РОЗРОБКА ТА ДОСЛІДЖЕННЯ ЕЛЕКТРОННОЇ СИСТЕМИ МОНІТОРИНГУ Й КЕРУВАННЯ ІНТЕНСИВНІСТЮ ОСВІТЛЕННЯ ТЕПЛИЦЬ

ШИФР – МОНІТОРИНГ ОСВІТЛЕННЯ

ЗМІСТ

АНОТАЦІЯ

Сільськогосподарські землі України в основному призначені для вирощування культур відкритого ґрунту. Оскільки якість врожаю нестабільна через різні кліматичних умов, то одним із варіантів підвищення врожайності протягом усього календарного року є вирощування культур у теплицях.

Теплиця дозволяє підтримувати сприятливі кліматичні умови для різних типів культур. Захищені культури менш схильні до пошкодження вітром, дощем, снігом, тому відсоток товарної продукції вище. Врожайність часто буває високою, якщо вдається забезпечити оптимальні умови для вирощування кожної культури.

Теплиці захищають посіви від багатьох хвороб, особливо тих, які вихлюпуються на рослини під час дощу. Захищають від звичайних польових шкідників. Так само є і особливі проблеми, такі як некоренева хвороба, тля.

Для підвищення врожайності культур є необхідність у введенні автоматизованої системи управління. Це дозволяє регулювати параметри сприятливих умов. Управління системи вентиляції, яка забезпечує постійний приплив вуглекислого газу, який необхідний для рослин. Регулювання вологості повітря.

Культури під відкритим ґрунтом покладаються виключно на вологість ґрунту, що може викликати водний стрес, особливо якщо починається період посухи. Зниження водного стресу шляхом поливання ґрунту і зволоження повітря дозволяє підвищити плодоношення.

Автоматичне управління допомагає підтримувати штучне навколишнє середовище відповідно до потреб різних культур без необхідності постійного моніторингу і регулювання. Однією з найбільших переваг, які може забезпечити дана система, є можливість зниження загальних виробничих витрат. Використання автоматизованого освітлення в теплиці допомагає задовольнити потреби рослин. Коригування спектра і інтенсивності дозволяє:

- поліпшити продуктивність і здатність підтримувати норми освітлення не залежно від пори року;
- зниження споживання електроенергії;
- збільшення врожаю;
- налаштування кольору, управління спектром.

Мета роботи: розробити та реалізувати модель управління штучним освітленням в теплиці на базі Matlab & Simulink for fuzzy logic Toolbox

Завдання досліджень: в процесі вирішення даного завдання передбачається створити модель, яка буде реалізована на базі нечіткої логіки для керування спектральним освітленням.

Загальна методика досліджень: теоретико-експериментальні дослідження розробленої електронної мікропроцесорної системи моніторингу й керування інтенсивність освітлення зони вирощування тепличних культур у видимому діапазоні світла отримані на підставі методів розрахунку структурних блоків електронних систем; методів неруйнівного інструментального вимірювального контролю; методик математичного планування експерименту; методів імітаційного моделювання технічних систем.

Структура та обсяг роботи. Робота складається зі вступу, двох розділів, обсягом 21 сторінки та 1 додатку на 9 сторінках, загальних висновків та переліку посилань. Робота містить 21 рисунок та 1 таблицю.

Ключові слова: *інтенсивність освітлення, теплиця, електронна система, нечітка логіка, модель.*

1 АНАЛІЗ ІСНУЮЧИХ РОЗРОБОК

Інтенсивність виробництва тепличних культур з кожним днем зростає і це призводить до необхідності удосконалення системи управління технологічними процесами.

В даний момент проводиться велика кількість розробок на тему досвітки рослин. Оскільки в зимовий період денна норма світла скорочується, рослина не отримує необхідну кількість світла. Будь-яка рослина потребує 12-16 годинну норму світлового дня. Існують безліч ламп для досвітки, але не всі актуальні на сьогоднішній день.

1.1 Опис існуючих метод для досвічування

У статті [3] автор описує види і вплив кожної лампи на рослину.

Наприклад візьмемо освітлення на звичайній лампі розжарювання. Висвітлюють досить непогано, але, спостерігалися опіки у рослин і деформування листа, при цьому ще підігрівають повітря. Не є енергоємними.

На сьогоднішній день дуже популярно використовувати для досвітки в теплиці світлодіодні лампи (фітолампи), як показано на рис. 1.1. Дані лампи дають багато світла при низькому споживання електроенергії.

Рисунок 1.1 – Фітолампи для рослин QvvCev

Так само часто в теплицях використовують натрієві лампи високого тиску, як показано на рис. 1.2. Один із найбільших плюсів даних ламп – це монохроматичність випромінювання в помаранчево-жовтому спектрі. Дані лампи не є довговічними.

Рисунок 1.2 – Приклад використання натрієвих ламп високого тиску

У статті [4] автор розповідає, як встановити штучне освітлення в теплиці, так само підбирає який вид ламп краще. Люмінесцентні лампи, як показано на рис. 1.3, є відмінним штучним замінником сонця. Вони довговічні, недорогі. Однак їх яскравість безпосередньо залежить від напруги в мережі. Якщо напруга буде низька, то на лампи подаватися не буде.

Рисунок 1.3 – Установка люмінесцентних ламп в теплиці

На даний момент існує безліч приладів для вимірювання рівня освітленості в приміщеннях, один із них – люксметр, як показано на рис. 1.4. У даній статті автор описує спосіб вимірювання освітленості [8]. Для забезпечення правильного рівня освітлення в теплиці, необхідно вимірювати рівень освітленості, адже недостатнє освітлення знижує рівень плодоношення. Тому контроль рівня освітленості є одним з необхідних параметрів під час створення оптимального мікроклімату в теплицях.

Рисунок 1.4 – Люксметр ТМ – 208

У статті [9], автори вивчили питання впливу оптичного випромінювання на розвиток і продуктивність рослин. Довели, що для нормального росту та розвитку рослин необхідним є світло певного спектрального складу та достатньої інтенсивності впродовж визначеного часу.

Для рослин життєво-необхідними є червоні й помаранчеві довжини хвиль. Саме вони є головним джерелом енергії під час фотосинтезу й впливають на розвиток рослини. Графік поглинання різних світлових спектрів рослинами представлено на рис. 1.5.

