Міністерство освіти і науки України

Національний університет «Львівська політехніка»

На правах рукопису

БЕРЕГОВСЬКИЙ ВАСИЛЬ ВАСИЛЬОВИЧ

УДК 004.942; 004.02

МАТЕМАТИЧНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ СИСТЕМ “ІНТЕЛЕКТУАЛЬНОГО БУДИНКУ”

Спеціальність: 05.13.12 – системи автоматизації проектувальних робіт

Дисертація на здобуття наукового ступеня
кандидата технічних наук

Науковий керівник:
Теслюк Василь Миколайович
доктор технічних наук,
професор

Львів – 2017

ЗМІСТ

ВСТУП

Актуальність теми. В умовах надзвичайно швидкого розвитку мікропроцесорної техніки та побудови на їх основі програмованих пристроїв керування усіма видами побутового обладнання виникає потреба і, головне можливість максимального забезпечення реалізації зростаючих вимог до комфортних умов проживання та праці мільйонів людей, гарантування їхньої безпеки та захисту від наслідків технічних аварій. Це стало одним з пріоритетних напрямків розвитку сучасної радіоелектронної та обчислювальної техніки. Крім того інтеграція обчислювальних програмованих систем в пристроях телекомунікацій забезпечує можливість поєднання різних комплексів, що знаходяться на значній відстані, а використання стандартизованих протоколів передачі даних робить можливим здійснення взаємодії комунальних служб для оперативного вирішення можливих надзвичайних ситуацій, а також здійснення дистанційного контролю та керування роботою таких систем. Реалізація цих завдань забезпечується широкомасштабним впровадженням технологій “інтелектуального будинку”. Ядром таких комплексів є апаратно-програмна система, що дає змогу забезпечити максимально комфортні умови проживання з задоволенням індивідуальних вимог та потреб користувача та дозволяє досягти суттєвого скорочення енергетичних витрат за рахунок раціонального використання енергоносіїв. Вирішення цих задач залежить від характеристик житла та рівня його комфортності.

Значний внесок в теорію та практику автоматизованого проектування систем “інтелектуального будинку” внесли: *Seaman M., Nabih A. K. та ін.* – моделі системного рівня проектування; *Ткаченко Р.О., Хайкіна С., Івахненко О., Бодянський Є. та ін.* – моделі опрацювання нечітких та неструктурованих даних на основі штучних нейронних мереж; *Walker I., Sherman M.* – моделювання систем вентиляції в системах “інтелектуального будинку”; *Lee H., Kwon J.* – використання

моделей на основі онтологій для моделювання в області розроблення систем “інтелектуального будинку”; *Ogawa T., Aydin E. A.* – керування пристроями без фізичного контакту, та ін.

В процесі автоматизованого проектування таких систем, використовується блочно-ієрархічний підхід, який передбачає використання методів, моделей та засобів, що враховують специфіку та особливості об’єкту проектування. Аналіз існуючих методів та моделей для автоматизації проектування систем “інтелектуального будинку”, дає змогу стверджувати про необхідність розроблення моделей для системного рівня автоматизованого проектування, методу автоматизованого синтезу таких моделей та моделей для опрацювання нечітких та неструктурованих даних від підсистеми давачів. Тому тема дисертаційного дослідження “Математичне та програмне забезпечення автоматизованого проектування систем “інтелектуального будинку” є актуальною.

Зв’язок роботи з науковими програмами, планами, темами.

Дисертаційні дослідження виконувалися відповідно до наукового напрямку кафедри “Системи автоматизованого проектування” Національного університету “Львівська політехніка”: “Автоматизація проектування та моделювання вбудованих систем”, “Автоматизація проектування та моделювання систем “розумного будинку”. Дисертація виконана в межах науково-дослідної роботи “Розроблення базових компонентів для синтезу інтелектуальних мобільних робототехнічних систем” (номер державної реєстрації 0113U003191).

Мета і завдання дослідження. Метою дисертаційної роботи є підвищення ефективності автоматизованого проектування систем “інтелектуального будинку” на основі розробленого методу, моделей та засобів.

Для досягнення поставленої мети необхідно було розв’язати такі задачі:

– здійснити аналіз існуючих методів, моделей і засобів автоматизованого проектування систем “інтелектуального будинку”;

– ввести інтелектуальну складову в процес автоматизованого проектування та розробити метод автоматизованого синтезу моделей на основі теорії мереж Петрі;

– розробити структурні моделі на основі мереж Петрі для системного рівня автоматизованого проектування систем “інтелектуального будинку”;

– розробити моделі для реалізації інтелектуальних функцій на основі штучних нейронних мереж, які дають змогу опрацьовувати неструктуровані та нечіткі дані від підсистеми збору даних про оточуюче середовище;

– розробити фізичні моделі роботи підсистем “інтелектуального будинку”;

– програмно реалізувати розроблені методи та моделі, розробити математичне, інформаційне та технічне забезпечення систем для автоматизації проектування систем “інтелектуального будинку”.

Об’єктом дослідження є процес автоматизованого проектування систем “інтелектуального будинку”.

Предметом досліджень є метод, моделі та засоби автоматизації проектування систем “інтелектуального будинку”.

Методи досліджень. У дисертаційній роботі для розв’язання поставлених задач використано: при розробленні методу, моделей, засобів і алгоритмів – теорія системного аналізу, теорія кольорових та простих мереж Петрі, теорія графів; при розробленні математичних моделей – теорія математичного моделювання та теорія штучних нейронних мереж; при розробленні програмних моделей – принципи об’єктно-орієнтованого програмування.

Наукова новизна одержаних результатів:

1. Вперше розроблено моделі системного рівня автоматизованого проектування систем “інтелектуального будинку”, які ґрунтуються на теорії кольорових мереж Петрі та дають змогу визначити динаміку роботи, перевірити спроектовану систему на наявність тупиків, на живучість та обмеженість.

2. Вперше введено інтелектуальний аспект на усіх рівнях автоматизованого проектування таких систем та сформульовано основні задачі на кожному з них, що дає змогу підвищити ефективність автоматизованого проектування систем “інтелектуального будинку”.

3. Вдосконалено моделі підсистем клімат-контролю, освітлення, захисту та запобігання технічних аварій “інтелектуального будинку”, які використовують

штучну нейронну мережу на основі багатошарового перцептрона, що дає змогу опрацьовувати нечіткі та неструктуровані дані від підсистеми давачів.

4. Отримав подальший розвиток метод автоматизованого синтезу моделей на основі теорії мереж Петрі для системного рівня автоматизованого проектування, який ґрунтується на інформації про структуру системи і теорію графів та дає змогу автоматизувати побудову структурних моделей підсистем “інтелектуального будинку”.

5. Отримали подальший розвиток фізичні моделі підсистем клімат-контролю, освітлення, захисту та запобігання технічних аварій “інтелектуального будинку” у формі нейроконтролера, які використовують мікроконтролер AVR та програмні моделі на основі штучної нейронної мережі, і дають змогу дослідити адекватність побудованих моделей, швидкодію, надійність та функціональність розроблених підсистем.

Практичне значення одержаних результатів.

1. На основі розробленого методу і моделей та їх програмної реалізації побудовано структуру системи автоматизованого синтезу моделей на основі теорії мереж Петрі та системи побудови і дослідження моделей на основі штучних нейронних мереж.

2. Розроблено алгоритми функціонування системи автоматизованої побудови моделей на основі теорії мереж Петрі та системи побудови та дослідження моделей на основі штучних нейронних мереж.

3. Розроблено програмне забезпечення нейроконтролерів, яке враховує специфіку та особливості мікроконтролера AVR, використовує мову високого рівня та програмні моделі штучних нейронних мереж, що дає змогу швидко вносити зміни в функціональність нейроконтролера та забезпечує зниження його вартості.

4. Розроблено математичне та інформаційне забезпечення системи автоматизованої побудови моделей на основі теорії мереж Петрі та системи побудови і дослідження моделей на основі штучних нейронних мереж.

Теоретичні та практичні результати дисертаційної роботи впроваджені при розробці перспективних радіоелектронних систем, які працюють у режимі реального часу в Фізико-механічному інституті ім. Г.В. Карпенка НАН України, що підтверджено відповідним актом.

Особистий внесок здобувача. Всі наукові результати теоретичних і практичних досліджень, викладених в дисертації, одержано автором особисто. У друкованих працях, опублікованих у співавторстві, автору належать: [1] – структура та модель системного рівня автоматизованого проектування “інтелектуального будинку”; [2] – запропоновано використовувати інтелектуальний аспект у процесі автоматизованого проектування систем “інтелектуального будинку”; [3] – структура розгалуженої системи; [4] – структура підсистеми, програмне та інформаційне забезпечення; [5] – моделі підсистеми для системного рівня проектування; [6-9] – моделі для опрацювання нечітких та неструктурованих даних від підсистеми давачів і фізичні моделі підсистем “інтелектуального будинку”; [10] – модель підсистеми клімат-контролю, структура системи та інформаційне забезпечення; [11] – інформаційне забезпечення методу автоматизованого синтезу моделей; [12] – метод автоматизованого синтезу моделей системного рівня проектування ; [13] – структура нейроконтролера, алгоритм його роботи та програмне забезпечення; [14] – модель підсистеми клімат-контролю на основі штучних нейронних мереж; [15] – моделі підсистеми “інтелектуального будинку” на основі мереж Петрі; [16, 17] – моделі на основі теорії мереж Петрі; [18] – модель на основі мереж Петрі; [19] – структура системи, алгоритм та модель на основі кольорових мереж Петрі; [20] – структура, програма та інформаційне забезпечення системи; [21] – фізична модель.

Апробація роботи. Основні теоретичні положення та практичні результати дисертаційної роботи доповідалися і обговорювались на: Міжнародній науково-технічній конференції “Перспективні технології і методи проектування МЕМС” (“Perspective Technologies and Methods in MEMS Design”), MEMSTECH, (Поляна – Свалява (Закарпаття), 2011, 2013); Міжнародній науково-технічній конференції “Комп’ютерні науки та інформаційні технології” (“Computer Sciences and

Information Technologies"), CSIT, (Львів, 2012); Міжнародній науково-практичній конференції "Сучасні проблеми і досягнення в галузі радіотехніки, телекомунікацій та інформаційних технологій" (Запоріжжя, 2012); Всеукраїнській школі-семінарі молодих вчених і студентів "Сучасні комп'ютерні інформаційні технології", ACIT'2013, (Тернопіль, 2013); XVIIIth International Seminar/Workshop on Direct and Inverse Problems of Electromagnetic and Acoustic Wave Theory, DIPED – 2013, (Львів, 2013); XIIth International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science, TCSET'2014, (Lviv-Slavsko, 2014); а також на наукових семінарах кафедри систем автоматизованого проектування Національного університету "Львівська політехніка" (2012-2016).

Публікації. За результатами досліджень, які викладені в дисертаційній роботі, опубліковано 21 наукова праця, у тому числі 12 статей, з них 11 у фахових наукових виданнях України та 1 у закордонному періодичному виданні, 6 включено до наукометричних баз даних, 9 матеріалів міжнародних та всеукраїнських науково-технічних конференцій.

Структура і обсяг роботи. Дисертація складається зі вступу, 4-х розділів, висновків, списку використаної літератури та додатків. Загальний обсяг дисертації становить 212 сторінок, у тому числі 149 сторінок основного тексту, 70 рисунків та 17 таблиць, список використаної літератури налічує 202 бібліографічних найменування.

РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ, МОДЕЛЕЙ ТА ЗАСОБІВ АВТОМАТИЗАЦІЇ ПРОЕКТУВАННЯ СИСТЕМ “ІНТЕЛЕКТУАЛЬНОГО БУДИНКУ”

1.1. Особливості систем “інтелектуального будинку”

Тенденції розвитку сучасних технічних систем передбачають зменшення їх розмірів та маси, нарощення функціональності та інтелектуалізації, що значною мірою обумовлено жорсткими вимогами ринкової економіки, необхідністю зменшення енергетичних затрат, підвищення надійності, задоволення зростаючих вимог користувачів тощо [22, 23].

В таких умовах з’являються нові системи, які ґрунтуються на нових технологіях, сучасній елементній базі, високому рівні інтелектуалізації функцій та інше [24].

Однією з таких технологій є системи “інтелектуального будинку” [25, 26, 27, 28, 29]. В літературі можна зустріти ряд інших назв цієї технології, а саме: “Smart House”, “Smart Home”, “Intelligent Building” та ін. Детальний аналіз вказаних термінів дає змогу виокремити та проаналізувати їх особливості, специфіку та відмінності.

Загалом під технологією “інтелектуального будинку” [30, 31, 32, 33, 34] здебільшого розуміють систему, що об’єднує в собі ряд підсистем, які забезпечують комфортні умови проживання мешканців в приміщенні та дають змогу суттєво зменшити витрати енергоносіїв. При цьому, в процесі управління такою системою забезпечується централізоване керування з одного технічного пристрою, виконаного у формі пульта. Ця система, як правило, включає такі основні підсистеми як: клімат-контролю, освітлення, захисту, підсистему централізованого керування аудіо- та відеосистемами і побутовими системами, підсистему запобігання технічних аварій та ін.

На практиці, при реалізації системи “інтелектуальний будинок”, в сучасному розумінні, в більшості випадків власники таких помешкань встановлюють

підсистеми клімат-контролю, освітлення, захисту, керування побутовими пристроями та інші.

Під терміном “Smart House” [35] мають на увазі сучасний житловий будинок призначений для комфортного проживання мешканців шляхом використання високотехнологічних пристроїв, які об'єднані в домашню Universal Plug'n'Play - мережу з можливістю виходу в мережі загального користування.

Назва “Intelligent building” [36, 37] використовується для системи, яка інтегрована в будівлю і має здатність розпізнавати різні ситуації, що відбуваються у приміщенні та відповідним чином реагувати на них. Ця реакція визначається згідно з наперед розробленим та запрограмованим алгоритмом. Можна стверджувати, що система наділена певним рівнем інтелекту, про що свідчить слово intelligent.

Під терміном “Розумний будинок” [37] розуміють систему, яка володіє технічними засобами, здатними адаптуватися, підлагоджуватися під різні зміни як в даний момент часу, так і в майбутньому. Для прикладу, це може бути адаптація під нового власника будинку чи квартири та його потреб. Досить часто таку систему можна охарактеризувати як “будівлю, готову до змін” або “гнучку будівлю”. Отже термін “розумний будинок” вимагає найвищого рівня інтелектуалізації інженерних систем.

Надалі, у дисертаційній роботі, будемо притримуватися однієї назви – “інтелектуальний будинок” (ІБ).

Багато сучасних пристроїв спершу з’явились у фантастичній літературі, а лише згодом в реальному житті. Це стосується також ідеї “інтелектуального будинку”, яка виникла ще наприкінці XIX сторіччя з ідеї дистанційного керування пристроями [38], але матеріалізувалася лишень у XX-му сторіччі після широкого впровадження у побуті електричних пристроїв та розвитку інформаційних технологій [32, 39, 40].

Поняття “інтелектуальний будинок” виникло в 70-ті роки минулого сторіччя [41] як будинок, що забезпечує продуктивне і ефективне використання робочого простору та приміщень. Невірно розуміти під терміном “інтелектуальний будинок”

дослівний переклад з англійської “мислячий будинок”, тому що цей термін означає систему, яка вміє розпізнавати конкретні ситуації, що відбуваються в будинку, і відповідним чином на них реагувати, тобто слово intelligent, у поєднанні зі словом building має значення “гнучкий, що пристосовується”.

Вперше термін “інтелектуальний будинок” було використано Американською Асоціацією Housebuilders у 1984 році [32]. Перший будинок був оснащений технологіями “розумного будинку” в 1986 році компаніями AT & T і Honeywell – виробниками кабельної продукції і автоматичних пристроїв, які шукали нові сфери збуту та застосування своєї продукції [32].

На сьогоднішній день в керований комплекс “інтелектуальний будинок” можна об’єднувати окремі підсистеми різних виробників. На етапі проектування передбачається можливість оптимального інтегрування систем з мінімальними витратами та можливістю видозміни конфігурації.

Сучасні побутові пристрої оснащені елементами “штучного інтелекту”, і система може відслідковувати їхню роботу та приймати рішення при зміні певних параметрів.

З наведених визначень і пояснень систем “інтелектуального будинку” випливає, що ця система переслідує дві основні мети – забезпечення комфорту власника будівлі та забезпечення економії енергоносіїв.

Як було вище зазначено, до особливостей систем “інтелектуального будинку” можна віднести і значні економічні переваги [42]. Зокрема, економію енергоносіїв, яка, за інформацією з джерел в Європі та США, може сягати до 60%. В Україні такого ефекту можна очікувати через 10-15 років, що дасть змогу покращити енергонезалежність держави [43, 44, 45], а необхідність розроблення та вдосконалення таких систем підтверджує актуальність даного дисертаційного дослідження.

Слід також додати, що в більшості розвинених країн ще на етапі проектування житла передбачається встановлення системи “інтелектуального будинку”. Це є вигідною інвестицією частини коштів як для держави, так і для власників житла. В середньому дана інвестиція складає понад 5 - 10% від вартості

будівництва житла [46]. В Україні при проектуванні, здебільшого, не передбачають встановлення таких систем. Відповідно, якщо власник помешкання бажає встановити систему “інтелектуального будинку” після виконання будівельних робіт, це призводить до додаткових затрат та викликає певні труднощі [47].

Інтелектуалізація торкається не тільки приватних квартир та будинків. Забезпечення комфортних умов необхідно реалізовувати і в адміністративних та виробничих приміщеннях.

Згідно з даними з різних джерел, капіталовкладення в систему “інтелектуального будинку” окуповується в середньому від 3 до 10 років, в залежності від відсотка інтелектуалізації приміщення.

Іншою особливістю системи “інтелектуального будинку” є те, що вона дає змогу підвищити рівень комфорту для особи, яка проживає в приміщенні, наприклад, коли ми говоримо про кліматичні параметри всередині приміщення. Разом з тим, ці параметри можна змінювати або ж вони можуть автоматично змінюватися системою в залежності від присутності чи відсутності власника, часу доби та пори року.

Система “інтелектуального будинку” забезпечує ефективне та економне керування освітленням, захистом як самого помешкання, так і периметру прибудинкової території, керування побутовими пристроями тощо.

Все частіше починають впроваджуватися в систему “інтелектуального будинку” нові підсистеми, які реалізують нові функції такі як підсистема керування робототехнічними засобами [48, 49, 50, 51, 52, 53] з метою підвищення рівня комфорту для неповносправних осіб; технологія “розумний холодильник” [54], що відслідковує кількість продуктів у холодильнику і формує список тих, які необхідно купити; “розумна пральна машина” [55], що може самостійно підбирати режим прання, дозувати миючі засоби в залежності від інформації на ярликах одягу; підсистема контролю фізичного стану особи та інше. Разом з тим, все частіше йде мова про об’єднання “інтелектуальних будинків” у системи та інтелектуальне керування ресурсами та послугами у цілому населеному пункті.

Таким чином розвивається технологія “розумного міста” [56, 57, 58, 59, 60, 61, 62, 63].

За даними закордонної статистики, в середньому, система “інтелектуального будинку” забезпечує [64]:

– зниження експлуатаційних витрат – 30%;

– зниження платежів за електроенергію – 30%;

– зниження платежів за воду – 40%;

– зниження платежів за тепло – 50%;

– зменшення викидів вуглекислого газу – 30%;

– зменшення пільг з страхування ризиків до 60%.

Отже, система “інтелектуального будинку” дає змогу істотно підвищити ефективність використання енергоресурсів, рівень комфорту, рівень безпеки тощо.

Коли ми говоримо про функції, які має реалізувати система “інтелектуального будинку”, то в кожного власника такої будівлі вони дещо відрізняються, що викликає певні труднощі в процесі її реалізації та обумовлює необхідність постійного вдосконалення таких систем, використання налаштування і програмування певних функцій. Зокрема, за результатами опитування, які були проведені Інтернет-виданням CNews [65], вимоги власників помешкань розподілились, а саме (табл. 1.1): на першому місці є зручність, потім – простота використання і т.д.

З отриманих досліджень можемо зробити висновок, що найбільші очікування від системи “інтелектуального будинку” пов’язані з комфортом проживання, а вимога економії енергоносіїв займає лише дев’яте місце. Така ситуація справедлива для закордонних власників систем “інтелектуального будинку”. Стосовно України, більша частина споживачів встановлює системи “інтелектуального будинку” передовсім для зменшення витрат енергоносіїв, а вже потім – для забезпечення комфорту та підтвердження свого статусу. Розробники та постачальники таких систем, як правило, мають певний базовий комплект для реалізації системи “інтелектуального будинку” і пропонують встановити його власнику будівлі.

Таблиця 1.1. Сучасні вимоги до систем “інтелектуального будинку”

Вимоги	Значення, %
- зручність	30
- простота використання	17
- центральне управління	10
- надійність	8
- задоволення/розваги	8
- комфорт	8
- вартість/якість	6
- естетика/зовнішній вигляд	5
- економія	4
- безпека	3
- сервіс	1

За даними з різних джерел сьогодні у світі ринок систем “інтелектуального будинку” складає близько 15 млрд. євро. Найбільше таких систем реалізовано в Європейському союзі (приблизно 40 %), трохи менше у Північній Америці (25 %) та Японії (20%). На усі інші країни припадає біля 15 %. Зазначені дані зображено в табл. 1.2. Щодо України, то такі дані відсутні. Разом з тим, оптимістичним показником є те, що кількість замовлень з кожним роком зростає в середньому на 20%.

Таблиця 1.2. Показник світового ринку систем “інтелектуального будинку”

Країна	Відсотковий показник, %
Європейський Союз	40%
Північна Америка	25%
Японія	20%
Інші	15%

Щодо вартості систем “інтелектуального будинку”, то ціна залежить від тих функцій, які має реалізувати розроблена система. За різними оцінками середня вартість системи “інтелектуального будинку” складає біля 10-15 тисяч євро. Згідно з даними, наведеними в пропозиціях фірм, що розробляють і встановлюють

системи “інтелектуального будинку” [66], вартість таких систем з кожним роком швидко зменшується. Буквально кілька років тому для квартири з площею 70 м^2 з можливістю повного автоматизованого керування – вартість розроблення та реалізації починалася з 9-10 тисяч євро. В даний час реалізація системи “інтелектуального будинку” з тими ж можливостями складає біля 4-5 тисяч євро, а у випадку використання бездротових технологій, які не потребують прокладання комунікацій – 2 тисячі євро. При цьому дана система реалізує такі функції як клімат-контроль, освітлення, захист, управління побутовими та розважальними пристроями.

Системи “інтелектуального будинку” для невеликих котеджів з площею близько 300м^2 коштують у середньому біля 16 тисяч євро. Слід зауважити, що до цієї ціни необхідно додати ще 15-30% від вартості коштів, пов’язаних з встановленням та налаштуванням систем “інтелектуального будинку”. Система для котеджів з керуванням 12-ма кліматичними приміщеннями, управлінням усіма елементами підсистеми освітлення, системою відеоспостереження на 8 камер, сенсорною панеллю керування, пожежно-охоронною підсистемою з найвищим рівнем інтелектуалізації обійдеться приблизно у 29-30 тисяч євро. Підвищення складності системи “інтелектуального будинку” пов’язане зі зростанням її ціни, яка може досягати за різними оцінками до 90-100 тисяч євро [67].

1.2. Рівні інтелектуалізації сучасних будівель

Будівля називається “інтелектуальною будівлею” [30, 31, 32, 33, 34], коли в ній наявна якась комп'ютерна чи контролююча система управління інженерним оснащенням. Ступінь керованості чи контролю встановленого обладнання (або підсистем) визначає рівень інтелектуалізації будівлі. Всі споруди можна поділити на 5 рівнів інтелектуалізації в залежності від оснащення інженерними системами. Кожен наступний рівень включає як можливості попереднього, так і додаткові можливості [68, 69].

Нульовий рівень – це звичайні сучасні споруди – “не інтелектуальні”, хоч і обладнані всіма необхідними системами життєзабезпечення [68].

Інтелектуальні споруди першого рівня – рівня “активного регулювання” – це системи, оснащені індикаторами стану та пристроями керування. Користувач має можливість переглянути інформацію про стан систем та, при потребі, регулювати роботу побутових пристроїв і систем: автоматичне включення/виключення освітлення та регулювання рівня освітлення в залежності від часу доби; опалення та вентиляція регулюється терморегуляторами та вентиляторами; автоматичне вимикання побутових пристроїв в разі їх невикористання протягом певного проміжку часу [68].

Другий рівень – рівень “централізованого керування” [68]. На цьому рівні в споруді окремі системи життєзабезпечення об’єднуються через загальну структуровану кабельну систему (інтеграція інженерних систем) та централізовано дистанційно керуються через центральний контролер: система клімат-контролю керує температурою, вологістю та іонізацією повітря в приміщенні, система безперебійного живлення контролює використання електроенергії та здійснює захист від перевантажень, система освітлення обладнана аварійним джерелом живлення, система контролю водогону керує нагрівом та температурою води, обладнана давачами протікання води та контролю каналізації, протипожежна система та система газопостачання обладнані давачами газу, диму та сигналізацією, охоронна система включає пристрої ідентифікації власника та відеоспостереження [68, 70], а система керування зв’язком контролює телефон, Інтернет та мультимедійні пристрої. Усі системи можуть керуватися з одного пульта чи можливе використання голосового керування. На цьому рівні проявляються відмінності в специфіці роботи для споруд різного призначення: адміністративних споруд (захист інформації та конференц-зв’язок), промислових споруд (автоматизації технологічних процесів) та житла підвищеного комфорту (регулювання температури води, поливу газонів, прибирання роботами-пилососами) [71].

Третій рівень – рівень “внутрішньої інтеграції” – вся споруда є однією інтегрованою системою. На цьому рівні відбувається оптимізація різних показників, взаємодія всіх підсистем та автоматичне керування (централізоване чи розподілене). Тут присутні всі системи, що й на другому рівні, але всі вони інтегровані в одну складну систему “інтелектуального будинку” [68, 72].

В залежності від типу, система “інтелектуального будинку” може бути відкрита чи закрита. Закрита – коли всі складові системи, всі пристрої виготовлені одним виробником. Її перевага – менша ймовірність втручання у систему, але це одночасно і недолік – прив’язування до певного виробника і висока монопольна ціна на пристрої. Майбутнє – за відкритими системами OSI/ISO (Open System Interconnection/ International Standard Organisation), в яких пристрої різних виробників будуть уніфіковані та взаємозамінні [68].

В залежності від способів керування – системи можуть бути централізовані (всі підсистеми керуються одним процесором) чи розподілені (в процесі керування беруть участь багато рівноцінних процесорів). Основний недолік централізованої система керування в тому, що на один процесор припадає все навантаження і можливі поломки, від такого інтенсивного використання, виводять з ладу всю систему. При розподіленому керуванні вихід з ладу процесорів менш ймовірний, так як на кожного з них припадає менше навантаження, що також підвищує швидкодію процесорів. Вихід з ладу одного процесора – не приводить до зупинки роботи всієї системи. Відповідно маємо надійнішу систему [68].

Четвертий рівень – рівень “зовнішньої інтеграції”, при якому система “інтелектуальної будівлі” взаємодіє з зовнішніми системами та мережами: комунальними службами, службами порятунку та охорони, провайдерами зв’язку і кабельного телебачення за допомогою засобів взаємодії користувача з інформаційною системою [68]. На четвертому рівні власник “інтелектуального будинку” має можливість керувати системами газо-, водо-, тепло- та енергопостачання як з будинку, так і дистанційно, навіть перебуваючи на великих відстанях від оселі. Сучасні телекомунікаційні можливості з’єднують системи

“інтелектуального будинку” з службами міста та у випадку аварій відправляють сигнал до служб порятунку тощо.

При такій інтеграції в загальні системи і мережі актуальним питанням є технічний захист інформації всіх споруд [68]. Цей рівень ще не до кінця реалізований, оскільки зовнішні системи не готові самостійно “спілкуватись” з “інтелектуальним будинком”.

П’ятий рівень – рівень “адаптації” на сьогоднішній день не реалізований і є перспективою найближчого майбутнього. За задумом розробників, на цьому рівні система адаптується до різних факторів, як то погодні умови, пора року, новий власник та інше [68].

Класифікація будівель за рівнями інтелектуалізації у відповідності до оснащення, призначення та можливостей їх систем, відображає всі існуючі системи та функції (таблиця 1.3).

Узагальнюючи, можна стверджувати, що системи “інтелектуального будинку” розвиваються від простих до складних, від окремих до інтегрованих, від централізованих до розподілених, від закритих до відкритих, від стаціонарних до адаптованих. Відповідно, інтелектуалізація функцій таких систем потребує вирішення ряду задач на кожному з ієрархічних рівнів автоматизованого проектування, що дає змогу стверджувати про необхідність введення інтелектуального аспекту в процес розроблення систем “інтелектуального будинку”.

Необхідно звернути увагу на розроблення мереж “інтелектуальних будинків”, квартир, кварталів, вулиць, міст та інтеграцію “інтелектуальних будинків” в систему “розумного міста” та керування такими системами в межах країни. На превеликий жаль такі задачі практично не ставляться і, відповідно, не розв’язуються, хоча є надзвичайно актуальними для України та інших держав з огляду на зростання вартості енергоносіїв, розвитку інформаційних технологій тощо.

Таблиця 1.3. Функції “інтелектуального будинку” відповідно до рівнів інтелектуалізації

Назва рівня	Функції
Рівень “адаптації”	- адаптація до внутрішнього та зовнішнього середовищ, потреб власника
Рівень “зовнішньої інтеграції”	- спілкування з муніципальними службами газо-, водо-, тепло- та енергопостачання - зв’язок з провайдерами - виклик аварійних служб та охорони - зовнішнє та внутрішнє керування
Рівень “внутрішньої інтеграції”	- інтеграція всіх систем і мереж в єдину систему - програмування підсистем - залежність роботи від присутності людини - віддалене керування через Інтернет - захист від зовнішніх інформаційних атак
Рівень “централізованого керування”	- контроль кліматичних параметрів - постійне якісне живлення - аварійне та чергове освітлення - контроль протікання води та газу - центральний пилосос - аутентифікація - відеоспостереження - охоронна та пожежна сигналізації - імітація присутності людей - внутрішня АТС - Інтернет, IP-телефонія - мультирум, телебачення - локальна мережа - структурована кабельна система - керування всіма системами з пульта
Рівень “активного регулювання”	- керування опаленням - активна вентиляція - керування рівнем освітлення - автовимикання побутових пристроїв - автовідповідач та перемикання розмови на мобільний телефон

1.3. Аналіз моделей штучних нейронних мереж

Реалізація функцій системи “інтелектуального будинку” потребує використання моделей, методів та засобів інтелектуального опрацювання даних та прийняття рішень на їх основі. Більшу частину таких функцій можна реалізувати з використанням продукційних правил [73], семантичних мереж [74], штучних нейронних мереж (ШНМ) [75], мереж Байєса [76] та ін. Особливий значний практичний інтерес, з позиції можливого застосування в підсистемах “інтелектуального будинку», є до штучних нейронних мереж оскільки вони дають змогу програмно та апаратно опрацьовувати нечітку та неструктуровану інформацію від підсистеми давачів, які забезпечують реєстрацію змін в довкіллі “інтелектуального будинку”. Таким чином, використання ШНМ дає змогу реалізувати функції системи “інтелектуального будинку” з різними значеннями швидкодії, складності та інтелектуалізації.

На сьогодні відомо багато різновидів штучних нейронних мереж [77] та їх модифікацій. Це з кожним роком призводить до зростання їхньої кількості, оскільки все більший спектр практичних задач можна розв’язати з використанням даного апарату, а саме: розпізнавання мови [78, 79], реалізації інтелектуальних агентів для збору інформації, пошуку вірусів [80, 81, 82, 83], класифікації, контролю, керування та зв’язку [84, 85, 86, 87, 88], в системах технічного зору [89, 90], в економіці [91, 92], аналізу електромеханічних систем [93, 94], в цифровій техніці [95] та інші. Розглянемо основні їх типи з позиції можливості застосування в системах “інтелектуального будинку”.

Найпоширенішими є нейронні мережі, які структуровані за шарами та, в залежності від свого функціонального призначення, можуть містити однотипні або різнотипні нейрони [96].

Один нейрон здатний виконувати найпростіші процедури розпізнавання об’єкта. З'єднання нейронів дають змогу реалізувати шар нейронів, а об’єднання шарів – мережу. Таким чином, нейронні структури з повним з’єднанням можуть бути як одношаровими, так і багатошаровими [96, 97].

В одношаровій структурі з повним з’єднанням усі вхідні сигнали можуть надходити на всі нейрони. Прикладом такого типу є перцептрон Розенблата – перша штучна нейронна мережа за сучасною термінологією. Загалом – це найпростіша нейронна структура, яка, відповідно, здатна розв’язувати лише найпростіші задачі, пов’язані з опрацюванням даних від підсистеми давачів у “інтелектуальному будинку”. До переваг таких нейронних структур можна віднести їх програмну та апаратну простоту і високу швидкість роботи алгоритму навчання [98], але реалізувати на їх основі складні інтелектуальні функції дуже важко.

Набагато більші функціональні можливості мають багатошарові штучні нейронні мережі. В більшості випадків використовуються багатошарові перцепрони [99]. Особливістю таких нейронних структур є наявність з’єднань між шарами, які йдуть від перших до наступних шарів. Відповідно, така ШНМ включає вхідний шар, на який безпосередньо подаються вхідні сигнали, прихований та вихідний шари. Кількість нейронів у шарах може бути різною, хоча існує ряд підходів для оцінки кількості шарів і нейронів у шарі штучної нейронної мережі [100]. Такі ШНМ типу багатошарового перцептрона [101] є простими при апаратній реалізації і можуть бути використані в процесі автоматизованого проектування підсистем “інтелектуального будинку”. Разом з тим, необхідно звернути увагу на недолік цих структур, а саме: потреба в спеціальних та повільних алгоритмах їх навчання.

Програмний варіант реалізації ШНМ типу багатошарового перцептрона в процесі проектування “інтелектуального будинку”, є перспективним методом інтелектуалізації функцій таких систем.

Особливої уваги заслуговує мережа Кохонена [100], яка має підвищену надійність, що забезпечує її роботу у випадку відмови одного з нейронів. Це є корисною властивістю в процесі автоматизованого проектування “інтелектуальних будинків” особливо при апаратній реалізації такої ШНМ для підвищення швидкодії підсистеми та її надійності.

В процесі реалізації системи “інтелектуального будинку” досить часто виникає проблема опрацювання зображень. В такій ситуації добре себе зарекомендувала мережа Хопфілда [102, 103], відповідно, для розв’язання саме таких чи подібних задач її варто використовувати в підсистемах “інтелектуального будинку”. Разом з тим, мережа Хопфілда має високу швидкодію та просту апаратну реалізацію.

В процесі реалізації функцій з використанням ШНМ виникає задача оцінки похибки прогнозованих результатів, забезпечення високої точності та вимога до опрацювання великих масивів даних. Для таких ситуацій підходить нейронна мережа на основі “функціоналу на множині табличних функцій” [104].

Окрім того, варто звернути увагу та можливість практичного використання в процесі реалізації систем “інтелектуального будинку” нейроподібних структур машини геометричних перетворень [105].

Особливістю таких нейроподібних структур є: швидке неітеративне навчання з наперед визначеною кількістю етапів, можливість розв’язання задач великої розмірності та використання єдиної методологічної бази в процесі синтезу нейронних мереж даного типу для розв’язання задач з різних областей науки та техніки.

Існує і багато інших видів штучних нейронних мереж: нейромережа зворотнього поширення похибки [106], Delta Bar Delta [107], мережа скерованого випадкового пошуку [108], мережа з квантуванням навчального вектора [109], мережа зустрічного поширення [109], двоскерована асоціативна пам'ять [110], мережа адаптивної резонансної теорії [110], які володіють певними перевагами та недоліками. Відповідно можливе їх використання в процесі реалізації специфічних функцій “інтелектуального будинку”.

З наведеного можна зробити висновок, що найчастіше для реалізації функцій “інтелектуального будинку” варто використовувати ШНМ типу багатошарового перцептрона, а для випадку складних варіантів функцій – машину геометричних перетворень.

1.4. Аналіз існуючих систем “інтелектуального будинку”

Проблеми енергозбереження та раціонального використання ресурсів, цікавили споживачів з моменту виникнення електричних побутових приладів. Саме тому, в 1990 році найбільші виробники електротехнічних пристроїв об’єднались для створення надійних і безпечних алгоритмів керування інженерними системами будинку – асоціація EIB (з англ. European Installation Bus – “Європейська інсталяційна шина”). Асоціація EIB – це виробники та інсталятори EIB-систем та технологій “інтелектуального будинку”.

На вітчизняному ринку працює досить багато великих всеукраїнських компаній та невеликих регіональних, що надають послуги встановлення систем “інтелектуального будинку”. Серед них є представництва відомих світових лідерів.

ABB [111] – пропонують електротехнічні пристрої, давачі, регулятори, системи дистанційного управління та унікальну комплексну систему “інтелектуальних будівель” [112]. Дана система оснащена сенсорною панеллю Busch-ComfortPanel, що дає можливість “вести діалог” з будинком, передає інформацію, звук, візуальну інформацію та управляється за допомогою смартфона, а також багатофункціональним сенсорним давачем Busch-priOn, що призначений для управління температурним режимом, регулювання освітлення в кімнатах, керування давачами руху для контролю та безпеки.

Система Instabus від Siemens [113] – інноваційна система для управління такими елементами “інтелектуального будинку” як: підсистема безпеки (пожежна та охоронна), комфорт (регулювання освітлення, опалення та кондиціонування) і робота побутових електроприладів.

Inteldome – перша в Україні компанія, член всесвітньої асоціації по автоматизації будівель LonMark® International, яка має статус авторизованого системного інтегратора американської корпорації Echelon – пропонує послуги енергозбереження, повний комплекс робіт з інженерними системами (енергозабезпечення, опалення, кондиціонування, протипожежною та охоронною) [114].

INTELCITY – українська компанія, що пропонує системи “інтелектуального будинку” Berker instabus KNX/EIB і комплексну систему Myhome Legrand для управління освітленням, мікрокліматом у домі, охоронними системами та медіацентром [115].

Hi-tech-house – компанія, яка є ексклюзивним дистриб’ютором бельгійської компанії DOMINTELL і пропонує проектні роботи, монтаж та сервіс системи “інтелектуального будинку” цього європейського виробника [116].

Всі сучасні розробки систем “інтелектуального будинку” використовують системи, що можна охарактеризувати як: системи з централізованою організацією (AMX, Crestron); децентралізовані (шинні) системи (EIB, LonWorks, C-Bus, BACnet); ті, які використовують радіоканал (GIRA, LEGRAND, BTicino).

Корпорація AMX (Даллас, США) – розробник систем інтегрованого управління мультимедійних систем, опалення, кондиціонування та освітлення з можливістю інтеграції з системами забезпечення життєдіяльності, представлення інформації, безпеки і охорони інших виробників [117].

Компания Crestron Electronics – повний комплекс обладнання, призначеного для автоматизації комерційних і приватних об’єктів, що складається з блоків: управління інженерними системами; управління мультимедіа; управління безпекою та органів управління [118].

Децентралізовані системи EIB оснащені двожильним кабелем – шина EIB, що з’єднує в будівлі всі електричні пристрої. Такий підхід має наступні переваги: набагато спрощує всі кабельні системи, знижує витрати на проектування та прокладку кабелів, зменшує час для монтажу. Такі кабельні системи здійснюють тільки підведення живлення безпосередньо до пристроїв з метою економії витрат електроенергії і зниження ризику виникнення пожеж. Система використовується для управління підсистемами освітлення, регулювання мікроклімату, керування шторами/ жалюзями/ ролетами та контролю доступу й оповіщення [119].

Технологія LonWorks – розроблена корпорацією “Echelon”. Платформа LonWorks стала стандартом для відкритих систем. Пристрої LonWorks, мають інтелектуальний Neuron Chip, що підтримує протокол LonTalk, програму

управління пристроєм та унікальний адрес для ідентифікації і можуть обмінюватися інформацією з іншими пристроями LonWorks, які об'єднані в мережу. Засоби LonWorks використовуються для автоматизації процесів та функцій систем "інтелектуального будинку" (підсистем опалення, вентиляції та кондиціювання повітря) [120].

Протокол C-Bus від австралійської компанії Clipsal – є одним з стандартів автоматизації систем освітлення "інтелектуального будинку", що ґрунтується на основі вже запрограмованих контролерів, які просто і швидко налаштовуються. В C-Bus – пристрій можна підключити в будь якому місці, в режимі Plug-n-Play. У цьому випадку він відразу одержує доступ до функцій системи, а система автоматично підлаштовується під новий прилад [121].

BACnet – це протокол обміну даними для систем автоматизації і управління будівлями – набір правил, обміну даними через комп'ютерні мережі. BACnet може об'єднувати в одному проекті програмне забезпечення і обладнання різних виробників та використовуватися в системах безпеки, охоронно-пожежному комплексі, системах освітлення та при автоматизації інших інженерних систем будівлі [122].

Радіошинна система Gira – спеціально розроблена для вже побудованих будівель. Вона використовується для додаткового оснащення сучасними пристроями та їх дистанційного керування за допомогою радіосигналу [123].

Система My Home Legrand – розроблена на основі технологій: шинна технологія SCS і радіотехнологія ZigBee – для управління: освітленням, мікрокліматом, мультирумом, домофоном; запобігання: затопленню, витоку газу; реалізації: антиобледеніння, відеоспостереження та автоматичного поливу. Використання таких технологій дає можливість суттєво економити енергоресурси [124].

Отже, з проведеного аналізу можна зробити висновок, що в більшості випадків реалізують такі підсистеми "інтелектуального будинку": клімат-контроль, освітлення, захисту та запобігання технічних аварій.

1.5. Аналіз методів та моделей автоматизації проектування систем “інтелектуального будинку”

Проведений вище аналіз показує, що системи “інтелектуального будинку” особливо першого, другого та третього рівнів, в більшості випадків, включають вже існуючі підсистеми керування пристроями різного функціонального призначення. Оскільки існує багато сучасних розробок в сфері керування та інтелектуалізації побутових пристроїв, керування системами життєзабезпечення, робототехнічними системами, то в процесі автоматизованого проектування систем “інтелектуального будинку” використовується класичний підхід до автоматизації розроблення складних об’єктів та систем, а саме блочно-ієрархічний підхід [125].

Блочно-ієрархічний підхід передбачає різні види проектування: проектування зверху вниз [126], знизу вгору, паралельне [127], наскрізне [128] та ін. Найбільш широко використовується вид проектування знизу вгору, що вимагає наявності вже розроблених складових нижчих рівнів проектування. Разом з тим, особлива увага приділена підсистемі інтеграції складових системи “інтелектуального будинку”, узгодженню протоколів обміну інформацією тощо. В ряді літературних джерел використано відповідні рішення [129, 130, 131, 132, 133, 134].

Унікальність системи “інтелектуального будинку”, складність технічних задач, які необхідно розв’язувати в процесі їх автоматизованого проектування обумовлюють використання специфічних підходів та методів до розроблення таких систем. Зокрема в ряді літературних джерел згадується про застосування теорії онтологій до розроблення окремих складових системи “інтелектуального будинку” [135], але використання онтологій до автоматизованого проектування усієї системи “інтелектуального будинку” на даному етапі носить обмежений характер.

Більшого поширення та значення набувають методи, моделі та алгоритми на основі онтологій при розробленні систем “інтелектуального будинку” в процесі автоматизації проектування 3-го, 4-го та 5-го рівнів. Оскільки системи

“інтелектуального будинку” потребують нових підходів, технічних рішень та вищого рівня інтелектуалізації функцій, тому, на сьогодні, використання моделей на основі онтологій для автоматизації розроблення систем “інтелектуального будинку” не є масовим.

Більш поширений, на даному етапі розвитку, підхід, який ґрунтується на нарощенні, що обумовлено необхідними функціями та вартістю системи.

Згідно з блочно-ієрархічним підходом до проектування системи “інтелектуального будинку” на системному рівні автоматизованого проектування, який характеризується низьким рівнем автоматизації, в ряді літературних джерел згадано про використання теорії мереж Петрі для аналізування проектованої системи [136], але “ручна” побудова такої моделі потребує значних зусиль верифікації розробленої моделі і, відповідно, необхідності в дисертаційній роботі розроблення методу та засобів автоматизації синтезу моделей на основі мереж Петрі. Разом з тим для розроблення на системному рівні можна скористатися класичними методами аналізу структури системи та визначення її основних характеристик, таких як теорії систем масового обслуговування, теорії множин та інші.

Схемотехнічний рівень автоматизованого проектування систем “інтелектуального будинку” [137, 138, 139] включає класичні методи, моделі та програмні системи (SPICE, Multisim, Proteus).

Враховуючи гетерогенність системи “інтелектуального будинку” в процесі побудови моделей, що використовують складові, які функціонують за різними фізичними принципами та законами, відповідно, можна використати мову VHDL-AMS [140] та відповідні програмні системи роботи з такими моделями [141].

Компонентний рівень розроблення “інтелектуального будинку” передбачає використання існуючих засобів, методів і моделей [142, 143, 144, 145].

Поняття методу математичного моделювання включає набір моделей та алгоритмів, які використовуються для розв’язання задачі, що вирішується в процесі проектування складних систем.

Одним з таких методів є використання теорії множин, що базується на мові діаграм Джона Венна – англійського логіка та філософа. Ця теорія містить потужний математичний апарат, який визначає поняття порівняння множин, підмножин і надмножин, а також операцій над множинами: об'єднання, перетин, різниця, симетрична різниця, доповнення, декартовий добуток. Також тут введені такі поняття як потужність множини та відношення множин [146].

Теорія множин займає вагоме місце для проектування програмних систем, наприклад, для створення структур даних, структур систем, ділення систем на компоненти та модулі. Математичний апарат теорії множин дає змогу здійснити формальне визначення та виведення даних понять [146].

Таким чином основними перевагами використання теорії множин є наочна, проста для розуміння математична абстракція, а також чіткий синтаксис і семантика.

Проте разом з тим цей метод характеризується такими недоліками як: складність представлення взаємозв'язків, необхідність використання теорії множин у сукупності з іншими методами для моделювання структур систем [147].

Іншим методом, що може бути застосований при вирішенні задач моделювання складних систем є теорія графів. Основні поняття теорії графів були описані задовго до появи теорії множин, як окремої наукової дисципліни, проте формальне визначення графа зручно представляти в теоретико-множинних термінах. Таким чином, граф – це сукупність двох множин: множини точок (вершин) і множини ліній (ребер), які їх з'єднують [148].

Розрізняють різні види графів в залежності від області їх використання. Вони відрізняються спрямованістю, обмеженнями на кількість зв'язків, а також додатковими даними про вершини чи ребра (наприклад, наявністю ваг ребер). В процесі проектування складних систем графи використовують для моделювання структури системи, де вершинами позначаються стани системи, а дугами – зв’язки між цими станами, переходи від одного стану до іншого [148].

У теорії графів існують методи аналізу для окремих класів графів. Одним з таких методів, що використовується для побудови системи є вирішення задачі про

найкоротший шлях. Ця задача зводиться до знаходження у зваженому графі мінімальних можливих шляхів від однієї вершини до інших [148]. Існує ряд алгоритмів для вирішення цієї задачі: хвильовий, Беллмана-Форда, Дейкстри, Флойда та інші. В результаті вирішення цієї задачі проводиться побудова оптимальної структури системи на основі інформації про характеристики її складових частин [147, 148].

Перевагами моделювання на основі теорії графів є: наочне представлення елементів системи та зв'язків між ними, чіткий синтаксис та семантика. Однак при цьому увага акцентується на понятті станів і відсутня можливість відображення динаміки модельованої системи [148].

Ще одним методом математичного моделювання є застосування скінченних автоматів. Скінченний автомат представляє собою особливий вид автомату – абстракції. Він використовується для опису шляху зміни стану об'єкта на основі даних про поточний стан, а також інформації отриманої ззовні. Скінченний автомат представляє собою абстрактний автомат без вихідного потоку, число можливих станів якого скінченне. Результат роботи автомата визначається відповідно до його кінцевого стану [147, 149].

Скінченний автомат може бути заданий у вигляді впорядкованої п'ятірки:

$$M = (V, Q, q_0, F, \delta),$$

де: V – вхідний алфавіт (скінченна множина вхідних символів), з якого формуються вхідні ланцюги, що допускаються автоматом; Q – множина станів; q_0 – початковий стан автомату ($q_0 \in Q$); F – множина кінцевих станів ($F \subset Q$); δ – функція переходів [149].

Автомат починає роботу в стані q_0 та поступово зчитує по одному символу вхідного рядка. Зчитаний символ переводить автомат в новий стан з множини станів Q, відповідно до функції переходів. Процес триває поки не буде досягнуто одного із станів множини F [149].

Поняття скінченних автоматів, як і теорія графів, базується на понятті станів, тому вони також є менш наочним способом представлення динаміки системи, аніж

формалізми, які застосовують функціональний підхід і засновані на понятті процесу.

У зв’язку з цим перевагами методу скінчених автоматів є: наочне представлення елементів системи та зв'язків між ними, чіткий синтаксис і семантика. Однак у них обмежені можливості відображення динаміки системи, акцентування уваги на понятті станів [149].

Мережі Петрі представляють собою окремий напрямок в теорії графів, який пов'язаний з явним включенням семантики в традиційні позначення. Він отримав самостійний розвиток у формі семантичних мереж [150].

Мережі Петрі засновані на класичній теорії графів і є розширенням теорії скінченних автоматів. На основі одного з варіантів мереж Петрі (High-level Petri Nets) в даний час створюється міжнародний стандарт ISO / IEC 15909 [150 - 156].

Мережа Петрі – дводольний орієнтований граф, що містить два типи вершин – позиції та переходи, які з’єднані між собою за допомогою дуг. Дводольність полягає в тому, що вершини одного типу не можуть бути з’єднаними безпосередньо. Кожна позиція може містити в собі маркери (мітки, токени), котрі в процесі моделювання роботи мережі, переміщаються по ній [150].

Маркери можуть бути інтерпретовані як матеріальні об’єкти (ресурси, об’єкти обслуговування), як інформація про активність того чи іншого етапу процесу або як матеріальна чи керуюча інформація.

Функціонування мережі Петрі виконується у вигляді послідовності спрацювання переходів.

Мережі Петрі – складна, добре розвинена теорія зі строго визначеними такими поняттями як стан, умова, перехід тощо. Цей математичний апарат добре досліджений, описано різні види мереж, встановлено їх властивості при різних варіантах проходження обчислень, доведено важливі теореми.

Практичне використання теорії мереж Петрі в основному пов'язане з описом поведінки складних систем, наприклад елементів інтегральних схем. Існує чимало робіт, які використовують мережі Петрі для опису і дослідження алгоритмів

роботи програмних систем і, на погляд багатьох науковців, даний підхід пропонує зручну формалізацію системи з метою її подальшого моделювання [151].

Мережі Петрі отримали широке застосування у багатьох сферах: від проектування мережевих протоколів до розробки логіки роботи домашніх кінотеатрів. Це стало можливм завдяки їх інтенсивному розвитку, модифікацій та різновидів, основними з яких є: ординарні, регулярні, ієрархічні, мережі подій, предикатні, кольорові, часові та подієві мережі Петрі [152].

Ординарні мережі – мережі, які мають тільки одне обмеження: кратність дуг повинна бути не більшою від одиниці. Неординарна мережа може бути перетворена в ординарну. Для цього знаходять максимальну кратність дуг кожного місця, а потім розмножують позиції відповідно до кратності. Ці позиції з'єднуються одна з одною в кільце, при цьому дуги прорізаються своїми переходами. Напрямок дуг є односпрямованим, таким чином, щоб утворився цикл. Далі відновлюють зв'язки даної розмноженої позиції з усіма переходами. Алгоритм проведення зв'язків жорстко не встановлений, але зв'язки проводяться так, щоб вони залишалися ординарними [152].

Регулярні мережі. Основною властивістю регулярних мереж є наявність певної алгебри, яка забезпечує проведення аналітичного представлення процесу топологічної побудови та розчленовування процесу аналізу мереж на сукупність етапів, на кожному з яких досить мати справу з більш простими фрагментами мережі [152].

У свою чергу узагальненням регулярних мереж є ієрархічні мережі. Вони призначені для адекватного моделювання ієрархічних динамічних систем. Така мережа функціонує, переходячи від маркування до маркування, з деякими відмінностями від роботи регулярних мереж, пов'язаними з присутністю складених переходів, спрацювання яких є не миттєвою, як у звичайних мережах, а тривалою дією. Тому складений перехід не спрацьовує, а працює, тобто знаходиться деякий час в активному стані [151, 152].

При моделюванні систем за допомогою мереж Петрі, часто виникають ситуації, при яких маркери мають відповідати об'єктам, переданим від компоненти

до компоненти (від переходу до переходу). Причому ці об'єкти мають додаткові атрибути, які дозволяють розрізняти їх і використовувати ці відмінності для управління функціонуванням системи. Для опису подібних ситуацій був розроблений підклас кольорових мереж Петрі [152, 153]. В згаданих вище видах всі маркери передбачалися абсолютно однаковими. Поява кольорових мереж Петрі пов'язана з концепцією використання розрізнюваних міток. Кольорові мережі – це мережі, в яких кожний маркер має свій певний колір і перехід, котрий пов'язаний з деякою умовою, що визначає наявність пов'язаних з ним вхідних позицій міток певного кольору. Колір маркера позначають буквою. З кожним переходом зв'язується таблиця правил його спрацьовування. У таблиці переходів стовпці ліворуч від роздільної лінії зв'язуються з вхідними місцями переходів і в сукупності містять поєднання конкретних фішок, при яких перехід може спрацювати. Стовпці, які стоять праворуч, вказують на ознаки або на вигляд фішок, які будуть передані вихідним місцям. Умови збудження і правила спрацьовування переходів для міток кожного кольору задаються незалежно. У даних мережах маркерам приписуються деякі ознаки, наприклад різні кольори (змінні), а кратності дуг інтерпретуються як функції від цих змінних.

В якості розширення кольорових мереж з'явилися предикатні мережі. Даний вид мереж Петрі дозволяє пов'язувати з переходами логічні формули (предикати або захист), що представляють класи можливих маркувань у вхідних і вихідних позиціях відповідно до міток дуг. Ці вирази задають умови відбору необхідних кольорів для спрацьовування переходів [154].

Часові мережі Петрі – мережі, в яких з кожним переходом пов'язують деяку тривалість (час). Для визначеності вважають, що вилучення фішок з вхідних позицій відбувається миттєво, а передача фішок здійснюється за встановлений час. У реальності це може відповідати роботі технічних пристроїв і підрозділів організацій [155].

Потокові мережі – мережі, що моделюють потокові системи, в яких здійснюється управління даними. Операції виконуються негайно за умови готовності даних. У потокової мережі Петрі переходи інтерпретуються як

оператори або обчислювальні функції, місця інтерпретуються як черги, а дані – як маркери (мітки). Якщо перехід має n входів, то він реалізується n-місною функцією, яка спрацьовує відразу ж при наявності фішок у всіх вхідних місцях. Дані не є адресованими, іншими словами вони містяться не в центральній, а в розподіленій пам'яті [156].

В табл. 1.4 представлено коротку порівняльну характеристику основних видів мереж Петрі.

Таблиця 1.4. Порівняльна таблиця основних видів мереж Петрі

Вид мережі	Переваги	Недоліки
Ординарні	Переваги у використанні та аналізі	Обмежене коло застосування (ієрархія не передбачена)
Ієрархічні	Дозволяють уникнути громіздких записів мереж	Обмереже коло застосування
Часові	Дозволяють моделювати та аналізувати системи, параметром яких є час	Вузький спектр застосування (ієрархія не передбачена)
Об’єктно-орієнтовані	Зручні при використанні в проектуванні ООП систем	Немає можливості аналізу та моделювання процесів, пов’язаних з часом, вузький спектр застосування (ієрархія не передбачена)
Кольорові	Дозволяють моделювати системи з різними типами об’єктів; компактність представлення роботи системи	Немає чітко вираженого представлення часових характеристик (ієрархія не передбачена)

Оскільки в даній роботі, при даній прикладній області ми маємо справу з розрізненою різнотипною інформацією (різні типи приміщень, відмінні алгоритми роботи систем при активації давачів різними користувачами), ми використовуємо саме кольорові мережі Петрі. Їх застосування дає змогу реалізувати розподіл процесів контролю в різних підсистемах окремо, провести аналіз динаміки роботи моделей цих підсистем, їх взаємодію між собою та роботу системи загалом з урахуванням всіх типів сигналів та умов спрацювання давачів.

