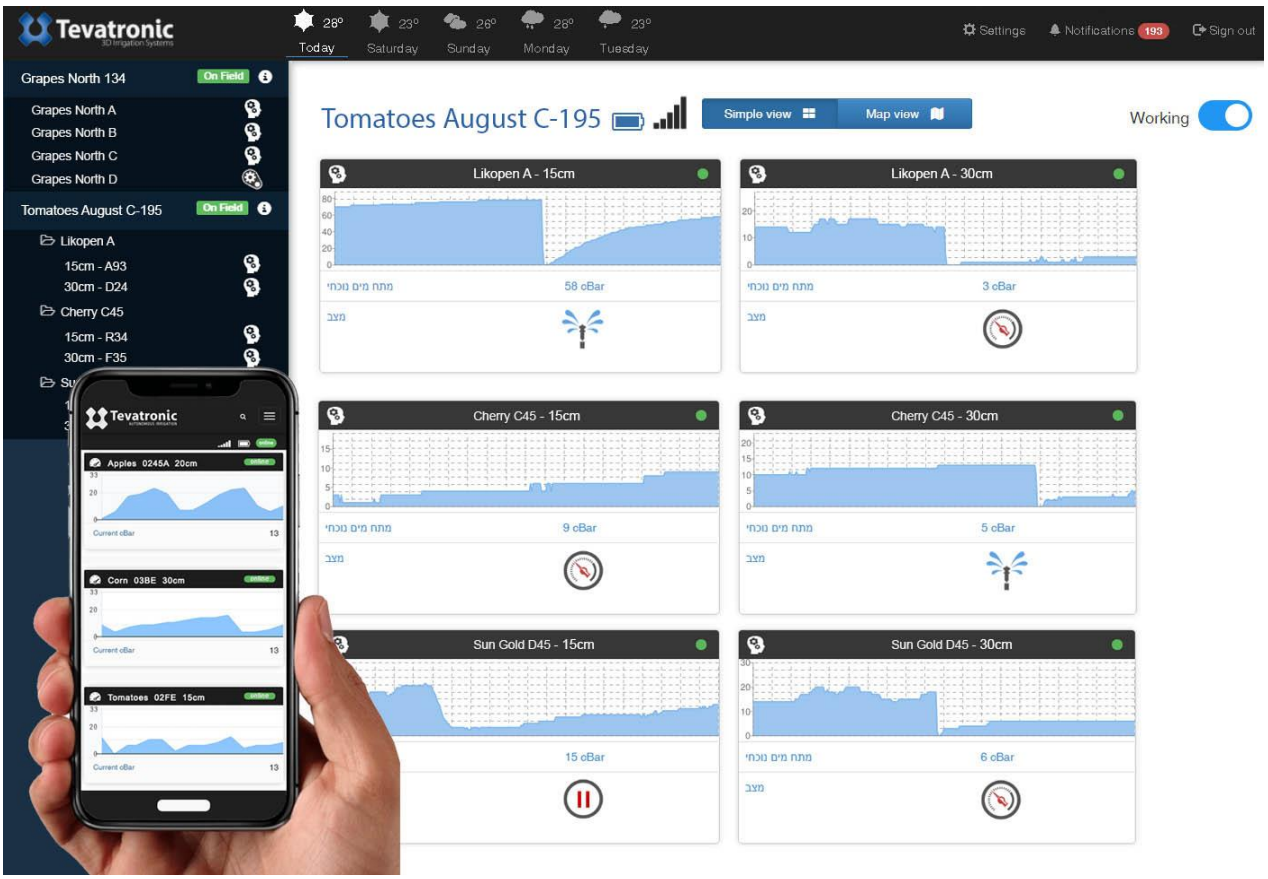


Рисунок 1.7 – Відображення отриманої інформації та її обробка для різного функціоналу у веб та мобільному додатку Tevatronic

Головним недоліком майже всіх перелічених вище розробок є відсутня можливість автоматичного керування поливом ґрунтів. Розглянуті системи виконують багато корисних функцій але виконують лише допоміжну функцію у вирощуванні рослин, полив ґрунту можливий тільки за командою користувача, керування за допомогою нечіткої логіки не використовується взагалі. Окрім цього в більшості досліджених систем не використовується врахування типу рослини та її віку (доби) вирощування. Це є великим недоліком, тому що в залежності від типу рослини і її фази росту залежить якість рослини та врожаю в цілому.

1.3 Методи вимірювання вологості в грунті

У сучасному світі актуальне використання доволі різних за принципом

роботи методів вимірювання вологості ґрунту, які наведено в табл. 1.1. [4-9]

Таблиця 1.1 – Методи вимірювання вологості ґрунту

Метод	Описання методу
Гравіметричний метод	засновується на відокремленні води зі зразка шляхом випаровування, вимивання або хімічної реакції.
Нейтронне розсіювання	середнє значення втрати енергії або термалізація нейтронів значно вищі, коли нейтрони стикаються з атомами з низькою атомною вагою, ніж коли з важчими атомами. В ґрунті атоми з малою вагою представлені переважно воднем. В результаті водень уповільнює швидкі нейтрони значно ефективніше, ніж будь-який інший елемент у ґрунті.
Згасання гамма-випромінювання	вимірює вологість ґрунту в шарі 1-2 см. Ступінь зниження інтенсивності гамма-променів під час їх проходження через ґрунт залежить від складу ґрунту та його щільності. Розсіяння і поглинання
Згасання гамма-випромінювання	вимірює вологість ґрунту в шарі 1-2 см. Ступінь зниження інтенсивності гамма-променів під час їх проходження через ґрунт залежить від складу ґрунту та його щільності. Розсіяння і поглинання гамма-променів залежить від щільності речовини, через яку вони проходять.
Електромагнітний метод	заснований на впливі вологості на електричні властивості ґрунту.
Тензометричний метод	прилади здатні фіксувати зміни вологоємності ґрунту, що є наслідком інфільтрації води, поливу, випаровування та транспірації. Також даний метод називають методом вимірювання натягу ґрунту.
Мікрохвильовий метод	випромінювання тепла поверхнею ґрунту в мікрохвильовому діапазоні визначається дистанційно відповідними вимірювальними

Продовження таблиці 1.1

Мікрохвильовий метод	приладами – пасивними (радіометричними) або активними (радар) методами
Ядерно магнітний резонанс	заснований на здатності резонансу виявити концентрацію атомів водню і, відповідно, вологи в ґрунті.
Термічний метод	заснований на зв’язку теплової інерції ґрунту та його вологості.
Рефлектометричний метод	вимірює час проходження електричного імпульсу по кабелю, який залежить від електропровідності, а, отже, вологості, ґрунту.
Оптичний метод	базується на тому, що за наявності вологи на поверхні світло, відбите від неї, поляризується
Діелькометричний	оснований на залежності діелектричних властивостей матеріалу від вологості. Оскільки для сухих речовин діелектрична проникність є = 2,0...5,0, а для води $e_в$= 81,0, то невелика зміна вологості матеріалу призводить до значної зміни результатної діелектричної проникності. Як вимірювальні кола в ємнісних вологомірах найчастіше використовуються трансформаторні мости з тісним індуктивним зв'язком плеч, а також резонансні вимірювальні кола.

Для проведення подальших досліджень та розробки системи було обрано діелькометричний метод. Датчики на його базі цього методу дають змогу вимірювати вологість ґрунту безконтактно.

Ємнісні датчики працюють шляхом швидкого зарядження і розрядження позитивного і негативного електрода (конденсатора) в ґрунті, створюючи електромагнітне поле, час заряду якого залежить від ємності ґрунту. Ємність пов'язана з діелектричної проникністю середовища між електродами конденсатора. На діелькометричні датчики впливають вологість ґрунту, температура і електропровідність. Частота вибирається для вимірювання вологості ґрунту або масової електропровідності. Основною особливістю таких сенсорів є обов'язкове калібрування на місці проведення вимірювань для

зменшення похибки вимірювання об'ємного вмісту води в ґрунті [10-11]. У даній роботі авторами використано типову модель ємнісного сенсору вологості ґрунту [12], який конструктивно та програмно сумісний із бюджетними мікропроцесорними платформами. [13-15], його характеристики наведено в табл. 1.2.

Таблиця 1.2 – Характеристики використовуваного датчика вологості DFRobot

Метод вимірювання вологості	Ємнісний
Тип виходу	Аналоговий
Напруга живлення	Від 3,3 В до 5,5 В
Вихідна напруга	Від 1,2 В до 2,5 В
Інтерфейсний роз’єм	PH2.0-3P
Розмір	98x23 мм
Вага	15 грам

Рисунок 1.8 – Ємнісний датчик вологості ґрунту від DFRobot

1.4 Програмне та матеріальне забезпечення

Перелік та характеристики датчиків, якими буде здійснюватися зчитування стану ґрунту, було обрано і охарактеризовано в попередньому пункті (п.1.3). Це

діелькометричний датчик вологості ґрунту торгової марки DFRobot, кількість датчиків залежить від площі землі яка зайнята для вирощування рослин.

Мікроконтролер для опрацювання інформації і керування системою

Мікропроцесорний контролер (далі МПК) являє собою плату з вхідними та вихідними пінами. Вони призначені для підключення датчиків та додаткового облаштування. Окрім цього МПК може опрацьовувати вхідну і вихідну інформацію. У якості МПК було обрано контролер Arduino UNO.

Arduino UNO – контролер побудований на ATmega328, платформа має 14 цифрових входи/виходи (6 з котрих можуть використовуватися як виходи ШІМ), 6 аналогових входів, кварцовий генератор 16 МГц, роз'єм USB, силовий роз'єм, роз'єм ICSP та кнопку перезавантаження. Мікроконтролер ATmega328 має 32кБ флеш пам'яті, з котрих 0.5 кБ використовується для збереження завантажувача, а також 2 кБ ОЗУ (SRAM) і 1 кБ EEPROM. [18]

До контролера будуть підключені датчики (див. п. 1.3) а також насос яким буде керувати система.

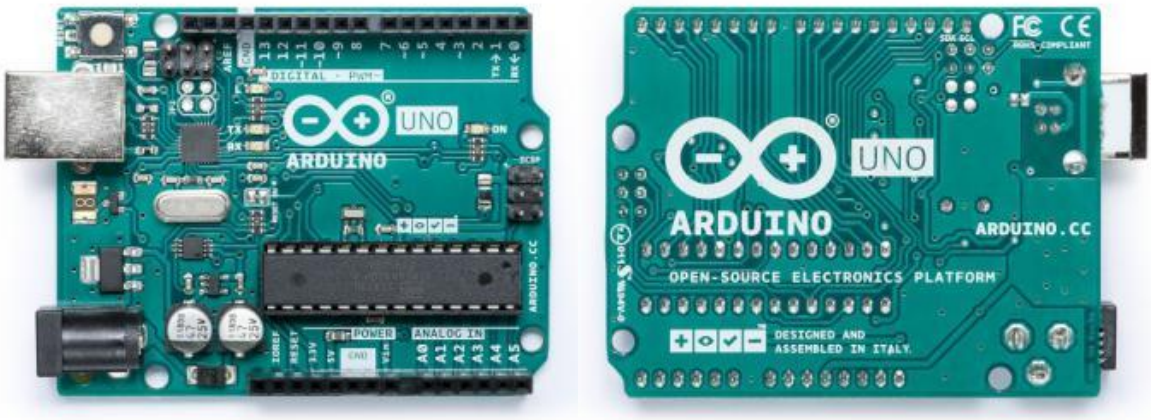


Рисунок 1.9 – Контролер Arduino UNO

Модуль реального часу

Даний модуль необхідний для обліку хронометричних даних (поточний

час, дата, день та ін.). Він представляє собою систему з враховуючого пристрою і автономного джерела живлення.

Модуль DS3231, можна сказати, є звичайним годинником. У МКП Arduino є заздалегідь вбудований датчик часу Millis, але він залежить від наявності постійного живлення. При відключенні живлення плати Arduino відлік часу Millis скинеться до нуля. Модуль DS3231 на відміну від МКП має зовнішню, змінну, літієву батарейку, за допомогою якої, навіть при відключенні живлення плати Arduino, продовжує «живити» модуль, дозволяючи йому вимірювати час. [16]

Рисунок 1.10 – Модуль реального часу DS3231

- Модуль проводить підрахунок годин, секунд, дат, місяців, років (високосні роки враховуються до 2100 року)
- Для підключення до різних пристроїв, модуль підключається по I2C інтерфейсу

1.5 Програмне забезпечення

Програмне забезпечення (ПЗ) потрібне для проведення обчислень, створення теоретичної моделі, пошуку недоліків та аналізу працездатності проекту до створення реального макету, написання коду та його завантаження в мікроконтролер.

До списку необхідних програм і джерел для створення системи входять:

Таблиця 1.3 – Програмне забезпечення для створення системи

Назва програми або джерела	Функція
MATLAB&SIMULINK	Проектування бази «нечітких» правил
	Проектування моделі системи та тестування її працездатності
Сайт онлайн компіляції	Перетворення Matlab документу в програмний код Arduino IDE
Arduino IDE	Написання коду з урахуванням підключення усього необхідного облаштування для роботи реальної системи

Пакет програмного забезпечення MATLAB & SIMULINK

Метою магістерської роботи є створення електронної системи контролю вологості тепличних ґрунтів на базі нечіткої логіки, тому для створення теоретичної моделі буде використано програмне забезпечення для створення цієї бази. Нечітка логіка, або Fuzzy logic, надає можливість створити систему багатьох різноманітних варіантів керування системою. Логіка складається з блоків вхідних та вихідних сигналів а також бази правил керування вихідним сигналом. За допомогою пакету ПЗ MATLAB&SIMULINK та вбудованного розширення FuzzyLogicToollbox буде створена автоматична та повністю автономна система моніторингу та керування поливом ґрунту. Дана система залежить від двох вхідних параметрів, що поділені на певну кількість вхідних функцій вологості ґрунту та фази росту рослини, вираховується в днях, а також певної кількості рівнів вихідних функцій керуванням насосом що подає воду в систему поливу. Рівні поділяються згідно з державними нормами вирощування сільськогосподарських культур в теплиці. [17]

Програма Arduino IDE

Програмне забезпечення Arduino це інтегроване середовище розробки програмного коду яке складається з вікна повідомлень, області виводу тексту (консолі) та панелі інструментів. Для того щоб завантажити програмний код до МКП або просто під'єднати плату до Arduino IDE необхідно скористуватися COM портом.

Програмний код, написаний ПЗ Arduino, називається скетч. Скетч створюється в текстовому редакторі та з ним можна проводити наступні операції: вирізка / вставка, пошук / заміна тексту. Створений та збережений програмний код зберігається у файл формату «Arduino file», цей файл у подальшому може бути використовуваний для конвертації у інші формати або для завантаження у МКП. [18]

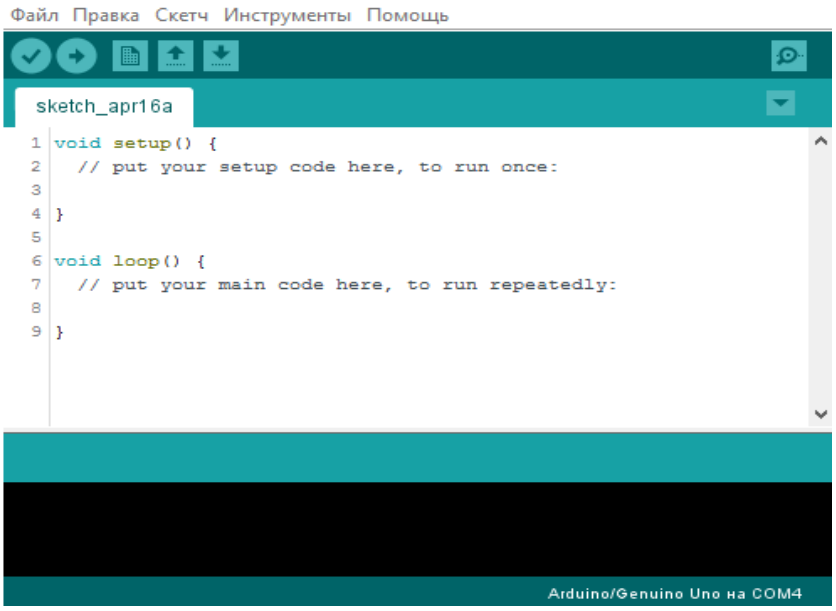


Рисунок 1.11 – Програма Arduino IDE

1.6 Дослідження розвитку та росту рослин

У якості досліджуваної рослини, за допомогою якої буде проводитись дослідження та практичний експеримент, було обрано доволі популярну в

Україні рослину - томат.

Сорти томатів можна розділити на три групи за характером їх росту, які розпізнаються по розташуванню і частоті листя і суцвіть на стеблі.

Фази росту томату, в дуже загальних рисах, можна розділити на чотири періоди [19]:

Таблиця 1.4 – Періоди фаз росту рослини

Фаза росту	Опис фази
Проростання	Насіння томатів проростають на 6-8 день після посадки. Оптимальною температурою для проростання насіння - вважається 20-25 градусів тепла. У цю фазу з насіння над землею з'являються 2 сім'ядольних листочки. Вони розташовані один на проти одного. Корінець рослини в цій фазі дуже тонкий, на ньому розташовані мікро волосинки для всмоктування поживних речовин.
Ріст	Рослина посилено починає рости. Перші 2, справжніх, листочка з'являються на 11-14 день. У цей час корінь рослини подовжується і нагадує білі тонкі ниточки. З'являються виражені бічні корені. У цій фазі необхідно провести перші підгодівлі томатів.
Цвітіння	Перше суцвіття з'являється через два з половиною місяці після посіву. Інші суцвіття з'являються над першим, між двома суцвіттями, з різною кількістю листків. Цвітіння, як правило, починається від низу до верху. У цій фазі проводяться обов'язкові підживлення рослин.

Продовження таблиці 1.4

Дозрівання та плодоношення	Це одна з найсерйозніших фаз у розвитку томатів. У цій фазі томатам потрібна значна кількість поживних речовин, тому обов'язково необхідно підживлювати рослину.

Ці періоди зростання також відображають різні потреби рослини в харчуванні. Тривалість кожної стадії може варіюватися залежно від способу вирощування, сортових характеристик і кліматичних умов:

Таблиця 1.5 – Тривалості стадій розвитку помідора від зерна до стиглого плода

Спосіб вирощування	Теплиця	
Кількість днів до першого цвітіння	30	
Кількість днів до першого збору врожаю	65	
Стадія зростання	**Тривалість етапу (днів)**	**Вік рослини (днів)**
Посадка	1	1
Вегетативна	14	15
Перше цвітіння	15	30
Перше зав'язування плодів	10	40
Стадія зростання	**Тривалість етапу (днів)**	**Вік рослини (днів)**
Ріст плодів	20	60
Початок збору врожаю – до кінця останнього збору врожаю	21-145	81-210

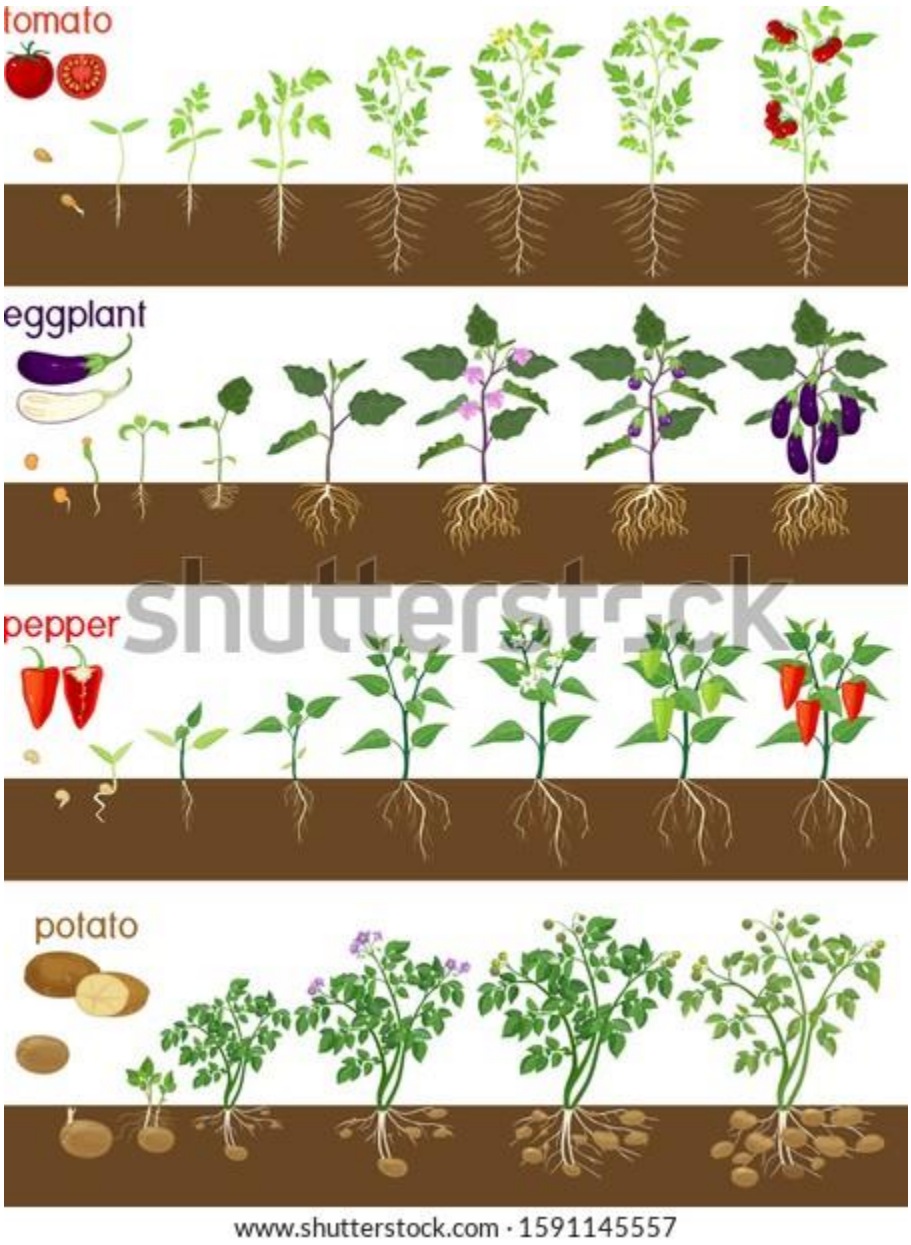


Рисунок 1.12 – Загальний вигляд розвитку рослин в залежності від фази росту

Після зав'язування плодів плоди дозрівають протягом 45-70 днів, в залежності від сорту, клімату і умов виростання. Плоди продовжують рости до стадії зеленої стиглості та визначається три стадії розвитку плода.

Дозрівання відбувається по мірі того, як плоди змінюють колір зі світло-зеленого на білуватий, рожевий, червоний і, нарешті, темно-червоний або помаранчевий. Залежно від тривалості часу необхідного для транспортування плодів до місця продажу або до споживача збір врожаю може відбуватися в будь-який час від рожевого до темно-червоної стадії, на більш пізніх стадіях

утворюються більш ароматні плоди.

Рисунок 1.13 – Відображення поступового розвитку рослини, помідора, від зерна до плодоносної рослини

Вирощування томатів. Метод вирощування томатів.

Томати можуть вирощуватися в грунті або без нього, в теплицях або просто неба. Помідори можна вирощувати на ґрунтах з широким спектром текстур, від легких піщаних ґрунтів до важких глинистих ґрунтів. Піщані ґрунти краще обирати, якщо необхідно отримати ранній урожай. Сприятливий рівень pH: 6,0-6,5. При більш високому або більш низькому рівні pH мікроелементи стають менш доступними для засвоєння рослинами. [20-21]

Температура є одним із основних факторів, що впливає на всі стадії розвитку рослини: вегетативний ріст, цвітіння, зав'язування плодів і дозрівання плодів. Для росту потрібна температура від 10°C до 30°C.

Таблиця 1.6 – Вимоги до температури на різних стадіях росту:

Стадія зростання	Температура, °C		
	Мінімальна	Максимальна	Оптимальна
Проростання	11	34	16-29

Продовження таблиці 1.6

Вегетативний ріст	18	32	21-24
Розвиток плодів (ніч / день)	10/18	20/30	13-18/19-24
Формування лікопіну	10	30	21-24
Формування каротину	10	40	21-32

Рисунок 1.14 – Процес проростання та первинного розвитку томату

Інтенсивність світла є одним з основних факторів, що впливають на кількість цукрів, що утворюються в листках під час фотосинтезу, а це, в свою чергу, впливає на кількість плодів, які рослина може створити, і на загальний урожай.

Зрошення

Рослини томатів досить стійкі до помірної посухи. Однак для забезпечення високого врожаю та якості росту необхідне належне керування вологістю ґрунту.

Потреба у воді томатів, вирощуваних на відкритому повітрі, коливається в

межах 4000 - 6000 м3/га. В теплицях потрібно до 10 000 м3/га води. 70% або більше кореневої системи знаходяться у верхніх 20 см ґрунту. Тому рекомендується крапельна система, оснащена пристроєм для фертигації.

На легких ґрунтах або при використанні солоної води необхідно збільшити кількість води на 20-30%. Потреби у воді будуть відрізнятися на різних стадіях росту. Потреба збільшується від проростання до початку дозрівання плодів, досягаючи піку під час розвитку плодів, а потім знижується під час дозрівання.

М'який стрес з водою під час розвитку і дозрівання плодів позитивно впливає на якість плодів: твердість, смак і якість при зберіганні, але може призвести до зменшення плодів. Пізній полив, близький до збирання врожаю, може погіршити якість і викликати гниття. Нестача води призведе до зниження зростання в цілому і зниження засвоєння кальцію зокрема. Дефіцит кальцію викликає гниль на кінці квітки. З іншого боку, надмірне зрошення створить анаеробні ґрунтові умови і, можливо, призведе до загибелі коренів, затримці цвітіння та порушення плодоношення.

Кисла (низький pH) поливна вода небажана, так як це може призвести до розчинення токсичних елементів у ґрунті (наприклад, Al3+).