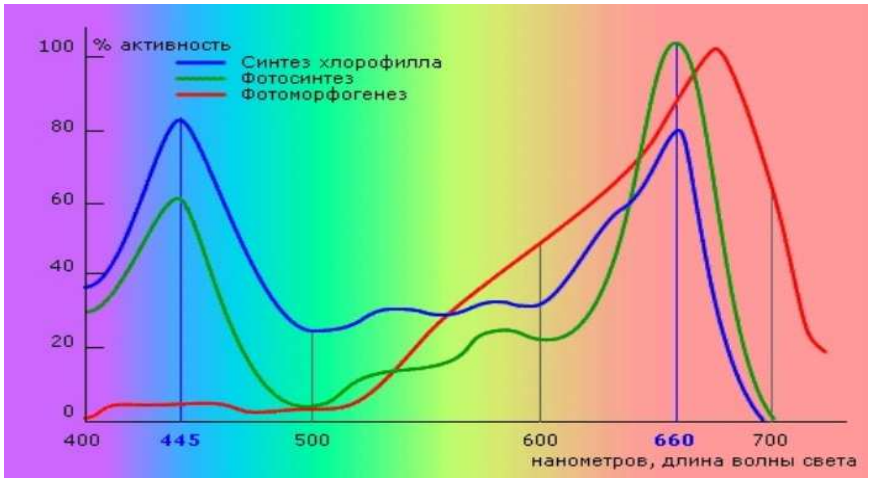


Рисунок 1.5 – Поглинання різних світлових спектрів

1.2 Розробки на базі Matlab & Simulink for fuzzy logic Toolbox

Актуальним питанням на сьогоднішній день стає інтелектуальне управління, яке здійснюється під час різних змін технологічних процесів. Застосування даної технології забезпечує підвищення рівня автоматизації та контролю різними процесами.

Виробництво тепличних культур пов'язано з необхідністю підтримання мікроклімату технологічного приміщення: температура повітря, вологість повітря, спектральний освітлення, вентиляція приміщення.

У роботі автора [5] було досліджено динамічне регулювання освітленням. Автор проводив дослідження залежності росту декоративного соняшнику під динамічним освітленням. Даний метод знижує енергоспоживання, не впливаючи на якість соняшнику.

На базі Matlab & Simulink for fuzzy logic Toolbox існує розробка системи управління поливом [1]. Автор у даній статті створив модель управління нечіткою логікою. Він використовував в якості вхідних змінних значення, які зчитуються з сенсорів у режимі реального часу. На виході встановлено двохпараметричний регулятор (світло, водна помпа) для підтримання нормованої вологості ґрунту. Результат роботи [1] був змодельований і дав позитивний результат.

У роботі автора [2] в якості об'єкта дослідження розглядалася система автоматичного управління параметрами мікроклімату. Параметри, які впливають на модель: вологість повітря, температура повітря, концентрація вуглекислого газу в повітрі. Була описана база правил, яка враховує значення параметрів для управління автоматичності режимом і відстеженням всіх змін в приміщення. Розроблена модель управління технологічним процесом показала принципові рішення і математичні моделі цієї роботи.

2 РЕЗУЛЬТАТИ ОСОБИСТИХ ДОСЛІДЖЕНЬ

Нечітка логіка (Fuzzy logic) – одна з технологій, яка застосовується в програмуванні для моделювання мислення і поведінки людини.

Дана технологія спрощує контроль регульованих процесів без конкретного людського втручання.

2.1 Характеристики матеріалів використаних в проекті.

Цифровий датчик освітленості BH-1750. Цифровий датчик освітленості, який наведено на рис. 2.1. має широкий робочий діапазон вимірювань від 1 до 65535 лк і спектр чутливості, який наближений до спектру людського ока.

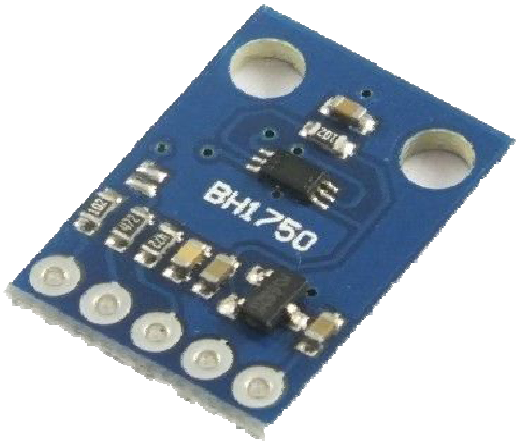


Рисунок 2.1 – Цифровий датчик освітленості

Основні характеристики датчика:

- тип: GY-30;
- чіп: BH1750FVI;
- вбудований сенсор і цифровий перетворювач;
- інтерфейс –I^2C;
- нечутливий до фонового кольору;
- спектральна характеристика близька до візуальної чутливості;

- широкий діапазон вимірювань;
- напруга живлення – 3..4,5 В.

Даний цифровий датчик побудовано на принципі управління параметром рівня освітлення в теплиці. Залежно від рівня освітлення генеруються сигнали управління режимами штучної досвітки рослин для забезпечення достатньої кількості світла.

Цифровий датчик температури DS18B20.

Даний датчик (див. рис. 2.2) дозволяє визначити температуру в діапазоні від -55 ºС до +125 ºС і отримувати дані у вигляді цифрового сигналу з 12-бітною роздільною здатністю через 1-Wire протокол. Даний протокол дозволяє підключити значну кількість таких же типів датчиків під час використання одного цифрового порту контролера.

Рисунок 2.2 – Цифровий датчик температури DS18B20

Герметичний датчик підключається по трьох проводах: живлення, загальний, цифровий сигнал. Сигнальний провід необхідно з'єднати з живленням через резистор 4,7 кОм.

Характеристики:

- діапазон вимірюваних температур: від -55 ºС до +125 ºС;
- абсолютна похибка вимірювання: ±0,5 ºС;
- час отримання даних: 750 мс;

- напруга живлення: від 3 до 5,5 В.

Даний герметичний датчик регулює вхідну змінну, що відповідає за поточне значення температури для визначення сприятливої температури зони вирощування теплиці й своєчасного включення системи освітлення. Макетна реалізація проекту здійснена на бюджетній платформі Arduino Uno, як показано на рис. 2.3.

Рисунок 2.3 – Arduino Uno

Мікроконтролер має: 14 цифрових входів/виходів (з них 6 можуть використовуватися в якості ШІМ-виходу), 6 аналогових входів, кварцовий резонатор, роз'єм USB, роз'єм живлення, роз'єм для програмування та кнопку скидання. Обсяг флеш-пам'яті становить 32 Кб.