Для аналізу мереж Петрі використовується побудова дерева досяжності. Дерево досяжності – орієнтоване кореневе дерево, вершини котрого – можливі маркування, а дуги – спрацьовуючі переходи [150].

Початковому маркуванню відповідає коренева вершина дерева, а дуги, відмічені переходами *Tj*, з'єднують вершини маркування, що являються безпосередньо доступними при спрацюванні *Tj*. Дерево досяжності в загальному випадку може бути безкінечним з вершинами таких типів:

– внутрішнє маркування, яке є досяжним з початкового маркування і не являється тупіковим;

– тупікове маркування, у якому не може спрацювати ні один перехід;

– дублююче маркування ($M_д$), яке відповідає вже введеному в дерево маркуванню $M=M_д$ (але якщо на шляху з початкового маркування в $M_д$ зустрічається маркування M, то $M_д$ – маркування-цикл);

– накопичуюче маркування $M_н$, у відповідності з яким на шляху з початкового маркування існує інше маркування M таке, що $M \leq M_н$.

Нескінченність дерева можлива тільки у випадку існування накопичуючих маркувань, що породжують циклічне повторення однакових послідовностей спрацьовуючих переходів [150, 151].

Щоб побудувати скінченне дерево досяжності і представити процес нескінченного накопичення маркерів у позиціях мережі, вводиться позначення у вигляді символу *w*, що володіє наступними властивостями:

$$w+a=w,\ w-a=w,\ a<w.$$

Алгоритм побудови скінченного дерева досяжності базується на наступних положеннях:

1) алгоритм послідовно обробляє вершини, перетворюючи кожну кінцеву (неопрацьовану) в одну з типових – тупікову чи дублюючу;

2) якщо поточне маркування *M* не кваліфікується як одне з двох наведених, то *M* стає внутрішнім, для якого формується підмножина безпосередньо досяжних маркувань, що у дереві стають кінцевими вершинами. Нові кінцеві маркування *M*

визначаються за результатами спрацьовування збуджених у M переходів за наступними правилами:

– якщо $M(Pi)=w$, то $M_1(Pi)=w$;

– якщо на шляху з початкового маркування в М’ існує таке маркування $M’’$, що $M’’<=M’$ і $M’’(Pi)<M’(Pi)$, то $M’(Pi)=w$;

– в іншому випадку $M’(P_i)$ зберігає своє значення, отримане результаті спрацьовування переходу.

3) коли усі вершини будуть оброблені, алгоритм зупиняється. Побудова кінцевого дерева досяжності дозволяє практично використовувати його для дослідження властивостей мережі Петрі [150, 151].

Таким чином, при виборі методу моделювання програмних систем, однозначне рішення приймається на користь кольорових мереж Петрі. Така позиція обгрунтовується численними перевагами цього математичного апарату над іншими методами. Серед основних плюсів можна назвати наступні: високий рівень формалізації дискретно-подієвих систем; можливість моделювання перехідних процесів будь-яких типів з урахуванням можливих конфліктів; зрозумілість моделі, її наглядність, легкість дослідження та компактність; можливість опису моделі на різних рівнях абстракції; можливість аналітичного дослідження властивостей моделі; масштабованість моделі; можливість швидкого конструювання алгоритму імітації системи з великою кількістю подій.

1.6. Висновки до розділу 1

1. Проаналізовано особливості систем “інтелектуального будинку”, розглянуто їхні переваги та недоліки. Аналіз рівнів інтелектуалізації дало підставу стверджувати, що необхідно розробляти нові моделі для опрацювання нечітких та неструктурованих даних від підсистеми давачів, а також розробляти системи “інтелектуального будинку” з нижчою вартістю для збільшення ефективності їх використання.

2. Аналіз існуючих підходів до автоматизованого проектування утверджує необхідність розроблення моделей для системного рівня автоматизованого проектування систем “інтелектуального будинку” та методу автоматизованого синтезування таких моделей.

3. Проведено аналіз існуючих штучних нейронних мереж з позиції можливості їх застосування для реалізації функцій систем “інтелектуального будинку”, на основі якого запропоновано використати штучні нейронні мережі типу багатошарового перцептрона та машини геометричних перетворень.

4. Аналіз рівнів інтелектуалізації систем “інтелектуального будинку” вимагає встановлення задач інтелектуального спрямування на усіх ієрархічних рівнях блочно-ієрархічного підходу.

РОЗДІЛ 2. МЕТОД ТА МОДЕЛІ ДЛЯ АВТОМАТИЗАЦІЇ СИСТЕМНОГО РІВНЯ ПРОЕКТУВАННЯ СИСТЕМ “ІНТЕЛЕКТУАЛЬНОГО БУДИНКУ”

Розвиток та впровадження систем “інтелектуального будинку” [32, 157, 158, 159, 160] дає змогу суттєво економити енергоносії та забезпечити комфортні умови проживання. На даний час, існує ряд фірм, що пропонують складові для систем “інтелектуального будинку” [98, 161, 162, 163, 164, 165], які можна об’єднати в єдину систему.

Жорсткі вимоги до термінів та якості розроблення складних об’єктів та систем обумовлюють широкомасштабне використання засобів автоматизації на усіх рівнях автоматизованого проектування та моделювання. На сьогодні, в більшості випадків в процесі проектування різних технічних пристроїв, використовують блочно-ієрархічний підхід. Блочно-ієрархічний підхід, в процесі розроблення складних систем, передбачає розбиття процесу проектування на ряд ієрархічних рівнів; в загальному випадку ми маємо метарівень (системний), макрорівень (схемотехнічний) та мікрорівень (компонентний). Кількість ієрархічних рівнів залежить від предметної області.

Отже, загалом в процесі автоматизованого проектування системи “інтелектуального будинку” з використанням блочно-ієрархічного підходу можна виокремити системний рівень ІБ, рівень підсистем та елементи підсистем.

В загальному випадку, застосування блочно-ієрархічного підходу до розроблення ІБ можна представити в математичній формі виразом:

$$Sys^{1}_{\text{ІБ}} = \bigcup_{i=1}^{n} P^{2,\text{i}}_{\text{ІБ}} \bigcup_{j=1}^{m} B^{3,j}_{\text{ІБ}} \bigcup_{k=1}^{l} E^{4,k}_{\text{ІБ}} , \qquad (2.1)$$

де $Sys^{1}_{\text{ІБ}}$ – система “інтелектуального будинку”, а верхній індекс означає перший ієрархічний рівень блочно-ієрархічного підходу до проектування;

$P^{2,\text{i}}_{\text{ІБ}}$ $(i = 1,2,...,n)$ – i-та підсистема “інтелектуального будинку”, а верхній індекс другий ієрархічний рівень блочно-ієрархічного підходу (для прикладу: підсистема клімат-контролю, підсистема освітлення та інше);

$B_{\text{ІБ}}^{3,j}$ $(j=1,2,...,m)$ – j-й блок системи “інтелектуального будинку” (третій ієрархічний рівень блочно-ієрархічного підходу);

$E_{\text{ІБ}}^{4,k}$ $(k=1,2,...,l)$ – k-й елемент системи “інтелектуального будинку” (різного роду давачі, виконуючі пристрої тощо).

Необхідно зауважити, що система ІБ охоплює об’єднання підсистем:

$$Sys_{\text{ІБ}}^{1}=\bigcup_{i=1}^{n}P_{\text{ІБ}}^{2,\text{i}},$$

i-та підсистема базується на об’єднанні завершених блоків:

$$P_{\text{ІБ}}^{2,\text{i}}=\bigcup_{j=1}^{m}B_{\text{ІБ}}^{3,j},$$

j-й блок об’єднує елементи:`

$$B_{\text{ІБ}}^{3,j}=\bigcup_{k=1}^{l}E_{\text{ІБ}}^{4,k}.$$

До рівня підсистеми можна віднести підсистему клімат-контролю, освітлення, захисту та інші. Елементний рівень включає давачі, виконуючі прилади, мікроконтролери тощо.

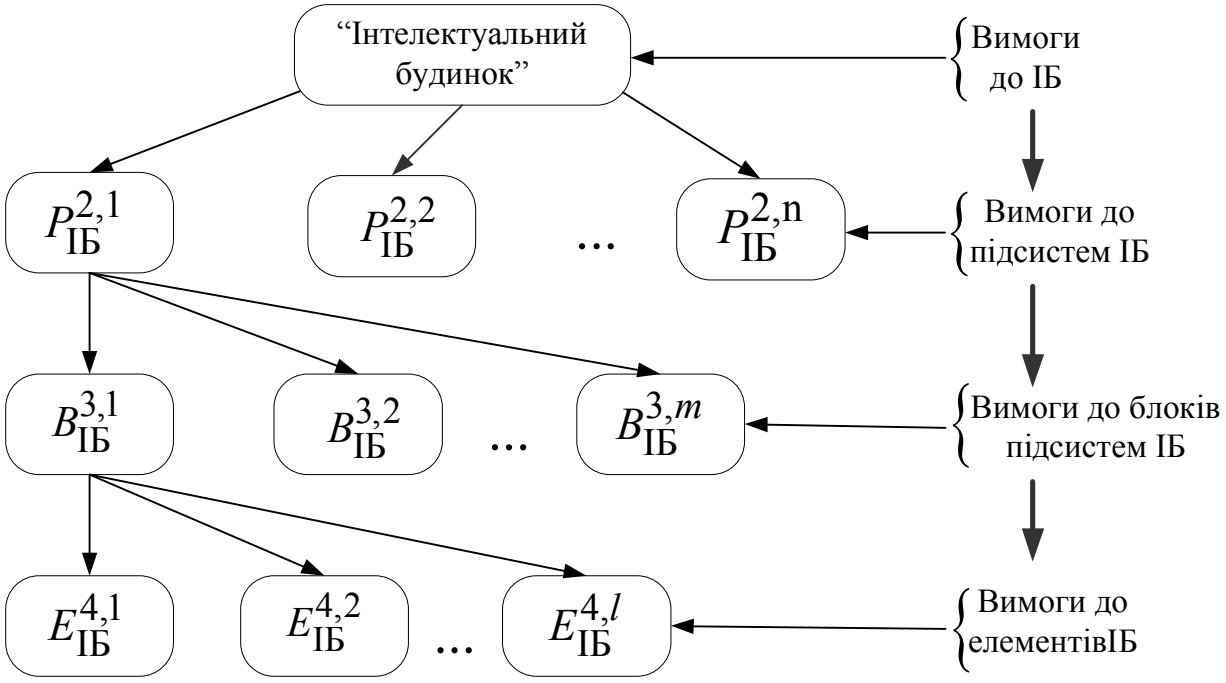


Рис. 2.1. Схема застосування проектування зверху вниз в процесі розроблення систем “інтелектуального будинку”

В процесі автоматизованого проектування систем “інтелектуального будинку” використовують як підхід проектування зверху вниз, так і знизу вгору та інші. Приклади формування вимог при застосуванні вище зазначених підходів, наведено на рис. 2.1. та рис. 2.2. На практиці використовується поєднання підходу проектування зверху вниз та знизу вгору.

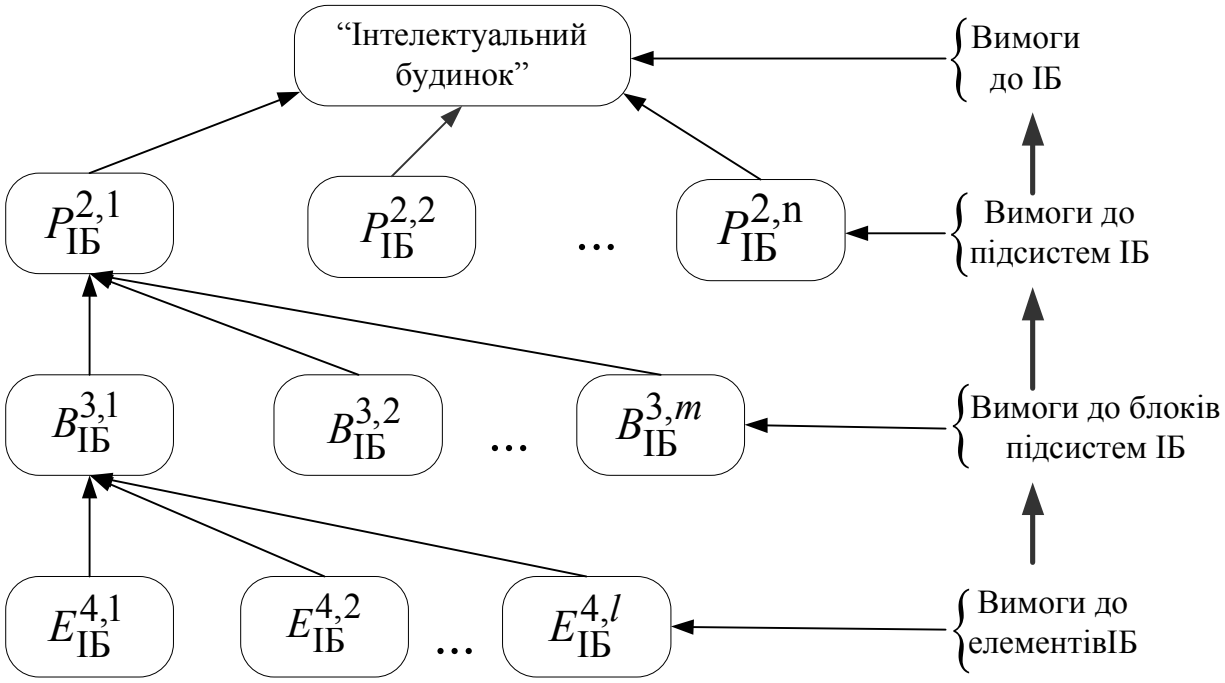


Рис. 2.2. Схема застосування проектування знизу вгору в процесі розроблення систем “інтелектуального будинку”

Кожен з цих підходів до розроблення систем “інтелектуального будинку” має свої особливості, переваги та недоліки. Зокрема, в процесі проектування системи “інтелектуального будинку” 1-го та 2-го рівнів інтелектуалізації використовують, в основному, підхід знизу вгору, що обумовлено наявністю великої кількості реалізованих пристроїв автоматики. Проблема, в даному випадку, виникає при їх інтегруванні в єдину систему, забезпечення єдиного формату обміну та керування, особливо в тих випадках, коли використовуються складові від різних виробників.

Реалізація третього та вищих рівнів систем “інтелектуального будинку” потребує використання нових підходів, методів та технологій. Такі особливості обумовлюють використання проектування зверху вниз.

Крім цього, процес розроблення системи “інтелектуального будинку” вимагає особливої уваги до аспектів проектування. Процес інтелектуалізації функцій розроблювальної системи вимагає введення інтелектуальних рис, їх детального аналізу і забезпечення на усіх ієрархічних рівнях та етапах. Основні задачі інтелектуального наповнення в процесі використання блочно-ієрархічного підходу наведено в табл. 2.1.

Таблиця 2.1. Основні задачі інтелектуального аспекту в процесі розроблення ІБ при використанні блочно-ієрархічного підходу

Назва рівня	Задачі
Системний рівень	Вибір рівня інтелектуалізації системи ІБ. Обґрунтування можливості реалізації інтелектуальних функцій. Вибір стратегії реалізації інтелектуальних функцій ІБ. Вибір методів та засобів реалізації інтелектуальних функцій ІБ. Побудова структурних схем реалізації інтелектуалізації функцій ІБ.
Схемотехнічний рівень	Задачі інтелектуального опрацювання даних в підсистемах ІБ. Вибір необхідних методів, засобів, моделей та методологій. Реалізація інтелектуальних функцій ІБ на рівні підсистем.
Компонентний рівень	Задачі інтелектуального аспекту в процесі проектування систем ІБ. Задачі інтелектуального опрацювання даних від давачів (відновлення втрачених даних тощо). Вибір методів, моделей та алгоритмів опрацювання даних від давачів.

Найскладніші задачі, які потребують величезних обсягів обчислень, виникають на етапі системного проектування складних об’єктів та систем. На цьому рівні виникає потреба в розв’язанні задач синтезу структур об’єкта проектованої системи та визначення її основних параметрів. Наступний крок передбачає, що аналіз розробленої структури можна виконати, використовуючи

моделі на основі теорії мереж Петрі, систем масового обслуговування та інші [3]. Процес побудови таких моделей вимагає великих затрат часу, досвіду, знань з предметної області та є надзвичайно громіздким. Відповідно, в даному розділі розроблено метод синтезу моделей на основі мереж Петрі та побудовано моделі підсистем "інтелектуального будинку" для системного рівня проектування.

2.1. Розроблення методу синтезу моделей на основі мереж Петрі для системного рівня автоматизованого проектування

Розроблений у дисертаційній роботі метод дає змогу на основі структури системи та алгоритму її функціонування, згенерувати в автоматичному режимі модель, яка ґрунтується на теорії мереж Петрі (МП) [11, 12]. Така модель призначена для дослідження побудованої системи на системному рівні автоматизованого проектування. Ідея розробленого методу полягає в тому, що спершу на основі структури будується цілісний орієнтований граф системи. А після цього будується модель на основі мереж Петрі (рис. 2.3).

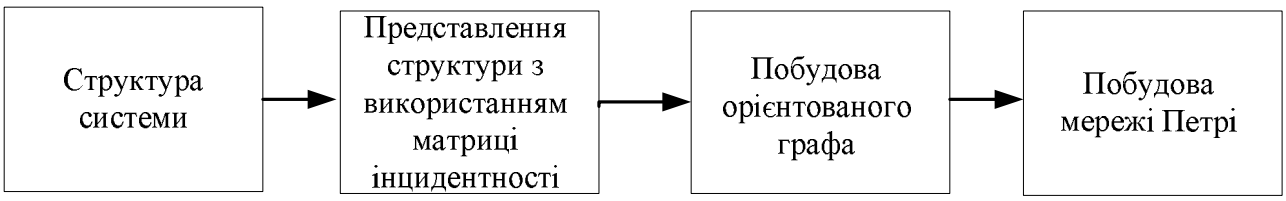


Рис.2.3. Схема реалізації методу синтезу моделей на основі мереж Петрі

2.1.1. Модель структури системи

В загальному випадку, під структурою об'єкта проектування розуміємо набір елементів та зв'язки між ними, що можна описати за допомогою моделі з використанням теорії графів:

$$G=(E,R), \qquad (2.2)$$

де $E=(e_1,e_2,...,e_t)$ – множина елементів структури системи;

t – кількість елементів структури системи;

$R = (r_1, r_2, ..., r_s)$ – множина зв’язків між елементами структури (ребер);

s – кількість зв’язків між елементами структури.

Окрім того, важливим елементом моделі структури, є матриці інцидентності та сумісності, які відображають зв’язки між елементами структури та їх напрям. Тому, в даному випадку, мають справу з орієнтованим графом, а елементи матриці інцидентності можуть приймати значення 0, якщо елементи (вершини графа) не інцидентні; +1, якщо дуга (зв'язок) орієнтована від елемента та -1, якщо дуга орієнтована до елемента. Структуру проектованої системи отримують після розв’язання задачі структурного синтезу.

Зокрема загальнена структура підсистеми клімат-контролю (рис. 2.4) описується графом, який зображено на рис. 2.5, та матрицею інцидентності (2.3):

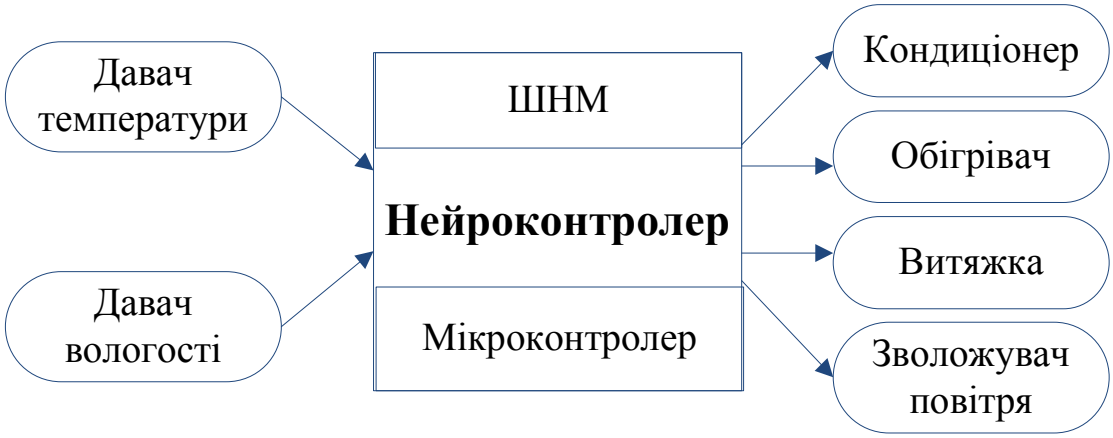


Рис. 2.4. Структура підсистеми клімат-контролю однієї з кімнат

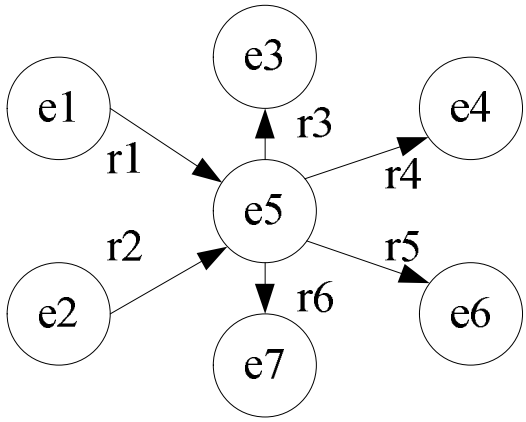


Рис.2.5. Граф структури підсистеми клімат-контролю однієї з кімнат

$$I = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \end{array} \begin{array}{c} \begin{array}{cccccc} r_1 & r_2 & r_3 & r_4 & r_5 & r_6 \end{array} \\ \left\| \begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ -1 & -1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{array} \right\| \end{array} \qquad (2.3)$$

2.1.2. Розроблення алгоритму генерування моделі на основі мереж Петрі

Для генерування МП на основі структури системи, а саме: складових та зв'язків між елементами використовується наступний підхід:

– формалізація структури системи;

– генерування цілісного орієнтованого графа системи;

– розташування елементів (визначення параметрів елементів мережі Петрі).

В процесі реалізації системи використовується об'єктно-орієнтований підхід. Відповідна система складається з об'єктів та дій над об'єктами. Тому для запису структури системи на етапі реалізації необхідно виокремити окремий об'єкт та дії/умови, які він виконує і дії/умови, що необхідні для зміни стану вибраного об'єкта.

Кожен об'єкт складається з:

– назви;

– початкового маркування;

– вихідних дій (дій, які виконує об'єкт);

– вхідних дій (дій, які призводять до зміни стану об'єкту).

Складність реалізації методу полягає в тому, що декілька об'єктів можуть посилатись на одну і ту ж дію/умову (в даному випадку перехід). Це означає, що для її виконання необхідне відповідне маркування усіх об'єктів, які на неї посилаються. Також певна вхідна дія/умова (перехід) може мати посилання на

декілька об’єктів. Тоді при її виконанні реалізується зміна маркування усіх зв’язаних з нею об’єктів.

Відповідно спираючись на структуру, яка описана об’єктами із посиланнями на вхідні та вихідні дії/умови (переходи), необхідно використати швидке та ефективне перетворення даного запису в орієнтований граф [166, 167].

Оскільки для графу немає різниці між об’єктом і дією/умовою/переходом, то кожна вершина повинна містити додаткові маркування, які відповідають типу вершини. Вершини графа з’єднані між собою дугами, які також повинні мати додаткові позначення, містити свою ціну та “колір”.

Для перетворення структури запису об’єкту у граф для побудови МП розроблено наступний алгоритм, який включає такі основні кроки:

Крок 1. Проініціалізувати список вершин та дуг.

Крок 2. Вибрати перший об’єкт.

Крок 3. Додати об’єкт до списку вершин.

Крок 4. Виокремити усі вхідні переходи.

Крок 5. Для кожного із вхідних переходів виконати дії:

Крок 5а. Перевірити чи існує перехід із такою ж унікальною назвою. Якщо відсутній, то створити нову додаткову вершину-перехід.

Крок 5б. Додати дугу від переходу до вершини-об’єкту.

Крок 6. Для кожного із вихідних переходів виконати дію:

Крок 6а. Перевірити чи існує перехід із такою ж унікальною назвою. Якщо відсутній, то створити нову додаткову вершину-перехід.

Крок 6б. Додати дугу від вершини-об’єкту до переходу.

Крок 7. Перевірити чи є наступні об’єкти. Якщо “Так”, то вибрати наступний об’єкт і перейти на крок 3. Якщо “Ні”, то перейти на крок 8.

Крок 8. Вивести результат (структуру орієнтованого графа).

Генерування орієнтованого графа-структури системи необхідне для того, щоб можна було ефективно розташувати елементи системи в єдину, зрозумілу чітко організовану структуру, яку можна перевести у мережу Петрі. Без такого перетворення буде складно пов’язати об’єкти та зв’язки між ними.

Проаналізувавши утворений граф, можна розташувати вершини у логічному порядку.

Розроблений метод дає змогу автоматизувати процес синтезування моделей підсистем “інтелектуального будинку” на основі теорії мереж Петрі для системного рівня автоматизованого проектування.

Побудова структурної моделі на основі мереж Петрі має свої особливості. Важливим аспектом синтезу схемної моделі є розміщення складових. При цьому можливий ряд різних випадків. Перший, і найпростіший, полягає в тому, що розміщення елементів мереж Петрі починається із координат з найменшими значеннями, тобто з лівого нижнього кута області побудови схемної моделі.

Другий варіант синтезування схемної моделі передбачає сортування елементів за критерієм кількості зв’язків і розміщення складових, передовсім, з найбільшою кількістю зв’язків в центрі робочої області побудови схемної моделі.

Третій запропонований до використання варіант розміщення елементів структурної моделі мережі Петрі передбачає використання модифікованого алгоритму Фрюхтермана-Рейнгольда [168, 169, 170]. Цей алгоритм використовується для візуального представлення графів різного виду та складності і ґрунтується на фізичних аналогіях, тобто використанні для опису зв’язків понять сили та енергії. Такі алгоритми отримали ще назву силових.

Для реалізації силового алгоритму необхідно задати модель, яка описує проектовану систему “інтелектуального будинку” та зв’язки між її складовими елементами та алгоритм визначення стану рівноваги для системи.

Запропонований метод дає змогу забезпечити дотримання вимог до зображення мережі Петрі: рівномірний розподіл вершин на площині, мінімальна кількість перетинів ребер та інші .

Алгоритм Фрюхтермана-Рейнгольда для візуалізації графа [168] запропоновано використати у модифікованому вигляді розглянувши усі пари вершин, визначивши правила дії сил між ними та ввести додаткові коефіцієнти балансування. У використаному методі граф розглядається як система об’єктів з’єднаних пружинами за певними правилами. Кожна пружина діє на вузли які вона

з’єднує притягуючи або відштовхуючи їх один від одного. Уся система вузлів переміщується зазнаючи дії сил створених пружиною, а також сили гравітації. Сила гравітації вводиться в алгоритм для того, щоб вершини слабо зв’язані з іншими не віддалялися на значну відстань від центру графа. Рух системи відбувається до тих пір поки усі сили не зрівноважаться, тобто їх сума, для кожного вузла не стане рівною нулю [171,172].

В отриманому зображенні для усунення можливості перетину ребрами вершин, а також зменшення кількості перетин ребер їх доцільно будувати не у вигляді прямих відрізків, а розраховувати їх форму щоб забезпечить можливість огинання вузлів.

2.2. Розроблення структури системи “інтелектуального будинку”

При проектуванні систем “інтелектуального будинку”, як і для більшості складних технічних систем, як було зазначено вище, застосовують блочно-ієрархічний підхід [173]. Для аналізу роботи систем та підсистем “інтелектуального будинку” запропоновано використовувати моделі на основі теорії мереж Петрі [1, 97, 174, 175], що дає змогу інтегрувати складові різного функціонального призначення, дослідити їх сумісну роботу та динаміку.

З метою забезпечення максимальної ефективності та функціональності система “інтелектуального будинку” повинна містити такі основні підсистеми (мінімальний набір підсистем), як: підсистему клімат-контролю, підсистему освітлення, підсистему захисту, підсистему запобігання технічних аварій, а також ряд інших додаткових підсистем.

Для забезпечення ефективного механізму синхронізації роботи основних підсистем та компонентів побудованої системи ІБ між собою та з користувачем, система повинна також охоплювати віддалені засоби керування ІБ, внутрішній та центральний модулі керування та контролери підсистем ІБ. З врахуванням вищеописаних вимог було розроблено структурну схему системи ІБ, яка представлена на рис. 2.6.

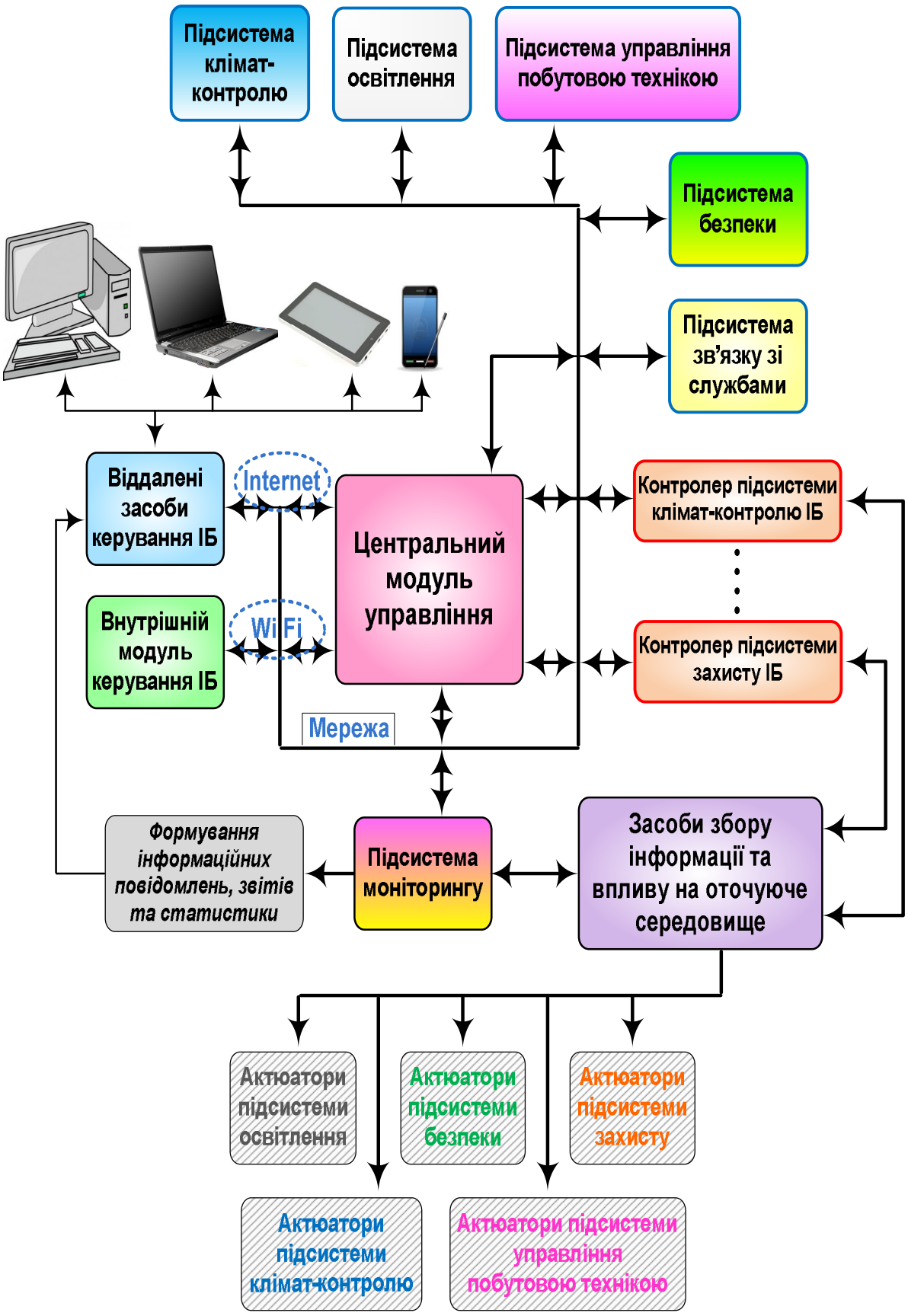


Рис. 2.6. Структурна схема системи "інтелектуального будинку"

Побудована структурна схема системи ІБ містить ряд основних підсистем, а саме – підсистему клімат-контролю, освітлення, керування побутовою технікою, запобігання технічних аварій, захисту та моніторингу. Кожна із підсистем відповідає за миттєве реагування на спрацювання давачів, що свідчить про зміну відповідного вхідного параметра системи ІБ з метою подальшого коректування системи в даній області (областях). Обмін даними між основними функціональними складовими системи ІБ здійснюється через внутрішню мережу.

Система може працювати в трьох режимах: в автоматичному режимі, в режимі користувача та в режимі очікування. При цьому автоматичний режим роботи системи ІБ передбачає реагування на зміну будь-якого вхідного параметра системи та запуск механізму корекції системи з допомогою відповідного контролера в автоматичному режимі, а функція користувача полягає лише в отриманні інформаційних повідомлень про відповідні зміни у системі. Режим користувача передбачає синхронізацію роботи системи ІБ та користувача через центральний модуль керування та віддалені засоби керування ІБ з допомогою Wi-Fi або мережі Internet. При цьому в разі зміни будь-якого вхідного параметра системи відбувається активація відповідної підсистеми, а також підсистеми моніторингу, яка формує інформаційне повідомлення та здійснює запит користувачу на активацію необхідного механізму корекції системи. Таким чином, запуск механізму корекції системи відбувається виключно за згодою користувача.

Режим очікування призначений для тимчасового призупинення (вимкнення) роботи системи ІБ.

Корекція системи ІБ здійснюється з допомогою відповідного контролера або контролерів та активаторів. Кожна підсистема ІБ охоплює ряд індивідуальних вхідних та вихідних параметрів – давачів та активаторів, і призначена для моніторингу та корекції певної області ІБ. Давачі відповідають за збір вхідної інформації про стан системи, в той час як активатори – за реалізацію механізму корекції системи в необхідному напрямку.

2.3. Розроблення алгоритму функціонування та моделі аналізу роботи системи “інтелектуального будинку” на основі кольорової мережі Петрі

Відповідно до розробленої структурної схеми (рис. 2.6), побудовано алгоритм роботи системи ІБ (рис. 2.7). На початку роботи система переходить в автоматичний режим. При появі події (спрацювання одного або кількох вхідних параметрів системи) встановлюється відповідний асоціативний зв’язок належності активованих вхідних параметрів системи до їхніх доменних підсистем і запуск даних підсистем. При обраному автоматичному режимі роботи запускається нейроконтролер відповідної підсистеми [6, 13, 151, 176, 177, 178], що приводить в дію механізм корекції потрібних параметрів системи ІБ.

При обраному режимі користувача система формує відповідний запит, і очікує підтвердження дозволу на здійснення корекції потрібних параметрів ІБ від власника, і у випадку підтвердження дозволу від користувача запускається нейроконтролер.

Після цього, відбувається перехід системи в початковий стан очікування події, або ж перехід в режим очікування і тимчасове призупинення роботи системи. На основі побудованого алгоритму розроблено також модель системи ІБ з використанням кольорових мереж Петрі [5, 175], яка представлена у схемному вигляді на рис. 2.8. Опис та призначення переходів та позицій розробленої моделі представлені в табл. 2.2 і табл. 2.3, відповідно.

Оскільки відповідно до запропонованої структурної схеми системи ІБ (рис. 2.6), в системі передбачено ряд структурно-функціональних підсистем різного функціонального призначення, то з метою унеможливлення виникнення будь-яких конфліктних ситуацій між цими підсистемами в системі передбачена строга ієрархія пріоритетних рівнів основних функціональних підсистем (від 1 до 5, де 1 – рівень максимального пріоритету). Вони представлені нижче в табл. 2.4. Зі збільшенням кількості підсистем, кількість пріоритних рівнів також можна збільшувати. Згідно запропонованої ієрархії пріоритетних рівнів найвищий

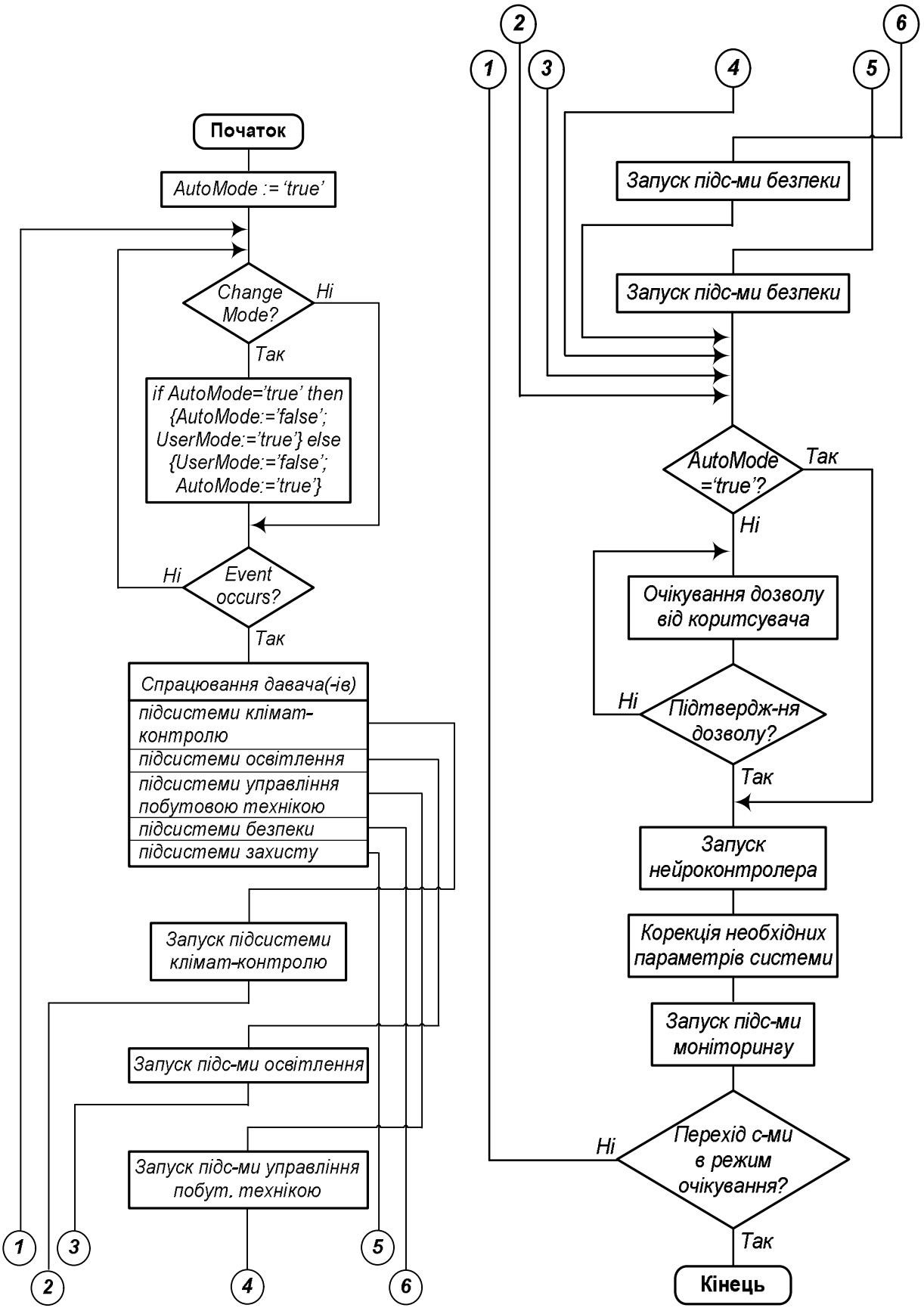


Рис. 2.7. Загальний алгоритм роботи системи ІБ

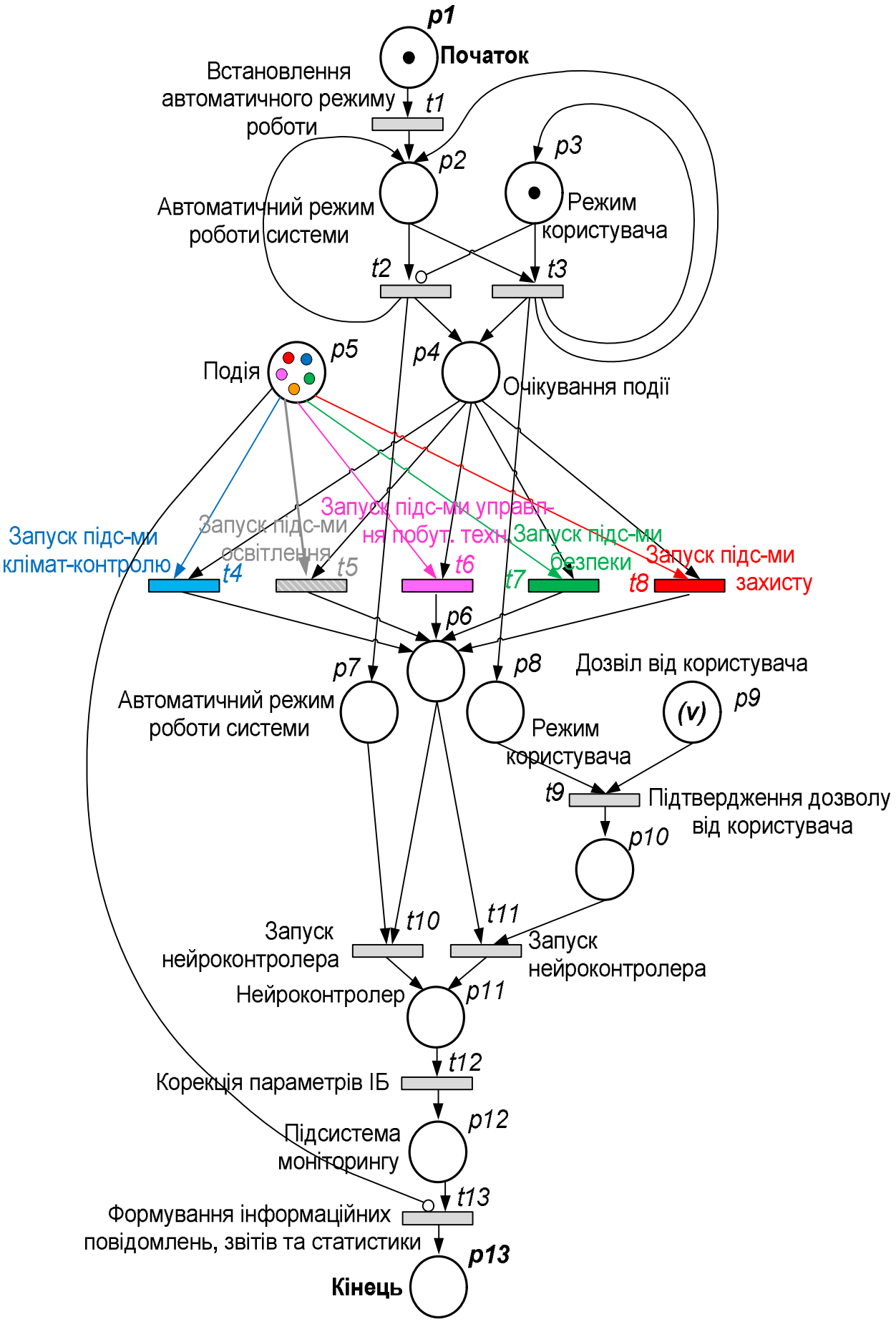


Рис. 2.8. Модель системи ІБ на основі кольорової мережі Петрі

Таблиця 2.2. Переходи моделі системи ІБ на основі кольорової мережі Петрі

№	Призначення
t1	Запуск роботи моделі
t2	Встановлення автоматичного режиму роботи системи ІБ
t3	Встановлення режиму користувача роботи системи ІБ
t4	Запуск підсистеми клімат-контролю системи ІБ
t5	Запуск підсистеми освітлення системи ІБ
t6	Запуск підсистеми управління побутовою технікою системи ІБ
t7	Запуск підсистеми безпеки системи ІБ
t8	Запуск підсистеми захисту системи ІБ
t9	Підтвердження дозволу від користувача на запуск нейрон-контролера з метою здійснення корекції вказаних параметрів ІБ
t10	Запуск нейроконтролера (в автоматичному режимі)
t11	Запуск нейроконтролера (в режимі користувача)
t12	Корекція параметрів ІБ
t13	Формування інформаційних повідомлень, звітів та статистики

рівень належить підсистемі захисту “інтелектуального будинку”, що відповідає за збереження матеріальних цінностей та “інтелектуального будинку” загалом від несанкціонованого проникнення та зовнішнього впливу на систему, в той час як підсистемі клімат-контролю належить найнижчий рівень пріоритету, що зумовлено насамперед високою інерційністю до зміни основних параметрів даної підсистеми. З розвитком системи “інтелектуального будинку”, значення пріоритетів також буде змінюватися. Зокрема, при встановленні підсистеми контролю стану здоров’я мешканців, значення пріоритету цієї підсистеми повинно бути найвищим. Нижче, на рис. 2.9, представлено граф досяжності станів, розробленої на основі кольорової мережі Петрі моделі системи “інтелектуального будинку” [19].

Таблиця 2.3. Позиції моделі системи ІБ на основі кольорової мережі Петрі

№	Позиція	Призначення
p1	Початок роботи	Позиція відповідає за запуск роботи моделі, розміщення маркера в дану позицію розпочинає роботу моделі
p2	Автоматичний режим роботи системи ІБ	Наявність маркера в даній позиції свідчить про встановлення автоматичного режиму роботи системи ІБ
p3	Режим користувача роботи системи ІБ	Наявність маркера в даній позиції свідчить про вибір режиму користувача роботи системи ІБ
p4	Очікування події	Позиція очікування події, наявність маркера в даній позиції свідчить про готовність виявлення активної події в системі
p5	Подія	Позиція — “магазин подій”. В даній позиції містять маркери активних подій, що виникли в різних підсистемах ІБ
p6	Готовність до запуску нейроконтролера	Наявність маркера в даній позиції свідчить про готовність системи до запуску механізму корекції параметрів системи ІБ
p7	Автоматичний режим роботи системи	Наявність маркера в даній позиції свідчить про обраний автоматичний режим роботи системи і подальший запуск нейроконтролера
p8	Режим користувача	Наявність маркера в даній позиції свідчить про обраний режим користувача роботи системи ІБ і очікування підтвердження дозволу користувача на здійснення корекції параметрів системи ІБ
p9	Дозвіл від користувача	Наявність маркера в даній позиції свідчить про ознайомлення користувача з необхідністю здійснення корекції параметрів системи ІБ у вказаних областях та його дозвіл на запуск нейроконтролера
p10	Підтвердження дозволу від користувача	Наявність маркера в даній позиції свідчить про обраний режим користувача роботи системи ІБ і подальший запуск нейроконтролера
p11	Нейроконтролер	Наявність маркера в даній позиції свідчить про успішний запуск нейроконтролера та перехід до фази корекції необхідних параметрів ІБ
p12	Підсистема моніторингу	Наявність маркера в даній позиції свідчить про запуск підсистеми моніторингу системи ІБ
p13	Кінець	Наявність маркера в даній позиції свідчить про успішне завершення роботи моделі

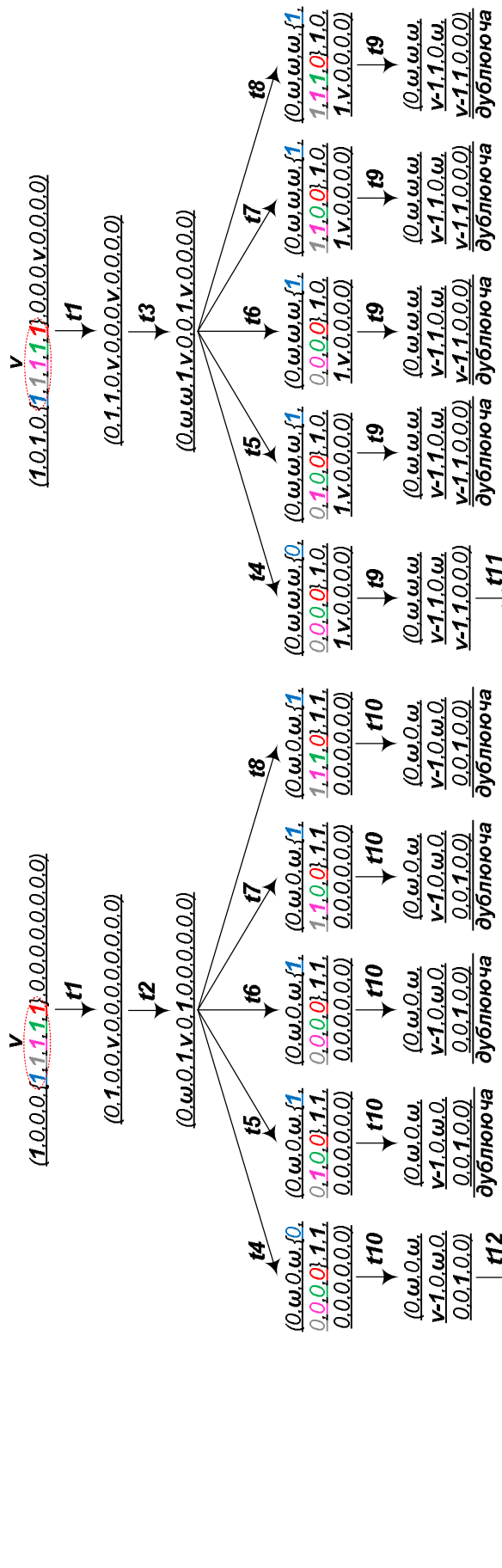


Рис. 2.9. Граф досяжності станів розробленої на основі кольорової мережі Петрі моделі системи ІБ

Таблиця 2.4. Пріоритетні рівні основних функціональних підсистем ІБ

Підсистема	Пріоритетний рівень
Підсистема захисту	1
Підсистема безпеки	2
Підсистема освітлення	3
Підсистема управління побутовою технікою	4
Підсистема клімат-контролю	5

В роботі розроблено структурну схему системи ІБ, загальний алгоритм роботи системи ІБ, а також модель системи ІБ на основі кольорової мережі Петрі. Розроблена структурна схема системи ІБ охоплює ряд основних структурно-функціональних підсистем, які дають змогу здійснити автоматичну корекцію основних параметрів ІБ для досягнення максимально комфортних внутрішньо-мікрокліматичних умов та максимальної економії енергоресурсів, забезпечуючи при цьому захист від проникнення зловмисників на територію ІБ, а також захист від імовірного пошкодження майна, спричиненого аварійними ситуаціями техногенного характеру (витік природного газу, протікання води, виникнення пожежі всередині приміщення ІБ, і т.п.).

Відповідно до запропонованої структурної схеми та загального алгоритму роботи системи ІБ, в роботі розроблена модель системи ІБ на основі кольорової мережі Петрі, що дає змогу дослідити динаміку поведінки всієї системи ІБ загалом, та внутрішньої взаємодії її основних структурно-функціональних підсистем, а також побудовано граф досяжності станів розробленої моделі на основі кольорової мережі Петрі.

2.4. Розроблення моделі на основі мереж Петрі для аналізу роботи підсистеми клімат-контролю “інтелектуального будинку”

Однією з основних підсистем “інтелектуального будинку” є підсистема клімат-контролю, що забезпечує створення комфортних умов проживання в будинку для власника.

Підсистема клімат-контролю системи ІБ складається з ряду давачів для збору вхідної інформації, контролера для її опрацювання, та ряду активаторів – пристроїв для здійснення необхідної корекції стану довкілля (обігрівач, витяжка, зволожувач повітря тощо).

Таким чином, отримують базову структуру підсистеми клімат-контролю певної структурної одиниці ІБ (кімнати), яку зображено на рис. 2.4.

Роль контролера виконує нейроконтролер, який моделює мережу багатошарового перцептрона [14]. Основним завданням нейроконтролера є забезпечення реагування на зміну вхідних параметрів системи та коректної взаємодії між вхідними та вихідними пристроями (між давачами та активаторами). На рис. 2.10, подано модель на основі кольорових мереж Петрі для підсистеми клімат-контролю ІБ.

Математична модель розробленої підсистеми має наступну форму:

$$CPN=\{PoS, TranS, ArcS, BBPoSMark, TpS, PoSTpS, ArcSTpS, Cnd\}, \quad (2.4)$$

де $PoS= \{p_1, p_2,\ldots,p_n\}$ – множина позицій (станів);

$TranS= \{t_1, t_2,\ldots, t_m\}$ – множина переходів;

ArcS – множина вхідних та вихідних дуг по відношенню до переходу;

BBPoSMark – множина, яка задає початкове маркування мережі Петрі;

TpS – множина типів;

PoSTpS – множина, яка відображає доступну множину типів у позиціях мережі;

ArcSTpS – множина типів маркерів, що збуджують перехід, або вказує які типи маркерів будуть згенеровані переходом;

Cnd – множина умов збудження переходів.

Застосування кольорових мереж Петрі дало змогу реалізувати розподіл основних процесів контролю мікроклімату в ІБ.

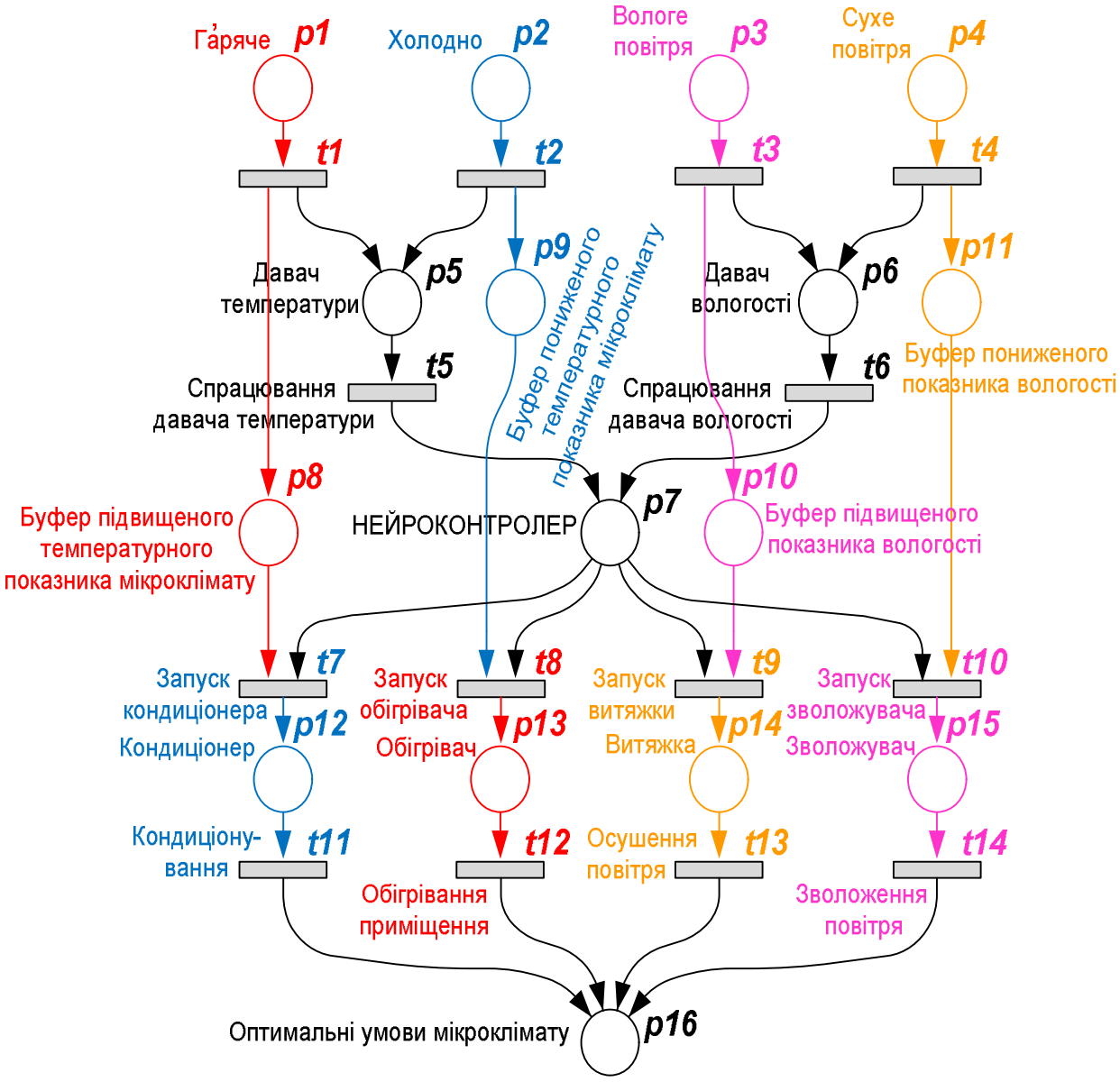


Рис. 2.10. Модель на основі кольорової мережі Петрі для підсистеми клімат-контролю

Розроблена модель на основі кольорової мережі Петрі для підсистеми клімат-контролю ІБ складається з позицій (табл. 2.5) та переходів (табл. 2.6), і дає змогу здійснити динамічний аналіз поведінки даної підсистеми.

