Міністерство освіти і науки України
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерної інженерії

ОЛІЙНИК Олександр Олександрович

«Багаторівнева система управління технологічними процесами / The multilevel process control system for technological processes»

Студент групи КІм – 21
ОЛІЙНИК Олександр Олександрович
_____

Науковий керівник
д.т.н., проф. Цмоць І. Г.
_____

Тернопіль – 2020

РЕЗЮМЕ

Кваліфікаційна робота на тему “Багаторівнева система управління технологічними процесами” зі спеціальності 123 «Комп’ютерна інженерія» освітнього ступеня «магістр» написана обсягом 71 сторінка і містить 9 ілюстрацій, 1 таблицю, 3 додатки та 50 джерел за переліком посилань.

Метою роботи є розроблення структури системи управління технологічними процесами з такими рівнями управління: збору даних та управління виконавчими механізмами; контролю та управління технологічними процесами; операторського контролю та формування управлінських рішень.

Методи досліджень: знайомлення та опрацювання джерел літератури, що стосуються даної теми, аналіз типових виробничих технологічних процесів діючого виробництва, структурування отриманих результатів.

Результати дослідження: на основі побудованого алгоритму розроблено інтегровану систему для обміну та накопичення інформації з розосереджених систем.

Результати роботи можуть бути використані в різного роду досліджень, для покращення уже існуючих систем.

Орієнтовні напрямки розвитку досліджень: інтеграція функцій управління, створення єдиного інформаційного простору з достовірною, повною та оперативною інформацією. Інтеграція в БСУТП здійснюється в таких напрямках: функціональному, організаційному, інформаційному, програмно-алгоритмічному, технічному та економічному.

КЛЮЧОВІ СЛОВА: БАГАТОРІВНЕВІ СИСТЕМИ УПРАВЛІННЯ, ІНТЕГРАЦІЯ СИСТЕМ, КОМПОНЕНТНО-ОРІЄНТОВАНА ТЕХНОЛОГІЯ, ТЕХНОЛОГІЧНІ ПРОЦЕСИ, ПРОГРАМОВАНІ ЛОГІЧНІ КОНТРОЛЕРИ.

RESUME

Qualification work on "Multilevel process control system" in the specialty 123 "Computer Engineering" of the master's degree is written in 71 pages and contains 9 illustrations, 1 table, 3 appendices and 50 sources from the list of references.

The purpose of the work is to develop the structure of the process control system with the following levels of management: data collection and management of executive mechanisms; control and management of technological processes; operator control and formation of management decisions.

Research methods: acquaintance and processing of literature sources related to this topic, analysis of typical production processes of existing production, structuring of the results.

Research results: on the basis of the built algorithm the integrated system for an exchange and accumulation of the information from the dispersed systems is developed.

The results of the work can be used in various kinds of research to improve existing systems.

Approximate directions of research development: integration of management functions, creation of a single information space with reliable, complete and operational information. Integration into BSUTP is carried out in the following areas: functional, organizational, informational, software-algorithmic, technical and economic.

KEY WORDS: MULTI-MANAGEMENT, SYSTEMS INTEGRATION, COMPONENT-ORIENTED TECHNOLOGY, INTELLIGENT MANUFACTURING, PROGRAMMABLE LOGIC CONTROLLERS.

ЗМІСТ

ВСТУП

Актуальність досліджень. Розвиток підприємств характеризується інтенсивним впровадженням інформаційних технологій, які забезпечують автоматизацію технологічних процесів і накопичення великих обсягів інформації про їх динаміку. Управління енергоефективністю на рівні технологічних процесів вимагає створення єдиного інформаційного простору з достовірною, повною та оперативною інформацією про протікання технологічних процесів на підприємстві. Одним із шляхів зменшення обсягів інформації є наближенням засобів опрацювання (мікроконтролерних систем) до джерел надходження інформації (давачів) та виконавчих механізмів. Накопичені дані опрацьовуються з використанням технологій обчислювального інтелекту. Отриманні результати використовуються для формування управлінських рішень. Підвищити рівень управління енергоефективністю технологічних процесів на підприємстві можна шляхом розроблення багаторівневої системи управління технологічними процесами (БСУТП), яка повинна інтегрувати всі функції моніторингу та управління в єдину систему. При розробці БСУЕТП доцільно орієнтуватися на широке використання телекомунікаційних і Web технологій, баз даних, засобів збору, оцінювання, оперативного аналітичного та інтелектуального опрацювання даних, візуалізації результатів їх опрацювання та прийняття управлінських рішень.

Зважаючи на широкий розвиток та впровадження безпровідних засобів обміну даними та зростаюча інтеграція різноманітних систем з Інтернет останніми роками постає задача створення універсальних платформ, які б підтримували як функції забезпечення гарантованого обміну даними з периферійними пристроями, так і функції створення універсальних захищених сховищ даних, доступних, відповідно до наданих повноважень, у будь-якій точці світу. Таким чином, при реалізації БСУТП постає завдання – формування інтегрованої системи обміну та накопичення інформації з розосереджених систем.

Метою є забезпечення об’єднання між собою провідними та безпровідними каналами, взаємодія з Інтернет давачами, мікроконтролерних систем і виконавчих механізмів у БСУТП.

Для досягнення мети магістерської роботи необхідно виконати наступні завдання:

– здійснити аналіз предметної області

– дослідити та проаналізувати базову концепцію БСУТП

– провести синтез структури БСУТП на основі компонентно-орієнтованої технології

Об’єкт дослідження - розробка та реалізація інтегрованої системи обміну та накопичення інформації з розосереджених систем.

Предмет дослідження - метод, який використовує засіб та технології для того, щоб розробити інтегровану систему обміну та накопичення інформації з розосереджених систем.

Методи дослідження - ознайомлення та опрацювання джерел літератури, що стосуються даної теми, аналіз типових виробничих технологічних процесів діючого виробництва, структурування отриманих результатів.

Наукова новизна одержаних результатів. Вперше був запропонований алгоритми функціонування інтегрованої системи обміну та накопичення інформації з розосереджених систем.

Практичне значення одержаних результатів. Практичне значення одержаних результатів полягає в тому, що на основі побудованого алгоритму розроблено інтегровану систему для обміну та накопичення інформації з розосереджених систем.

Публікації та апробація. На основі досліджень зроблено публікації [6,7] в межах конференцій «III Науково-практична конференція молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі» та «Комп’ютерні інформаційні технології: Матеріали школи-семінару молодих вчених і студентів СІТ’2020». (додаток Б, В).

1.АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Опис предметної області

Сучасні БСУТП є зручним інструментом для підтримки прийняття управлінських рішень на рівні управління технологічними процесами. Оператори, які управляють технологічними процесами, концентруються не на задачах інтеграції та опрацювання даних, а безпосередньо на задачах формування ефективних управлінських рішень.

Для забезпечення управління енергоефективністю підприємства на рівні технологічних процесів БСУТП повинна забезпечувати розв’язання таких задач:

1. зберігання та збір у реальному часі, попереднє опрацювання даних про стан технологічних процесів;
2. управління технологічними процесами, прогнозування, та виконавчими механізмами;
3. покращення параметрів технічних засобів у залежності від умов навколишнього середовища;
4. аналіз та складання енергобалансів виробництва та споживання енергоносіїв;
5. зхрещення різноманітних даних за допомогою баз даних і Інтернет-серверів;
6. захищення даних у системі від несанкціонованого доступу;
7. обгрунтування накопичених даних і визначення шляхів зменшення технологічних і невиробничих втрат енергоресурсів;
8. візуалізація багатовимірних даних щодо енергоефективності і представлення результатів опрацювання даних у вигляді графіків і діаграм;
9. формування та контроль управлінських рішень.

Для того, щоб розв’язати поставлені задачі розроблена базова структура БСУТП, яка наведена на рисунку 1.1.

Для опису базової структури використано три рівні:

1. рівень збору даних та керування механізмом виконання;
2. управління та контроль різного роду технологічними процесами;

3. формування та операторський контроль рішень управління.

Завдяки апаратно-програмним компонентам можна побачити специфіку кожного рівня системи управління технологічними процесами. Кожен ієрархічний рівень управління включає в себе задачі відповідного рівня складності.