Чутливість до дефіциту кальцію

Помідори дуже чутливі до дефіциту кальцію, який проявляється симптомом гнилі в кінці цвітіння на плодах. Умови засолення різко підвищують інтенсивність забруднення. Нещодавно було виявлено, що марганець (Mn) служить антиоксидантом в плодах томатів, тому його застосування до помідорів, вирощених в умовах солоності, може полегшити симптоми загнивання в плодах. Необхідно проявляти особливу обережність, щоб уникнути умов для зростання, які посилюють це явище. Стосовно якості води, то помідори переносять солонувату воду з електропровідністю близько 2-3 мм рт. ст./см.. [22-23]

1.7 Висновки

1. Згідно проведеного аналізу існуючих розробок було виявлено їх основні переваги та недоліки. Серед основних недоліків була відсутність використання системи заснованій на базі нечіткої логіки та відсутність врахування налаштування системи на типи рослин.
2. На підставі проведеного аналізу було обрано метод вимірювання вологості ґрунту, а саме діелькометричний метод та підібрано датчики які вимірюють вологість за обраним методом.
3. Згідно з попередньо проведеного аналізу та подальшої розробки системи було обрано програмну та матеріальну базу. За допомогою програмної бази буде створено математичну модель динаміки вологості у грунті, базу правил нечіткої логіки, програмний код, імітаційну та реальну моделі. За допомогою матеріальної бази буде створено реальну досліджувану модель системи.
4. Визначено вид досліджуваної рослини, а саме помідор. Проведено загальний аналіз фаз росту рослини, тривалість стадій розвитку помідора від зерна до стиглого плода, метод та умови вирощування помідора, сформовано цілі для подальшого дослідження росту рослини та її кореня у грунті.

РОЗДІЛ 2
РОЗРОБКА СТРУКТУРНОЇ СХЕМИ ТА МАТЕМАТИЧНОЇ МОДЕЛІ ЗМІНИ ВОЛОГОСТІ ҐРУНТУ

2.1 Алгоритм роботи та структурна схема системи контролю вологості тепличних ґрунтів

Структурна схема даної розробки у загальному вигляді побудована з використанням загальнонаукових підходів та містить у своєму складі серійні компоненті. Під поверхнею ґрунту розміщено датчики вологості, вони зчитують фактичну вологість ґрунту та передають інформацію до мікроконтролера. Мікроконтролер Arduino UNO, маючи завантажену в собі програму на базі нечіткої логіки, оброблює інформацію за певним алгоритмом і видає керуючий сигнал увімкнення насосу на певний час. Час увімкнення насосу еквівалентний об’єму води який необхідно вилити під рослини для забезпечення найліпшого відсотка вологості ґрунту з урахуванням віку (доби) рослини.

Окрім вхідного сигналу від датчиків вологості вхідними даними для керування системою є вік рослини, її орієнтирна фаза розвитку та глибина розташування основної маси кореня.

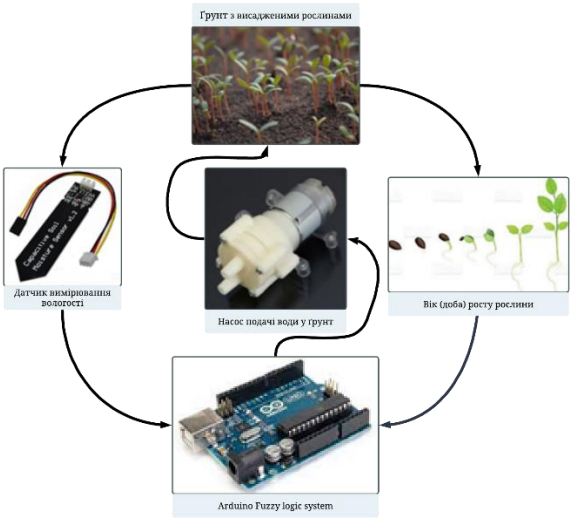


Рисунок 2.1 – Загальна структурна схема системи контролю вологості ґрунту

2.2 Дослідження та прогнозування глибини росту кореня в залежності від віку (доби)

Для отримання реальних даних з дослідження росту кореневої системи рослини в залежності від віку, від замочування насіння до готової розсади, необхідно провести експеримент. Дослідження буде проводитись з використанням насіння помідора сорту «Мобіл» середньоранній.

Пророщене насіння буде посаджено в спеціальні пластикові касети для розвитку розсади., касета поділяється на комірки розміри яких: висота – 70 мм, ширина та довжина – 65 мм (див. рис. 2.2).

Отримані результати дослідження зображено у таблиці 2.1 з відповідним співвідношенням віку (доби) до довжини кореня.

Дослідження проводилося в домашніх умовах з кімнатною температурою 17-20 ℃ та відносною вологістю повітря 74%. Згідно до таблиці 2.1 було сформовано графік глибини росту рослини від доби. Дана залежність, в подальшому, допоможе розробці математичної моделі зрощення ґрунту на різній глибині [24-27].

Рисунок 2.2 – Касета з комірками що призначена для вирощування розсади

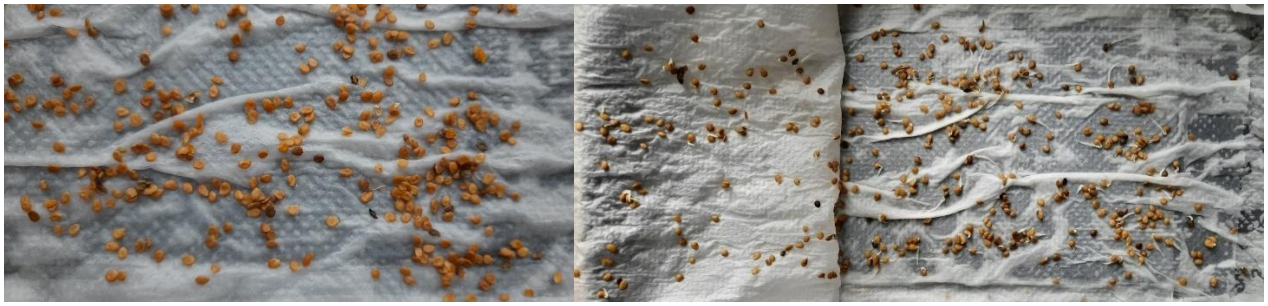


Рисунок 2.3 – Зображення дослідження поступового росту рослини

Таблиця 2.1 – Залежність глибини росту кореня від віку (доби) рослини

Вік (доба) рослини	Середня довжина кореня, мм	Середня глибина занурення кореня, мм
1-2	0	5
3-4	1	6
5-6	5	8
7-8	10	13
9-10	20	22
11-12	30	27
13-15	38	36
16-20	45	42

Продовження таблиці 1.6

20-25	65	60
25-30	78	70

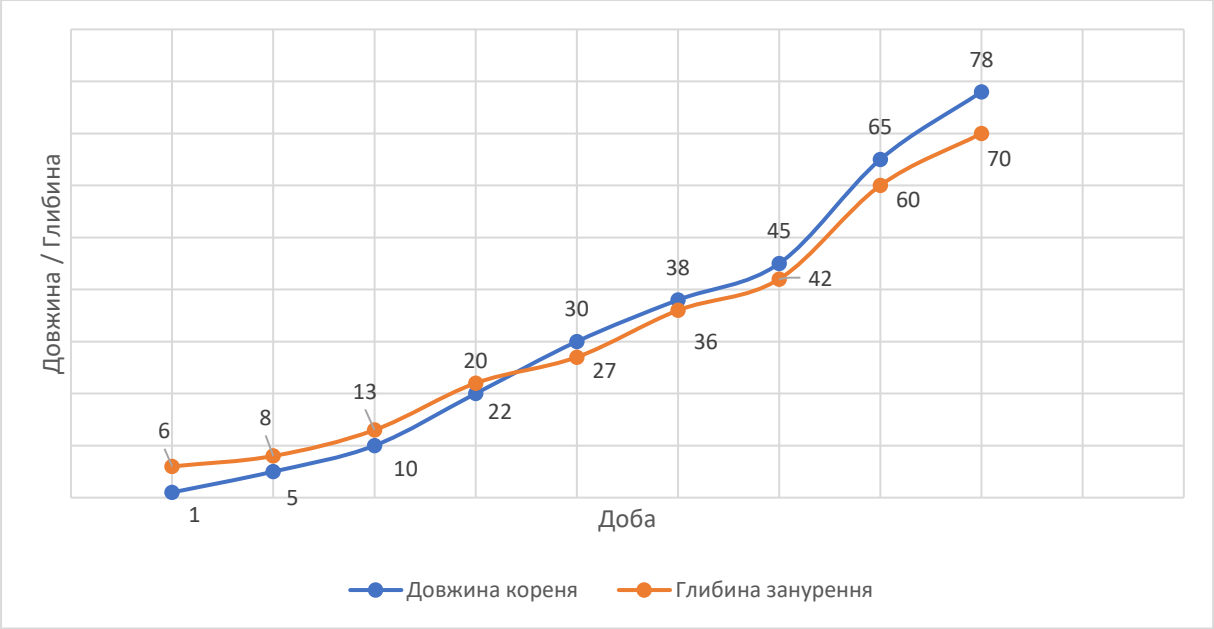


Рисунок 2.4 – Графічне зображення залежності глибини росту кореня від віку (доби) рослини

Аналізуючи отримані результати можна приступити до розрахунку математичної моделі динаміки вологи на глибині залягання кореня.

2.3 Розрахунок математичної моделі зміни вологості ґрунту

По-перше, передусім як приступити до розрахунку математичної моделі необхідно визначитися з принципами руху води у насиченому вологою грунті (фільтрації) [28].

Фільтрація – рух речовини у насиченому цією речовиною грунті. У відсутності спеціальних вказівок фільтрація – це рух води у насиченому водою грунті. Користаючись законом Анрі Дарсі, визначається потік вологи q_w в насиченому грунті пропорційний коефіцієнту фільтрації $K_Ф$ та градієнту гідравлічного напору $\left(\frac{\Delta h}{l}\right) q_W = K_Ф * \frac{\Delta h}{l}$, де гідравлічний напор Δh та довжина

колонки (l) мають однакові розмірності довжини; розмірності q_W та $K_Ф$ також однакові – (довжина/час), наприклад м/добу, см/добу і т.д.

Коефіцієнт фільтрації ($К_Ф$) – це здатність ґрунту проводити насичений потік вологи під дією градієнта гідравлічного тиску, зазвичай дорівнюючого одиниці.

Окрім фільтрації та коефіцієнта фільтрації слід враховувати здатність ґрунту на поглинання води, або, інакше кажучи, інфільтрації ґрунту. Це надходження води у ненасичений водою грунт під впливом градієнтів сорбційних та капілярних сил та гідравлічного напору. Поглинання води ненасиченим грунтом характеризують коефіцієнтом поглинання ($K_{пог}$). В загальному вигляді обидва вищезазначених процеси являють собою водопроникність, тобто спочатку інфільтрацію, насичення ґрунту вологою, а потім фільтрацію, рух води у насиченому вологою грунті. Основним значенням водопроникності є її використання при розрахунках переносу води в ненасичених вологою грунтах, і саме це необхідне для проведення розрахунку математичної моделі зміни вологості ґрунту.

Використовуючи модифіковані закони Дарсі, у ненасичених вологою ґрунтах важливим є коефіцієнт вологопровідності та визначення декількох важливих висновків:

- Для опису переносу вологи необхідно використовувати градієнт тиску вологи.
- Для кожного моменту розрахунку потоку вологи необхідно використовувати значення коефіцієнта вологопровідності
- Волога переміщується від зони високого тиску у зону низького, найчастіше це напрямок від поверхні у глибину, але, при деяких умовах, буває і навпаки.

Динаміка поширення вологи в ґрунті залежить від різних факторів які впливають на процес просочування та насичення вологою ґрунтів. Починаючи від геометричних параметрів та об’єму ємності в якій знаходиться ґрунт закінчуючи коефіцієнтом поглинання та фільтрації вологи грунтом. Пошарова

динаміка ґрунтової вологи описується диференційним рівнянням Річардса.

Опираючись на дисертаційну роботу Лактіонова І.С. [29] було створено теоретичну схему динаміки фізичних параметрів тепличних ґрунтів (див. рис. 2.5) та її моделювання у програмному забезпечені Mathcad. Як видно зі схеми, розрахунок може проводитись за трьома координатами, як разом, так і окремо. В залежності від того скільки осей задіяно в розрахунках модель може бути як одномірною, для одного напрямку (зазвичай для осі *z*), так і двох, трьохмірною. У випадку розрахунку трьохмірних моделей з'являється можливість розрахунку динаміки вологи в трьох координатах – *z, x, y*. В даному випадку для розрахунку моделі двомірною та трьохмірною моделями можна знехтувати, так як розміри досліджуваної комірки з посадженим насінням і, в подальшому, рослиною доволі невеликі.

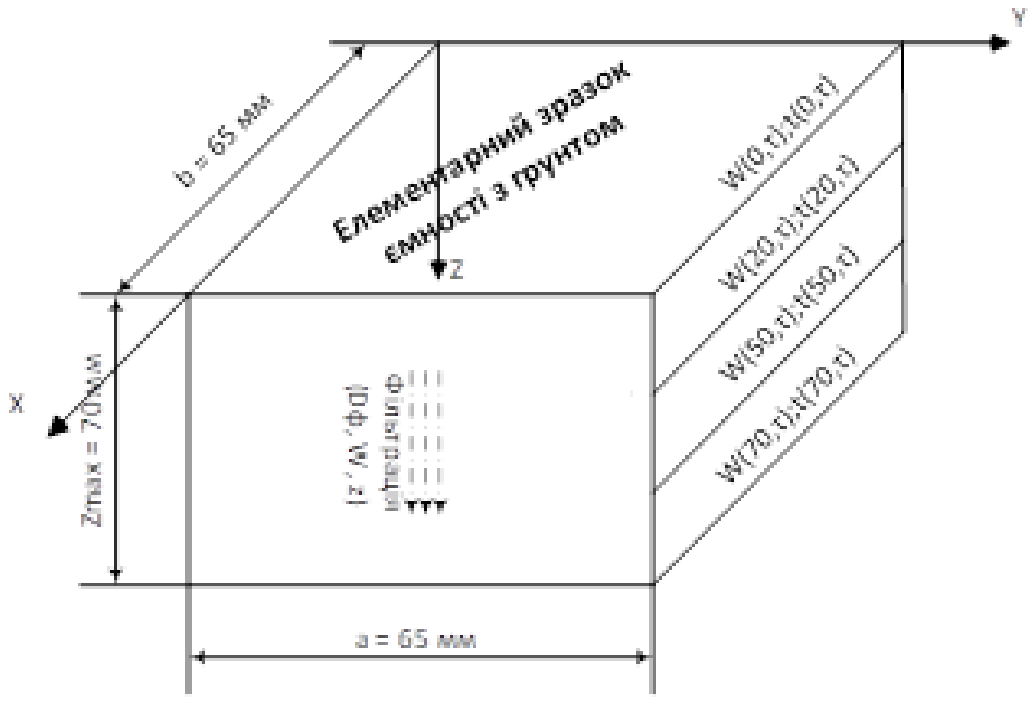


Рисунок 2.5 – Схема динаміки фізичних параметрів води в ємності з ґрунтом

Розрахунок моделі буде проводитись у межах від мінімальної довжини кореня, у момент посадки насіння у грунт, до максимальної довжини кореня розвиненої розсади рослини, у момент перед пересадженням окріплої розсади у

більшу теплицю чи у відкритий грунт.

Основними вхідними параметрами розрахунку даної математичної моделі буде:

- Глибина занурення кореня
- Коефіцієнт фільтрації
- Початкова вологість ґрунту на поверхні

Модель кінетики вологи ґрунтів описується наступним диференційним рівнянням у часткових похідних:

$$\frac{dW}{d\tau} = D_{\Phi} \frac{d^2W}{dZ^2} \quad (2.1)$$

де W – вологість ґрунту, %; τ – час, с; D_{Φ} – коефіцієнт дифузивності вологи, см2/с;, см2/с; z – відстань від верхньої межі до шару, що зондується, см.

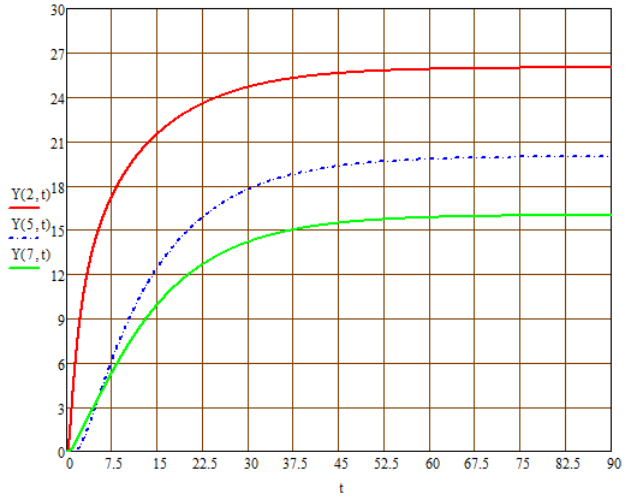
Використовуючи інформацію з таблиці 2.2 та використовуючи формулу (2.2) розрахуємо коефіцієнт дифузивності вологи для основних типів ґрунтів які найчастіше можуть використовуватися в теплицях та за різних значень початкової вологості верхнього шару ґрунту (W_{max}),%.

$$D_{\Phi} = \frac{K_{\Phi}}{W_{max}} \quad (2.2)$$

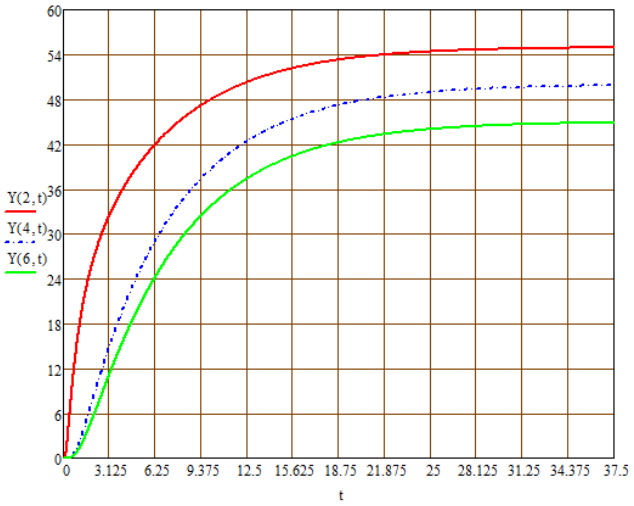
Таблиця № 2.2 – Оціночні значення параметрів D_{Φ} та K_{Φ}

№ з/п	Вологість верхнього шару ґрунту (W_{max}),%	D_{Φ}, $см^2/с$	K_{Φ}, см/с
1	30	0,82	0,031
2	60	1,64	0,062
3	90	2,46	0,092

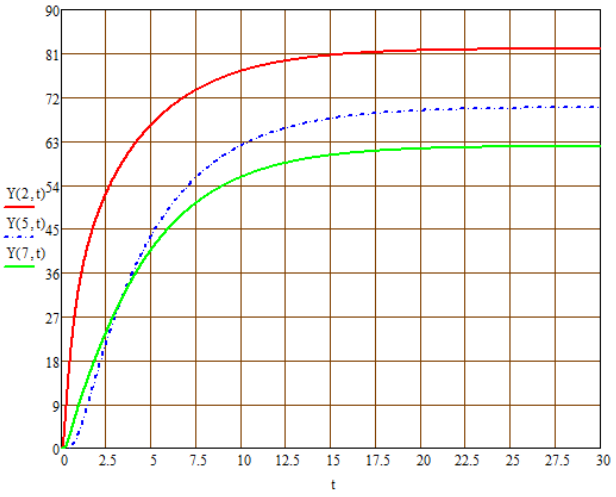
На підставі цих даних та за допомогою спрощеного диференційного рівняння кінетики вологи в ємності з тепличним ґрунтом отримано графік моделювання пошарової динаміки вологи (рис. № 2.5).



а) $W_{поч.}$= 30%



б) $W_{поч.}$= 60%



в) $W_{поч.}$=90%

Рисунок 2.6 – Результати моделювання динаміки вологи в ємності з тепличним ґрунтом (глибина: червоний – 2 см, синій – 5 см, зелений – 7 см)

Аналіз отриманих результатів моделювання динаміки поширення вологи відображено у таблиці № 2.3.

Таблиця 2.3 – Співвідношення початкової та залишкової вологості ґрунту на різній глибині в залежності від часу

Поч. вологість ($W_{поч.}$), %	Глибина (h), см	τ, с	Залишкова вологість ($W_{зал.}$), %
30	2	33,4	25
	5	40	19
	7	37,4	15
60	2	22,06	54
	5	25,03	49
	7	29,32	44,5
90	2	16,3	81
	5	18,5	69
	7	20,48	61,5

Виходячи з цих даних стає можливим:

- Вимірювання вологості ґрунту необхідно проводити не раніше, ніж через 1-2 хвилини, корективи до цього проміжку вимірювання також вносить час необхідний для доливання недостатньої кількості води;
- Прогнозування пошарової вологості ґрунту маючи дані лише з верхнього шару;

2.4 Розрахунок часу вмикання насосу від його потужності та початкової вологи ґрунту

Для розрахунку часу на який необхідно вмикати систему поливу необхідно розрахувати об’єм який є потрібним для насичення ґрунту на певний відсоток вологості. Для цього, в першу чергу, необхідно визначити скільки води

необхідно налити в грунт для підвищення вологості останньої на 1%. Склад вологи в грунті можна визначити за гравіметричним методом [30]:

$$(\%) = [\frac{\text{маса вологого грунту –маса сухого грунту(г)}}{\text{маса сухого грунту (г)}}] \text{ x } 100; \qquad (2.3)$$

Для досліджуваної заповненої грунтом комірки касети з висадженим насінням маса сухого ґрунту становить 40 г, маса повністю вологого ґрунту становить 80 г, тож можна зробити висновок що для збільшення вологості ґрунту на 1% необхідно додати 1 мл води.

Наступним етапом розрахунку математичної моделі є співвідношення потужності насосу та об’єму води яка необхідна для насичення ґрунту вологою на певний відсоток. Перш за все необхідно визначитися з насосом який буде використано в системі.

У якості досліджуваного насосу буде використано мембранний насос із наступними характеристиками:

Таблиця 2.4 – Характеристики мембранного насосу

Робоча напруга, U_{zhNas}	12 В
Мінімальна напруга, U_{minNas}	6 В
Тип струму	Постійний
Робочий струм, I_{rabNas}	0,5-0,7 А
Холостий струм, I_{holNas}	0,18 А
Максимальна висота всасування	2 м.
Продуктивність	1,5-2 л/хв.
Максимальна температура речовини	80 ℃

Виходячи з цих характеристик видно що продуктивність, інакше кажучи потужність, P_{nas} насосу 1.5-2 л/хв, або 25-33 мл/с.

Кінцевим етапом розрахунку є співвідношення часу вмикання насосу від

об’єму води яку необхідно додати.

Використовуючи дані отримані з п.2.3 та таблиці 2.4 побудовано таблицю співвідношення часу роботи та об’єму води з урахуванням прогнозування необхідної кількості води на різних глибинах.

Таблиця 2.5 – Співвідношення початкової та залишкової вологості ґрунту на різній глибині в залежності від часу

Необхідна вологість ($W_{поч.}$), %	Глибина (h), см	Об’єм води, мл (при 1 мл = 1%) на 100 г грунту	Об’єм води, мл (при 1 мл = 1%) на 40 г грунту	Час роботи насосу, с (при $P_{srnas} = 29$ мл/с)	Залишкова вологість ($W_{зал.}$), %
30	2	35	14	0,483	29,9
	5	50	20	0,69	29,9
	7	76	30,4	1,048	29,76
60	2	66	26,4	0,91	59,8
	5	85	34	1,172	59,9
	7	85+30	46	1,586	58,9
90	2	100	40	1,379	90
	5	100+30	52	1,793	69
	7	100+85	74	2,552	90,5

Звертаючи увагу на таблицю 2.5 необхідно зробити декілька пояснень:

- У третьому стовпчику з розрахунком об’єму води на 100 г ґрунту, деякі значення відображено у вигляді суми. Дана сума є відображенням перенасичення поверхневого шару ґрунту для забезпечення необхідного відсотка вологи на нижчих шарах.
- У п’ятому стовпчику з розрахунком часу роботи насосу, відображено час необхідний для перекачування тільки необхідного ґрунту об’єму води без

урахування часу необхідного для прокачування системи водопостачання від насосу до рослини. У реальних умовах, до розрахованого часу необхідно додавати стале значення часу на заповнення системи.

Орієнтований час для заповнення розраховується за формулою:

$$t = \frac{V,\text{мл}}{P_{srnas}} \text{ (с)}; \quad (2.4)$$

$$V, \text{мл} = \frac{R*\pi*L}{1000} = \frac{3*3.14*600}{1000} = 5.652 \quad (2.5)$$

де R - радіус трубки системи (мм), π – число Пі, L – довжина трубки (мм).

У якості трубки, що буде розраховано, було використано медичну трубку для капельниць з такими параметрами: $R = 3$ мм, $L = 600$ мм, тоді час для заповнення системи для однієї комірки з рослиною дорівнює:

$$t = \frac{5.652}{29} = 0.195 \text{ с} \quad 2.6)$$

Аналізуючи розрахунки вище, для однієї комірки з рослиною додаткова стала часу становить 0.195 с, або 195 мс.

Усі проведені розрахунки математичної моделі, об’єму води та часу роботи насосу надають змогу у створенні співвідношення тривалості поливу ґрунту в сукупності залежностей початкової вологості та фази росту рослини.

2.5 Висновки до другого розділу

1. Розроблено структурну схему та алгоритм системи контролю вологості в тепличних грунтах. Структурна схема є загальною та відображає основний принцип роботи і взаємозв’язку елементів схеми.
2. Проведено дослідження росту кореня рослини в залежності від доби рослини. Дослідження проводилося експериментально з використанням насіння

помідора сорту «Мобіл» середньоранній. В результаті дослідження було отримано залежність довжини кореня та глибина на якій знаходиться корінь від доби росту рослини.

3. Побудовано та досліджено математичну модель динаміки вологи в ємності з тепличним ґрунтом. Математична модель була розрахована для ємності з параметрами: довжина – 65 мм, ширина – 65 мм, висота – 70 мм. Така математична модель дозволяє виконувати вимірювання вологості тепличних грунті під поверхнею ґрунту та прогнозувати вологість ґрунту на різних глибинах через певний час.

4. Розраховано час роботи поливного насосу в залежності від початкової вологості ґрунту під поверхнею та потужності насосу. Було визначено що для досліджуваного насосу та поливного об’єму ґрунту час роботи не перевищує 2,6 с. Окрім цього було розраховано додаткову сталу часу на заповнення системи водопостачання. Усі отримані результати будуть використані у подальшій розробці бази правил нечіткої логіки, імітаційної та реальної моделей.