Таблиця 2.5. Основні позиції розробленої моделі на основі кольорової мережі Петрі та їх призначення

№	Позиція	Призначення
p1	Температурний показник стану мікроклімату однієї кімнати ІБ – “Га́ряче”	Наявність маркера в даній позиції свідчить про підвищений рівень температури повітря в конкретній кімнаті ІБ
p2	Температурний показник стану мікроклімату однієї кімнати ІБ – “Холодно”	Наявність маркера в даній позиції свідчить про понижений рівень температури повітря в конкретній кімнаті ІБ
p3	Показник вологості стану мікроклімату кімнати ІБ – “Вологе повітря”	Наявність маркера в даній позиції свідчить про підвищений рівень вологості повітря в конкретній кімнаті ІБ
p4	Показник вологості стану мікроклімату кімнати ІБ – “Сухе повітря”	Наявність маркера в даній позиції свідчить про понижений рівень вологості повітря в конкретній кімнаті ІБ
p5	Давач температури	Наявність маркера в даній позиції свідчить про відхилення температурного показника мікроклімату конкретної кімнати ІБ від оптимального значення
p6	Давач вологості	Наявність маркера в даній позиції свідчить про відхилення показника вологості мікроклімату конкретної кімнати ІБ від оптимального значення
p7	НЕЙРОКОНТРОЛЕР	Наявність маркера в даній позиції свідчить про запуск нейроконтролера з метою оптимізації мікроклімату всередині конкретної кімнати ІБ
p8	Буферна позиція	Буфер підвищеного температурного показника мікроклімату
p9	Буферна позиція	Буфер пониженого температурного показника мікроклімату
p10	Буферна позиція	Буфер підвищеного показника вологості

Продовження таблиці 2.5

p11	Буферна позиція	Буфер пониженого показника вологості
P12	Кондиціонер	Наявність маркера в даній позиції свідчить про запуск кондиціонера керуючим сигналом від нейроконтролера
P13	Обігрівач	Наявність маркера в даній позиції свідчить про запуск обігрівача керуючим сигналом від нейроконтролера
p14	Витяжка	Наявність маркера в даній позиції свідчить про запуск витяжки керуючим сигналом від нейроконтролера
p15	Зволожувач	Наявність маркера в даній позиції свідчить про запуск зволожувача керуючим сигналом від нейроконтролера
p16	Оптимальні умови мікроклімату	Наявність маркера в даній позиції свідчить про досягнення варіативних оптимальних умов мікроклімату всередині конкретної кімнати ІБ, та перехід підсистеми клімат-контролю в режим очікування

Нижче, на рис. 2.11, зображено граф досяжності станів побудованої моделі на основі кольорової мережі Петрі, що демонструє досяжність усіх її запланованих станів, відсутність тупиків та можливість визначення інших параметрів в процесі аналізу роботи підсистеми клімат-контролю ІБ.

Слід додати, що досить часто окрім контролю температури та вологості підсистема клімат-контролю може містити функцію іонізації повітря, що не створить проблем в процесі реалізації даної підсистеми.

Таблиця 2.6. Таблиця переходів розробленої моделі на основі кольорової мережі Петрі та їх призначення

Перехід	Призначення
t1	Вплив підвищеної температури повітря на давач температури всередині конкретної кімнати ІБ
t2	Вплив пониженої температури повітря на давач температури всередині конкретної кімнати ІБ
t3	Вплив підвищеної вологості повітря на давач вологості всередині конкретної кімнати ІБ
t4	Вплив пониженого рівня вологості повітря на давач вологості всередині конкретної кімнати ІБ
t5	Спрацювання давача температури
t6	Спрацювання давача вологості
t7	Запуск кондиціонера всередині конкретної кімнати ІБ
t8	Запуск обігрівача всередині конкретної кімнати ІБ
t9	Запуск витяжки всередині конкретної кімнати ІБ
t10	Запуск зволожувача всередині конкретної кімнати ІБ
t11	Кондиціонування конкретної кімнати ІБ
t12	Обігрівання конкретної кімнати ІБ
t13	Осушення повітря всередині конкретної кімнати ІБ
t14	Зволоження повітря всередині конкретної кімнати ІБ

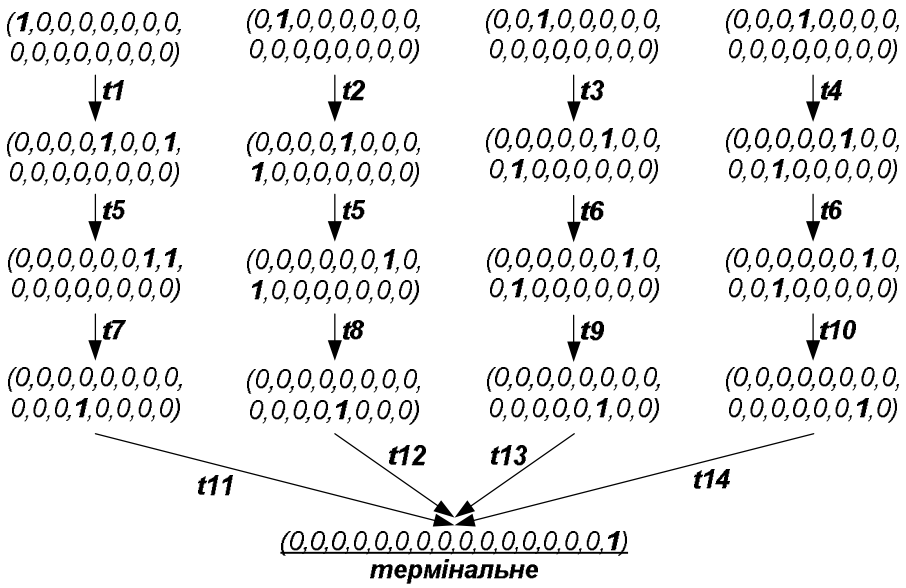


Рис. 2.11. Граф досяжності станів кольорової мережі Петрі підсистеми клімат-контролю

2.5. Розроблення моделі на основі мереж Петрі для аналізу роботи підсистеми освітлення “інтелектуального будинку”

Керування освітленням – одна з найважливіших підсистем, що забезпечує не тільки комфорт у будинку, але й значну економію споживання електроенергії, адже, за статистикою, від 20 до 50 відсотків від загального обсягу споживаної енергії в будинках і офісах використовуються саме для освітлення.

Однією з основних функцій даної підсистеми є повна автоматизація управління освітленням, що досягається використанням спеціальних давачів. Для прикладу, зовнішня лампа при вході в будинок спалахує як тільки людина підійшла до дверей, і вимикається через деякий час після її відходу завдяки давачу руху. У житлових кімнатах (спальня, вітальня, та інші) можна встановити звукові давачі, завдяки яким світло вмикається чи вимикається спеціальним звуковим сигналом (наприклад, поплескуванням долонями). Регулювання яскравості задається відповідними пристроями: димерами, реле тощо.

Спеціальні давачі освітлення (фоторезистори) можна розмістити також зовні будинку і при зміні освітленості протягом доби автоматично вмикатиметься або вимикатиметься зовнішнє освітлення фасаду, а також відповідним чином реагуватимуть жалюзі всередині будинку.

Розроблена структура підсистеми освітлення [6, 15, 17] містить звукові давачі, давачі освітлення та давачі руху. В ролі активаторів виступають пристрої вмикання зовнішнього і внутрішнього освітлення будинку та регулювання жалюзями. Приклад спрощеної структури підсистеми освітлення зображено на рис.2.12.

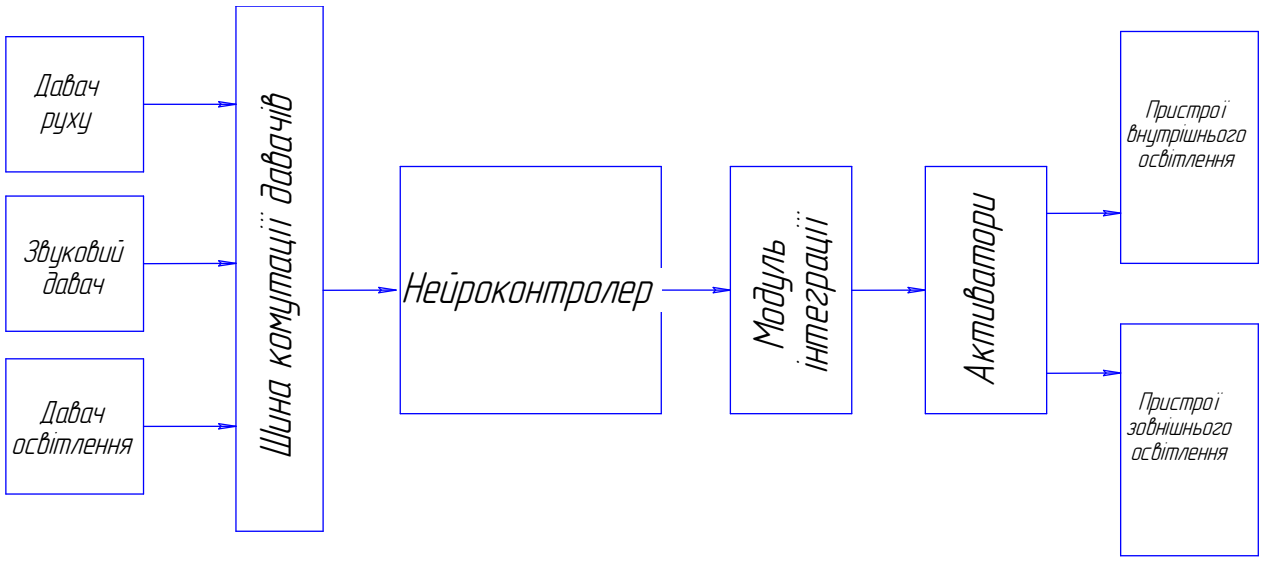


Рис. 2.12. Спрощена структурна схема підсистеми освітлення

Від давачів інформація про стан підсистеми передається через спеціальну шину комутації до центрального блоку керування, роль якого виступає розроблений нейроконтролер.

Важливою складовою проектування будь-якої підсистеми є дослідження динаміки та надійності її функціонування. Саме з цією метою розроблено відповідну модель підсистеми освітлення ІБ. Модель розроблена на основі мереж Петрі і якраз дає змогу дослідити динаміку та надійність підсистеми ще на етапі системного проектування.

В основу розробленої моделі покладено відповідні давачі, які передають інформацію до нейроконтролера, а він, в свою чергу, активізує відповідні виконуючі пристрої. Основна задача нейроконтролера полягає в автоматичному регулюванні освітлення в приміщеннях та зовні будинку в такий спосіб, щоб досягнути мінімальних затрат електроенергії, забезпечуючи при цьому максимальний комфорт для користувача. В розробленій моделі (рис. 2.13) використано принцип, відповідно до якого при спрацюванні певного давача відбувається його перехід із стану спокою (0) в активний стан (1).

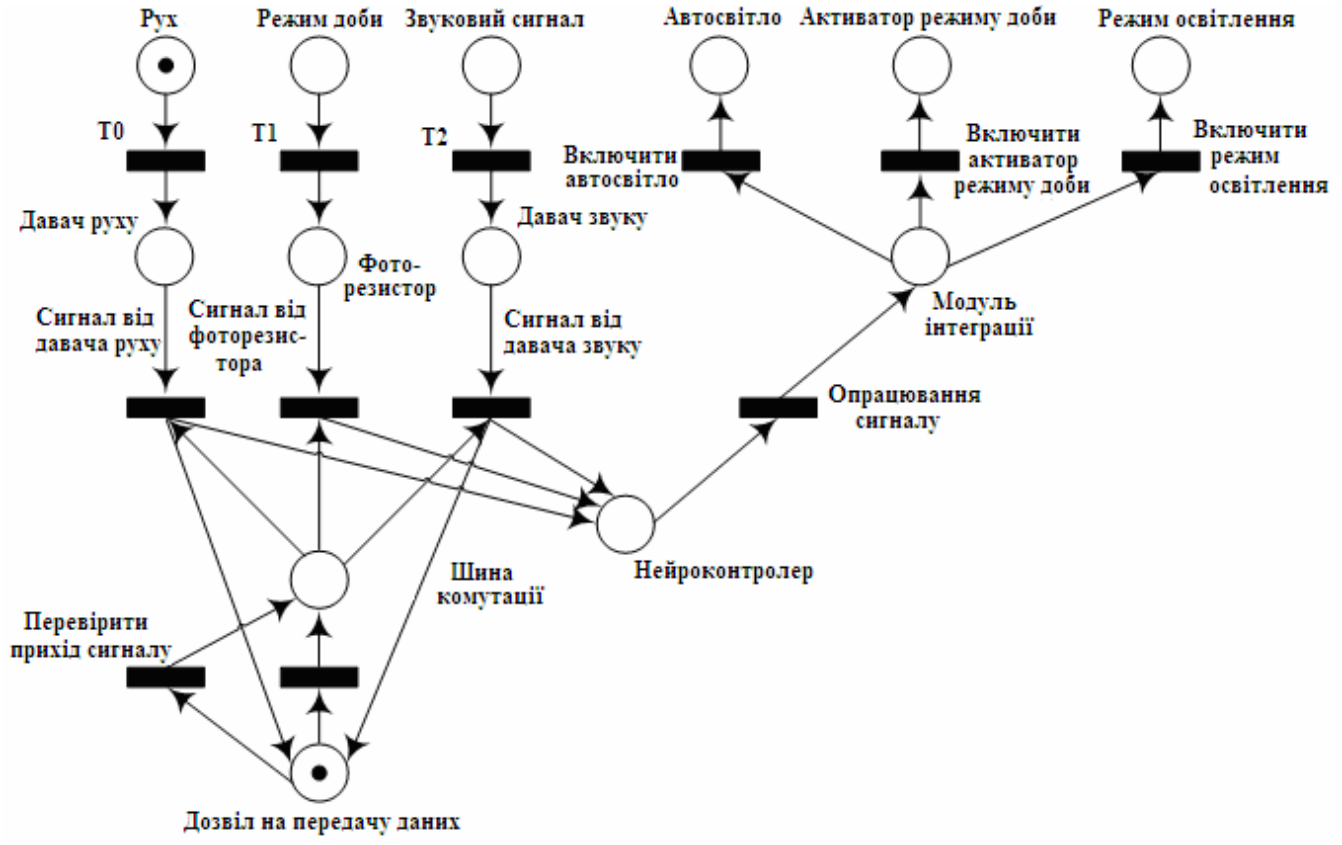


Рис. 2.13. Модель на основі простої мережі Петрі для підсистеми освітлення ІБ

Наприклад, у випадку переміщення користувача в межах ІБ, давач руху активізується і, при умові дозволу на передачу даних, сигнал від давача по шині комутації поступає на вхід нейроконтролера. Дозвіл на передачу даних формує чергу на опрацювання і запобігає, таким чином, втраті інформації. При активізації інших давачів робота системи відбувається аналогічно.

Нейроконтролер опрацьовує отриману від давачів інформацію і, відповідно до умов та місця перебування користувача, визначає, де саме вмикати або вимикати світло при русі чи поданні звукового сигналу; при зміні режиму доби автоматично розсувати чи засувати жалюзі, вмикати або вимикати зовнішнє освітлення будинку тощо. Множини позицій та переходів розробленої на основі мереж Петрі моделі підсистеми освітлення ІБ подана в табл. 2.7 та 2.8, відповідно.

Таблиця 2.7. Позиції моделі підсистеми освітлення ІБ на основі мереж Петрі

Позиція	Призначення
Рух	Рух у місці з встановленим давачем руху
Режим доби	Визначення режиму доби (день/ніч)
Звуковий сигнал	Подання звукового сигналу для увімкнення/ вимкнення освітлення
Давач руху	Давач готовий до передачі інформації
Давач освітлення	Давач готовий до передачі інформації
Давач звуку	Давач готовий до передачі інформації
Шина комутації	Забезпечення передачі даних з давачів на нейроконтролер
Дозвіл на передачу даних	Формує чергу даних на опрацювання
Нейроконтролер	Центральний елемент, який здійснює керування підсистемою освітлення
Модуль інтеграції	Дозволяє взаємодіяти з іншими підсистемами ІБ
Автосвітло	Автоматичне увімкнення/вимкнення світла вхідних дверей та входу до ванної кімнати
Активатор режиму доби	Керування зовнішнім освітленням будинку та жалюзі
Режим освітлення	Увімкнення та вимкнення світла в кімнаті будинку за звуковим сигналом

Розроблена модель на основі простої мережі Петрі складається з позицій, переходів, вхідних та вихідних дуг, а також множини початкових позицій (які ініціюються безпосередньо перед запуском моделі). Її структура представляється у такому вигляді [179]:

$$N_{simle} = \{P, T, F, M_0\}, \qquad (2.5)$$

де P – множина позицій;

T – множина переходів;

F – множина вхідних та вихідних дуг;

M_0 – множина початкових позицій, які ініціюються перед запуском моделі.

Таблиця 2.8. Переходи моделі підсистеми освітлення ІБ на основі мереж Петрі

Перехід	Призначення переходу
t1	Відбувався рух у місці з встановленим давачем руху
t2	Режим доби змінився з дня на ніч/ з ночі на день
t3	Пролунав звуковий сигнал для увімкнення/вимкнення світла
t4	Передача даних на нейроконтролер від давача руху
t5	Передача даних на нейроконтролер від фоторезистора
t6	Передача даних на нейроконтролер від давача звуку
t7	Передача даних заборонена
t8	Передача даних дозволена
t9	Передача сигналів з нейроконтролера на модуль інтеграції
t10	Запуск автоматичного ввімкнення/вимкнення світла вхідних дверей та входу до ванної кімнати
t11	Запуск керування зовнішнім освітленням будинку та жалюзями
t12	Запуск режиму освітлення в кімнатах будинку

При дослідженні розробленої на основі мереж Петрі моделі підсистеми освітлення ІБ встановлено, що всі стани побудованої мережі Петрі досяжні, а множина станів є скінченою, про що свідчить побудований граф досяжності станів, зображений на рис. 2.14.

Модель на основі кольорових мереж Петрі надає набагато ширші можливості щодо аналізування роботи підсистем даного типу, проте було прийняте рішення про використання простої мережі Петрі для побудови моделі підсистеми освітлення ІБ, оскільки вона має кращі показники економічності.

Побудовані моделі дають змогу дослідити основні вихідні параметри системного рівня проектування.

2.6. Розроблення моделі на основі мереж Петрі для аналізу роботи підсистеми захисту “інтелектуального будинку”

Без перебільшення, однією з найважливіших підсистем "інтелектуального будинку" є також підсистема захисту, яка призначена для забезпечення захисту

приміщень від несанкціонованого доступу, а також охорони життя та здоров'я користувача. Для виконання цих функцій підсистема включає такі основні елементи:

– охоронно-пожежну сигналізацію;

– засоби контролю доступу в приміщення;

– підсистему відеоспостереження.

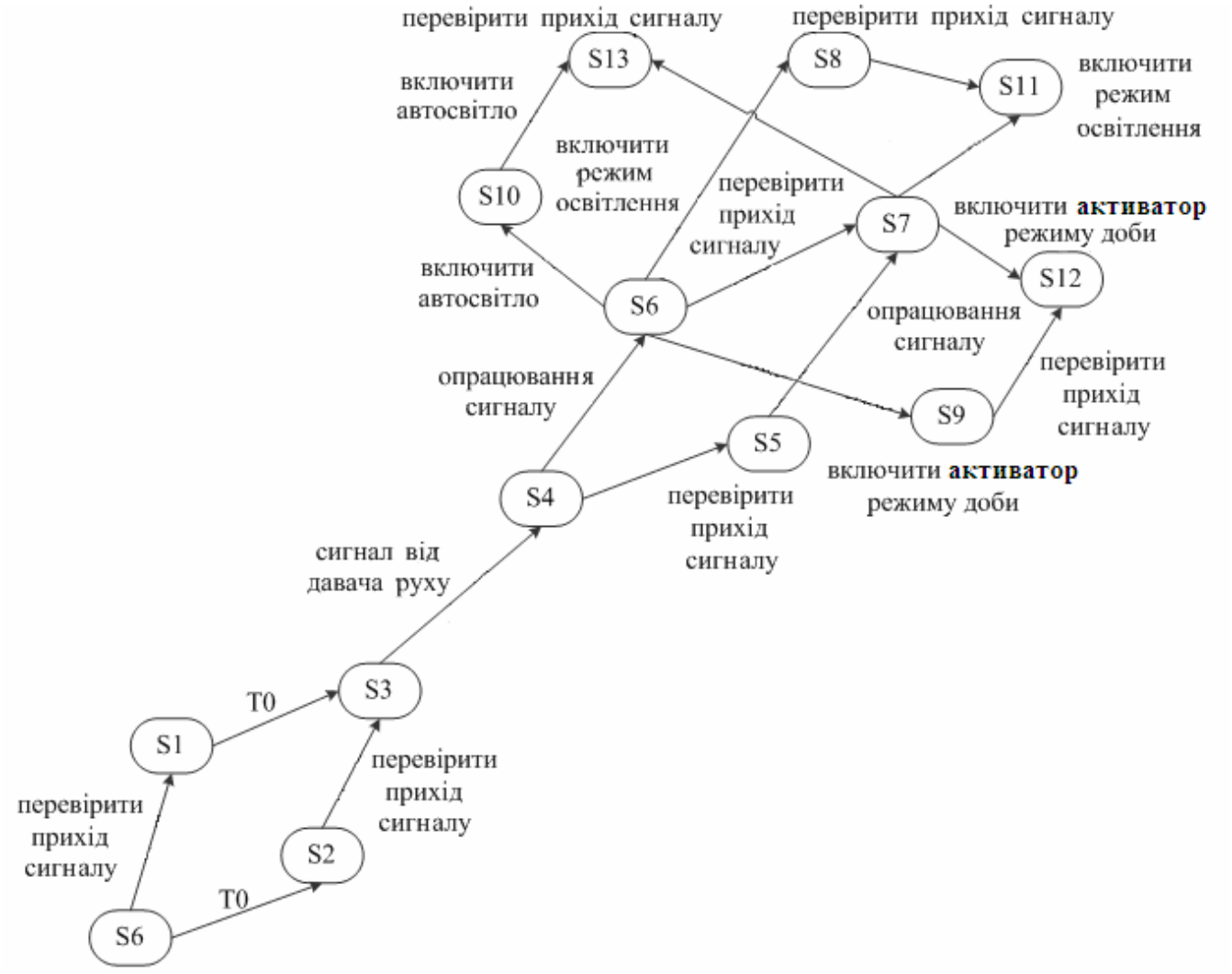


Рис. 2.14. Граф досяжності станів мережі Петрі підсистеми освітлення ІБ

Основна функція підсистеми захисту — перешкоджання несанкціонованому проникненню сторонніх осіб на приватну територію, виявлення таких спроб та інформування про них користувача та спеціальні служби. Саме тому першочерговим завданням даної підсистеми є відстежування цілісності периметру ІБ, і у випадку, якщо стороння особа намагатиметься проникнути всередину

периметру через двері чи вікно, чутливі пристрої одразу ж зафіксують переривання заданого контуру, в результаті цього активізуються системи світлового і звукового сповіщення, користувач отримає голосове або SMS-повідомлення про подію, а в службу охорони надходить тривожний сигнал. При цьому користувач ІБ може вибрати один з декількох рівнів захисту, зокрема, досконаліші системи при проникненні сторонньої особи блокують вікна і двері, перешкоджаючи, тим самим, порушнику покинути приміщення до прибуття працівників служби охорони.

Для захисту житла передбачені різні набори пристроїв. Можуть бути встановлені давачі руху або давачі перетину периметра, які працюють у поєднанні з підсистемою відеоспостереження. Завдяки їм будь-яке несанкціоноване проникнення людини або тварини на приватну територію негайно стає відомим користувачу. Давачі руйнування стін і розбиття вікон, домофони, ідентифікатори відвідувача, кодові замки, також підвищують рівень комфорту і безпеки користувачів систем ІБ.

Нейроконтролер виконує керування всією підсистемою відповідно до зазначених вище завдань, аналогічно до реалізації вищерозглянутих підсистем. На сьогодні ринок нейроконтролерів швидко розвивається, що дає змогу ефективно використовувати їх для конкретних задач нейрокерування. Саме тому нейроконтролери широко впроваджують, зокрема, і в системи ІБ.

Розроблена в роботі модель нейроконтролера ґрунтується на централізованому управлінні та системі інтеграції з іншими підсистемами. Це дозволяє значно зменшити кошти на реалізацію та підвищити ступінь ефективності у порівнянні з існуючими аналогами.

Розроблена структура підсистеми захисту ІБ (рис. 2.19) містить давачі руху, давачі розмикання контактів вікон та дверей, та давачі розбитого скла. З активаторів використано звукову та світлову сигналізацію, а також GSM-сповіщення користувача з допомогою SMS-повідомлень.

Відповідно до розробленої структури (рис. 2.15), центральним елементом є нейроконтролер [7, 16], основна задача якого полягає в опрацюванні отриманої від давачів інформації та прийняття рішення відповідно до умов навчання, які

визначаються на стадії проектування. Нейроконтролер і давачі взаємодіють між собою через шину комутації, яка також виконує функцію маршрутизатора даних, що надходять від давачів. Давачі працюють на основі булевої алгебри, тобто, в режимі очікування на виході давача присутній сигнал низького рівня (“логічний 0”), а у активному режимі – сигнал високого рівня (“логічна 1”).

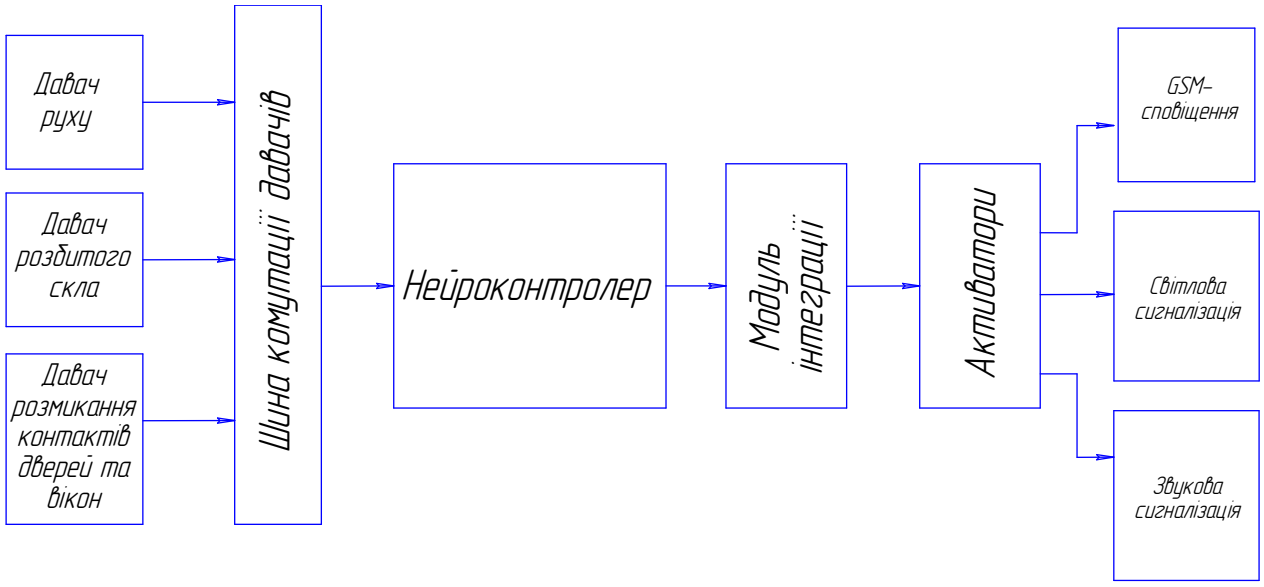


Рис. 2.15. Структурна схема підсистеми захисту

Модуль інтеграції представляє собою адресний простір на шині передачі даних контролера, і призначений для того, щоб підсистема захисту могла взаємодіяти з іншими підсистемами ІБ. Наприклад, коли відбулося потрапляння зловмисника всередину периметра ІБ, підсистема захисту реагує відповідно до умов та здійснює надсилання керуючих сигналів на активатори, які перебувають в інших підсистемах (світлова сигналізація – підсистема освітлення, звукова сигналізація – мультимедійна підсистема тощо).

Для дослідження роботи розробленої підсистеми захисту ІБ розроблена відповідна модель (рис. 2.16) на основі мереж Петрі, яка дає змогу покроково проаналізувати роботу нейроконтролера розробленої підсистеми та визначити місця можливих колізій з метою забезпечення можливості виправлення їх ще на етапі системного проектування. Підсистема базується на основі трьох видів

давачів, які надсилають інформацію на опрацювання до нейроконтролера, а він, в свою чергу, приймає відповідні рішення, та активізує відповідні активатори. В таблицях нижче представлений опис позицій (табл. 2.9) та переходів (табл. 2.10) розробленої моделі.

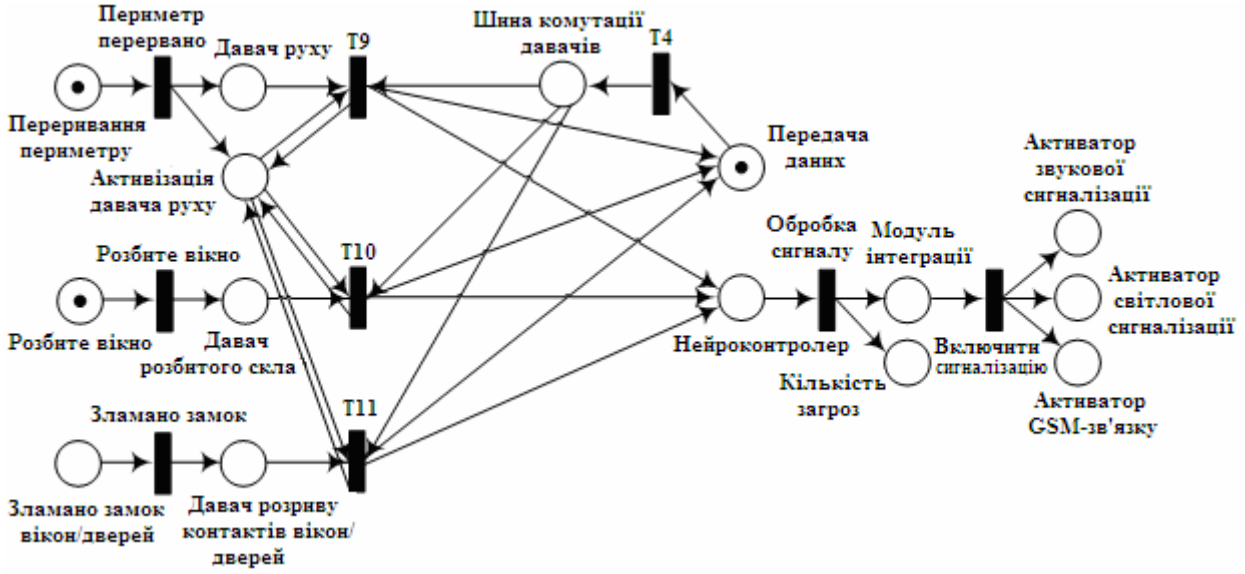


Рис. 2.16. Модель підсистеми захисту на основі простої мережі Петрі

В розробленій підсистемі захисту ІБ в якості домінуючої взята насамперед функція захист периметра будинку, оскільки вона попереджає та дає можливість відвернути більш масштабну загрозу пошкодження чи знищення приватного майна, та функція захисту життя і здоров’я користувачів в цілому. В зв’язку з цим основними функціональними елементами підсистема захисту є, насамперед, давачі руху, які контролюють зовнішній та внутрішній периметри ІБ, а при виникненні будь-якої загрози миттєво реагують та повідомляють користувача про небезпеку.

У випадку, коли відбувся несанкціонований перетин зловмисником, або сторонньою особою, периметра будинку, давач реєструє загрозу і активізує підсистему, яка до того перебувала в режимі очікування. В режимі очікування активними є лише давачі руху, а всі інші давачі активуються вже після його спрацювання, що дає змогу значно економити енергоресурси, не погіршуючи, при цьому, ефективність самого захисту.

Таблиця 2.9. Позиції мережі Петрі підсистеми захисту

Позиція	Призначення
Перетин периметру	Моделювання події проникнення зловмисників на приватну територію будинку
Розбито вікно	Моделювання події розбиття вікна зловмисниками
Зламаний замок вікон/дверей	Моделювання події зламу дверей/вікна зловмисниками
Давач руху	Давач готовий до передачі інформації
Давач розриву контактів	Давач готовий до передачі інформації
Давач розбитого скла	Давач готовий до передачі інформації
Активізація давача руху	Активізація отримання даних від давача руху
Активізація давача розбитого скла	Активізація отримання даних від давача розбитого скла
Активізація давача розриву контактів	Активізація отримання даних від давача розриву контактів дверей/вікон
Шина комутації давачів	Забезпечує передачу даних від давачів до контролера
Передача даних	Формування черги даних, отриманих від давачі, для їх подальшого опрацювання нейроконтролером
Нейроконтролер	Здійснює керування підсистемою захисту
Модуль інтеграції	Взаємодія з іншими підсистемами ІБ
Активатор звукової сигналізації	Керування звуковою сигналізацією
Активатор світлової сигналізації	Керування світловою сигналізацією
Активатор GSM-зв’язку	Відправлення SMS-повідомлення власнику про загрозу
Кількість загроз	Визначає кількість та вид загроз

Після активації давача інформація через шину комутації давачів поступає на вхід нейроконтролера. Отримавши дані від давачів, нейроконтролер опрацьовує їх та, відповідно до умов навчання нейронної мережі, приймає рішення щодо

подальших дій із забезпечення захисту будинку, що включає ввімкнення звукової сигналізації та надсилання SMS-повідомлення із статусом загрози всім користувачам, включеним в розсилку. Крім того, в контролері передбачена також функція визначення загрози (вид загрози, ступінь загрози), відповідно до якої спрацьовує необхідний активатор.

На початковому етапі спрацювання підсистеми захисту (при перериванні периметру) спрацьовує звукова сигналізація та надсилається SMS-повідомлення користувачам про загрозу, яку зареєстрував контролер. Коли спрацював давач розбитого скла чи давач розриву контактів дверей/вікон, контролер реєструє факт наявності зловмисників всередині будинку. При цьому надсилається нове SMS-повідомлення та активізуються всі інші активатори. На основі такого принципу функціонування підсистеми відповідним чином працює також і розроблена модель на основі мережі Петрі.

Таблиця 2.10. Переходи мережі Петрі підсистеми захисту

Перехід	Призначення переходу
t0	Відбулося переривання периметру будинку
t1	Розбито вікно
t2	Зламаний замок вікон чи дверей
t3	Активація давача розбитого скла та давача розриву контактів
t4	Передача даних на контролер від давача розбитого скла
t5	Передача даних на контролер від давача розриву контактів
t6	Повторна активація давача руху
t7	Передача даних на контролер від давача руху
t8	Заборона на передачу даних
t9	Дозвіл на передачу даних
t10	Передача керуючих сигналів на модуль інтеграції
t11	Запуск активатора звукової сигналізації
t12	Запуск активатора світлової сигналізації
t13	Запуск активатора GSM-зв’язку (GSM-сповіщення)

В результаті дослідження розробленої моделі на основі мережі Петрі підсистеми захисту ІБ встановлено, що всі стани розробленої моделі досяжні, а

множина станів є скінченою, що демонструє відповідний побудований граф досяжності станів, поданий на рис. 2.17.

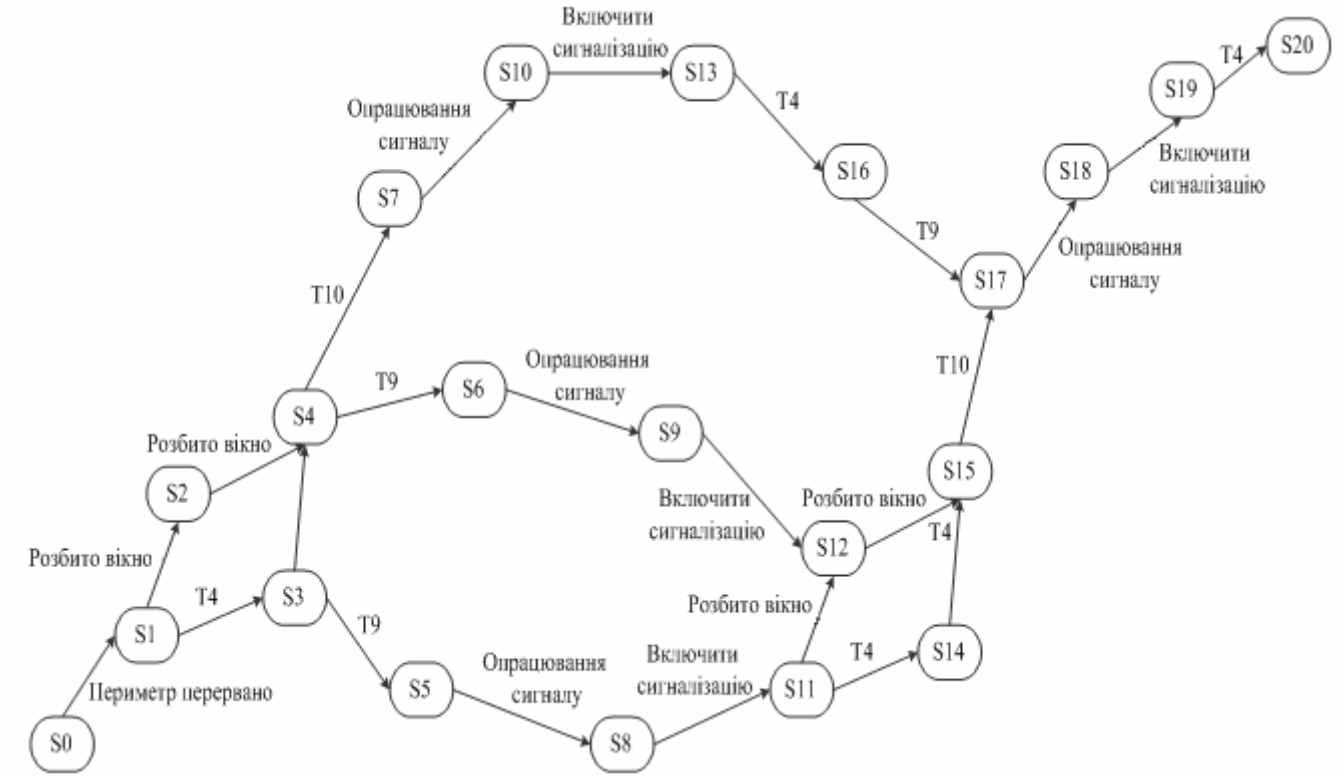


Рис. 2.17. Граф досяжності станів мережі Петрі підсистеми захисту ІБ

2.7. Розроблення моделі на основі мереж Петрі для аналізу підсистеми запобігання технічних аварій системи “інтелектуального будинку”

Отже, “інтелектуальний будинок” є складною системою, яка контролює різноманітні процеси як всередині будинку (такі як: клімат, освітлення, захист та інші), так і ззовні будинку (зовнішнє освітлення, захист периметру прилеглої території, зовнішнє відео-спостереження, зовнішня звукова та світлова сигналізація, і т. д.). Система “інтелектуального будинку” повинна визначати небезпечні ситуації та сприяти їх вирішенню. З цією метою розробляються та досліджуються можливості підсистеми запобігання технічних аварій.

Для прикладу, розглянемо одну з складових частин будівлі, а саме – кухню “інтелектуального будинку”, а для інших складових будівлі можна використати

аналогічні або ж навіть простіші технічні рішення. Зазвичай, кухня має водо- та газопостачання. Одними із основних проблем при цьому є людський фактор та проблеми із постачанням. Виходячи з цього, можуть виникнути наступні небезпечні ситуації:

– пожежа внаслідок необережного поводження з вогнем;

– задимлення кухні, викликане пожежею або необережним поводженням з вогнем;

– витік газу, викликаний неповним закриванням крану або прориванням труб підведення газу;

– протікання води, викликане проблемами із комунікаціями водопостачання чи водовідведення.

Система “інтелектуального будинку” повинна відслідкувати момент настання небезпеки і визначити порядок дій в кожній із зазначених ситуацій. Для цього в приміщенні розміщені наступні давачі: давач вогню; давач диму та газу; давач протікання води. Варто також відзначити, що підсистема спеціально розроблена таким чином, що при подальшому вдосконаленні підсистеми можна з легкістю інтегрувати в неї й інші необхідні давачі.

Для вирішення описаних вище проблем підсистема запобігання технічних аварій (ЗТА) може керувати такими пристроями як: електричний ключ перекривання газу; електричний ключ перекривання водопостачання; протипожежна система; витяжка та ін. Окрім того, при виникненні однієї із небезпечних ситуацій підсистема має повідомити про це користувача, наприклад, звуковими та/або світловими сигналами. Приклад розробленої спрощеної структури підсистеми ЗТА [8, 18] з врахуванням розглянутих вище складових, зображено на рис. 2.18.

Можливі сценарії роботи підсистеми ЗТА описані нижче.

При появі загоряння на кухні повинен спрацювати давач загоряння та давач задимлення. В цьому випадку необхідно перекрити газ, ввімкнути протипожежну систему та подати звукові і світлові сигнали, щоб сповістити користувачів всередині будинку.

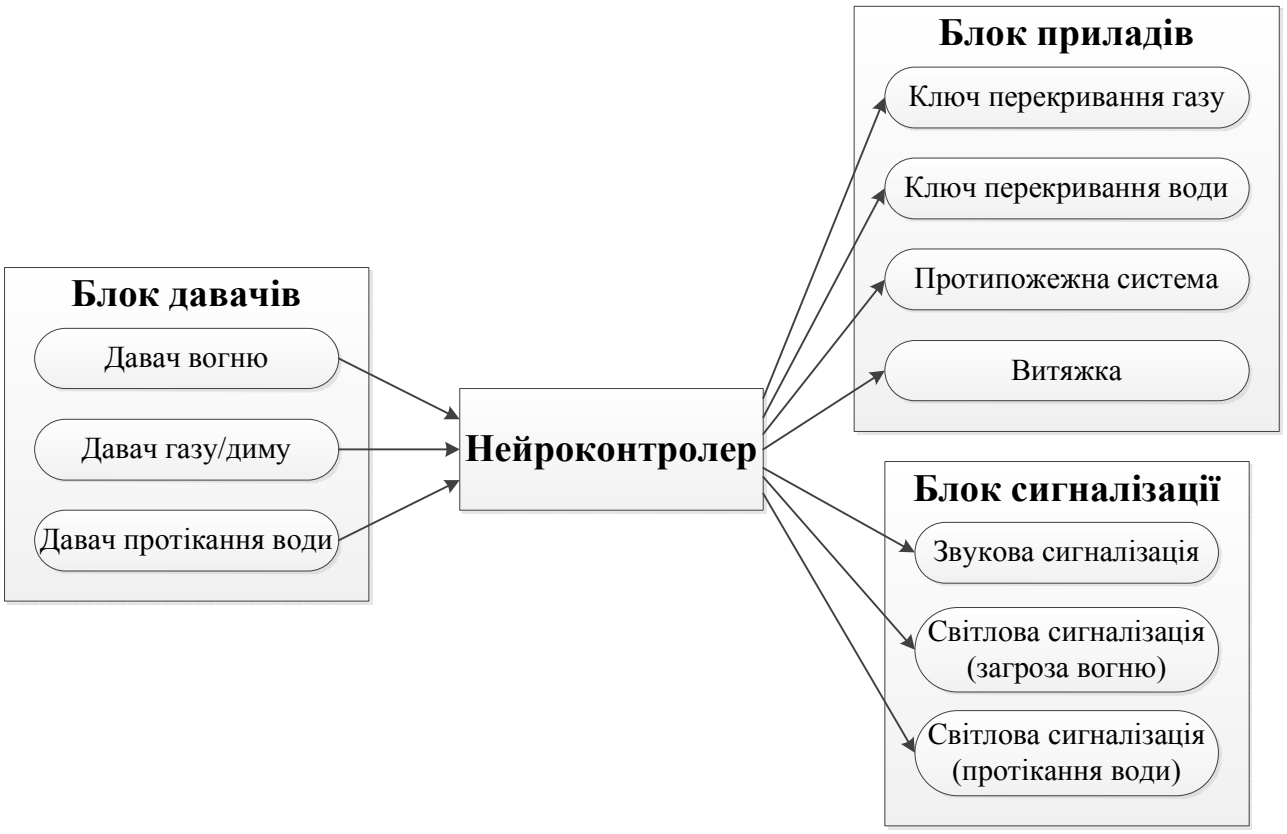


Рис.2.18. Структура підсистеми ЗТА системи “інтелектуального будинку”

При виявленні витікання води в приміщенні спрацьовує відповідний давач. В цьому випадку необхідно перекрити воду та подати звукові і світлові сигнали, щоб сповістити користувачів всередині будинку.

Якщо відбувається витік газу, тоді спрацьовує давач диму/газу. При цьому необхідно перекрити газ і ввімкнути витяжку для зменшення концентрації диму та газу, а також звукову та світлову сигналізацію.

Якщо відбувається задимлення, тоді спрацьовує давач диму/газу. При цьому необхідно перекрити газ і ввімкнути витяжку для зменшення концентрації диму та газу, а також звукову та світлову сигналізацію.

Для дослідження можливостей підсистеми ЗТА та її здатності коректно виконувати поставлені завдання, розроблена модель підсистеми для системного рівня проектування на основі кольорових мереж Петрі рис. 2.19. Граф досяжності станів розробленої моделі зображений на рис. 2.20.

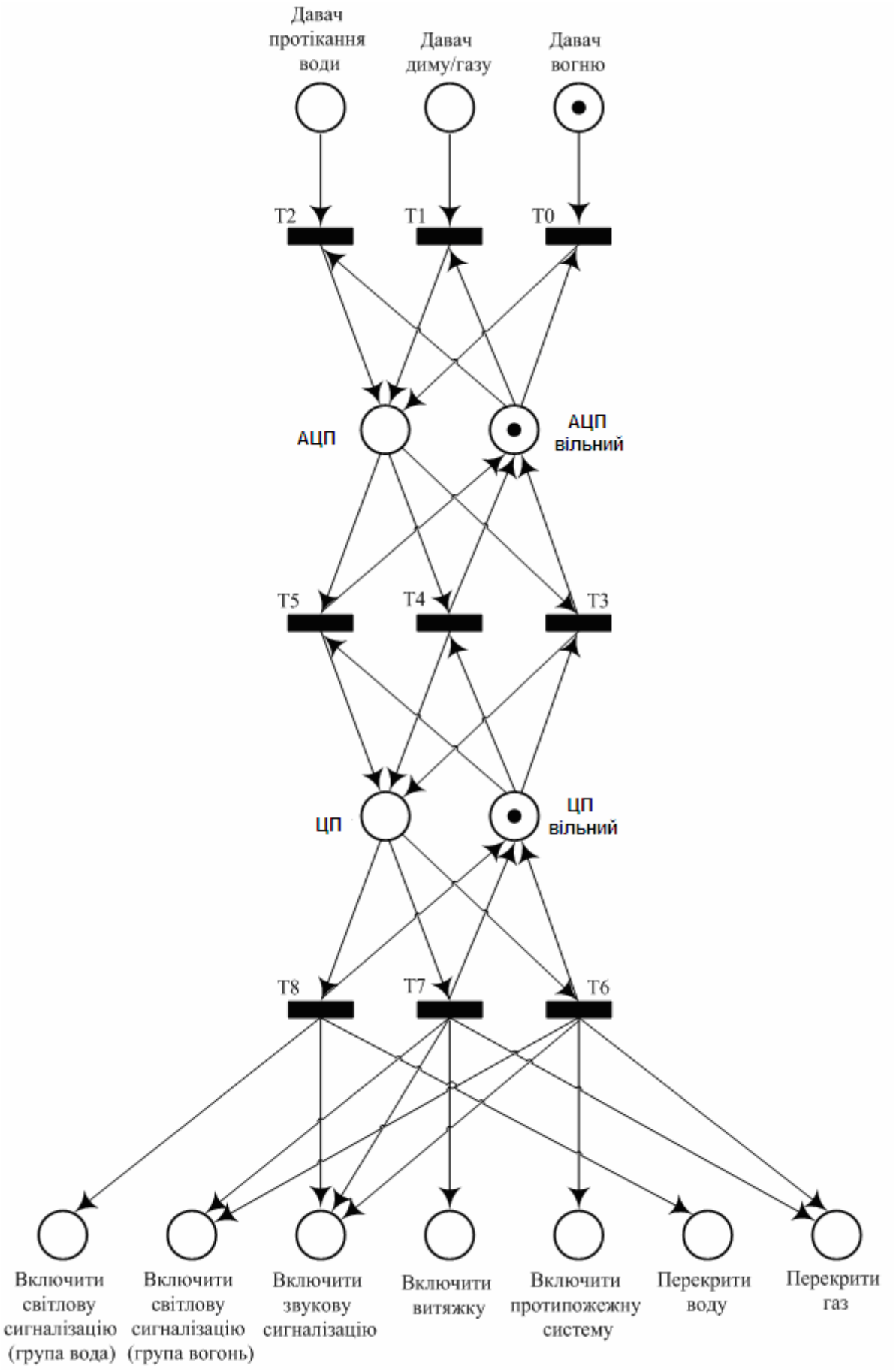


Рис.2.19. Структурна модель підсистеми ЗТА на основі кольорової мережі Петрі

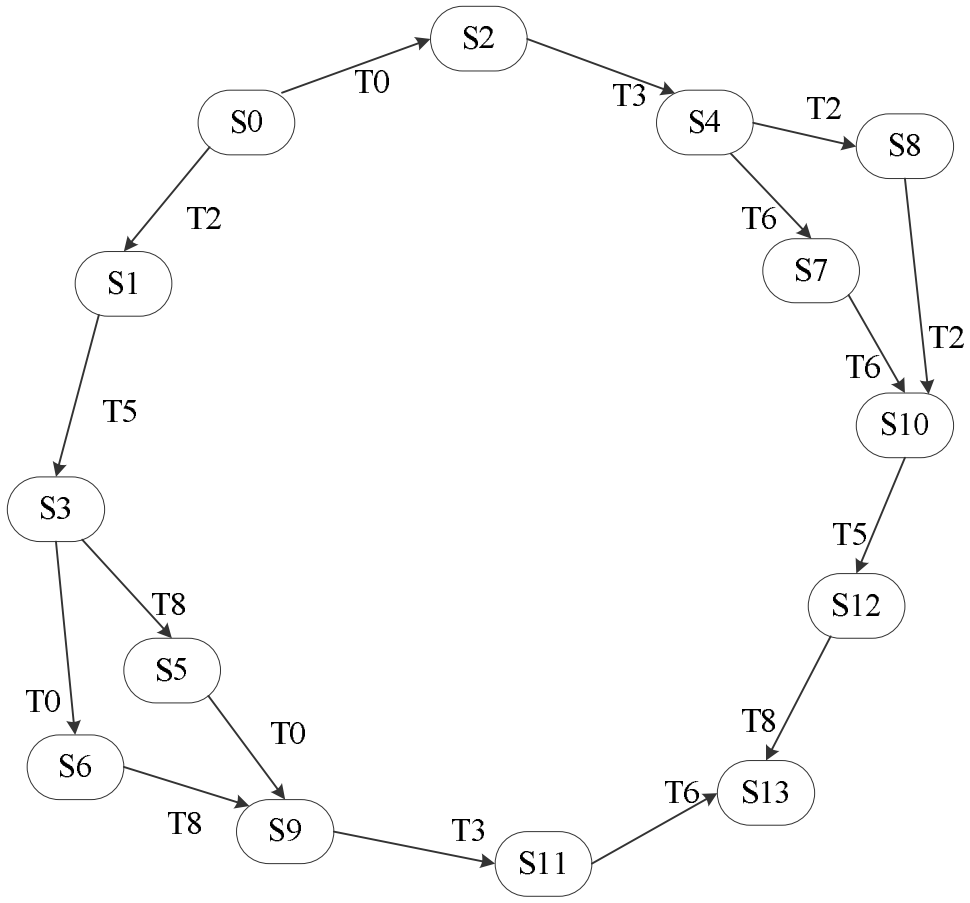


Рис.2.20. Граф досяжності станів роботи підсистеми

Кольорові мережі Петрі використані для того, щоб розрізняти сигнали від кожного із давачів, а саме (для розроблювальної підсистеми): червоний – сигнал від давача вогню, жовтий – сигнал від давача диму/газу, блакитний – сигнал від давача протікання води.

Розроблену модель на основі кольорових мереж Петрі можна умовно розбити на чотири частини, кожна з яких виконує власні функції:

– блок давачів – містить у собі давачі та переходи до блоку аналого-цифрового перетворювача (АЦП);

– блок аналогово-цифрового перетворювача, який містить АЦП та переходи у центральний процесор (ЦП). У блоці реалізовано послідовне опрацювання сигналів від кожного із давачів;

– блок центрального процесора, який містить ЦП та переходи до активаторів. У блоці реалізовано послідовне опрацювання даних від давачів;

– блок активаторів, який містить всі активатори згідно із структурою підсистеми ЗТА.

Побудована модель працює у відповідності до розробленого алгоритму. Він передбачає, що коли один з давачів фіксує у середовищі “інтелектуального будинку” негативні зміни, він формує відповідний сигнал і передає його на опрацювання в АЦП. Перетворена в цифровий формат інформація з виходу АЦП надходить у буферні каскади та на подальшу обробку ЦП. В залежності від виду технічної загрози яка виникла в середовищі ЦП приймає рішення, згідно з програмою роботи, про необхідні дії для усунення небезпеки яка виникла та інформування про це власника і, при потребі, комунальні служби. З цією метою він формує сигнали керування активаторами які повинні бути задіяні для аварійного перекривання систем подачі води, газу чи електроенергії, вимкнення необхідних приладів, усунення пошкоджень, ввімкнення сигналізації та систем зв’язку.

В результаті дослідження підсистеми ЗТА з використанням моделі на основі кольорових МП були побудовані відповідні графи досяжності станів (рис. 2.21 – рис. 2.24), які демонструють досяжність кожного із станів розроблених моделей, а також скінченість кожної множини станів.