Характеристики:

- мікроконтролер: ATmega328;
- робоча напруга: 5 В;
- напруга живлення (рекомендована): 7 – 12 В;
- напруга живлення (гранична): 6 – 20 В;
- цифрові входи/виходи: 14;
- аналогові виходи: 6;
- максимальний струм із одного виходу: 40 мА;
- максимальний вихідний струм виводу 3,3 В: 50 мА;

- Flash-пам'ять: 32 Кб із яких 0,5 Кб використовуються завантажчиком;
- SRAM: 2 Кб;
- EEPROM: 1 Кб;
- тактова частота: 16 МГц.

2.2 Опис роботи реалізованої в Matlab & Simulink® моделі

У даній роботі управління нечіткою логікою було здійснено на базі Matlab & Simulink®. Управління логікою, яке наведено на рис. 2.4, було реалізовано на алгоритмі роботи Mamdani.

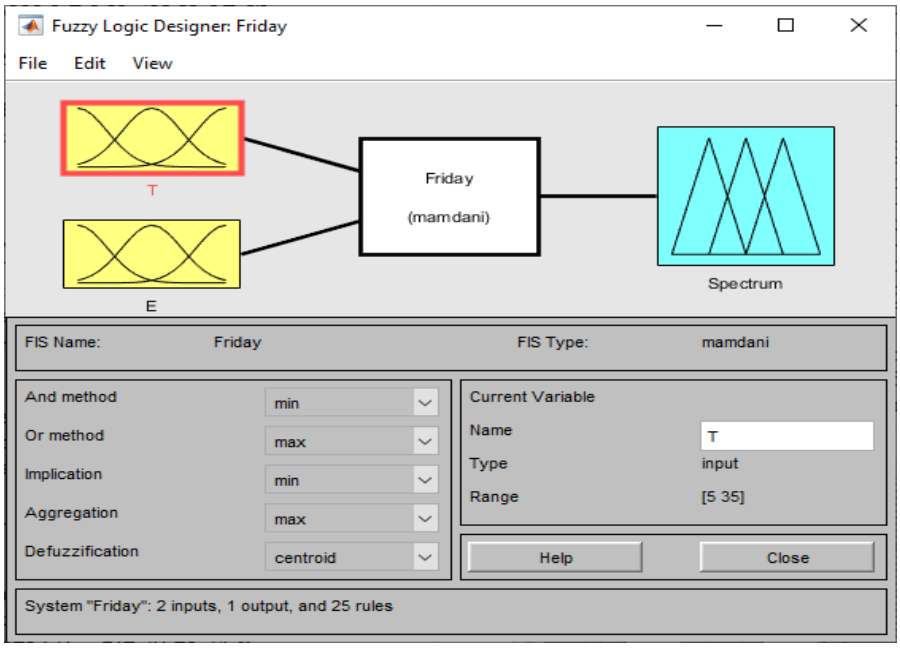


Рисунок 2.4 – Стартова сторінка Fuzzy logic Designer

На входах системи відбувається зчитування інформації з датчиків (датчик рівня освітлення, датчик температури). На виході маємо управління спектральним світлодіодним освітленням. Виходячи з вхідних показань система підбирає сприятливі умови досвітки згідно з нормами [6].

Під час побудови моделі було використано п'ятирівневу систему трикутних вхідних функцій. Даний параметр дозволяє більш чітко знімати

показання з датчиків для управління вихідним сигналом. Приклад реалізації для датчика температури наведено на рис.2.5.

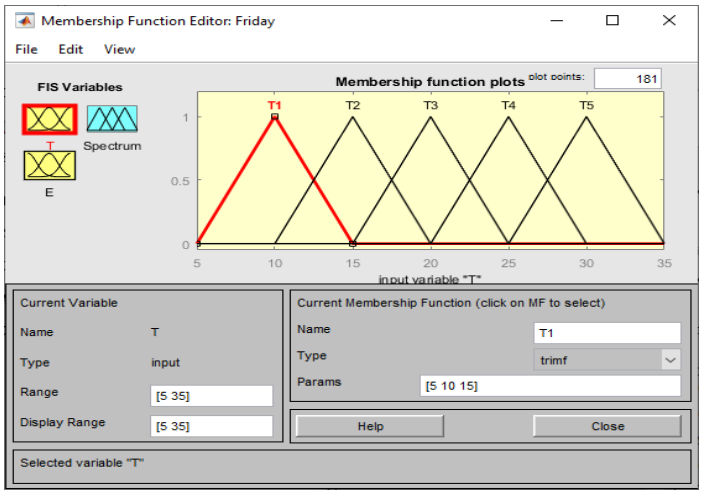


Рисунок 2.5 – Приклад побудови підсистеми для визначення температури

Кожен параметр записується в базу правил для подальшого управління системою спектрального освітлення в теплиці. На виході маємо систему Гауссових функцій, як показано на рис. 2.6. Це дозволяє перемикати кожне правило в залежності від заданих умов. Використання Гауссових функцій дозволяє плавно переходити з одного режиму в інший відповідно до змінюваних параметрів.

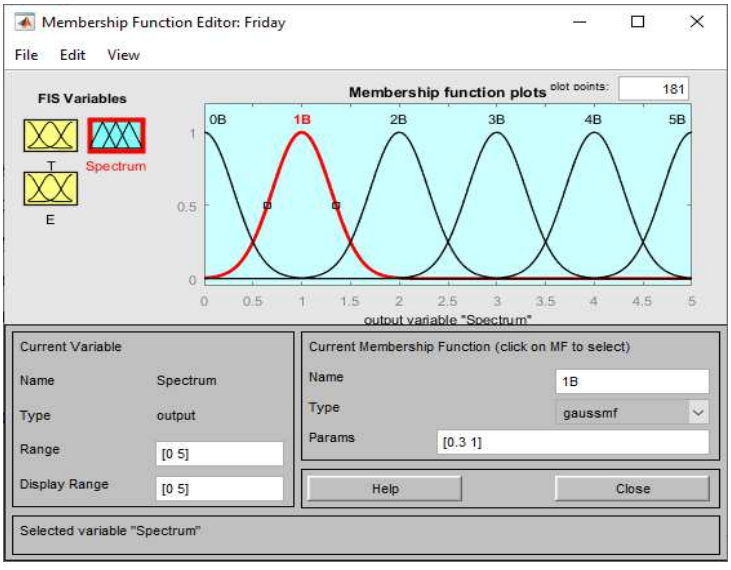


Рисунок 2.6 – Налаштування вихідних параметрів

Орієнтовна схема роботи розробленої системи показана на рис. 2.7. На схемі передбачено інтелектуальне управління нечіткою логікою в мікроконтролерному блоці системи.