РОЗДІЛ 3
СТВОРЕННЯ ІМІТАЦІЙНОЇ МОДЕЛІ СИСТЕМИ З УРАХУВАННЯМ ДОСЛІДЖУВАНИХ МАТЕРІАЛІВ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Для створення імітаційної моделі системи контролю вологості тепличних ґрунтів на базі нечіткої логіки необхідне використання різного програмного забезпечення (ПЗ). Створення моделі буде відбуватися поетапно:

- Першим етапом буде розробка бази правил нечіткої логіки та її первинне тестування за допомогою пакету програмного забезпечення MATLAB&SIMULINK із використанням додатку FuzzyLogicToollbox. Д буде конвертація отриманого файлу бази правил у файл з програмним кодом для програми ArduinoIDE.
- Другий етап це конвертація отриманого файлу з нечіткою логікою в програмний код за допомогою онлайн сайту.
- Третій етап – проектування відконвертованого програмного коду та моделі для контролера Arduino UNO за допомогою програми ArduinoIDE.
- Розробка імітаційної моделі в пакеті програм автоматизованого проектування PROTEUS.

Вхідною інформацію для створення імітаційної моделі буде використання отриманих даних з другого розділу.

3.1 Розробка бази правил нечіткої логіки в та її первинне тестування в ПЗ MATLAB&SIMULINK

Вхідні блоки даних

Так, як розроблювана система має дві вхідні змінні, вік (доба) росту рослини та вологість ґрунту, то і на вході нечіткої логіки теж буде два блоки.

- Перший блок це вхідні дані з датчику вологості, діапазон вхідного сигналу встановлюється від 30 до 95 %. Якщо реальне значення вологості виходить

за максимальне чи мінімальне значення діапазону зазначеного у блоці, то виконується та ж сама операція як і для граничної з цим діапазоном операції.

- Другий блок це вхідна інформація з самого мікроконтролеру, це вік рослини який вимірюється у добах. Діапазон розмежовано на доби від 0, першої доби, до 30, максимальний вік здорової рослини готової до пересадження. Відлік доби виконується за допомогою модуля відліку часу.

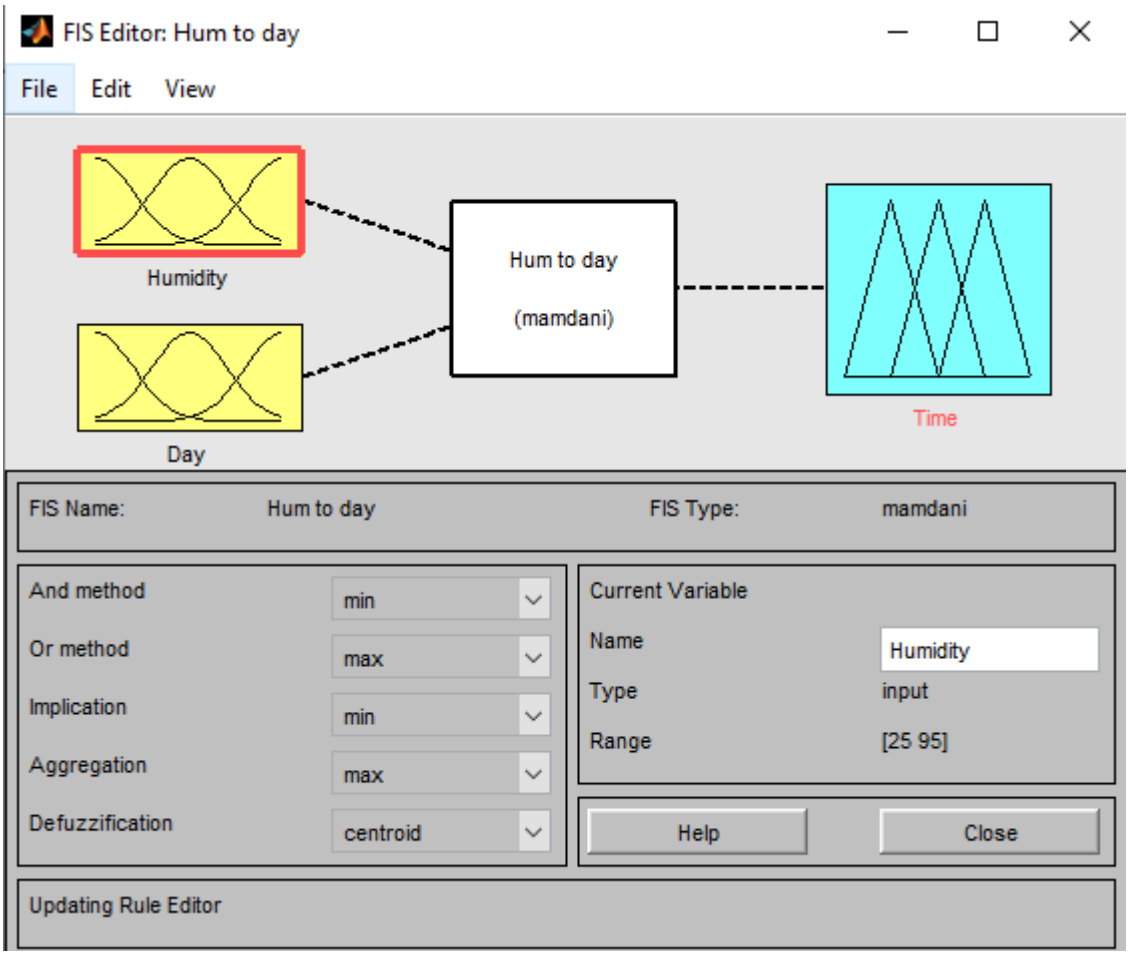


Рисунок 3.1 – загальний вигляд програми створення бази правил нечіткої логіки

Блок вхідних даних вологості ґрунту:

Діапазон даного блоку було поділено на п’ять частин:

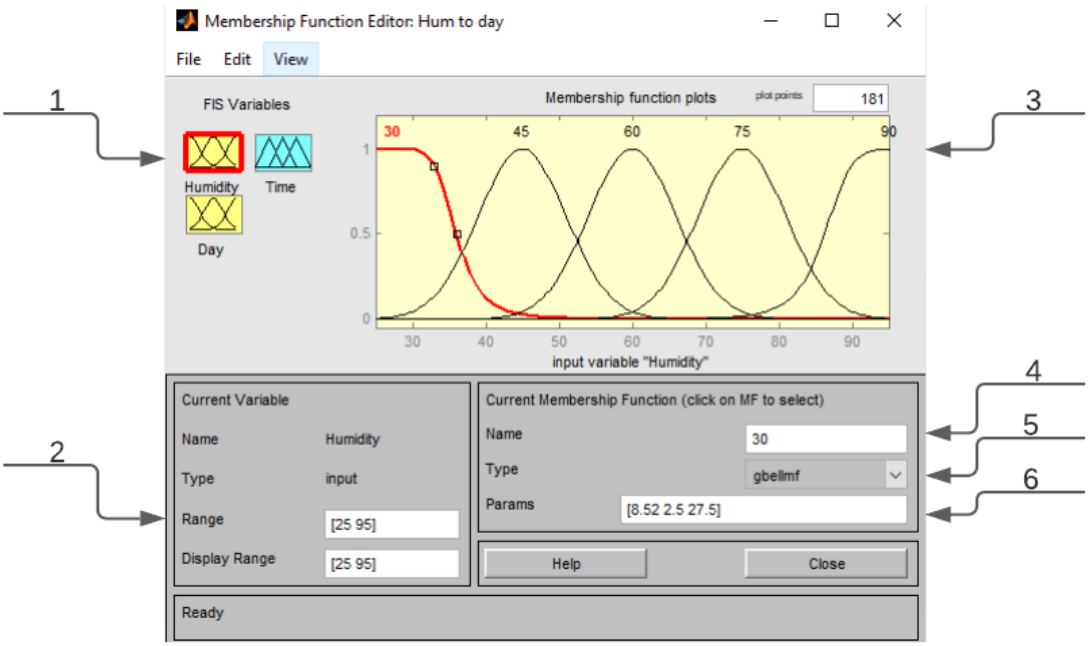


Рисунок 3.2 – Блок вхідних даних з датчика вологості

На рисунку 3.2 – відображено блок для поділення вхідної інформації датчика вологості, для детального опису цього блоку, відповідно до цифр на зображені, в таблиці 3.1 знаходяться пояснення елементів блоку:

Таблиця 3.1 – Описання елементів налаштування блоку вхідної інформації

Номер позначення	Пояснення
1	Кнопка вибору вікна налаштування вхідної інформації
2	Поле задавання діапазону сигналу
3	Поле налаштування та відображення створених рівнів розподілення сигналу
4	Поле задання назви функції рівня
5	Кнопка для вибору типу функції рівня
6	Вікно налаштування діапазону рівня функції

Діапазон блоку поділено на п’ять рівнів, починаючи від 25 до 95.

Таку ж саму конструкцію має блок вхідної інформації з модуля реального часу, який відповідає за вік рослини.

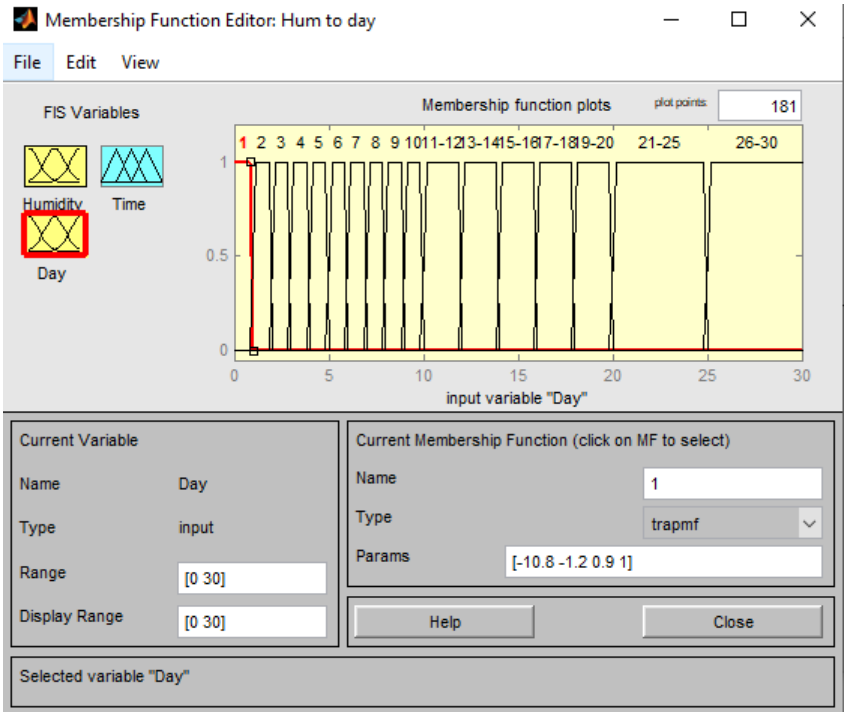


Рисунок 3.3 – Блок вхідних даних з модуля часу

Діапазон відліку часу поділено на 17 рівнів, від першого до десятого рівня включно кожному рівню відповідає один день, від одинадцятого до п’ятнадцятого рівня – два дні, а шістнадцятий та сімнадцятий рівні займають по п’ять днів. Таке розподілення пояснюється інтенсивністю росту та занурення кореня рослини в ґрунт. В перші дні корінь швидко росте та занурюється у ґрунт, в подальшому ріст кореня сповільнюється та заповнює простір комірки з ґрунтом, тому останні два рівні займають по п’ять днів.

Блок вихідних даних

Блок вихідних даних складається з однієї частини. В ній задаються рівні керування вихідним сигналом, в розрізі розробляємої системи це керування часом роботи насосу. В залежності від обраного базою правил рівня блоку вихідного сигналу змінюється час роботи насосу поливу ґрунту.

Час роботи поділено на дев’ять рівнів, що розміщені в діапазоні від 0 до 3 у відповідності до необхідного часу роботи насосу розрахованому у таблиці 2.4

з додаванням постійної часу на заповнення системи водопостачання:

Таблиця 3.2 – Рівні вихідного сигналу

Рівень	1 (0,483)	2 (0,69)	3 (0,91)	4 (1,05)	5 (1,17)	6 (1,38)	7 (1,59)	8 (1,79)	9 (2,55)
Час роботи, с	0 – 0,6	0,48 – 0,89	0,734 - 1,036	0,925 - 1,194	1,08 - 1,313	1,2 - 1,56	1,37 - 1,798	1,6 - 2,377	1,813 - 3,0

Для забезпечення плавного переходу від одного рівня до іншого між рівнями було зроблено накладання границь рівнів. Це по-перше допомагає запобігти виникнення провалів в роботі системи, тобто моментів коли при певному збіганні сигналів програма не знатиме що робити, і по-друге для більш плавного переключення між рівнями. Такі правила накладання і переходу було використано в усіх блоках, як вихідному, так і у вхідних.

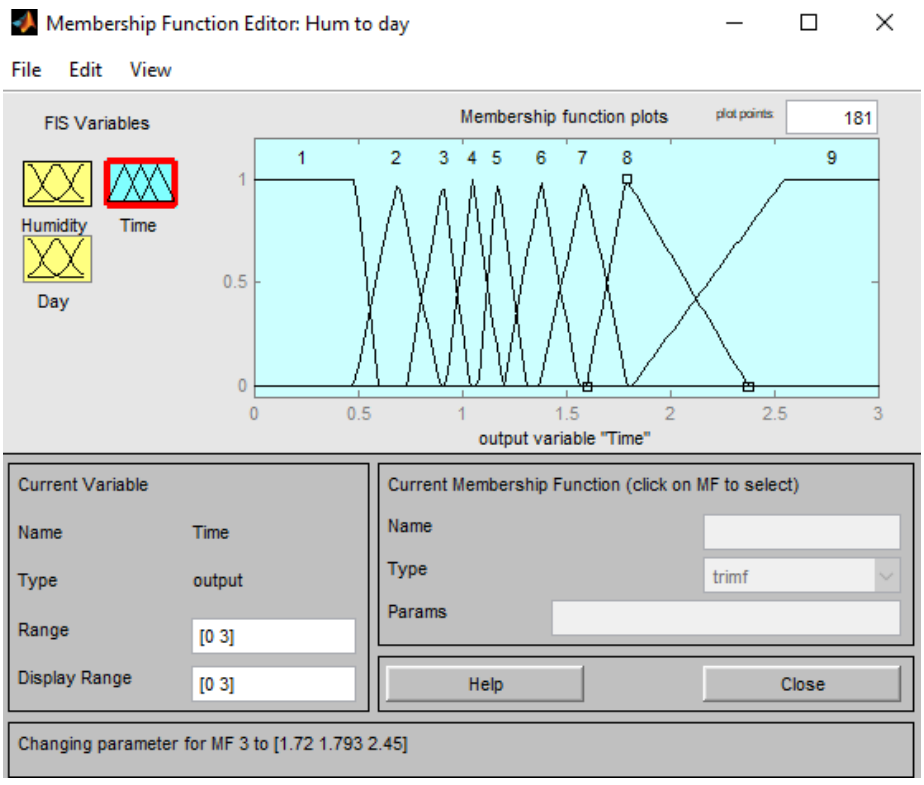


Рисунок 3.4 – Блок вихідного сигналу для керування часом роботи насосу

Блок створення бази правил нечіткої логіки

Головною частиною у створенні бази правил нечіткої логіки, це блок бази правил, інакше кажучи, Редактор правил. За допомогою цього блоку проектувальник створює рівняння залежності вихідного сигналу від вхідних. Дані правила проектуються таким чином, щоб на кожну можливу комбінацію було створено свою функцію реагування результатом якої є певний вихідний сигнал. Для більш кращого уявлення цієї системи керування наведено матрицю відповідності правил до функцій належності параметрів. [31]

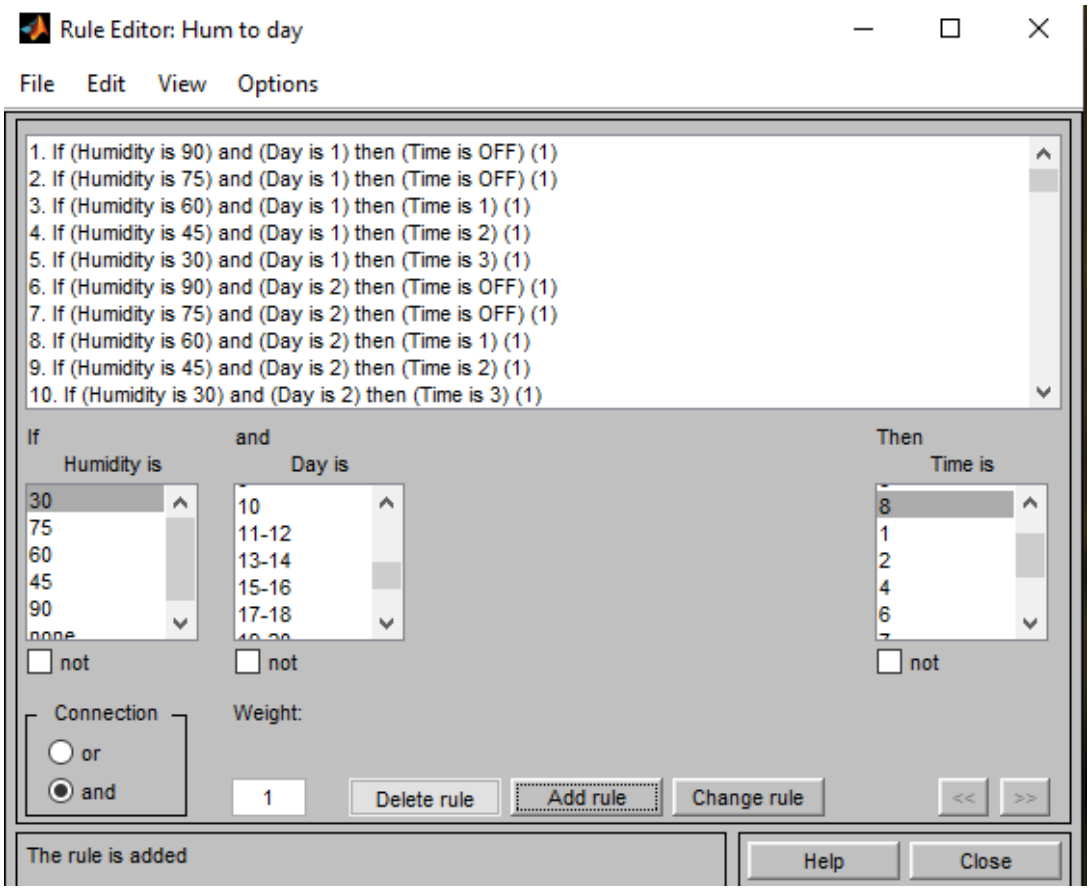


Рисунок 3.5 – Редактор правил обробки вхідних сигналів

Створення імітаційної моделі контролю вологості тепличних ґрунтів на базі нечіткої логіки в програмі MATLAB

Після того як було створено базу правил нечіткої логіки необхідно її

протестувати в програмному забезпеченні MATLAB&SIMULINK

Для імітації вхідних сигналів було використано модулі генераторів випадкового сигналу, діапазон яких відповідає вимірюваним діапазонам вологості та віку (доби) рослини. Окрім цього вхідний сигнал датчика вологості було зашумлено за допомогою додатково підключеного генератора шуму із амплітудою яка відповідає реальній, допустимій похибці датчика.

1. If (Humidity is 90) and (Day is 1) then (Time is OFF) (1)
2. If (Humidity is 75) and (Day is 1) then (Time is OFF) (1)
3. If (Humidity is 60) and (Day is 1) then (Time is 1) (1)
4. If (Humidity is 45) and (Day is 1) then (Time is 2) (1)
5. If (Humidity is 30) and (Day is 1) then (Time is 3) (1)
6. If (Humidity is 90) and (Day is 2) then (Time is OFF) (1)
7. If (Humidity is 75) and (Day is 2) then (Time is OFF) (1)
8. If (Humidity is 60) and (Day is 2) then (Time is 1) (1)
9. If (Humidity is 45) and (Day is 2) then (Time is 2) (1)
10. If (Humidity is 30) and (Day is 2) then (Time is 3) (1)
11. If (Humidity is 90) and (Day is 3) then (Time is OFF) (1)
12. If (Humidity is 75) and (Day is 3) then (Time is OFF) (1)
13. If (Humidity is 60) and (Day is 3) then (Time is 1) (1)
14. If (Humidity is 45) and (Day is 3) then (Time is 2) (1)
15. If (Humidity is 30) and (Day is 3) then (Time is 3) (1)
16. If (Humidity is 90) and (Day is 4) then (Time is OFF) (1)
17. If (Humidity is 75) and (Day is 4) then (Time is OFF) (1)
18. If (Humidity is 60) and (Day is 4) then (Time is 1) (1)
19. If (Humidity is 45) and (Day is 4) then (Time is 2) (1)
20. If (Humidity is 30) and (Day is 4) then (Time is 3) (1)
21. If (Humidity is 90) and (Day is 5) then (Time is OFF) (1)
22. If (Humidity is 75) and (Day is 5) then (Time is OFF) (1)
23. If (Humidity is 60) and (Day is 5) then (Time is 1) (1)
24. If (Humidity is 45) and (Day is 5) then (Time is 2) (1)
25. If (Humidity is 30) and (Day is 5) then (Time is 3) (1)
26. If (Humidity is 90) and (Day is 6) then (Time is 1) (1)
27. If (Humidity is 75) and (Day is 6) then (Time is 1) (1)
28. If (Humidity is 60) and (Day is 6) then (Time is 2) (1)
29. If (Humidity is 45) and (Day is 6) then (Time is 3) (1)
30. If (Humidity is 30) and (Day is 6) then (Time is 4) (1)
31. If (Humidity is 90) and (Day is 7) then (Time is 1) (1)
32. If (Humidity is 75) and (Day is 7) then (Time is 2) (1)
33. If (Humidity is 60) and (Day is 7) then (Time is 2) (1)
34. If (Humidity is 45) and (Day is 7) then (Time is 3) (1)
35. If (Humidity is 30) and (Day is 7) then (Time is 4) (1)
36. If (Humidity is 90) and (Day is 8) then (Time is 1) (1)
37. If (Humidity is 75) and (Day is 8) then (Time is 2) (1)
38. If (Humidity is 60) and (Day is 8) then (Time is 2) (1)
39. If (Humidity is 45) and (Day is 8) then (Time is 3) (1)
40. If (Humidity is 30) and (Day is 8) then (Time is 4) (1)
41. If (Humidity is 90) and (Day is 9) then (Time is 1) (1)
42. If (Humidity is 75) and (Day is 9) then (Time is 2) (1)
43. If (Humidity is 60) and (Day is 9) then (Time is 2) (1)
44. If (Humidity is 45) and (Day is 9) then (Time is 3) (1)
45. If (Humidity is 30) and (Day is 9) then (Time is 4) (1)
46. If (Humidity is 90) and (Day is 10) then (Time is 1) (1)
47. If (Humidity is 75) and (Day is 10) then (Time is 2) (1)
48. If (Humidity is 60) and (Day is 10) then (Time is 2) (1)
49. If (Humidity is 45) and (Day is 10) then (Time is 3) (1)
50. If (Humidity is 30) and (Day is 10) then (Time is 4) (1)
51. If (Humidity is 90) and (Day is 11-12) then (Time is 2) (1)
52. If (Humidity is 75) and (Day is 11-12) then (Time is 3) (1)
53. If (Humidity is 60) and (Day is 11-12) then (Time is 3) (1)
54. If (Humidity is 45) and (Day is 11-12) then (Time is 4) (1)
55. If (Humidity is 30) and (Day is 11-12) then (Time is 5) (1)
56. If (Humidity is 90) and (Day is 13-14) then (Time is 2) (1)
57. If (Humidity is 75) and (Day is 13-14) then (Time is 3) (1)
58. If (Humidity is 60) and (Day is 13-14) then (Time is 3) (1)
59. If (Humidity is 45) and (Day is 13-14) then (Time is 4) (1)
60. If (Humidity is 30) and (Day is 13-14) then (Time is 5) (1)
61. If (Humidity is 90) and (Day is 15-16) then (Time is 2) (1)
62. If (Humidity is 75) and (Day is 15-16) then (Time is 3) (1)
63. If (Humidity is 60) and (Day is 15-16) then (Time is 4) (1)
64. If (Humidity is 45) and (Day is 15-16) then (Time is 5) (1)
65. If (Humidity is 30) and (Day is 15-16) then (Time is 5) (1)
66. If (Humidity is 90) and (Day is 17-18) then (Time is 3) (1)
67. If (Humidity is 75) and (Day is 17-18) then (Time is 3) (1)
68. If (Humidity is 60) and (Day is 17-18) then (Time is 4) (1)
69. If (Humidity is 45) and (Day is 17-18) then (Time is 5) (1)
70. If (Humidity is 30) and (Day is 17-18) then (Time is 6) (1)
71. If (Humidity is 90) and (Day is 19-20) then (Time is 3) (1)
72. If (Humidity is 75) and (Day is 19-20) then (Time is 3) (1)
73. If (Humidity is 60) and (Day is 19-20) then (Time is 4) (1)
74. If (Humidity is 45) and (Day is 19-20) then (Time is 5) (1)
75. If (Humidity is 30) and (Day is 19-20) then (Time is 6) (1)
76. If (Humidity is 90) and (Day is 21-25) then (Time is 3) (1)
77. If (Humidity is 75) and (Day is 21-25) then (Time is 4) (1)
78. If (Humidity is 60) and (Day is 21-25) then (Time is 5) (1)
79. If (Humidity is 45) and (Day is 21-25) then (Time is 6) (1)
80. If (Humidity is 30) and (Day is 21-25) then (Time is 7) (1)
81. If (Humidity is 90) and (Day is 26-30) then (Time is 4) (1)
82. If (Humidity is 75) and (Day is 26-30) then (Time is 5) (1)
83. If (Humidity is 60) and (Day is 26-30) then (Time is 6) (1)
84. If (Humidity is 45) and (Day is 26-30) then (Time is 7) (1)
85. If (Humidity is 30) and (Day is 26-30) then (Time is 8) (1)

Рисунок 3.6 – Загальний перелік правил обробки вхідних сигналів та генерації вихідного сигналу

Від генераторів випадкового сигналу інформація надходить до блоку

обробки інформації. В даному блоці завантажено файл з базою правил нечіткої логіки системи. Сигнали проходять обробку, і в залежності від їх значень, блок обробки видає відповідний тому чи іншому вихідному сигналу. Вихідний сигнал, відповідно, це є час роботи насосу.