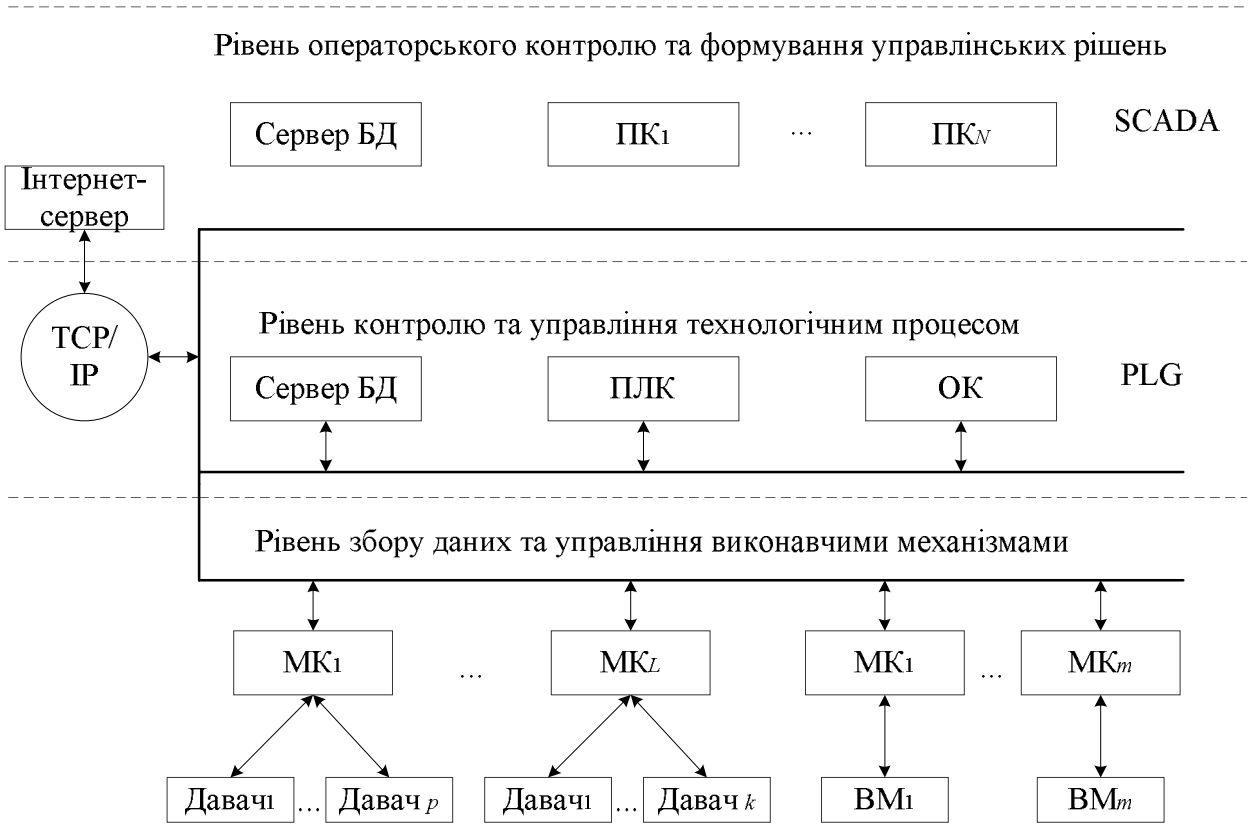

Рисунок 1.1 – Базова структура БСУТП

Рівень управління та збору даних виконавчими механізмами. Формування первинної інформації відбувається на цьому рівні, також ця інформація попередньо опрацьовується, надходить та накопичується на засоби контролю. Формування сигналів для управління виконавчими механізмами та технологічними процесами відбувається за допомогою використання даної інформації. Слід мати на увазі, що підприємства, побудовані за останні роки, вже оснащені засобами збору технологічної інформації, а підбір та інтеграція систем збору технологічних даних здійснюється вже на етапі проектування. У цьому випадку для вирішення задач БСУТП використовуються штатні засоби вказаних систем.

Більш складна ситуація спостерігається на підприємствах, що вже функціонують. На вказаних підприємствах, особливо на підприємствах малого та середнього бізнесу, як правило, не планувалося розгортання систем збору та обробки інформації внаслідок великої їх вартості у минулому. Проведений аналіз типових виробничих технологічних процесів діючого виробництва дозволяє сформулювати низку особливостей, які необхідно враховувати при створенні БСУТП:

1. для моніторингу параметрів технологічного процесу бажано застосовувати технології, що забезпечують оперативність та простоту розгортання;
2. кількість точок вимірювання параметрів може змінюватися, а отже, система повинна мати відкриту архітектуру з можливістю масштабування;
3. апаратна частина системи повинна розроблятися на основі сучасних типових рішень для забезпечення простоти та низької вартості;
4. необхідне широке застосування базових телекомунікаційних протоколів.

Рівень контролю та управління технологічним процесом. На цьому рівні передбачається достатня автономність, яка при відсутності зв’язку з верхнім рівнем – здатна тривалий час працювати автономно, без втрати інформації.

Рівень формування управлінських рішень та операторського контролю. Цей рівень показує операторський контроль який представлений автоматизованим робочим місцем оператора. Рівень включає в собі апаратні засоби, вони можуть вибиратись для виконання на цьому рівні, формується періодичність даних, та показує надмірну складністю алгоритмів, тому його опрацювання показує надійність, яка висувається до системи управління. Перед цим рівнем показані такого роду задачі: мікроконтролерні системи виконують збір даних з периферійних контролерів; збереження та опрацювання даних; обробка, розпізнавання зображень та відео-потоків і сцен в системах технічного зору; розподілені підсистеми виконують синхронізацію; демонстрування візуалізація виконання технологічного процесу та відображення ходу; формування управлінських рішень. Рівень операторського контролю та формування управлінських рішень є особливістю задач:

1. протиріччя та неповнота даних;

2. висока інтенсивність надходження вхідних даних та його постійність;

3. великий обсяг обчислень з переважанням обчислювальних операцій над логічними при опрацюванні відео-потоків, розпізнавання зображень і сцен в системах технічного зору;

4. постійне ускладнення алгоритмів опрацювання та підвищення вимог до точності результатів;

5. можливість розпаралелення опрацювання даних як у часі, так і у просторі.

1.2. Алгоритми функціонування багаторівневої системи управління технологічними процесами

Розроблення алгоритмів функціонування БСУТП**.** Роботу БСУТП можна описати алгоритми функціонування кожного із рівнів системи. Зокрема, система на першому рівні реалізує наступні функції: збір даних з давачів; вплив на стан актюаторів; автономна робота; комунікація з вищими рівнями; виконання команд вищого рівня; попереднє опрацювання даних та ін.

Зазвичай елементи першого рівня є однопоточними та виконують основну програму у безкінечному циклі. Основними завданнями першого рівня БСУТП є виконання команд із ззовні та виконання внутрішньої логіки. Для цього необхідно враховувати можливість перемикання між основними командами.

Саме тому розроблений алгоритм роботи включає в себе ці особливості, а саме циклічне опитування наявності вхідних команд та їх виконання або виконання внутрішнього циклу роботи системи.

Крок 1: Під час запуску системи відбувається ініціалізація системи, яка включає в себе зчитування конфігурації роботи, ініціалізацію портів, ініціалізацію каналів зв'язку, тощо.

Крок 2: Після ініціалізації системи, відбувається сканування вхідних каналів зв’язку на факту отримання зовнішніх команд. У випадку, якщо отримано вхідну команду, необхідно перейти на крок 10. Якщо жодної вхідної команди не отримано, то перехід на крок 3.

Крок 3: необхідно, щоб пристрій виконував команди згідно із внутрішньою логікою роботи системи. Для аналізу стану середовища необхідно послідовно опитати стан і виміри від давачів.

Крок 4: На основі отриманих даних від давачів відбувається їхній попередній аналіз та трансформація, яка може включати в себе нормалізацію, масштабування та інші нескладні маніпуляції.