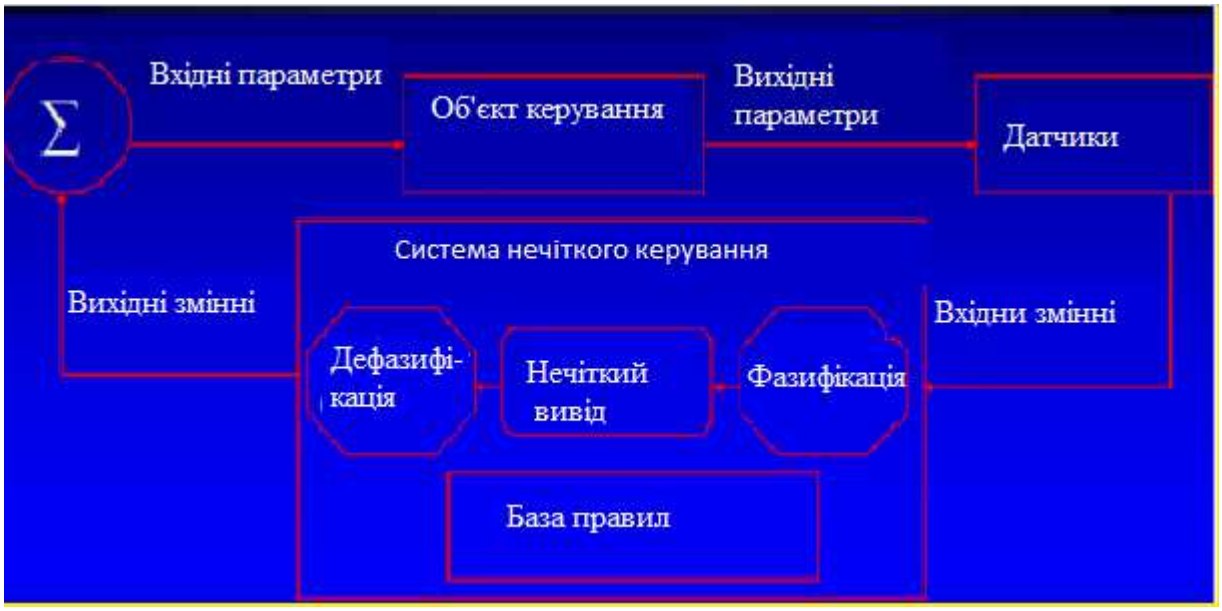


Рисунок 2.7 – Блок-схема роботи Fuzzy logic

Кожне правило обумовлено параметрами і даними з датчиків, для забезпечення роботи логіки. Згідно з даними, що наведено в табл. 1.1 у Fuzzy logic блоці реалізується регулювання параметрів мікроклімату теплиць.

Таблиця 1.1 – Матриця станів вхідних та вихідної змінних

T/E	**Night**	**Dawn**	**Morning**	**Noon**	**After-noon**
T_1 (20 °C)	Medium	Low	0ff	Off	Off
T_2 (21 °C)	More medium	Medium	Off	Off	Off
T_3 (22 °C)	High	More medium	Low	Off	Off
T_4 (23 °C)	More medium	Medium	Off	Low	Low
T_5 (24 °C)	Medium	Low	Off	Low	Low

База правил записаних у Matlab & Simulink® представлена на рис. 2.8. Записані правила регулюють освітлення виходячи з поточних значень двох вхідних параметрів: температура й інтенсивність освітлення зони вирощування.

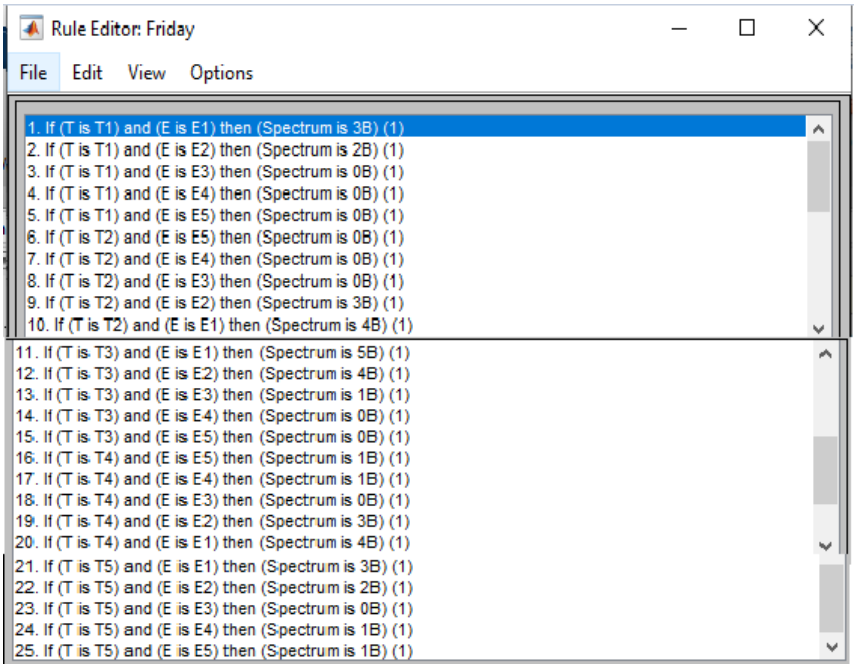


Рисунок 2.8 – База правил в Matlab & Simulink®

Дана робота бази правил відображається графічно для більш ергономічного сприйняття функціонування режимів роботи мікроконтролерного блоку, як показано на рис. 2.9.