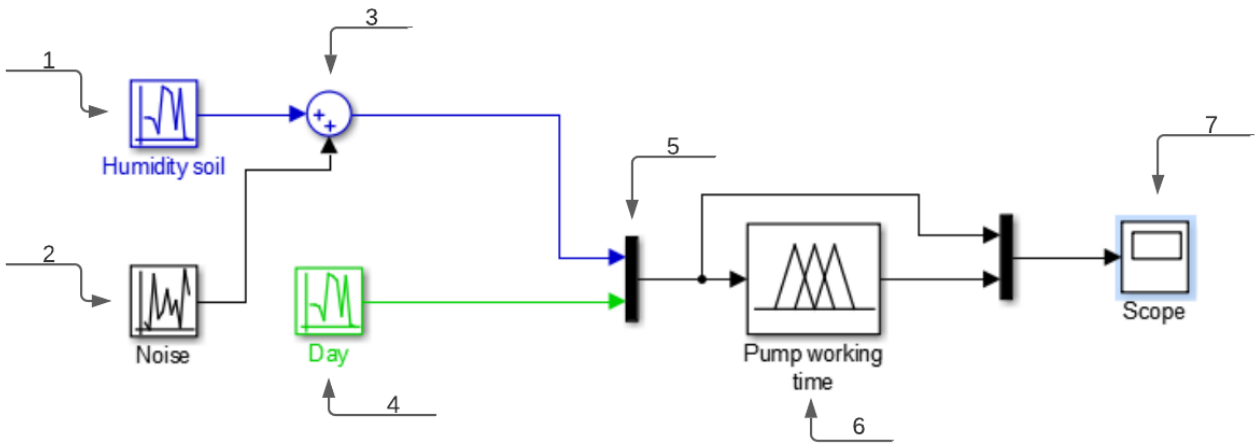


Рисунок 3.7 – Імітаційна модель контролю вологості тепличних ґрунтів

На рисунку 3.7 пронумеровано елементи які складають дану модель, а саме:

Таблиця 3.3 – Складові імітаційної моделі

Номер позначення	Пояснення
1	Генератор випадкового сигналу (датчик вологості)
2	Генератор шуму
3	Суматор сигналів, накладає шум на «чистий» сигнал з датчика вологості
4	Генератор випадкового сигналу (доба)
5	Мультиплексор вхідних сигналів
6	Контролер нечіткої логіки, в ньому оброблюються вхідні сигнали та виходить керуючий сигнал
7	Осцилограф, наглядно демонструє залежність сигналів та принцип роботи нечіткої логіки

На виході системи розташовано осцилограф який частково виконує функцію насосу, а саме відображує вихідний керуючий сигнал. Окрім цього

осцилограф відображує вхідні сигнали до обробки нечіткою логікою, це відображення дає змогу побачити залежність вихідного сигналу від вхідного.

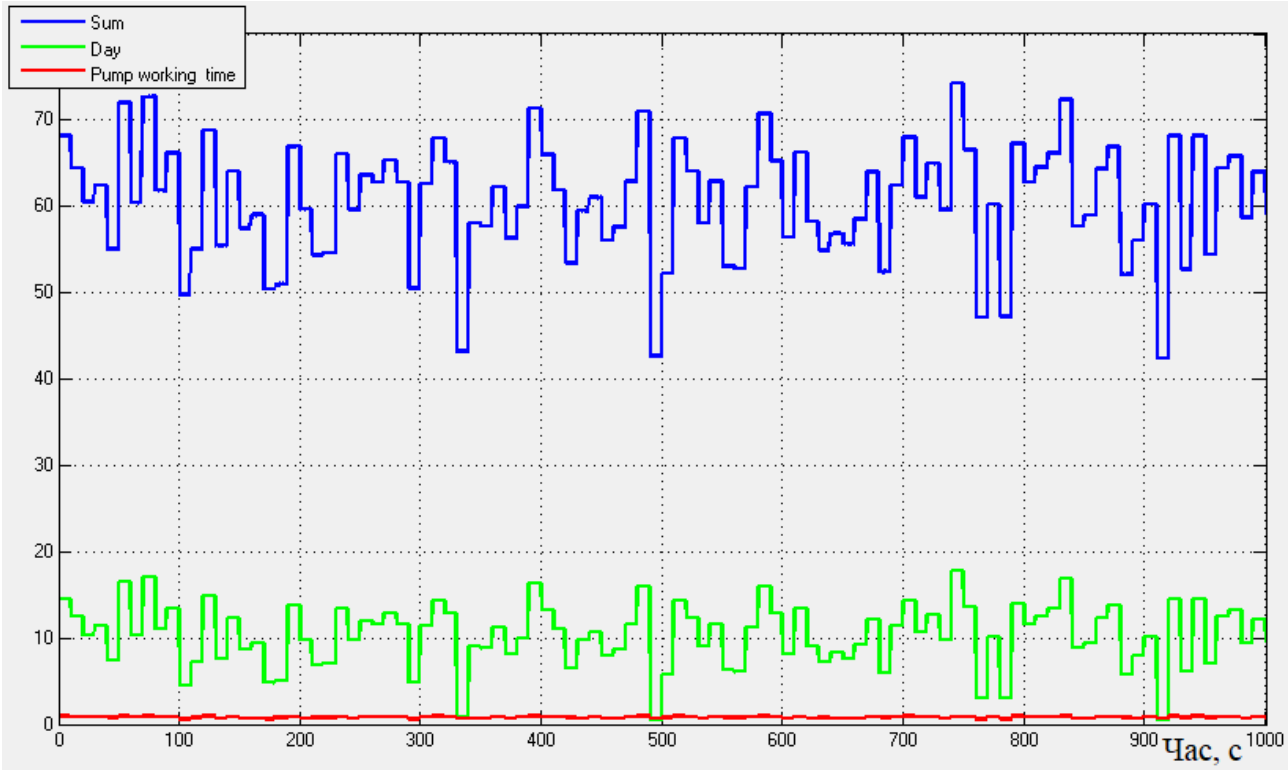
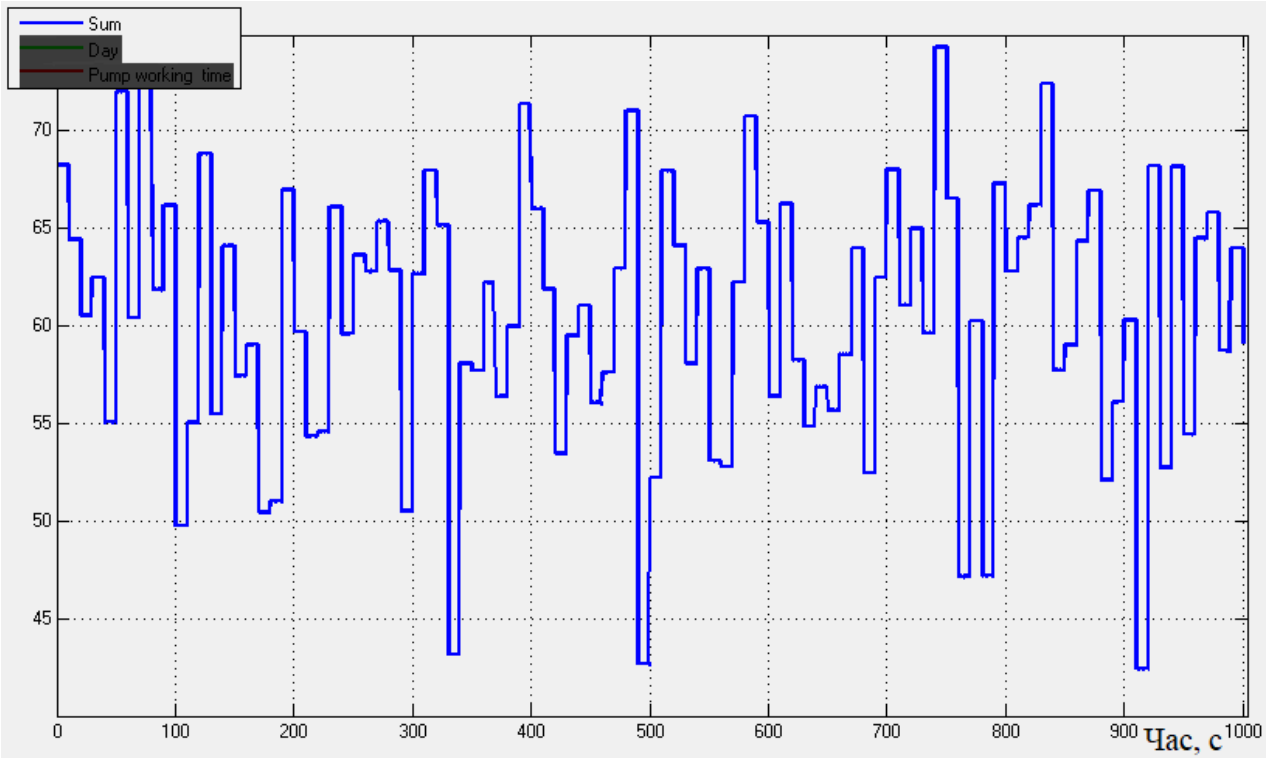
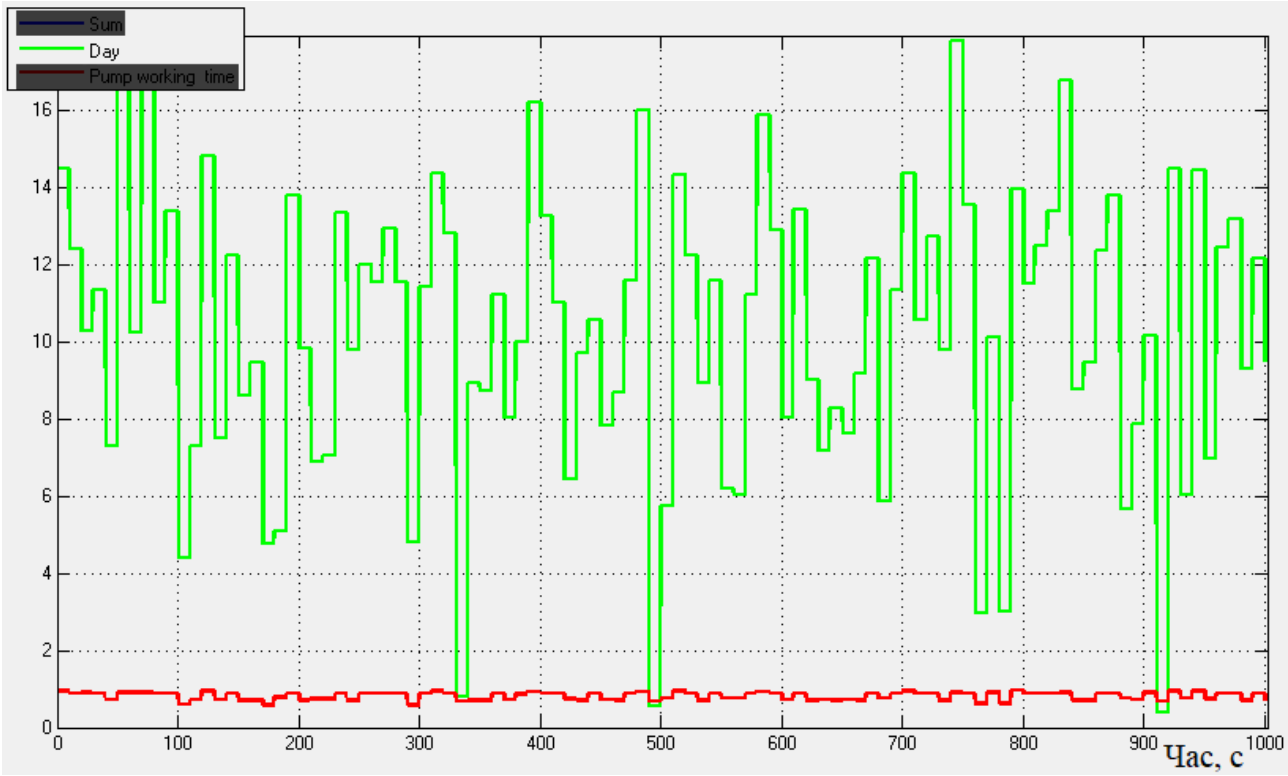


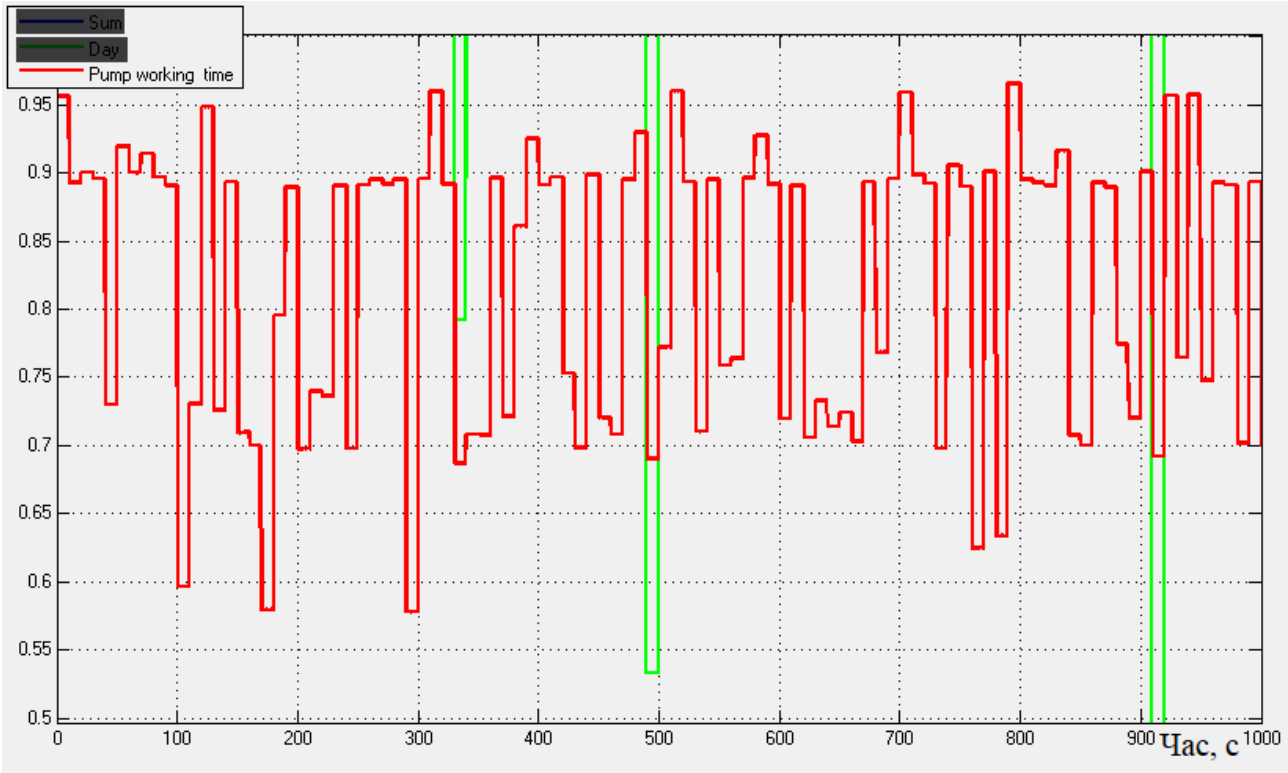
Рисунок 3.8 – Загальне відображення сигналів на осцилограмі



а) Блакитний сигнал – імітаційний сигнал зміни вологості грунту



б) Зелений сигнал – імітаційний сигнал зміни доби росту рослини



в) Червоний сигнал – імітаційний сигнал часу роботи насосу

Рисунок 3.9 – Окреме відображення сигналів на осцилограмі імітаційної моделі в залежності від часу роботи імітаційної моделі

Виходячи з результатів отриманих при тестуванні імітаційної моделі в програмному забезпеченні MATLAB&SIMULINK, модель є повністю робочою та готовою для подальшого випробовування та тестування.

3.2 Компіляція файлу .fis в .ino

В програмному забезпеченні MATLAB&SIMULINK та його розширенні FuzzyLogicToollbox було створено файл типу .fis. Даний формат файлу не підтримується в програмі програмування мікроконтролеру Arduino IDE. Для коректної та нормальної роботи з іншими програмами необхідно конвертувати отриманий файл в файл типу .ino. В інтернет мережі є сайт з швидким та легким конвертуванням саме такого формату файлів. [32]

Конвертування проводиться за наступним алгоритмом:

- На сайт завантажується файл
- Натискаємо на кнопку “Convert”
- Після обробки файлу його можна завантажити на комп’ютер
- Відкриваємо завантажений файл за допомогою програми Arduino IDE

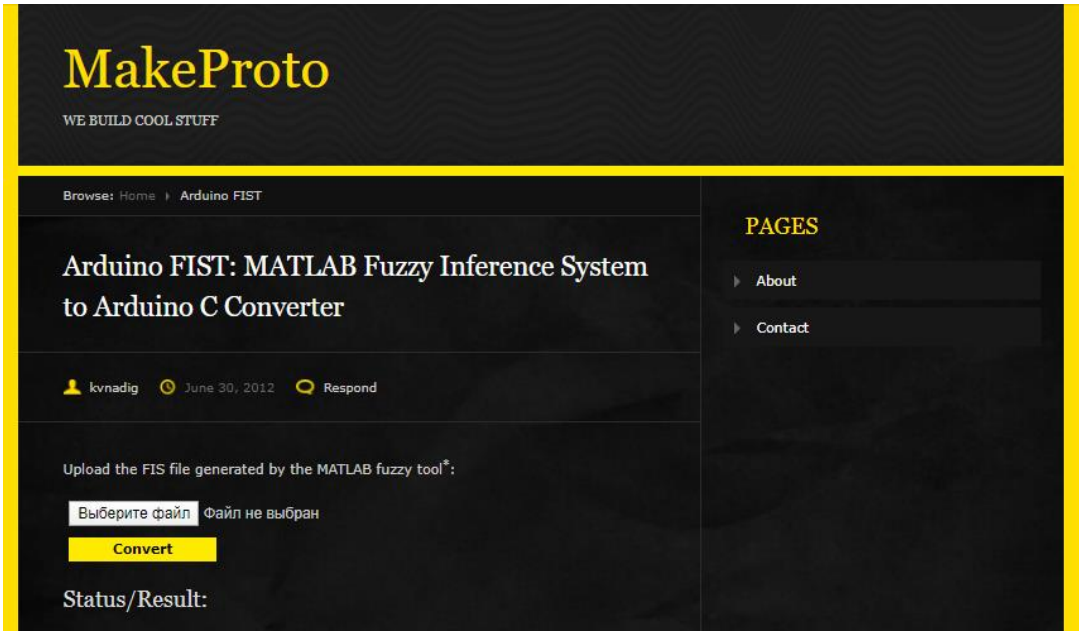


Рисунок 3.10 – Відображення візуальної частини сайту з конвертації

У додатку В (див. додаток В) знаходиться лістинг первинного програмного коду після конвертування.

Після отримання первинного програмного коду робота з програмним забезпеченням MATLAB&SIMULINK та його розширенням FuzzyLogicToollbox завершується. Головною метою даного етапу розробки було отримання бази правил нечіткої логіки в програмному коді. Наступним кроком роботи буде створення та тестування кінцевого програмного коду та його випробовування спочатку в якості імітаційної моделі в програмі Proteus.

3.3 Проектування програмного коду для контролера Arduino UNO

Основна частина роботи налаштування первинного коду програми заснована на підключенні периферичного обладнання, а саме датчика тиску, насосу та модуля реального часу. Датчик тиску підключається до контролера за допомогою аналогового входу плати Arduino UNO, в даному випадку до нульового входу «A0»

```
// Pin mode for Input: Humidity
pinMode(A0 , INPUT);
```

Рисунок 3.11 – Частина коду ініціалізації підключення аналогового входу для датчика вологості

Наступним кроком для цього датчика буде присвоєння змінної та межі перетворення вхідного аналогового сигналу відповідно до меж які задані в нечіткій логіці.

```
// Read Input: Humidity
int H = analogRead(A0);
g_fisInput[0] = map(H, 0, 1023, 25, 95);
```

Рисунок 3.12 – Частина коду конвертування аналогового сигналу в необхідний діапазон

Наступним кроком є підключення другого вхідного сигналу, а саме віку (доби) рослини. У процесі розробки було протестовано декілька варіантів завдання віку рослини. Першим, та найпростішим способом було задання константи через програмний код, и тим самим перевірка роботи системи відносно зміни вологості. Другим методом було використання аналогового сигналу, як і з датчиком вологості, за допомогою підключення потенціометрів. Даний метод допоміг реалізувати імітаційну модель в програмі Proteus. І врешті-решт третій варіант відліку віку рослини – це відлік часу за допомогою модуля реального часу.

В модуль часу реальний час можна завантажити двома засобами, або автоматично під час завантаження програмного коду у плату, або самостійно виставляючи необхідний час. Потім за допомогою команди зчитується дата та присвоюється у певну змінну. Наступним кроком є умова порівняння актуальної дати з датою яку було присвоєно попередньо у змінну. Якщо записана дата не співпадає з актуальною датою, то всередені умови до попередньо створеної змінної лічильника додається одиниця. В свою чергу контролер з нечіткою логікою зчитує змінну лічильника яка, по суті є добою росту рослини.

Модуль DS3231 підключається до плати Arduino UNO через інтерфейс I2C, використовуються виводи SDA та SCL. Для програмування буде використано бібліотеки DS3231 та RTCDateTime. На виході через COM-порт буде виводитись вік рослини та реальний час і дата у форматі “година / хвилина / секунда / день / місяць / рік”.

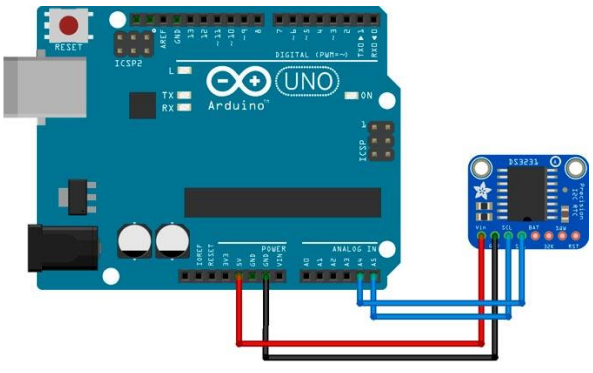


Рисунок 3.13 – Схема підключення модулю часу до плати Arduino UNO

```
  dt = clock.getDateTime();

if (dt.day != SDay)
{
  SDay = dt.day;
  Day ++;
}
```

Рисунок 3.14 – Фрагмент коду підключення модуля реального часу

Останнім кроком налаштування програмного коду буде підключення насосу до вихідного сигналу. Принцип роботи системи, як вже було сказано раніше, полягає у включенні насосу на певний час для забезпечення зрощення ґрунту на необхідний відсоток вологи. Вихідний сигнал це насамперед елементарне значення змінної яке є результатом обробки вхідних сигналів через базу правил нечіткої логіки. Для того щоб з вихідного сигналу бази правил отримати час роботи насосу було використано дворівневе вмикання/вимикання живлення з піну до якого підключено насос а час затримки вмикання насосу дорівнює помноженню значення вихідної змінної на 1000 мілісекунд.

```
float pompout =(g_fisOutput[0]*1000);
digitalWrite(11, HIGH);
delay(pompout);
digitalWrite(11, LOW);
```

Рисунок 3.15 – Фрагмент коду підключення піна для керування насосом

3.4 Створення імітаційної моделі в пакеті програм автоматизованого проектування Proteus

Після тестування імітаційної моделі в пакеті програмного забезпечення MATLAB&SIMULINK необхідно перевірити працездатність моделі в більш реальних умовах без використання генератора шуму. Для реалізації цієї моделі буде використано програму Proteus, де в якості елементів системи буде використано:

Таблиця 3.4 – Складові елементи імітаційної моделі системи в програмі Proteus

Складовий елемент системи		Функції та призначення
Існуючий	Аналог в програмі Proteus	
Arduino UNO	Arduino UNO	Платформа для підключення датчиків, модуля реального часу та, насамперед, для опрацювання вхідної і вихідної інформації
Датчик вологості DFRobot	Потенціометр	Зчитування відсотку вологості в ґрунті діелькометричним методом та надсилання інформації на мікроконтролер
Модуль реального часу	Потенціометр	Відлік віку рослини через різницю між реальною датою та датою посадки рослини
Водяний насос	DC Мотор	Підкачка води в систему з часом роботи який залежить від вихідного сигналу
Осцилограф	Осцилограф та COM (послідовчий) порт	Допоміжний елемент системи, який допомагає відобразити вхідні сигнали та вихідний сигнал на насос

Складові елементи імітаційної системи було додано за допомогою вбудованих бібліотек програми Proteus.