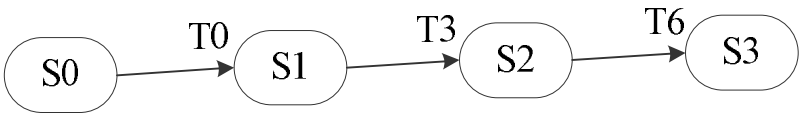


Рис.2.21. Граф досяжності станів для випадку спрацювання давача вогню

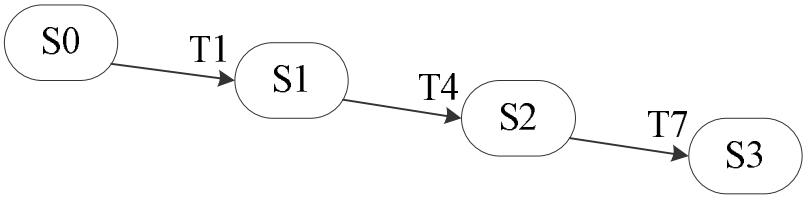


Рис.2.22. Граф досяжності станів для випадку спрацювання давача диму

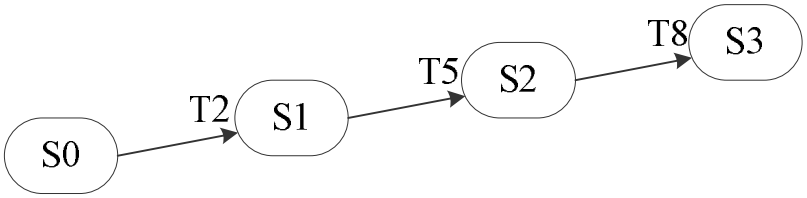


Рис.2.23. Граф досяжності станів для при спрацюванні давача протікання води

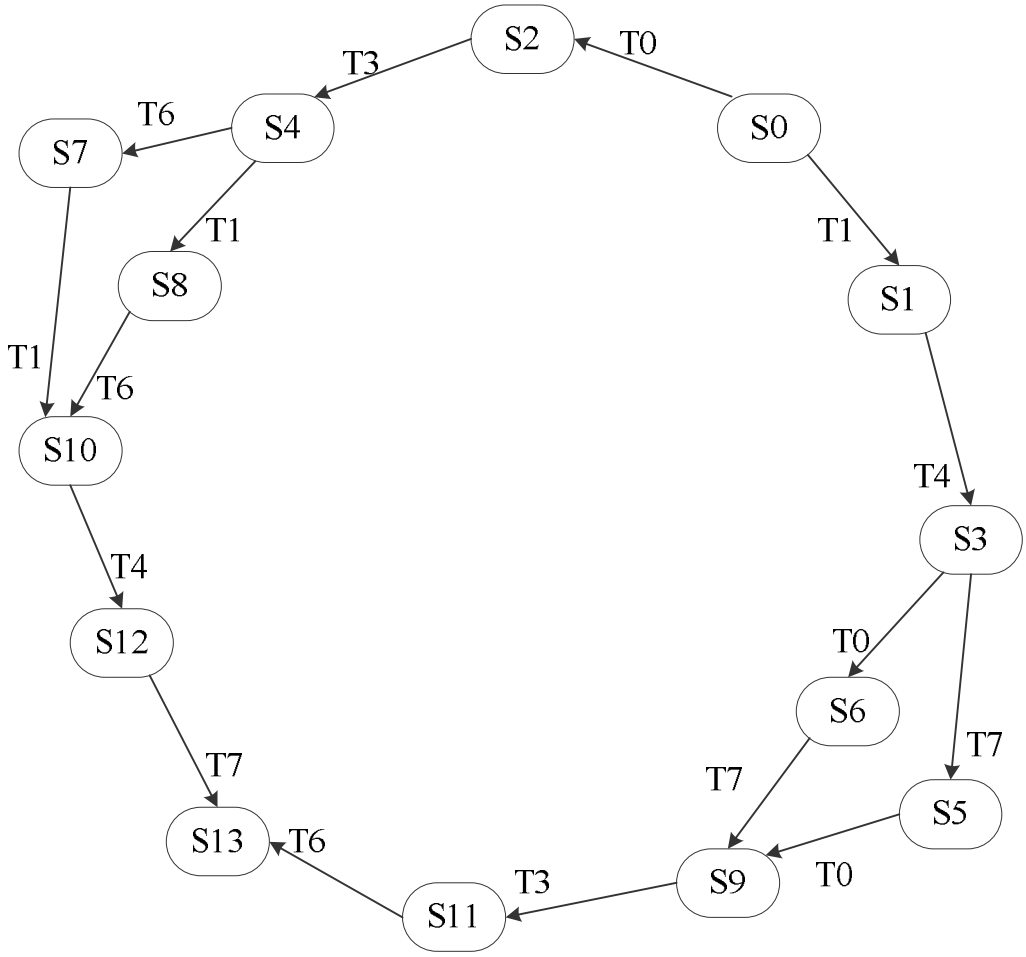


Рис.2.24. Граф досяжності станів при спрацюванні давачів вогню і диму

2.8 Висновки до розділу 2

1. Розроблено метод автоматизованого синтезування моделей на основі мереж Петрі для системного рівня автоматизованого проектування, який ґрунтується на інформації про структуру системи та дає змогу автоматизувати побудову структурних моделей підсистем “інтелектуального будинку”.

2. Запропоновано ввести інтелектуальну складову на усіх рівнях автоматизованого проектування таких систем та сформульовано основні задачі на кожному з ієрархічних рівнів, що дасть змогу підвищити ефективність проектування систем “інтелектуального будинку”.

3. Розроблено моделі для аналізування роботи системи “інтелектуального будинку”, які ґрунтується на теорії кольорових мереж Петрі і дають змогу

відслідковувати динаміку роботи, перевірити спроектовану систему на наявність тупиків, на живучість та обмеженість.

4. Розроблено моделі підсистем інтелектуального будинку на основі простих мереж Петрі, які дають змогу детально проаналізувати динаміку досліджуваних процесів всередині кожної з підсистем на системному рівні проектування системи ІБ та дослідити наявність надлишковості підсистем.

5. Розроблено загальну структурну схему системи “інтелектуального будинку”, що забезпечує ефективний механізм синхронізації основних підсистем та компонентів побудованої системи ІБ між собою, а також із користувачем. Також в роботі розроблено ряд структур основних підсистем ІБ, таких як підсистема клімат-контролю, підсистема освітлення, підсистема захисту та підсистема запобігання технічних аварій. Наведено результати дослідження розробленої системи “інтелектуального будинку” та її підсистем з допомогою побудованих моделей на основі мереж Петрі у формі графів досяжності станів.

РОЗДІЛ 3. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ТА МОДЕЛІ ОПРАЦЮВАННЯ НЕЧІТКИХ ДАНИХ В ПРОЦЕСІ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ СИСТЕМ “ІНТЕЛЕКТУАЛЬНОГО БУДИНКУ”

Система управління “інтелектуальним будинком” є складною системою та охоплює значну кількість складових. Разом з тим, можна виокремити три основні групи елементів ІБ, а саме:

– елементи, які призначені для збору інформації в середовищі ІБ – давачі різного функціонального призначення;

– підсистема опрацювання даних, яка передбачає наявність комп’ютера чи мікрокомп’ютера (можна використати синтез мікроконтролерів);

– група елементів, які безпосередньо впливають на середовище, в даному випадку йдеться про різноманітні перемикачі, крани, нагрівачі, зволожувачі [32, 180, 181, 182] тощо.

Реалізація системи “інтелектуального будинку” надзвичайно складна. Вона потребує розв’язання великої кількості практичних задач з різних областей науки і техніки.

Однією з проблем, з якою стикаються інженери в процесі розроблення різних підсистем “інтелектуального будинку”, це необхідність опрацювання сигналів та даних від підсистеми давачів. Кожний давач видає дані в певному форматі, з різною амплітудою, частотою тощо, тому розробники стикаються з необхідністю опрацювання неструктурованих та нечітких даних [10].

Під нечіткими множинами (множинами нечітких даних, чисел) розуміють узагальнення звичайних множин, коли ми відмовляємось від бінарного характеру характеристичної функції (функції, визначеної на множині X, яка вказує на приналежність елемента $x \in X$ деякій підмножині A, де $A \subseteq X$) і допускаємо, що вона може приймати будь-які значення з інтервалу [0;1] [183].

Відповідно, в дисертаційній роботі запропоновано використати штучні нейронні мережі для опрацювання даних від підсистеми давачів. В даному випадку

можна використати різні типи штучних нейронних мереж в залежності від складності та виду практичної задачі.

Практичну реалізацію підсистем “інтелектуального будинку” виконано із застосуванням нейроконтролерів [13], що передбачає використання мікроконтролерів з програмною реалізацією штучної нейронної мережі. Така реалізація дає змогу опрацьовувати нечіткі та неструктуровані дані від підсистеми давачів.

3.1. Розроблення нейроконтролера управління підсистемою клімат контролю “інтелектуального будинку”

При розв’язуванні прикладних задач з використанням методу математичного моделювання виникає потреба знаходження деякої складної функції, яка виконує багатовимірне перетворення вектора вхідних параметрів у вектор вихідних параметрів. Універсальним інструментом побудови такої функції є технології нейронних мереж.

Для відповіді на питання, чи завжди можна побудувати нейронну мережу для виконання заданого перетворення і яким вимогам вона повинна задовольняти, необхідно пригадати, що кожен нейрон нейронної мережі виконує сумування сигналів, що поступають від інших нейронів, в яких вони пройшли нелінійне перетворення. Тобто, перцептрон апроксимує залежності наочної області, які в загальному випадку є функціями багатьох аргументів, причому цю апроксимацію він виконує за допомогою суми функцій, кожна з яких є функцією тільки одного аргументу [184].

Професором Р. Хехт-Нільсеном в результаті перетворення теорем Арнольда – Колмогорова стосовно нейронних мереж, було доведено, що для будь-якої безлічі несуперечливих між собою пар довільної розмірності (X_q, D_q), q = 1, ..., Q, існує двошаровий перцептрон з сигмоїдними активаційними функціями і з кінцевим числом нейронів, який для кожного вхідного вектора X_q формує відповідний йому вихідний вектор D_q [184].

Таким чином, була доведена принципова можливість побудови нейронної мережі, що виконує перетворення, задане будь-якою множиною навчальних прикладів, що розрізняються між собою, і встановлено, що такою універсальною нейронною мережею є двошаровий перцептрон, тобто перцептрон з одним прихованим шаром, причому активаційні функції його нейронів мають бути сигмоїдними.

Необхідна кількість нейронів в прихованих шарах перцептрона можна визначити за формулою 3.1, що є наслідком з теорем Арнольда – Колмогорова – Хехт-Нільсена:

$$\frac{\mathrm{N_y Q}}{1+\log_2(\mathrm{Q})} \le N_w \le N_y\left(\frac{Q}{\mathrm{Nx}}+1\right)\left(N_x+N_y+1\right)+N_y \ , \qquad (3.1)$$

де $\mathrm{N_y}$ – розмірність вихідного сигналу;

Q – число елементів множини навчальних прикладів;

N_w – необхідне число синаптичних зв'язків;

N_x – розмірність вхідного сигналу.

Оцінивши за допомогою цієї формули необхідне число синаптичних зв'язків Nw, можна розрахувати необхідне число нейронів в прихованих шарах. Наприклад, число нейронів прихованого шару двошарового перцептрона буде рівне:

$$N = \frac{\mathrm{N_w}}{\mathrm{N_x}+\mathrm{N_y}}, \qquad (3.2)$$

Як випливає з теорем Арнольда – Колмогорова – Хехт-Нільсена, для побудови моделі на основі нейронної мережі будь-якого як завгодно складного об'єкту досить використовувати перцептрон з одним прихованим шаром сигмоїдних нейронів, число яких визначається формулами (3.1) та (3.2). Проте в практичних реалізаціях перцептронів як кількість шарів, так і число нейронів в кожному з них часто відрізняються від теоретичних. Іноді доцільно використовувати перцептрони з великою кількістю прихованих шарів. Такі перцептрони можуть мати менші розмірності матриць синаптичних ваг, ніж двошарові перцептрони, що реалізовують те ж саме перетворення.

Строгої теорії вибору оптимальної кількості прихованих шарів і нейронів в прихованих шарах поки не існує. На практиці найчастіше використовуються перцептрони, що мають один або два приховані шари, причому кількість нейронів в прихованих шарах зазвичай коливається від $N_x/2$ до $3N_x$ [184].

При проектуванні перцептронів необхідно розуміти, що перцептрон повинен не тільки правильно реагувати на приклади, на яких він навчений, але і уміти узагальнювати придбані знання, тобто правильно реагувати на приклади, яких в навчальній множині прикладів не було.

Таким чином при проектуванні штучних нейронних мереж типу багатошарового перцептрона були враховані такі правила:

1. Кількість входів перцептрона повинна збігатися з розмірністю вектора вхідних параметрів X, який визначений умовами вирішуваного завдання.

2. Кількість нейронів вихідного шару повинна збігатися з розмірністю вихідного вектора D, що також визначене умовами завдання.

3. Кількість прихованих шарів перцептрона згідно теоремам Арнольда – Колмогорова – Хехт-Нільсена повинна бути не менш одного, причому нейрони в прихованих шарах повинні мати сигмоїдну активаційну функцію.

4. Кількість нейронів в прихованих шарах може бути приблизно оцінена за формулами 3.1 та 3.2, проте її бажано оптимізувати для кожного конкретного завдання або експериментальним шляхом [184].

Зокрема підсистема клімат контролю “інтелектуального будинку” реалізована з використанням нейроконтролерів. Нейроконтролер симулює попередньо навчену штучну нейронну мережу типу багатошарового перцептрона [10, 185].

Загалом розроблена підсистема клімат-контролю “інтелектуального будинку” має структуру, яка зображена на рис. 3.1. Вона охоплює нейроконтролер для кожного приміщення, кожний з яких працює під керуванням своєї підсистеми ІБ. Особливістю такої реалізації є те, що використовуються відносно недорогі мікрокроконтролери, процес керування параметрами кожної з кімнат є незалежним. При цьому забезпечується висока надійність такої структури порівняно з використанням одного потужного контролера, при якій, в разі виходу з

ладу мікроконтролера – вся підсистема перестає працювати. Разом з тим, для підвищення надійності роботи, можна використати промислові мікроконтролери типу AVR чи інші.

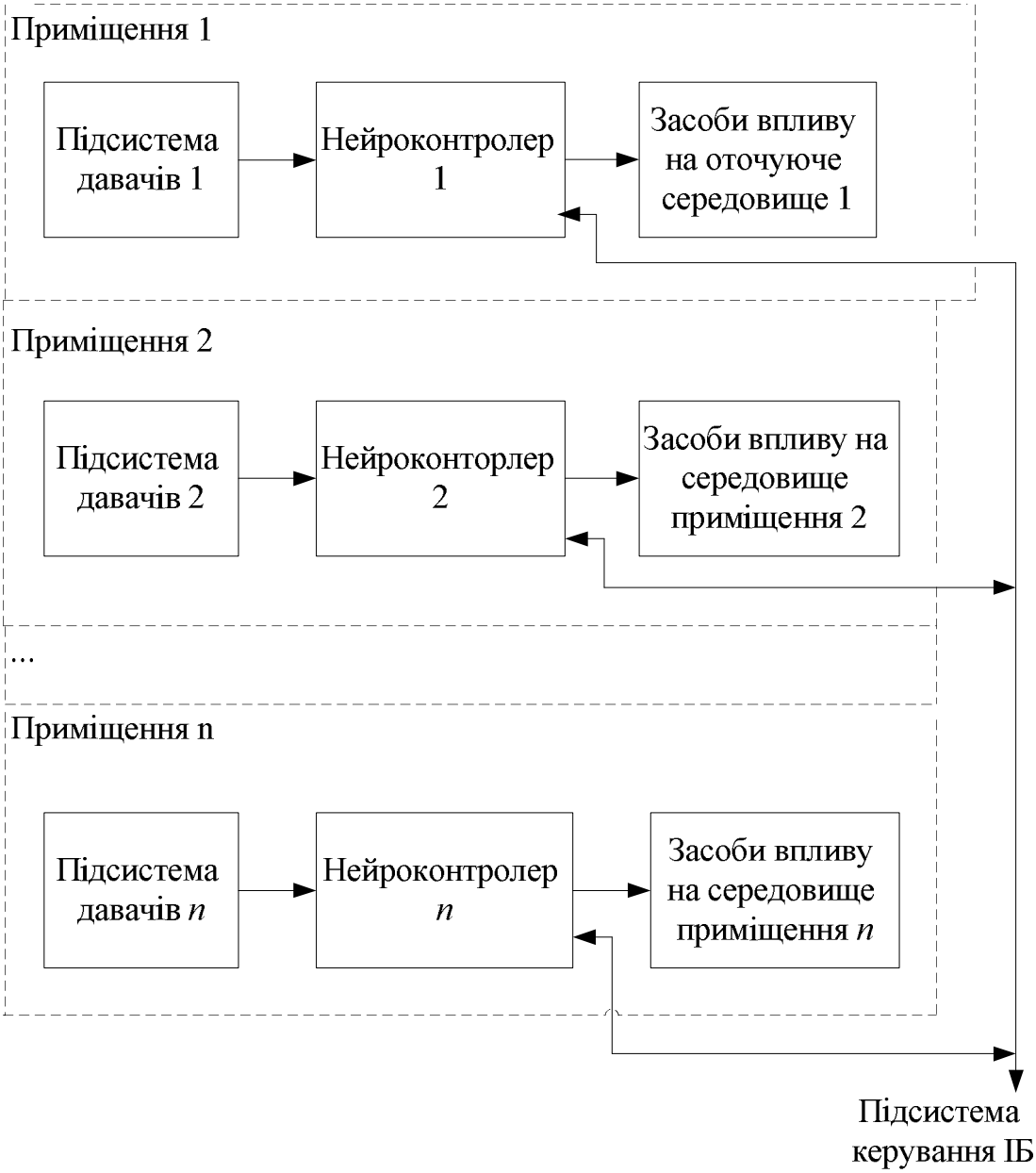


Рис.3.1 Структура підсистеми клімат-контролю “інтелектуального будинку”

В загальному випадку, таку структуру можна описати з використанням наступної моделі:

$$Mod_{кімн.контр} = \left(S_{\text{давачів}}, N_{\text{нейроконтролерів}}, Z_{\text{виконуючих пристроїв}} \right),$$

де $S_{\text{давачів}}$ – множина підсистем давачів для кожного приміщення;

$N_{\text{нейроконтролерів}}$ – множина підсистем нейроконтролерів;

$Z_{\text{виконуючих пристроїв}}$ – множина підсистем виконуючих пристроїв.

В свою чергу, $S_{\text{давачів}}$ містить n елементів, які є підсистемами давачів для кожного окремого взятого приміщення. В разі, якщо маємо n приміщень для підсистеми клімат-контролю, то будемо мати n елементів, кожний з яких позначається відповідним індексом:

$$S_{\text{давачів}} = \left(s_1, s_2, ..., s_n \right).$$

Аналогічні вирази можна записати для множини підсистем нейроконтролерів та виконавчих пристроїв, а саме:

$$N_{\text{нейроконтролерів}} = \left(n_1, n_2, ..., n_n \right),$$

$$Z_{\text{виконуючих пристроїв}} = \left(z_1, z_2, ..., z_n \right),$$

де s_i – підсистема давачів для i-го приміщення ($i = \overline{1,n}$);

n_i – нейроконтролер для i-го приміщення;

z_i – підсистема виконавчих пристроїв для i-го приміщення.

Відповідно, наведемо приклад розроблення моделей та програмного забезпечення нейроконтролера для одного приміщення, а для інших – він буде аналогічний (або реалізується за подібною схемою).

Отже, потрібно розробити нейроконтролер, який симулює просту штучну нейронну мережу типу багатошарового перцептрона для керування температурою та вологістю в одній кімнаті. Вибраний тип штучної нейтронної мережі дає змогу реалізувати задані функції підсистеми. У випадку необхідності реалізації складніших функцій, можна використати інші типи ШНМ.

Проектований нейроконтролер підсистеми клімат контролю, для прикладу, містить два давачі: давач температури і давач вологості. За допомогою даних від давачів необхідно керувати опаленням, вентилятором і зволожувачем повітря. Стратегія керування окремо взятим приміщенням, проілюстрована схемою, що зображена на рис. 3.2.

3.1.1. Розроблення моделі для опрацювання нечітких даних від давачів підсистеми клімат-контролю “інтелектуального будинку”

Розроблюваний нейроконтролер має виконувати вкладену логіку керування пристроями в залежності від отриманих сигналів від давачів.

Згідно з гігієнічними вимогами до мікроклімату житла температура в приміщенні має бути в межах 18 – 20 ^{o}C, а вологість в межах 30 – 60 %, хоча за бажанням власника приміщення ці параметри можна змінити. Тому при будь-якому відхиленні від зони комфорту необхідно коригувати мікроклімат у кімнаті. В залежності від значення мікроклімату було розроблено стратегію керування, яку можна зобразити у вигляді схеми, зображеної на рис.3.2.

За необхідності, стратегію керування мікрокліматом в кімнаті, можна змінити чи вдосконалити.

З наведеного рисунка випливає, що при температурі меншій від 18°C, а вологості меншій 30% необхідно ввімкнути зволожувач та обігрівач. Коли температура перебуває в межах від 18°C до 22°C і вологість в межах від 30% до 60% – нічого змінювати не потрібно. Аналогічним чином можна пояснити будь-яку ситуацію, яка зображена на рис.3.2.

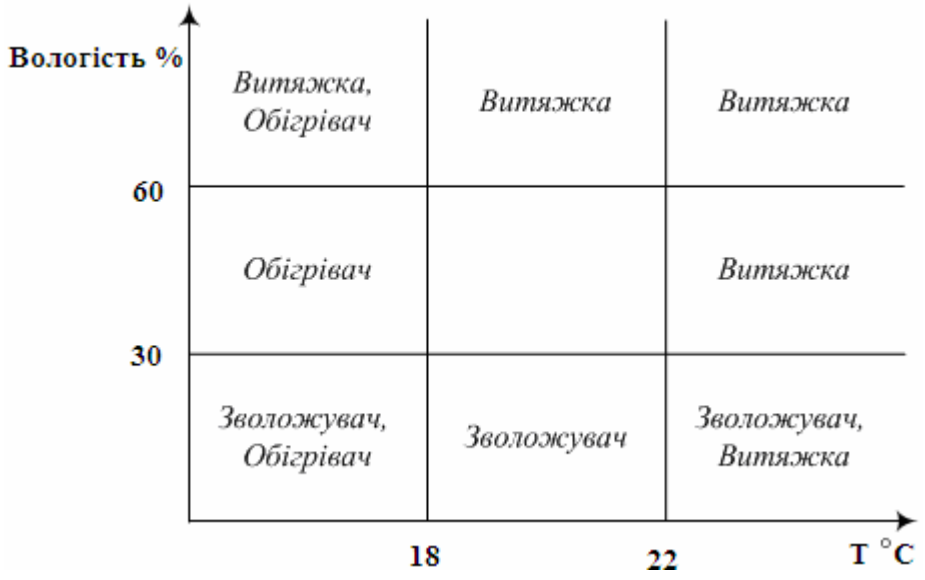


Рис. 3.2. Стратегія керування клімат контролем

В основі системи є мікроконтролер, який відповідає за роботу з виконавчими пристроями та давачами. Мікроконтролер виводить дані на екран комп'ютера через COM-порт чи пристрій відображення системи керування ІБ. Мікроконтролер можна запрограмувати на різноманітні функції і він здатний робити нескладні обчислення. Щоб не завантажувати повністю пам'ять і процесор мікроконтролера, використаємо урізану версію програми, яка містить усі необхідні елементи.

Отже, побудуємо модель для опрацювання нечітких даних на основі штучної нейронної мережі, яку буде емулювати програма на основі мережі типу багатошарового перцептрона.

В загальному випадку, математична модель штучного нейрона з n входами описується співвідношенням (3.3):

$$y = F\left(\sum_{i=1}^{n} w_i \times x_i\right) + b, \qquad (3.3)$$

де w_i – вагові коефіцієнти; x_i – значення входів; b – зміщення; F – функція активації; y – значення виходу нейрона.

Для опису структури ШНМ, в процесі реалізації, в роботі використано відповідну матрицю зв'язків.

Структура розробленої моделі нейронної мережі для підсистеми клімат-контролю, зображеної на рис. 2.5 в процесі реалізації вищенаведеної стратегії (рис. 3.2) містить 3 шари та загалом 12 нейронів.

Побудована штучна нейронна мережа має 2 вхідні нейрони і 3 вихідні нейрони. Кількість шарів і нейронів у шарах визначалась експериментально так, щоб після навчання похибка результатів була прийнятною. Найкращим отриманим варіантом є мережа з одним внутрішнім шаром на 4 нейрони. Також, в процесі реалізації, необхідно додати 2 балансуючі нейрони.

В результаті виконання навчання ШНМ з використанням програмного засобу, який детально описано в четвертому розділі дисертаційної роботи, було отримано мережу з параметрами, які наведено на рис. 3.3 – рис. 3.5, а на рис. 3.6 зображено графічне представлення структурної моделі розробленої ШНМ [9, 20].

Net
Neuron ID = 0 number of function = 2 1.0000
Neuron ID = 1 number of function = 2 1.0000
Neuron ID = 2 number of function = 2 1.0000
Neuron ID = 3 number of function = 3 1.0000
Neuron ID = 4 number of function = 3 1.0000
Neuron ID = 5 number of function = 3 1.0000
Neuron ID = 6 number of function = 3 1.0000
Neuron ID = 7 number of function = 3 1.0000
Neuron ID = 8 number of function = 3 1.0000
Neuron ID = 9 number of function = 1 1.0000
Neuron ID = 10 number of function = 1 1.0000
Neuron ID = 11 number of function = 1 1.0000

Рис.3.3. Параметри нейронів в розробленій моделі на основі ШНМ

Кожному нейрону, в процесі реалізації розробленої ШНМ, присвоєно ідентифікатор, номер функції перетворення та параметр функції.

Загалом, структурну модель ШНМ можна описати наступним кортежем:

$$Str\_\text{ШНМ} =< ID, N\_Fun, Par, N >, \qquad (3.4)$$

де $ID=id(i)$ – множина ідентифікаторів ШНМ ($i = \overline{1, N}$);

$N\_Fun=n\_fun(i)$ – множина функцій нейронів;

$Par=par(i)$ – множина параметрів нейронів;

N – кількість нейронів у ШНМ.

n n n o o o o o n n n n
n n n o o o o o n n n n
n n n o o o o o n n n n
i i i n n n n n n o o o
i i i n n n n n n o o o
i i i n n n n n n o o o
i i i n n n n n n o o o
i i i n n n n n n o o o
n n n n n n n n n o o o
n n n i i i i i i n n n
n n n i i i i i i n n n
n n n i i i i i i n n n

Рис.3.4. Параметри матриці зв’язків між нейронами моделі на основі ШНМ

Для представлення матриці зв’язків між нейронами, використовується матриця розміром $(N+1)\times(N+1)$. Кожний елемент матриці може приймати одне з трьох можливих значень: “*n*” – зв’язку немає; “*o*” – вихідне ребро; “*i*” – вхідне ребро. Приклад відповідної матриці зображено на рис. 3.4.

Для представлення матриці значень (величин) ваг зв’язків, використовується матриця суміжності. Кожен елемент є дійсною величиною. У випадку, коли елемент даної матриці дорівнює нулеві, тоді зв'язок відсутній. Приклад матриці суміжності зі значеннями ваг зв’язків, яку отримали в результаті навчання ШНМ, наведено в додатку Е. Графічне представлення структури моделі на основі ШНМ з використанням багатошарового перцептрона зображено на рис. 3.5.

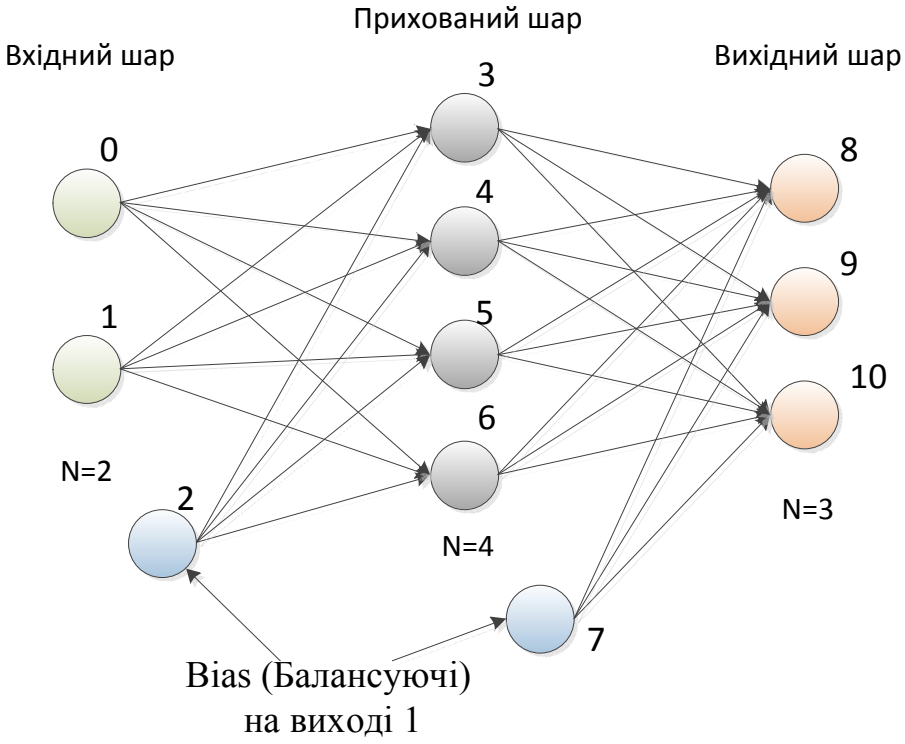


Рис.3.5. Графічне представлення структури моделі на основі ШНМ з використанням багатошарового перцептрона

3.1.2. Розроблення структури та алгоритму роботи нейроконтролера

Фізична реалізація нейроконтролера передбачає використання мікроконтролера, який буде виконувати одну і ту ж програму в циклі, тому алгоритм роботи має виконуватись необмежену кількість ітерацій. Програмна

реалізація симулює роботу штучної нейронної мережі типу багатошарового перцептрона. Розроблений у роботі алгоритм включає кроки, які наведено нижче.

Крок 1: Зчитування вхідних даних від давачів.

Крок 2: Ініціалізація змінних.

Крок 3: i = 0 (початок циклу).

Крок 4: i < netSize ?, так – перехід на крок 5, ні – перехід на крок 17.

Крок 5: vin[i] = 0.

Крок 6: i <cntIn ?, так – перехід на крок 7; ні – перехід на крок 8.

Крок 7: vout[i] = input[i], перехід на крок 16.

Крок 8: balance[i] == 1?, так – перехід на крок 9, ні – перехід на крок 10.

Крок 9: vout[i] = 1, перехід на крок 16.

Крок 10: j = 0.

Крок 11: j < netSize ?, так – перехід на крок 12, ні – перехід на крок 15.

Крок 12: type[i][j]==’i’ ?, так – перехід на крок 13, ні – перехід на крок 14.

Крок 13: vin+=vout[j]*value[j][i].

Крок 14: j++, перехід на крок 11.

Крок 15: vout[i] = calc(i).

Крок 16: i++, перехід на крок 4.

Крок 17: first_out = netSize – cntOut.

Крок 18: i = first_out.

Крок 19: i<netSize ?, так – перехід на крок 20, ні – перехід на крок 22.

Крок 20:output[i-first_out] = vout[i].

Крок 21: i++, перехід на крок 19.

Крок 22: виведення результатів.

Розроблена структура нейроконтролера складається з двох взаємопов’язаних рівнів реалізації (рис.3.7): апаратної та програмної.

Апаратна реалізація (модель) містить у собі:

– підсистему роботи з виконуючими пристроями, які відповідають за виведення значень аналогових сигналів;

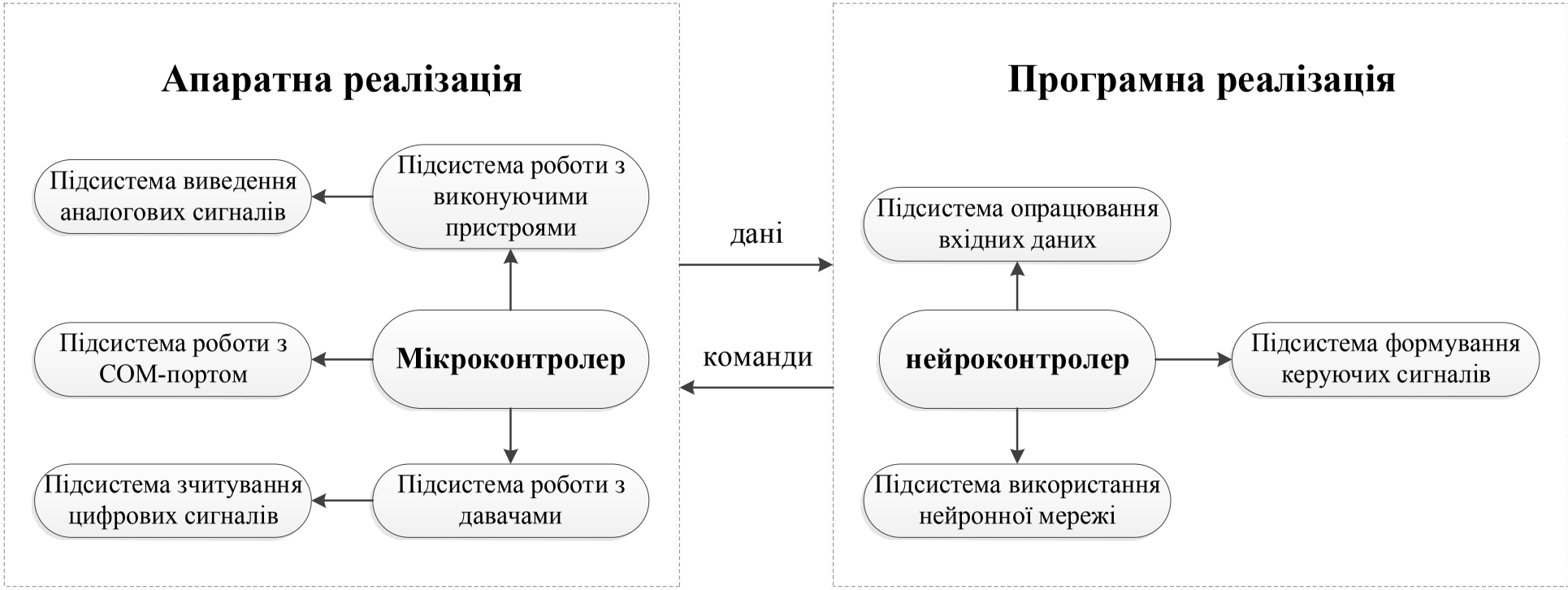


Рис.3.6. Розроблена структура нейроконтролера

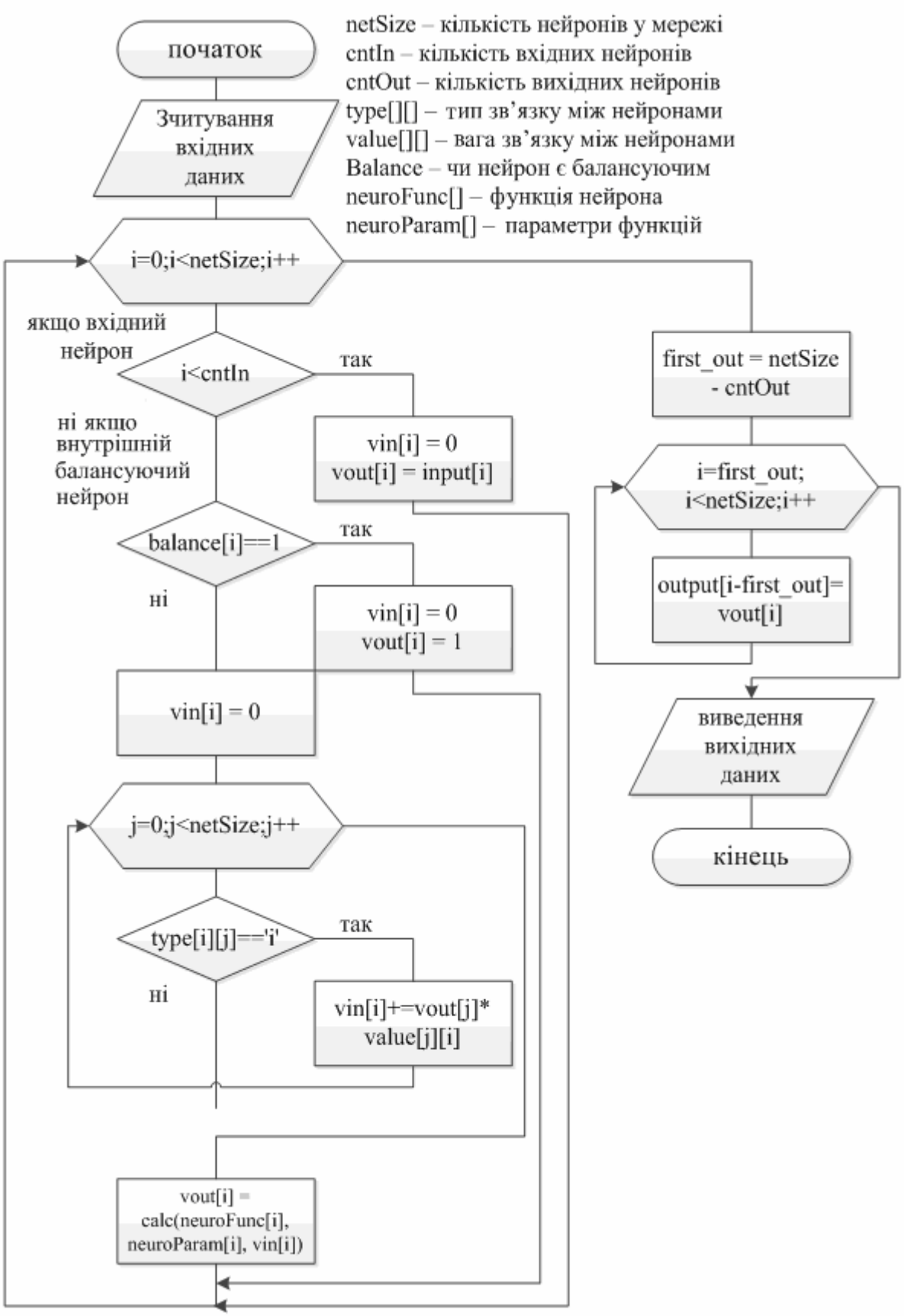


Рис.3.7. Блок-схема алгоритму роботи програми, яка реалізує ШНМ типу багатошарового перцептрона

– підсистему роботи з давачами, яка відповідає за зчитування цифрових сигналів;

– підсистему роботи з послідовним портом, яка відповідає за надсилання повідомлень, проміжних і остаточних результатів до комп'ютера.

Програмна реалізація (модель) містить у собі:

– підсистему опрацювання вхідних даних, отриманих від давачів;

– підсистему формування керуючих сигналів, яка на основі отриманих результатів від нейронної мережі формує команди для виконання мікроконтролером;

– підсистему використання нейронної мережі, яка вносить дані в мережу, запускає мережу і отримує результати.

В даній системі штучна нейронна мережа реалізується програмно.

На рис. 3.7 зображено блок-схему алгоритму, який програмно реалізує ШНМ типу багатошарового перцептрона.

Як було зазначено вище, нейроконтролер виконує в циклі одну і ту ж програму (рис. 3.8) і алгоритм виконання програми є наступним:

Крок 1: Зчитування аналогового сигналу від давачів.

Крок 2: Переведення аналогових сигналів у цифрові.

Крок 3: Опрацювання вхідних даних.

Крок 4: Внесення даних на вхідні нейрони.

Крок 5: Запуск нейронної мережі.

Крок 6: Зчитування результатів з вихідних нейронів.

Крок 7: Опрацювання вихідних даних.

Крок 8: Виведення цифрових сигналів управління пристроями.

Крок 9: Затримка, перехід на крок 1.

Припиняється робота нейроконтролера після вимкнення живлення.

3.1.3. Розроблення програмної моделі та особливості програмної реалізації нейроконтролера

Розроблена програмна модель для нейроконтролера містить ряд модулів, які виконуються один раз, до них належить модуль ініціалізації мережі в якому відбувається ініціалізація ваг, зв'язків та параметрів нейронів. Також один раз виконується модуль ініціалізації портів, який встановлює статуси портів і початкові значення. Інша частина програми виконується в циклі та включає ряд модулів (рис. 3.9). В процесі реалізації використано модульний підхід, що дає змогу швидко модифікувати систему при її вдосконаленні.

Програмне забезпечення розбито на ряд окремих підзадач таких, як:

– зчитування даних – опрацювання даних від давачів, перевірка справності давача, приведення даних до необхідних форматів;

– нормалізація даних – організація даних у відповідності з навчальною вибіркою мережі;

– запуск мережі – внесення даних в мережу і проходження даних через мережу;

– виведення результатів – отримання результатів з мережі і вивід їх через послідовний порт;

– формування керуючих команд – в залежності від отриманих результатів формування сигналів на вихідних портах;

– формування затримки – створення часової затримки перед проходженням наступного циклу роботи програми.

Приклад коду програми, яка реалізує функції нейроконтролера для підсистеми клімат-контролю наведено в додатку А.

Рис.3.8. Блок-схема алгоритму роботи нейроконтролера

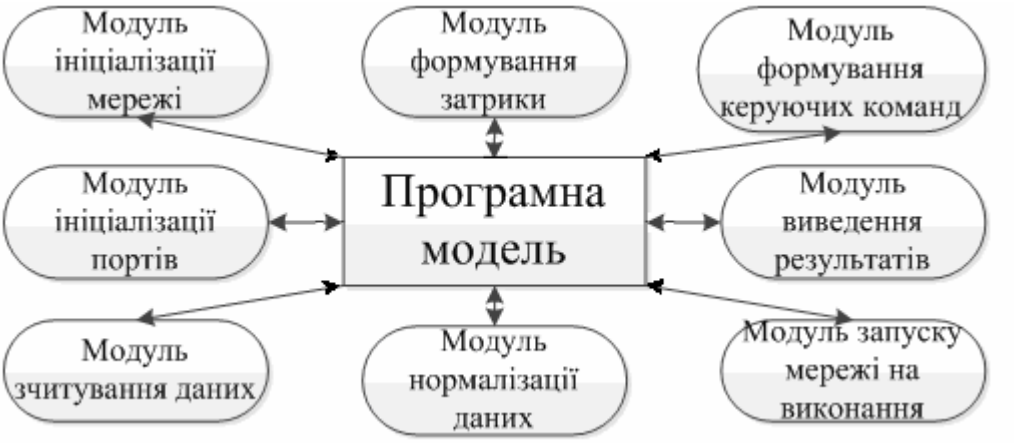


Рис.3.9. Структура програмної моделі нейроконтролера

Використання модульного підходу до розроблення програмного забезпечення дає змогу швидко та ефективно вдосконалювати програмну частину нейроконтролера.

3.2. Розроблення нейроконтролера управління підсистемою освітлення “інтелектуального будинку”

Однією з найважливіших підсистем “інтелектуального будинку” є підсистема освітлення, що передбачає не тільки створення комфортних умов у будинку, але й значну економію споживання електроенергії, а також забезпечує повну автоматизація управління освітленням.

3.2.1. Розроблення сценаріїв роботи нейроконтролера підсистеми освітлення

Для реалізації підсистеми освітлення аналогічно використовуємо нейроконтролери. Особливості проектування нейроконтролера для ІБ описано вище. Розглянемо, як приклад, розроблення нейроконтролера для керування освітленням двохкімнатної квартири. Припустимо, що план квартири відповідає зображеному на рис.3.10, тобто квартира складається з шести окремих приміщень.

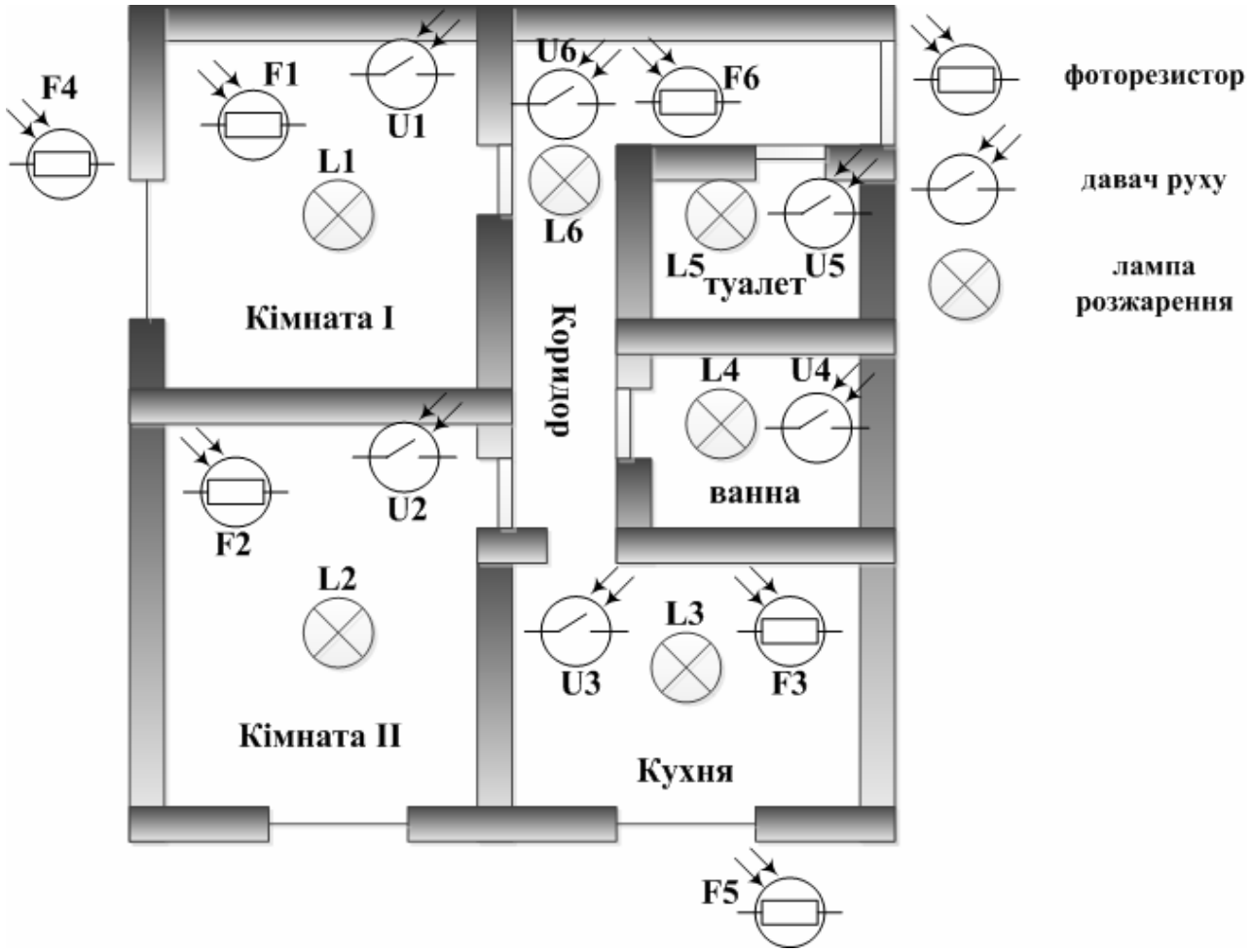


Рис.3.10. Схема розміщення давачів у приміщенні

Наведена структура мікроконтролера аналогічна до структури підсистеми клімат-контролю, яка передбачає використання нейроконтролера, що складається з апаратної і програмної частин. Всередині та за межами квартири розміщені давачі. При цьому, використовуються найпоширеніші 2 типи давачів: давач руху та давач освітлення (фоторезистор).

Припустимо, що у кожному приміщенні розміщено виконавчий пристрій – лампу. В залежності від побажань власника та необхідної функціональності, яку має реалізувати нейроконтролер кількість давачів та їх види можна збільшити.

Отже, в кожному приміщенні є давач руху, який реєструє факт перебування людини в даній кімнаті (рис. 3.11). В кімнатах, кухні та коридорі додатково встановлено фоторезистори для визначення освітленості у приміщенні. Також 2

фоторезистори розміщено з боків будинку ззовні, для того, щоб визначати рівень освітлення наявне на вулиці (час доби).

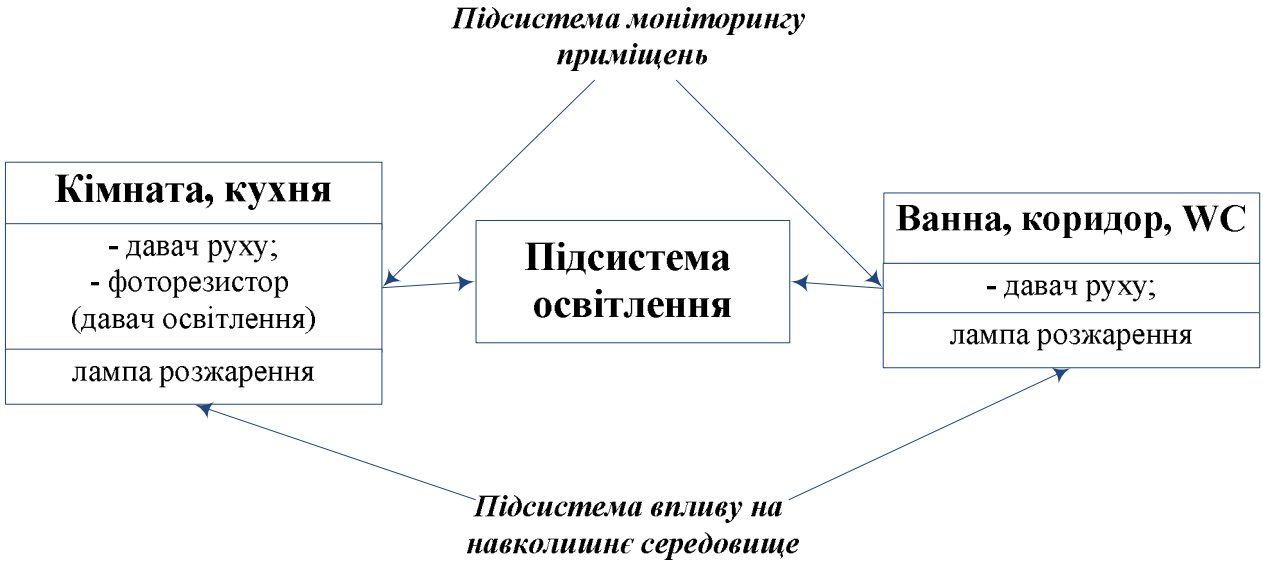


Рис.3.11. Складові підсистема освітлення

Для спрощення організації підсистеми варто виокремити подібні функціональні елементи. Приміщення можна згрупувати за їх функціональним призначенням. Оскільки, у кімнатах та кухні мешканці проводять більшу частину свого часу, то, відповідно, світло там ввімкнено триваліше.

Припустимо, що сценарій роботи системи відповідає наведеному в таблиці 3.1. Фоторезистори $F_{внутрішнє} = F_1$ і $F_{зовнішнє} = F_2$ видають сигнали: 0 – недостатньо освітлення; 1 – достатньо.

Давач руху U видає сигнал: 1 – є рух; 0 – немає руху.

Лампа L має стани: 1 – включено; 0 – виключено.

Функція L обчислюється за наступною булевою формулою.

$$L = U_1 \cdot \overline{F_2} + \overline{F_1} \cdot F_2 \ , \qquad (3.5)$$

Для приміщень які знаходяться в середині квартири та ізольовані від доступу сонячного світла, функція L обчислюється за наступною булевою формулою:

$$L = U \ , \qquad (3.6)$$

Таблиця 3.1. Сценарій роботи підсистеми у кімнатах і кухні

$F_{внутрішнє}$	$F_{зовнішнє}$	U	L
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

Сценарій роботи підсистеми в інших приміщеннях відповідає наведеному в табл. 3.2.

Таблиця 3.2. Сценарій роботи підсистеми в ізольованих приміщеннях

U	L
0	0
1	1

Описані сценарії функціонування підсистеми освітлення ІБ необхідно реалізувати на основі нейроконтролерів.

3.2.2. Розроблення програмної моделі нейроконтролера

Важливим елементом будь якої сучасної технічної системи є спеціалізоване програмне забезпечення.

В процесі реалізації програмної моделі нейроконтролера для розроблення підсистеми навчання нейромережі використано мову високого рівня програмування Java, яка має ряд переваг порівняно з іншими [186]: безпечність, ефективність, об’єктно-орієнтована спрямованість, стійкість до помилок, підтримка багатопоточності, незалежність від архітектури, переваги

інтерпретованості в поєднанні з високою продуктивністю, розподіленість, доступність інструментарію та ефективність розробок.

Структура розробленої програмної моделі нейроконтролера має три основні модулі (рис. 3.12):

- модуль контролю змін в зовнішньому середовищі;
- модуль опрацювання (нейромережа);
- модуль впливу на довкілля.

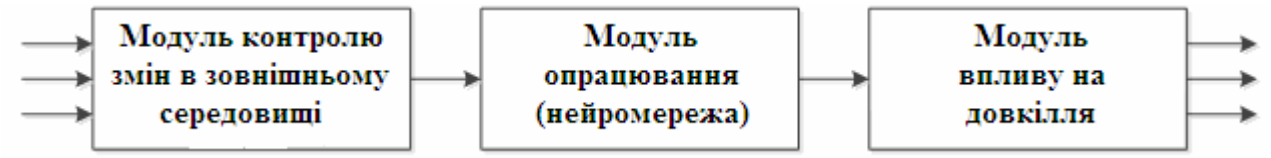


Рис.3.12. Структура програмної моделі нейроконтролера

Перший модуль призначено для реєстрації змін в довкіллі (на основі вхідних даних від давачів руху і фоторезисторів).

Другий модуль призначений для опрацювання даних та прийняття рішення за допомогою нейромережі, яка програмно реалізована і розміщена в пам’яті мікроконтролера.

Третій модуль призначений для внесення змін в довкілля (вмикання ламп тощо).

Діаграма зв’язків та класів програмного забезпечення нейроконтролера освітлення “інтелектуального будинку” наведена на рис. 3.13.

Відповідно, розроблене ПЗ дає змогу реалізувати ШНМ, забезпечує перенесення ПЗ на інші платформи та гарантує задану швидкодію. Приклад коду програми наведено в додатку Б.