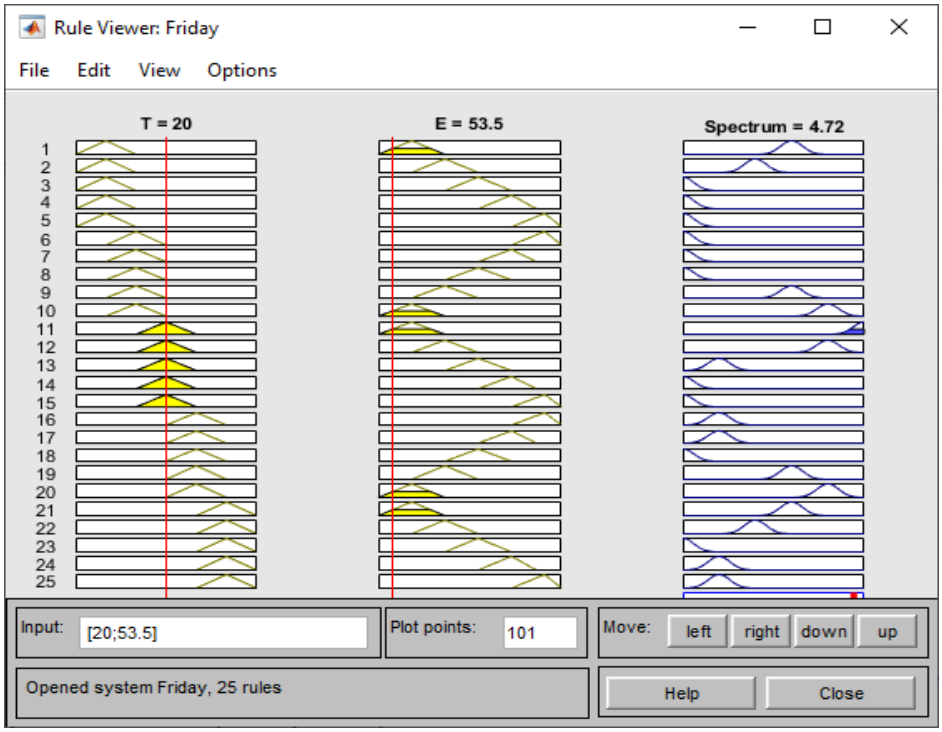


Рисунок 2.9 – Графічне зображення бази правил в Matlab & Simulink®

Виходячи з даного графіка можна проаналізувати режими роботи бази правил виходячи із записаних даних, де: *T* – вихідна інформація з датчика температури; *E* – вихідна інформація з датчика освітленості; *Spectrum* – вихідна спектральна потужність. Якщо значення температури знаходяться в нормі (від 20 до 24 °С) і показання освітленості знаходяться нижче норми (рослині недостатньо природного освітлення) вихідна змінна, що відповідає за систему досвічування встановлюється у максимальне значення потужності.

Залежність вихідних параметрів від вхідних відображена на рис. 2.10. Даний графік показує яким чином впливають вхідні параметри на режими управління спектральним освітленням. Графік складено в програмі Matlab & Simulink®. Параметри *T* і *E* знаходяться на осях *X* та *Y*. Вихідний сигнал спектра представлений на осі *Z*.

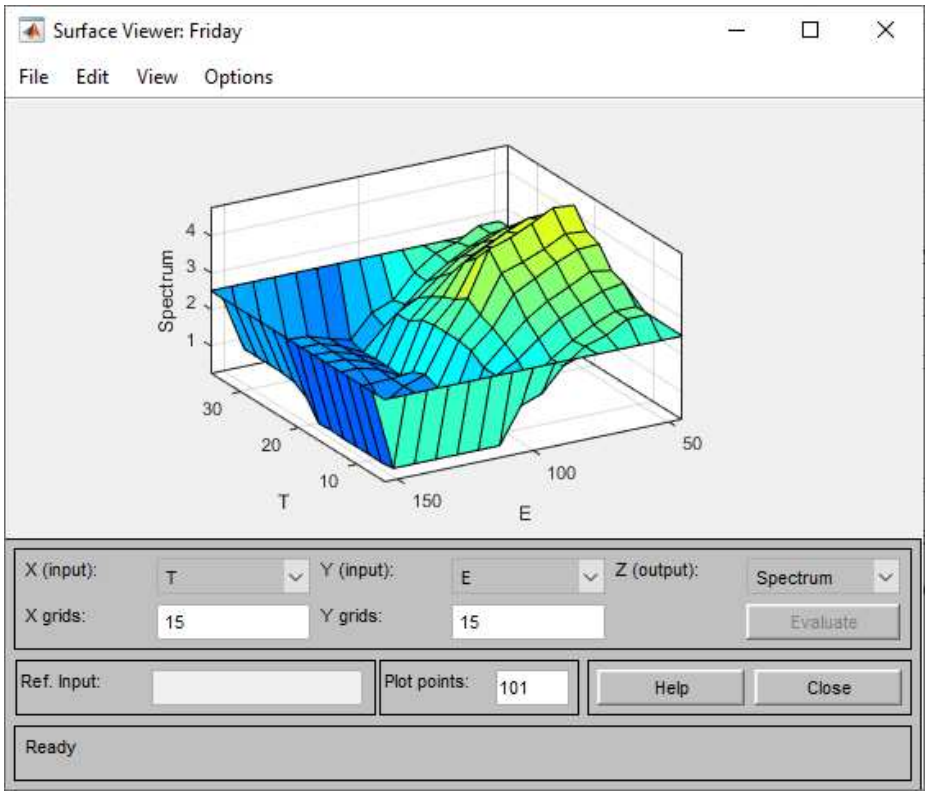
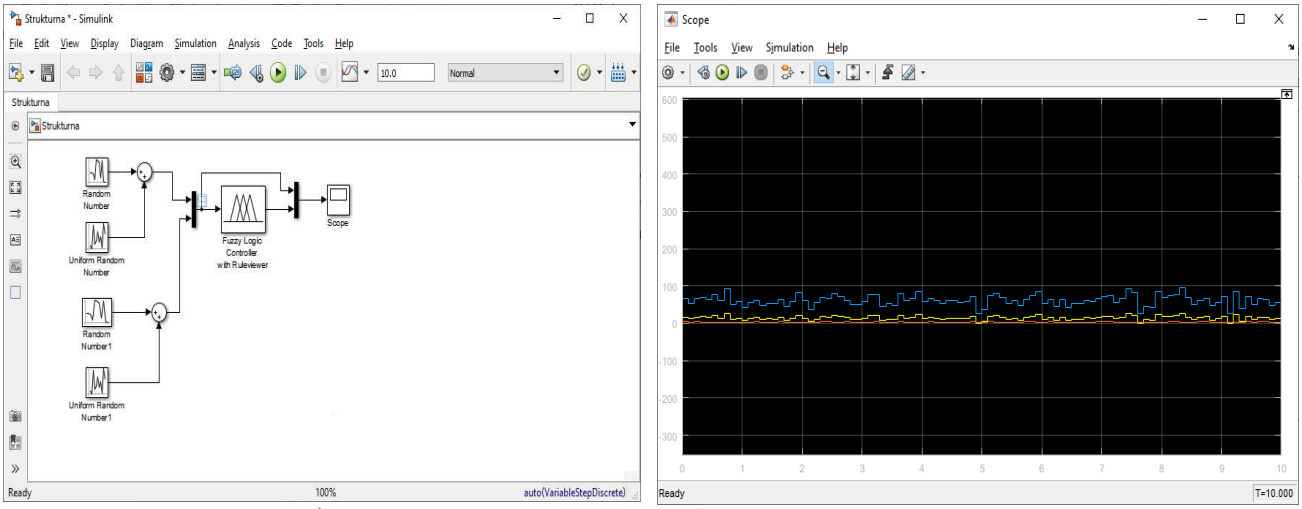