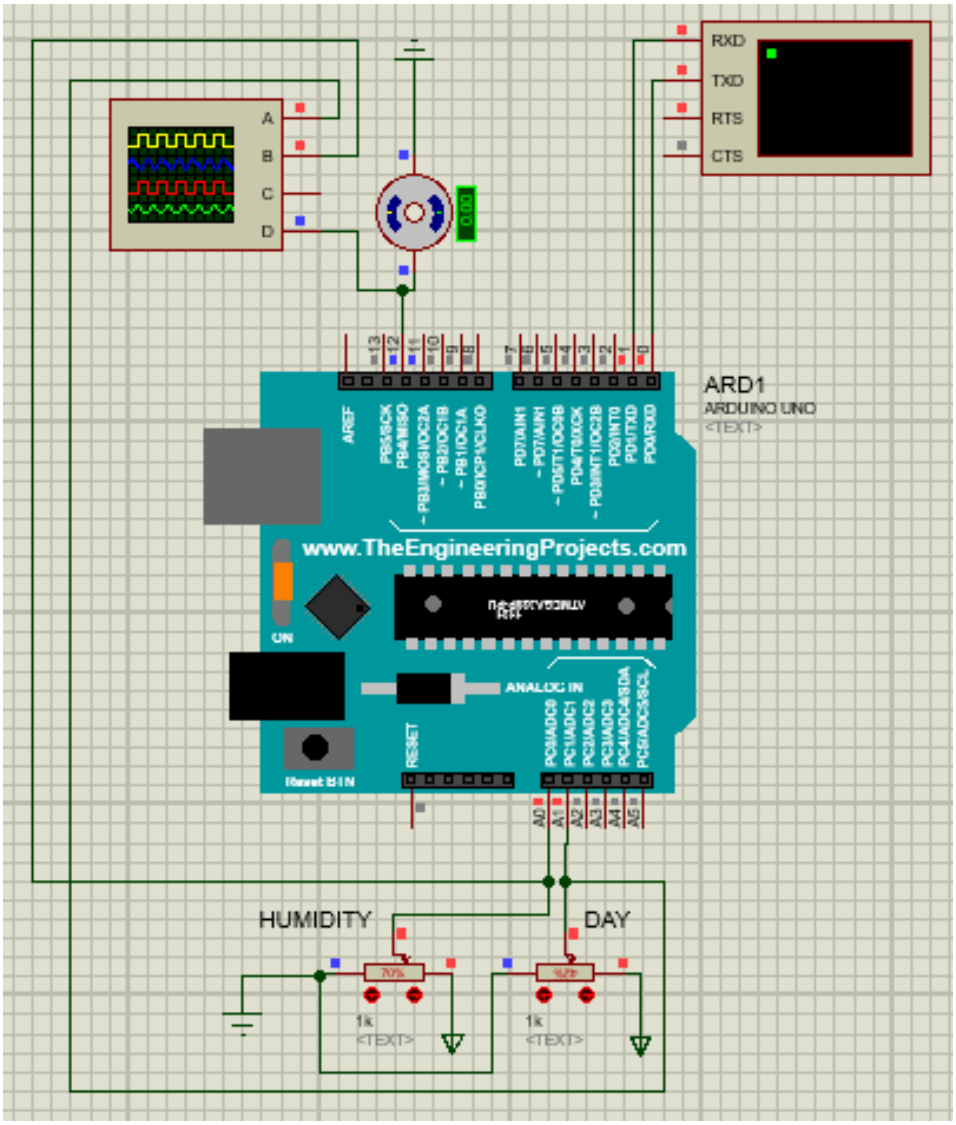
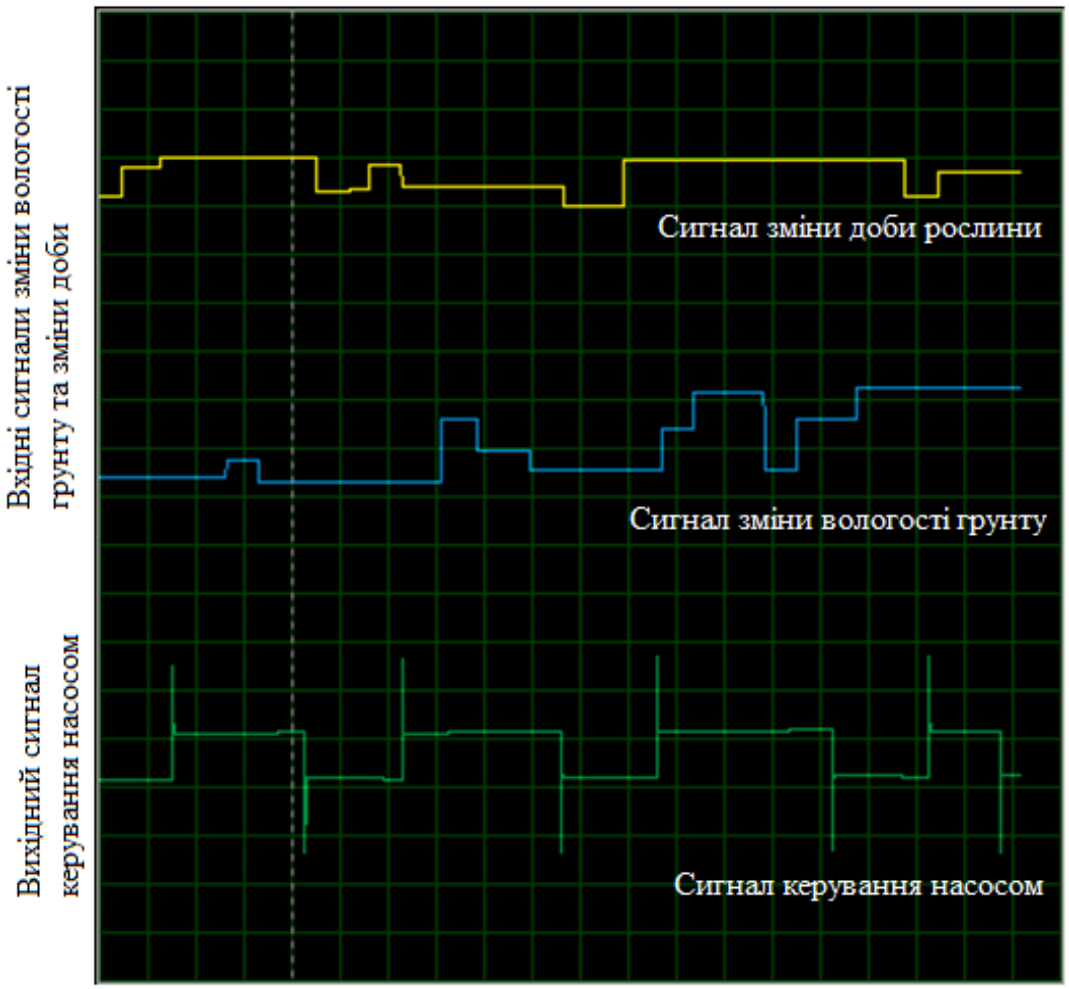


Рисунок 3.16 – Загальний вигляд структури імітаційної моделі в Proteus

```
H:53.00    D: 15.00   Time taken (sec.) = 1.11   pompout = 1112.92
H:41.00    D: 7.00    Time taken (sec.) = 0.69   pompout = 687.86
H:38.00    D: 16.00   Time taken (sec.) = 1.19   pompout = 1192.02
H:83.00    D: 22.00   Time taken (sec.) = 1.03   pompout = 1029.80
H:60.00    D: 30.00   Time taken (sec.) = 1.38   pompout = 1379.88
H:72.00    D: 8.00    Time taken (sec.) = 0.26   pompout = 257.86
H:54.00    D: 8.00    Time taken (sec.) = 0.38   pompout = 381.75
H:72.00    D: 26.00   Time taken (sec.) = 1.25   pompout = 1247.40
H:72.00    D: 12.00   Time taken (sec.) = 0.89   pompout = 890.42
H:49.00    D: 12.00   Time taken (sec.) = 1.00   pompout = 995.76
H:36.00    D: 12.00   Time taken (sec.) = 1.06   pompout = 1058.62
H:30.00    D: 27.00   Time taken (sec.) = 1.92   pompout = 1922.30
H:90.00    D: 27.00   Time taken (sec.) = 1.50   pompout = 1500.00
H:90.00    D: 0.00    Time taken (sec.) = 0.00   pompout = 0.00
H:90.00    D: 0.00    Time taken (sec.) = 0.00   pompout = 0.00
```

а)



б)

Рисунок 3.17 – Відображення інформації з послідовного порту (а) та осцилограма роботи імітаційної системи (б)

З результатів роботи імітаційної моделі можна побачити що розроблена система працює справно і виконує свої функції.

3.5 Висновки

1. За допомогою ПЗ MATLAB&SIMULINK розроблено базу правил нечіткої логіки. Було створено блоки вхідних та вихідних сигналів, налаштовано редактор правил обробки вхідних сигналів для генерації вихідного сигналу. Створено імітаційну модель контролю вологості тепличних ґрунтів на базі нечіткої логіки в програмі MATLAB. Тестування проводилося за допомогою

генераторів сигналу, зчитування вихідного сигналу було виконане за допомогою осцилографа.

2. Виконано конвертацію файлу з створеною базою правил у програмний код. Після отримання первинного програмного коду на його базі було виконано доробку згідно до потреб досліджуваної системи та її елементів. У програмі було додано зчитування інформації з аналогового датчика вологості, модуль реального часу, поливний насос.

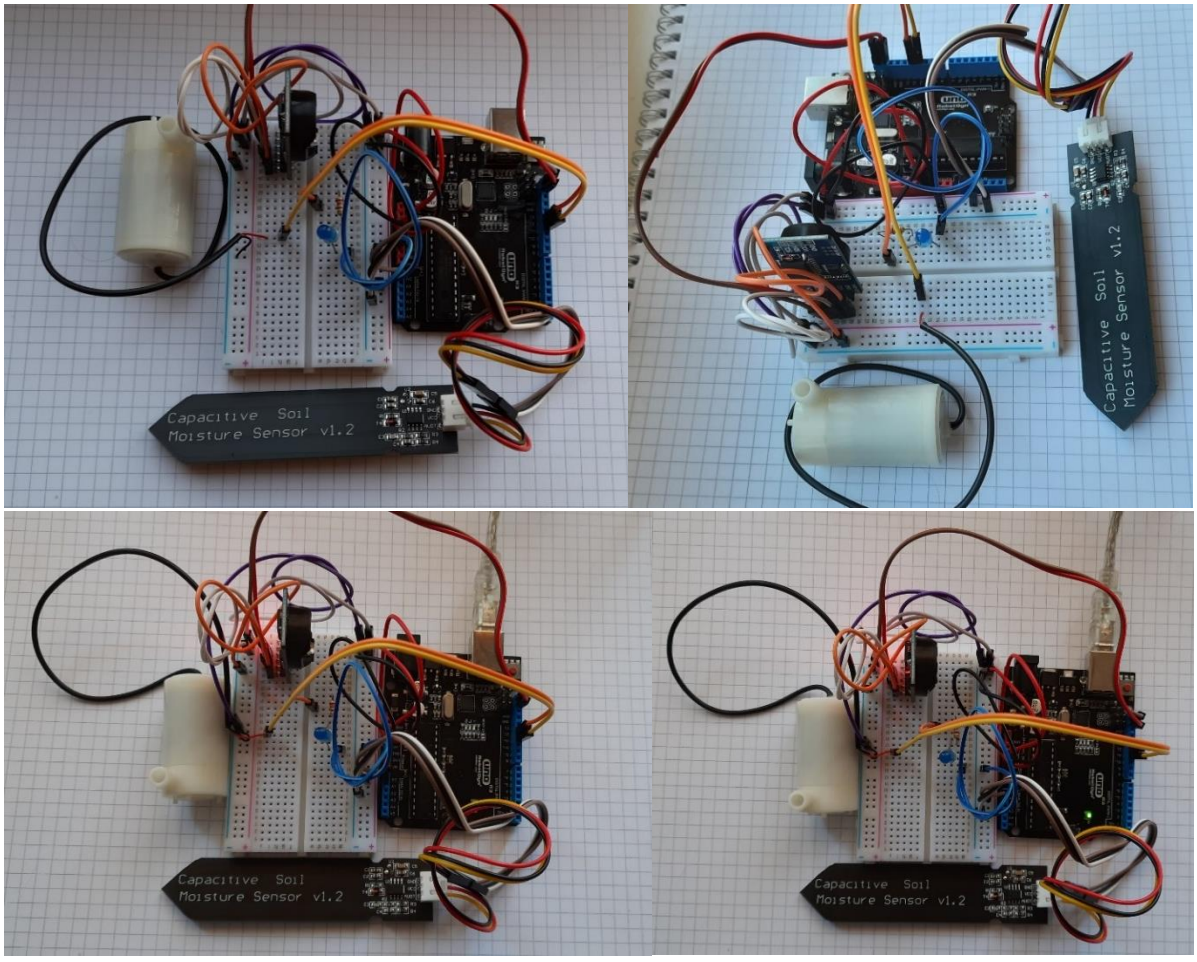
3. За допомогою програми Proteus було побудовано та протестовано імітаційну модель системи. Деякі елементи системи було замінено аналогами для кращого відображення кобінації сигналів. За результатами тестування моделі було визначено повну справність роботи програмного коду та системи в цілому.

РОЗДІЛ 4
РОЗРОБКА ДОСЛІДНОГО ЗРАЗКА ЕЛЕКТРОННОЇ СИСТЕМИ КОНТРОЛЮ ВОЛОГОСТІ ҐРУНТУ

4.1 Монтаж, запуск та тестування дослідного зразка електронної системи

В попередніх розділах роботи було визначено елементну базу системи, програмну частину системи та протестовано на імітаційних моделях. Останнім кроком розробки системи буде створення дослідної моделі системи.

Перш за все слід зазначити, що дослідна модель була розрахована для однієї комірки з рослиною. Окрім цього в модель було додано світлодіод для більш кращого відображення моменту роботи насосу. Вмикання і вимикання світлодіоду одночасне із пуском насосу. На ілюстраціях нижче відображено досліджувану моделі до підключення живлення та у процесі роботи.



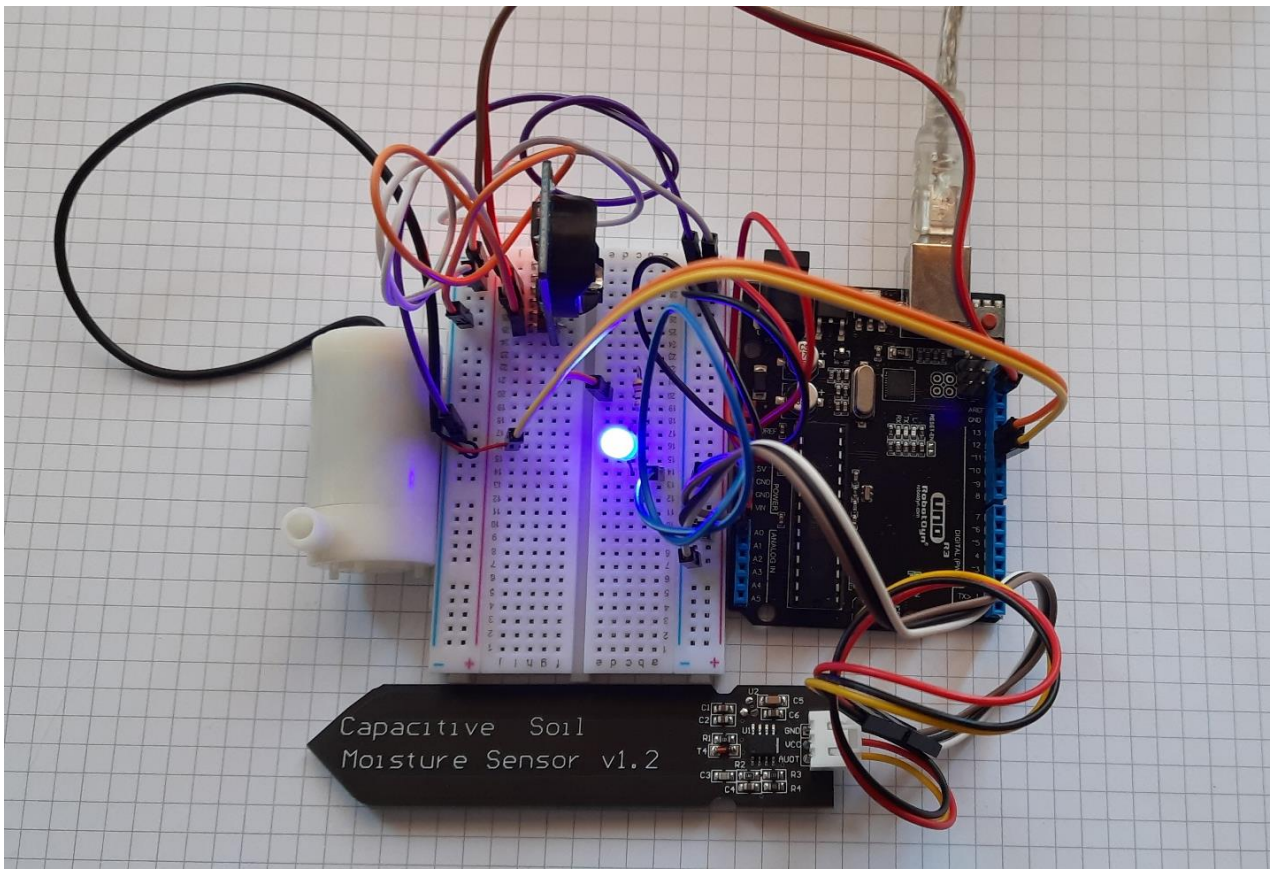


Рисунок 4.1 – Відображення роботи дослідного зразка електронної системи контролю вологості ґрунту

Зокрема фактичного відображення роботи системи через вмикання насосу і світлодіода, справність роботи системи також можна спостерігати за допомогою послідовного порту програми Arduino IDE. В послідовний порт, за допомогою попередньо прописаних команд в програмному коді системи, передаються значення:

- реальна дата та час згідно до місцевого часу – «Today : »
- з датчика вологості – «Humidity : »
- вік рослини в днях – «Day : »
- час роботи насосу в секундах – «Run time (sec.) = »

```
Initialize DS3231
Today: 2021-11-8 13:15:34
Humidity:83.00    Day: 1.00   Run time (sec.) = 0.28
Today: 2021-11-8 13:15:36
Humidity:83.00    Day: 1.00   Run time (sec.) = 0.28
Today: 2021-11-8 13:15:37
Humidity:82.00    Day: 1.00   Run time (sec.) = 0.28
Today: 2021-11-8 13:15:39
Humidity:80.00    Day: 1.00   Run time (sec.) = 0.29
Today: 2021-11-8 13:15:41
Humidity:79.00    Day: 1.00   Run time (sec.) = 0.29
Today: 2021-11-8 13:15:42
Humidity:77.00    Day: 1.00   Run time (sec.) = 0.30
Today: 2021-11-8 13:15:44
Humidity:75.00    Day: 1.00   Run time (sec.) = 0.30
Today: 2021-11-8 13:15:46
Humidity:74.00    Day: 1.00   Run time (sec.) = 0.30
Today: 2021-11-8 13:15:48
Humidity:73.00    Day: 1.00   Run time (sec.) = 0.30
Today: 2021-11-8 13:15:49
Humidity:78.00    Day: 1.00   Run time (sec.) = 0.29
Today: 2021-11-8 13:15:51
Humidity:77.00    Day: 1.00   Run time (sec.) = 0.30
Today: 2021-11-8 13:15:53
Humidity:75.00    Day: 1.00   Run time (sec.) = 0.30
Today: 2021-11-8 13:15:54
Humidity:81.00    Day: 1.00   Run time (sec.) = 0.29
Todav: 2021-11-8 13:15:56
Humidity:84.00    Day: 1.00   Run time (sec.) = 0.27
Today: 2021-11-8 13:16:26
Humidity:92.00    Day: 1.00   Run time (sec.) = 0.00
Today: 2021-11-8 13:16:28
Humidity:94.00    Day: 1.00   Run time (sec.) = 0.00
Today: 2021-11-8 13:16:29
Humidity:94.00    Day: 1.00   Run time (sec.) = 0.00
Today: 2021-11-8 13:16:31
Humidity:94.00    Day: 1.00   Run time (sec.) = 0.00
Today: 2021-11-8 13:16:32
Humidity:95.00    Day: 1.00   Run time (sec.) = 0.00
Today: 2021-11-8 13:16:33
Humidity:95.00    Day: 1.00   Run time (sec.) = 0.00
Today: 2021-11-8 13:16:35
Humidity:95.00    Day: 1.00   Run time (sec.) = 0.00
Today: 2021-11-8 13:16:36
Humidity:93.00    Day: 1.00   Run time (sec.) = 0.00
Today: 2021-11-8 13:16:38
Humidity:81.00    Day: 1.00   Run time (sec.) = 0.29
```

Рисунок 4.2 – Відображення інформації яку було виведено з послідовного порту під час тестування досліджуваного макету

Як видно з результатів розробки – система справно функціонує та повністю виконує всі поставлені для неї задачі.

4.2 Перспективи розвитку системи та майбутні дослідження

Стосовно самого макету та математичної моделі системи слід зазначити такі проблеми які потребують доробки та модернізації:

- Математична модель
  - розрахунок проводився для одного виду комірки зі своїми параметрами, в подальшому бажано б було розширити базу різновидів комірок та площ яку будуть займати рослини.
  - Через невеликий розмір засаджуваної комірки було знехтувано розрахунком трьох осьового поширення вологи, та не було враховано коефіцієнт випаровування вологи з ґрунту.
  - Розрахунок часу роботи насосу було проведено для насосу потужністю 2 л/хв, через це було отримано відносно невеликий час роботи насосу. Часте та короткочасне вмикання/вимикання роботи насосу можуть швидко використати його працездатний ресурс та призвести до поломки насосу. Рішенням цього питання буде заміна насосу на менш потужний з, відповідно, більшим часом роботи у ввімкненому стані.
  - Розрахунок загального часу роботи насосу було проведено для однієї комірки з рослиною, в подальшому необхідно провести розрахунок і дослідження для більш масштабного формату, для касети з рослинами, для декількох касет та для повноцінної теплиці.
- Програмний код та реальна модель
  - Масштабованість системи, в дослідженні цієї роботи було створено код і модель для однієї комірки та одного датчика вологості. В подальших дослідженнях та розробках постає ціль покрокового масштабування моделі від касети до теплиці.
  - Можлива заміна мікроконтролерної плати на більш потужну та з більшим об’ємом пам’яті. Це обумовлено перш за все збільшенням масштабу системи та програмного коду. Для моделювання досліджуваної системи скоріш за все плати ArduinoUNO буде недостатньо тому в процесі

дослідження постане питання переносу системи на більш потужну плату ArduinoMEGA або навіть мікрокомп’ютер Raspberry PI.

- Додавання зовнішнього джерела живлення для насосу. Під час дослідження реальної моделі було визначено що внутрішньої напруги живлення з мікроконтролера недостатньо для повноцінного живлення насосу. Рішенням цього питання буде додавання зовнішнього джерела напруги та керування насосом через розрив ланцюга живлення. На вихідний пін керування насосом буде підключено дворівневе реле вмикання з закорочуванням чи розривом проводу живлення насосу.
- Додавання розширеного функціоналу для керування системою та індікації стану системи і ґрунту. Для керування системою в перспективі розробки є додавання клавіатури. Це допоможе налаштовувати систему на вибір певного виду рослини та доби її розвитку. Для індікації стану системи мінімум буде додано РК екран відображення вологості ґрунту, типу і віку рослини та час останнього вмикання насосу.

- Довгострокові цілі та перспективи розвитку системи
  - Створення певного переліку баз правил для різних типів рослин, ґрунтів та ємностей в яких буде вирощуватися рослина
  - Створення зовнішнього додатку для моніторингу та керування системою, наприклад через хмарне сховище або через мобільний додаток.
  - Створення повністю автоматизованої системи контролю та керуванням кліматом у теплиці на базі нечіткої логіки.

ВИСНОВОК

Розробка повністю або частково нової автоматичної системи різного плану це важлива ланка розвитку автоматики та виробництва на підприємствах. Сільськогосподарська галузь дуже потребує впровадження автоматичних засобів моніторингу та керування різними процесами вирощування та догляду за рослинами і тваринами, обробки та підготовки ґрунтів і різного виду сировини, обробки та фасування готової продукції та інше.

Сама ідея розробки електронної системи контролю вологості тепличних ґрунтів не була новою, але проаналізувавши існуючі системи моніторингу та контролю вологості було прийнято рішення створення трохи нової, в якомусь сенсі, інакшої системи моніторингу та контролю вологості ґрунтів. Результат аналізу виявив що майже ніхто не використовував системи контролю на базі правил нечіткої логіки, тому було прийняте рішення зайнятися розробкою та дослідженням такого типу системи.

Ще однією причиною використання нечіткої логіки було зменшення перевитрачання ресурсів води та забезпечення найліпших умов для вирощування рослини.

Із вже існуючих систем моніторингу та контролю було взято існуючі методи вимірювання вологості ґрунту та крапельний полив рослини. Окрім цього важливою частиною розробки системи було звертання уваги на потребу рослиною різного відсотка вологості ґрунту в залежності від віку рослини. Це зумовлено тим, що в різні етапи росту рослина потребує різної кількості вологи та поживних речовин. Опираючись на увесь перелік необхідних функцій системи було створено алгоритм роботи та принцип взаємодії елементів системи. Алгоритм, у загальному вигляді, мало чим відрізняється від вже існуючих систем, але його головною відзнакою є використання тієї самої бази правил нечіткої логіки. Це означає що в контролерну частину системи розробником завантажується перелік різнопланових співвідношень вхідної інформації та окремих правил реакції, вихідного сигналу, на кожну комбінацію вхідної

інформації. Вихідний сигнал створюється через обробку контролером всіх можливих варіантів та генерує свій, майже неповторний результат вихідної реакції. Ця реакція, в подальшому, оброблюється вже в необхідний для керування сигнал, та дає команду на керований елемент (в даній роботі насос).

Для створення всієї бази правил нечіткої логіки попередньо було проведено дослідження росту рослини від пророщення до готової до пересадження розсади. На основі отриманих даних було розроблено математичну модель з розрахунком часу поглинання та зміни відсотка вологості ґрунту на різних глибинах а також час роботи насосу із співвідношення об’єму ємності з грунтом та наливною потужністю насосу.

Для роботи системи і зокрема насосу було створено програмний код моніторингу та керування вологості тепличного ґрунту. Проведено випробування на імітаційних моделях та реалізація досліджуваного зразка системи. Результати випробовувань показали що система повністю справна та виконує свою функцію, зокрема було визначено і подальші кроки для доробки та модернізації системи. В цілому ідея даної роботи була повністю реалізована та подає великі сподівання для подальшого дослідження і розвитку системи.

Використання нечіткої логіки в системі автоматизації різних процесів дає змогу більш точно та плавно керувати виробничими процесами у теплиці, господарстві та виробництві в цілому а також вона має великий потенціал у майбутніх дослідженнях і розробках різного плану систем.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Офіційний сайт розробників системи від т.м. «Hortau» [електронний ресурс] // Режим доступу до ресурсу: https://hortau.com/
2. Офіційний сайт розробників системи від т.м. «Tule» [електронний ресурс] // Режим доступу до ресурсу: https://tuletechnologies.com/
3. Офіційний сайт розробників системи від т.м. «Tevatronic» [електронний ресурс] // Режим доступу до ресурсу:http://tevatronic.net/
4. Методи вимірювання вологості ґрунту [електронний ресурс] / Богдан Малиновський // - 2018. – Режим доступу до ресурсу: https://propozitsiya.com/ua/metody-vymiryuvannya-vologosti-gruntu
5. Методи і засоби агрометеорологічних вимірювань параметрів ґрунтів [електронний ресурс] // Режим доступу до ресурсу: https://uhmi.org.ua/rozr/agro/index.php#Z1
6. Larry Pitts (2016) Monitorynh volohosti gruntu dlya optymal′noho zrostannya vrozhayu [Monitoring Soil Moisture for Optimal Crop Growth] (Electronic resource), Available at: https://observant.zendesk.com/hc/en-us/articles/208067926-Monitoring-Soil-Moisture-for-Optimal-Crop-Growth
7. Вимірювання вологості [електронний ресурс] // Режим доступу до ресурсу: https://studopedia.com.ua/1_125406_vimiryuvannya-vologosti.html
8. Комп’ютеризована система комплексного моніторингу й керування мікрокліматом промислових теплиць на базі нечіткої логіки / [Лактіонов І. С., Вовна О. В., Бережний М. О., Лебедєв В. А.]. – К. : Вістник КрНУ імені Михайла Остроградського. Випуск 3/2019 (116). – 120 с.
9. Обґрунтування плану експериментальних досліджень комп'ютеризованої системи моніторингу та керування зволоженням тепличних культур / [Лактіонов І. С., Лебедєв В. А., Лактіонова Г. А., Бережний М. О.]. – Наукові праці ДонНТУ. Серія “Обчислвальна техніка та автоматизація”. – Покровськ. 2019. - Випуск №1(32). – 103-113 с.
10. Лактіонов, І.С. Методика та результати лабораторних випробувань

комп'ютеризованого вимірювача вологості тепличних ґрунтів / І. С. Лактіонов, О. В. Вовна, В. А. Лебедєв // Вісник ЖДТУ. Сер. Технічні науки. – Житомир, 2018. – № 1 (81). – С. 136 – 143.