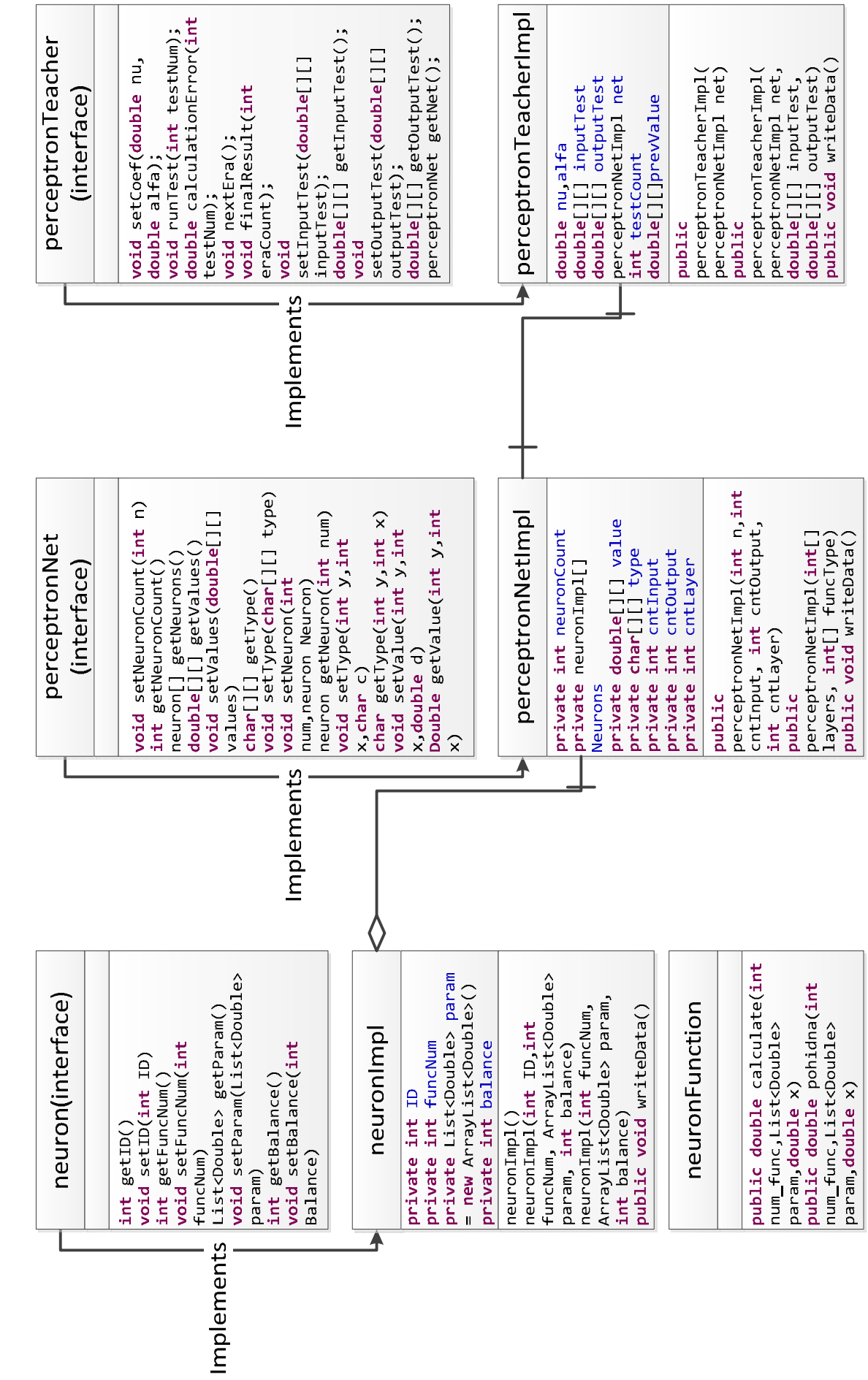


Рис. 3.13. Діаграма зв'язків класів програмної моделі нейроконтролера

3.2.3. Розроблення моделі для опрацювання нечітких даних від давачів підсистеми освітлення “інтелектуального будинку”

В системі “інтелектуального будинку” маємо справу з нечіткими даними.

Побудована модель опрацювання нечітких даних від давачів для підсистеми освітлення “інтелектуального будинку” ґрунтується на використанні штучної нейронної мережі. В даному випадку, як і для підсистеми клімат-контролю, використано ШНМ типу багатошарового перцептрона. В ході розроблення моделі відповідна ШНМ була розроблена та навчена з допомогою програмної системи, яка детально описана в четвертому розділі дисертаційної роботи і отримані проміжні та остаточні результати. Усі результати виводяться у консоль. Структура виведених даних системою є наступною:

– середньоквадратична похибка на останній епосі навчання;

– загальна кількість нейронів;

– кількість вхідних нейронів;

– кількість вихідних нейронів;

– перелік усіх нейронів (ідентифікатор нейрона, тип передавальної функції, чи балансуючий нейрон, кількість параметрів передавальної функції);

– ваги зв’язків між нейронами представлені у вигляді матриці суміжності, розміри якої рівні neuronCount * neuronCount;

– типи зв’язку між нейронами представлені у вигляді матриці суміжності, розміри якої рівні neuronCount * neuronCount.

Зокрема результати навчання розробленої штучної нейронної мережі та основні пояснення наведено на рис.3.14, рис.3.15 та в додатку З.

Після навчання мережі необхідно перевірити точність навчання, тобто наскільки добре мережа здатна виконувати поставлене завдання. Щоб перевірити коректність мережі, необхідно отримати вихідні значення на наборі тестових прикладів. Розроблену штучну нейронну мережу протестовано і отримано результати, які зображено на рис. 3.16 та рис. 3.17.

Структура отриманих вихідних результатів наступна:

– *n* блоків з наступною структурою (*n* – кількість навчальних тестів): вхідні значення, вихідний результат;

– *n* рядків з необхідними кінцевими значеннями.

З отриманих даних бачимо, що побудована штучна нейронна мережа емулює вихідні параметри з заданою точністю. Як видно з отриманих результатів, вихідні дані не відхиляються від реальних результатів більш ніж на 0.02, тобто максимальна похибка близька до 2%.

9999 Average Error = 0.014791171787897562 //середня похибка
Perceptron Teacher
PerceptronNet
neuronCount: 14 //кількість нейронів
InputCount = 3 //кількість вхідних нейронів
OutputCount = 1 //кількість вихідних нейронів
Neuron #0 funcNum = 1 balance = 1 count of parameters = 1 //ідентифікатор нейрона, тип передавальної функції, чи балансуючий нейрон, кількість параметрів.
Neuron #1 funcNum = 2 balance = 0 count of parameters = 1
Neuron #2 funcNum = 2 balance = 0 count of parameters = 1
Neuron #3 funcNum = 2 balance = 0 count of parameters = 1
Neuron #4 funcNum = 1 balance = 1 count of parameters = 1
Neuron #5 funcNum = 2 balance = 0 count of parameters = 1
Neuron #6 funcNum = 2 balance = 0 count of parameters = 1
Neuron #7 funcNum = 2 balance = 0 count of parameters = 1
Neuron #8 funcNum = 2 balance = 0 count of parameters = 1
Neuron #9 funcNum = 1 balance = 1 count of parameters = 1
Neuron #10 funcNum = 2 balance = 0 count of parameters = 1
Neuron #11 funcNum = 2 balance = 0 count of parameters = 1
Neuron #12 funcNum = 1 balance = 1 count of parameters = 1
Neuron #13 funcNum = 2 balance = 0 count of parameters = 1

Рис.3.14. Властивості розробленої нейронної мережі для підсистеми освітлення

Якщо для розв’язування завдання використовувати порогову функцію з порогом 0.5, то ці відхилення не впливають на остаточні результати.

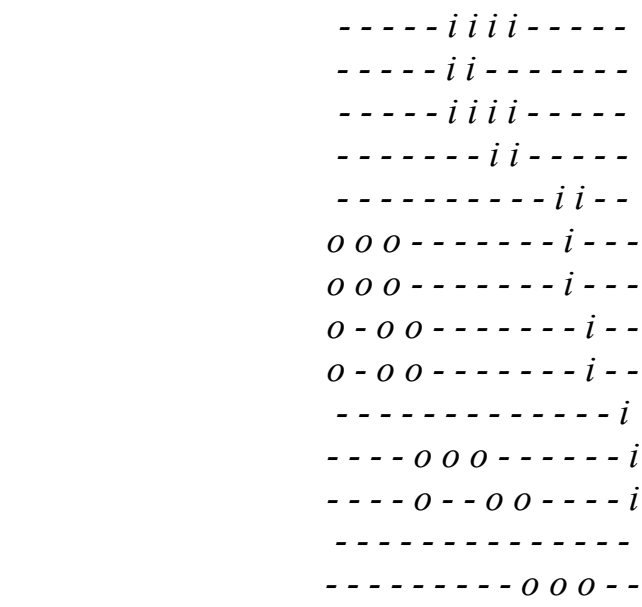


Рис. 3.15. Типи зв’язків між нейронами розробленої ШНМ

Input Tests
0.0 0.0 0.0 //вхідні значення
Run test: y[0] = 0.022998815821855253 //вихідний результат
0.0 0.0 1.0 //вхідні значення
Run test: y[0] = 0.9914181945113373 //вихідний результат
0.0 1.0 0.0 //вхідні значення
Run test: y[0] = 0.9825204241667216 //вихідний результат
0.0 1.0 1.0 //вхідні значення
Run test: y[0] = 0.9884257410224 //вихідний результат
1.0 0.0 0.0 //вхідні значення
Run test: y[0] = 0.01074182505350505 //вихідний результат
1.0 0.0 1.0 //вхідні значення
Run test: y[0] = 0.9815800076959352 //вихідний результат
1.0 1.0 0.0 //вхідні значення
Run test: y[0] = 0.011366077788593377 //вихідний результат
1.0 1.0 1.0 //вхідні значення
Run test: y[0] = 0.01716702303562091 //вихідний результат

Рис. 3.16. Приклади вхідних тестів

Output Tests //справжні вихідні значення
0.0
1.0
1.0
1.0
0.0
1.0
0.0
0.0

Рис.3.17. Результати тестування розробленої ШНМ

Приклад розробленої структури штучної нейронної мережі зображено на рис. 3.18.

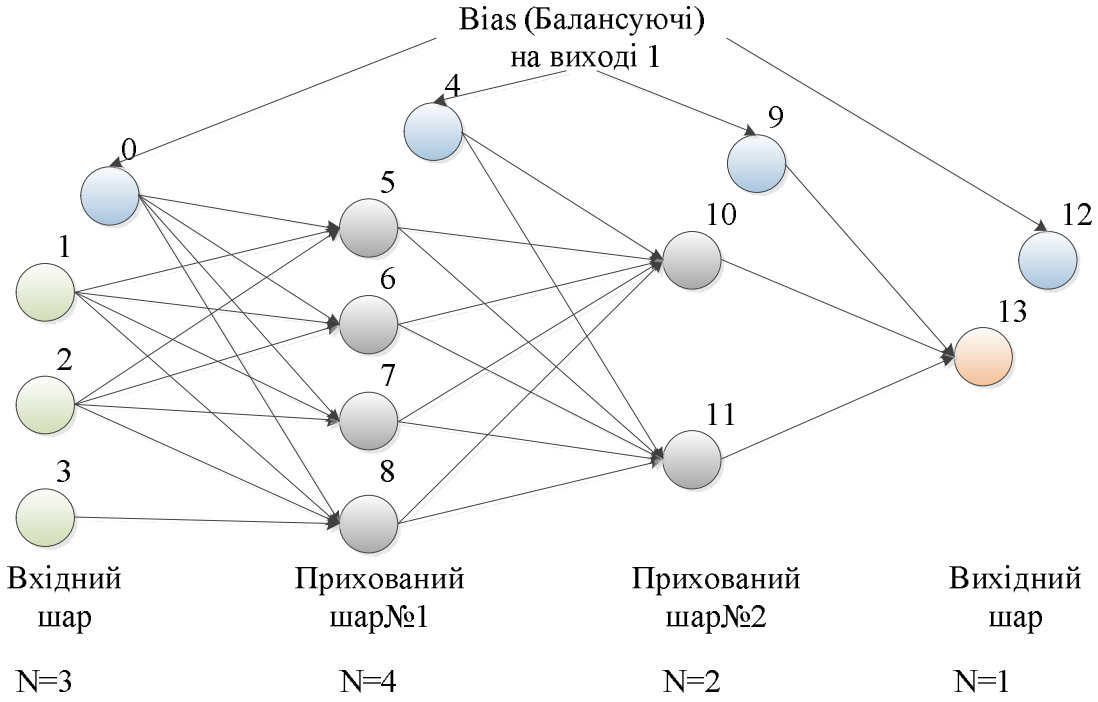


Рис. 3.18. Розроблена структура ШНМ для моделі нечітких даних від давачів підсистеми освітлення ІБ

Таким чином розроблено модель для опрацювання нечітких даних для підсистеми освітлення “інтелектуального будинку”.

3.3. Розроблення нейроконтролера управління підсистемою захисту “інтелектуального будинку”

Важливою складовою “інтелектуального будинку” є забезпечення охорони майна власника. Для реалізації такої функції в роботі розроблена підсистема захисту, яка призначена для забезпечення захисту приміщень від несанкціонованого доступу, а також захисту життя, здоров’я та майна користувача, спричинених небажаним проникненням сторонніх осіб в межі ІБ.

3.3.1. Розроблення структури нейроконтролера для підсистеми захисту “інтелектуального будинку”

В дисертаційній роботі розроблена структура нейроконтролера підсистеми захисту “інтелектуального будинку”, яка наведена на рис. 3.19. Нейроконтролер побудований для керування підсистемою захисту “інтелектуального будинку”. Він симулює попередньо навчену нейронну мережу на основі багатошарового перцептрона. В даному випадку, використано один нейроконтролер для реалізації цієї підсистеми.

У випадку, коли один мікроконтролер не зможе реалізувати потрібні функції з заданою швидкодією, можна використати структуру реалізації підсистеми захисту, яка зображена на рис. 3.19. Відповідне рішення залежить від потужності використаного мікроконтролера та об’єму його пам’яті, в яку необхідно завантажити спеціалізоване програмне забезпечення, що реалізує виконання штучної нейтронної мережі.

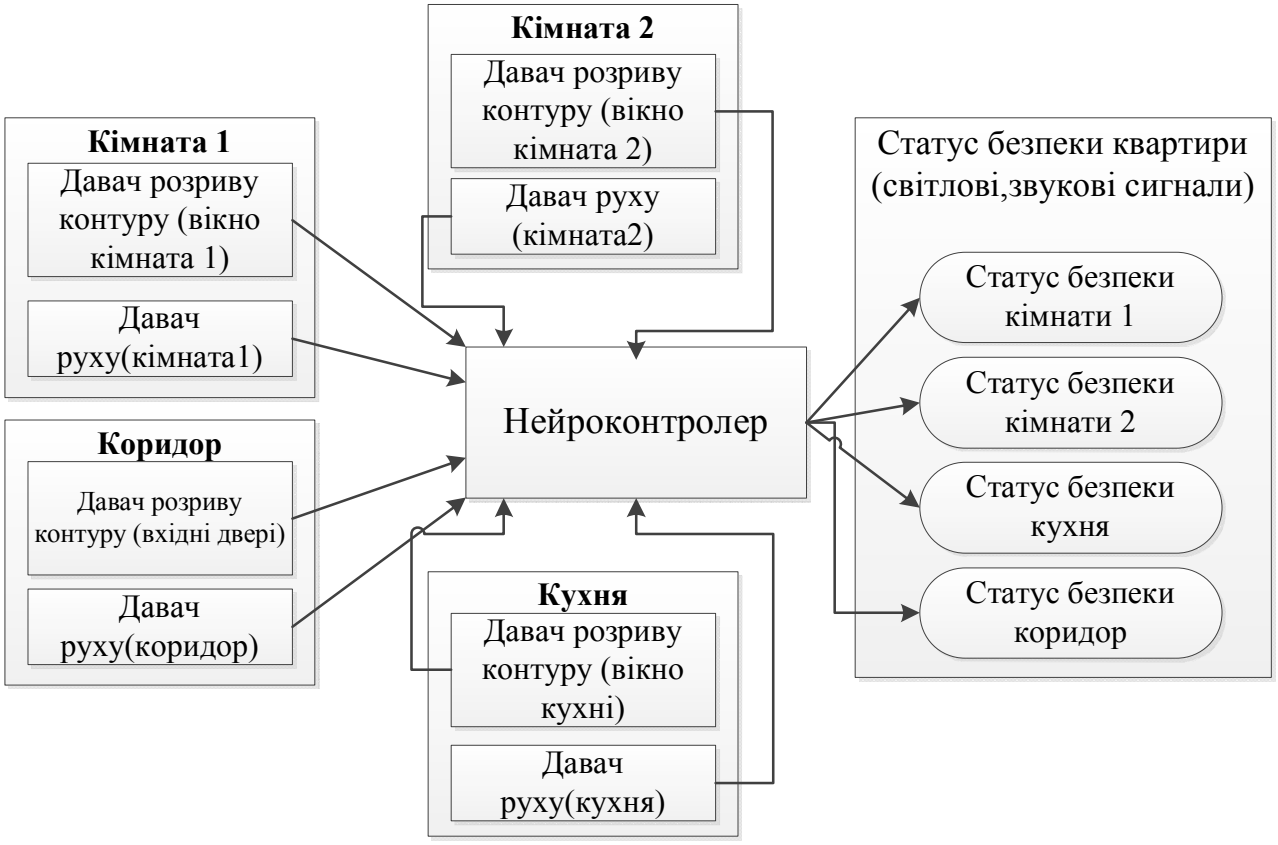


Рис.3.19. Структура нейроконтролера для підсистеми захисту ІБ

Структура та алгоритми роботи нейроконтролера детально розглянуті в параграфі 3.1 даного дисертаційного дослідження.

Розроблена структура включає нейроконтролер, набір давачів про стан приміщень та засоби впливу на приміщення “інтелектуального будинку”.

В наведеному прикладі використано один мікроконтролер і, відповідно, така структура є, порівняно, дешевою. Хоча, для підвищення надійності системи, можна використовувати декілька мікроконтролерів (як у випадку реалізації підсистеми клімат-контролю). Більше того, використання декількох мікроконтролерів дає змогу підвищити функціональність підсистеми безпеки.

3.3.2. Розроблення моделі для опрацювання нечітких даних від давачів підсистеми захисту “інтелектуального будинку”

Аналогічно до розроблених вище нейроконтролерів, для опрацювання нечітких даних від підсистеми давачів, використано модель на основі штучної нейронної мережі. Штучна нейронна мережа реалізована програмним чином і записана у пам'ять мікроконтролера. Отже, програмна реалізація ШНМ використовує багатошаровий перцептрон і містить 3 шари з 16 нейронами.

Побудована мережа містить 8 вхідних нейронів і 1 вихідний нейрон. Кількість шарів і нейронів у шарах визначалась експериментально та так, щоб наочно було видно статус в кожній із областей, щоб після навчання похибка результатів була прийнятною. Найкращим отриманим варіантом є мережа з одним внутрішнім шаром на 4 нейрони, кожний з яких відповідає за одну із областей (кімната 1, кімната 2, кухня, коридор). Також, в процесі практичної реалізації підсистеми, необхідно 3 балансуючі нейрони для кожного із шарів.

В результаті навчання з допомогою програмних засобів, які детально описано в 4-му розділі дисертаційного дослідження, було отримано мережу з наступними властивостями, які зображено на рис. 3.20, на рис. 3.21 та в додатку И, а структура розробленої штучної нейронної мережі в графічному вигляді зображена на рис. 3.22.

Neuron #0 funcNum = 1 balance = 1 count of parameters = 1
Neuron #1 funcNum = 2 balance = 0 count of parameters = 1
Neuron #2 funcNum = 2 balance = 0 count of parameters = 1
Neuron #3 funcNum = 2 balance = 0 count of parameters = 1
Neuron #4 funcNum = 2 balance = 0 count of parameters = 1
Neuron #5 funcNum = 2 balance = 0 count of parameters = 1
Neuron #6 funcNum = 2 balance = 0 count of parameters = 1
Neuron #7 funcNum = 2 balance = 0 count of parameters = 1
Neuron #8 funcNum = 2 balance = 0 count of parameters = 1
Neuron #9 funcNum = 1 balance = 1 count of parameters = 1
Neuron #10 funcNum = 2 balance = 0 count of parameters = 1
Neuron #11 funcNum = 2 balance = 0 count of parameters = 1
Neuron #12 funcNum = 2 balance = 0 count of parameters = 1
Neuron #13 funcNum = 2 balance = 0 count of parameters = 1
Neuron #14 funcNum = 1 balance = 1 count of parameters = 1
Neuron #15 funcNum = 2 balance = 0 count of parameters = 1

Рис.3.20. Параметри розробленої штучної нейронної мережі

Type Matrix (матриця зв'язків між нейронами (n – немає зв'язку, o – вихідне ребро, i – вхідне ребро)

n n n n n n n n n n i i i i n n
n n n n n n n n n n i i i i n n
n n n n n n n n n n i i i i n n
n n n n n n n n n n i i i i n n
n n n n n n n n n n i i i i n n
n n n n n n n n n n i i i i n n
n n n n n n n n n n i i i i n n
n n n n n n n n n n i i i i n n
n n n n n n n n n n i i i i n n
n n n n n n n n n n i i i i n n
n n n n n n n n n n n n n n n i
o o o o o o o o o n n n n n n i
o o o o o o o o o n n n n n n i
o o o o o o o o o n n n n n n i
o o o o o o o o o n n n n n n i
n n n n n n n n n n n n n n n n
n n n n n n n n n o o o o o n n

Рис.3.21. Матриця зв'язків між нейронами

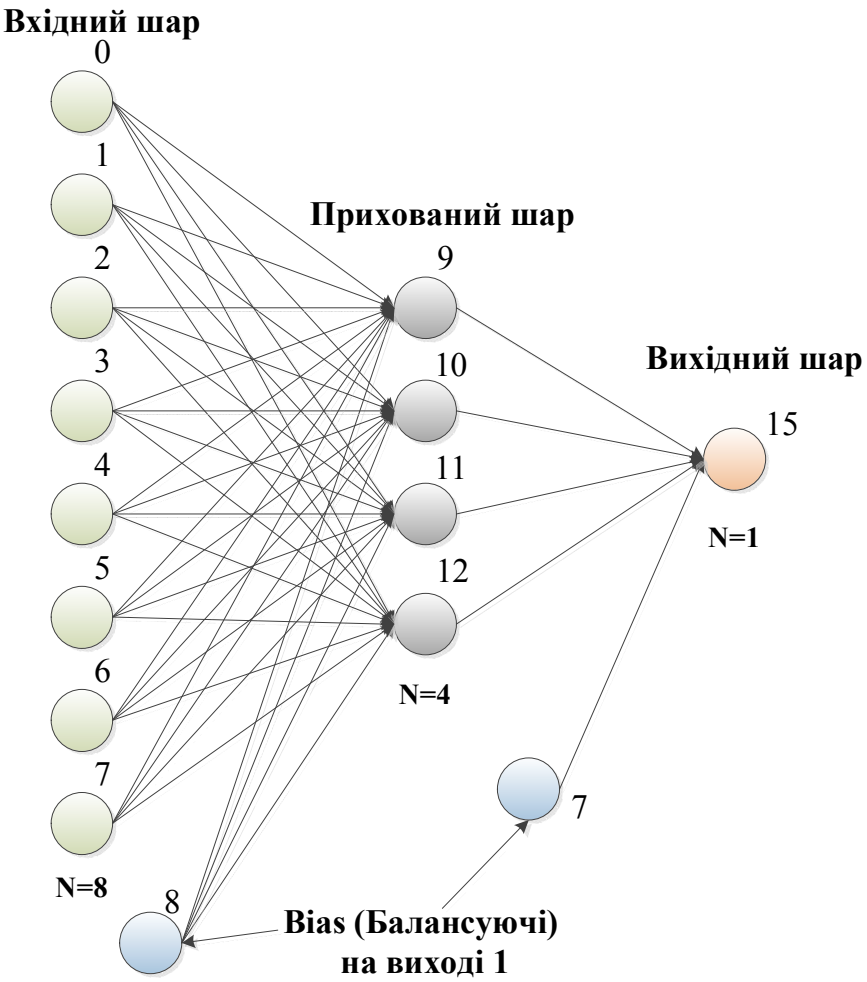


Рис. 3.22. Структура мережі багатошарового перцептрона

Код програми, яка реалізує функції нейроконтролера для підсистеми захисту ІБ наведено у додатку В.

3.4. Розроблення нейроконтролера для підсистеми запобігання технічних аварій “інтелектуального будинку”

У функції “інтелектуального будинку” включено підсистему запобігання технічним аваріям (ЗТА), яка розпізнає такі небезпечні ситуації і, відповідно по можливості, їх усуває. До таких небезпечних ситуацій належать протікання води у туалеті, ванні чи на кухні; витік газу, поява пожежі та ін.

Для реалізації підсистеми ЗТА – побудовано функціональну мережу, яка реалізована на основі штучної нейронної мережі.

Штучна нейронна мережа, яка використовується для управління підсистемою ЗТА, навчена зчитувати дані від давачів та реагувати на них у відповідності до сценаріїв, які наведено в розділі 2 дисертаційного дослідження. Використана штучна нейронна мережа типу багатошарового перцептрона. Структуру розробленої моделі штучної нейронної мережі підсистеми ЗТА зображено на рис. 3.23.

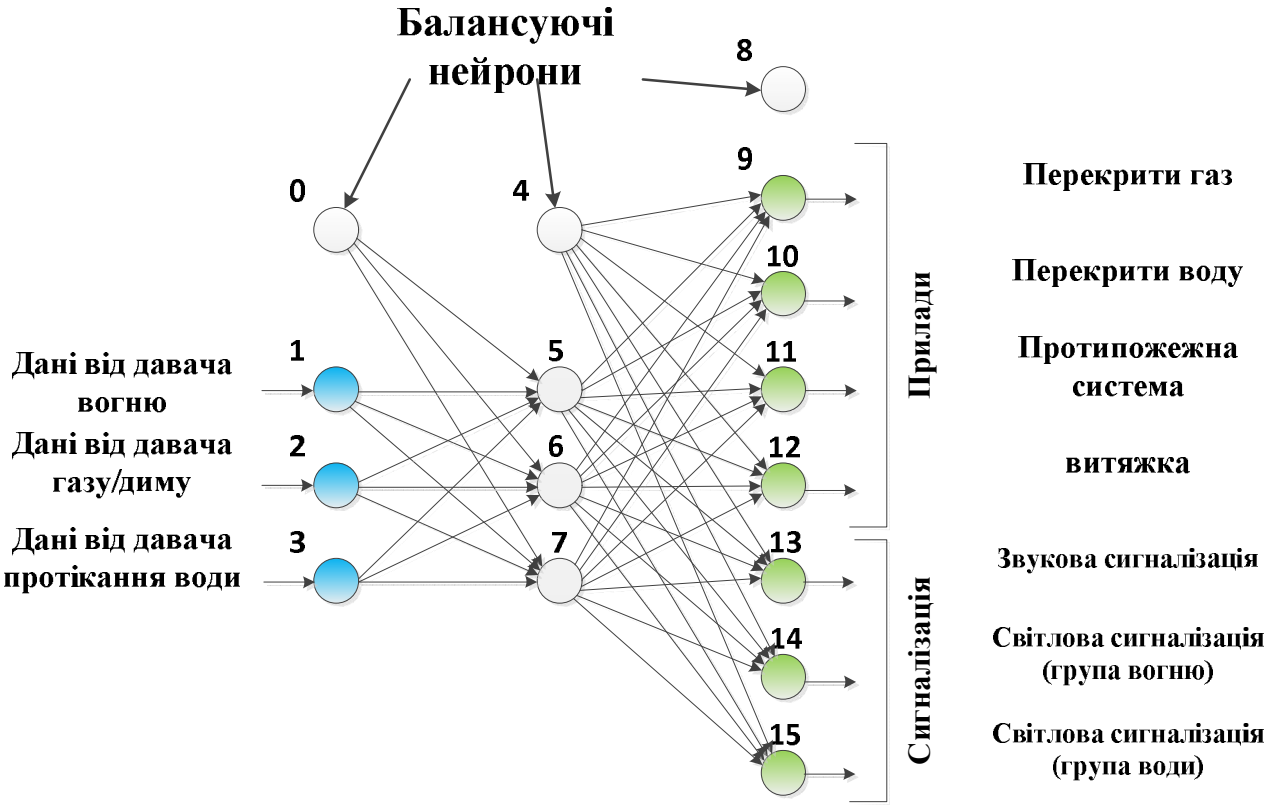


Рис. 3.23. Структура штучної нейронної мережі підсистеми ЗТА

Нейронна мережа складається із трьох шарів, кожний з який виконує призначену функцію. Перший шар є вхідним і містить нормалізовані дані від давачів, та складається із трьох нейронів і одного балансуючого. Кожний із вхідних нейронів відповідає одному із давачів. Другий шар є проміжний і у ньому відбувається класифікація подій, які відбулись. В даному випадку, маємо 3 різні класи подій. Третій шар є вихідним і у ньому відбувається вивід керуючих команд для активаторів. Кожний із нейронів вихідного шару відповідає одному із активаторів підсистеми.

Аналогічно до попередніх підсистем, підсистему ЗТА можна реалізувати як набір нейроконтролерів, що обумовлено необхідними параметрами технічної та програмної реалізації.

Сформувавши структуру нейронної мережі було розроблено навчальні тести згідно із вищенаведеними сценаріями роботи (табл. 3.3).

Таблиця 3.3. Навчальні тести для нейронної мережі

Вхід	Вихід
{0,0,0}	{0,0,0,0,0,0,0}
{0,0,1}	{0,1,0,0,1,0,1}
{0,1,0}	{1,0,1,0,1,1,0}
{1,0,0}	{1,0,1,0,1,1,0}

Використовуючи отримані тести, було навчено нейронну мережу. Навчання відбувалось у 3 етапи: спочатку навчались 1 та 2 шари нейронної мережі (додаток К), щоб нейронна мережа правильно класифікувала події, які виникнули; потім навчалися 2 та 3 шари нейронної мережі (додаток Л), щоб нейронна мережа правильно вмикала активатори згідно із сценарієм роботи; після того було об’єднано усі шари нейронної мережі та проведено фінальне навчання нейтронної мережі на тестах записаних в таблиці 3.3.

Значення отриманих коефіцієнтів наведено в додатках К-М, а результати тестування в додатку Н. Отримані результати тестування підтверджують, що розроблена нейронна мережа працює правильно та коректно. Отже, побудована модель дає змогу опрацювати нечіткі вхідні дані в процесі роботи підсистеми ЗТА. Приклад розробленого спеціалізованого програмного забезпечення наведено в додатку Д.

3.5 Висновки до розділу 3

1. Розроблено модель підсистеми клімат-контролю ІБ, яка використовує штучну нейронну мережу на основі багатошарового перцептрона, що дає змогу опрацьовувати нечіткі та неструктуровані дані від підсистеми давачів.

2. Розроблено модель підсистеми освітлення ІБ, яка використовує штучну нейронну мережу на основі багатошарового перцептрона, що дає змогу опрацьовувати нечіткі та неструктуровані дані від підсистеми давачів.

3. Розроблено модель підсистеми захисту ІБ, яка використовує штучну нейронну мережу на основі багатошарового перцептрона, що дає змогу опрацьовувати нечіткі та неструктуровані дані від підсистеми давачів.

4. Розроблено модель підсистеми запобігання технічних аварій ІБ, яка використовує штучну нейронну мережу на основі багатошарового перцептрона, що дає змогу опрацьовувати нечіткі та неструктуровані дані від підсистеми давачів.

5. Розроблено програмне забезпечення нейроконтролерів, яке враховує специфіку та особливості мікроконтролера AVR, використовує мову високого рівня та моделі на базі ШНМ, що дає змогу швидко вносити зміни в функціональність нейроконтролера та забезпечує його низьку вартість.

6. Проведено тестування розроблених моделей для опрацювання нечітких даних з використанням моделей на основі ШНМ та розробленого спеціалізованого програмного забезпечення. Результати тестів дають підставу стверджувати, що побудовані моделі працюють правильно та коректно.

РОЗДІЛ 4. РЕАЛІЗАЦІЯ ЗАСОБІВ АВТОМАТИЗАЦІЇ ПРОЕКТУВАННЯ СИСТЕМ “ІНТЕЛЕКТУАЛЬНОГО БУДИНКУ”

В процесі виконання дисертаційної роботи розроблено засоби автоматизації, які включають два програмних продукти. Перший програмний продукт призначений для автоматизованого синтезу моделей на основі мереж Петрі для системного рівня проектування, а другий – для автоматизованого синтезу штучних нейронних мереж на основі вхідних даних (функцій підсистем “інтелектуального будинку”).

При цьому, розроблені програмні продукти повинні відповідати певним вимогам [187], а саме:

– забезпечувати здатність розширення функціональних можливостей програмної системи при незмінній його структурі;

– стандартизованість і оптимальність структури програмної системи;

– автоматичність виявлення помилок в процесі редагування вхідного завдання;

– гнучкість, що забезпечує можливість вибору користувачем необхідного варіанту розв`язання задачі із набору варіантів, реалізованих в програмній системі;

– здатність роботи програмної системи на машинах різних типів та різних операційних системах;

– чітка орієнтація програмної системи на категорію “користувач – інженерно-технічний персонал”.

Разом з тим, необхідно додати вимогу щодо забезпечення ефективного та швидкого обміну даними про об’єкт розроблення з іншими програмними системами, що є надзвичайно актуальним питанням сьогодення за умов наявності величезної кількості розроблених програмних засобів різного функціонального призначення [188]. Такий підхід не потребує розроблення повністю нового програмного забезпечення (що здебільшого є надзвичайно коштовним), але лише тих модулів та підсистем, які відсутні в існуючих системах і дають змогу розв’язати поставлені задачі.

4.1. Особливості розроблення системи для синтезу моделей на основі мереж Петрі

4.1.1. Розроблення структури системи синтезу моделей на основі мереж Петрі

Один з перших етапів розроблення будь-якого програмного продукту передбачає визначення вимог та побудову структури такої системи, яка включає основні складові та зв’язки між ними [189].

В процесі розроблення системи синтезу моделей на основі мереж Петрі для побудованої структури об’єкту проектування, розроблено структуру програмного засобу (рис. 4.1), яка включає такі основні елементи:

– блок опрацювання даних/команд від користувача, який містить методи для зчитування та опрацювання інформації від користувача;

– конвертор даних, який конвертує дані із внутрішнього формату до формату опису мережі Петрі з використанням мови XML [190];

– блок побудови зв’язків між вершинами, який встановлює зв’язки між вершинами;

– блок розтягування вершин зліва-направо, який визначає наближені значення координат кожної вершини для кращого (рівномірного) візуального сприйняття графічного представлення мережі Петрі;

– блок розташування переходів між вершинами, який визначає координати переходів;

– блок роботи з XML-файлами, який містить методи для зчитування і запису файлів системи у XML-форматі, що забезпечує ефективний обмін даними з іншими програмними системами, які використовуються для опрацювання інформації, записаної в XML-форматі.

Отже, розроблена структура програмної системи використовує модульний принцип [191], який забезпечує швидку модифікацію та вдосконалення програмної

системи в майбутньому, а використання XML-формату для збереження даних про моделі – це зручний обмін інформацією з існуючими програмними системами.

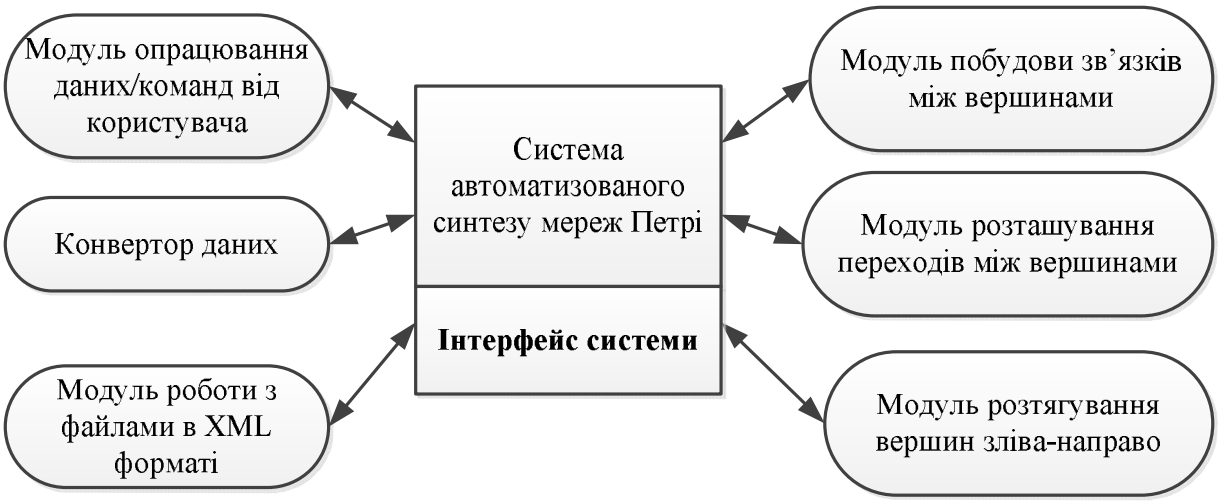


Рис.4.1.Структура системи автоматизованого синтезу моделей системного рівня проектування на основі мереж Петрі

4.1.2. Розроблення основних алгоритмів роботи системи автоматизованого синтезу моделей на основі мереж Петрі

Важливим питанням розроблення ПЗ є побудова алгоритму роботи системи. Зокрема: розроблена система працює за алгоритмом, який зображено на рис. 4.2.

Отже, після запуску системи, якщо мережа була вже набрана чи сформована в іншому редакторі, тоді необхідно:

- завантажити мережу;
- запустити конвертацію вихідної мережі у мережу в XML-форматі;
- після закінчення конвертації необхідно зберегти отриманий результат.

Після виконання даних операцій, можна відкрити мережу в тих відомих середовищах, які передбачають можливість опрацювання інформації в XML-форматі.

Якщо після запуску програми не було введено мережу, тоді необхідно діяти наступним чином:

- створити нову мережу;

Рис.4.2. Блок-схема алгоритму роботи системи

– поки мережа не набрана (не введено всі вершини), виконати наступні операції:

  - ввести вершину;
  - ввести вхідні ребра;
  - ввести вихідні ребра.

– зберегти мережу;

– запустити конвертацію мережі у мережу Петрі з можливістю збереження даних про модель в XML-форматі;

– після закінчення конвертації необхідно зберегти мережу.

Для того, щоб краще візуально сприймати мережу, необхідно правильно розташувати вершини і переходи. Припустимо, що для відображення мережі будемо розташовувати вузли і переходи зліва направо, оскільки це зручно для розробника.

Введемо такі правила для розташування складових мережі Петрі:

1) ті вершини, які є тільки вихідними повинні розміщуватися в крайній лівій колонці;

2) вершини, які відвідуються пізніше повинні розташовуватися правіше;

3) якщо є декілька вихідних, то нижчою буде та, яка оголошена пізніше;

4) якщо є декілька вхідних, то нижчою буде та, яка оголошена пізніше;

5) пошук в ширину, для визначення стовпчика;

6) рядок залежить від порядку розташування у черзі.

Алгоритм автоматизованого розташування вершин і переходів працює наступним чином (рис. 4.3):

Крок 1. Початок.

Крок 2. Ініціалізація початкових значень і параметрів.

Крок 3. Перебір усіх вузлів у пошуку вершин і переходів, які не містять вхідних ребер, усі ці вузли маркуються як відвідувані і вносяться у чергу.

Крок 4. Поки черга містить елементи, то:

– переглядається поточна максимальна дистанція від лівого краю і номер рядка;

– для поточного вузла, який перебуває на початку черги переглянути усі елементи, з якими він з’єднаний через вихідне ребро. Якщо елемент ще не відвідувався, тоді:

- додати елемент у чергу;
- позначити елемент як відвіданий;
- визначити номер стовпчика.

– видалити вузол, який знаходиться на початку черги.

Крок 5. Завершити роботу алгоритму.

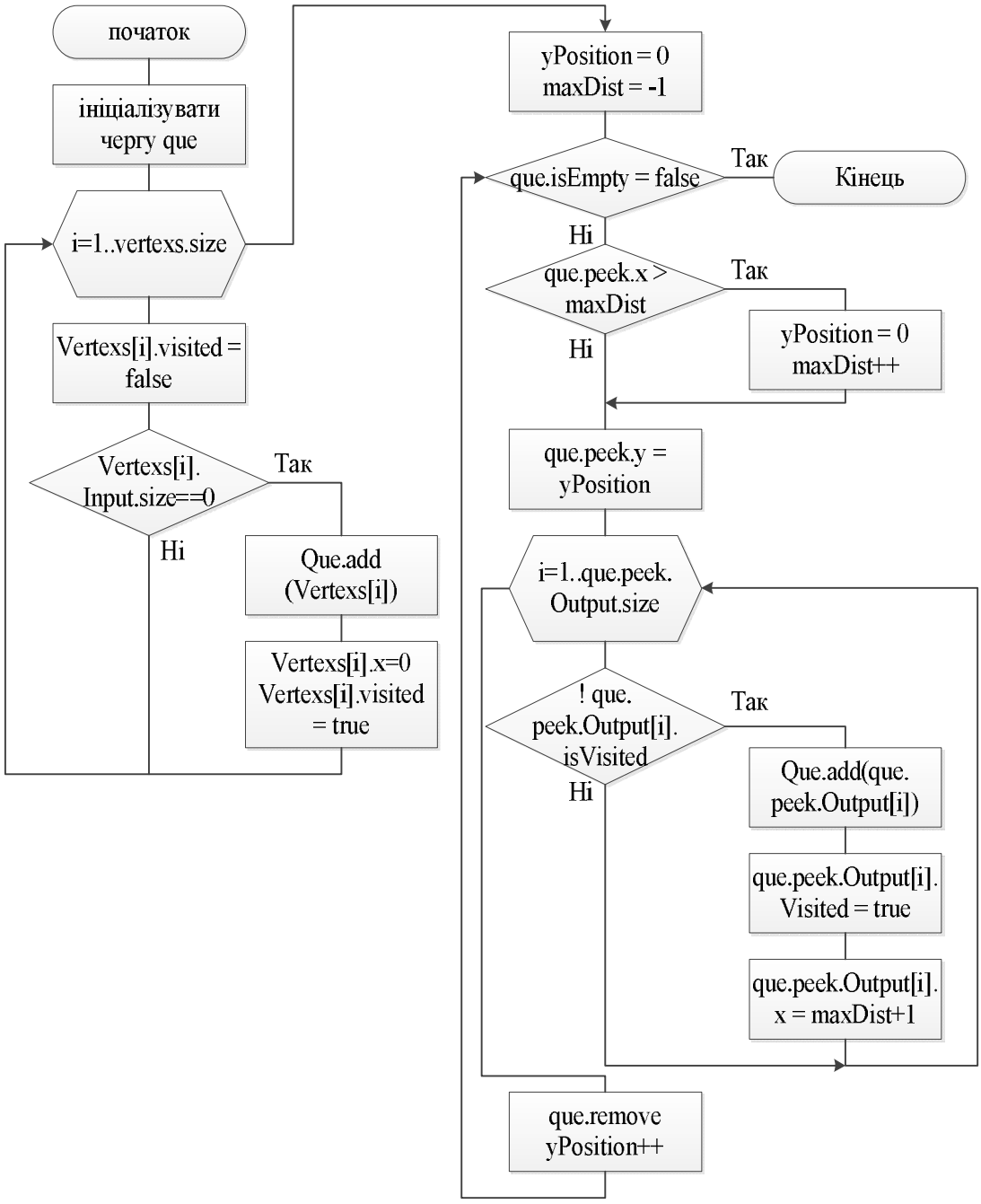


Рис. 4.3 Блок-схема алгоритму розташування вершин і переходів мережі Петрі

4.1.3. Розроблення програмного забезпечення системи для автоматизованого синтезу моделей на основі мереж Петрі

Перші кроки розроблення ПЗ пов’язані з визначенням середовища розроблення, яке визначає характеристики програмного продукту.

Програмний продукт, розроблений в даній роботі, виконано за допомогою об’єктно-орієнтованої мови програмування Java.

Використання Java дає високий рівень гнучкості та виразності. Завдяки об’єктно-орієнтованості, Java надзвичайно добре підходить в якості мови розробки програмного продукту саме для моделювання систем та процесів, адже будь-який об’єкт розроблюваної моделі можна зручно представити як клас Java. До того ж в даному випадку розробник має можливість легко додавати в модель нові класи і розробляти цілі класові бібліотеки [195].

Методологія об’єктно-орієнтованого дизайну, яким володіє мова програмування Java допомагає розділити розроблювану програму на множину об’єктів, модульних підпрограм, вихідний код яких можна легко редагувати чи використовувати повторно.

Для швидшої та надійнішої реалізації програмного забезпечення, необхідно сформувати набір незалежних модулів, кожний з який виконує свої власні функції. Взаємодія між компонентами відбувається через зовнішні інтерфейси модулів. Для задачі синтезу моделей на основі мереж Петрі, програмне забезпечення розбите на наступні підсистеми (рис. 4.4):

– модуль роботи з користувачем, який містить інтерфейс користувача і методи для опрацювання різноманітних сценаріїв роботи;

– модуль роботи з моделлю, що містить клас мережі, який охоплює множину вузлів і ребер, методи для маніпуляції з мережею та обчислення координат вузлів;

– модуль для роботи з файлами, який містить клас, що здатний відкрити, зберегти мережу у вигляді структури мережі і зберегти мережу у форматі відомих середовищ роботи з мережами Петрі.

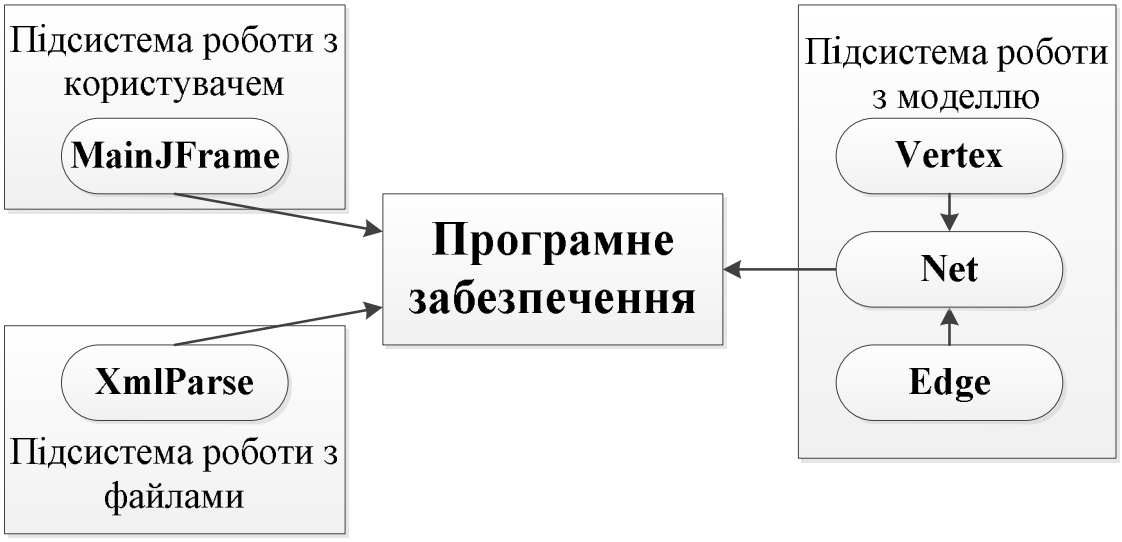


Рис.4.4. Структура розробленого ПЗ системи синтезу моделей на основі мереж Петрі

Розроблене програмне забезпечення системи містить в собі складну структуру класів та об’єктів, що зображено на рис.4.5. Опишемо коротко призначення кожного з розроблених класів.

Клас Vertex – описує властивості вершин (які поділяються на два види) та методи для роботи з ними. Кожна з вершин містить список вхідних та вихідних ребер, назву та тип вершини (позиція чи перехід). Кожна вершина має свою координату, номер та маркування.

Клас Edge – описує ребра між вершинами. Усі вони є орієнтованими. Кожне ребро містить посилання на вхідну та вихідну вершини, вагу, колір та тип. Цей клас містить методи для роботи із властивостями ребер.

Клас Net – описує властивості мережі, яка утворює орієнтований граф. Мережа містить список вершин та ребер, які утворюють граф. Клас містить методи для роботи із мережею.

Клас XmlParse – відповідає за зчитування вхідних файлів і формування мережі, запис мережі у файл для відкривання його у середовищі Pipe 4.1, та сортування вершин, розташування їх відносно початку координат.

Використання об’єктно-орієнтованого підходу до розроблення ПЗ системи дає змогу швидко та зручно його модифікувати і вдосконалювати.

4.1.4. Розроблення інформаційного забезпечення системи

Наступним важливим етапом розроблення програмної системи є побудова інформаційного забезпечення [29]. В даному випадку розробляються структури класів даних системи синтезу моделей на основі мереж Петрі.

В роботі представлена ієрархічна структура даних вхідного файлу, структура якого зображена на рис. 4.5.

Вхідний файл має структуру, яку можна зобразити у формі дерева:

- мережа, яка містить перелік вершин;
    - вершина, кожна з яких містить наступні властивості:
        - назва вершини;
        - вхідні ребра, які містять наступні властивості:
            - перехід, від якого іде ребро;
            - вага ребра;
            - колір ребра;
            - ребро нормальне чи інвертоване.
        - вихідні ребра, які містять наступні властивості:
            - перехід, до якого іде ребро;
            - вага ребра;
            - колір ребра;
            - ребро нормальне чи інвертоване.

Приклад вхідного файлу системи в XML-форматі наведено в додатку П.

Використання XML-формату має такі переваги [193]:

- є сучасним промисловим стандартом для збереження та обміну даними між програмними системами;
- орієнтований на мережу Інтернет: застосування XML-формату дає можливість полегшити онлайн-режим обміну даними;

Рис.4.5. Структура класів даних системи автоматизованого синтезу моделей на основі мереж Петрі

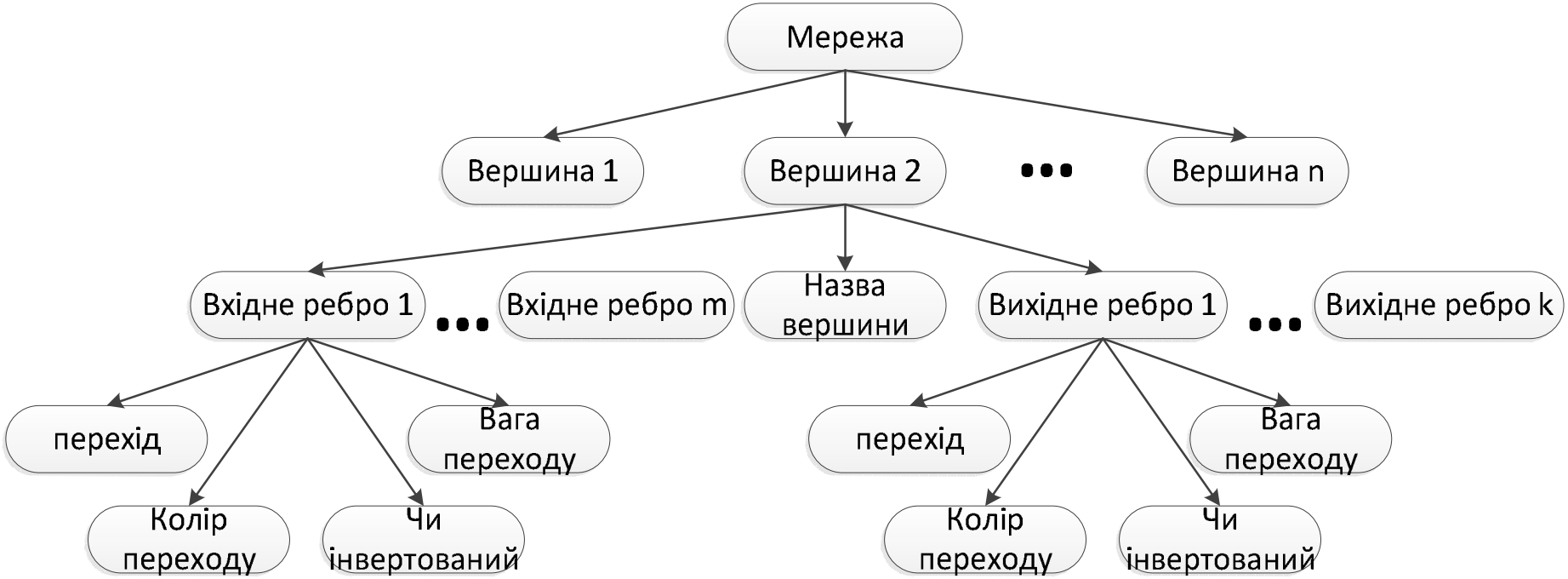


Рис.4.6. Структура вхідного файлу системи

– зручний інструментарій значно спрощує внесення змін до програмного забезпечення;

– єдине програмне та алгоритмічне рішення для онлайн і офлайн (файлового обміну);

– усунення проблеми з кодовими сторінками, оскільки кодова сторінка безпосередньо вказується в наборі даних, і відповідно, не потрібно виставляти будь-які вимоги, перекодування виконується автоматично;

– оптимізація структури звітності завдяки більшої гнучкості, замість лінійної таблиці, в якій, наприклад, наявна множина полів, використовувати набір таблиць, пов'язаних між собою логічно. Це спрощує завдання конвертації даних при імпорті/експорті та використанні тих же даних для побудови аналітики;

– рішення стає по-справжньому багатоплатформним, оскільки для читання/ запису формату XML існує ряд безкоштовних та доступних програмних розв’язків для різних операційних систем;

– формат підтримується більшістю серверів систем керування базами даних, а також офісними пакетами (MS Office 2007 використовує XML як основний формат зберігання даних);

– спрощуються процедури імпорту та експорту в різні формати і способи візуалізації інформації;

– дає можливість використовувати універсальний механізм XSLT-перетворень для автоматичного під’єднання форм друку, перетворення у формат HTML для розміщення на веб-сторінках;

– спрощене впровадження електронного підпису та документообігу; додавання інформації про особу, що подає звітність безпосередньо до звіту, можливість ідентифікації звітності за змістом при архівному зберіганні.

Структура вихідного файлу розробленої системи є вхідним файлом мережі Петрі та має наступну структуру (рис. 4.7):

– мережа, яка містить наступні властивості:

  - опис міток;

- перелік вершин, кожна з яких містить наступні властивості:
  - координати вершини;
  - назва вершини, з підпунктами:
    - сама назва;
    - розташування напису.
  - початкове маркування, з підпунктами:
    - перелік кожного типу вершин;
    - розташування міток;
  - потужність вершини.
- перелік переходів, кожний з яких містить наступні властивості:
  - координати переходу;
  - назва переходу, з підпунктами:
    - сама назва;
    - координати напису;
  - орієнтація переходу (кут повороту);
  - швидкість дії;
  - чи часовий перехід;
  - чи може спрацьовувати без зупинок;
  - пріоритет.
- перелік ребер, кожне з яких містить наступні властивості:
  - ребро, з підменю:
    - колір;
    - вага ребра.
  - чи позначене ребро;
  - перелік точок, які з’єднуються ребром (ламана лінія).
  - тип ребра.

Частина файлу, яка відображає структуру мережі зображено в додатку Р.