Рисунок 2.10 – Вихідний сигнал системи

Для перевірки роботи бази правил й моделі логіки було виконане моделювання в Simulink, як показано на рис 2.11, *а*. На входах підсумовується

значення випадкових значень для аналізу того яким чином буде себе вести модель. На виході маємо осцилограми роботи моделі в залежності від використання Fuzzy logic моделі, як показано на рис 2.11, *б*.



а) б)

Рисунок 2.11 – Результати моделювання структурної схеми на базі Simulink

2.3 Реалізація Fuzzy logic на базі мікроконтролеру Arduino

На базі мікроконтролеру Arduino було реалізовано програмну компоненту системи штучного досвічування зони вирощування тепличних культур. У програмі спектральний склад джерела штучного досвічування встановлюється в певному відсотковому співвідношенні спектра кожного типу світлодіодів: у діапазоні від 380 до 490 нм, де *R* – red (40 %), *B* – blue (20 %), *G* – green (40 %) [8].

За допомогою COM–портів на базі Arduino представлено наглядну реалізацію проекту і контроль регульованих параметрів рис. 2.12.

Вивід даних, який представлено на рис. 2.12, показує принципи роботи логіки в різних умовах освітлення з квазінезмінною кімнатною температурою, де: *Temp* – температура приміщення; *E* – рівень освітлення в приміщенні; *V* – діюче значення напруги на світлодіоді.

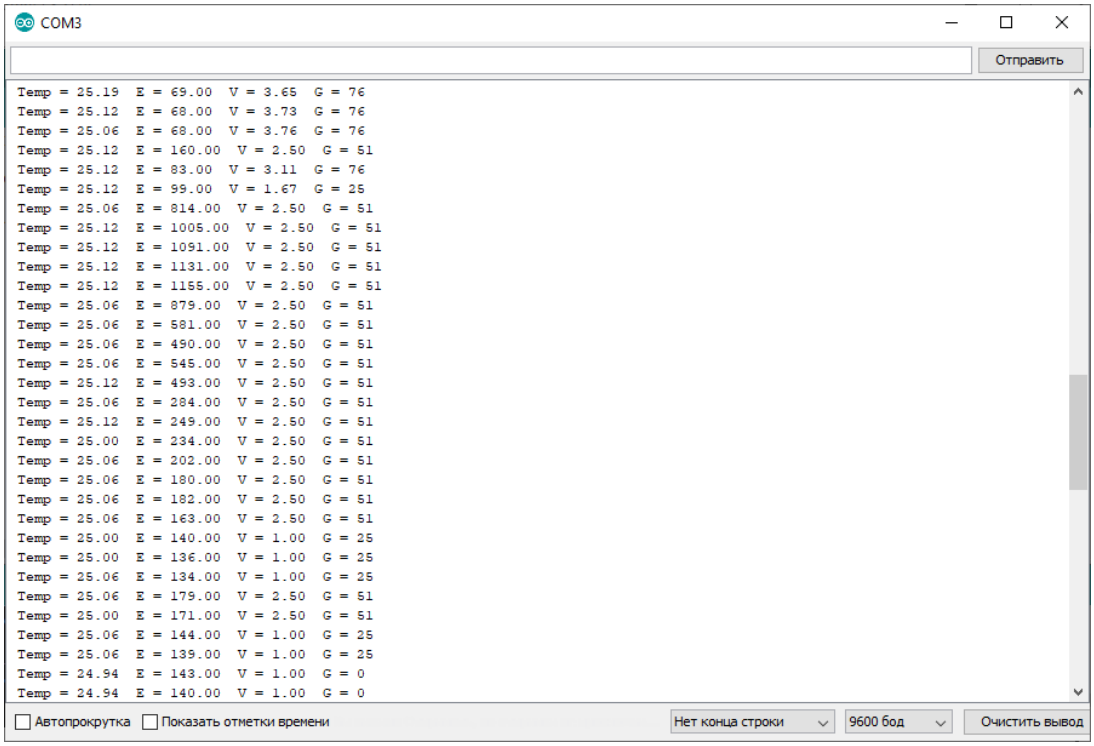


Рисунок 2.12 – Вивід результату вимірювання

Макетна реалізація досліджуваної системи представлена на рис. 2.16. В різних умовах освітлення та при різних значеннях температур виконується ШІМ-керування потужністю світлодіода згідно заданої бази правил.