11. Лактіонов, І. С. Комп'ютеризована технологія агрегації та обробки результатів градуювання вимірювача вологості тепличних ґрунтів / І. С. Лактіонов, В. А. Лебедєв // Наук. пр. Донецьк. нац. техн. ун-ту. Сер. Обчислювальна техніка та автоматизація. – Покровськ, 2018. – Вип. 1 (31). – С. 97 – 107.
12. Arduino.ua. [Електронний ресурс]: Ємнісний сенсор вологості ґрунту від DFRobot. – Режим доступу: https://arduino.ua/prod3255-emkostnii-datchik-vlajnosti-pochvi-ot-dfrobot. – Назва з титул. екрана.
13. Produkty vymiryuvannya volohosti - z·hidno do rehionu [Soils Moisture Products - Select Region] (Electronic resource), Available at: http://www.ictinternational.com/products/soils/moisture-sensors/
14. Sensor volohosti hruntu SM150T [SM150T Soil Moisture Sensor] (Electronic resource), Available at: https://www.delta-t.co.uk/product/sm150t/
15. Ємнісний датчик вологості від DFRobot [електронний ресурс] // Режим доступу до ресурсу: https://arduino.ua/prod3255-emkostnii-datchik-vlajnosti-pochvi-ot-dfrobot .
16. Часи реального часу DS3231 [електронний ресурс] // Режим доступу до ресурсу: https://3d-diy.ru/wiki/arduino-moduli/chasy-realnogo-vremeni-ds3231/
17. Перспективні напрямки сучасної електроніки, інформаційних і комп’ютерних систем: тези доповідей на IV Всеукр. наук.-практ. конф. MEICS-2019. – К. : м. Дніпро 2019 – 77 с.
18. Офіційний сайт Arduino [електронний ресурс] // Режим доступу до ресурсу: https://www.arduino.cc/
19. Crop Guide: Tomato Cultivation “About the crop” (Electronic resource), Available at: https://www.haifa-group.com/ru/tomato-0/crop-guide-tomato

20. Crop Guide: Tomato Cultivation “Growing conditions” (Electronic resource), Available at: https://www.haifa-group.com/ru/articles/growing-tomato-%20and-how-to-grow-tomato
21. Crop Guide: Tomato Cultivation “Plant nutrilon” (Electronic resource), Available at: https://www.haifa-group.com/ru/crop-guide/vegetables/tomato/crop-guide-tomato-plant-nutrition
22. Crop Guide: Tomato Cultivation “Fertilization recommendations” (Electronic resource), Available at: https://www.haifa-group.com/ru/tomato-0/crop-guide-tomato-fertilizer-recommendations
23. Crop Guide: Tomato Cultivation “Appendix I & II” (Electronic resource), Available at: https://www.haifa-group.com/ru/tomato-fertilizer/crop-guide-tomato-cultivation
24. Hongjun Li, Xiaopeng Zhang, Weiliang Meng and Lin Ge Visualization of Tomato Growth Based on Dry Matter Flow / Hindawi International Journal of Computer Games Technology Volume 2017, Article ID 2302731, 12 pages Available at : https://pdfs.semanticscholar.org/23e5/d1e0c74607a67e26091fcb9167ddd71494af.pdf
25. José A. Mancilla-Morales; Mario A. Tornero-Campante; Irineo L. López-Cruz. Evaluation of a mathematical model to predict growth and nitrogen content in tomatoes (Solanum lycopersicum L.) under greenhouse conditions / Ingeníeria Agrícola y Biosistemas | Vol. 11, issue 2, July-December 2019. – MEXICO, 2019. – P. 111-125.
26. Wouter J.P. Kuijpers, Marinus J.G.van de Molengraft, Simon van Mourik, Albertus van ’t Ooster, Silke Hemming, Eldert J.van Henten Model selection with a common structure: Tomato crop growth models / Biosystems Engineering Volume 187, November 2019, Pages 247-257 (Electronic resource)), Available at: https://www.sciencedirect.com/science/article/pii/S1537511019308323
27. Redmond R. Shamshiri, James W Jones, Kelly Thorp, Sima Taheri Review of optimum temperature, humidity, and vapour pressure deficit for microclimate

evaluation and control in greenhouse cultivation of tomato: A review / International Agrophysics 32(2), March 2018. – P. 287-302. (Electronic resource)), Available at: https://www.researchgate.net/publication/323225604_Review_of_optimum_temperature_humidity_and_vapour_pressure_deficit_for_microclimate_evaluation_and_control_in_greenhouse_cultivation_of_tomato_A_review

28. Фізика ґрунту / Нерпін С.В., Чудновський А.Ф. – Москва: Видавництво «Наука», 1967. – 584 с.
29. Лактіонов І.С. Комп'ютеризована інформаційно-вимірювальна система фізичних параметрів ґрунтів промислових теплиць з компенсацією дестабілізуючих впливів : Дисертація на здобуття наукового ступеня кандидата технічних наук / І.С. Лактіонов, О.В. Вовна – Красноармійськ :ДВНЗ «ДонНТУ», 2015. – 225 с.
30. Вологість ґрунту та її значення для розвитку культур [електронний ресурс] // Режим доступу до ресурсу: https://eos.com/ru/blog/vlazhnost-pochvy/#:~:text=Формула%20Влажности%20Почвы&text=Содержание%20влаги%20в%20почве%20по,г%20см3)%5D%20x%20100.
31. Лактіонов, І.С. Методика та результати лабораторних випробувань комп'ютеризованого вимірювача вологості тепличних ґрунтів / І. С. Лактіонов, О. В. Вовна, В. А. Лебедєв // Вісник ЖДТУ. Сер. Технічні науки. – Житомир, 2018. – № 1 (81). – С. 136 – 143.
32. Arduino FIST: Systema nechitkoyi lohiky MATLAB dlya peretvorennya v Arduino C [Arduino FIST: MATLAB Fuzzy Inference System to Arduino C Converter] (Electronic resource), Available at: http://www.makeproto.com/projects/fuzzy/matlab_arduino_FIST/index.php

ДОДАТОК А

Охорона праці та безпека під час надзвичайних ситуаціях на підприємстві

До приміщення науково-дослідного відділу й організації робочого місця з обліком шкідливих виробничих факторів пред'являється ряд вимог. Приміщення в якому знаходиться робоче місце з ПК повинно мати природне освітлення, бажано з однобічним розміщенням світопрорізів, площа осклянілості яких не повинна перевищувати 25 % від площі стіни світопрорізами. Віконні прорізи в приміщенні з ПК повинні мати регульовані жалюзі чи занавеси або інші сонцезахисні пристрої. Не допускається розташування робочих місць із ПК у підвальних і цокольних поверхах. Робочі місця з ПК рекомендується розміщати в окремих приміщеннях. Площа на одного працюючого з ПК повинна складати 6 м^2, об’єм – 20 м^3. Неприпустиме розташування ПК, при якому працюючий звернений обличчям, або спиною до вікон чи кімнати задньої частини ПК, у яку монтуються вентилятори.

Забороняється застосовувати для обробки інтер'єра приміщень із ПК полімерні матеріали (дерев’яностружечні плити, шпалери що миються, плівкові та рулоні синтетичні матеріали, шаруватий паперовий пластик та ін.), що виділяються в повітря шкідливі хімічні речовини, що перевищують гранично допустимі концентрації, не включені в «Перелік дозволених , МЗ» 1977-1985 р.

В лабораторії вимірювальної техніки та науково-дослідної роботи робочі місця з ПК розташовані від стіни з вікнами на відстані 1 м, відстань між столами складає 3 м. Екрани відеомоніторів ПК знаходяться від очей користувача на відстані 700 мм відповідно до СН 512-78, приміщення (S=21 м^2, V=73,5 м^3) дозволяє розташовувати більше 3 робочих місця.

Робочі місця в положенні сидячі відповідають вимогам ДСТ 12.2.032 – 78 та ДСТ 12.2.029 – 77. Поверхня робочого столу знаходиться на висоті 0,75 метрів від підлоги, розміри робочої поверхні стільниці складають 1050х590 міліметрів, розміри вільного простору для ніг під столом складає висота 650, глибина 550, ширина 450 міліметрів відповідно. Робочий стілець оснащений підйомно-

поворотнім пристроєм, що забезпечує регуляцію висоти сидіння і спинки, пневматичним і гідравлічними амортизаторами та обладнанні підлокітниками.

А.1 Мікроклімат робочого місця

У приміщенні науково-дослідного відділу є джерела тепловиділення, тому необхідно визначити необхідні умови його вентилювання. Витрату повітря в приміщенні з додатковим тепловиділенням визначаємо за формулою:

$$L = \frac{Q_{НАД}}{c \cdot p \cdot (t_B - t_H)}, \quad (А.1)$$

де $Q_{НАД}$ – надлишкове виділення тепла в робочому приміщенні, ккал/год.; c – теплоємкість повітря (0,237 ккал/кг);

p – обсягова вага повітря (1,226 кг/м^3);

t_B – температура витяжного повітря (30°С);

t_H – температура приточного повітря (20°С).

Розрахуємо надлишкове надходження тепла за наступною формулою:

$$Q_{НАД} = Q_{УСТ} + Q_{ПЕР} + Q_{ОСВ} + Q_{СР}, \quad (А.2)$$

де $Q_{УСТ}$ – виділення тепла від устаткування;

$Q_{ПЕР}$ – виділення тепла робітниками;

$Q_{ОСВ}$ – надходження тепла від електричного освітлення;

$Q_{СР}$ – надходження тепла від сонячної радіації через вікна.

Визначимо виділення тепла від устаткування за формулою:

$$Q_{УСТ} = P \cdot K_a \cdot K_б \cdot 860, \quad (А.3)$$

де P – сумарна потужність устаткування, кВт/год;

$K_а$ – коефіцієнт установленої потужності (0,95);

$K_б$ – коефіцієнт одночасної роботи (1,0).

$$Q_{УСТ} = [x_1 \cdot k_1 + x_2 \cdot k_2 + x_3 \cdot k_3 + x_4 \cdot k_4 + x_5 \cdot k_5 + x_6 \cdot k_6 + x_7 \cdot k_7 + x_8 \cdot k_8 + x_9 \cdot k_9] \cdot K_а \cdot K_б \cdot 860 =$$
$$= [1 \cdot 0{,}5 + 1 \cdot 0{,}1 + 1 \cdot 0{,}07 + 2 \cdot 0{,}4 + 1 \cdot 0{,}06 + 1 \cdot 0{,}05 + 1 \cdot 0{,}6 + 4 \cdot 0{,}15 + 1 \cdot 3{,}5] \cdot 0{,}95 \cdot 1 \cdot 860 = 5131 \text{ ккал/год.}$$

Визначимо виділення тепла від обслуговуючого персоналу за допомогою наступної формули:

$$Q_{ПЕР} = n \cdot g = 2 \cdot 100 = 200 \text{ ккал/год,} \qquad (А.4)$$

де n – кількість працюючих;

g – кількість тепла, що виділяє один працівник за годину (100 ккал/год.).

Визначимо надходження тепла від електричного освітлення за формулою:

$$Q_{ОСВ} = E_М \cdot g_1 \cdot S = 300 \cdot 0{,}05 \cdot 21 = 315 \text{ ккал/год,} \qquad (А.5)$$

де $E_М$ – нормована освітленість для цієї зорової роботи, величина якої дорівнює 300 лк;

g_1 – питоме тепловиділення на 1 м^2 підлоги при 1 лк освітленості (для / люмінесцентних ламп – 0,05 ккал/год.);

S – площа приміщення, м^2.

Визначимо надходження тепла від сонячної радіації через вікна за наступною формулою:

$$Q_{СР} = F \cdot g_2 \cdot K_{ОСЛ} = 7{,}5 \cdot 65 \cdot 0{,}4 = 195 \text{ ккал/год,} \qquad (А.6)$$

де F – площа віконних прорізів (3х2,5=7,5 м^2);

g_2 – кількість тепла, що надходить через 1 м^2 віконного прорізу (65 ккал/год.);

$K_{ОСЛ}$ – коефіцієнт ослаблення, приймаємо 0,4.

Визначимо кількість надлишкового тепла:

$$Q_{НАД} = Q_{УСТ} + Q_{ПЕР} + Q_{ОСВ} + Q_{СР} = 5131 + 200 + 315 + 195 = 5841 \text{ккал/год.}$$

Визначимо витрати повітря в приміщенні:

$$L = \frac{Q_{НАД}}{c \cdot p \cdot (t_В - t_Н)} = \frac{5848}{0{,}237 \cdot 1{,}226 \cdot (30 - 20)} = 2010 \text{ м}^3\text{/ год.}$$

Існуюча в наявності система кондиціонування та вентилювання має продуктивність 2200 куб. м./годину, що задовольняє необхідним нормативам.

Параметри мікроклімату на робочих місцях регламентуються ДНАОІІ 0.03.3.15 – 86 «Санітарні норми мікроклімату виробничих приміщень № 4088–86». Відповідно доданих санітарних норм температура повітря, швидкість руху повітря та відносна вологість у холодні періоди року повинна складати (22 – 24) градуса за Цельсієм, 0,1 метра в секунду та 40-60 % відповідно. При збереженні всім параметрів можливе коливання температури від 21 до 25 градусів Цельсія. У теплі періоди року температура повітря повинна складати (23 – 25) градусів Цельсія, рухливість повітря (0,1 – 0,2) метрів секунду, вологість (40 – 60) %. Температура може коливатися від 22 до 26 градусів Цельсія при збереженні всіх інших параметрів мікроклімату. Вище зазначені норми цілком відповідають фактичним даним приміщення лабораторії вимірювальної техніки та науково-дослідним відділом.

А.2.2 Розрахунок системи загального рівномірного освітлення з лампами розжарювання для приміщення, в якому використовуються зорові роботи

високої точності

Розміри приміщення: довжина (a=6 м), ширина (b=3,5 м), висота (h=3,5 м). Приміщення має світлу побілку: коефіцієнт відбиття $P_{стелі}$ = 70 %, $P_{стін}$ = 50 %. Висота робочих поверхонь (столів) h_p = 0,7 м. Для освітлення прийнято світильники типу УПМ-15, які підвищуються до стелі, відстань від світильника до стелі h_c =0,4 м. Мінімальна освітленість за нормами E=200 лк.

1) Визначимо висоту підвісу світильників над підлогою

$$h_0 = H - h_c = 3{,}5 - 0{,}4 = 3{,}1 \text{ м.}$$

Для світильників загального освітлення з лампами розжарювання потужністю до 200 Вт мінімальна висота підвісу над підлогою відповідно до СНІП П-4-79 повинна бути у межах (2,5 –4,0) м, залежно від характеристики світильника. В лабораторії вимірювальної техніки та науково-дослідному відділу відповідає цій вимозі.

2) Визначимо висоту підвісу світильника над робочою поверхнею:

$$h = h_0 - h_p = 3{,}5 - 0{,}4 = 2{,}8 \text{ м.}$$

Рівномірність освітлення досягається при відповідному співвідношенні відстані між світильниками (L) та висоти їх підвісу (h).

3) Визначимо рекомендовану відстань між світильниками

$$L = 0{,}7 \cdot h = 0{,}7 \cdot 2{,}8 = 2{,}0 \text{ м.}$$

4) Розрахуємо необхідну кількість світильників

$$N = \frac{a \cdot b}{L^2} = \frac{6 \cdot 3{,}5}{2^2} = 6.$$

Приймаємо 6 світильників, враховуючи розміри приміщення розміщуємо їх у два ряди по 3 штуки.

5) Світловий потік лампи світильника $(\Phi_{Л})$ визначається за формулою:

$$\Phi_{Л} = \frac{E \cdot K_{З} \cdot S \cdot Z}{N \cdot n \cdot \eta},$$

де E – нормативна освітленість, лк;

$K_{З}$ – коефіцієнт запасу, який враховує зниження освітленості в результаті забруднення та старіння ламп;

S – площа приміщення, що освітлюється, м^2;

Z – коефіцієнт нерівномірності освітлення для ламп розширювання (1,15);

N – кількість світильників;

n – кількість ламп у світильнику;

η – коефіцієнт використання світового потоку, який визначається за світлотехнічними таблицями залежно від показника приміщення (i) та коефіцієнтів відбиття стін та стелі.

6) Визначимо показник приміщення:

$$i = \frac{a \cdot b}{h \cdot (a + b)} = \frac{6 \cdot 3{,}5}{3{,}5 \cdot (6 + 3{,}5)} = 2{,}21.$$

Коефіцієнт використання η = 0,48 для світильника УПМ-15 (при i= 2,5, $Р_{стелі}$=70%, $Р_{стін}$ = 50 %)

Світловий потік одного світильника, а значить і лампи, оскільки за конструктивним виконанням у світильнику цього типу встановлена лише одна лампа, дорівнює:

$$\Phi_{Л} = \frac{E \cdot S \cdot Z}{N \cdot n \cdot \eta} = \frac{200 \cdot 21 \cdot 1{,}15}{6 \cdot 1 \cdot 0{,}48} = 1677 \text{ лм.}$$

9) обираємо лампу Б-150 потужністю 150 Вт, світловий потік якої становить 2100 лм. Хоча це значення на 18% більше розрахованого, однак не перевищує встановлену норму (– 0 %< $\Phi_{Л}$ <+20 %). Сумарна електрична потужність усіх світильників, встановлених у приміщенні становить:

$$P_{СВ} = P \cdot N = 150 \cdot 6 = 900 \text{ Вт.}$$

ДОДАТОК Б
Лістинг програми в ППП MATHCAD "математичне моделювання зміни вологості тепличного грунту

Початкова вологість 30%

D ≡ 0.8

L ≡ 1

T ≡ 100

x := 0, 0.1.. 1

Given

$u_t(x,t) = D \cdot u_{xx}(x,t)$

u(0, t) = 3

u(x, 0) = 0

u(L, t) = 0

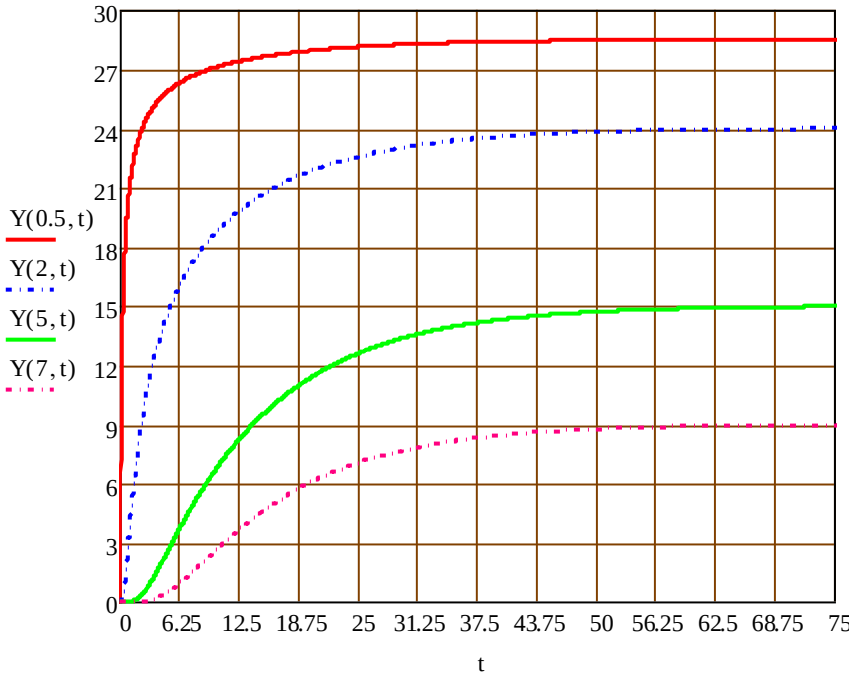
T ≡ 75

L ≡ 1

D ≡ 0.8

$Y := \text{Pdesolve}\left[u, x, \begin{pmatrix} 0 \\ L \end{pmatrix}, t, \begin{pmatrix} 0 \\ T \end{pmatrix}\right]$

t := 0, 0.01.. 9



Початкова вологість 60%

Given

$u_t(x,t) = D \cdot u_{xx}(x,t)$

$u(0,t) = 60$

$u(x,0) = 0$

$u(L,t) = 0$

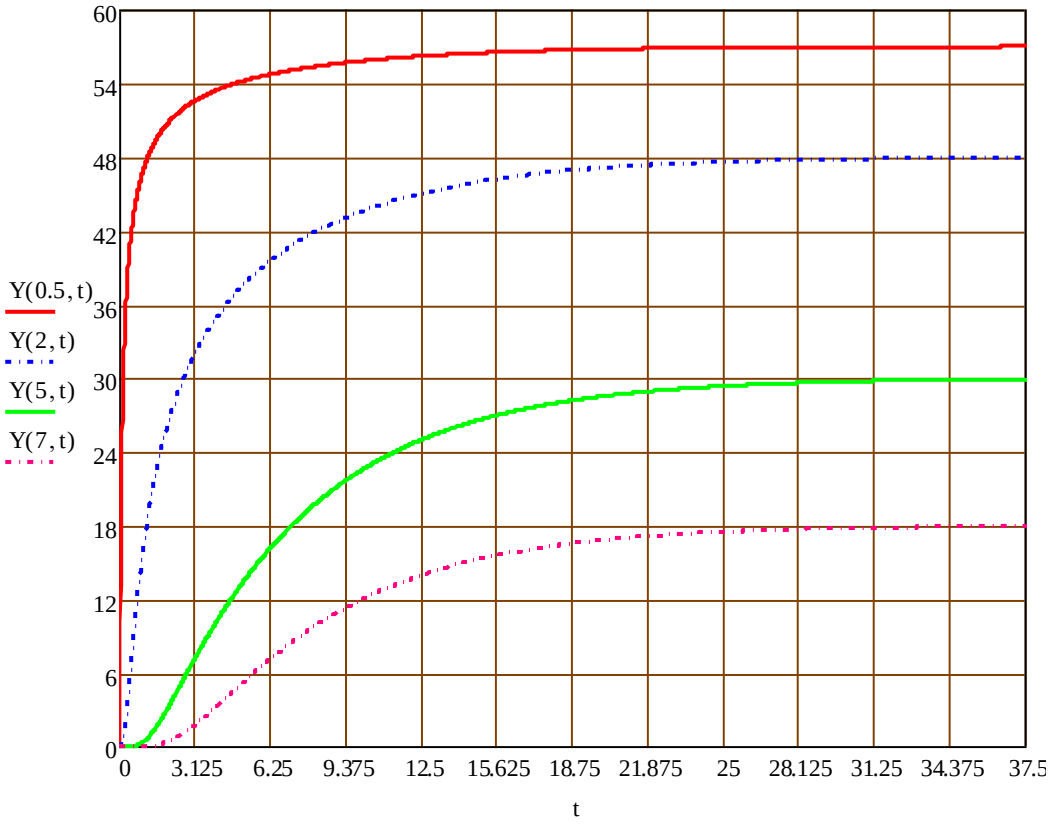
$T \equiv 570$

$L \equiv 10$

$D \equiv 1.64$

$Y := \text{Pdesolve}\left[u, x, \begin{pmatrix}0\\L\end{pmatrix}, t, \begin{pmatrix}0\\T\end{pmatrix}\right]$

$t := 0, 0.01 .. 90$



Початкова вологість 90%

Given

$u_t(x,t) = D \cdot u_{xx}(x,t)$

$u(0,t) = 90$

$u(x,0) = 0$

$u(L,t) = 0$

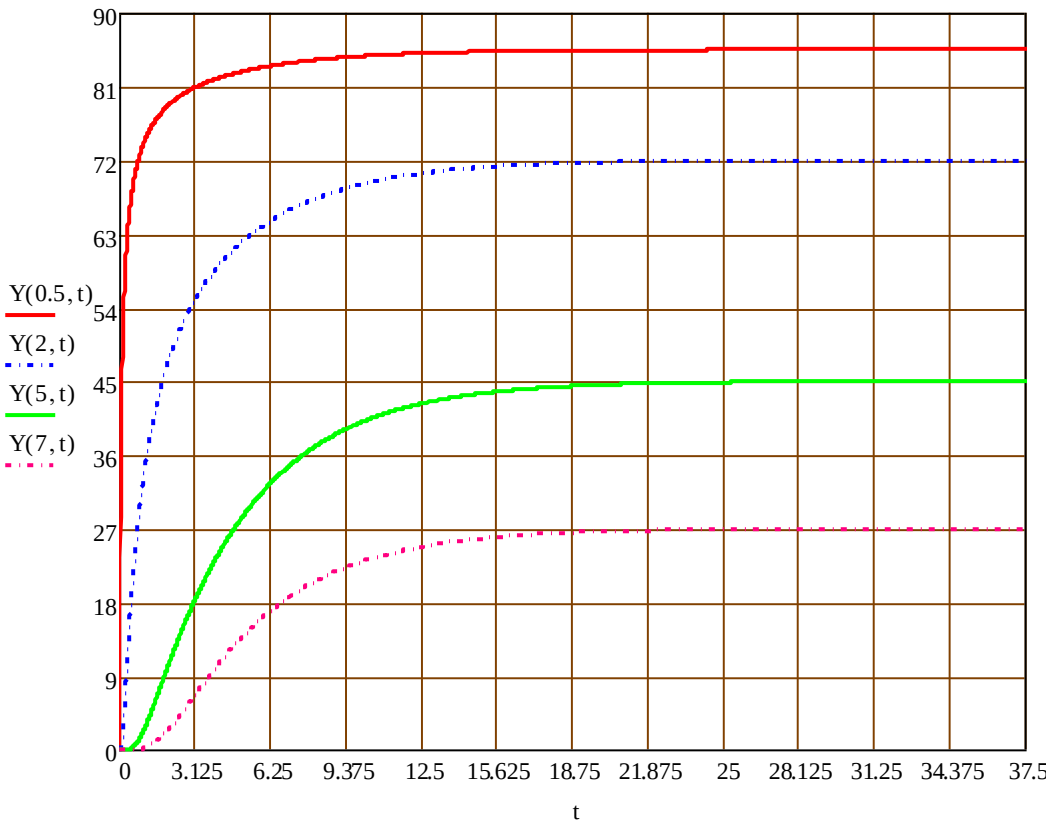
$T \equiv 400$

$L \equiv 10$

$D \equiv 2.46$

$Y := \text{Pdesolve}\left[u, x, \begin{pmatrix}0\\L\end{pmatrix}, t, \begin{pmatrix}0\\T\end{pmatrix}\right]$