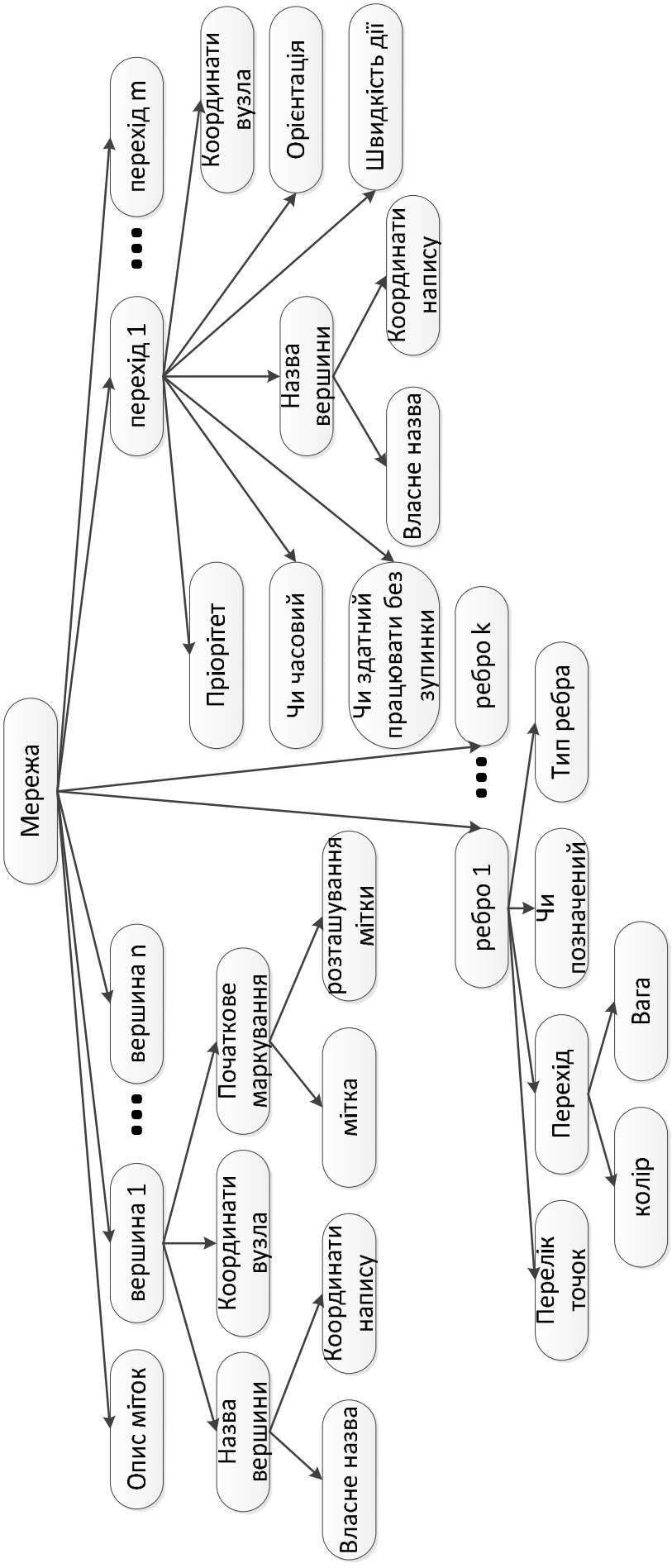
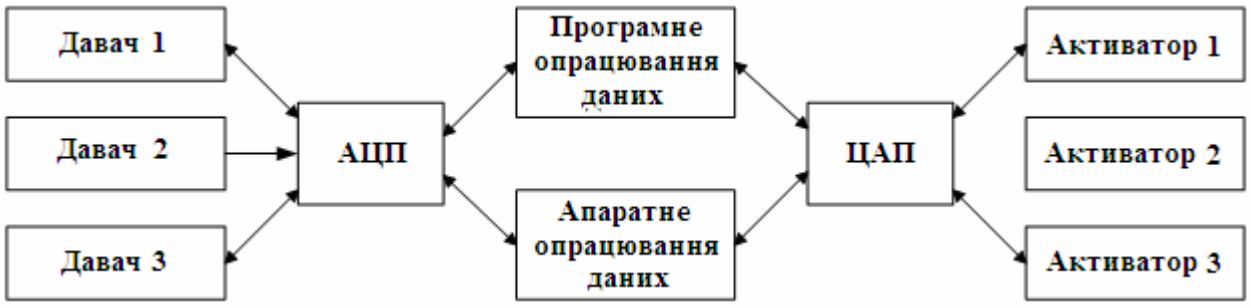
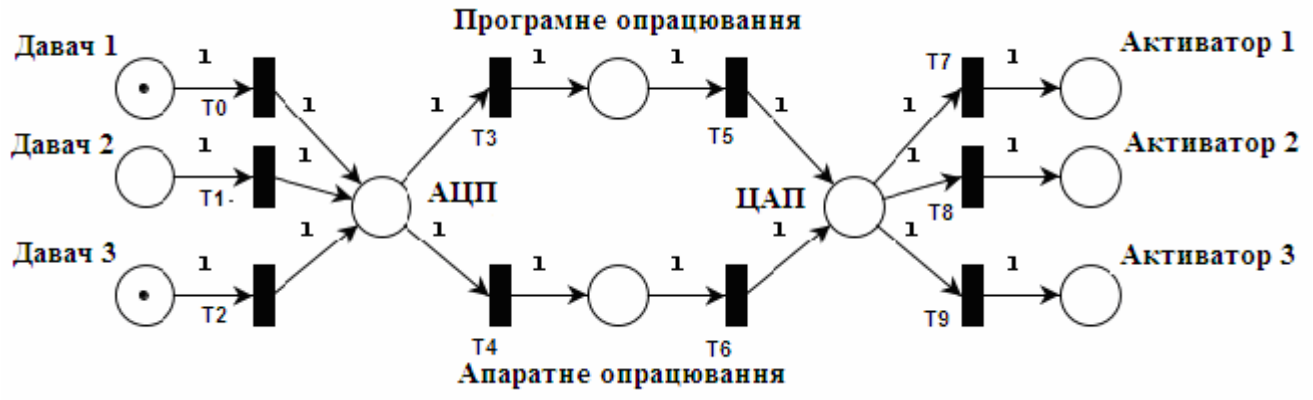


Рис.4.7. Структура вихідного файлу мережі Петрі

На рис. 4.8а наведено структуру підсистеми, а на рис. 4.8б згенеровану модель з використанням мережі Петрі. Представлення даної моделі в XML-форматі наведено в додатку Д.



а)



б)

Рис. 4.8. Приклад структури підсистеми та згенерованої моделі на основі мережі Петрі

Отже, можна зробити висновок, що розроблена система для генерування моделей на основі мереж Петрі працює правильно та коректно.

4.2. Пакет прикладних програм для побудови та дослідження моделей на основі штучних нейронних мереж

На сьогодні процес проектування складних систем неможливий без використання спеціального програмного забезпечення. Розроблене програмне

забезпечення дає змогу врахувати особливості та специфіку об’єкта проектування, вимоги замовника щодо інтерфейсу та алгоритму побудови моделей і їх дослідження.

4.2.1. Розроблення структури пакету прикладних програм та основних алгоритмів

Отже, розроблена система реалізована у формі пакету прикладних програм. Пакет прикладних програм (ППП) дає змогу розробляти підсистеми “інтелектуального будинку” [194, 195] на основі наступних видів нейронних мереж (в процесі реалізації системи опрацювання інформації в системі “інтелектуального будинку”), а саме [101]: багатошарового перцептрона [196]; мережі Хопфілда; нейроподібних структур машини геометричних перетворень (НС МГП); нейронної мережі на основі моделі “Функціонал на множині табличних функцій” (НМ ФМТФ).

ППП складається з таких основних елементів:

– підсистеми вводу-виводу – відповідає за ввід/вивід даних з консолі та у файлах;

– інтерфейсів користувача – відповідають за взаємодію користувача і системи та містить декілька інтерфейсів для виконання різних функцій;

– підсистеми запуску мережі – відповідає за використання поточної навченої системи;

– підсистема навчання мережі – складається з декількох блоків, які реалізують навчання системи різними методами (алгоритмом багатошарового перцептрона, мережа Хопфілда, НС МГП, НМ ФМТФ);

– підсистема виводу/візуалізації мережі – відповідає за представлення мережі у зручному для користувача вигляді.

Розроблена структура використовує модульний підхід, що дає змогу швидко та ефективно вдосконалювати розроблену систему.

4.2.2. Особливості розроблення програмного забезпечення ППП

В процесі реалізації програмного забезпечення ППП було досліджено що, усі методи мають певні загальні властивості, відповідно, використовуючи методи об'єктно-орієнтованого програмування (ООП) [197], було розроблено початкові об'єкти з найвищим рівнем абстракції. Структура програмного забезпечення наведена на рис. 4.9.

Система працює за алгоритмом зображеним на рис. 4.10. Спочатку відбувається введення вхідних даних. Після того відбувається перевірка даних на коректність. В залежності від вибору методу відбувається навчання мережі.

Розглянемо роботу алгоритму на прикладі навчання мережі багатошарового перцептрона (рис. 4.11). Навчання мережі відбувається протягом попередньо визначеної кількості епох. Кожна епоха складається з певної кількості навчальних тестів.

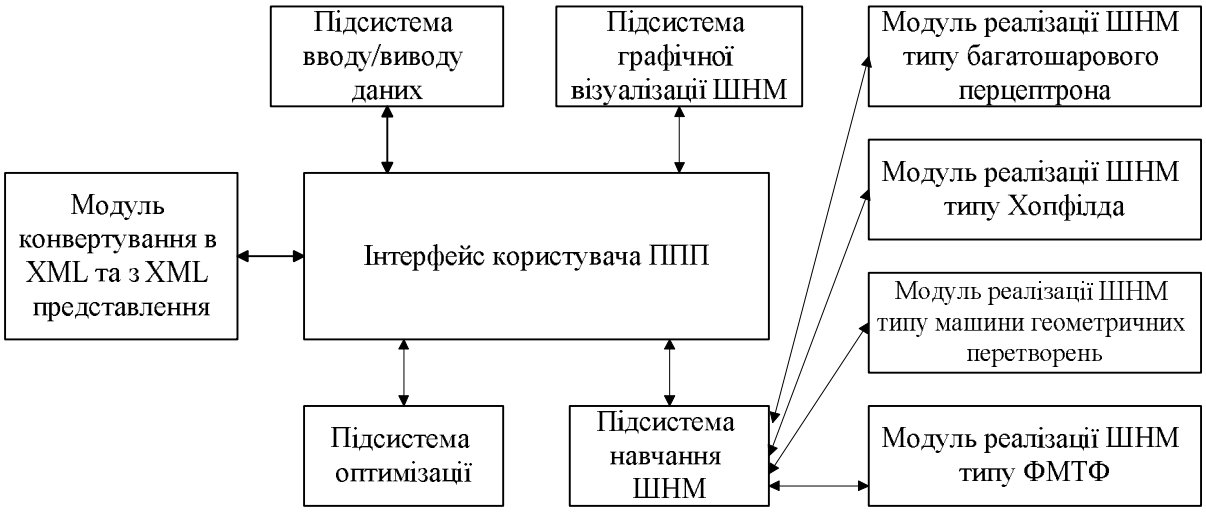


Рис.4.9. Структура розробленого ППП

Після вибору поточного тесту, вводяться вхідні значення мережі. На наступному кроці, послідовно обчислюють вхідні і вихідні значення нейронів у кожному шарі. На основі різниці між отриманим на виході значенням і необхідним, обчислюється похибка. У зворотному напрямку поширюється

отримана похибка. Використовуючи зміщення перераховуються вагові коефіцієнти зв’язків між шарами.

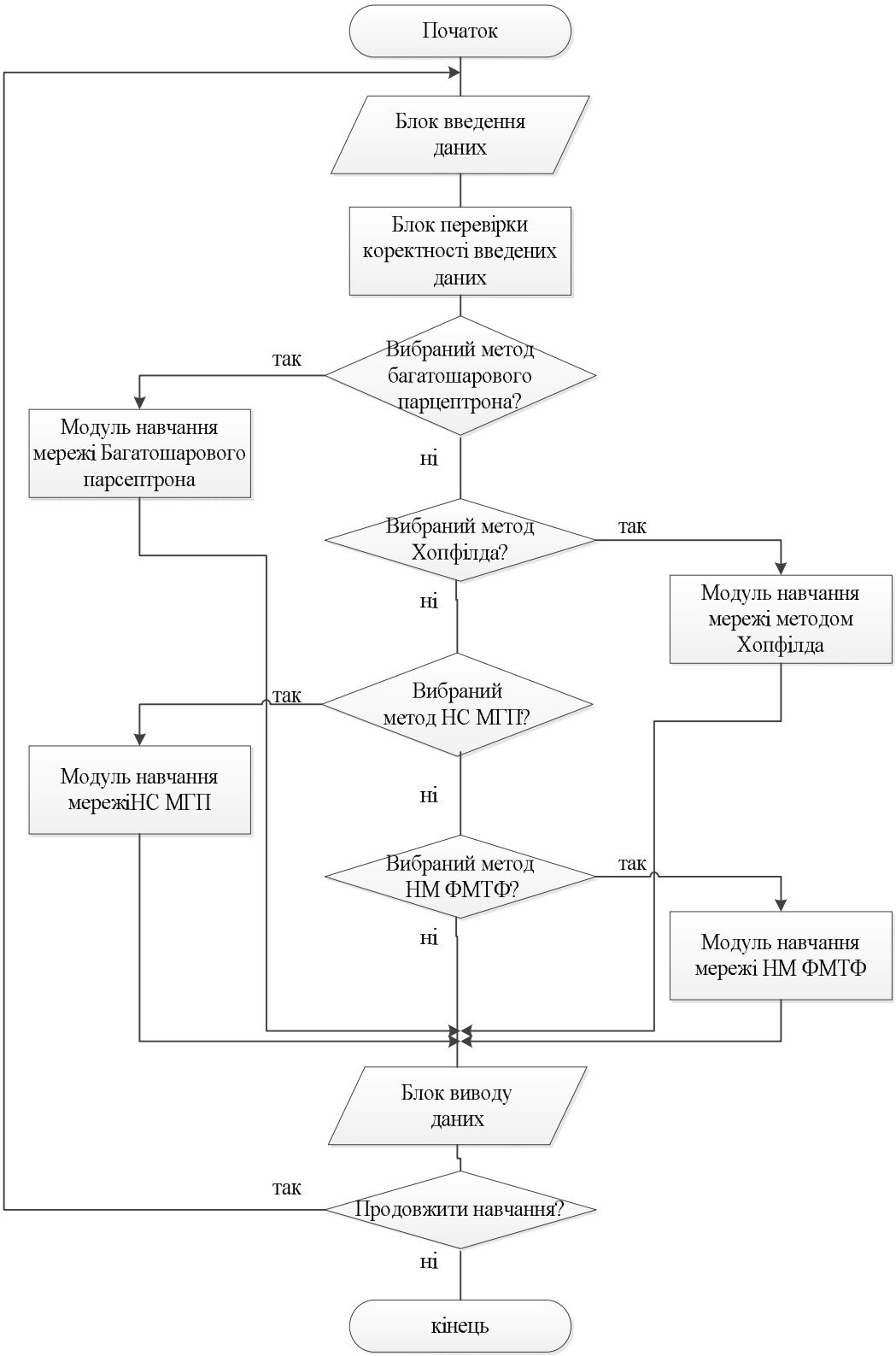


Рис.4.10. Блок-схема алгоритму роботи ППП

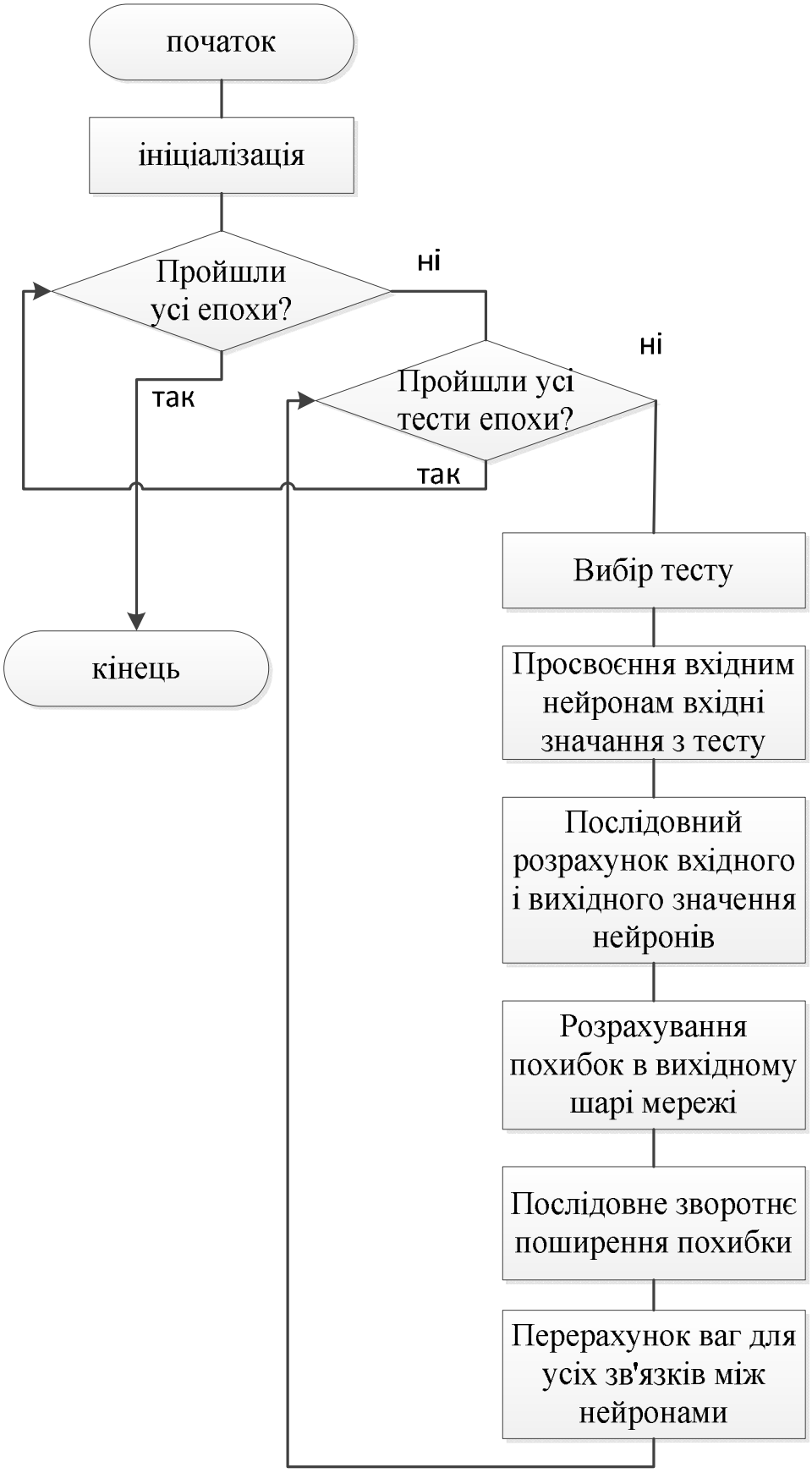


Рис.4.11. Блок-схема алгоритму навчання багатошарового перцептрона

4.2.3. Розроблення інформаційного забезпечення ППП

Великий обсяг вхідних і вихідних даних при обміні між підсистемами програмних засобів, опрацювання слабоструктурованих та нечітких даних всистемах ІБ з використанням штучних нейронних мереж, різні формати представлення об’єктів у системі, жорсткі вимоги до часу опрацювання таких даних потребують розв’язання задачі пов’язаної з ефективним опрацюванням та представленням даних в процесі реалізації програмного засобу. Одне із рішень, яке дало б змогу розв’язати поставлену задачу ефективного опрацювання даних, полягає у використанні мови XML для опису вхідних, внутрішніх та вихідних даних [198, 199].

Для представлення інформаційних моделей в програмних засобах опрацювання слабоструктурованих та нечітких даних в системах ІБ з використанням штучних нейронних мереж використано XML-формат, що дає змогу забезпечити єдиний спосіб обміну даними між підсистемами та забезпечує ефективний обмін даними з іншими існуючими системами.

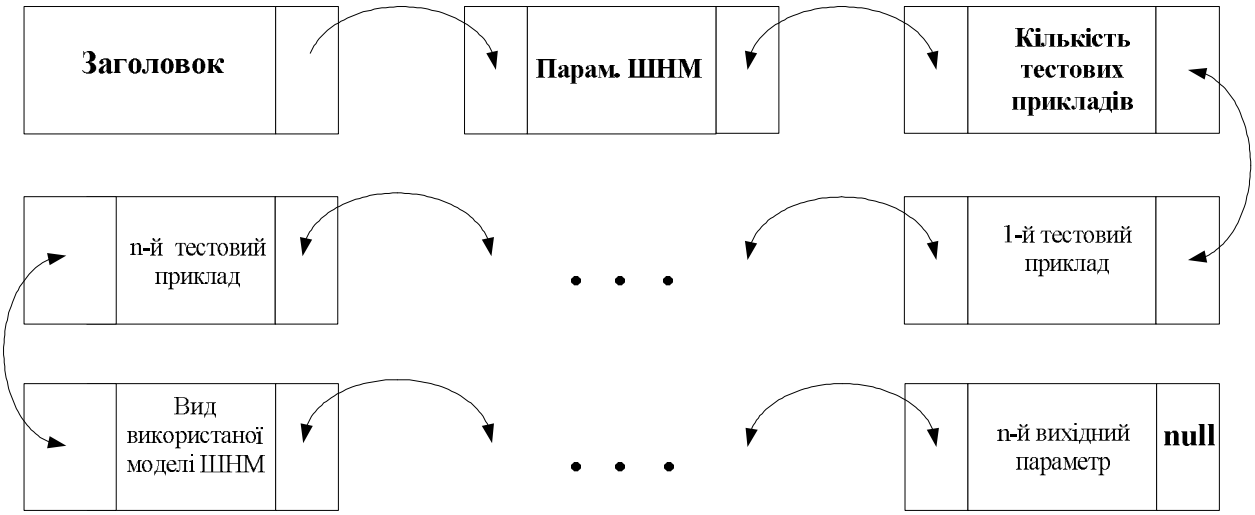


Рис.4.12. Структура даних на основі двозв'язних списків для збереження інформації про ШНМ

Зокрема, одна з структур робочого файлу з використанням XML-формату містить такі блоки:

– заголовок;

– вхідні параметри структури ШНМ;

– список тестових прикладів;

– вихідні результати.

Для реалізації розробленої інформаційної моделі використано структури даних на основі двозв’язних списків рис. 4.12 [200].

Меню для роботи з багатошаровим перцептроном складається з трьох частин: меню; область введення даних (червоний контур), куди заносяться всі необхідні дані для навчання мережі; область управління (синій контур).

4.3. Розроблення підсистеми віддаленого керування “інтелектуальним будинком”

4.3.1. Особливості розроблення підсистеми віддаленого керування “інтелектуальним будинком”

В дисертаційній роботі розроблена структурна схема організації обміну даними між системою інтелектуального будинку та підсистемою віддаленого керування (рис. 4.13).

Наведена схема передбачає використання спеціального файлу з основними параметрами інтелектуального будинку, який передається від підсистеми віддаленого керування до системи ІБ з використанням мережі Інтернет. Слід зауважити, що в процесі вдосконалення системи віддаленого керування будуть використані й інші канали передачі інформації.

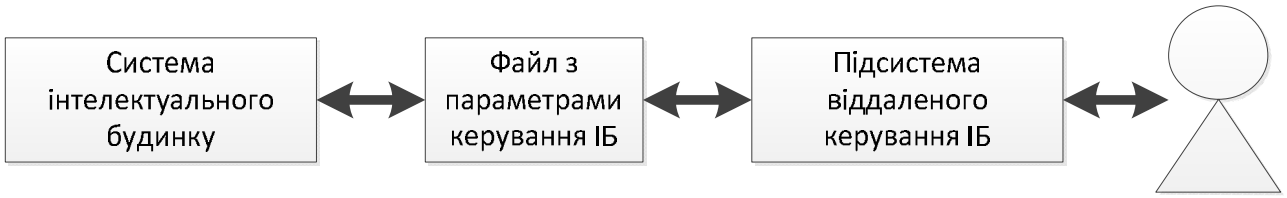


Рис.4.13. Структурна схема організації віддаленого керування ІБ

Підсистему віддаленого керування ІБ програмно реалізовано і приклад основного меню зображено на рис. 4.14, де можна побачити, що кожна кімната ІБ представляється окремою вкладкою, котра розділена на 3 значущі частини: освітлення, клімат-контроль та жалюзі. Основне меню, за бажанням користувача, можна змінювати з використанням спеціального конструктора основного меню.

В якості прикладу на рис. 4.14 зображено засоби керування підсистемою клімат-контролю і освітлення, де можна регулювати за допомогою повзунків параметри зазначених підсистем. Жалюзі в кожній кімнаті можна піднімати і опускати за допомогою спеціальної кнопки, що зразу ж відображається відповідним написом на кнопці. Освітлення та клімат-контроль можна повністю відключити кнопками On/Off. Використання таких елементів в основному меню дає змогу зробити його більш інтуїтивним та зрозумілим для користувача початківця.

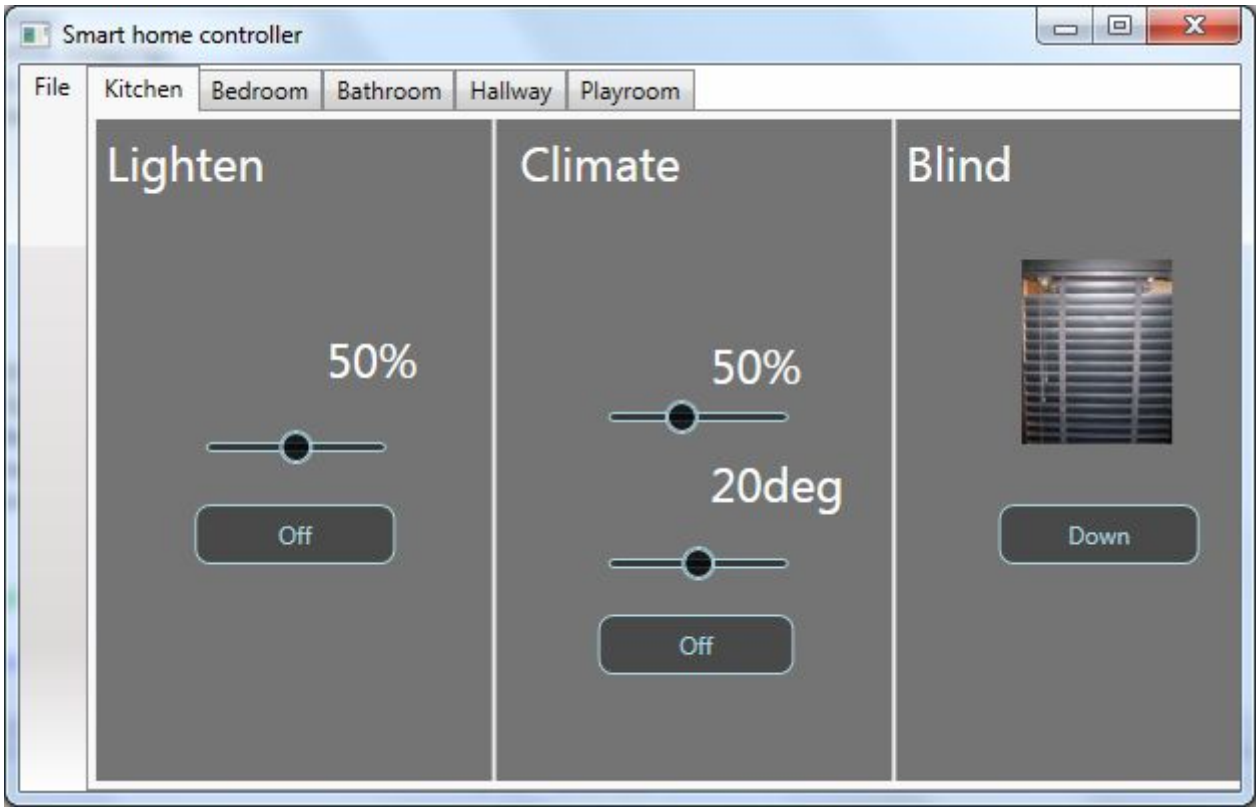


Рис.4.14. Приклад основного вікна підсистеми віддаленого керування ІБ

4.3.2. Розроблення програмного забезпечення підсистеми

Структура програмного забезпечення включає 2 класи: стандартний MainWindow, в якому виконуються весь функціонал і опрацьовуються запити від користувача, та класу Room, котрий містить в собі усі поля, в яких зберігаються значення. Їх користувач може змінювати власноруч, або ж вони змінюються стандартно (за замовчуванням).

Підсистема дає змогу управляти окремо кожною з кімнат, які програмно представлені окремим екземпляром об’єкта класу Room. Ці екземпляри опрацьовуються окремими обробниками подій. Дані налаштування для кожної кімнати зберігаються в кожному об’єкті окремо. Це все забезпечує незалежність даних між собою.

Після виконання настроювань параметрів налагодження системи ІБ дані зберігаються в зовнішньому файлі формату *.xml і відправляються, з використанням мережі Інтернет до основної системи ІБ. Більше того, в розробленій підсистемі присутні засоби, які дають змогу визначити стан (параметри) ІБ на даний момент часу.

4.3.3. Розроблення інформаційного забезпечення підсистеми

Важливим елементом підсистеми віддаленого керування “інтелектуальним будинком” є інформаційне забезпечення, зокрема файл з параметрами настроювання підсистем ІБ, для опису якого в роботі використано мову XML [198, 199]. Приклад такого файлу зображено на рис. 4.15.

Використання XML-формату для опису даних про параметри “інтелектуального будинку” дає змогу ефективно організувати обмін даними з різними підсистемами ІБ, системою ІБ вцілому та засобами віддаленого керування. Запропонований формат забезпечує простоту формування та редагування даного файлу, що є надзвичайно важливим елементом в процесі розвитку та вдосконалення системи “інтелектуального будинку”.

```
<?xml version="1.0" encoding="utf-8"?>
<ArrayOfRoom xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<Room>
<title>kitchen</title>
<lighten>0.99</lighten>
<degree>15</degree>
<humidity>0.67</humidity>
<louvre>false</louvre>
</Room>
<Room>
<title>bedroom</title>
<lighten>0</lighten>
<degree>26.27</degree>
<humidity>0.5</humidity>
<louvre>false</louvre>
</Room>
<Room>
<title>bathroom</title>
<lighten>0</lighten>
<degree>20</degree>
<humidity>0.5</humidity>
<louvre>true</louvre>
</Room>
<Room>
<title>hallway</title>
<lighten>0.27</lighten>
<degree>24.34</degree>
<humidity>0.3</humidity>
<louvre>true</louvre>
</Room>
<Room>
<title>platroom</title>
<lighten>0</lighten>
<degree>22.41</degree>
<humidity>0.64</humidity>
<louvre>false</louvre>
</Room>
</ArrayOfRoom>
```

Рис. 4.15. Приклад файлу з параметрами настроювання “інтелектуального будинку” в XML-форматі

4.4. Результати розроблення фізичних моделей нейроконтролерів системи “інтелектуального будинку”

4.4.1. Фізична модель нейроконтролера для підсистеми клімат-контролю “інтелектуального будинку”

Апаратна реалізація нейроконтролера для підсистеми клімат-контролю містить в собі мікроконтролер, давач температури і вологості DHT11, три світлодіоди і обмежуючі резистори. У мікроконтролері використовуються чотири порти для роботи з зовнішньою схемою, а саме: 2-й порт використовуються для зчитування сигналу від давача, а 5-й, 6-й, 7-й порти використовуються для відображення активної роботи пристроїв (обігрівач, вентилятор і зволожувач повітря), що сигналізується спалахом відповідного світлодіода (рис. 4.16).

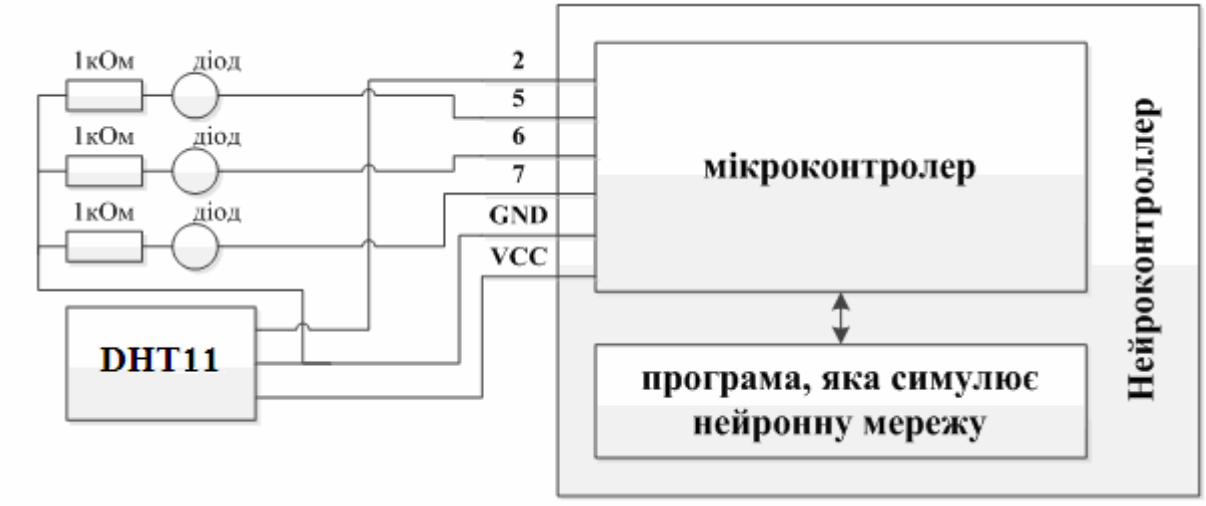


Рис. 4.16. Схема підключення підсистеми клімат-контролю

Програма роботи мікроконтролера описана в третьому розділі дисертаційної роботи (Додаток А).

Після запуску нейроконтролера на один цикл роботи програми, через послідовний порт на комп’ютер було надіслано наступні проміжні і вихідні результати (рис. 4.17), що дає змогу перевірити правильність та коректність роботи розробленого нейроконтролера.

DHT11 TEST PROGRAM
LIBRARY VERSION: 0.3.2
Read sensor: OK
Humidity (%): 70.00
Temperature (oC): 29.00
Temperature (oF): 84.20
Temperature (K): 302.15
Dew Point (oC): 22.99
Dew PointFast (oC): 22.96
x0(T) = 1.63
x1(H) = 0.92
Result
0 out = 0.45
1 out = 1.34
2 out = -0.06

Рис. 4.17. Дані отримані від давача через COM-порт

Отже, розроблена модель на основі штучних нейронних мереж забезпечує опрацювання нечітких та неструктурованих даних від підсистеми давачів.

Розроблена модель на основі ШНМ та побудованого спеціалізованого програмного забезпечення дають змогу побудувати нейроконтролер підсистеми клімат-контролю.

4.4.2. Фізична модель нейроконтролера для підсистеми освітлення “інтелектуального будинку”

Для реалізації нейроконтролера для підсистеми освітлення ІБ, необхідно використати програмований мікроконтролер, який буде містити в собі запрограмовану нейронну мережу. Більшість мікроконтролерів програмуються на С-подібній мові, і необхідно написати керуючу програму з навченою нейронною мережею. Для реалізації було вибрано мікроконтролер сімейства AVR. Цей мікроконтролер був вибраний через свою поширеність і простоту роботи. Окрім мікроконтролера необхідні давачі. Для роботи з давачами сумісними з

мікроконтролерами сімейства AVR написано багато безкоштовних бібліотек, що значно зменшує час створення програм.

Програмна модель включає такі блоки:

– блок вибору портів нейроконтролера та введення даних нейроконтролера;

– блок ініціалізації даних мережі;

– блок роботи мережі;

– блок ініціалізації портів нейроконтролера;

– блок основного циклу програми нейроконтролера.

У пам'ять мікроконтролера було записано програму, яку наведено у додатку Б і, яка реалізує функції нейроконтролера.

Приклад схеми підключення нейроконтролера до давачів і виконуючого пристрою зображено на рис. 4.18.

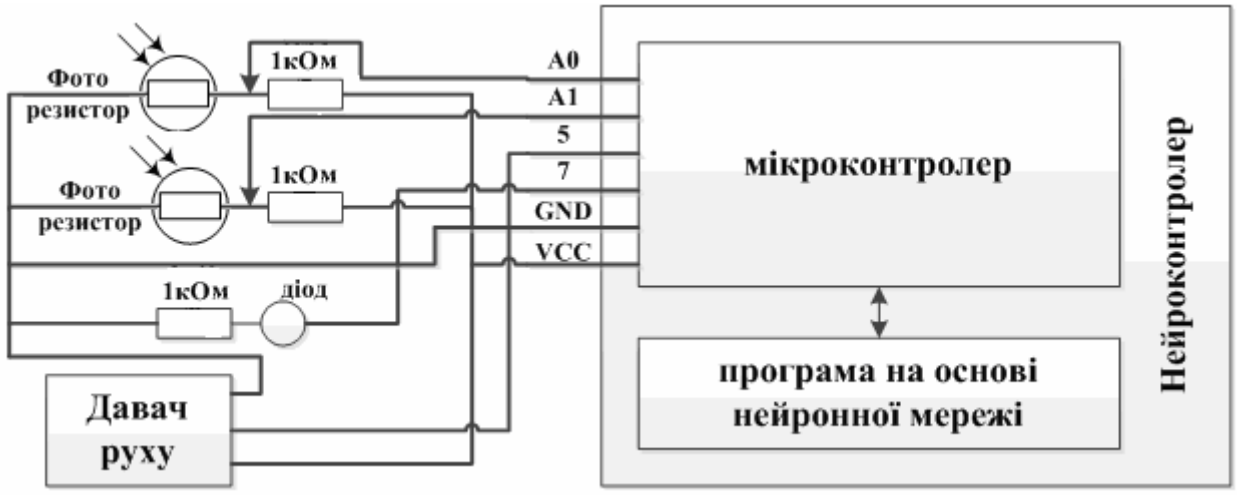


Рис. 4.18. Приклад схеми підключення нейроконтролера до давачів і виконуючого пристрою

Результати тестування розробленого нейроконтролера – на рис. 4.19. Результати роботи нейроконтролера виводять через USB (віртуальний COM-порт).

NeuroController lighting subsystem
MotionSensor1 Digital pin = 0 //немає руху
FotoResistor1 Analog pin = 1 //є світло надворі
FotoResistor2 Analog pin = 1// є світло в кімнаті
Result
0 out = 0.01 //непотрібно запалювати лампочку
Turn off LED //виключити фотодіод

Рис. 4.19. Результати тестування розробленого нейроконтролера

4.4.3. Фізична модель нейроконтролера підсистеми захисту “інтелектуального будинку”

Тестова фізична модель підсистеми містить в собі мікроконтролер, давачі руху і перемикачі контурів, 6 світлодіодів і обмежуючі резистори. У мікроконтролері використовуються 14 портів для роботи з зовнішньою схемою, а саме: A1-A4 – зчитування аналогового сигналу від ключів

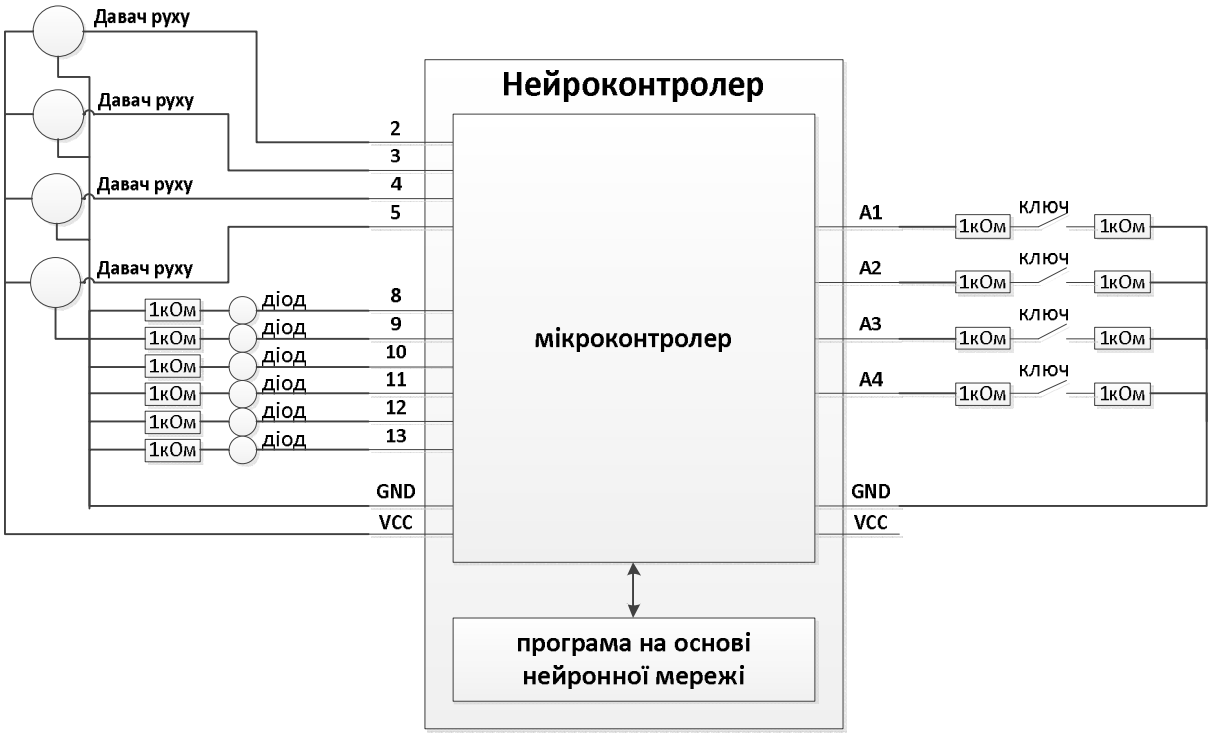


Рис.4.20. Схема підключення системи

2-5 цифрові вхідні контакти, з них зчитуються дані від давачів руху, 8-13 цифрові

вихідні контакти, вони подають напруги на світлодіоди, які засвічуються залежно від стану системи (рис. 4.20). Також мікроконтролер програмується програмою наведеною в додатку В.

Після запуску нейроконтролера на один цикл програми, через COM-порт на комп'ютер було надіслано наступні проміжні результати і вихідні результати (рис. 4.21). Отримані дані дають змогу стверджувати, що розроблена підсистема працює правильно та коректно.

TEST
key1 = 1023 key2 = 1023 key3 = 1023 key4 = 1023// аналогові сигнали на ключах
mov1 = 0 mov2 = 1 mov3 = 1 mov4 = 0//дані від давачів руху
0 vout[] = 1.00 vin[] = 0.00 //балансуючий нейрон
1 vout[] = 0.00 vin[] = 0.00 //вхідний шар
2 vout[] = 0.00 vin[] = 0.00 //вхідний шар
3 vout[] = 0.00 vin[] = 0.00 //вхідний шар
4 vout[] = 1.00 vin[] = 0.00 //вхідний шар
5 vout[] = 0.00 vin[] = 0.00 //вхідний шар
6 vout[] = 1.00 vin[] = 0.00 //вхідний шар
7 vout[] = 0.00 vin[] = 0.00 //вхідний шар
8 vout[] = 0.00 vin[] = 0.00 //вхідний шар
9 vout[] = 1.00 vin[] = 0.00 //балансуючий нейрон
10 vout[] = 0.00 vin[] = -6.21 //проміжний шар шар
11 vout[] = 0.99 vin[] = 4.43 //проміжний шар
12 vout[] = 0.99 vin[] = 4.64 //проміжний шар
13 vout[] = 0.00 vin[] = -5.98 //проміжний шар
14 vout[] = 1.00 vin[] = 0.00 //балансуючий нейрон
15 vout[] = 1.00 vin[] = 10.78 //вихідний шар

Рис. 4.21. Дані отримані від давача через COM-порт

4.4.4. Фізична модель нейроконтролера підсистеми запобігання технічних аварій "інтелектуального будинку"

В інженерії, найвищою точністю володіють фізичні моделі. Відповідно було реалізовано таку тестову модель підсистеми ЗТА ІБ. Для її реалізації було вибрано наступні давачі: давач газу та диму MQ-2 [201]; давач вогню Arduino Compatible Mini Flame Fire Wavelength Sensor [202]; давач протікання води, базується на принципі замикання кола водою із розчином солі.

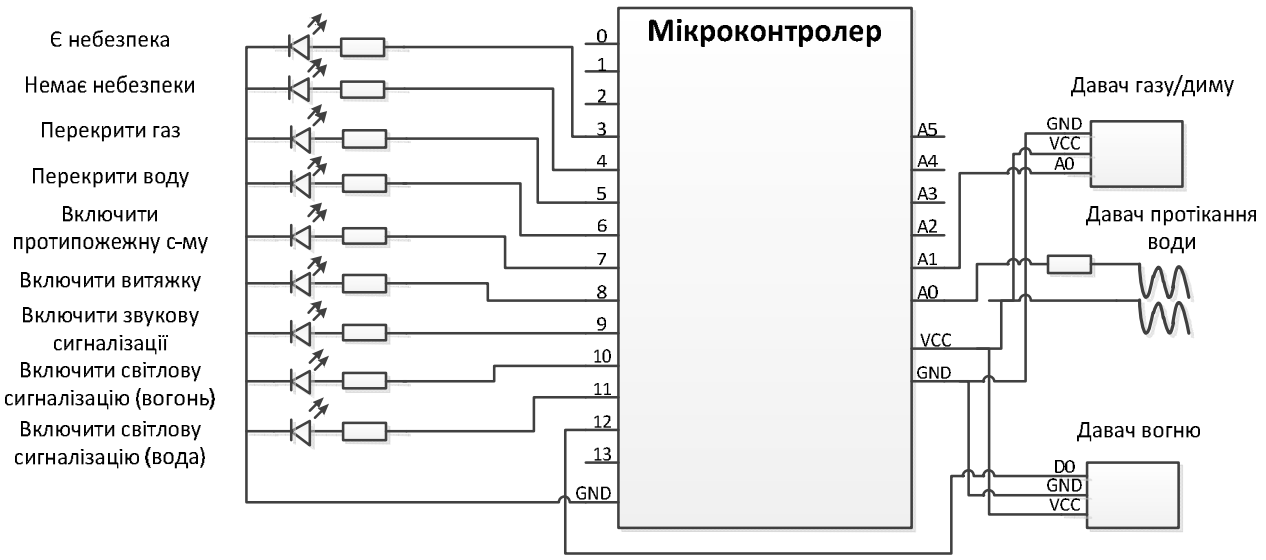


Рис. 4.22. Схема підключення основних елементів фізичної моделі підсистеми запобігання технічних аварій ІБ

Виведені результати від нейроконтролера через COM- порт.

NeuroController protection subsystem
Fire D =0, Smoke A =425, Water A =85
0 vout[] = 1.000 vin[] = 0.000
1 vout[] = 0.000 vin[] = 0.000
2 vout[] = 0.000 vin[] = 0.000
3 vout[] = 1.000 vin[] = 0.000
4 vout[] = 1.000 vin[] = 0.000
5 vout[] = 0.003 vin[] = -5.754
6 vout[] = 0.677 vin[] = 0.738
7 vout[] = 0.988 vin[] = 4.437
8 vout[] = 1.000 vin[] = 0.000
9 vout[] = 0.997 vin[] = 5.794
10 vout[] = 0.002 vin[] = -6.324
11 vout[] = 0.994 vin[] = 5.065
12 vout[] = 0.005 vin[] = -5.383
13 vout[] = 1.000 vin[] = 10.430
14 vout[] = 0.997 vin[] = 5.795
15 vout[] = 0.002 vin[] = -6.324
Result
0 out=0.997, 1 out=0.002, 2 out=0.994, 3 out=0.005,
4 out=1.000, 5 out=0.997 6 out=0.002

Рис. 4.23. Результати тестування розробленого нейроконтролера підсистеми запобігання технічних аварій ІБ

Вибрані давачі є сумісними із мікроконтролером AVR. Також дані давачі є дешевими та простими у використанні та опрацюванні результатів. Для відображення спрацювання відповідного активатора використано світлодіоди.

Приклад схеми фізичної моделі зображено на рис. 4.22. Результати тестування роботи побудованої підсистеми – на рис. 4.23.

Отримані результати підтверджують точність, адекватність та достовірність розроблених теоретичних моделей.

4.5. Висновки до розділу 4

1. Розроблено структуру системи для автоматизованого синтезу моделей на основі теорії мереж Петрі. Побудована система використовує інформацію про структуру досліджуваної системи і дає змогу автоматизувати процес побудови моделей системного рівня проектування складних об’єктів та систем.

2. Розроблено програмне забезпечення системи синтезу моделей на основі мереж Петрі, яке включає розроблену структуру ПЗ та її реалізацію на мові Java, що забезпечує кросплатформність розроблюваної системи.

3. Побудовано інформаційне забезпечення системи автоматизованого синтезу моделей системного рівня проектування на основі мереж Петрі. Для опису вхідного та вихідного файлів використано ієрархічну структуру та XML-формат, який забезпечує зручний формати обміну інформацією з системами, які дають змогу ефективно опрацьовувати інформацію про моделі на основі мереж Петрі.

4. Розроблено алгоритм автоматизованого синтезу моделей системного рівня проектування на основі мереж Петрі, який дає змогу автоматизувати побудову моделей для верхніх рівнів проектування.

5. Розроблено структурну схему організації віддаленого керування інтелектуальним будинком, яка передбачає використання модульного принципу організації системи та файлу з параметрами настроювань ІБ в процесі обміну даними між складовими структури.

6. Побудовано програмне та інформаційне забезпечення підсистеми

віддаленого керування ІБ. В процесі реалізації інформаційного забезпечення використано мову XML, що забезпечує швидкий та ефективний розвиток і вдосконалення підсистеми віддаленого керування інтелектуального будинку.

7. Розроблено структуру, програмне та інформаційне забезпечення пакету прикладних програм для побудови моделей опрацювання нечітких та неструктурованих даних на основі штучних нейронних мереж.

8. Побудовано фізичну модель підсистеми клімат-контролю ІБ у формі нейроконтролера, яка використовує мікроконтролер AVR та програмну модель на основі штучної нейронної мережі і дає змогу дослідити адекватність побудованих моделей, швидкодію, надійність та функціональність розробленої підсистеми.

9. Побудовано фізичну модель підсистеми освітлення ІБ у формі нейроконтролера, яка використовує мікроконтролер AVR та програмну модель на основі штучної нейронної мережі і дає змогу дослідити адекватність побудованих моделей, швидкодію, надійність та функціональність розробленої підсистеми.

10. Побудовано фізичну модель підсистеми захисту ІБ у формі нейроконтролера, яка використовує мікроконтролер AVR та програмну модель на основі штучної нейронної мережі і дає змогу дослідити адекватність побудованих моделей, швидкодію, надійність та функціональність розробленої підсистеми.

11. Побудовано фізичну модель підсистеми запобігання технічних аварій ІБ у формі нейроконтролера, яка використовує мікроконтролер AVR та програмну модель на основі штучної нейронної мережі і дає змогу дослідити відповідність побудованих моделей, швидкодію, надійність та функціональність розробленої підсистеми.

ВИСНОВКИ

У дисертаційній роботі розв’язано наукове завдання підвищення ефективності автоматизованого проектування систем “інтелектуального будинку” на основі розробленого методу, моделей та засобів.

Отримано такі наукові та практичні результати:

1. Проведено аналіз методів, моделей та засобів автоматизованого проектування систем “інтелектуального будинку”, що дало змогу зробити висновок про необхідність підвищення рівня автоматизації процедур синтезу схемних моделей, розроблення моделей для інтелектуалізації функцій таких систем.

2. Отримав подальший розвиток метод автоматизованого синтезу моделей на основі мереж Петрі для системного рівня автоматизованого проектування, який ґрунтується на інформації про структуру системи та дає змогу підвищити рівень автоматизації побудови структурних моделей підсистем “інтелектуального будинку” на 50 – 70 %.

3. Вперше введено інтелектуальний аспект на усіх рівнях автоматизованого проектування таких систем та сформульовано основні задачі на кожному з ієрархічних рівнів, що дає змогу підвищити ефективність проектування систем “інтелектуального будинку”.

4. Вперше розроблено структурні моделі для аналізу роботи системи “інтелектуального будинку”, які ґрунтуються на теорії кольорових мереж Петрі і дають змогу визначити динаміку роботи, перевірити спроектовану систему на наявність тупиків, на живучість та обмеженість.

5. Вдосконалено для кожної підсистеми “інтелектуального будинку” розроблену спеціалізовану модель на основі мереж Петрі – класичних (для

підсистем освітлення та захисту) і кольорових (для підсистеми клімат-контролю). Розроблені в роботі моделі на основі мереж Петрі основних підсистем “інтелектуального будинку” дають змогу здійснити детальний аналіз динаміки досліджуваних процесів всередині кожної підсистеми ще на системному рівні автоматизованого проектування системи “інтелектуального будинку” у цілому, забезпечуючи тим самим підвищення надійності реалізації системи “інтелектуального будинку”.

6. Вперше розроблено моделі підсистем клімат-контролю, освітлення, захисту та запобігання технічних аварій “інтелектуального будинку”, які використовують штучну нейронну мережу на основі багатошарового перцептрона, що дає змогу опрацьовувати нечіткі та неструктуровані дані від підсистеми давачів.

7. Отримали подальший розвиток фізичні моделі підсистем клімат-контролю, освітлення, захисту та запобігання технічних аварій “інтелектуального будинку”, у формі нейроконтролера, які використовують мікроконтролер AVR та програмні моделі на основі штучних нейронних мереж і дають змогу дослідити адекватність побудованих моделей, швидкодію та функціональність розроблених підсистем.

8. Розроблено програмно-апаратні засоби автоматизованого проектування систем “інтелектуального будинку”, які дають змогу організувати обмін даними з існуючими системами шляхом використання XML-формату.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Teslyuk V. Automation of the smart house system-level design / Teslyuk V., Beregovskyi V., Pukach A. // Informatyka Automatyka Pomiary w Gospodarce i Ochronie Środowiska. Polish magazin. Zeszyt 4. – 2013. – P. 81-84. *(BazTech)*
2. Береговський В. В. Методи та моделі автоматизованого проектування системи “інтелектуального будинку” на базі нейроконтролерів / Береговський В. В., Теслюк В. М., Матвійчук К. В., Денисюк П. Ю. // Науковий вісник НЛТУ України : Збірник науково-технічних праць. – Вип. 26.7. – Львів : РВВ НЛТУ України, 2016. – С. 342-349. (*Index Copernicus*)
3. Сидор А. Р. Математичне моделювання параметрів надійності несиметричних розгалужених систем / Сидор А. Р., Теслюк В. М., Береговський В. В. // Вісник Національного університету “Львівська політехніка” : № 771 : Комп’ютерні науки та інформаційні технології. – Львів, 2013. – С. 167-173. *(Inspec)*
4. Теслюк В. М. Підсистема віддаленого керування інтелектуальним будинком / Теслюк В. М., Береговський В. В., Нижник А. Р., Береговська Х. В. // Науковий вісник НЛТУ України : Збірник науково-технічних праць. – Вип. 23.12. – Львів : РВВ НЛТУ України, 2013. – С. 348-351. (*Index Copernicus*)
5. Теслюк В. М. Автоматизація системного рівня проектування інтелектуального будинку / Теслюк В. М., Береговський В. В., Пукач А. І., Сидор А. Р. // Збірник наукових праць ІППМЕ ім. Г. Є. Пухова НАН України. – Вип. 67. – Київ, 2013. – С. 138-147.
6. Теслюк В. М. Розроблення нейроконтролера для управління підсистемою освітлення інтелектуального будинку / Теслюк В. М., Березький О. М., Береговський В. В., Теслюк Т. В. // Збірник наукових праць ІППМЕ ім. Г. Є. Пухова НАН України. – Вип. 64. – Київ, 2012. – С. 137-143.

7. Теслюк В. М. Модель роботи підсистеми освітлення та охорони інтелектуального будинку / Теслюк В. М., Береговська Х. В., Береговський В. В. // Науковий вісник НЛТУ України : Збірник науково-технічних праць. – Вип. 23.10. – Львів : РВВ НЛТУ України, 2013. – С. 297-303. (*Index Copernicus*)
8. Теслюк В. М. Розроблення структури та моделі підсистеми запобігання технічних аварій для системи інтелектуального будинку / Теслюк В. М., Береговський В. В., Денисюк П. Ю., Теслюк Т.В. // Науковий вісник НЛТУ України : Збірник науково-технічних праць. – Вип. 23.18. – Львів : РВВ НЛТУ України, 2013. – С. 241-245. (*Index Copernicus*)
9. Теслюк В. М. Пакет прикладних програм для побудови та дослідження моделей на основі штучних нейронних мереж / Теслюк В. М., Денисюк П. Ю., Береговський В. В., Ляпандра А. С., Теслюк Т. В. // Моделювання та інформаційні технології. Збірник наукових праць ІППМЕ ім. Г. Є. Пухова НАН України. – Вип. 66. – Київ, 2012. – С. 167-174.
10. Теслюк В.М. Програмно-апаратна реалізація макетного взірця для дослідження методів опрацювання нечітких та неструктурованих даних з використанням штучних нейронних мереж / Теслюк В. М., Березький О. М., Береговський В. В., Денисюк П.Ю., Теслюк Т. В. // Збірник наукових праць ІППМЕ ім. Г. Є. Пухова НАН України. – Вип. 65. – Київ, 2012. – С.125-132.
11. Теслюк В. М. Використання xml для систем автоматизованого генерування моделей на основі мереж Петрі / Теслюк В. М., Береговський В. В., Денисюк П. Ю., Теслюк Т. В., Лозинський А. Я. // Моделювання та інформаційні технології. Збірник наукових праць ІППМЕ ім. Г. Є. Пухова НАН України. – Вип. 70. – Київ, 2013. – С. 129-136.
12. Теслюк В. М. Метод автоматизованого синтезу моделей системного рівня на основі теорії мереж Петрі / Теслюк В. М., Береговський В. В., Теслюк Т. В., Сидор А. Р., Лозинський А. Я. // Збірник наукових праць ІППМЕ ім. Г. Є. Пухова НАН України. – Вип. 68. – Київ, 2013. – С. 85-92.
13. Теслюк В. М. Програмно-апаратна реалізація нейроконтролера для підсистеми клімат контролю інтелектуального будинку / Теслюк В. М., Денисюк П. Ю.,

Теслюк Т. В., Береговський В. В. // Тези доповідей Шостої Міжнародної науково-практичної конференції “Сучасні проблеми і досягнення в галузі радіотехніки, телекомунікацій та інформаційних технологій”. – Запоріжжя : ЗНТУ, 2012. – С. 211-212.