Крок 5: Необхідно верифікувати зміни у стані середовища, для цього виконується порівняння із попередніми записаними значеннями стану системи. У випадку, якщо виявлено зміни у стані системи – перехід на крок 6. Якщо відсутні зміни – перехід на крок 7.

Крок 6: Необхідно повідомити вищі рівні про зміну у стані системи. Для цього формується вхідне повідомлення та надсилається до рівня ТП.

Крок 7: Оскільки елементи 1-го рівня здатні виконувати нескладну логіку та працювати у якості автоматика, то необхідно перевірити відповідність між внутрішніми сценаріями роботи системи та поточним станом системи. У випадку, якщо необхідно зміні стан актюатора необхідно перейти на крок 8, у протилежному варіанти перехід на крок 2.

Крок 8: Надіслати керуючі сигнали для зміни стану актюаторів згідно із внутрішніми сценаріями роботи.

Крок 9: Надіслати звіт до ТП про зміни у стані актюаторів, перехід на крок 2.

Крок 10: Після отримання вхідної команди, необхідно перевірити та проаналізувати отриману стрічку. Також потрібно отримати тип команди та допоміжні параметри команди.

Крок 11: У випадку, якщо дана команда передбачає отримання даних про стан давача, необхідно перейти на крок 12. У протилежному випадку перехід на крок 14.

Крок 12: Необхідно зчитати стан і поточні результати у давача, який є описаний у команді.

Крок 13: Провести попередній аналіз та трансформації даних від давача, перехід на крок 17.

Крок 14: У випадку, якщо дана команда передбачає зміну стану актюатора, то необхідно перейти на крок 15. У протилежному випадку перехід на крок 16.

Крок 15: Надіслати керуючі сигнали для зміни стану актюаторів, який є описаний у вхідній команді. Перехід на крок 17.

Крок 16: Виконати іншу сторонню команду, згідно із розширеними протоколами зв’язку.

Крок 17: Сформувати та надіслати звіт про виконання вхідної команди, перехід на крок 2.

Блок-схема алгоритму роботи першого рівня БСУТП наведено на рисунку 1.2.

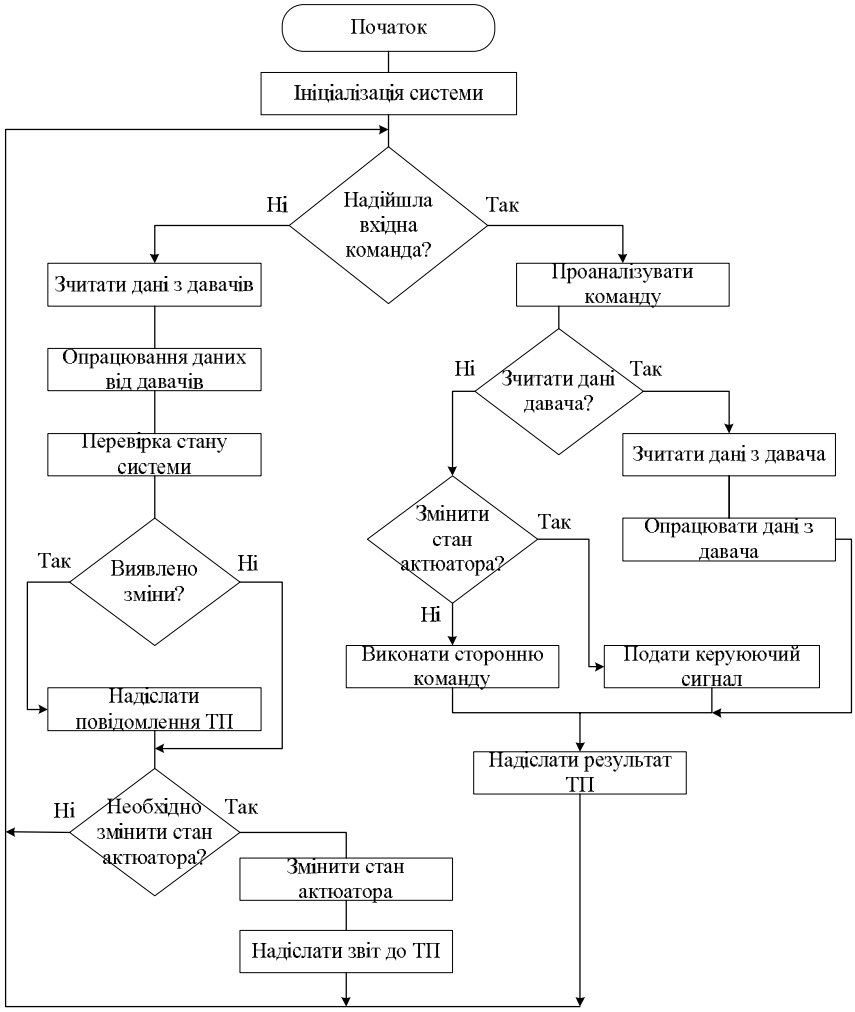

Рисунок 1.2 – Блок-схема алгоритму роботи першого рівня БСУТП

На другому рівні система повинна виконувати такі функції: збереження даних про поточний стан системи ТП; аналіз вхідних даних від нижчого рівня; перевірка стану системи ТП; формування керуючих команд для стабілізації стану ТП; передавання та опитування пристроїв нижчого рівня; контроль зв’язку елементами нижчого рівня та комунікація з елементами нижчого рівня.

На другому рівні проектування можна реалізувати складнішу логіку, яка може в себе включати виконання періодичних завдань, багатопоточність та складнішу комунікацію між рівнями.

Крок 1: Під час запуску системи відбувається ініціалізація системи, яка включає в себе зчитування конфігурації роботи, ініціалізацію портів, ініціалізацію каналів зв'язку, ініціалізацію ядра системи.

Крок 2: Після ініціалізації системи, відбувається сканування вхідних каналів зв’язку на факт отримання зовнішніх команд. У випадку, якщо отримано вхідну команду, необхідно перейти на крок 15. Якщо жодної вхідної команди не отримано, то перехід на крок 3.

Крок 3: Необхідно перевірити статус періодичних завдань. У випадку, якщо настав час для виконання завдання, необхідно перейти на крок 4. У протилежному випадку перехід на крок 8.

Крок 4: Кожному періодичному завданню ставиться у відповідність перелік підзадач для його виконання. Формується перелік підзавдань.

Крок 5: На даному кроці відбувається перетворення підзавдань у команди для системи та елементів нижчого рівня для зчитування даних, зміни стану актюаторів, тощо.

Крок 6: Формується черги із команд для надсилання їх до елементів нижчого рівня.

Крок 7: Послідовне надсилання команд до елементів нижчого рівня, перехід на крок 2.

Крок 8: Запустити внутрішнє логічне ядро. Оновити значення стану середовище та запущених актуаторів.

Крок 9: Провести перевірку роботи системи згідно із внутрішньою логікою роботи ТП. Необхідно перевірити, чи параметри системи знаходяться у межах норми.

Крок 10: Надіслати звіт із змінами у стані ТП до рівня ОК.

Крок 11: У випадку виявлення виходу параметрів системи необхідно перейти на крок 12. У протилежному випадку перехід на крок 2.

Крок 12: Сформувати перелік необхідних завдань для стабілізації стану системи.

Крок 13: Перевести завдання для стабілізації стану системи у команди для елементів МК.

Крок 14: Сформувати чергу із команд для елементів нижчого рівня, послідовне виконання надсилання команд, перехід на крок 2.

Крок 15: Перевірка адресата команди. У випадку, якщо команда надійшла від ОК перехід на крок 16. У протилежному випадку перехід на крок 18.

Крок 16: Перевірити та проаналізувати отриману команду від ОК. Необхідно отримати тип команди та допоміжні параметри команди.

Крок 17: Змінити параметри роботи ТП згідно із командою від ОК. Надіслати звіт про виконання команди, перехід на крок 8.

Крок 18: Перевірити та проаналізувати отриману команду чи звіт від МК. Необхідно отримати тип повідомлення та допоміжні параметри повідомлення.

Крок 19: Оновити значення стану системи, перехід на крок 8.

Блок-схема алгоритму роботи другого рівня БСУТП на рисунку 1.3.