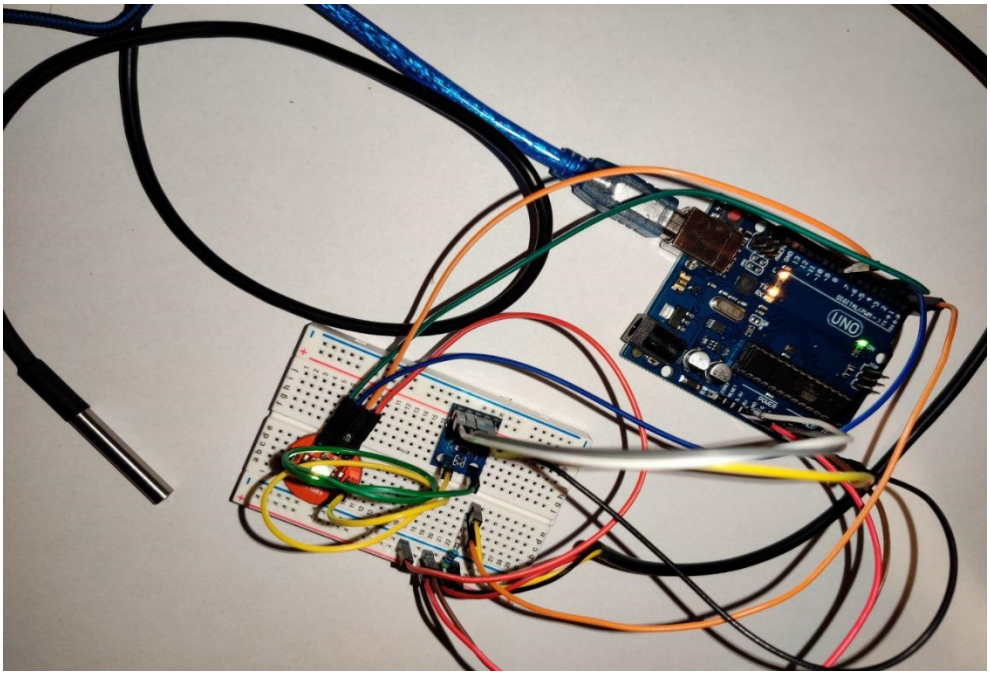


Рисунок 2.16 – Макетна реалізація системи

ВИСНОВКИ

У роботі вирішено проблему комп’ютеризованої системи керування штучним досвічуваням на базі пакету Matlab&Simulink for fuzzy logic Toolbox. Основні науково-практичні результати роботи можна представити в таких положеннях:

1. Проведено аналіз результатів досліджень щодо методів побудови моделі на базі Matlab & Simulink for fuzzy logic Toolbox. Реалізація такоъ технології у тепличних умовах дозволяє значно підвищити рівень їх комп'ютеризації й автоматизація задля приросту тепличних культур.

2. Обґрунтовано необхідний перелік матеріалів, комплектуючих та програмного забезпечення для проведення лабораторних експериментальних досліджень електронної системи.

3. Розроблено модель та базу правил нечіткого керування. Представлено реалізацію роботи на платформі Arduino. Встановлено спектральну складову у відсотковому співвідношенні спектра кожного типу світлодіодів: у діапазоні від 380 до 490 нм –20 %, від 490 до 590 нм – 40 %, від 590 до 700 нм – 40 %.

4. Визначено основні напрямки подальших досліджень електронної системи моніторингу й керування інтенсивністю штучного досвічування зони вирощування культур на базі бездротових мереж сенсорів.

ПЕРЕЛІК ПОСИЛАНЬ

1. T.A. Izzuddin, M.A. Johari, M. Z. A Rashid and M. H Jali: smart irrigation using fuzzy logic method. ARPN J. of Eng. and App. Sc. – 2018. Vol. 13, No. 2. – P. 1 – 6.

2. Пешко, М.С. Адаптивна система управління параметрами мікроклімату процесів виробництва та зберігання харчових продуктів: дис. … канд. тех. наук: 05. 03. 06. Омськ, 2015, 200 с.

3. Освітлення теплиць. URL: vasha-teplitsa.html (дата звернення: 05.11.2019)

4. Освітлення для теплиць своїми руками. URL: https://mirfermera.ru/371-kak-sdelat-osveschenie-dlya-teplic-svoimi-rukami.html (дата звернення: 15.11.2019).

5. T. Schwend, M. Beck, D. Prucker, S. Peisl, H. Mempel: Test of a PAR sensor-based, dynamic regulation of LED lighting in greenhouse cultivation of Helianthus annuus. European Journal of Horticultural Science. 2016. P. 152-155.

6. A.J. Both, L. Benjamin, J. Franklin, G. Holroyd, L.D. Incoll, M.G. Lefsrud, G. Pitkin. Guidelines for measuring and reporting environmental parameters for experiments in greenhouses. Plant Methods. 2015. Vol. 11. P. 1–18.

7. O. Vovna, I. Laktionov, S. Sukach, M. Kabanets, E. Cherevko. Method of adaptive control of effective energy lighting of greenhouses in the visible optical range. Bulg. Journal of Agricultural Science. Vol. 24, no. 2. (2018). P. 335–340.

8. Люксметри і принцип їх роботи. URL: https://simvolt.ua/sovremennye-lyuksmetry-nadezhnyy-kontrol-osveschennosti-v-lyubyh-usloviyah.html (дата звернення: 02. 01. 2020).

9. Л. Червінський, Т. Книжка, О. Романенко. Обґрунтування ефективного світлодіодного освітлення у спорудах закритого ґрунту. Науковий вісник НУБП України. Серія: Техніка та енергетика АПК. Київ, 2017. С.111-118.

10. Лактіонов І., Вовна О., Боричевський В., Лактіонова Г. Комп'ютеризована система моніторингу та керування штучним досвічуванням рослин у теплицях на базі нечіткої логіки. Перспективні напрямки сучасної електроніки, інформаційних та комп'ютерних систем: тези доп. IV; всеукр. наук. - практ. конф.(м. Дніпро 27-29 листопада 2019 р.). Дніпро, 2019. – С. 69 – 70.