14. Denysyuk P. Neural controller of intelligent building climate control subsystem on the basis of a multilayer perceptron / Pavlo Denysyuk, Taras Teslyuk, Vasyl Beregovskyi, Ivan Cheremisin, Marta Duda // Proceedings of the 7th International Conference of Computer Science and Information Technologies (CSIT’2012), April 14-19, 2012. – Lviv: Publishing House Vezha&Co., 2012. – P. 26-27.
15. Teslyuk T. The model of smart house lighting subsystem analysis on the basis of Petri net theory / Taras Teslyuk, Vasyl Beregovskyi, Yuryy Tertula, Roman Chupa // Proceedings of the 7th International Conference of Computer Science and Information Technologies (CSIT’2012), April 14-19, 2012. – Lviv: Publishing House Vezha&Co., 2012. – P. 172-173.
16. Москаль Б. М. Структурна модель контролера для підсистеми захисту інтелектуального будинку / Москаль Б. М., Береговський В. В // Матеріали III Всеукраїнської школи-семінару молодих вчених і студентів “Сучасні комп’ютерні інформаційні технології”, (ACIT’2013), Травень 23-25, 2013. – Тернопіль : ТНЕУ, 2013. – С. 230-234.
17. Конончук О. О. Структурна модель нейроконтролера для підсистеми освітлення інтелектуального будинку / Конончук О. О., Береговський В.В. // Матеріали III Всеукраїнської школи-семінару молодих вчених і студентів “Сучасні комп’ютерні інформаційні технології”, (ACIT’2013), Травень 23-25, 2013. – Тернопіль : ТНЕУ, 2013 – С. 100-103.
18. Teslyuk V. Structural Model of the Subsystem, which Prevents Industrial Accidents in the System of Smart House / Teslyuk V., Beregovskyi V., Denysyuk P., Teslyuk T. // Proceedings of the 12th International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET’2014), February 25-March 1, 2014. – Lviv-Slavsko, 2014. – P. 607-609.

19. Teslyuk V. M. Development of smart house system model based on colored petri nets / Teslyuk V. M., Beregovskyi V. V., Pukach A. I // Proceedings of the 18th International Seminar : Workshop On Direct And Inverse Problems Of Electromagnetic And Acoustic Wave Theory (DIPED'2013), September 23-26, 2013. – Lviv, 2013. – P. 205-208. (*Scopus*)
20. Teslyuk V. System for Building and Studying of Models Based on Artificial Neural Networks / Vasyl Teslyuk, Vasyl Beregovskyi, Andriy Kernytskyy, Tatyana Teslyuk, Pavlo Denysyuk, Oleksandr Moryshko // Proceedings of the 7th International Conference of Computer Science and Information Technologies (CSIT'2012), April 14-19, 2012. – Lviv : Publishing House Vezha&Co., 2012. – P. 169-171.
21. Kryvyy R. Neurocontroller of the Smart House Climat Control Subsystem Based on RASPBERRY PI / Rostyslav Kryvyy, Pavlo Denysyuk, Vasyl Beregovskyi, Olha Savitska, Zoriana Rybchak // Proceedings of 9th International Conference "Perspective Technologies and Methods in MEMS Design" (MEMSTECH'2013), April 16-20, 2013. – Lviv : Lviv Polytechnic Publishing House, 2013. – P. 181-182.
22. Системы "Умный дом" [Електронний ресурс]. URL: http://www.vashdom.ru/articles/research_2.htm
23. Перспективы рынка систем "Умный дом" [Електронний ресурс]. URL: http://www.vira.ru/exp/reviews/umdom.html
24. Euronews. Новітні технології. Майбутнє – за голосовими технологіями [Електронний ресурс]. URL: http://ua.euronews.com/2013/12/12/the-voice-is-the-future/
25. Jiang L. Smart home research / Jiang L., Liu D. Y., Yang B. // Proceedings of the 2004 International Conference on Machine Learning and Cybernetics. – Shanghai, China. – 2004. – V. 2. – P. 659-663.
26. Noury N. New trends in health smart homes / Noury N., Virone G., Barralon P., Ye J., Rialle V., Demongeot J. // Proceedings of the 5th International Workshop on Enterprise Networking and Computing in Healthcare Industry (Healthcom'03) June 6-7, 2003. – 2003. – P. 118-127.

27. Helal S. The gator tech smart house: a programmable pervasive space / Helal S., Mann W., El-Zabadani H., King J., Kaddoura Y., Jansen E. // Computer. – 2005. – V. 38, No. 3. – P. 50-60.
28. Chan M. A review of smart homes-present state and future challenges / Chan M., Estève D., Escriba C., Campo E. // Computer Methods and Programs in Biomedicine. – 2008. – V. 91, No. 1. – P. 55-81.
29. Гололобов В. Н. "Умный дом" своими руками / В.Н.Гололобов. – Москва : НТ Пресс, 2007. – 416 с.
30. Briere D. Hurley Smart Homes For Dummies, Third Edition / Danny Briere. – John Wiley & Sons, 2011. – 432 p.
31. Mahmoud A. Smart Home Systems / Mahmoud A. Al-Qutayri. – Publisher: InTech, 2010. – 194 p.
32. Harper R. Inside the Smart Home / Harper R. – London : Springer, 2003. – 275 p.
33. Niezabitowska E. Budynek inteligentny – Tom I, II Potrzeby użytkownika a standard budynku inteligentnego / Niezabitowska E. – Gliwice : Politechniki Śląskiej, 2005.
34. Элсенпитер Р. Умный Дом строим сами / Роберт К. Элсенпитер, Тоби Дж. Велт. – М. : Кудиц – Образ, 2005. – 384 с.
35. Волошин О. "Home Smart Home" журнал Компьютерра №18 13.05.2008 / О. Волошин
36. Building Intellectualisation system market research, Techart marketing group, 01.04.2008.
37. Заборский Г. "Умный дом" и проблемы развития / Г. Заборский // Архитектура и строительство : №7 (206). – 2009.
38. Tesla N. Method of and apparatus for controlling mechanism of moving vessels and vehicles / Tesla Nikola // Patent 613809. – United States Patent and Trademark Office, 8 November 1898.
39. Gerhart J. Home Automation and Wiring / Gerhart James. – McGraw-Hill Professional, 1999.

40. Mann W. The state of the science / Mann William C // Smart technology for aging, disability and independence. – John Wiley and Sons, 2005.
41. Anogianakis G. Advancement of assistive technology / Anogianakis G., Buhler C., Soese M. – IOS Press, 1997. – 400 p.
http://www.kievbud.in.ua/index.php/articles/189-27154249
42. Xiao. J. The Design and Implementation of an Energy-Smart Home in Korea / J. Xiao, R. Boutaba // Journal of Computing Science and Engineering. – Korean Institute of Information Scientists and Engineers, 2013. – V. 7, No. 3. – P. 204-210.
43. Теслюк В. М. Використання технологій розумного будинку для поліпшення енергетичної ситуації в Україні / Теслюк В. М., Денисюк П. Ю., Береговська Х. В. // Матеріали XIII-го міжнародного наукового семінару "Сучасні проблеми інформатики в управлінні, економіці, освіті". – Київ, 2014. – С. 215-219.
44. Аналітична довідка щодо розгляду на засіданні науково-технічної ради Департаменту житлово-комунальної інфраструктури питання: "Про розроблення програми реалізації концепції "Інтелектуальне місто (енергетична складова)" для міста Києва" [Електронний ресурс]. URL: http://dzki.kievcity.gov.ua//content/pro-rozroblennya-programy-realizacii-koncepcii-intelektualne-misto
45. Інтелектуальне вуличне світлодіодне освітлення – перший крок до "розумного" міста [Електронний ресурс]. URL: http://www.rada.cherkassy.ua/ua/newsread.php?view=11191&s=1&s1=17
46. Українці готові інвестувати від $ 5 до $ 30 тисяч в "розумні будинки" [Електронний ресурс]. URL: http://vkurse.ua/ua/business/gotovy-investirovat-ot-5-do-30-tysyach-v-umnye-doma.html
47. Полякова О. В. Особливості впровадження систем енергозбереження та інтелектуального керування житловим середовищем в Україні / Полякова О. В., Сафронов О. О. // Обладнання, електротехнічні та автоматизовані системи і комплекси. – Київ, 2014. – №5 (79). – С. 57-63.
48. Грицик В. В. Методологія системного проектування нейрокомп'ютерних засобів мобільних робототехнічних систем / Грицик В. В., Цмоць І. Г.,

Теслюк В. М. // Доповіді Національної академії наук України. – Київ, 2013. – № 1. – С. 30-36.

49. Tsmots I. Hardware and Software tools for motion control of mobile robotic system / Ivan Tsmots, Vasyl Teslyuk, Iryna Vavruk // Proceedings of the 12th International Conference on The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM'2013), February 19-23, 2013. – Lviv : Lviv Polytechnic Publishing House, 2013. – P. 368.
50. Vavruk I. The selection of construction principles and intelligent technologies for mobile robotic systems implementation / Vavruk I., Tsmots I., Teslyuk V // Proceeding of the 9th International Conference "Perspective Technologies and Methods in MEMS Design" (MEMSTECH'2013), April 16-20, 2013. – Lviv : Lviv Polytechnic Publishing House, 2013. – P. 179.
51. Теслюк В. М. Структура та реалізація колісною робототехнічною системою / Теслюк В. М., Ваврук І. Є., Цмоць І. Г., Ткаченко Р. О. // Матеріали міжнародної наукової конференції "Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту" (ISDMCI'2013), May 20-24, 2013. – Херсон: ХНТУ, 2013. – С. 302-303.
52. Цмоць І. Г. Моделювання інтелектуального управління рухом мобільної роботехнічної системи / Цмоць І. Г., Ваврук І. Є., Теслюк В. М. // Науковий вісник НЛТУ України : Збірник науково-технічних праць. – Вип. 23.15. – Львів : РВВ НЛТУ України, 2013. – С. 290-300.
53. Цмоць І. Г. Оцінювання складності методу управління робототехнічною системою на базі нечіткої логіки за кількістю операцій / Цмоць І. Г., Теслюк В. М., Ваврук І. Є. // Збірник наукових праць ІППМЕ ім. Г. Є. Пухова НАН України. – Вип. 69. – Київ, 2013. – С.114-119.
54. Кухонна продукція фірми Samsung Electronics [Електронний ресурс]. URL: http://www.samsung.com/ua/consumer/home-appliances/refrigerator/side-by-side/RF905QBLAXW/WT
55. Кухонна продукція фірми Siemens [Електронний ресурс]. URL: http://siemens-home.com/Files/SiemensNew/Ru/ru/idos/idos.html

56. Boreiko O. Structural Model Of Passenger Counting And Public Transport Tracking System Of Smart City / Oleh Boreiko, Vasyl Teslyuk // Proceeding of the 12th International Conference “Perspective Technologies and Methods in MEMS Design” (MEMSTECH’2016), April 20-24, 2016. – Lviv : Lviv Polytechnic Publishing House, 2016 – P. 124-126.
57. Система керування інтелектуальним містом [Електронний ресурс]. URL: http://nure.ua/uk/university/structure/science/innovacijni-proekti-i-rozrobki/2-5-sistema-keruvannya-intelektualnim-mistom/
58. Гайков А. Р. Інтелектуальні транспортні системи в Україні / А. Р. Гайков, О. П. Євсєєва, О. В. Баранов, В. Ю. Баранов // Вісник НТУ “ХПІ”. Серія: Автомобіле та тракторобудування. – Харків : НТУ “ХПІ”, 2014. – № 9 (1052). – С. 106-112.
59. Byun J. H. Smart city implementation models based on IoT(Internet of Things) technology / J.H. Byun, S.Y. Kim, J.H. Sa, Y.T. Shin, S.P. Kim and J.B. Kim // Proceedings of Advanced Science and Technology Letters. – 2016. – V.129. – P. 209-212.
60. Gauer A. Smart city architecture and its applications based on IoT / A. Gauer, B. Scotney, G. Parr, and S. McClean // Procedia Computer Science. – 2015. – V.52. – P. 1089-1094.
61. Park Y. Analysis on Smart City service technology with IoT / Y. Park, S. Rue // Korea institute of information Technology Review. – 2015. – V. 13, Is. 2. – P. 31-37.
62. Bagula A. On the design of smart parking networks in the smart cities: An optimal sensor placement model / A. Bagula, L. Castelli, M. Zennaro // Sensors. – 2015. – V. 15, Is. 7. – P. 15443-15467.
63. Nowicka K. Smart City logistics on cloud competing model / K. Nowicka // Procedia-Social and Behavioral Sciences. – 2014. – V. 151. – P. 266-281.
64. “Розумний будинок” – економія чи дорога іграшка [Електронний ресурс]. URL: http://sofit.com.ua/articles/rozumnij_budinok_ekonom_ya_chi_doroga_grashka

65. “Розумний будинок” з інтелектуальною начинкою [Електронний ресурс]. URL: http://alls.in.ua/12199-rozumnijj-budinok-z-intelektualnoyu-nachinkoyu.html
66. Сколько стоит “умный дом” и что он умеет [Електронний ресурс]. URL: http://www.segodnya.ua/economics/realty/sovremennye-tehnologi-kotorye-pomogayut-domu-stat-umnee-473319.html
67. Hi-Tech House. Публикации в СМИ. Інтелектульний дім [Електронний ресурс]. URL: http://hi-tech-house.com/mass-media/000001/
68. Державна архітектурно-будівельна інспекція України. Гірник М.А.: Інтелектуальна споруда – інтегрована інформаційна система [Електронний ресурс]. URL: http://www.dabi.gov.ua
69. Boreiko O. Model of Telecommunication Networks for Intelligent Building / Boreiko O., Teslyuk V., Beregovska C., Mykhailiuk A. // Proceeding of the 13th International Conference on The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM’2015), February 24-27, 2015. – Lviv : Lviv Polytechnic Publishing House, 2015. – P. 349-351.
70. Boreiko O. Video Monitoring System’s Module for Smart Home / Boreiko Oleg, Teslyuk Vasyl, Berezsky Oleg, Beregovska Cristina // Proceeding of the 10th International Conference “Perspective Technologies and Methods in MEMS Design” (MEMSTECH’2014), June 22-24, 2014. – Lviv : Lviv Polytechnic Publishing House, 2014. – P. 126.
71. Борейко О. Ю. Розроблення компонентів системи відеонагляду “Інтелектуального будинку” на базі Raspberry PI / Борейко О. Ю., Теслюк В. М., Березький О. М. // Моделювання та інформаційні технології : Збірник наукових праць. – Вип. 71. – Київ : Інститут моделювання в енергетиці ім. Г. Є. Пухова НАН України, 2014. – С. 66-71.
72. Теслюк В. М. Модель телекомукаційної мережі “інтелектуального будинку” / Теслюк В. М., Борейко О. Ю., Сидор А. Р., Береговська Х. В. // Науковий вісник НЛТУ України : Збірник науково-технічних праць. – Вип. 26.1. – Львів : РВВ НЛТУ України, 2016. – С. 351-357.

73. Субботін С.О. Подання й обробка знань у системах штучного інтелекту та підтримки прийняття рішень: Навчальний посібник / Субботін С.О. – Запоріжжя: ЗНТУ, 2008. – 341 с.
74. Грэхем И. Объектно-ориентированные методы. Принципы и практика / Иан Грэхем. – Вильямс, 2004. – 880 с.
75. Миркес Е. М. Нейроинформатика: Учеб. пособие для студентов / Е. М. Миркес. – Красноярск: ИПЦ КГТУ, 2002. – 347 с.
76. Бідюк П. І. Комп'ютерні системи підтримки прийняття рішень / П. І. Бідюк, О. П. Гожий, Л. О. Коршевнюк. – Київ, 2012. – 379 с.
77. Лєсна Н. С. Інтелектуальний аналіз даних / Н. С. Лєсна, В. Б. Рєпка, Т. Б. Шатовська. – Харків : Харківський національний університет радіоелектроніки, 2003. – 110 с.
78. Davies K. H. Automatic Speech Recognition of Spoken Digits / Davies K. H., Biddulph R., Balashek S. // J. Acoust. Soc. Am. – 1952. – V.24, No. 6. – P. 637-642
79. Ли У. Методы автоматического распознавания речи / Ли У. – Москва : Мир, 1983. – 328 с.
80. Kasabov N. Introduction: Hybrid intelligent adaptive systems / N. Kasabov // International Journal of Intelligent Systems. – 1998. – V.6. – P. 453-454.
81. Ємельянов В. В. Теория и практика эволюционного моделирования / Ємельянов В. В., Курейчик В. В., Курейчик В. М. – Москва : ФИЗМАТЛИТ, 2003. – 432 с.
82. Shoham Y. Algorithmic, Game-Theoretic, and Logical Foundations / Y. Shoham, K. Leyton-Brown. – London : Cambridge University Press, 2009. – 513 p.
83. Оссовский С. Нейронные сети для обработки информации / Станислав Осовский. – Москва : Финансы и статистика, 2002. – 344 с.
84. Бублик Б. Н. Основы теории управления / Бублик Б. Н., Кириченко Н. Ф. – Київ : Вища школа, 1975. – 328 с.
85. Глушков В. Енциклопедія кібернетики / В. Глушков. – Київ : Головна редакція Української радянської енциклопедії. – 1973.

86. Кватер Т. Нейромережеві інформаційні технології контролю та діагностики динамічних об’єктів в умовах невизначеності / Кватер Тадеуш. – Львів : Видавництво Тараса Сороки, 2005. – 270с.
87. Комашинский В. И. Нейронные сети и их применение в системах управления и звязи / В. И. Комашинский, Д. А. Смирнов. – Москва : Горячая Линия-Телеком, 2003. – 94с.
88. Терехов В. А. Нейросетевые системы управления / Терехов В. А., Ефимов Д. В., Тюкин И. Ю. – Москва : Радиотехника, 2002. – 480 с.
89. Форсайт Д. Компьютерное зрение. Современный подход / Дэвид Форсайт, Жан Понс. – Москва : Издательський дом “Вильямс”, 2004. – 928 с.
90. Боюн В. П. Інтелектуальні відео системи та пристрої реального часу / Боюн В. П. // МК Інтелектуальні системи прийняття рішень та прикладні аспекти інформаційних технологій. – Київ, 2007. – С. 101-107.
91. Бэстенс Д. Э. Нейронные сетии финансовые рынки: принятие решений в торговых операциях / Бэстенс Д. Э., Ванденберг В. М., Вуд Д. – Москва : ТВП, 1997. – 236 с.
92. Ежов А. А. Нейрокомпьютинг и его применения в экономике и бизнесе / Ежов А. А., Шумский С. А. – Москва : МИФИ, 1998. – 224 с.
93. Чабан В. Діаґностика електромаґнетного кола за допомогою штучної нейронної мережі / Чабан В., Кватер Т., Бартман Я. // Електромеханіка. Теорія і практика (Праці науково-технічної конференції, присвяченої 100-річчю Т. Губенка), 25-28 вересня, 1996. – Львів – Славськ. – С. 191-193.
94. Чабан А. Застосування штучних нейронних мереж для аналізу електромеханічних систем / Чабан А. // Научные журналы НТУ “ХПИ” : №1 : Электротехника и электромеханика. – Харків : НТУ “ХПИ”, 2008.
95. Круг П.Г. Нейронные сети и нейрокомпьютеры: Учебное пособие по курсу “Микропроцессоры” / П.Г. Круг. – Москва : МЭИ, 2002. – 176 с.
96. Wasserman P. D. Neural computing theory and practice / Wasserman P. D. – New York : Van Nostrand Reinhold Co., 1989. – P. 230.

97. Rosenblatt F. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain / Rosenblatt F. // Cornell Aeronautical Laboratory, Psychological Review. – 1958. – V. 65, No. 6. – P. 386-408.
98. Розенблатт Ф. Принципы нейродинамики: перцептроны и теория механизмов мозга / Розенблатт Ф. – Москва : Мир, 1965. – 473 с.
99. Mohamad H. H. Fundamentals of Artificial Neural Networks / Mohamad H. Hassoun. – London : MIT Press, 1995. – 511 p.
100. Новотарський М. А. Штучні нейронні мережі: Обчислення / М. А. Новотарський, Б. Б. Нестеренко // Праці Інституту математики НАН України. – Т. 50. – Київ : Інститут математики НАН України, 2004. – 408 с.
101. Апостолюк В. О. Інтелектуальні системи керування / Апостолюк В. О., Апостолюк О. С. // Конспект лекцій НТУУ “КПІ”. – Київ, 2008. – 88 с.
102. Hopfield J. J. Neural networks and physical systems with emergent collective computational abilities / J. J. Hopfield // Proceedings of National Academy of Sciences. – 1982. – V. 79, N. 8. – P. 2554-2558.
103. Уоссермен Ф. Нейрокомпьютерная техника: Теория и практика / Уоссермен Ф. – Москва : Мир, 1992. – 240 с.
104. Нейронна мережа на основі моделі “Функціонал на множині табличних функцій” [Електронний ресурс]. URL: http://www.victoria.lviv.ua/html/neural_nets/Lecture4.htm
105. Ткаченко Р. Нейроподібні структури машини геометричних перетворень у завданнях інтелектуального аналізу даних / Ткаченко Р., Дорошенко А. // Вісник Національного університету “Львівська політехніка” : № 638 : Комп'ютерні науки та інформаційні технології. – Львів, 2009. – С. 179-184.
106. Томашевський О. М. Інформаційні технології та моделювання бізнес-процесів / О. М. Томашевський, Г. Г. Цегелик, М. Б. Вітер, В. І. Дубук. – Київ : ЦУЛ, 2012. – 296 с.
107. Штучні нейронні мережі [Електронний ресурс]. URL: http://archive.is/jt0yH#selection-9.0-9.22

108. Організація інтелектуальних обчислень [Електронний ресурс]. URL: http://www.victoria.lviv.ua/html/oio/index.html
109. Класифікація відомих нейромереж по основних категоріях застосування [Електронний ресурс]. URL: http://www.victoria.lviv.ua/html/oio/html/theme7.htm
110. Вибір мережі [Електронний ресурс]. URL: http://www.uatur.com/html/neural_nets/Lecture3.htm
111. ABB [Електронний ресурс]. URL: http://www.abb.ua
112. TESLI [Електронний ресурс]. URL: http://www.tesli.com/ru/products/knx-eib-lon/abb/
113. Siemens [Електронний ресурс]. URL: https://eb.automation.siemens.com/mall/ru/ru/Catalog/Products/2449999
114. Inteldome [Електронний ресурс]. URL: http://www.inteldome.com.ua/company
115. IntelCity [Електронний ресурс]. URL: http://intelcity.com.ua/index.php
116. Hi-Tech House. Умный дом от Domintell. Краткое описание [Електронний ресурс]. URL: http://hi-tech-house.com/smart-home/
117. AMX [Електронний ресурс]. URL: http://www.amxrus.ru/about/
118. СИС-Сервис [Електронний ресурс]. URL : http://crestron.spb.ru/sample-sites-2/umniy-dom
119. Інформаційний портал електротехнічної галузі [Електронний ресурс]. URL: http://www.electrotema.com.ua/ru/content/detail/3181
120. Elchelon [Електронний ресурс]. URL: http://www.echelon.com/
121. C-Bus [Електронний ресурс]. URL: http://c-bus.ru/index.php/pro/about-cbus
122. BACnet [Електронний ресурс]. URL: http://www.bacnet.ru/about_BACnet/
123. Электроустановочные изделия Gira для умных домов [Електронний ресурс]. URL: http://download.gira.com/data2/192288900311.pdf
124. Legrand [Електронний ресурс]. URL: http://legrand.com.ua/index.php?categoryID=341
125. Норенков И. П. Основы автоматизированного проектирования / Норенков И.П. – Москва : Издательство МГТУ им. Н. Э. Баумана, 2002. – 336 с.

126. Теслюк В. М. Моделі та інформаційні технології синтезу мікроелектромеханічних систем / В. М. Теслюк // Монографія. – Львів : Вежа і Ко, 2008. – 192 с.
127. Теслюк В. М. Методи проектування мікроелектромеханічних систем / В. М. Теслюк // АСУ и приборы автоматики : Всеукраинский межведомственный научно – технический сборник. – Вып. 134. – Харьков : ХТУРЭ, 2006. – С. 82-89.
128. Лобур М. В. Методи та моделі для наскрізного проектування вбудованих систем / М. В. Лобур. – Київ : НТУУ "КПІ", 2004. – 32 с.
129. Xu M. Design and Implementation of a Wireless Sensor Network for Smart Homes / M. Xu, L. Ma, F. Xia, T. Yuan, J. Qian, M. Shao // Ubiquitous Intelligence & Computing and 7th International Conference on Autonomic & Trusted Computing (UIC/ATC), Ocrober 26-29, 2010. – 2010. – P. 239 – 243.
130. Sriskanthan N. Bluetooth based home automation system / N. Sriskanthan, F. Tan, A. Karande // Microprocessors and Microsystems. – 2002. – V. 26. – P. 281-289.
131. Piyare R. Bluetooth Based Home Automation System Using Cell Phone / R. Piyare, M. Tazil // IEEE 15th International Symposium on Consumer Electronics, June 14-17, 2011. – 2011. – P. 192-195.
132. Mowad M. A. L. Smart Home Automated Control System Using Android Application and Microcontroller / M. A. L. Mowad, A. Fathy, A. Hafez // International Journal of Scientific & Engineering Research. – 2014. – V. 5, No. 5. – P. 935-939.
133. Yuan D. The design of smart home monitoring system based on WiFi electronic trash / D. Yuan, S. Fang, Y. Liu // Journal of Software. – 2014. – V. 9, No. 2. – P. 425-428.
134. Liang L. Design and implementation of wireless Smart-home sensor network based on ZigBee protocol / L. Liang, L. Huang, X. Jiang, Y. Yao // International Conference "Communications, Circuits and Systems" (ICCCAS'2008), October 14-17, 2008. – 2008. – P. 434-438.

135. Lee H. Ontology Model-based Situation and Socially-Aware Health Care Service in a Smart Home Environment / Lee H., Kwon J. // International Journal of Smart Home. – 2013. – V. 7, No. 5. – P. 239-250.
136. Azza K. Modeling, Simulation, and Control of Smart Homes Using Petri Nets / Azza K. Nabih, Mostafa M. Gomaa, Hossam S. Osman, Gamal M. Aly // International Journal of Smart Home. – 2011. – V. 5, No. 3.
137. Star – HSPICE, Avant. Corp., Fremont, CA. [Електронний ресурс]. URL: http://www.avanticorp.com.
138. Влах И. Машинные методы анализа и проектиравания электронных схем / И. Влах, К. Сингхал. – Москва : Радио и связь, 1988. – 560 с.
139. Hamed B. Design & Implementation of Smart House Control Using LabVIEW / B. Hamed // International Journal of Soft Computing and Engineering (IJSCE). – 2012. – V. 1, No. 6.
140. Сніжко Є. М. Моделювання дискретно-аналогових систем мовою VHDL-AMS / Є. М. Сніжко, М. М. Мілих. – Дніпропетровськ : РВВ ДНУ, 2008. – 72 с.
141. Simplorer 7 VHDL-AMS Tutorial [Електронний ресурс]. URL: http://www.etti.tuiasi.ro/pac/LDH/LDH_Laborator/Simplorer_VHDLAMS_Tutorial.pdf
142. ANSYS/Multiphysics ver. 5.5, Ansys, Inc., Canonsburg, PA [Електронний ресурс]. URL: http://www.ansys.com.
143. Robles R. J. Applications, Systems and Methods in Smart Home Technology: A Review / R. J. Robles, T. H. Kim // International Journal of Advanced Science and Technology. – 2010. – V. 15.
144. CFD – ACE+ and add – on modules [Електронний ресурс]. URL: http://www.cfdrc.com.
145. Zhai Y. Design of smart home remote monitoring system based on embedded system / Y. Zhai, X. Cheng // IEEE 2nd International Conference “Computing, Control and Industrial Engineering” (CCIE’2011), September 9, 2011. – P. 41-44.
146. Теорія множин і комбінаторика [електронний ресурс]. URL: http://mcomb.ks8.ru/index.php.

147. Стеценко І.В. Моделювання систем: навч.посіб. / І.В.Стеценко // М-во освіти і науки України, Черкас. держ. технолог. ун-т. – Черкаси: ЧДТУ, 2010. – 399 с.
148. Оре О. Графы и их применение / Оре О. – Москва : Мир, 2002. – 175 с.
149. Скінченні автомати [електронний ресурс]. URL: http://oim.asu.kpi.ua/files/DM/38_Final_automates.pdf
150. Питерсон Дж. Теория сетей Петри и моделирование систем / Питерсон Дж. – Москва : Мир, 1984. – 264 с.
151. Котов В.Е. Сети Петри / В.Е. Котов. – Москва : Наука, 1984 – 160 с.
152. Классификация сетей Петри. [електронний ресурс]. URL: http://minska.ru/klassifikaciya-setej-petri.html.-17.03.2011.
153. Мережі Петрі [електронний ресурс]. URL: http://lib.lntu.info/books/knit/ki/2010/10-127/page10.html
154. Мурашко А. Г. Первое знакомство с сетями Петри / Мурашко А. Г. // Учебное пособие. – Киев : УМК ВО, 1988. – 71 с.
155. Зайцев Д.А. Инварианты временных сетей Петри / Д. А. Зайцев // Кибернетика и системный анализ. – 2004. – №2. – С. 92-106.
156. Ординарные сети [електронний ресурс]. URL: http://minska.ru/ordinarnye-seti.html
157. Перспективи ринку систем "Розумний будинок" [Електронний ресурс]. URL: http://alls.in.ua/17818-perspektivi-rinku-sistem-rozumnijj-budinok.html
158. Системи безпеки “Інтелектуального будинку” [Електронний ресурс]. URL: http://dim.promotion-soft.com/bud-remont-2012-07-07-5508/
159. Peterson J. L. A Note on Colored Petri Nets / James L. Peterson // Information Processing Letters. – 1980. – V. 11, No. 1. – P. 40-43.
160. Хайкин С. Нейронные сети: полный курс / Саймон Хайкин. – Москва : Вильямс, 2006. – 1104 с.
161. Teslyuk V. Schematic Model of Protection and Lighting Subsystems for Analysis of Intellectual House / Teslyuk V., Beregovska C. // Proceedings of the 12th International Conference on The Experience of Designing and Application of CAD

Systems in Microelectronics (CADSM’2013), February 19-23, 2013. – Lviv : Lviv Polytechnic Publishing House, 2013. – P. 436-437.

162. Teslyuk V. Developing Information Model Of The Reachability Graph / Teslyuk V., Denysyuk P., Hamza Ali Yousef Al Shawabkeh, Kernytskyy A. // Proceedings of the 15th International Seminar : Workshop Of Direct And Inverse Problems Of Electromagnetic And Acoustic Wave Theory (DIPED’2010), September 27-30, 2010. – Tbilisi, 2010. – P. 210-214.
163. Teslyuk V. The formalization of the MEMS automated design process by usage of Petri Networks / Teslyuk V., Hamza Al-Shavabkeh, Pereyma M., Al Omari Tarik // Proceedings of the 3rd International Conference of Young Scientists (MEMSTECH’2007), May 23-26. – Lviv : Lviv Polytechnic Publishing House, 2007. – P. 133-134.
164. Питерсон Дж. Теория сетей Петри и моделирование систем / Питерсон Дж. – Москва : Мир, 1984. – 435 с.
165. Qin N. A foundational ontology-based model for human activity representation in smart homes / Ni Qin, Pau de la Cruz, Iván García Hernando, Ana Belén // Journal of Ambient Intelligence and Smart Environments. – 2016. – V. 8, No. 1. – P. 47-61.
166. Уилсон Р. Введение в теорию графов / Уилсон Р. – Москва : Мир, 1977. – 208с.
167. Татт У. Теория графов / Татт У. – Москва : Мир, 1988. – 424 с.
168. Fruchterman T. M. J. Graph Drawing by Force-Directed Placement / Fruchterman T. M. J., Reingold E. M. // Software-Practice and Experience. – 1991. – V. 21, No. 11. – P. 1129-1164.
169. Kamada T. An algorithm for drawing general undirected graphs / Kamada T. Kawai S. // Information Processing Letters. – 1989. –V. 31. – P. 7-15.
170. Frick A. Fast Adaptive Layout Algorithm for Undirected Graphs / Frick A., Ludwig A., Mehldau H. // Springer-Verlag. – 1995. –V. 894. – P. 388-403.
171. Штогрина Е. С. Метод визуализации метаграфа / Е. С. Штогрина, А. С. Кривенко // Научно-технический вестник информационных технологий, механики и оптики Санкт-Петербургского национального исследовательского Университета ИТМО, №3 (91), 2014. – С. 126-132.

172. Globa L. Based on force-directed algorithms method for metagraph visualization / L. Globa, M. Temovoy, O. Shtogrina, O. Kryvenko // Soft Computing in Computer and Information Science The series “Advances in Intelligent and Soft Computing” (ACS), Springer, Vol. 342, 2015. – P. 359-369.
173. Теслюк В. М. Автоматизація проектування мікроелектромеханічних систем на компонентному рівні: Монографія / Теслюк В. М., Денисюк П. Ю. – Львів : Видавництво “Львівська політехніка”, 2011. – 192 с.
174. Теслюк В.М. Модель підсистеми клімат контролю для аналізу роботи інтелектуального будинку / Теслюк В. М., Теслюк Т. В., Ляпандра А. С. // Науковий вісник НЛТУ України : Збірник науково-технічних праць. – Вип. 22.9. – Львів : РВВ НЛТУ України, 2012. – С. 132-135.
175. Jensen K. Coloured Petri Nets: modelling and validation of concurrent systems / Kurt Jensen, Lars M. Kristensen. – London : Springer, 2009. – 395 p.
176. Гірник М. А. “Інтелектуальна споруда” – інтегрована інформаційна система [Електронний ресурс]. URL: http://www.dabi.gov.ua
177. Kis Y. P. Methods and tools of authentication biometric data in information systems / Kis Y. P., Teslyuk V. M. // Actual Problems of Economics. – Lviv, 2012. – № 12 (138). – P. 174-182.
178. Grynyk O. System for Automation Testing Components of the Smart Home / Grynyk O., Denysyuk P., Teslyuk V. // Proceedings of the 7th International Conference of Computer Science & Information Technologies (CSIT’2012), April 14-19, 2012. – Lviv: Publishing House Vezha&Co, 2012. – P. 44-46.
179. Теслюк В. М. Інформаційна модель графа досяжності / В. М. Теслюк, Хамза Алі Юсеф Альшавабкех // Збірник наукових праць ІППМЕ ім. Г. Є. Пухова НАН України. – Вип. 58. – Київ, 2010. – С. 166-171.
180. Тесленко В. А. Датчики в системах сбора данных и управления / В. А. Тесленко // Промышленные измерения, контроль, автоматика, диагностика (ПиКАД). – 2004. – №2. – С. 50-56.

181. Ghayvat H. WSN- and IOT-Based Smart Homes and Their Extension to Smart Buildings / H. Ghayvat, S. Mukhopadhyay, X. Gui, N. Suryadevara // Sensors. – 2015. – V. 15. – P. 10350-10379.
182. Ding D. Sensor technology for smart homes / D. Ding, R. A. Cooper, P. F. Pasquina, L. F. Pasquina // Maturitas the European Menopause Journal. – 2011. – V. 69, No. 2. – P. 131-136.
183. Перепелица В. А. Дискретная оптимизация и моделирование в условиях неопределенности данных / В. А. Перепелица, Ф. Б. Тебуева – Москва : Академия естествознания, 2007.
184. Ясницкий Л.Н. Искусственный интеллект / Ясницкий Л.Н., Черепанов Ф.М. // Методическое пособие – Москва : БИНОМ. Лаборатория знаний, 2011. – 216 с.
185. Чернодуб А. М. Вибір нейроемулятора на основі методу керуючих локальних градієнтів у методі нейроуправління з еталонною моделлю / Чернодуб А. М. // Математичні машини і системи. – Вип. 3. – Київ : Інститут проблем математичних машин і систем Національної академії наук, 2012. – С. 61-68.
186. Вікіпідручник [Електронний ресурс]. URL: http://uk.wikibooks.org/wiki/
187. Норенков И. П. САПР : Системы автоматизированного проектирования. Принципы построения и структура / Норенков И. П. – Москва : Издательство МГТУ имени Н. Э. Баумана, 1987. – 123 с.
188. Світличний О. О. Основи геоінформатики. Навчальний посібник / Світличний О. О. Плотницький С. В. – Суми : ВТД “Університетська книга”, 2006. – 295 с.
189. Жоголев Е. А. Лекции по технологии программирования: Учебное пособие / Жоголев Е. А. – Москва : Издательский отдел факультета ВМиК МГУ, 2001. – 2016 с.
190. Denysyuk P. XML application for microfluidic devices description / Denysyuk P., Teslyuk V., Khimich I., Farmaga I. // Proceedings of 9th International Conference on the Experience of Designing and Application of CAD Systems in Microelectronics (CADSM’2013), February 19-23, 2013. – Lviv : Lviv Polytechnic Publishing House, 2013. – P. 567-569.

191. Галіцин В. К. Програмні оболонки і пакети / Галіцин В. К., Сидоренко Ю. Т. – Київ : КНЕУ, 2003. – 212 с.
192. Жидков А. Культура программирования [Електронний ресурс]. URL: http://www.javaportal.ru/articles/culture_of_programming.html
193. Лавріщева К. М. Програмна інженерія /Лавріщева К. М. – Київ : Академперіодика, 2008. – 319 с.
194. Сергеев А. П. HTML и XML. Профессиональная работ / Сергеев А. П. – Москва : Диалектика-Вильямс, 2004. – 880 с.
195. Элсенпитер Р. К. Умный Дом строим сами / Роберт К. Элсенпитер, Тоби Дж. Велт. – Москва : Кудиц-Образ, 2005. – 384 с.
196. Тимощук П. Основи теорії проектування нейронних мереж / Тимощук П., Лобур М. – Львів : Видавництво “Львівська політехніка”. – 2007. – 328 с.
197. Буч Г. Объектно-ориентированное проектирование с примерами применения / Буч Г. – Москва : Конкорд, 1996. – 519 с.
198. XML [Електронний ресурс]. URL: http://uk.wikipedia.org/ wiki/XML.
199. Денисюк П. Ю. Застосування XML-формату для опису конструкцій гідравлічних МЕМС / П. Ю. Денисюк // Збірник наукових праць УАД “Комп’ютерні технології друкарства”. – Львів, 2007. – №17. – С. 93-99.
200. Бублик В. В., Личман В. В., Обвінцев О. В. Конспект лекцій з программування на мовах C++ та Pascal для студентів 1-2-го курсів. Конспект лекцій “Інформатика та програмування”. [Електронний ресурс]. URL: www.univ.kiev.ua.
201. Датчик дыма MQ-2 [Електронний ресурс]. URL: http://arduino-ua.com/prod188-Datchik_dima_MQ-2
202. Датчик огня от DFRobot [Електронний ресурс]. URL: http://arduino-ua.com/prod299-Datchik_ognya

Додаток А

Приклад коду програми, яка реалізує нейрофункції для підсистеми клімат-контролю

```
#include <dht11.h> //підключення бібліотеки для роботи з давачем

dht11 DHT11; //створення екземпляру класу для давача

#define DHT11PIN 2 //визначення порта, з якого зчитується сигнал від давача

double T = 0.0; //змінна в якій записується поточна температура
double H = 0.0; //змінна в якій записується поточна вологість


const int netSize = 12; //кількість нейронів в мережі
const int cntIn = 2; //кількість вхідних нейронів
const int cntOut = 3; //кількість вихідних нейронів

        //- - - - - NET parameters - - - - - - - - - - - - - - - - - - - - - - -

        char type[netSize][netSize] = //матриця зв'язків між нейронами (n – немає зв'язку, o – вихідне ребро, i – вхідне ребро)
        {
                {'n', 'n', 'n', 'o', 'o', 'o', 'o', 'o', 'n', 'n', 'n', 'n'},
                {'n', 'n', 'n', 'o', 'o', 'o', 'o', 'o', 'n', 'n', 'n', 'n'},
                {'n', 'n', 'n', 'o', 'o', 'o', 'o', 'o', 'n', 'n', 'n', 'n'},
                {'i', 'i', 'i', 'n', 'n', 'n', 'n', 'n', 'n', 'o', 'o', 'o'},
                {'i', 'i', 'i', 'n', 'n', 'n', 'n', 'n', 'n', 'o', 'o', 'o'},
                {'i', 'i', 'i', 'n', 'n', 'n', 'n', 'n', 'n', 'o', 'o', 'o'},
                {'i', 'i', 'i', 'n', 'n', 'n', 'n', 'n', 'n', 'o', 'o', 'o'},
                {'i', 'i', 'i', 'n', 'n', 'n', 'n', 'n', 'n', 'o', 'o', 'o'},
                {'n', 'n', 'n', 'n', 'n', 'n', 'n', 'n', 'n', 'o', 'o', 'o'},
                {'n', 'n', 'n', 'i', 'i', 'i', 'i', 'i', 'i', 'n', 'n', 'n'},
                {'n', 'n', 'n', 'i', 'i', 'i', 'i', 'i', 'i', 'n', 'n', 'n'},
                {'n', 'n', 'n', 'i', 'i', 'i', 'i', 'i', 'i', 'n', 'n', 'n'}
        };

        double value[netSize][netSize] = //матриця суміжності(відображено ваги зв'язків)
        {
        {0.0000, 0.0000, 0.0000, 2.0156, -0.8100, 3.2359, 1.1984, 0.5179, 0.0000, 0.0000, 0.0000, 0.0000},
        {0.0000, 0.0000, 0.0000, 0.5459, -1.1247, -0.7054, -0.9870, -2.8024, 0.0000, 0.0000, 0.0000, 0.0000},
        {0.0000, 0.0000, 0.0000, -0.8159, 1.2048, -0.4461, -0.0313, 0.3071, 0.0000, 0.0000, 0.0000, 0.0000},
        {2.0156, 0.5459, -0.8159, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, -0.7920, 2.3595, -0.8381},
        {-0.8100, -1.1247, 1.2048, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, -0.5456, -0.7423, -0.7051},
        {3.2359, -0.7054, -0.4461, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, -0.6181, -1.9484, 0.4867},
        {1.1984, -0.9870, -0.0313, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.7170, 0.1019, -0.4017},
        {0.5179, -2.8024, 0.3071, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, -0.1512, 0.9573, 1.0055},
        {0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.7314, 1.1097, 0.9167},
        {0.0000, 0.0000, 0.0000, -0.7920, -0.5456, -0.6181, 0.7170, -0.1512, 0.7314, 0.0000, 0.0000, 0.0000},
        {0.0000, 0.0000, 0.0000, 2.3595, -0.7423, -1.9484, 0.1019, 0.9573, 1.1097, 0.0000, 0.0000, 0.0000},
        {0.0000, 0.0000, 0.0000, -0.8381, -0.7051, 0.4867, -0.4017, 1.0055, 0.9167, 0.0000, 0.0000, 0.0000}
        };
```

```
int balance[netSize] = {0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0}; //матриця яка відображає, чи нейрон є балансуючим

int neuroFunc[netSize] = {2, 2, 2, 3, 3, 3, 3, 3, 3, 1, 1, 1};//матриця, яка відображає тип функції перетворення нейрона

double neuroParam[netSize] = {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}; //матриця, яка відображає параметр функції перетворення

//- - - - - - - - - - - - - - - - - - - - - - - - - - - -
double vin[netSize] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};//матриця, яка відображає вхідні сигнали нейронів

double vout[netSize] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}; //матриця, яка відображає вихідні сигнали нейронів

//- - - - - - - - - - - - - - - - - - - - - - - - - - - -
double input[cntIn] = {0, 0};//вхідні значення мережі

double output[cntOut] = {0, 0, 0}; //вихідні значення мережі

//- - - - - - - - - - - - - - - - - - - - - - - - - - - -

double calc(int num_func,double param,double x) //функція перетворення нейрона
  {
        switch(num_func)
        {
            case 1:{return param*x; break;}//лінійна(через точку(0,0) )
            case 2:{return 1/(1+exp(-param*x));break;}//логістична сигмоїдальна
            case 3:{return (exp(param*x)-exp(-param*x))/(exp(param*x)+exp(-param*x));break;}//Гіперболічний тангенс
            case 4:{
                  if(x>0)return 1;
                    else return -1; //сігнум
                    }
            default: return 0;
        }
  }

void TEST()//функція запуску і проганяння значень через нейронну мережу
{
    for(int i = 0 ; i < netSize; i++)//розрахунок вхідних значень
    {

        if(i<cntIn)//якщо вхідний нейрон
        {
            vin[i] = 0;
            vout[i] = input[i];
        }
        else if(balance[i]==1)//якщо внутрішній балансуючий нейрон
        {
```

```
                        vin[i] = 0;
                        vout[i] = 1;
                }
                else
                {
                        vin[i] = 0;
                        for(int j = 0;j < netSize;j++)//посумувати всі вхідні
                                if(type[i][j]=='i')
                                {
                                        vin[i]+=vout[j]*value[j][i];
                                }
                                //printf()
                        vout[i] = calc(neuroFunc[i], neuroParam[i], vin[i]);
                }
        }

        int first_out = netSize - cntOut;// номер першого вихідного нейрона

        for(int i = first_out; i < netSize; i++) //переприсвоєння вихідних значень мережі
        {
                output[i - first_out] = vout[i];
        }
    }

void netCalc() //функція використання мережі
{

    input[0] = (T-(double)16)/(double)8;//нормалізація отриманого значення температури
    input[1] = (H-(double)15)/(double)60;//нормалізація отриманого значення вологості

    Serial.print("x0(T) = "); //вивід нормалізованого значення температури
    Serial.println(input[0]);

    Serial.print("x1(H) = "); //вивід нормалізованого значення вологості
    Serial.println(input[1]);
        TEST();

        //output
        Serial.println("Result");
        for(int i = 0;i < cntOut; i++) //вивід результатів
        {
                Serial.print(i);
                Serial.print("  out = ");
                Serial.println(output[i]);
        }
    }
//
//   FILE:  dht11_test1.pde
// PURPOSE: DHT11 library test sketch for Arduino
//

//Celsius to Fahrenheit conversion
```

```
double Fahrenheit(double celsius) //перетворення градусів Цельсія у градуси Фаренгейта
{
        return 1.8 * celsius + 32;
}
//Celsius to Kelvin conversion
double Kelvin(double celsius) //перетворення градусів Цельсія у градуси Кельвіна
{
        return celsius + 273.15;
}
// dewPoint function NOAA
// reference: http://wahiduddin.net/calc/density_algorithms.htm
double dewPoint(double celsius, double humidity) // обчислення точки роси
{
        double A0= 373.15/(273.15 + celsius);
        double SUM = -7.90298 * (A0-1);
        SUM += 5.02808 * log10(A0);
        SUM += -1.3816e-7 * (pow(10, (11.344*(1-1/A0)))-1) ;
        SUM += 8.1328e-3 * (pow(10,(-3.49149*(A0-1)))-1) ;
        SUM += log10(1013.246);
        double VP = pow(10, SUM-3) * humidity;
        double T = log(VP/0.61078);   // temp var
        return (241.88 * T) / (17.558-T);
}
// delta max = 0.6544 wrt dewPoint()
// 5x faster than dewPoint()
// reference: http://en.wikipedia.org/wiki/Dew_point
double dewPointFast(double celsius, double humidity) //спрощене обчислення точки роси
{
        double a = 17.271;
        double b = 237.7;
        double temp = (a * celsius) / (b + celsius) + log(humidity/100);
        double Td = (b * temp) / (a - temp);
        return Td;
}
void setup() //ініціалізація портів
{
  pinMode(7, OUTPUT); //ініціалізація 7-го вихідного порту
  pinMode(6, OUTPUT); //ініціалізація 6-го вихідного порту
  pinMode(5, OUTPUT); //ініціалізація 5-го вихідного порту

  Serial.begin(9600);         //  setup serial
  Serial.println("DHT11 TEST PROGRAM ");
  Serial.print("LIBRARY VERSION: ");
  Serial.println(DHT11LIB_VERSION);
  Serial.println();
}
void read_and_write_DT11() //зчитування і виведення отриманих даних від давача
{
    Serial.println("\n");

  int chk = DHT11.read(DHT11PIN); //стан давача
  Serial.print("Read sensor: ");
```

```
  switch (chk)
  {
    case 0: Serial.println("OK"); break; // з давачем все добре
    case -1: Serial.println("Checksum error"); break; //помилка контрольної суми
    case -2: Serial.println("Time out error"); break; //перевищено ліміт часу
    default: Serial.println("Unknown error"); break; //невідома помилка
  }
  Serial.print("Humidity (%): ");
  Serial.println((float)DHT11.humidity, 2); //виведення вологості

  H = (float)DHT11.humidity; //присвоєння значення вологості

  Serial.print("Temperature (oC): ");
  Serial.println((float)DHT11.temperature, 2); //виведення температури у градусах цельсія

  T = (float)DHT11.temperature; //присвоєння значення температури
  Serial.print("Temperature (oF): "); //виведення температури у градусах Фаренгейта
  Serial.println(Fahrenheit(DHT11.temperature), 2);
  Serial.print("Temperature (K): "); //виведення температури у градусах Кельвіна
  Serial.println(Kelvin(DHT11.temperature), 2);

  Serial.print("Dew Point (oC): "); //виведення температури точки роси
  Serial.println(dewPoint(DHT11.temperature, DHT11.humidity));

  Serial.print("Dew PointFast (oC): "); //виведення температури точки роси
  Serial.println(dewPointFast(DHT11.temperature, DHT11.humidity));

}

void loop()
{
  read_and_write_DT11();//зчитування даних давачем
  //обігрівач
  if(output[0]>0.5)digitalWrite(7,HIGH); //чи необхідно включити обігрівач
  else digitalWrite(7,LOW);
  //витяжка
  if(output[1]>0.5)digitalWrite(6,HIGH); //чи необхідно включити витяжку
  else digitalWrite(6,LOW);
  //зволожувач
  if(output[2]>0.5)digitalWrite(5,HIGH); //чи необхідно включити зволожувач
  else digitalWrite(5,LOW);
  netCalc();
  delay(2000);
}
```

Додаток Б

Приклад коду програми, яка реалізує нейрофункції для підсистеми освітлення

```
int val = 0;
//початок блоку вибору портів нейроконтролера та введення даних нейроконтролера
#define MotionSensor1 2 //визначення порта, з якого зчитується сигнал від давача
#define FotoResistor1 A0 //визначення порта, з якого зчитується сигнал від давача
#define FotoResistor2 A1 //визначення порта, з якого зчитується сигнал від давача
#define FR1 0 //визначення порта, з якого зчитується сигнал від давача
#define FR2 1 //визначення порта, з якого зчитується сигнал від давача
#define lightConst 980 // константа граничного сигналу від фото резисторів, при якому освітленість прирівнюється до 0
int countMotion1Active = 10;     // кількість циклів активності сигналу від давачу руху
int motion1Status = 0;    //значення давача руху
int FR1Status = 0;  //значення фото резистора ззовні
int FR2Status = 0;  //значення фото резистора всередині
//кінець блоку вибору портів нейроконтролера та введення даних нейроконтролера

//початок блоку ініціалізації даних мережі
const int netSize = 14; //кількість нейронів в мережі
const int cntIn = 3; //кількість вхідних нейронів
const int cntOut = 1; //кількість вихідних нейронів
      //- - - - - NET parameters - - - - - - - - - - - - - - - - - - - - - - - - - - -
      char type[netSize][netSize] = //матриця зв’язків між нейронами (n – немає зв’язку, o – вихідне ребро, i – вхідне ребро)
      {
          {'-', '-', '-', '-', '-', 'i', 'i', 'i', 'i', '-', '-', '-', '-', '-'},
          {'-', '-', '-', '-', '-', 'i', 'i', '-', '-', '-', '-', '-', '-', '-'},
          {'-', '-', '-', '-', '-', 'i', 'i', 'i', 'i', '-', '-', '-', '-', '-'},
          {'-', '-', '-', '-', '-', '-', '-', 'i', 'i', '-', '-', '-', '-', '-'},
          {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'i', 'i', '-', '-'},
          {'o', 'o', 'o', '-', '-', '-', '-', '-', '-', '-', 'i', '-', '-', '-'},
          {'o', 'o', 'o', '-', '-', '-', '-', '-', '-', '-', 'i', '-', '-', '-'},
          {'o', '-', 'o', 'o', '-', '-', '-', '-', '-', '-', '-', 'i', '-', '-'},
          {'o', '-', 'o', 'o', '-', '-', '-', '-', '-', '-', '-', 'i', '-', '-'},
          {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'i'},
          {'-', '-', '-', '-', 'o', 'o', 'o', '-', '-', '-', '-', '-', '-', 'i'},
          {'-', '-', '-', '-', 'o', '-', '-', 'o', 'o', '-', '-', '-', '-', 'i'},
          {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-'},
          {'-', '-', '-', '-', '-', '-', '-', '-', '-', 'o', 'o', 'o', '-', '-'}
      };

      double value[netSize][netSize] = //матриця суміжності(відображено ваги зв’язків)
      {
          {0.0, 0.0, 0.0, 0.0, 0.0, -3.9257025717240333, -3.9257025717240333, -1.8202437839197347, -1.8202437839197347, 0.0, 0.0, 0.0, 0.0, 0.0},
          {0.0, 0.0, 0.0, 0.0, 0.0, 2.737875581433616, 2.737875581433616, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0},
          {0.0, 0.0, 0.0, 0.0, 0.0, 2.3787428296448283, 2.3787428296448283, 3.1384669773803213, 3.1384669773803213, 0.0, 0.0, 0.0, 0.0, 0.0},
          {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 3.960413057592896, 3.960413057592896, 0.0, 0.0, 0.0, 0.0, 0.0},
          {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -4.40926068958452, -4.397735769494833, 0.0, 0.0},
          {-3.9257025717240333, 2.737875581433616, 2.3787428296448283, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 4.588144692929326, 0.0, 0.0, 0.0}, //Ваги матриці суміжності
          {-3.9257025717240333, 2.737875581433616, 2.3787428296448283, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 4.588144692929326, 0.0, 0.0, 0.0}, //Ваги матриці суміжності
          {-1.8202437839197347, 0.0, 3.1384669773803213, 3.960413057592896, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 4.593210578190639, 0.0, 0.0}, //Ваги матриці суміжності
```

```
          {-1.8202437839197347, 0.0, 3.1384669773803213, 3.960413057592896, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 4.593210578190639, 0.0, 0.0}, //Ваги матриці суміжності
          {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -
3.9950388996800092}, //Ваги матриці суміжності
          {0.0, 0.0, 0.0, 0.0, -4.40926068958452, 4.588144692929326, 4.588144692929326,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -9.705586149906091}, //Ваги матриці суміжності
          {0.0, 0.0, 0.0, 0.0, -4.397735769494833, 0.0, 0.0, 4.593210578190639,
4.593210578190639, 0.0, 0.0, 0.0, 0.0, 9.077956110400317}, //Ваги матриці суміжності
          {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0}, //Ваги
матриці суміжності
          {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -3.9950388996800092, -
9.705586149906091, 9.077956110400317, 0.0, 0.0} //Ваги матриці суміжності
};
       int balance[netSize] = {1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0}; //матриця яка
відображає, чи нейрон є балансуючим
       int neuroFunc[netSize] = {1, 2, 2, 2, 1, 2, 2, 2, 2, 1, 2, 2, 1, 2};//матриця, яка
відображає тип функції перетворення нейрона
       double neuroParam[netSize] = {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}; //матриця,
яка відображає параметр функції перетворення

       //- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
       double vin[netSize] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};//матриця, яка
відображає вхідні сигнали нейронів

       double vout[netSize] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}; //матриця, яка
відображає вихідні сигнали нейронів

       //- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
       double input[cntIn] = {0, 0, 0};//вхідні значення мережі

       double output[cntOut] = {0}; //вихідні значення мережі

       //- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
//кінець блоку ініціалізації даних мережі

//початок блоку роботи мережі
       double calc(int num_func,double param,double x) //функція перетворення нейрона
          {
                 switch(num_func)
                 {
                     case 1:{return param*x; break;}//лінійна(через точку(0,0) )
                     case 2:{return 1/(1+exp(-param*x));break;}//логістична сигмоїдальна
                     case 3:{return (exp(param*x)-exp(-param*x))/(exp(param*x)+exp(-
param*x));break;}//Гіперболічний тангенс
                     case 4:{
                            if(x>0)return 1;
                               else return -1; //сігнум
                               }
                     default: return 0;
                 }
          }

       void TEST()//функція запуску і проганяння значень через нейронну мережу
       {
              for(int i = 0 ; i < netSize; i++)//розрахунок вхідних значень
              {

                     if(i<cntIn+1 && i>0)//якщо вхідний нейрон
                     {
                            vin[i] = 0;
                            vout[i] = input[i-1];
```

```
                }
                else if(balance[i]==1)//якщо внутрішній балансуючий нейрон
                {
                        vin[i] = 0;
                        vout[i] = 1;
                }
                else
                {
                        vin[i] = 0;
                        for(int j = 0;j < netSize;j++)//посумувати всі вхідні
                                if(type[i][j]=='o')
                                {
                                        vin[i]+=vout[j]*value[j][i];
                                }
                                //printf()
                        vout[i] = calc(neuroFunc[i], neuroParam[i], vin[i]);
                }
        }

        int first_out = netSize - cntOut;// номер першого вихідного нейрона

        for(int i = first_out; i < netSize; i++) //переприсвоєння вихідних значень
мережі
        {
                output[i - first_out] = vout[i];
        }
    }

void netCalc() //функція використання мережі
{
        TEST();
        //output
        Serial.println("Result");
        for(int i = 0;i < cntOut; i++) //вивід результатів
        {
                Serial.print(i);
                Serial.print("  out = ");
                Serial.println(output[i]);
        }
    }
//кінець блоку роботи мережі
//початок блоку ініціалізації портів
void setup() //ініціалізація портів
{
  pinMode(MotionSensor1,INPUT);  //ініціалізація порту для давача руху
  pinMode(FotoResistor1,INPUT);  //ініціалізація порту для фото резистора ззовні
  pinMode(FotoResistor2,INPUT);  //ініціалізація порту для фото резистора всередині
  pinMode(7,OUTPUT);

  Serial.begin(9600);          //  setup serial
  Serial.println("NeuroController lighting subsystem");
}
//кінець блоку ініціалізації портів
//початок блоку основного циклу програми
void loop()
{
  countMotion1Active--; //зменшити кількість циклів повторення сигналу від давача руху
  val = digitalRead(MotionSensor1);//зчитати значення від давача руху
  if(val==1) //якщо зявився рух
  {
    motion1Status = 1; //встановити статус давача руху як 1
```

```
    countMotion1Active = 10;     //кількість циклів повтору = 10
  }else if(countMotion1Active<0)
  {
    countMotion1Active = 10;     //кількість циклів повтору = 10
    motion1Status = 0;    //встановити статус давача руху як 0
  }
  Serial.print("MotionSensor1 Digital pin = ");      //вивести значення від давача руху
  Serial.println(motion1Status);

  val = analogRead(FR1);    // зчитати значення від зовнішнього фоторезистора
  if(val<lightConst)FR1Status = 1;// якщо значення менше за порогове, то встановити статус
фото резистора як 1
  else FR1Status = 0;      //навпаки  - встановити статус 0
  Serial.print("FotoResistor1 Analog pin = ");      //вивести значення від зовнішнього
фоторезистора
  Serial.println(FR1Status);

  val = analogRead(FR2);    // зчитати значення від внутрішнього фоторезистора
  if(val<lightConst)FR2Status = 1; // якщо значення менше за порогове, то встановити статус
фото резистора як 1
  else FR2Status = 0;      //навпаки  - встановити статус 0
  Serial.print("FotoResistor2 Analog pin = ");      //вивести значення від внутрішнього
фоторезистора
  Serial.println(FR2Status);
  Serial.println();

  input[0] = FR1Status;   //перенести значення від давачі у масив вхідних значень мережі
  input[1] = FR2Status;
  input[2] = motion1Status;
  netCalc();
  if(output[0]<0.5) //якщо вихідний результат менший за 0.5
  {
    digitalWrite(7,LOW); //погасити фотодіод
    Serial.print("Turn off LED");
  }
  if(output[0]>0.5) //якщо вихідний результат більший за 0.5
  {
    digitalWrite(7,HIGH);//запалити фотодіод
    Serial.print("Turn on LED");
  }
  delay(1000);  //встановити паузу 1000 мс.
}
//кінець блоку основного циклу програми
```