На третьому рівні реалізуються такі функції: збереження великого об’єму даних про стан системи в часі; глибокий аналіз даних про систему; представлення стану системи у зрозумілому для оператора вигляді (SCADA системи); взаємодія з оператором; формування команд для елементів нижчого рівня; контроль зв’язку з елементами нижчого рівня.

На третьому рівні проектування має бути реалізована комунікація із оператор. Тому необхідно враховувати завдання по відображенню стану та двосторонньої взаємодії із оператором.

Блок-схема алгоритму роботи третього рівня БСУТП на рисунку 1.4

Розроблені алгоритми функціонування системи можуть бути використані для побудови моделі на основі теорії мереж Петрі, що дасть змогу дослідити динаміку роботи БСУТП на системному рівні розроблення. За результатами дослідження динаміки системи можна робити висновки про роботу БСУТП.

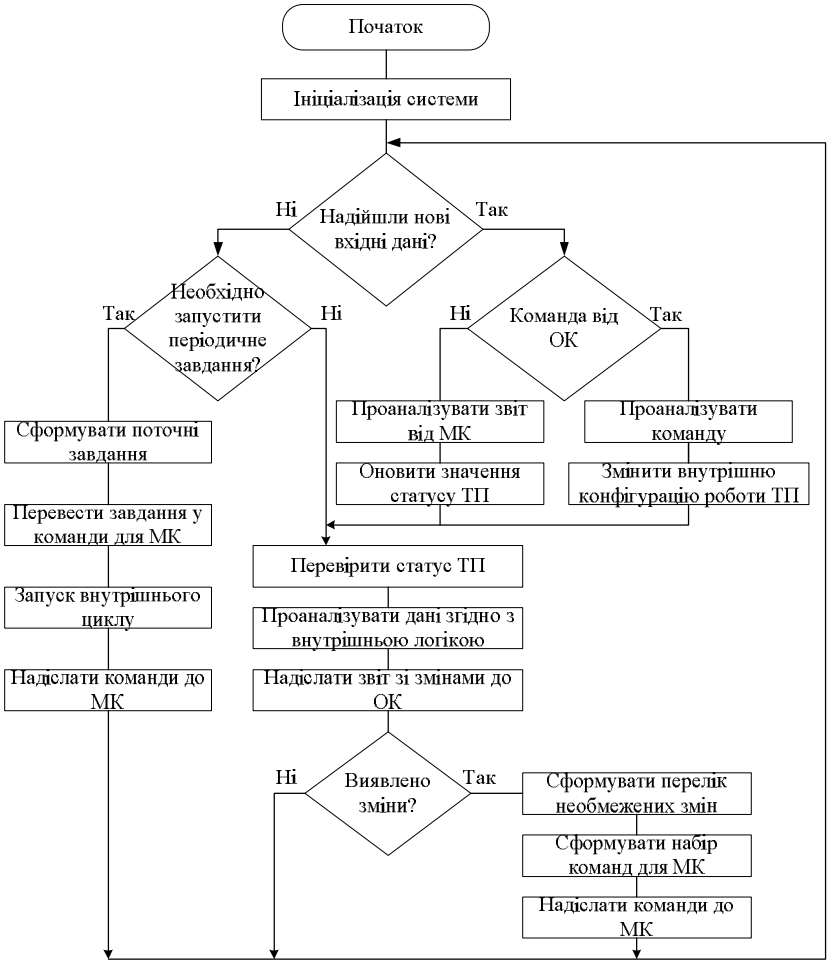

Рисунок 1.3 – Блок-схема алгоритму роботи другого рівня БСУТП

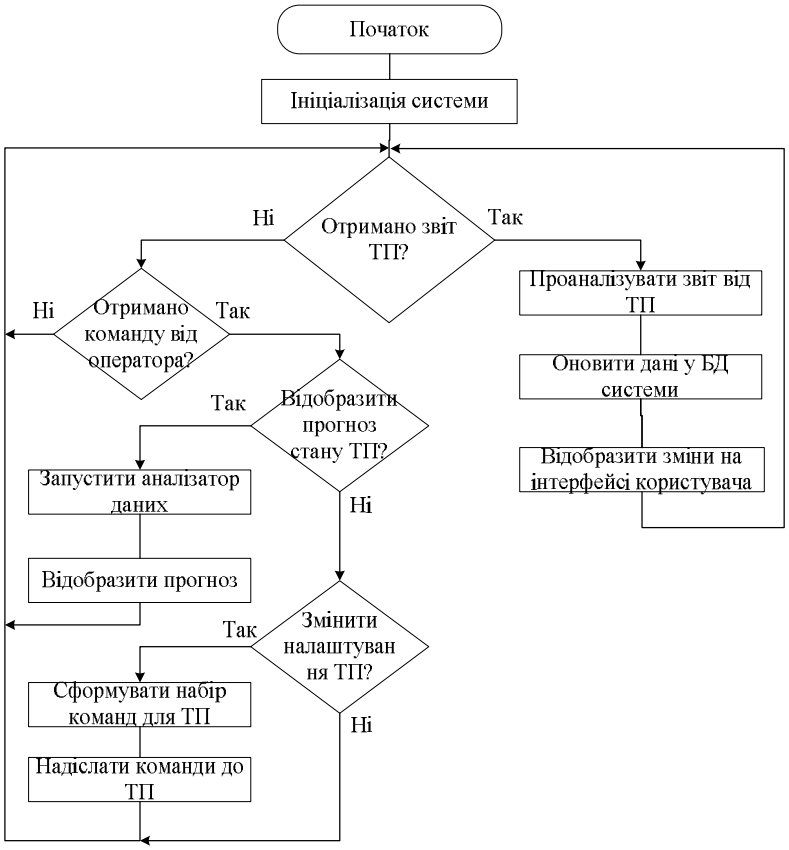

Рисунок 1.4 – Блок-схема алгоритму роботи третього рівня БСУТП

Аналіз нейротехнологій, архітектури, типових проблем та програм алгоритмів нейронної мережі в режимі реального часу показав, що вони мають такі характеристики:

1. Висока інтенсивність та послідовність потоків даних;
2. Постійне ускладнення алгоритмів обробки та підвищення вимог до точності результатів;
3. Можливість паралельного лікування як у часі, так і в просторі;
4. Здатність до узагальнення та абстрагування;
5. Навчання, самонавчання та самоорганізація під впливом навколишнього середовища.

Аналіз показав, що доставка в режимі реального часу інтенсивного потоку даних нейронної мережі може бути апаратною. Апаратні нейронні мережі в режимі реального часу базуються на операційній основі, показаній на рисунку 1.5.

Оперативну основу нейронної мережі складають основні операції попередньої обробки та обробки. Для апаратної реалізації даних основних операцій використовуються елементарні арифметичні операції (рисунок 1.5).

На етапі попередньої обробки перші дані, що доставляються до мережевих об'єктів, повинні бути перетворені у форму, яка дає найкращий результат. Вектор навчання містить значення для кожного вхідного сигналу мережі, а залежно від типу навчання (з викладачем чи без нього), значення для кожного вихідного результату мережі. Підготовка мережевого пакету "сировина" зазвичай не дає якісних результатів. Існує ряд способів поліпшити «сприйняття» мережі [8]:

– Нормалізація виконується, коли дані різного розміру подаються через різні входи мережі. Наприклад, перший вхід подається в сітку зі значеннями від нуля до одиниці, а другий - від ста до тисячі. За відсутності нормування значення другого входу завжди матимуть набагато більший вплив на вихід мережі, ніж значення першого входу. При нормалізації розмірність усіх вхідних та вихідних даних зменшується до однієї області;