Додаток А – Лістинг програми

```
#include "fis_header.h"
#include "Wire.h"
#include "BH1750.h"
#include "OneWire.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 = 25;

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

// Setup routine runs once when you press reset:
BH1750 lightMeter;
OneWire ds(2);
void setup()
{
    Serial.begin(9600);
    // initialize the Analog pins for input.
    // Pin mode for Input: T
    pinMode(8 , INPUT);
    // Pin mode for Input: E
    lightMeter.begin();
    // initialize the Analog pins for output.
    // Pin mode for Output: Green
    pinMode(5 , OUTPUT);
    // Pin mode for Output: Blue
    pinMode(6 , OUTPUT);
    // Pin mode for Output: Red
    pinMode(9 , OUTPUT);

}

// Loop routine runs over and over again forever:
void loop()
{
//     Определяем температуру от датчика DS18b20
      byte data[2]; // Место для значения температуры

      ds.reset(); // Начинаем взаимодействие со сброса всех предыдущих
команд и параметров
      ds.write(0xCC); // Даем датчику DS18b20 команду пропустить поиск по
адресу. В нашем случае только одно устрйоство
```

```
      ds.write(0x44);   //   Даем   датчику   DS18b20   команду   измерить
температуру. Само значение температуры мы еще не получаем - датчик его
положит во внутреннюю память

      delay(300); // Микросхема измеряет температуру, а мы ждем.

      ds.reset();  //  Теперь  готовимся  получить  значение  измеренной
температуры
      ds.write(0xCC);
      ds.write(0xBE);  //  Просим  передать  нам  значение  регистров  со
значением температуры
      // Получаем и считываем ответ
      data[0] = ds.read(); // Читаем младший байт значения температуры
      data[1] = ds.read(); // А теперь старший
      // Формируем итоговое значение:
      //    - сперва "склеиваем" значение,
      //        -  затем  умножаем  его  на  коэффициент,  соответсвующий
разрешающей способности (для 12 бит по умолчанию - это 0,0625)
      float Temp =  ((data[1] << 8) | data[0]) * 0.0625;
    // T
    g_fisInput[0] = Temp;
    // Read Input: E
    uint16_t lux = lightMeter.readLightLevel();
   g_fisInput[1] =lux;

    g_fisOutput[0] = 0;

    fis_evaluate();

    // Set output vlaue: Green
    int R = map(g_fisOutput[0],0,5,0,102);
    analogWrite(9,R);
    int G = map(g_fisOutput[0],0,4.8,0,102);
    analogWrite(5,G);
    int B = map(g_fisOutput[0],0,5,0,51);
    analogWrite(6,B);

    Serial.print(" Temp = ");
    Serial.print(g_fisInput[0]);

    Serial.print("  E = ");
    Serial.print(g_fisInput[1]);

    Serial.print("  V = ");
    Serial.print(g_fisOutput[0]);

    Serial.print("  G = ");
    Serial.println(G);
    delay(600);
}
```

```
//*********************************************************************
// Support functions for Fuzzy Inference System
//*********************************************************************
// 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);
}

// Gaussian Member Function
FIS_TYPE fis_gaussmf(FIS_TYPE x, FIS_TYPE* p)
{
    FIS_TYPE s = p[0], c = p[1];
    FIS_TYPE t = (x - c) / s;
    return exp(-(t * t) / 2);
}

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_trimf, fis_gaussmf
};

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

// Count of member function for each Output
int fis_gOMFCount[] = { 6 };

// Coefficients for the Input Member Functions
FIS_TYPE fis_gMFI0Coeff1[] = { 5, 10, 15 };
FIS_TYPE fis_gMFI0Coeff2[] = { 10, 15, 20 };
FIS_TYPE fis_gMFI0Coeff3[] = { 15, 20, 25 };
FIS_TYPE fis_gMFI0Coeff4[] = { 20, 25, 30 };
FIS_TYPE fis_gMFI0Coeff5[] = { 25, 30, 35 };
FIS_TYPE*   fis_gMFI0Coeff[]   =   {   fis_gMFI0Coeff1,   fis_gMFI0Coeff2,
fis_gMFI0Coeff3, fis_gMFI0Coeff4, fis_gMFI0Coeff5 };
FIS_TYPE fis_gMFI1Coeff1[] = { 45, 65, 85 };
FIS_TYPE fis_gMFI1Coeff2[] = { 65, 85, 105 };
FIS_TYPE fis_gMFI1Coeff3[] = { 85, 105, 125 };
FIS_TYPE fis_gMFI1Coeff4[] = { 105, 125, 140 };
FIS_TYPE fis_gMFI1Coeff5[] = { 125, 145, 155 };
FIS_TYPE*   fis_gMFI1Coeff[]   =   {   fis_gMFI1Coeff1,   fis_gMFI1Coeff2,
fis_gMFI1Coeff3, fis_gMFI1Coeff4, fis_gMFI1Coeff5 };
FIS_TYPE** fis_gMFICoeff[] = { fis_gMFI0Coeff, fis_gMFI1Coeff };

// Coefficients for the Output Member Functions
FIS_TYPE fis_gMFO0Coeff1[] = { 0.3, 1 };
FIS_TYPE fis_gMFO0Coeff2[] = { 0.3, 3 };
FIS_TYPE fis_gMFO0Coeff3[] = { 0.3, 2 };
FIS_TYPE fis_gMFO0Coeff4[] = { 0.3, 4 };
FIS_TYPE fis_gMFO0Coeff5[] = { 0.3, 5 };
FIS_TYPE fis_gMFO0Coeff6[] = { 0.3, 0 };
FIS_TYPE*   fis_gMFO0Coeff[]   =   {   fis_gMFO0Coeff1,   fis_gMFO0Coeff2,
fis_gMFO0Coeff3, fis_gMFO0Coeff4, fis_gMFO0Coeff5, fis_gMFO0Coeff6 };
FIS_TYPE** fis_gMFOCoeff[] = { fis_gMFO0Coeff };

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

// Output membership function set
```

```
int fis_gMFO0[] = { 1, 1, 1, 1, 1, 1 };
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 };

// 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 };

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

// Rule Outputs
int fis_gRO0[] = { 2 };
int fis_gRO1[] = { 3 };
int fis_gRO2[] = { 6 };
int fis_gRO3[] = { 6 };
int fis_gRO4[] = { 6 };
int fis_gRO5[] = { 6 };
int fis_gRO6[] = { 6 };
```

```
int fis_gRO7[] = { 6 };
int fis_gRO8[] = { 2 };
int fis_gRO9[] = { 4 };
int fis_gRO10[] = { 5 };
int fis_gRO11[] = { 4 };
int fis_gRO12[] = { 1 };
int fis_gRO13[] = { 6 };
int fis_gRO14[] = { 6 };
int fis_gRO15[] = { 1 };
int fis_gRO16[] = { 1 };
int fis_gRO17[] = { 6 };
int fis_gRO18[] = { 2 };
int fis_gRO19[] = { 4 };
int fis_gRO20[] = { 2 };
int fis_gRO21[] = { 3 };
int fis_gRO22[] = { 6 };
int fis_gRO23[] = { 1 };
int fis_gRO24[] = { 1 };
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 };

// Input range Min
FIS_TYPE fis_gIMin[] = { 5, 45 };

// Input range Max
FIS_TYPE fis_gIMax[] = { 35, 155 };

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

// Output range Max
FIS_TYPE fis_gOMax[] = { 5 };

//***************************************************************************
// 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 };
    FIS_TYPE* fuzzyInput[fis_gcI] = { fuzzyInput0, fuzzyInput1, };
    FIS_TYPE fuzzyOutput0[] = { 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);
            }
        }
    }
```