Додаток В

Приклад коду програми, яка реалізує нейрофункції для підсистеми захисту

```
#define CORRECT_PIN 13
#define ERROR_PIN 12
#define ERROR1_PIN 11
#define ERROR2_PIN 10
#define ERROR3_PIN 9
#define ERROR4_PIN 8
#define MOVEMENT1_PIN 5
#define MOVEMENT2_PIN 4
#define MOVEMENT3_PIN 3
#define MOVEMENT4_PIN 2
#define KEY1_PIN A4
#define KEY2_PIN A3
#define KEY3_PIN A2
#define KEY4_PIN A1
int key1 = 0;
int key2 = 0;
int key3 = 0;
int key4 = 0;
int movement1 = 0;
int movement2 = 0;
int movement3 = 0;
int movement4 = 0;
int error = 0;
int error1 = 0;
int error2 = 0;
int error3 = 0;
int error4 = 0;
//- - - - - - - - - - - - - - - - - - - - - - - - - -
//- - - - - - - - - - - - - - - - - - - - - - - - - -
const int netSize = 16; //кількість нейронів в мережі
const int cntIn = 8; //кількість вхідних нейронів
const int cntOut = 1; //кількість вихідних нейронів
    //- - - - - NET parameters - - - - - - - - - - - - - - - - - - - - - - - -
    char type[netSize][netSize] = //матриця зв’язків між нейронами (n – немає зв’язку, o – вихідне ребро, i – вхідне ребро)
    {
   {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'i', 'i', 'i', 'i', '-', '-'},
   {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'i', 'i', 'i', 'i', '-', '-'},
   {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'i', 'i', 'i', 'i', '-', '-'},
   {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'i', 'i', 'i', 'i', '-', '-'},
   {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'i', 'i', 'i', 'i', '-', '-'},
   {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'i', 'i', 'i', 'i', '-', '-'},
   {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'i', 'i', 'i', 'i', '-', '-'},
   {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'i', 'i', 'i', 'i', '-', '-'},
   {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'i', 'i', 'i', 'i', '-', '-'},
   {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'i'},
   {'o', 'o', 'o', 'o', 'o', 'o', 'o', 'o', 'o', '-', '-', '-', '-', '-', '-', 'i'},
   {'o', 'o', 'o', 'o', 'o', 'o', 'o', 'o', 'o', '-', '-', '-', '-', '-', '-', 'i'},
   {'o', 'o', 'o', 'o', 'o', 'o', 'o', 'o', 'o', '-', '-', '-', '-', '-', '-', 'i'},
```

```
        {'o', 'o', 'o', 'o', 'o', 'o', 'o', 'o', 'o', '-', '-', '-', '-', '-', '-', 'i'},
        {'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'-', '-', '-', '-', '-', '-', '-', '-', '-', 'o', 'o', 'o', 'o', 'o', '-', '-'}

         };
         double value[netSize][netSize] = //матриця суміжності(відображено ваги зв'язків)
         {
        {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -4.035067580782305, -4.036862147839079, -
4.036469690109577, -4.033713676758726, 0.0, 0.0},
        {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 9.069084775930811, -0.5994327826727128, -
1.5717433324551555, -0.3943285768629514, 0.0, 0.0},
        {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 9.074742931984566, -0.388865445098405, -
1.6001052883156244, -0.59684507918157, 0.0, 0.0},
        {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -0.3902961218577336, 9.071618652515337, -
0.5981055107750697, -1.5900896677138716, 0.0, 0.0},
        {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -0.6024832471675343, 9.066137909260112, -
0.38824156435006507, -1.55599631310285, 0.0, 0.0},
        {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -1.5770948099861921, -0.384786700235302,
9.072430405721757, -0.5983297060454281, 0.0, 0.0},
        {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -1.5677773250215863, -0.6015615026930282,
9.068604092919239, -0.38973392488782727, 0.0, 0.0},
        {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -0.6066052558546915, -1.5817894886123327, -
0.3867434240060377, 9.072007352436831, 0.0, 0.0},
        {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -0.3944622441137218, -1.5750061644058797, -
0.5980975761579511, 9.069752880813587, 0.0, 0.0},
        {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -3.1254359066953232},
        {-4.035067580782305, 9.069084775930811, 9.074742931984566, -0.3902961218577336, -
0.6024832471675343, -1.5770948099861921, -1.5677773250215863, -0.6066052558546915, -
0.3944622441137218, 0.0, 0.0, 0.0, 0.0, 0.0,  0.0, 7.008650226091391},
        {-4.036862147839079, -0.5994327826727128, -0.388865445098405, 9.071618652515337,
9.066137909260112, -0.384786700235302, -0.6015615026930282, -1.5817894886123327, -
1.5750061644058797, 0.0, 0.0, 0.0, 0.0, 0.0,  0.0, 7.015445856509271 },
        {-4.036469690109577, -1.5717433324551555, -1.6001052883156244, -0.5981055107750697, -
0.38824156435006507, 9.072430405721757, 9.068604092919239, -0.3867434240060377, -
0.5980975761579511, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 7.007371957224729},
        {-4.033713676758726, -0.3943285768629514, -0.59684507918157, -1.5900896677138716, -
1.55599631310285, -0.5983297060454281, -0.38973392488782727, 9.072007352436831,
9.069752880813587, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 7.014285481058233},
        {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0},
        {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -3.1254359066953232, 7.008650226091391,
7.015445856509271, 7.007371957224729, 7.014285481058233, 0.0, 0.0}
};
         int balance[netSize] = {1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0}; //матриця яка відображає, чи
нейрон є балансуючим

         int neuroFunc[netSize] = {1, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 1, 2};//матриця, яка відображає
тип функції перетворення нейрона

         double neuroParam[netSize] = {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}; //матриця, яка
відображає параметр функції перетворення
         //- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

```
        double vin[netSize] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};//матриця, яка відображає
вхідні сигнали нейронів
        double vout[netSize] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}; //матриця, яка відображає
вихідні сигнали нейронів
        //- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
        double input[cntIn] = {0, 0, 0, 0, 0, 0, 0, 0};//вхідні значення мережі
        double output[cntOut] = {0}; //вихідні значення мережі
        //- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
        double calc(int num_func,double param,double x) //функція перетворення нейрона
          {
                  switch(num_func)
                  {
                        case 1:{return param*x; break;}//лінійна(через точку(0,0) )
                        case 2:{return 1/(1+exp(-param*x));break;}//логістична сигмоїдальна
                        case 3:{return (exp(param*x)-exp(-param*x))/(exp(param*x)+exp(-
param*x));break;}//Гіперболічний тангенс
                        case 4:{
                                if(x>0)return 1;
                                  else return -1; //сігнум
                                  }
                        default: return 0;
                  }
          }

        void TEST()//функція запуску і проганяння значень через нейронну мережу
        {
                for(int i = 0 ; i < netSize; i++)//розрахунок вхідних значень
                {
                        if(i<cntIn+1 && i>0)//якщо вхідний нейрон
                        {
                                vin[i] = 0;
                                vout[i] = input[i-1];
                        }
                        else if(balance[i]==1)//якщо внутрішній балансуючий нейрон
                        {
                                vin[i] = 0;
                                vout[i] = 1;
                        }
                        else
                        {
                                vin[i] = 0;
                                for(int j = 0;j < netSize;j++)//посумувати всі вхідні
                                        if(type[i][j]=='o')
                                        {
                                                vin[i]+=vout[j]*value[j][i];
                                        }
                                        //printf()
                                vout[i] = calc(neuroFunc[i], neuroParam[i], vin[i]);
                        }
                }
           for(int i = 0; i < netSize; i++)
           {
```

```
                Serial.print(i);
                        Serial.print("  vout[] = ");
                        Serial.print(vout[i]);
                Serial.print("  vin[] = ");
                        Serial.println(vin[i]);
            }
                int first_out = netSize - cntOut;// номер першого вихідного нейрона

                for(int i = first_out; i < netSize; i++) //переприсвоєння вихідних значень мережі
                {
                        output[i - first_out] = vout[i];
                }
        }
void netCalc() //функція використання мережі
{
  //input
  if(key1<900)input[0] = 1;
  else input[0] = 0;
  input[1] = movement1;
  if(key2<900)input[2] = 1;
  else input[2] = 0;
  input[3] = movement2;
  if(key3<900)input[4] = 1;
  else input[4] = 0;
  input[5] = movement3;
  if(key4<900)input[6] = 1;
  else input[6] = 0;
  input[7] = movement4;
  TEST();
  //output
  /*Serial.println("Result");
  for(int i = 0;i < cntOut; i++) //вивід результатів
    {
     Serial.print(i);
     Serial.print("  out = ");
     Serial.println(output[i]);
    }
   */
  if(vout[15]>0.5)error = 1;
  else error = 0;

  if(vout[10]>0.5)error1 = 1;
  else error1 = 0;
  if(vout[11]>0.5)error2 = 1;
  else error2 = 0;
  if(vout[12]>0.5)error3 = 1;
  else error3 = 0;
  if(vout[13]>0.5)error4 = 1;
  else error4 = 0;
}
void setup(){
  pinMode(CORRECT_PIN, OUTPUT);
```

```
  pinMode(ERROR_PIN, OUTPUT);

  pinMode(ERROR1_PIN, OUTPUT);
  pinMode(ERROR2_PIN, OUTPUT);
  pinMode(ERROR3_PIN, OUTPUT);
  pinMode(ERROR4_PIN, OUTPUT);

  pinMode(MOVEMENT1_PIN, INPUT);
  pinMode(MOVEMENT2_PIN, INPUT);
  pinMode(MOVEMENT3_PIN, INPUT);
  pinMode(MOVEMENT4_PIN, INPUT);

  Serial.begin(9600);
}
void readData(){
  key1 = analogRead(KEY1_PIN);
  key2 = analogRead(KEY2_PIN);
  key3 = analogRead(KEY3_PIN);
  key4 = analogRead(KEY4_PIN);
  movement1 = digitalRead(MOVEMENT1_PIN);
  movement2 = digitalRead(MOVEMENT2_PIN);
  movement3 = digitalRead(MOVEMENT3_PIN);
  movement4 = digitalRead(MOVEMENT4_PIN);
}
void writeData(){
  Serial.println("TEST");
  Serial.print("key1 = ");
  Serial.print(key1);
  Serial.print("\tkey2 = ");
  Serial.print(key2);
  Serial.print("\tkey3 = ");
  Serial.print(key3);
  Serial.print("\tkey4 = ");
  Serial.println(key4);
  Serial.print("mov1 = ");
  Serial.print(movement1);
  Serial.print("\tmov2 = ");
  Serial.print(movement2);
  Serial.print("\tmov3 = ");
  Serial.print(movement3);
  Serial.print("\tmov4 = ");
  Serial.println(movement4);
}

void act(){
  /*if(movement1 || key1<900) error1 = 1;
    else error1 = 0;
  if(movement2 || key2<900) error2 = 1;
    else error2 = 0;
  if(movement3 || key3<900) error3 = 1;
    else error3 = 0;
  if(movement4 || key4<900) error4 = 1;
```

```
    else error4 = 0;
  if(error1 || error2 || error3 || error4) error = 1;
  else error = 0;
  if(error) digitalWrite(ERROR_PIN, HIGH);
  else digitalWrite(ERROR_PIN, LOW);
  if(!error) digitalWrite(CORRECT_PIN, HIGH);
  else digitalWrite(CORRECT_PIN, LOW);

  if(error1) digitalWrite(ERROR1_PIN, HIGH);
  else digitalWrite(ERROR1_PIN, LOW);

  if(error2) digitalWrite(ERROR2_PIN, HIGH);
  else digitalWrite(ERROR2_PIN, LOW);

  if(error3) digitalWrite(ERROR3_PIN, HIGH);
  else digitalWrite(ERROR3_PIN, LOW);

  if(error4) digitalWrite(ERROR4_PIN, HIGH);
  else digitalWrite(ERROR4_PIN, LOW);
}

void loop(){
  readData();
  writeData();
  netCalc();
  act();
  delay(1000);
}
```

Додаток Г

Приклад коду програми, яка реалізує нейрофункції для підсистеми запобігання технічних аварій

```
#define FireSensor 12 //визначення порту для давача вогню
#define SmokeSensor A0 //визначення порту для давача диму/газу
#define WaterSensor A1 //визначення порту для давача протікання води

#define Led1 11 //визначення порту для 1-го активатора
#define Led2 10 //визначення порту для 2-го активатора
#define Led3 9  //визначення порту для 3-го активатора
#define Led4 8  //визначення порту для 4-го активатора
#define Led5 7  //визначення порту для 1-ї сигналізації
#define Led6 6  //визначення порту для 2-ї сигналізації
#define Led7 5  //визначення порту для 3-ї сигналізації

#define Sum_Good 4  //визначення порту для статусу системи
#define Sum_Bad 3

int FireStatus;  //статус загрози вогню
int SmokeStatus;  //статус загрози задимлення,протікання газу
int WaterStatus;

int val = 0;
const int netSize = 16; //кількість нейронів в мережі
const int cntIn = 3; //кількість вхідних нейронів
const int cntOut = 7; //кількість вихідних нейронів

//- - - - - NET parameters - - - - - - - - - - - - - - - - - - - - - - -
  const float value[netSize][netSize] = //матриця суміжності(відображено ваги зв'язків)
      {
{0.0, 0.0, 0.0, 0.0, 0.0, -4.689294795590509, -3.3988706207830064, -4.269634668533792, 0.0, 0.0, 0.0,
   0.0, 0.0, 0.0, 0.0, 0.0},
{0.0, 0.0, 0.0, 0.0, 0.0, 9.231720568364084, -7.477936003780324, -1.2911886237754364, 0.0, 0.0, 0.0,
   0.0, 0.0, 0.0, 0.0, 0.0},
{0.0, 0.0, 0.0, 0.0, 0.0, 0.09770150867223379, 8.451531384220349, -1.6114376928300227, 0.0, 0.0, 0.0,
   0.0, 0.0, 0.0, 0.0, 0.0},
{0.0, 0.0, 0.0, 0.0, 0.0, -1.064528287006327, 4.137240168750305, 8.707029889788123, 0.0, 0.0, 0.0, 0.0,
   0.0, 0.0, 0.0, 0.0},
{0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -5.72917062851063, -4.844487656027435, -6.188626353691188,
   -5.399160700601852, -5.424855523135816, -5.728722858654511, -4.844487656027435},
{-4.689294795590509, 9.231720568364084, 0.09770150867223379, -1.064528287006327, 0.0, 0.0, 0.0,
   0.0, 0.0, 0.09032241677360754, 9.985909128101364, 0.3460236071213439, -1.3899810598938396,
   11.156738059947896, 0.08967889483518768, 9.985909128101364},
{-3.3988706207830064, -7.477936003780324, 8.451531384220349, 4.137240168750305, 0.0, 0.0, 0.0,
   0.0, 0.0, 11.18708086781511, -4.133210945190818, 1.0518659864627287, 10.512579856232833,
   10.47845405141086, 11.186744005557276, -4.133210945190818},
{-4.269634668533792, -1.2911886237754364, -1.6114376928300227, 8.707029889788123, 0.0, 0.0, 0.0,
   0.0, 0.0, 4.000198312282817, 1.300370278774557, 10.665336365726679, -7.176956501982124,
   8.83248679292553, 4.000999487407349, 1.300370278774557},
{0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0},
```

```
{0.0, 0.0, 0.0, 0.0, -5.72917062851063, 0.09032241677360754, 11.18768086781511,
  4.000198312282817, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0},
{0.0, 0.0, 0.0, 0.0, -4.844487656027435, 9.985909128101364, -4.133210945190818,
  1.300370278774557, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0},
{0.0, 0.0, 0.0, 0.0, -6.188626353691188, 0.3460236071213439, 1.0518659864627287,
  10.665336365726679, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0},
{0.0, 0.0, 0.0, 0.0, -5.399160700601852, -1.3899810598938396, 10.512579856232833, -
  7.176956501982124, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0},
{0.0, 0.0, 0.0, 0.0, -5.424855523135816, 11.156738059947896, 10.47845405141086, 8.83248679292553,
  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0},
{0.0, 0.0, 0.0, 0.0, -5.728722858654511, 0.08967889483518768, 11.186744005557276,
  4.000999487407349, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0},
{0.0, 0.0, 0.0, 0.0, -4.844487656027435, 9.985909128101364, -4.133210945190818,
  1.300370278774557, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0}

    };

  int balance[netSize] = {1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0}; //матриця яка відображає, чи нейрон
  є балансуючим

  //- - - - - - - - - - - - - - - - - - - - - - - - - - - -
  double vin[netSize] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};//матриця, яка відображає вхідні
  сигнали нейронів

  double vout[netSize] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}; //матриця, яка відображає вихідні
  сигнали нейронів

  //- - - - - - - - - - - - - - - - - - - - - - - - - - - -
  double input[cntIn] = {0, 0, 0};//вхідні значення мережі

  double output[cntOut] = {0, 0, 0, 0, 0, 0, 0}; //вихідні значення мережі

  //- - - - - - - - - - - - - - - - - - - - - - - - - - - -

  double calc(int num_func,double param,double x) //функція перетворення нейрона
    {
          switch(num_func)
          {
                //case 1:{return param*x; break;}//лінійна(через точку(0,0) )
                case 2:{return 1/(1+exp(-param*x));break;}//логістична сигмоїдальна
                /*case 3:{return (exp(param*x)-exp(-param*x))/(exp(param*x)+exp(-
  param*x));break;}//Гіперболічний тангенс
                case 4:{
                      if(x>0)return 1;
                        else return -1; //сігнум
                        }
                default: return 0;*/
          }
    }


  void TEST()//функція запуску і проганяння значень через нейронну мережу
```

```
  {
        for(int i = 0 ; i < netSize; i++)//розрахунок вхідних значень
        {
            if(i<cntIn+1 && i>0)//якщо вхідний нейрон
            {
                vin[i] = 0;
                vout[i] = input[i-1];
            }
            else if(balance[i]==1)//якщо внутрішній балансуючий нейрон
            {
                vin[i] = 0;
                vout[i] = 1;
            }
            else
            {
                vin[i] = 0;
                for(int j = 0; j < i; j++)//посумувати всі вхідні
                    /*if(type[i][j]=='o')
                    {
                        vin[i]+=vout[j]*value[j][i];
                    }*/
                if(value[i][j]!=0.0)
                    {
                        vin[i]+=vout[j]*value[j][i];
                    }

                vout[i] = calc(2, 1, vin[i]);
            }
        }


        for(int i = 0; i < netSize; i++)//вивести вхідні та вихідні значення усіх нейронів
        {
          Serial.print(i, DEC);
            Serial.print("  vout[] = ");
            Serial.print(vout[i],3);
          Serial.print("  vin[] = ");
            Serial.println(vin[i],3);
          Serial.flush();
        }

        int first_out = netSize - cntOut;// номер першого вихідного нейрона

        for(int i = first_out; i < netSize; i++) //переприсвоєння вихідних значень мережі
        {
            output[i - first_out] = vout[i];
        }
  }

void netCalc() //функція використання мережі
{
```

```
            TEST();//запустити нейронну мережу
            //output
            Serial.println("Result");
            for(int i = 0;i < cntOut; i++) //вивід результатів
            {
                    Serial.print(i);
                    Serial.print(" out=");
                    Serial.println(output[i],3);
                // Serial.flush();
            }
    }


void setup() //ініціалізація портів
{
  pinMode(Led1,OUTPUT);
  pinMode(Led2,OUTPUT);
  pinMode(Led3,OUTPUT);
  pinMode(Led4,OUTPUT);
  pinMode(Led5,OUTPUT);
  pinMode(Led6,OUTPUT);
  pinMode(Led7,OUTPUT);
  pinMode(FireSensor,OUTPUT);
  pinMode(Sum_Good,OUTPUT);
  pinMode(Sum_Bad,OUTPUT);
  Serial.begin(9600);          //  setup serial
  Serial.println("NeuroController protection subsystem");
}

int freeRam () {//перевірка вільної памяті
  extern int __heap_start, *__brkval;
  int v;
  return (int) &v - (__brkval == 0 ? (int) &__heap_start : (int) __brkval);
}

void loop()
{
  Serial.println(freeRam ());

  val = digitalRead(FireSensor);//зчитати дані від давача вогню
  if(val==1)FireStatus = 1;
  else FireStatus = 0;
  Serial.print("Fire D =");//вивести дані
  Serial.println(val);

  val = analogRead(SmokeSensor);//зчитати дані від давачі диму
  if(val<330)SmokeStatus = 1;
  else SmokeStatus = 0;
  Serial.print("Smoke A =");//вивести дані
  Serial.println(val);
```

```
  val = analogRead(WaterSensor);//зчитати дані від давача протікання води
  if(val<500)WaterStatus = 1;
  else WaterStatus = 0;
  Serial.print("Water A =");//вивести дані
  Serial.println(val);

  input[0] = FireStatus;//внесення вхідних даних у масив
  input[1] = SmokeStatus;
  input[2] = WaterStatus;
  delay(1000);

  netCalc();//запуск обчислень

  if(output[0]<0.5)digitalWrite(Led1,LOW);//вивід статусу першого активатора(якщо включений, то
    запалити)
  else digitalWrite(Led1,HIGH);

  if(output[1]<0.5)digitalWrite(Led2,LOW);//вивід статусу другого активатора(якщо включений, то
    запалити)
  else digitalWrite(Led2,HIGH);

  if(output[2]<0.5)digitalWrite(Led3,LOW);//вивід статусу третього активатора(якщо включений, то
    запалити)
  else digitalWrite(Led3,HIGH);

  if(output[3]<0.5)digitalWrite(Led4,LOW);//вивід статусу четвертого активатора(якщо включений, то
    запалити)
  else digitalWrite(Led4,HIGH);

  if(output[4]<0.5)digitalWrite(Led5,LOW);//вивід статусу першої сигналізації(якщо включений, то
    запалити)
  else digitalWrite(Led5,HIGH);

  if(output[5]<0.5)digitalWrite(Led6,LOW);//вивід статусу другої сигналізації(якщо включений, то
    запалити)
  else digitalWrite(Led6,HIGH);

  if(output[6]<0.5)digitalWrite(Led7,LOW);//вивід статусу третьої сигналізації(якщо включений, то
    запалити)
  else digitalWrite(Led7,HIGH);

  if(output[4]<0.5)digitalWrite(Sum_Good,HIGH);//вивід загального статусу загрози системі(якщо
    немає загрози, то запалити)
  else digitalWrite(Sum_Good,LOW);
  if(output[4]>0.5)digitalWrite(Sum_Bad,HIGH);
  else digitalWrite(Sum_Bad,LOW);

   Serial.println();
  delay(1000);

}
```

Додаток Д

Фрагмент моделі з використанням мереж Петрі в XML-форматі

```
<?xml version="1.0" encoding="iso-8859-1" ?>
- <pnml>
- <net id="Net-One" type="P/T net">
  <token id="Default" enabled="true" red="0" green="0" blue="0" />
- <place id="Actuator_1">
- <graphics>
  <position x="585.0" y="60.0" />
  </graphics>
- <name>
  <value>Actuator_1</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </name>
- <initialMarking>
  <value>Default,0</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </initialMarking>
- <capacity>
  <value>0</value>
  </capacity>
  </place>
- <place id="Actuator_2">
- <graphics>
  <position x="585.0" y="105.0" />
  </graphics>
- <name>
  <value>Actuator_2</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </name>
- <initialMarking>
  <value>Default,0</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </initialMarking>
- <capacity>
  <value>0</value>
  </capacity>
  </place>
- <place id="Actuator_m">
- <graphics>
  <position x="585.0" y="165.0" />
  </graphics>
- <name>
  <value>Actuator_m</value>
```

```
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </name>
- <initialMarking>
  <value>Default,0</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </initialMarking>
- <capacity>
  <value>0</value>
  </capacity>
  </place>
- <place id="AD_Convertor">
- <graphics>
  <position x="225.0" y="120.0" />
  </graphics>
- <name>
  <value>AD_Convertor</value>
- <graphics>
  <offset x="107.0" y="8.0" />
  </graphics>
  </name>
- <initialMarking>
  <value>Default,0</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </initialMarking>
- <capacity>
  <value>0</value>
  </capacity>
  </place>
- <place id="DA_Convertor">
- <graphics>
  <position x="465.0" y="120.0" />
  </graphics>
- <name>
  <value>DA_Convertor</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </name>
- <initialMarking>
  <value>Default,0</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </initialMarking>
- <capacity>
  <value>0</value>
  </capacity>
  </place>
- <place id="Hardware_Processing">
```

```
- <graphics>
  <position x="345.0" y="165.0" />
  </graphics>
- <name>
  <value>Hardware_Processing</value>
- <graphics>
  <offset x="91.0" y="-3.0" />
  </graphics>
  </name>
- <initialMarking>
  <value>Default,0</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </initialMarking>
- <capacity>
  <value>0</value>
  </capacity>
  </place>
- <place id="Program_Processing">
- <graphics>
  <position x="345.0" y="60.0" />
  </graphics>
- <name>
  <value>Program_Processing</value>
- <graphics>
  <offset x="89.0" y="-5.0" />
  </graphics>
  </name>
- <initialMarking>
  <value>Default,0</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </initialMarking>
- <capacity>
  <value>0</value>
  </capacity>
  </place>
- <place id="Sensor_1">
- <graphics>
  <position x="105.0" y="60.0" />
  </graphics>
- <name>
  <value>Sensor_1</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </name>
- <initialMarking>
  <value>Default,0</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </initialMarking>
```

```
- <capacity>
  <value>0</value>
  </capacity>
  </place>
- <place id="Sensor_2">
- <graphics>
  <position x="105.0" y="105.0" />
  </graphics>
- <name>
  <value>Sensor_2</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </name>
- <initialMarking>
  <value>Default,0</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </initialMarking>
- <capacity>
  <value>0</value>
  </capacity>
  </place>
- <place id="Sensor_n">
- <graphics>
  <position x="105.0" y="165.0" />
  </graphics>
- <name>
  <value>Sensor_n</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </name>
- <initialMarking>
  <value>Default,0</value>
- <graphics>
  <offset x="0.0" y="0.0" />
  </graphics>
  </initialMarking>
- <capacity>
  <value>0</value>
  </capacity>
  </place>
- <transition id="T0">
- <graphics>
  <position x="165.0" y="60.0" />
  </graphics>
- <name>
  <value>T0</value>
- <graphics>
  <offset x="-5.0" y="35.0" />
  </graphics>
  </name>
- <orientation>
```

```
<value>0</value>
</orientation>
- <rate>
<value>1.0</value>
</rate>
- <timed>
<value>false</value>
</timed>
- <infiniteServer>
<value>false</value>
</infiniteServer>
- <priority>
<value>1</value>
</priority>
</transition>
- <transition id="T1">
- <graphics>
<position x="165.0" y="105.0" />
</graphics>
- <name>
<value>T1</value>
- <graphics>
<offset x="-5.0" y="35.0" />
</graphics>
</name>
- <orientation>
<value>0</value>
</orientation>
- <rate>
<value>1.0</value>
</rate>
- <timed>
<value>false</value>
</timed>
- <infiniteServer>
<value>false</value>
</infiniteServer>
- <priority>
```

Додаток Е

Результати навчання штучної нейронної мережі підсистеми клімат-контролю

0.0000	0.0000	0.0000	2.0156	-0.8100	3.2359	1.1984	0.5179	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.5459	-1.1247	-0.7054	-0.9870	-2.8024	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	-0.8159	1.2048	-0.4461	-0.0313	0.3071	0.0000	0.0000	0.0000	0.0000
2.0156	0.5459	-0.8159	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-0.7920	2.3595	-0.8381
-0.8100	-1.1247	1.2048	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-0.5456	-0.7423	-0.7051
3.2359	-0.7054	-0.4461	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-0.6181	-1.9484	0.4867
1.1984	-0.9870	-0.0313	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7170	0.1019	-0.4017
0.5179	-2.8024	0.3071	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-0.1512	0.9573	1.0055
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7314	1.1097	0.9167
0.0000	0.0000	0.0000	-0.7920	-0.5456	-0.6181	0.7170	-0.1512	0.7314	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	2.3595	-0.7423	-1.9484	0.1019	0.9573	1.1097	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	-0.8381	-0.7051	0.4867	-0.4017	1.0055	0.9167	0.0000	0.0000	0.0000

balance_neuron (матриця яка відображає, чи нейрон є балансуючим)

0 0 1 0 0 0 0 0 1 0 0 0

Додаток Ж

Параметри ваг матриці суміжності штучної нейронної мережі підсистеми освітлення

0.0 0.0 0.0 0.0 0.0 -3.9257025717240333 -3.9257025717240333 -1.8202437839197347

-1.8202437839197347 0.0 0.0 0.0 0.0 0.0 //

0.0 0.0 0.0 0.0 0.0 2.737875581433616 2.737875581433616 0.0 0.0 0.0 0.0 0.0 0.0 0.0

0.0 0.0 0.0 0.0 0.0 2.3787428296448283 2.3787428296448283 3.1384669773803213 3.1384669773803213 0.0 0.0 0.0 0.0 0.0 //

0.0 0.0 0.0 0.0 0.0 0.0 0.0 3.960413057592896 3.960413057592896 0.0 0.0 0.0 0.0 0.0 //

0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -4.40926068958452 -4.397735769494833 0.0 0.0 //

-3.9257025717240333 2.737875581433616 2.3787428296448283 0.0 0.0 0.0 0.0 0.0 0.0 0.0 4.588144692929326 0.0 0.0 0.0 //

-3.9257025717240333 2.737875581433616 2.3787428296448283 0.0 0.0 0.0 0.0 0.0 0.0 0.0 4.588144692929326 0.0 0.0 0.0 //

-1.8202437839197347 0.0 3.1384669773803213 3.960413057592896 0.0 0.0 0.0 0.0 0.0 0.0 0.0 4.593210578190639 0.0 0.0 //

-1.8202437839197347 0.0 3.1384669773803213 3.960413057592896 0.0 0.0 0.0 0.0 0.0 0.0 0.0 4.593210578190639 0.0 0.0 //

0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -3.9950388996800092 //

0.0 0.0 0.0 0.0 -4.40926068958452 4.588144692929326 4.588144692929326 0.0 0.0 0.0 0.0 0.0 0.0 -9.705586149906091 //

0.0 0.0 0.0 0.0 -4.397735769494833 0.0 0.0 4.593210578190639 4.593210578190639 0.0 0.0 0.0 0.0 9.077956110400317 //

0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 //

0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -3.9950388996800092 -9.705586149906091 9.077956110400317 0.0 0.0 //

Додаток И

Матриця суміжності підсистеми захисту

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -4.035, -4.036, -4.036, -4.033, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 9.069, -0.599, -1.571, -0.394, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 9.074, -0.388, -1.600, -0.596, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -0.390, 9.071, -0.598, -1.590, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -0.602, 9.066, -0.388, -1.555, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -1.577, -0.384, 9.072, -0.598, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -1.567, -0.601, 9.068, -0.389, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -0.606, -1.581, -0.386, 9.072, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -0.394, -1.575, -0.598, 9.069, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -3.125
-4.035, 9.069, 9.074, -0.390, -0.602, -1.577, -1.567, -0.606, -0.394, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 7.008
-4.036, -0.599, -0.388, 9.071, 9.066, -0.384, -0.601, -1.581, -1.575, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 7.0154
-4.036, -1.571, -1.600, -0.598, -0.388, 9.072, 9.068, -0.386, -0.598, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 7.007
-4.033, -0.394, -0.596, -1.590, -1.555, -0.598, -0.389, 9.072, 9.069, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 7.014
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -3.125, 7.008, 7.015, 7.007, 7.014, 0.0, 0.0

balance_neuron (матриця яка відображає, чи нейрон є балансуючим)

1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0

Додаток К

Результати навчання першого та другого шарів нейронної мережі підсистеми запобігання технічних аварій

9999 Average Error = 0.014680192614776572
Perceptron Teacher
PerceptronNet
neuronCount: 8
InputCount = 3
OutputCount = 3
Neuron #0 funcNum = 1 balance = 1 count of parameters = 1
Neuron #1 funcNum = 2 balance = 0 count of parameters = 1
Neuron #2 funcNum = 2 balance = 0 count of parameters = 1
Neuron #3 funcNum = 2 balance = 0 count of parameters = 1
Neuron #4 funcNum = 1 balance = 1 count of parameters = 1
Neuron #5 funcNum = 2 balance = 0 count of parameters = 1
Neuron #6 funcNum = 2 balance = 0 count of parameters = 1
Neuron #7 funcNum = 2 balance = 0 count of parameters = 1

Values
0.0, 0.0, 0.0, 0.0, 0.0, -4.038283746491612, -3.7432252029244193, -3.7823703690233694,
0.0, 0.0, 0.0, 0.0, 0.0, 8.40528233158501, -7.472099451713839, -1.4773121996475447,
0.0, 0.0, 0.0, 0.0, 0.0, -0.16490637647812734, 7.3039296100072955, -1.4958169418554546,
0.0, 0.0, 0.0, 0.0, 0.0, -1.1740759461129155, -1.406642010603528, 7.7133749933092135,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
-4.038283746491612, 8.40528233158501, -0.16490637647812734, -1.1740759461129155, 0.0, 0.0, 0.0, 0.0,
-3.7432252029244193, -7.472099451713839, 7.3039296100072955, -1.406642010603528, 0.0, 0.0, 0.0, 0.0,
-3.7823703690233694, -1.4773121996475447, -1.4958169418554546, 7.7133749933092135, 0.0, 0.0, 0.0, 0.0,

Input Tests
0.0 0.0 0.0
Run test: y[0] = 0.01732234693484016 y[1] = 0.023129952632622863 y[2] = 0.022261786306942272
0.0 0.0 1.0
Run test: y[0] = 0.005419272473058694 y[1] = 0.0057667268364424455 y[2] = 0.9807537396590824
0.0 1.0 0.0
Run test: y[0] = 0.014727668835773962 y[1] = 0.9723665113094178 y[2] = 0.005075775227110669
1.0 0.0 0.0
Run test: y[0] = 0.9874697307289388 y[1] = 1.3466059884140418E-5 y[2] = 0.005170085436086292
1.0 1.0 0.0
Run test: y[0] = 0.9852563910964032 y[1] = 0.01961991830582517 y[2] = 0.0011631036446008716

Output Tests
0.0 0.0 0.0
0.0 0.0 1.0
0.0 1.0 0.0
1.0 0.0 0.0
1.0 0.0 0.0

Додаток Л

Результати навчання другого та третього шарів нейронної мережі підсистеми запобігання технічних аварій

9999 Average Error = 0.017771992384982227
Perceptron Teacher
PerceptronNet
neuronCount: 12
InputCount = 3
OutputCount = 7

Neuron #0 funcNum = 1 balance = 1 count of parameters = 1
Neuron #1 funcNum = 2 balance = 0 count of parameters = 1
Neuron #2 funcNum = 2 balance = 0 count of parameters = 1
Neuron #3 funcNum = 2 balance = 0 count of parameters = 1
Neuron #4 funcNum = 1 balance = 1 count of parameters = 1
Neuron #5 funcNum = 2 balance = 0 count of parameters = 1
Neuron #6 funcNum = 2 balance = 0 count of parameters = 1
Neuron #7 funcNum = 2 balance = 0 count of parameters = 1
Neuron #8 funcNum = 2 balance = 0 count of parameters = 1
Neuron #9 funcNum = 2 balance = 0 count of parameters = 1
Neuron #10 funcNum = 2 balance = 0 count of parameters = 1
Neuron #11 funcNum = 2 balance = 0 count of parameters = 1

Values
0.0, 0.0, 0.0, 0.0, 0.0, -3.463990806489677, -3.8014108484108977, -3.7992552473540484, -3.8068721059945942,
-3.2634755707655563, -3.463990806489677, -3.8014108484108977,
0.0, 0.0, 0.0, 0.0, 0.0, 7.362720525338391, -1.3232111523217613, 7.72182210160895, -1.313470150963293,
7.148477657520414, 7.362720525338391, -1.3232111523217613,
0.0, 0.0, 0.0, 0.0, 0.0, 7.3706482059566625, -1.338605448079142, -1.336192686566655, 7.732954772438837,
7.156193925388169, 7.3706482059566625, -1.338605448079142,
0.0, 0.0, 0.0, 0.0, 0.0, -1.8176413342608508, 7.745349051292951, -1.3498203888922928, -1.3451220760167175,
7.172562940955555, -1.8176413342608508, 7.745349051292951,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
-3.463990806489677, 7.362720525338391, 7.3706482059566625, -1.8176413342608508, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
-3.8014108484108977, -1.3232111523217613, -1.338605448079142, 7.745349051292951, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
-3.7992552473540484, 7.72182210160895, -1.336192686566655, -1.3498203888922928, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
-3.8068721059945942, -1.313470150963293, 7.732954772438837, -1.3451220760167175, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
-3.2634755707655563, 7.148477657520414, 7.156193925388169, 7.172562940955555, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
-3.463990806489677, 7.362720525338391, 7.3706482059566625, -1.8176413342608508, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
-3.8014108484108977, -1.3232111523217613, -1.338605448079142, 7.745349051292951, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

Додаток М

Результати навчання нейронної мережі підсистеми запобігання технічних аварій

Input Tests

0.0 0.0 0.0
Run test: y[0] = 0.03035435158675503 y[1] = 0.02185109563978102 y[2] = 0.021897216167468314
y[3] = 0.02173467306970239 y[4] = 0.03684566927102612 y[5] = 0.03035435158675503 y[6] = 0.02185109563978102
1.0 0.0 0.0
Run test: y[0] = 0.9801349764571012 y[1] = 0.005913290443257231 y[2] = 0.9805938218908447
y[3] = 0.0059385014331870675 y[4] = 0.9798659247833148 y[5] = 0.9801349764571012 y[6] = 0.005913290443257231
0.0 1.0 0.0
Run test: y[0] = 0.9802887454993826 y[1] = 0.0058234826557866425 y[2] = 0.0058499913085573755
y[3] = 0.9806606133895012 y[4] = 0.9800175943221185 y[5] = 0.9802887454993826 y[6] = 0.0058234826557866425
0.0 0.0 1.0
Run test: y[0] = 0.005058408422197062 y[1] = 0.9809963598804637 y[2] = 0.00577126709840495
y[3] = 0.005754544733486269 y[4] = 0.9803356443873844 y[5] = 0.005058408422197062 y[6] = 0.9809963598804637

Output Tests

0.0 0.0 0.0 0.0 0.0 0.0 0.0
1.0 0.0 1.0 0.0 1.0 1.0 0.0
1.0 0.0 0.0 1.0 1.0 1.0 0.0
0.0 1.0 0.0 0.0 1.0 0.0 1.0

99999 Average Error = 0.004657454966800628
Perceptron Teacher PerceptronNet
neuronCount: 16 InputCount = 3 OutputCount = 7
Neuron #0 funcNum = 1 balance = 1 count of parameters = 1 Neuron #1 funcNum = 2 balance = 0 count of parameters = 1
Neuron #2 funcNum = 2 balance = 0 count of parameters = 1 Neuron #3 funcNum = 2 balance = 0 count of parameters = 1
Neuron #4 funcNum = 1 balance = 1 count of parameters = 1 Neuron #5 funcNum = 2 balance = 0 count of parameters = 1
Neuron #6 funcNum = 2 balance = 0 count of parameters = 1 Neuron #7 funcNum = 2 balance = 0 count of parameters = 1
Neuron #8 funcNum = 1 balance = 1 count of parameters = 1 Neuron #9 funcNum = 2 balance = 0 count of parameters = 1
Neuron #10 funcNum = 2 balance = 0 count of parameters = 1 Neuron #11 funcNum = 2 balance = 0 count of parameters = 1
Neuron #12 funcNum = 2 balance = 0 count of parameters = 1 Neuron #13 funcNum = 2 balance = 0 count of parameters = 1
Neuron #14 funcNum = 2 balance = 0 count of parameters = 1 Neuron #15 funcNum = 2 balance = 0 count of parameters = 1

Values

0.0, 0.0, 0.0, 0.0, 0.0, -4.689294795590509, -3.3988706207830064, -4.269634668533792, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 9.231720568364084, -7.477936003780324, -1.2911886237754364, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.09770150867223379, 8.451531384220349, -1.6114376928300227, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, -1.064528287006327, 4.137240168750305, 8.707029889788123, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -5.72917062851063, -4.844487656027435, -6.188626353691188, -5.399160700601852, -
5.424855523135816, -5.728722858654511, -4.844487656027435,
-4.689294795590509, 9.231720568364084, 0.09770150867223379, -1.064528287006327, 0.0, 0.0, 0.0, 0.0, 0.0,
0.09032241677360754, 9.985909128101364, 0.3460236071213439, -1.3899810598938396, 11.156738059947896,
0.08967889483518768, 9.985909128101364,
-3.3988706207830064, -7.477936003780324, 8.451531384220349, 4.137240168750305, 0.0, 0.0, 0.0, 0.0, 0.0,
11.18708086781511, -4.133210945190818, 1.0518659864627287, 10.512579856232833, 10.47845405141086,
11.186744005557276, -4.133210945190818,
-4.269634668533792, -1.2911886237754364, -1.6114376928300227, 8.707029889788123, 0.0, 0.0, 0.0, 0.0, 0.0,
4.000198312282817, 1.300370278774557, 10.665336365726679, -7.176956501982124, 8.83248679292553, 4.000999487407349,
1.300370278774557,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, -5.72917062851063, 0.09032241677360754, 11.18768086781511, 4.000198312282817, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, -4.844487656027435, 9.985909128101364, -4.133210945190818, 1.300370278774557, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, -6.188626353691188, 0.3460236071213439, 1.0518659864627287, 10.665336365726679, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, -5.399160700601852, -1.3899810598938396, 10.512579856232833, -7.176956501982124, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, -5.424855523135816, 11.156738059947896, 10.47845405141086, 8.83248679292553, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, -5.728722858654511, 0.08967889483518768, 11.186744005557276, 4.000999487407349, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, -4.844487656027435, 9.985909128101364, -4.133210945190818, 1.300370278774557, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0,

Додаток Н

Результати тестування нейронної мережі підсистеми запобігання технічних аварій

Type
- - - - - i i i - - - - - - - -
- - - - - i i i - - - - - - - -
- - - - - i i i - - - - - - - -
- - - - - i i i - - - - - - - -
- - - - - - - - - i i i i i i i
o o o o - - - - - i i i i i i i
o o o o - - - - - i i i i i i i
o o o o - - - - - i i i i i i i
- - - - - - - - - - - - - - - -
- - - - o o o o - - - - - - - -
- - - - o o o o - - - - - - - -
- - - - o o o o - - - - - - - -
- - - - o o o o - - - - - - - -
- - - - o o o o - - - - - - - -
- - - - o o o o - - - - - - - -
- - - - o o o o - - - - - - - -

Input Tests
0.0 0.0 0.0
Run test: y[0] = 0.004910471050732457 y[1] = 0.007620871635561738 y[2] = 0.0024619184159460597 y[3] = 0.005647148713126297 y[4] = 0.007670963232868201 y[5] = 0.00491253679825193 y[6] = 0.007620871635561738

1.0 0.0 0.0
Run test: y[0] = 0.0035962382938116407 y[1] = 0.9935750432527941 y[2] = 0.0030022970971108416 y[3] = 0.0011105449911774329 y[4] = 0.9964891367224062 y[5] = 0.0035955721441685144 y[6] = 0.9935750432527941
0.0 1.0 0.0
Run test: y[0] = 0.9955014544327063 y[1] = 1.4370315476727274E-4 y[2] = 0.0059982173435912875 y[3] = 0.9933911655513731 y[4] = 0.9940806394066569 y[5] = 0.9954992713168531 y[6] = 1.4370315476727274E-4
0.0 0.0 1.0
Run test: y[0] = 0.9969652310580394 y[1] = 0.0017887502683054098 y[2] = 0.9937249312125759 y[3] = 0.0045711949745834155 y[4] = 0.9999704615477739 y[5] = 0.9969670568249054 y[6] = 0.0017887502683054098

Output Tests
0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 1.0 0.0 0.0 1.0 0.0 1.0
1.0 0.0 0.0 1.0 1.0 1.0 0.0
1.0 0.0 1.0 0.0 1.0 1.0 0.0

Додаток П

Приклад моделі на основі мережі Петрі в XML-форматі

```
<?xml version="1.0" encoding="UTF-8"?>
<net> //мережа
        <vertex>                                        //перша вершина
                <name>command to find A</name>          //назва першої вершини
                <output>                                //опис вихідного ребра
                        <in>calculate A</in>            //назва переходу, до якого іде
ребро
                        <weight>1</weight>              //вага ребра
                        <color>1</color>                //колір ребра
                        <isnormal>true</isnormal>       //чи ребро нормальне
                </output>
        </vertex>
                <vertex>                                //друга вершина
                <name>A</name>                          //назва другої вершини
                <input>                                 //опис вхідного ребра
                        <out>calculate A</out>          //назва переходу, від якого іде
ребро
                        <weight>1</weight>              //вага ребра
                        <color>1</color>                //колір ребра
                        <isnormal>true</isnormal>       // чи ребро нормальне
                </input>
                <output>
                        <in>read command2</in>
                        <weight>1</weight>
                        <color>1</color>
                        <isnormal>true</isnormal>
                </output>
        </vertex>
                <vertex>                                //третя вершина
                <name>command to find B</name>
                <input>
                        <out>read command2</out>
                        <weight>1</weight>
                        <color>1</color>
                        <isnormal>true</isnormal>
                </input>
                <output>
                        <in>calculate B</in>
                        <weight>1</weight>
                        <color>1</color>
                        <isnormal>true</isnormal>
```

```
        </output>
</vertex>

<vertex>                                        //четверта вершина
        <name>command to find C</name>
        <input>
                <out>read command2</out>
                <weight>1</weight>
                <color>1</color>
                <isnormal>true</isnormal>
        </input>
        <output>
                <in>calculate C</in>
                <weight>1</weight>
                <color>1</color>
                <isnormal>true</isnormal>
        </output>
</vertex>
<vertex>                                        //п’ята вершина
        <name>B</name>
        <input>
                <out>calculate B</out>
                <weight>1</weight>
                <color>1</color>
                <isnormal>true</isnormal>
        </input>
        <output>
                <in>read command3</in>
                <weight>1</weight>
                <color>1</color>
                <isnormal>true</isnormal>
        </output>
</vertex>
        <vertex>                                //шоста вершина
        <name>C</name>
        <input>
                <out>calculate C</out>
                <weight>1</weight>
                <color>1</color>
                <isnormal>true</isnormal>
        </input>
        <output>
                <in>read command3</in>
                <weight>1</weight>
                <color>1</color>
```

```
                <isnormal>true</isnormal>

</output>
        </vertex>
<vertex>                                                    //сьома вершина
                <name>command to find D</name>
                <input>
                        <out>read command3</out>
                        <weight>1</weight>
                        <color>1</color>
                        <isnormal>true</isnormal>
                </input>
                <output>
                        <in>calculate D</in>
                        <weight>1</weight>
                        <color>1</color>
                        <isnormal>true</isnormal>
                </output>
        </vertex>
                <vertex>                                    //восьма вершина
                <name>D</name>
                <input>
                        <out>calculate D</out>
                        <weight>1</weight>
                        <color>1</color>
                        <isnormal>true</isnormal>
                </input>
        </vertex>
</net>
```

Додаток Р

Частина структури вхідного файлу в XML-форматі

```
<?xml version="1.0" encoding="iso-8859-1"?>
<pnml>
<net id="Net-One" type="P/T net">
<token id="Default" enabled="true" red="0" green="0" blue="0"/>
<place id="P0">                                          //вершина
        <graphics>
                <position x="45.0" y="30.0"/>            //координати вершини
        </graphics>
        <name>
                <value>command to find A</value>         //назва вершини
                <graphics>
                        <offset x="64.0" y="46.0"/>      //розташування напису
                </graphics>
        </name>
        <initialMarking>
                <value>Default,0</value>                 //початкове маркування
                <graphics>
                        <offset x="0.0" y="0.0"/>
                </graphics>
        </initialMarking>
        <capacity>
                <value>0</value>                         //потужність вершини
        </capacity>
</place>
<transition id="T1">                                     //перехід
        <graphics>
                <position x="105.0" y="30.0"/>           //координати переходу
        </graphics>
        <name>
                <value>calculate A</value>               //назва переходу
                <graphics>
                        <offset x="41.0" y="-12.0"/>     //координати напису
                </graphics>
        </name>
        <orientation>
                <value>0</value>                         //орієнтація переходу
        </orientation>
```

```
        <rate>
                <value>1.0</value>                          //швидкість дії


        </rate>
        <timed>
                <value>false</value>                        //чи часовий перехід
        </timed>
        <infiniteServer>
                <value>false</value>                        //чи може спрацьовувати без
зупинок
        </infiniteServer>
        <priority>
                <value>1</value>                    //пріорітет
        </priority>
</transition>
<arc id="P0 to T1" source="P0" target="T1">             //ребро
        <graphics/>
        <inscription>
                <value>Default,1</value>                    //колір та вага ребра
                <graphics/>
        </inscription>
        <tagged>
                <value>false</value>                        //чи позначене ребро
        </tagged>
        <arcpath id="000" x="71" y="42" curvePoint="false"/>   //перелік координат,
які з’єднує ребро
        <arcpath id="001" x="111" y="42" curvePoint="false"/>//перелік координат, які
з’єднує ребро
        <type value="normal"/>                      //тип ребра
</arc>
</net>
</pnml>
```

“ЗАТВЕРДЖУЮ”
В.о. заст. директора з
наукової роботи
ФМІ НАН України
_____ Д.Г. Досин
“10” 11 2016 р.

АКТ

про впровадження результатів дисертаційної роботи
БЕРЕГОВСЬКОГО ВАСИЛЯ ВАСИЛЬОВИЧА
«Математичне та програмне забезпечення автоматизованого проектування систем «інтелектуального будинку»

Даним актом засвідчується, що наступні наукові результати, отримані Береговським В.В. в дисертаційній роботі «Математичне та програмне забезпечення автоматизованого проектування систем «інтелектуального будинку», впроваджені в процесі розробки перспективних радіоелектронних систем, які працюють в режимі реального часу:

- метод синтезу моделей для системного рівня автоматизованого проектування складних систем на основі теорії мереж Петрі;
- моделі, які використовують штучні нейронні мережі, що дає змогу опрацьовувати нечіткі та неструктуровані дані від підсистеми давачів;
- програмно-апаратні моделі реалізації нейроконтролерів для підсистем «інтелектуального будинку».

Отримані в дисертаційній роботі результати представляють практичну цінність при розробці програмно-апаратних засобів для автоматизації проектування систем «інтелектуального будинку». Запропоновані метод та моделі порівняно з відомими відзначаються ефективністю та високою точністю. Це дозволяє, застосовуючи розроблені моделі та метод, підвищити ефективність автоматизованого проектування систем «інтелектуального будинку» і уникати, при цьому, дорогих та довготривалих фізичних експериментальних досліджень та випробувань.

Даний акт не є основою для проведення фінансових взаєморозрахунків.

Зав. відділу методів та систем
дистанційного зондування
ФМІ НАН України,
д.т.н., професор Б.П. Русин