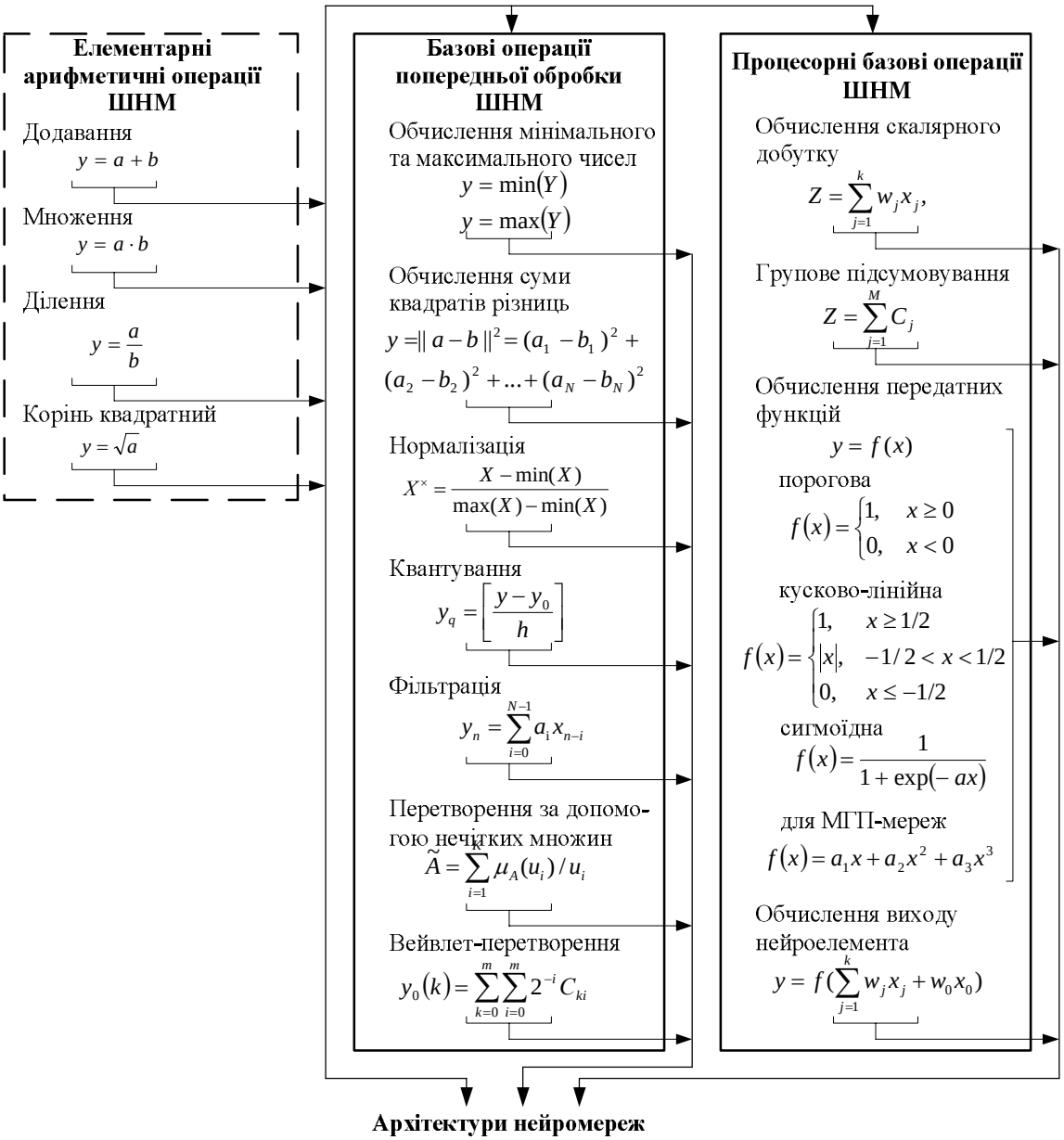

Рисунок 1.5 - Операційний базис апаратних нейромереж реального часу

– Кількісне визначення виконується для безперервних величин, де присвоюється остаточний набір дискретних значень. Наприклад, квантування використовуються для визначення частот звукових сигналів при розпізнаванні мови;

Фільтрування здійснюється для "галасливих" даних і викидає значення, які, ймовірно, є неправильними.

Рекомендується завжди нормалізувати введення. Нормалізація - це процедура попередньої обробки вхідних даних (навчання, випробування та зразки

роботи), при якій значення характеристик конструктора вхідних векторів зменшуються в межах заданого діапазону. Після нормалізації всі значення вхідних характеристик зменшуються до вузького діапазону (зазвичай [0, 1] або [-1, 1].

Існує багато способів нормалізації вхідних значень. Найпростіша, але в більшості випадків ефективна лінійна нормалізація. Якщо вихідні дані потрібно зменшити в діапазоні [0, 1], вони будуть виконані наступним чином:

$$X^{\times} = \frac{X - \min(X)}{\max(X) - \min(X)}. \tag{1.1}$$

Для приведення початкових даних до діапазону [-1, 1] лінійна нормалізація здійснюється таким чином:

$$X^{\times} = \frac{X}{\max(|X|)} \tag{1.2}$$

Якщо вхід X заповнює певний інтервал щільно, оптимальним є використання лінійної нормалізації, оскільки воно не вимагає складних розрахунків.

Однак лінійна нормалізація не завжди є ефективною, наприклад у випадках, коли є рідкісні відхилення, які значно перевищують типові значення. У таких випадках лінійна нормалізація призведе до того, що більшість вихідних значень даних будуть близькими до нуля.

Стандартизація зі стандартними відхиленнями не має цього недоліку:

$$X^{\times} = \frac{X - \overline{X}}{\sigma},\ \overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i,\ \sigma = \frac{1}{n\text{-}1}\sum_{i=1}^{n}\left(X_i - \overline{X}\right)^2. \tag{1.3}$$

Ця нормалізація призводить до більш рівномірного розподілу вхідних даних. Але нормалізація стандартного відхилення має суттєвий недолік - отримані нормовані значення не обов'язково належать до одного інтервалу. Це не важливо для вхідних даних, але вихідні дані можна використовувати як еталонні значення для виходу нейронів. І це неприпустимо, коли функція активації нейронів сигмоїдна, оскільки вона отримує значення лише в одній області.

Нелінійна нормалізація не має цього недоліку, наприклад:

$$X^{\times} = f\left(\frac{X - \overline{X}}{\sigma}\right), \quad f(a) = \frac{1}{1 + e^{-a}} \tag{1.4}$$

Дана функція нормалізує більшість значень рівномірно і гарантує, що нормалізоване значення буде в діапазоні [0, 1]. Недоліком такої нормалізації є складність для апаратної реалізації.

У більшості випадків, якщо вхідні дані є більш-менш рівномірними, для апаратної реалізації найкраще використовувати лінійну нормалізацію (1.2). Для апаратної реалізації такої нормалізації потрібно розробити методи та структури для обчислення наступних базових операцій:

– визначення максимального числа з групи чисел;

– ділення.

Після нормалізації даних, залежно від типу мережі, можуть використовуватися інші процедури попередньої обробки даних.

Зокрема, після нормалізації даних у RBF- та GRNN-мережах потрібно здійснити обчислення евклідової відстань від кожного вхідного вектора до всіх інших. Для цього обчислення використовується базова операція обчислення суми квадратів різниць:

$$y = \| x_i^e - x_i^b \|^2 = (x_1^e - x_1^b)^2 + (x_2^e - x_2^b)^2 + \ldots + (x_N^e - x_N^b)^2. \tag{1.5}$$

Інші типи попередньої обробки можуть бути використані для інших типів нейронних мереж. Наприклад, перетворення для нейронних мереж здійснюються з використанням незрозумілих наборів та нечітких логічних правил щодо вхідних даних, які мають хороші апроксимаційні властивості. Вейвлет-нейронні мережі використовують хвильові перетворення, такі як хвилі Хаара або хвилі Добеші, для аналізу компонентів різних частот вхідних даних.

1.3. Базові апаратно-програмні компоненти багаторівневих систем управління технологічними процесами

На сучасному етапі розвитку інформаційних технологій розвиток BSUTP зводився до вирішення проблеми інтеграції готових апаратних та програмних компонентів, оскільки розробка та виробництво нових вимагає значних ресурсів та часу. При вирішенні проблеми інтеграції необхідно враховувати багато факторів, а саме інформацію про готові апаратні та програмні компоненти, їх технічні характеристики, відповідність інтерфейсам стандартам, можливість закупівель тощо.

Розробка сучасних систем управління базується на системній інтеграції, яка базується на структурованій структурі, яка охоплює всі рівні взаємодії в процесах, об’єктах, пристроях та інфраструктурі, враховуючи їх ефективність та конкретні вимоги до застосування.