$t := 0, 0.01 .. 90$



ДОДАТОК В

Початковий код після компіляції

```
//*****************************************************************
#include "fis_header.h"

// Number of inputs to the fuzzy inference system
const int fis_gcI = 2;
// Number of outputs to the fuzzy inference system
const int fis_gcO = 1;
// Number of rules to the fuzzy inference system
const int fis_gcR = 85;

FIS_TYPE g_fisInput[fis_gcI];
FIS_TYPE g_fisOutput[fis_gcO];
// Setup routine runs once when you press reset:
void setup()
{
   // initialize the Analog pins for input.
   // Pin mode for Input: Humidity
   pinMode(0 , INPUT);
   // Pin mode for Input: Day
   pinMode(1 , INPUT);

   // initialize the Analog pins for output.
   // Pin mode for Output: Time
   pinMode(2 , OUTPUT);

}
```

```
// Loop routine runs over and over again forever:
void loop()
{
   // Read Input: Humidity
   g_fisInput[0] = analogRead(0);
   // Read Input: Day
   g_fisInput[1] = analogRead(1);

   g_fisOutput[0] = 0;

   fis_evaluate();

   // Set output vlaue: Time
   analogWrite(2 , g_fisOutput[0]);

}
//************************************************************************
// Support functions for Fuzzy Inference System
//************************************************************************
// Trapezoidal Member Function
FIS_TYPE fis_trapmf(FIS_TYPE x, FIS_TYPE* p)
{
   FIS_TYPE a = p[0], b = p[1], c = p[2], d = p[3];
   FIS_TYPE t1 = ((x <= c) ? 1 : ((d < x) ? 0 : ((c != d) ? ((d - x) / (d - c)) : 0)));
   FIS_TYPE t2 = ((b <= x) ? 1 : ((x < a) ? 0 : ((a != b) ? ((x - a) / (b - a)) : 0)));
   return (FIS_TYPE) min(t1, t2);
}

// Triangular Member Function
FIS_TYPE fis_trimf(FIS_TYPE x, FIS_TYPE* p)
```

```
{
    FIS_TYPE a = p[0], b = p[1], c = p[2];
    FIS_TYPE t1 = (x - a) / (b - a);
    FIS_TYPE t2 = (c - x) / (c - b);
    if ((a == b) && (b == c)) return (FIS_TYPE) (x == a);
    if (a == b) return (FIS_TYPE) (t2*(b <= x)*(x <= c));
    if (b == c) return (FIS_TYPE) (t1*(a <= x)*(x <= b));
    t1 = min(t1, t2);
    return (FIS_TYPE) max(t1, 0);
}

FIS_TYPE fis_min(FIS_TYPE a, FIS_TYPE b)
{
    return min(a, b);
}

FIS_TYPE fis_max(FIS_TYPE a, FIS_TYPE b)
{
    return max(a, b);
}

FIS_TYPE fis_array_operation(FIS_TYPE *array, int size, _FIS_ARR_OP pfnOp)
{
    int i;
    FIS_TYPE ret = 0;

    if (size == 0) return ret;
    if (size == 1) return array[0];

    ret = array[0];
```

```
    for (i = 1; i < size; i++)
    {
        ret = (*pfnOp)(ret, array[i]);
    }

    return ret;
}

//*********************************************************************
// Data for Fuzzy Inference System
//*********************************************************************
// Pointers to the implementations of member functions
_FIS_MF fis_gMF[] =
{
    fis_trapmf, fis_trimf
};

// Count of member function for each Input
int fis_gIMFCount[] = { 5, 17 };

// Count of member function for each Output
int fis_gOMFCount[] = { 10 };

// Coefficients for the Input Member Functions
FIS_TYPE fis_gMFI0Coeff1[] = { 15.95, 22.01, 32.99, 39.05 };
FIS_TYPE fis_gMFI0Coeff2[] = { 60.87, 75, 89.13 };
FIS_TYPE fis_gMFI0Coeff3[] = { 45.87, 60, 74.13 };
FIS_TYPE fis_gMFI0Coeff4[] = { 30.87, 45, 59.13 };
FIS_TYPE fis_gMFI0Coeff5[] = { 82.5, 89.92, 107.7, 115.1 };
FIS_TYPE* fis_gMFI0Coeff[] = { fis_gMFI0Coeff1, fis_gMFI0Coeff2,
```

```
fis_gMFI0Coeff3, fis_gMFI0Coeff4, fis_gMFI0Coeff5 };
FIS_TYPE fis_gMFI1Coeff1[] = { -10.8, -1.2, 0.9, 1 };
FIS_TYPE fis_gMFI1Coeff2[] = { 0.9, 1.1, 1.9, 2 };
FIS_TYPE fis_gMFI1Coeff3[] = { 1.9, 2.1, 2.9, 3 };
FIS_TYPE fis_gMFI1Coeff4[] = { 2.9, 3.1, 3.9, 4 };
FIS_TYPE fis_gMFI1Coeff5[] = { 3.9, 4.1, 4.9, 5 };
FIS_TYPE fis_gMFI1Coeff6[] = { 4.9, 5.1, 5.9, 6 };
FIS_TYPE fis_gMFI1Coeff7[] = { 5.9, 6.1, 6.9, 7 };
FIS_TYPE fis_gMFI1Coeff8[] = { 6.9, 7.1, 7.9, 8 };
FIS_TYPE fis_gMFI1Coeff9[] = { 7.9, 8.1, 8.9, 9 };
FIS_TYPE fis_gMFI1Coeff10[] = { 8.9, 9.1, 9.9, 10 };
FIS_TYPE fis_gMFI1Coeff11[] = { 9.9, 10.1, 11.9, 12 };
FIS_TYPE fis_gMFI1Coeff12[] = { 11.9, 12.1, 13.9, 14 };
FIS_TYPE fis_gMFI1Coeff13[] = { 13.9, 14.1, 15.9, 16 };
FIS_TYPE fis_gMFI1Coeff14[] = { 15.9, 16.1, 17.9, 18 };
FIS_TYPE fis_gMFI1Coeff15[] = { 17.9, 18.1, 19.9, 20 };
FIS_TYPE fis_gMFI1Coeff16[] = { 19.9, 20.1, 24.9, 25 };
FIS_TYPE fis_gMFI1Coeff17[] = { 24.9, 25.1, 32, 32 };
FIS_TYPE* fis_gMFI1Coeff[] = { fis_gMFI1Coeff1, fis_gMFI1Coeff2,
fis_gMFI1Coeff3, fis_gMFI1Coeff4, fis_gMFI1Coeff5, fis_gMFI1Coeff6,
fis_gMFI1Coeff7, fis_gMFI1Coeff8, fis_gMFI1Coeff9, fis_gMFI1Coeff10,
fis_gMFI1Coeff11, fis_gMFI1Coeff12, fis_gMFI1Coeff13, fis_gMFI1Coeff14,
fis_gMFI1Coeff15, fis_gMFI1Coeff16, fis_gMFI1Coeff17 };
FIS_TYPE** fis_gMFICoeff[] = { fis_gMFI0Coeff, fis_gMFI1Coeff };

// Coefficients for the Output Member Functions
FIS_TYPE fis_gMFO0Coeff1[] = { 0.734, 0.91, 1.03571428571429 };
FIS_TYPE fis_gMFO0Coeff2[] = { 1.08, 1.17, 1.31349206349206 };
FIS_TYPE fis_gMFO0Coeff3[] = { 1.6, 1.79, 2.37698412698413 };
FIS_TYPE fis_gMFO0Coeff4[] = { 0.194, 0.2, 0.483, 0.6 };
```

```
FIS_TYPE fis_gMFO0Coeff5[] = { 0.48, 0.69, 0.892857142857143 };
FIS_TYPE fis_gMFO0Coeff6[] = { 0.925, 1.05, 1.19444444444444 };
FIS_TYPE fis_gMFO0Coeff7[] = { 1.2, 1.38, 1.55952380952381 };
FIS_TYPE fis_gMFO0Coeff8[] = { 1.37, 1.59, 1.79761904761905 };
FIS_TYPE fis_gMFO0Coeff9[] = { 1.81349206349206, 2.55, 3, 3.5 };
FIS_TYPE fis_gMFO0Coeff10[] = { 0, 0, 0, 0 };
FIS_TYPE* fis_gMFO0Coeff[] = { fis_gMFO0Coeff1, fis_gMFO0Coeff2,
fis_gMFO0Coeff3, fis_gMFO0Coeff4, fis_gMFO0Coeff5, fis_gMFO0Coeff6,
fis_gMFO0Coeff7, fis_gMFO0Coeff8, fis_gMFO0Coeff9, fis_gMFO0Coeff10 };
FIS_TYPE** fis_gMFOCoeff[] = { fis_gMFO0Coeff };

// Input membership function set
int fis_gMFI0[] = { 0, 1, 1, 1, 0 };
int fis_gMFI1[] = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
int* fis_gMFI[] = { fis_gMFI0, fis_gMFI1};

// Output membership function set
int fis_gMFO0[] = { 1, 1, 1, 0, 1, 1, 1, 1, 0, 0 };
int* fis_gMFO[] = { fis_gMFO0};

// Rule Weights
FIS_TYPE fis_gRWeight[] = { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 };

// Rule Type
int fis_gRType[] = { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 };
```

```
// Rule Inputs
int fis_gRI0[] = { 5, 1 };
int fis_gRI1[] = { 2, 1 };
int fis_gRI2[] = { 3, 1 };
int fis_gRI3[] = { 4, 1 };
int fis_gRI4[] = { 1, 1 };
int fis_gRI5[] = { 5, 2 };
int fis_gRI6[] = { 2, 2 };
int fis_gRI7[] = { 3, 2 };
int fis_gRI8[] = { 4, 2 };
int fis_gRI9[] = { 1, 2 };
int fis_gRI10[] = { 5, 3 };
int fis_gRI11[] = { 2, 3 };
int fis_gRI12[] = { 3, 3 };
int fis_gRI13[] = { 4, 3 };
int fis_gRI14[] = { 1, 3 };
int fis_gRI15[] = { 5, 4 };
int fis_gRI16[] = { 2, 4 };
int fis_gRI17[] = { 3, 4 };
int fis_gRI18[] = { 4, 4 };
int fis_gRI19[] = { 1, 4 };
int fis_gRI20[] = { 5, 5 };
int fis_gRI21[] = { 2, 5 };
int fis_gRI22[] = { 3, 5 };
int fis_gRI23[] = { 4, 5 };
int fis_gRI24[] = { 1, 5 };
int fis_gRI25[] = { 5, 6 };
int fis_gRI26[] = { 2, 6 };
int fis_gRI27[] = { 3, 6 };
int fis_gRI28[] = { 4, 6 };
```

```
int fis_gRI29[] = { 1, 6 };
int fis_gRI30[] = { 5, 7 };
int fis_gRI31[] = { 2, 7 };
int fis_gRI32[] = { 3, 7 };
int fis_gRI33[] = { 4, 7 };
int fis_gRI34[] = { 1, 7 };
int fis_gRI35[] = { 5, 8 };
int fis_gRI36[] = { 2, 8 };
int fis_gRI37[] = { 3, 8 };
int fis_gRI38[] = { 4, 8 };
int fis_gRI39[] = { 1, 8 };
int fis_gRI40[] = { 5, 9 };
int fis_gRI41[] = { 2, 9 };
int fis_gRI42[] = { 3, 9 };
int fis_gRI43[] = { 4, 9 };
int fis_gRI44[] = { 1, 9 };
int fis_gRI45[] = { 5, 10 };
int fis_gRI46[] = { 2, 10 };
int fis_gRI47[] = { 3, 10 };
int fis_gRI48[] = { 4, 10 };
int fis_gRI49[] = { 1, 10 };
int fis_gRI50[] = { 5, 11 };
int fis_gRI51[] = { 2, 11 };
int fis_gRI52[] = { 3, 11 };
int fis_gRI53[] = { 4, 11 };
int fis_gRI54[] = { 1, 11 };
int fis_gRI55[] = { 5, 12 };
int fis_gRI56[] = { 2, 12 };
int fis_gRI57[] = { 3, 12 };
int fis_gRI58[] = { 4, 12 };
```

```
int fis_gRI59[] = { 1, 12 };
int fis_gRI60[] = { 5, 13 };
int fis_gRI61[] = { 2, 13 };
int fis_gRI62[] = { 3, 13 };
int fis_gRI63[] = { 4, 13 };
int fis_gRI64[] = { 1, 13 };
int fis_gRI65[] = { 5, 14 };
int fis_gRI66[] = { 2, 14 };
int fis_gRI67[] = { 3, 14 };
int fis_gRI68[] = { 4, 14 };
int fis_gRI69[] = { 1, 14 };
int fis_gRI70[] = { 5, 15 };
int fis_gRI71[] = { 2, 15 };
int fis_gRI72[] = { 3, 15 };
int fis_gRI73[] = { 4, 15 };
int fis_gRI74[] = { 1, 15 };
int fis_gRI75[] = { 5, 16 };
int fis_gRI76[] = { 2, 16 };
int fis_gRI77[] = { 3, 16 };
int fis_gRI78[] = { 4, 16 };
int fis_gRI79[] = { 1, 16 };
int fis_gRI80[] = { 5, 17 };
int fis_gRI81[] = { 2, 17 };
int fis_gRI82[] = { 3, 17 };
int fis_gRI83[] = { 4, 17 };
int fis_gRI84[] = { 1, 17 };
int* fis_gRI[] = { fis_gRI0, fis_gRI1, fis_gRI2, fis_gRI3, fis_gRI4, fis_gRI5,
fis_gRI6, fis_gRI7, fis_gRI8, fis_gRI9, fis_gRI10, fis_gRI11, fis_gRI12, fis_gRI13,
fis_gRI14, fis_gRI15, fis_gRI16, fis_gRI17, fis_gRI18, fis_gRI19, fis_gRI20,
fis_gRI21, fis_gRI22, fis_gRI23, fis_gRI24, fis_gRI25, fis_gRI26, fis_gRI27,
```

```
fis_gRI28, fis_gRI29, fis_gRI30, fis_gRI31, fis_gRI32, fis_gRI33, fis_gRI34,
fis_gRI35, fis_gRI36, fis_gRI37, fis_gRI38, fis_gRI39, fis_gRI40, fis_gRI41,
fis_gRI42, fis_gRI43, fis_gRI44, fis_gRI45, fis_gRI46, fis_gRI47, fis_gRI48,
fis_gRI49, fis_gRI50, fis_gRI51, fis_gRI52, fis_gRI53, fis_gRI54, fis_gRI55,
fis_gRI56, fis_gRI57, fis_gRI58, fis_gRI59, fis_gRI60, fis_gRI61, fis_gRI62,
fis_gRI63, fis_gRI64, fis_gRI65, fis_gRI66, fis_gRI67, fis_gRI68, fis_gRI69,
fis_gRI70, fis_gRI71, fis_gRI72, fis_gRI73, fis_gRI74, fis_gRI75, fis_gRI76,
fis_gRI77, fis_gRI78, fis_gRI79, fis_gRI80, fis_gRI81, fis_gRI82, fis_gRI83,
fis_gRI84 };

// Rule Outputs
int fis_gRO0[] = { 10 };
int fis_gRO1[] = { 10 };
int fis_gRO2[] = { 4 };
int fis_gRO3[] = { 5 };
int fis_gRO4[] = { 1 };
int fis_gRO5[] = { 10 };
int fis_gRO6[] = { 10 };
int fis_gRO7[] = { 4 };
int fis_gRO8[] = { 5 };
int fis_gRO9[] = { 1 };
int fis_gRO10[] = { 10 };
int fis_gRO11[] = { 10 };
int fis_gRO12[] = { 4 };
int fis_gRO13[] = { 5 };
int fis_gRO14[] = { 1 };
int fis_gRO15[] = { 10 };
int fis_gRO16[] = { 10 };
int fis_gRO17[] = { 4 };
int fis_gRO18[] = { 5 };
```

```
int fis_gRO19[] = { 1 };
int fis_gRO20[] = { 10 };
int fis_gRO21[] = { 10 };
int fis_gRO22[] = { 4 };
int fis_gRO23[] = { 5 };
int fis_gRO24[] = { 1 };
int fis_gRO25[] = { 4 };
int fis_gRO26[] = { 4 };
int fis_gRO27[] = { 5 };
int fis_gRO28[] = { 1 };
int fis_gRO29[] = { 6 };
int fis_gRO30[] = { 4 };
int fis_gRO31[] = { 5 };
int fis_gRO32[] = { 5 };
int fis_gRO33[] = { 1 };
int fis_gRO34[] = { 6 };
int fis_gRO35[] = { 4 };
int fis_gRO36[] = { 5 };
int fis_gRO37[] = { 5 };
int fis_gRO38[] = { 1 };
int fis_gRO39[] = { 6 };
int fis_gRO40[] = { 4 };
int fis_gRO41[] = { 5 };
int fis_gRO42[] = { 5 };
int fis_gRO43[] = { 1 };
int fis_gRO44[] = { 6 };
int fis_gRO45[] = { 4 };
int fis_gRO46[] = { 5 };
int fis_gRO47[] = { 5 };
int fis_gRO48[] = { 1 };
```

```
int fis_gRO49[] = { 6 };
int fis_gRO50[] = { 5 };
int fis_gRO51[] = { 1 };
int fis_gRO52[] = { 1 };
int fis_gRO53[] = { 6 };
int fis_gRO54[] = { 2 };
int fis_gRO55[] = { 5 };
int fis_gRO56[] = { 1 };
int fis_gRO57[] = { 1 };
int fis_gRO58[] = { 6 };
int fis_gRO59[] = { 2 };
int fis_gRO60[] = { 5 };
int fis_gRO61[] = { 1 };
int fis_gRO62[] = { 6 };
int fis_gRO63[] = { 2 };
int fis_gRO64[] = { 2 };
int fis_gRO65[] = { 1 };
int fis_gRO66[] = { 1 };
int fis_gRO67[] = { 6 };
int fis_gRO68[] = { 2 };
int fis_gRO69[] = { 7 };
int fis_gRO70[] = { 1 };
int fis_gRO71[] = { 1 };
int fis_gRO72[] = { 6 };
int fis_gRO73[] = { 2 };
int fis_gRO74[] = { 7 };
int fis_gRO75[] = { 1 };
int fis_gRO76[] = { 6 };
int fis_gRO77[] = { 2 };
int fis_gRO78[] = { 7 };
```

```
int fis_gRO79[] = { 8 };
int fis_gRO80[] = { 6 };
int fis_gRO81[] = { 2 };
int fis_gRO82[] = { 7 };
int fis_gRO83[] = { 8 };
int fis_gRO84[] = { 3 };
int* fis_gRO[] = { fis_gRO0, fis_gRO1, fis_gRO2, fis_gRO3, fis_gRO4, fis_gRO5,
fis_gRO6, fis_gRO7, fis_gRO8, fis_gRO9, fis_gRO10, fis_gRO11, fis_gRO12,
fis_gRO13, fis_gRO14, fis_gRO15, fis_gRO16, fis_gRO17, fis_gRO18, fis_gRO19,
fis_gRO20, fis_gRO21, fis_gRO22, fis_gRO23, fis_gRO24, fis_gRO25, fis_gRO26,
fis_gRO27, fis_gRO28, fis_gRO29, fis_gRO30, fis_gRO31, fis_gRO32, fis_gRO33,
fis_gRO34, fis_gRO35, fis_gRO36, fis_gRO37, fis_gRO38, fis_gRO39, fis_gRO40,
fis_gRO41, fis_gRO42, fis_gRO43, fis_gRO44, fis_gRO45, fis_gRO46, fis_gRO47,
fis_gRO48, fis_gRO49, fis_gRO50, fis_gRO51, fis_gRO52, fis_gRO53, fis_gRO54,
fis_gRO55, fis_gRO56, fis_gRO57, fis_gRO58, fis_gRO59, fis_gRO60, fis_gRO61,
fis_gRO62, fis_gRO63, fis_gRO64, fis_gRO65, fis_gRO66, fis_gRO67, fis_gRO68,
fis_gRO69, fis_gRO70, fis_gRO71, fis_gRO72, fis_gRO73, fis_gRO74, fis_gRO75,
fis_gRO76, fis_gRO77, fis_gRO78, fis_gRO79, fis_gRO80, fis_gRO81, fis_gRO82,
fis_gRO83, fis_gRO84 };

// Input range Min
FIS_TYPE fis_gIMin[] = { 25, 0 };

// Input range Max
FIS_TYPE fis_gIMax[] = { 95, 30 };

// Output range Min
FIS_TYPE fis_gOMin[] = { 0 };

// Output range Max
```

```
FIS_TYPE fis_gOMax[] = { 3 };

//*************************************************************************
****
// Data dependent support functions for Fuzzy Inference System
//*************************************************************************
****
FIS_TYPE fis_MF_out(FIS_TYPE** fuzzyRuleSet, FIS_TYPE x, int o)
{
   FIS_TYPE mfOut;
   int r;

   for (r = 0; r < fis_gcR; ++r)
   {
      int index = fis_gRO[r][o];
      if (index > 0)
      {
         index = index - 1;
         mfOut = (fis_gMF[fis_gMFO[o][index]])(x, fis_gMFOCoeff[o][index]);
      }
      else if (index < 0)
      {
         index = -index - 1;
         mfOut = 1 - (fis_gMF[fis_gMFO[o][index]])(x, fis_gMFOCoeff[o][index]);
      }
      else
      {
         mfOut = 0;
      }
```

```
        fuzzyRuleSet[0][r] = fis_min(mfOut, fuzzyRuleSet[1][r]);
    }
    return fis_array_operation(fuzzyRuleSet[0], fis_gcR, fis_max);
}

FIS_TYPE fis_defuzz_centroid(FIS_TYPE** fuzzyRuleSet, int o)
{
    FIS_TYPE step = (fis_gOMax[o] - fis_gOMin[o]) / (FIS_RESOLUSION - 1);
    FIS_TYPE area = 0;
    FIS_TYPE momentum = 0;
    FIS_TYPE dist, slice;
    int i;

    // calculate the area under the curve formed by the MF outputs
    for (i = 0; i < FIS_RESOLUSION; ++i){
        dist = fis_gOMin[o] + (step * i);
        slice = step * fis_MF_out(fuzzyRuleSet, dist, o);
        area += slice;
        momentum += slice*dist;
    }

    return ((area == 0) ? ((fis_gOMax[o] + fis_gOMin[o]) / 2) : (momentum / area));
}

//************************************************************************
****
// Fuzzy Inference System
//************************************************************************
****
void fis_evaluate()
```

```
{
    FIS_TYPE fuzzyInput0[] = { 0, 0, 0, 0, 0 };
    FIS_TYPE fuzzyInput1[] = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
    FIS_TYPE* fuzzyInput[fis_gcI] = { fuzzyInput0, fuzzyInput1, };
    FIS_TYPE fuzzyOutput0[] = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
    FIS_TYPE* fuzzyOutput[fis_gcO] = { fuzzyOutput0, };
    FIS_TYPE fuzzyRules[fis_gcR] = { 0 };
    FIS_TYPE fuzzyFires[fis_gcR] = { 0 };
    FIS_TYPE* fuzzyRuleSet[] = { fuzzyRules, fuzzyFires };
    FIS_TYPE sW = 0;

    // Transforming input to fuzzy Input
    int i, j, r, o;
    for (i = 0; i < fis_gcI; ++i)
    {
        for (j = 0; j < fis_gIMFCount[i]; ++j)
        {
            fuzzyInput[i][j] =
                (fis_gMF[fis_gMFI[i][j]])(g_fisInput[i], fis_gMFICoeff[i][j]);
        }
    }

    int index = 0;
    for (r = 0; r < fis_gcR; ++r)
    {
        if (fis_gRType[r] == 1)
        {
            fuzzyFires[r] = FIS_MAX;
            for (i = 0; i < fis_gcI; ++i)
            {
```

```
            index = fis_gRI[r][i];
            if (index > 0)
               fuzzyFires[r] = fis_min(fuzzyFires[r], fuzzyInput[i][index - 1]);
            else if (index < 0)
               fuzzyFires[r] = fis_min(fuzzyFires[r], 1 - fuzzyInput[i][-index - 1]);
            else
               fuzzyFires[r] = fis_min(fuzzyFires[r], 1);
         }
      }
      else
      {
         fuzzyFires[r] = FIS_MIN;
         for (i = 0; i < fis_gcI; ++i)
         {
            index = fis_gRI[r][i];
            if (index > 0)
               fuzzyFires[r] = fis_max(fuzzyFires[r], fuzzyInput[i][index - 1]);
            else if (index < 0)
               fuzzyFires[r] = fis_max(fuzzyFires[r], 1 - fuzzyInput[i][-index - 1]);
            else
               fuzzyFires[r] = fis_max(fuzzyFires[r], 0);
         }
      }

      fuzzyFires[r] = fis_gRWeight[r] * fuzzyFires[r];
      sW += fuzzyFires[r];
   }

   if (sW == 0)
   {
```

```
        for (o = 0; o < fis_gcO; ++o)
        {
            g_fisOutput[o] = ((fis_gOMax[o] + fis_gOMin[o]) / 2);
        }
    }
    else
    {
        for (o = 0; o < fis_gcO; ++o)
        {
            g_fisOutput[o] = fis_defuzz_centroid(fuzzyRuleSet, o);
        }
    }
}
```