Збільшення кількості ієрархічних рівнів рівносильно підвищенню класифікації апаратного та програмного забезпечення. Рейтинги високого рівня, інформаційні блоки, алгоритми, програмне забезпечення та засоби організовані у вигляді інформації та типів алгоритмів, програмного забезпечення та інструментів на нижчих рівнях. Метод послідовного розподілу, що використовується для розробки BSPC, відображає процес розробки зверху вниз.

Інструменти та програмне забезпечення, що використовуються для створення BSUTP, повинні забезпечувати вирішення проблем у режимі реального часу. Зокрема, це функції фільтрації, обробки даних та контролю шаблонів виконання. Комп’ютерні деталі, прикріплені до датчиків і виконавчих механізмів, мають високі вимоги до безпеки та надійності. У той же час до цих категорій пред'являються жорсткі вимоги щодо ваги, розміру та енергоспоживання. Частини бази даних BSUTP повинні перевірити ефективність та швидко виявити проблеми.

Запропоновано проводити BCSTP із використанням системного підходу, що враховує всі стадії процесу та інтеграцію пристрою. BSUTP був розроблений на основі компонентів готового обладнання, оскільки розробка та виробництво нового

вимагає великих грошей та часу. При виборі компонентів необхідно враховувати низку факторів, а саме: інформацію про готові інструменти та програмне забезпечення, їх технічні характеристики, відповідність стандартам інтеграції, варіанти придбання та багато іншого. На кожному рівні ICS рекомендується визначити перелік найважливіших компонентів пристрою на основі інтеграції пристроїв на цих рівнях.

Нещодавно на ринку з’явились 32-розрядні мікроконтролери та готові блоки на їх основі, як і архітектура ARM від ST Microelectronics. З частотою в сотні МГц, оперативна пам'ять за командою в сотні Кбайт, Flash, вбудований у Flash, і обмін UART, SPI, I2C, CAN забезпечують створення пристроїв збору та управління даними, які забезпечують реалізація складних алгоритмів.

З практичної точки зору створення пристроїв на цьому базовому рівні архітектури BSUTP, перш за все, необхідно враховувати одну назву з назвою флеш-пам'ять, оперативна пам'ять, Eeprom, а також віддалені записи. Це робить дуже простим написання коду, а номери різних пакунків та бібліотек можна легко змінити, як це зазвичай стосується дизайнів фон Неймана (тобто адресного рядка). Автобуси для доступу до різних типів пам'яті відокремлені, що вказує на існування Гарвардської будівлі. Ці 8- та 32-розрядні мікроконтролери легко інтегруються в систему з різними датчиками та органами управління, оскільки вони складають найбільші промислові біржі.

Часто для створення складних інструментів використовують мови C, C ++ та композитні мови. Безкоштовні бібліотеки драйверів та середовище програмування розроблені для різноманітних мікроконтролерів, що значно полегшує розробку програмного забезпечення.

Нещодавно корпоративні передачі даних були розроблені на основі технологій Ethernet та Wi-Fi завдяки наявності кабельних систем, активного та пасивного мережевого обладнання та широкого спектру. Технологія бездротового Wi-Fi найзручніша з точки зору виробництва, коли потрібні спрощені вимоги щодо мобільності, встановлення та використання (звичайно, з огляду на електромагнітну сумісність існуючого технологічного обладнання та відсутність перешкод та впливу на роботу мережі). Перевагою є гнучкість архітектури мережі, можливість

динамічно змінювати топологію мережі, швидкість проектування та реалізації, що важливо при жорстких вимогах до довжини мережі, відсутність необхідних проводів та кабелів, підтримка стеку TCP / IP телекомунікаційних протоколів.

Для передачі даних через бездротову мережу потрібен спеціальний радіомодем, щоб забезпечити, щоб фізичний і канальний рівень моделі OSI працював у бездротовій мережі, а для підтримки вищого рівня моделі OSI, продуктивність цього мікропроцесора недостатня для складних розрахунків. Сьогодні розробляються спеціалізовані пристрої - модулі Wi-Fi RS232, але вартість порівнянна з вартістю основної системи мікроконтролера.

Однак за останні роки з’явився новий клас спеціалізованих мікроконтролерів, які забезпечують повну підтримку Wi-Fi та стеку протоколів TCP / IP. Крім того, в мікросхемі розміщений досить потужний процесор, оперативна пам'ять і досить вдосконалені зовнішні пристрої з підтримкою SPI, I2C та UART. На основі такого пристрою можна створити повноцінний бездротовий пристрій, який може приймати дані від датчика, виконувати попередню обробку та передавати інформацію за допомогою протоколів зв'язку на локальний або віддалений сервер. Такі інструменти дали новий поштовх розвитку технології Інтернет речей. Крім того, готові промислові модулі на основі цих спеціалізованих мікроконтролерів можна використовувати на рівні BSUETP. Серед таких пристроїв слід зазначити мікроконтролер Nufront NL6621 від RTL8710 від Realtek, ESP8266 від Espressif. Наприклад, ESP8266 надає повну підтримку Wi-Fi (режими хосту та точки доступу з протоколами автентифікації WEP та WPA / WPA2), а також стек протоколів TCP / IP. ESP8266 має вбудований 32-розрядний процесор Tensilica Xtensa LX106 RISC зі швидкістю 80 МГц, 64 КБ оперативної пам'яті та 96 КБ оперативної пам'яті, а також підтримує зовнішню пам'ять мікропрограми від 512 КБ до 16 МВт. Чіп підтримує UART, SPI, I2C і містить один канал з 10-бітовим АЦП.

Для побудови прошивки доступні програмні засоби та бібліотеки, що підтримують широкий спектр датчиків. Мову C можна використовувати для розробки, в деяких випадках розвитку сприяє використання перекладача мови Lua.

Один комп'ютер, заснований на спеціальних процесорах SoC, може бути використаний для реалізації більш складних обчислювальних алгоритмів на

нижчому рівні. Єдиний мікрокомп’ютер - це перспективна платформа для систем автоматизації, яка має відкриту архітектуру, низьку вартість і використовує операційну систему Linux. Мікрокомп’ютер Raspberry Pi був одним з перших, хто з’явився в цьому класі в 2012 році, і побудований на системі Broadcom BCM2835 SoC, яка включає в себе ARM-процесор 700 МГц, графічний процесор VideoCore IV і 512 або 256 МБ оперативної пам’яті. Пам'ять, SD-карта використовується як додаткова пам'ять. Мікрокомп'ютер доступний у декількох версіях: молодший (A) (700 МГц, 256 Мб оперативної пам'яті, один порт USB), старіший (B) (1,2 ГГц, Ethernet, до 1 Гб оперативної пам'яті, 4 USB (порти), нещодавно був дешевша версія - нуль (тактова частота до 1 ГГц, 512 МБ оперативної пам'яті, один порт microUSB) Raspberry Pi має порти GPIO, підтримку інтерфейсів UART, SPI, I2C, які можна використовувати для підключення та управління датчиками.

Нещодавно список комп’ютерів на одній картці SoC значно розширився, наприклад, процесор BeagleBone Black AM3359 ARM Cortex-A8 від Texas Instruments; Intel Edison Про процесор Intel Atom; процесор pcDuino ARM A10 від AllWinner; Процесор Cubieboard2 A20 ARM A20, повнолінійний процесор OrangePI H3, процесор H2 + від AllWinner. Всі ці однокамерні комп'ютери мають вдосконалену периферію, яку можна підключити до різноманітних датчиків і забезпечити бездротову мережу.

Що стосується розвитку, то це повноцінні системи Linux. Для реалізації алгоритмів використовуйте мови C, C ++, Python, рідко складені мови. Проблема може полягати у відсутності програмного забезпечення для конкретного окремого комп’ютера, але проект Armbian, який розробляє дистрибутив Debian для SoC на базі ARM, може допомогти вам вирішити цю проблему, і кожен комп’ютер вже має готові та готові до роботи інструменти. використання.

DCS - розподілена система управління та PLC - програмований логічний контролер (DCS) використовуються для безпосереднього управління приводами.

Для вирішення проблем BSUETP на цьому рівні, що вимагають високої пропускної здатності, рекомендується використовувати FPGA як готові промислові модулі FPGA / CPLD Altera - MAXII EPM240, Altera Cyslone ll EP2C5T144, Altera Cyclone IV EP4CE6, XilinCA II Sp3 Productive. Розробка FPGA початкового рівня

може бути здійснена за допомогою безкоштовних версій програмного забезпечення.

Перетворювач коду повинен забезпечувати паралельне послідовне та послідовне паралельне перетворення. Паралельне послідовне перетворення виконується шляхом взяття кількості N-чисел у кожному циклі та видачі бітових бітів усіх чисел у кожному циклі. Це перетворення використовується для завантаження вертикальних робочих одиниць, де виконується вертикальна обробка, забезпечуючи послідовну обробку всіх 4 цифр кожного числа в кожному циклі, або паралельну обробку цифр шляхом послідовної обробки бітів. Послідовне паралельне перетворення використовується для перетворення результатів обробки вертикальних робочих одиниць. При послідовно-паралельному перетворенні вхідні результати приймаються як біти, а вихідні дані подаються послідовно.

На другому етапі операції реєстрації даних виконуються безпосередньо під час навчання та роботи в мережі. Аналіз існуючих алгоритмів показав, що всі основні операції з датою в нейронних мережах можуть бути зведені до таких основних операцій:

- скалярний розрахунок товару;
- резюме груп;
- розрахунок передавальних функцій.

Серед числа операцій, що найчастіше використовуються в нейро-алгоритмах, особлива увага приділяється підрахунку суми взаємопов'язаних продуктів [20, 21, 26]. Традиційно розрахунок такої операції виконується за такою формулою:

$$Z = \sum_{j=1}^{k} W_j X_j,, \tag{1.6}$$

де k – кількість входів нейроелемента,

W_j – j-й ваговий коефіцієнт, X_j – значення j-го входу.

Існує два підходи до впровадження техніки для обчислення суми взаємопов’язаних продуктів [100, 101]. Перший заснований на операціях множення та додавання, другий - на операціях додавання, інверсії та зсуву. Перший підхід в

основному використовується для обчислення суми спарених продуктів при синтезі пристроїв на основі окремих мікросхем (мультиплікаторів, надбудов), а другий - у додатках VLSI. Крім того, використовуючи алгоритми для реалізації VLSI, крім того, за допомогою інструментів на основі операцій інверсії та компенсації можна оптимізувати пристрій для збільшення швидкості, апаратних витрат та регулярності структури. Основою таких алгоритмів є генерація часткових добутків з подальшим їх додаванням.

Ключовим елементом розділу розрахунку скалярної продукції є додавання позашляхового обладнання. Загалом, розрахунок скалярного добутку на основі елементарних арифметичних операцій зводиться до макрооперації групової суми продуктів макрочастинок:

$$Z = \sum_{j=1}^{M} C_j \,, \tag{1.7}$$

де M – кількість доданків;

C_j – j-й доданок [100].

Сигнал, отриманий після розрахунку скалярного добутку, перетворюється у вихідний сигнал за допомогою алгоритмічного процесу, відомого як передавальна функція. Сума в передавальній функції для визначення виходу нейронів порівнюється з деякими порогами. Якщо сума перевищує порогове значення, обробний елемент генеруватиме сигнал, інакше сигнал не генерується або гальмовий сигнал.

Бажано використовувати нелінійну передавальну функцію, оскільки лінійні (лінійні) функції обмежені, а вихідний вхід прямо пропорційний. Використання функцій лінійної передачі було проблемою на ранніх моделях мереж, і було доведено їх обмеження, а не доцільність [10].

На рисунку 1.14 зображені типові передатні функції [7].

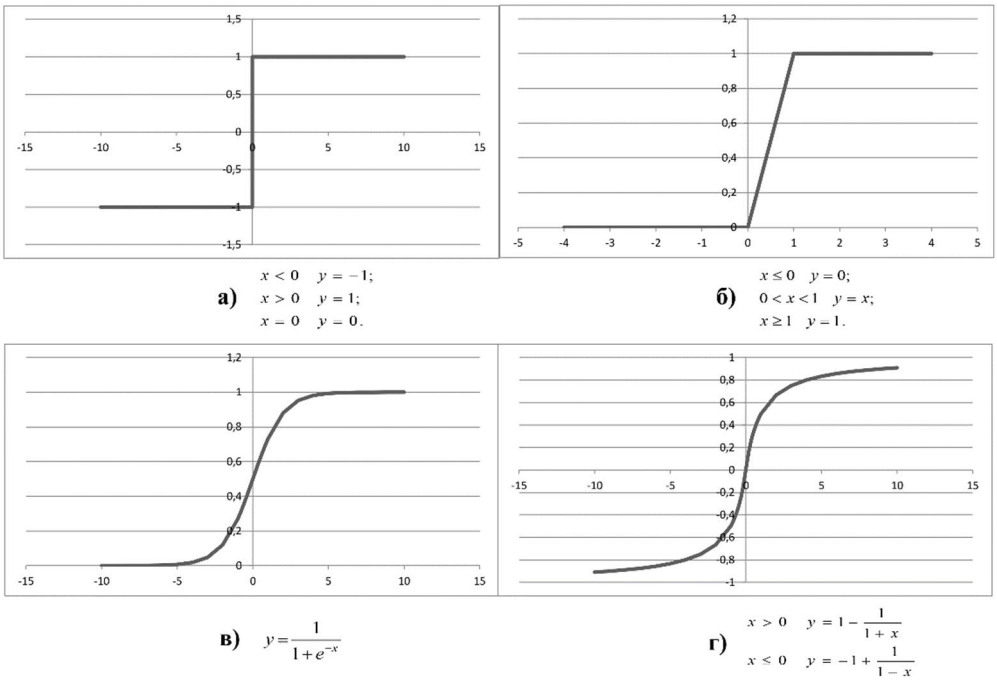

Рисунок 1.6 - Типові передатні функції: жорстка порогова функція (а), лінійна з насиченням (б), сигмоїда (в), гіперболічний тангенс (г)

Для полегшення функції передачі нейронні мережі можуть надсилати 0 та 1, 1 та -1 або інші числові комбінації. У таких випадках функція передачі є "жорстким обмежувачем" або пороговою функцією (рис. 1.14a).

Інший тип передавальної функції - лінійна насиченість - відображає вхід до даної області і діє як тверде обмеження поза цією областю. Це лінійна функція, зменшена на мінімальне та максимальне значення, що робить її нелінійною (рис. 1.14b).

Наступним вибором є сигмоїдна або S-подібна крива, яка наближається до мінімальних і максимальних значень в асимптотах і називається сигмоїдною (рис. 1.14c), коли її площа дорівнює [0, 1], або гіперболічною дотичною (рис. 1.14d) [-1, 1] Важливою особливістю кривих є неперервність функцій та їх похідних. Використання сигмоподібних функцій дає хороші результати і широко використовується.

Нарешті, інші функції передавача можна вибрати для різних нейронних мереж.

Перед обчисленням функції передачі вхідний сигнал іноді доповнюється

рівномірно розподіленим випадковим шумом, де джерело та кількість визначаються в режимі навчання. У літературі цей шум називають "температурою" штучних нейронів, надаючи математичній моделі елемент реальності.

1.4 Висновки до розділу

В даному розділі описано предметну область, алгоритми функціонування багаторівневої системи управління технологічними процесами, базові апаратно-програмні компоненти багаторівневих систем управління технологічними процесами.

При реалізації БСУТП постає ще одна задача – формування інтегрованої системи обміну та накопичення інформації з розосереджених систем, метою якої є забезпечення об’єднання між собою провідними та безпровідними каналами, взаємодія з Інтернет давачів, мікроконтролерних систем і виконавчих механізмів у БСУТП.