ДОДАТОК Г

Відредагований лістинг програмного коду який використовується в системі

```
#include "fis_header.h"
#include <Wire.h>
#include <DS3231.h>

DS3231 clock;
RTCDateTime dt;
int Day;
int SDay;

// Number of inputs to the fuzzy inference system
const int fis_gcI = 2;
// Number of outputs to the fuzzy inference system
const int fis_gcO = 1;
// Number of rules to the fuzzy inference system
const int fis_gcR = 85;

FIS_TYPE g_fisInput[fis_gcI];
FIS_TYPE g_fisOutput[fis_gcO];

// Setup routine runs once when you press reset:
void setup()
{
  Serial.begin(9600);

  Serial.println("Initialize DS3231");;
  clock.begin();
  // Set sketch compiling time
```

```
// Встановлюємо час на годиннику, базуючись на часі компіляції скетча
  //clock.setDateTime(__DATE__, __TIME__);
// Встановлюмо час самостійно, відповідно: (Рік, Місяць, День, Година,
Хвилина, Секунда)
  //  clock.setDateTime(2021, 11, 8, 09, 36, 0);
     // initialize the Analog pins for input.
  // Pin mode for Input: Humidity
  pinMode(A0 , INPUT);
   // initialize the Analog pins for output.
  // Pin mode for Output: Time
  pinMode(11 , OUTPUT);
  pinMode(12 , OUTPUT);  // led indication

}

// Loop routine runs over and over again forever:
void loop()
{
    dt = clock.getDateTime();

  if (dt.day != SDay)
  {
    SDay = dt.day;
    Day ++;
  }

    // Read Input: Humidity
    int H = analogRead(A0);
    g_fisInput[0] = map(H, 0, 1023, 25, 95);
    // Read Input: Day
```

```
    g_fisInput[1] = Day;

    g_fisOutput[0] = 0;
    fis_evaluate();

    float pompout =((g_fisOutput[0]+0.195)*1000);
    digitalWrite(11, HIGH);
    digitalWrite(12, HIGH);     // led indication
    delay(pompout);
    digitalWrite(11, LOW);
    digitalWrite(12, LOW);

//*****************************************************************************
    Serial.print("Today: ");
    Serial.print(dt.year);   Serial.print("-");
    Serial.print(dt.month);  Serial.print("-");
    Serial.print(dt.day);    Serial.print(" ");
    Serial.print(dt.hour);   Serial.print(":");
    Serial.print(dt.minute); Serial.print(":");
    Serial.print(dt.second); Serial.println("");

//*****************************************************************************
Serial.print("Humidity:");
    Serial.print(g_fisInput[0]);
    Serial.print("    Day: ");
    Serial.print(g_fisInput[1]);

    Serial.print("   Run time (sec.) = ");
    Serial.print(g_fisOutput[0]);
```

```
    Serial.print("   Real run time (ms) = ");
    Serial.println(pompout);

    delay(1000);

 // Set output vlaue: Time
    digitalWrite(11 , g_fisOutput[0]);

}
//*************************************************************************
****
// Support functions for Fuzzy Inference System
//*************************************************************************
****
// Trapezoidal Member Function
FIS_TYPE fis_trapmf(FIS_TYPE x, FIS_TYPE* p)
{
    FIS_TYPE a = p[0], b = p[1], c = p[2], d = p[3];
    FIS_TYPE t1 = ((x <= c) ? 1 : ((d < x) ? 0 : ((c != d) ? ((d - x) / (d - c)) : 0)));
    FIS_TYPE t2 = ((b <= x) ? 1 : ((x < a) ? 0 : ((a != b) ? ((x - a) / (b - a)) : 0)));
    return (FIS_TYPE) min(t1, t2);
}

// Triangular Member Function
FIS_TYPE fis_trimf(FIS_TYPE x, FIS_TYPE* p)
{
    FIS_TYPE a = p[0], b = p[1], c = p[2];
    FIS_TYPE t1 = (x - a) / (b - a);
    FIS_TYPE t2 = (c - x) / (c - b);
    if ((a == b) && (b == c)) return (FIS_TYPE) (x == a);
```

```
    if (a == b) return (FIS_TYPE) (t2*(b <= x)*(x <= c));
    if (b == c) return (FIS_TYPE) (t1*(a <= x)*(x <= b));
    t1 = min(t1, t2);
    return (FIS_TYPE) max(t1, 0);
}

FIS_TYPE fis_min(FIS_TYPE a, FIS_TYPE b)
{
    return min(a, b);
}

FIS_TYPE fis_max(FIS_TYPE a, FIS_TYPE b)
{
    return max(a, b);
}

FIS_TYPE fis_array_operation(FIS_TYPE *array, int size, _FIS_ARR_OP pfnOp)
{
    int i;
    FIS_TYPE ret = 0;

    if (size == 0) return ret;
    if (size == 1) return array[0];

    ret = array[0];
    for (i = 1; i < size; i++)
    {
        ret = (*pfnOp)(ret, array[i]);
    }
```

```
    return ret;
}


//*************************************************************************
****
// Data for Fuzzy Inference System
//*************************************************************************
****
// Pointers to the implementations of member functions
_FIS_MF fis_gMF[] =
{
    fis_trapmf, fis_trimf
};

// Count of member function for each Input
int fis_gIMFCount[] = { 5, 17 };

// Count of member function for each Output
int fis_gOMFCount[] = { 10 };

// Coefficients for the Input Member Functions
FIS_TYPE fis_gMFI0Coeff1[] = { 15.95, 22.01, 32.99, 39.05 };
FIS_TYPE fis_gMFI0Coeff2[] = { 60.87, 75, 89.13 };
FIS_TYPE fis_gMFI0Coeff3[] = { 45.87, 60, 74.13 };
FIS_TYPE fis_gMFI0Coeff4[] = { 30.87, 45, 59.13 };
FIS_TYPE fis_gMFI0Coeff5[] = { 82.5, 89.92, 107.7, 115.1 };
FIS_TYPE* fis_gMFI0Coeff[] = { fis_gMFI0Coeff1, fis_gMFI0Coeff2,
fis_gMFI0Coeff3, fis_gMFI0Coeff4, fis_gMFI0Coeff5 };
FIS_TYPE fis_gMFI1Coeff1[] = { -10.8, -1.2, 0.9, 1 };
```

```
FIS_TYPE fis_gMFI1Coeff2[] = { 0.9, 1.1, 1.9, 2 };
FIS_TYPE fis_gMFI1Coeff3[] = { 1.9, 2.1, 2.9, 3 };
FIS_TYPE fis_gMFI1Coeff4[] = { 2.9, 3.1, 3.9, 4 };
FIS_TYPE fis_gMFI1Coeff5[] = { 3.9, 4.1, 4.9, 5 };
FIS_TYPE fis_gMFI1Coeff6[] = { 4.9, 5.1, 5.9, 6 };
FIS_TYPE fis_gMFI1Coeff7[] = { 5.9, 6.1, 6.9, 7 };
FIS_TYPE fis_gMFI1Coeff8[] = { 6.9, 7.1, 7.9, 8 };
FIS_TYPE fis_gMFI1Coeff9[] = { 7.9, 8.1, 8.9, 9 };
FIS_TYPE fis_gMFI1Coeff10[] = { 8.9, 9.1, 9.9, 10 };
FIS_TYPE fis_gMFI1Coeff11[] = { 9.9, 10.1, 11.9, 12 };
FIS_TYPE fis_gMFI1Coeff12[] = { 11.9, 12.1, 13.9, 14 };
FIS_TYPE fis_gMFI1Coeff13[] = { 13.9, 14.1, 15.9, 16 };
FIS_TYPE fis_gMFI1Coeff14[] = { 15.9, 16.1, 17.9, 18 };
FIS_TYPE fis_gMFI1Coeff15[] = { 17.9, 18.1, 19.9, 20 };
FIS_TYPE fis_gMFI1Coeff16[] = { 19.9, 20.1, 24.9, 25 };
FIS_TYPE fis_gMFI1Coeff17[] = { 24.9, 25.1, 32, 32 };
FIS_TYPE* fis_gMFI1Coeff[] = { fis_gMFI1Coeff1, fis_gMFI1Coeff2,
fis_gMFI1Coeff3, fis_gMFI1Coeff4, fis_gMFI1Coeff5, fis_gMFI1Coeff6,
fis_gMFI1Coeff7, fis_gMFI1Coeff8, fis_gMFI1Coeff9, fis_gMFI1Coeff10,
fis_gMFI1Coeff11, fis_gMFI1Coeff12, fis_gMFI1Coeff13, fis_gMFI1Coeff14,
fis_gMFI1Coeff15, fis_gMFI1Coeff16, fis_gMFI1Coeff17 };
FIS_TYPE** fis_gMFICoeff[] = { fis_gMFI0Coeff, fis_gMFI1Coeff };

// Coefficients for the Output Member Functions
FIS_TYPE fis_gMFO0Coeff1[] = { 0.734, 0.91, 1.03571428571429 };
FIS_TYPE fis_gMFO0Coeff2[] = { 1.08, 1.17, 1.31349206349206 };
FIS_TYPE fis_gMFO0Coeff3[] = { 1.6, 1.79, 2.37698412698413 };
FIS_TYPE fis_gMFO0Coeff4[] = { 0.194, 0.2, 0.483, 0.6 };
FIS_TYPE fis_gMFO0Coeff5[] = { 0.48, 0.69, 0.892857142857143 };
FIS_TYPE fis_gMFO0Coeff6[] = { 0.925, 1.05, 1.19444444444444 };
```

```
FIS_TYPE fis_gMFO0Coeff7[] = { 1.2, 1.38, 1.55952380952381 };
FIS_TYPE fis_gMFO0Coeff8[] = { 1.37, 1.59, 1.79761904761905 };
FIS_TYPE fis_gMFO0Coeff9[] = { 1.81349206349206, 2.55, 3, 3.5 };
FIS_TYPE fis_gMFO0Coeff10[] = { 0, 0, 0, 0 };
FIS_TYPE* fis_gMFO0Coeff[] = { fis_gMFO0Coeff1, fis_gMFO0Coeff2,
fis_gMFO0Coeff3, fis_gMFO0Coeff4, fis_gMFO0Coeff5, fis_gMFO0Coeff6,
fis_gMFO0Coeff7, fis_gMFO0Coeff8, fis_gMFO0Coeff9, fis_gMFO0Coeff10 };
FIS_TYPE** fis_gMFOCoeff[] = { fis_gMFO0Coeff };

// Input membership function set
int fis_gMFI0[] = { 0, 1, 1, 1, 0 };
int fis_gMFI1[] = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
int* fis_gMFI[] = { fis_gMFI0, fis_gMFI1};

// Output membership function set
int fis_gMFO0[] = { 1, 1, 1, 0, 1, 1, 1, 1, 0, 0 };
int* fis_gMFO[] = { fis_gMFO0};

// Rule Weights
FIS_TYPE fis_gRWeight[] = { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 };

// Rule Type
int fis_gRType[] = { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 };

// Rule Inputs
int fis_gRI0[] = { 5, 1 };
```

```
int fis_gRI1[] = { 2, 1 };
int fis_gRI2[] = { 3, 1 };
int fis_gRI3[] = { 4, 1 };
int fis_gRI4[] = { 1, 1 };
int fis_gRI5[] = { 5, 2 };
int fis_gRI6[] = { 2, 2 };
int fis_gRI7[] = { 3, 2 };
int fis_gRI8[] = { 4, 2 };
int fis_gRI9[] = { 1, 2 };
int fis_gRI10[] = { 5, 3 };
int fis_gRI11[] = { 2, 3 };
int fis_gRI12[] = { 3, 3 };
int fis_gRI13[] = { 4, 3 };
int fis_gRI14[] = { 1, 3 };
int fis_gRI15[] = { 5, 4 };
int fis_gRI16[] = { 2, 4 };
int fis_gRI17[] = { 3, 4 };
int fis_gRI18[] = { 4, 4 };
int fis_gRI19[] = { 1, 4 };
int fis_gRI20[] = { 5, 5 };
int fis_gRI21[] = { 2, 5 };
int fis_gRI22[] = { 3, 5 };
int fis_gRI23[] = { 4, 5 };
int fis_gRI24[] = { 1, 5 };
int fis_gRI25[] = { 5, 6 };
int fis_gRI26[] = { 2, 6 };
int fis_gRI27[] = { 3, 6 };
int fis_gRI28[] = { 4, 6 };
int fis_gRI29[] = { 1, 6 };
int fis_gRI30[] = { 5, 7 };
```

```
int fis_gRI31[] = { 2, 7 };
int fis_gRI32[] = { 3, 7 };
int fis_gRI33[] = { 4, 7 };
int fis_gRI34[] = { 1, 7 };
int fis_gRI35[] = { 5, 8 };
int fis_gRI36[] = { 2, 8 };
int fis_gRI37[] = { 3, 8 };
int fis_gRI38[] = { 4, 8 };
int fis_gRI39[] = { 1, 8 };
int fis_gRI40[] = { 5, 9 };
int fis_gRI41[] = { 2, 9 };
int fis_gRI42[] = { 3, 9 };
int fis_gRI43[] = { 4, 9 };
int fis_gRI44[] = { 1, 9 };
int fis_gRI45[] = { 5, 10 };
int fis_gRI46[] = { 2, 10 };
int fis_gRI47[] = { 3, 10 };
int fis_gRI48[] = { 4, 10 };
int fis_gRI49[] = { 1, 10 };
int fis_gRI50[] = { 5, 11 };
int fis_gRI51[] = { 2, 11 };
int fis_gRI52[] = { 3, 11 };
int fis_gRI53[] = { 4, 11 };
int fis_gRI54[] = { 1, 11 };
int fis_gRI55[] = { 5, 12 };
int fis_gRI56[] = { 2, 12 };
int fis_gRI57[] = { 3, 12 };
int fis_gRI58[] = { 4, 12 };
int fis_gRI59[] = { 1, 12 };
int fis_gRI60[] = { 5, 13 };
```

```
int fis_gRI61[] = { 2, 13 };
int fis_gRI62[] = { 3, 13 };
int fis_gRI63[] = { 4, 13 };
int fis_gRI64[] = { 1, 13 };
int fis_gRI65[] = { 5, 14 };
int fis_gRI66[] = { 2, 14 };
int fis_gRI67[] = { 3, 14 };
int fis_gRI68[] = { 4, 14 };
int fis_gRI69[] = { 1, 14 };
int fis_gRI70[] = { 5, 15 };
int fis_gRI71[] = { 2, 15 };
int fis_gRI72[] = { 3, 15 };
int fis_gRI73[] = { 4, 15 };
int fis_gRI74[] = { 1, 15 };
int fis_gRI75[] = { 5, 16 };
int fis_gRI76[] = { 2, 16 };
int fis_gRI77[] = { 3, 16 };
int fis_gRI78[] = { 4, 16 };
int fis_gRI79[] = { 1, 16 };
int fis_gRI80[] = { 5, 17 };
int fis_gRI81[] = { 2, 17 };
int fis_gRI82[] = { 3, 17 };
int fis_gRI83[] = { 4, 17 };
int fis_gRI84[] = { 1, 17 };
int* fis_gRI[] = { fis_gRI0, fis_gRI1, fis_gRI2, fis_gRI3, fis_gRI4, fis_gRI5,
fis_gRI6, fis_gRI7, fis_gRI8, fis_gRI9, fis_gRI10, fis_gRI11, fis_gRI12, fis_gRI13,
fis_gRI14, fis_gRI15, fis_gRI16, fis_gRI17, fis_gRI18, fis_gRI19, fis_gRI20,
fis_gRI21, fis_gRI22, fis_gRI23, fis_gRI24, fis_gRI25, fis_gRI26, fis_gRI27,
fis_gRI28, fis_gRI29, fis_gRI30, fis_gRI31, fis_gRI32, fis_gRI33, fis_gRI34,
fis_gRI35, fis_gRI36, fis_gRI37, fis_gRI38, fis_gRI39, fis_gRI40, fis_gRI41,
```

```
fis_gRI42, fis_gRI43, fis_gRI44, fis_gRI45, fis_gRI46, fis_gRI47, fis_gRI48,
fis_gRI49, fis_gRI50, fis_gRI51, fis_gRI52, fis_gRI53, fis_gRI54, fis_gRI55,
fis_gRI56, fis_gRI57, fis_gRI58, fis_gRI59, fis_gRI60, fis_gRI61, fis_gRI62,
fis_gRI63, fis_gRI64, fis_gRI65, fis_gRI66, fis_gRI67, fis_gRI68, fis_gRI69,
fis_gRI70, fis_gRI71, fis_gRI72, fis_gRI73, fis_gRI74, fis_gRI75, fis_gRI76,
fis_gRI77, fis_gRI78, fis_gRI79, fis_gRI80, fis_gRI81, fis_gRI82, fis_gRI83,
fis_gRI84 };

// Rule Outputs
int fis_gRO0[] = { 10 };
int fis_gRO1[] = { 10 };
int fis_gRO2[] = { 4 };
int fis_gRO3[] = { 5 };
int fis_gRO4[] = { 1 };
int fis_gRO5[] = { 10 };
int fis_gRO6[] = { 10 };
int fis_gRO7[] = { 4 };
int fis_gRO8[] = { 5 };
int fis_gRO9[] = { 1 };
int fis_gRO10[] = { 10 };
int fis_gRO11[] = { 10 };
int fis_gRO12[] = { 4 };
int fis_gRO13[] = { 5 };
int fis_gRO14[] = { 1 };
int fis_gRO15[] = { 10 };
int fis_gRO16[] = { 10 };
int fis_gRO17[] = { 4 };
int fis_gRO18[] = { 5 };
int fis_gRO19[] = { 1 };
int fis_gRO20[] = { 10 };
```

```
int fis_gRO21[] = { 10 };
int fis_gRO22[] = { 4 };
int fis_gRO23[] = { 5 };
int fis_gRO24[] = { 1 };
int fis_gRO25[] = { 4 };
int fis_gRO26[] = { 4 };
int fis_gRO27[] = { 5 };
int fis_gRO28[] = { 1 };
int fis_gRO29[] = { 6 };
int fis_gRO30[] = { 4 };
int fis_gRO31[] = { 5 };
int fis_gRO32[] = { 5 };
int fis_gRO33[] = { 1 };
int fis_gRO34[] = { 6 };
int fis_gRO35[] = { 4 };
int fis_gRO36[] = { 5 };
int fis_gRO37[] = { 5 };
int fis_gRO38[] = { 1 };
int fis_gRO39[] = { 6 };
int fis_gRO40[] = { 4 };
int fis_gRO41[] = { 5 };
int fis_gRO42[] = { 5 };
int fis_gRO43[] = { 1 };
int fis_gRO44[] = { 6 };
int fis_gRO45[] = { 4 };
int fis_gRO46[] = { 5 };
int fis_gRO47[] = { 5 };
int fis_gRO48[] = { 1 };
int fis_gRO49[] = { 6 };
int fis_gRO50[] = { 5 };
```

```
int fis_gRO51[] = { 1 };
int fis_gRO52[] = { 1 };
int fis_gRO53[] = { 6 };
int fis_gRO54[] = { 2 };
int fis_gRO55[] = { 5 };
int fis_gRO56[] = { 1 };
int fis_gRO57[] = { 1 };
int fis_gRO58[] = { 6 };
int fis_gRO59[] = { 2 };
int fis_gRO60[] = { 5 };
int fis_gRO61[] = { 1 };
int fis_gRO62[] = { 6 };
int fis_gRO63[] = { 2 };
int fis_gRO64[] = { 2 };
int fis_gRO65[] = { 1 };
int fis_gRO66[] = { 1 };
int fis_gRO67[] = { 6 };
int fis_gRO68[] = { 2 };
int fis_gRO69[] = { 7 };
int fis_gRO70[] = { 1 };
int fis_gRO71[] = { 1 };
int fis_gRO72[] = { 6 };
int fis_gRO73[] = { 2 };
int fis_gRO74[] = { 7 };
int fis_gRO75[] = { 1 };
int fis_gRO76[] = { 6 };
int fis_gRO77[] = { 2 };
int fis_gRO78[] = { 7 };
int fis_gRO79[] = { 8 };
int fis_gRO80[] = { 6 };
```

```
int fis_gRO81[] = { 2 };
int fis_gRO82[] = { 7 };
int fis_gRO83[] = { 8 };
int fis_gRO84[] = { 3 };
int* fis_gRO[] = { fis_gRO0, fis_gRO1, fis_gRO2, fis_gRO3, fis_gRO4, fis_gRO5,
fis_gRO6, fis_gRO7, fis_gRO8, fis_gRO9, fis_gRO10, fis_gRO11, fis_gRO12,
fis_gRO13, fis_gRO14, fis_gRO15, fis_gRO16, fis_gRO17, fis_gRO18, fis_gRO19,
fis_gRO20, fis_gRO21, fis_gRO22, fis_gRO23, fis_gRO24, fis_gRO25, fis_gRO26,
fis_gRO27, fis_gRO28, fis_gRO29, fis_gRO30, fis_gRO31, fis_gRO32, fis_gRO33,
fis_gRO34, fis_gRO35, fis_gRO36, fis_gRO37, fis_gRO38, fis_gRO39, fis_gRO40,
fis_gRO41, fis_gRO42, fis_gRO43, fis_gRO44, fis_gRO45, fis_gRO46, fis_gRO47,
fis_gRO48, fis_gRO49, fis_gRO50, fis_gRO51, fis_gRO52, fis_gRO53, fis_gRO54,
fis_gRO55, fis_gRO56, fis_gRO57, fis_gRO58, fis_gRO59, fis_gRO60, fis_gRO61,
fis_gRO62, fis_gRO63, fis_gRO64, fis_gRO65, fis_gRO66, fis_gRO67, fis_gRO68,
fis_gRO69, fis_gRO70, fis_gRO71, fis_gRO72, fis_gRO73, fis_gRO74, fis_gRO75,
fis_gRO76, fis_gRO77, fis_gRO78, fis_gRO79, fis_gRO80, fis_gRO81, fis_gRO82,
fis_gRO83, fis_gRO84 };

// Input range Min
FIS_TYPE fis_gIMin[] = { 25, 0 };

// Input range Max
FIS_TYPE fis_gIMax[] = { 95, 30 };

// Output range Min
FIS_TYPE fis_gOMin[] = { 0 };

// Output range Max
FIS_TYPE fis_gOMax[] = { 3 };
```

```
//*********************************************************************
****
// Data dependent support functions for Fuzzy Inference System
//*********************************************************************
****
FIS_TYPE fis_MF_out(FIS_TYPE** fuzzyRuleSet, FIS_TYPE x, int o)
{
   FIS_TYPE mfOut;
   int r;

   for (r = 0; r < fis_gcR; ++r)
   {
      int index = fis_gRO[r][o];
      if (index > 0)
      {
         index = index - 1;
         mfOut = (fis_gMF[fis_gMFO[o][index]])(x, fis_gMFOCoeff[o][index]);
      }
      else if (index < 0)
      {
         index = -index - 1;
         mfOut = 1 - (fis_gMF[fis_gMFO[o][index]])(x, fis_gMFOCoeff[o][index]);
      }
      else
      {
         mfOut = 0;
      }

      fuzzyRuleSet[0][r] = fis_min(mfOut, fuzzyRuleSet[1][r]);
   }
```

```
    return fis_array_operation(fuzzyRuleSet[0], fis_gcR, fis_max);
}

FIS_TYPE fis_defuzz_centroid(FIS_TYPE** fuzzyRuleSet, int o)
{
    FIS_TYPE step = (fis_gOMax[o] - fis_gOMin[o]) / (FIS_RESOLUSION - 1);
    FIS_TYPE area = 0;
    FIS_TYPE momentum = 0;
    FIS_TYPE dist, slice;
    int i;

    // calculate the area under the curve formed by the MF outputs
    for (i = 0; i < FIS_RESOLUSION; ++i){
        dist = fis_gOMin[o] + (step * i);
        slice = step * fis_MF_out(fuzzyRuleSet, dist, o);
        area += slice;
        momentum += slice*dist;
    }

    return ((area == 0) ? ((fis_gOMax[o] + fis_gOMin[o]) / 2) : (momentum / area));
}

//************************************************************************
****
// Fuzzy Inference System
//************************************************************************
****
void fis_evaluate()
{
    FIS_TYPE fuzzyInput0[] = { 0, 0, 0, 0, 0 };
```

```
FIS_TYPE fuzzyInput1[] = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
FIS_TYPE* fuzzyInput[fis_gcI] = { fuzzyInput0, fuzzyInput1, };
FIS_TYPE fuzzyOutput0[] = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
FIS_TYPE* fuzzyOutput[fis_gcO] = { fuzzyOutput0, };
FIS_TYPE fuzzyRules[fis_gcR] = { 0 };
FIS_TYPE fuzzyFires[fis_gcR] = { 0 };
FIS_TYPE* fuzzyRuleSet[] = { fuzzyRules, fuzzyFires };
FIS_TYPE sW = 0;

// Transforming input to fuzzy Input
int i, j, r, o;
for (i = 0; i < fis_gcI; ++i)
{
   for (j = 0; j < fis_gIMFCount[i]; ++j)
   {
      fuzzyInput[i][j] =
         (fis_gMF[fis_gMFI[i][j]])(g_fisInput[i], fis_gMFICoeff[i][j]);
   }
}

int index = 0;
for (r = 0; r < fis_gcR; ++r)
{
   if (fis_gRType[r] == 1)
   {
      fuzzyFires[r] = FIS_MAX;
      for (i = 0; i < fis_gcI; ++i)
      {
         index = fis_gRI[r][i];
         if (index > 0)
```

```
            fuzzyFires[r] = fis_min(fuzzyFires[r], fuzzyInput[i][index - 1]);
        else if (index < 0)
            fuzzyFires[r] = fis_min(fuzzyFires[r], 1 - fuzzyInput[i][-index - 1]);
        else
            fuzzyFires[r] = fis_min(fuzzyFires[r], 1);
      }
   }
   else
   {
      fuzzyFires[r] = FIS_MIN;
      for (i = 0; i < fis_gcI; ++i)
      {
         index = fis_gRI[r][i];
         if (index > 0)
            fuzzyFires[r] = fis_max(fuzzyFires[r], fuzzyInput[i][index - 1]);
         else if (index < 0)
            fuzzyFires[r] = fis_max(fuzzyFires[r], 1 - fuzzyInput[i][-index - 1]);
         else
            fuzzyFires[r] = fis_max(fuzzyFires[r], 0);
      }
   }

   fuzzyFires[r] = fis_gRWeight[r] * fuzzyFires[r];
   sW += fuzzyFires[r];
}

if (sW == 0)
{
   for (o = 0; o < fis_gcO; ++o)
   {
```

```
            g_fisOutput[o] = ((fis_gOMax[o] + fis_gOMin[o]) / 2);
        }
    }
    else
    {
        for (o = 0; o < fis_gcO; ++o)
        {
            g_fisOutput[o] = fis_defuzz_centroid(fuzzyRuleSet, o);
        }
    }
}
```