2 БАЗОВА КОНЦЕПЦІЯ БСУТП

2.1 Базова концепція

За останні десять років було розроблено кілька мов програмування PLC для рівня IEC 61131. Цей стандарт контролює зміни у складі мови програмування системи управління системами. Цей стандарт охоплює вимоги до обладнання, встановлення, тестування, документації, зв'язку та програмування ПЛК. Стандарт визначає 5 мов програмування, які поділяються на текст та зображення. Письмові мови СФС включають: - Індекс (ІЛ) - мову викладання; - Структурований текст (ST) - Мова структурованого тексту. IL - традиційний компілятор з акумулятором, який несе сигнали. Інструкції з голосування стандартизовані та не залежать від конкретного цільового набору. Ця мова забезпечує роботу для всіх типів даних, функцій виклику та функціональних блогів, реалізованих іншими мовами. Використання «складних» алгоритмів ІЛ.

ST - це вдосконалена мова, яка, звичайно, схожа на Паскаль. Замість використання мовної системи Pascal ST використовує частини програми IEC. До графічних мов PLC належать: - Системне планування (SFC) - мова для послідовних діаграм продуктивності; - Блок діаграми продуктивності (FBD) - мова діаграми продуктивності; - Сходи (LD) - мови ланцюга передачі; SFC - це графічний інструмент високого рівня, діаграми якого складаються з кроків і переходів. Схвалення переходу є ситуативним, а дії пов'язані з конкретними діями. Мова FBD зосереджена на описі структури електронних пристроїв та силових модулів. Нагрівачі FBD використовуються для передачі будь-якого типу сигналу (логічного, аналогового, синхронізації тощо). Сигнали від блочного виробництва можуть надходити до іншого блок-виробництва або безпосередньо до виробництва ПЛК. Блоки, показані на діаграмі чорного ящика, можуть виконувати будь-які завдання. Мова FBD пояснює взаємозв'язок між виготовленням та виготовленням діаграм. Якщо алгоритм добре описаний з точки зору символів, його подання у FBD завжди чіткіше, ніж у мовах тексту. Мова LD зосереджена на реалізації структури схеми. Схема LD представлена у вигляді двох стаціонарних нерухомих шин. Між ними

складаються схеми з'єднання з'єднань. Допускається будь-яке кругле навантаження.

Кожне реле має клеми, які можна використовувати для інших ланцюгів. Послідовний (I), підключений (AMA) підключений і підключений (NO) - основа логічного типу. Як результат, мова LD є корисною не лише для побудови передавальних машин, а й для реалізації програмного забезпечення для інтегральних схем. Завдяки можливості використання служб LD та функціональних блогів іншими мовами, область застосування розширюється. Мови IEC 61131 засновані на найпопулярніших мовах програмування для інтелектуальних годинникових виробників. Програми, написані для інтелектуальних контролерів, можуть бути перенесені в середовище IEC 61131-3 за нижчою вартістю.

Особливістю стандарту IEC 61131 є можливість створення незалежних від апаратного забезпечення бібліотек для реалізації процесорів, фільтрів, елементів керування приводами, модулів, включаючи інтелектуальні модулі, та багато іншого. CoDeSys (Supervisor Monitoring Programs) розроблений для моніторів, редакторів та інструментів з використанням мови програмування IEC 61131-3 і заснований на принципах загальних професійних середовищ програмування (VisualC ++ та ін.). Особливістю програмного середовища CoDeSys є те, що воно не призначене для певної апаратної програми. Середовище програми змінюється для різних ПЛК шляхом адаптації програми до низькорівневих ресурсів, таких як розподіл пам'яті, обмін даними та взаємодія вводу-виводу.

Середовище CoDeSys пропонує: пряме формування машинного коду; повне впровадження мов IEC 61131-3; інтелектуальні мовні редактори для виправлення помилок початківців; вбудований емулятор управління, що дозволяє усувати проблеми проекту без додаткового обладнання; Вбудовані елементи візуалізації дозволяють створити модель об'єкта управління та налаштувати проект без використання інструментів моделювання. Використовуйте готові бібліотеки та сервісні функції. Пристрій технології вуликів. Технологія Device Hive використовується для підключення дротових та бездротових каналів, а також для підключення датчиків, систем мікроконтролера та виконавчих механізмів до

Інтернету. Ця технологія є гнучкою, масштабованою та простою у використанні, забезпечуючи зв'язок M2M між апаратними компонентами BSUTP. Використання технології Device Hive дозволяє створювати середовище спілкування, керувати програмним забезпеченням та використовувати багатоплатформні бібліотеки для розробки дистанційного керування та моніторингу, телеметрії, дистанційного управління та моніторингу. Технологію Device Hive можна використовувати з різними технологіями, такими як: B. Вбудовані бібліотеки Linux, Python, C ++, протокол JSON або Connect AVR, мікроконтролери Microchip.

Однією з особливостей роботи з Device Hive є організація першого доступу до мережі та подальшого програмування конкретних програм, що скорочує час, необхідний для розробки методів передачі даних. Мікроконтролерні системи. Програмовані контролери Mitsubishi FX3U - це найпотужніші та найпотужніші контролери MELSECFX (FX1S, FX3S, FX1N, FX3G). Архітектура контролера складається з двох шин, які розширюють функції. PLC Mitsubishi з'єднує як старі модулі розширення, так і модулі нового покоління FX3U. Коли модулі FX3U підключені, контролер автоматично перемикає шину зміни у високошвидкісний режим, і обмін даними відбувається на більш високій швидкості. Модулі FX0N, FX2N працюють з контролером на звичайних швидкостях. За допомогою модулів розширення кількість входів / виходів збільшується до 256 (з безпосередньою адресою) та до 384 через децентралізовані станції вводу-виводу. Крім того, програмований контролер Mitsubishi пропонує підключення до модулів швидкісного адаптера FX3U-XXX-ADP, які розширюють функції контролера при роботі з аналоговими сигналами та збільшують кількість додаткових комунікаційних інтерфейсів (RS232 / 422/485) [11].

Однокомп’ютерний мікрокомп’ютер Raspberry Pi - це перспективна платформа для систем автоматизації, відома своєю низькою вартістю, операційною системою та відкритою архітектурою. Мікрокомп'ютер Raspberry Pi заснований на системі Broadcom BCM2835 (SoC), яка включає процесор ARM 700 МГц, графічний процесор VideoCore IV і 512 або 256 мегабайт оперативної пам'яті, а для додаткового зберігання використовує SD-карту. [12]. Цей мікрокомп'ютер доступний у двох версіях: молодша (A) (256 Мб оперативної пам'яті, один порт

USB) та старіша (B) (з Ethernet, 512 Мб оперативної пам'яті, два порти USB). Для Raspberry Pi випущено спеціальний дистрибутив Linux, Raspbian, ОС (заснований на дистрибутиві Debian) та ряд додатків Pi Store, які мають як платні, так і безкоштовні програми. Raspberry Pi має власні порти GPIO, які можна використовувати для різних функцій. Серія мікроконтролерів STM8 - це відносно нова серія мікроконтролерів від ST Microelectronics, яка містить майже все, що можна очікувати від 8-бітного програмованого контролера. Переваги контролерів STM8 включають, по-перше, низьку вартість не тільки самого контролера, але також плат для усунення несправностей, програмістів та засобів усунення несправностей. За допомогою вбудованого завантажувача ви можете завантажувати програми через UART, SPI, CAN або I2C. Надається безкоштовна бібліотека периферійних драйверів. STM розробив бібліотеку мікропрограмного забезпечення STM8x, структуру для мікроконтролерів.

Загалом, вимоги до електромагнітної сумісності та надійності зростають протягом усієї серії STM8, що також є привабливим для їх використання [14]. До недоліків можна віднести відсутність контролерів з USB і в невеликих корпусах виводу (тип SO8). В даний час існує три сімейства STM8: - STM8S - «Стандартний» універсальний контролер, зазвичай 10-бітний аналоговий периферійний пристрій, із середніми стандартами потужності та діапазоном потужності - 2,95-5,5 В. - STM8L - контролери "низької потужності" з низьким енергоспоживанням, 12-бітна аналогова периферія, поліпшена електромагнітна сумісність та діапазон потужності - 1,8-3,6 В. У порівнянні зі стандартними контролерами, це сімейство додає деякі зовнішні пристрої, включаючи DMA. - STM8A - "Автомобіль" - сімейство, орієнтоване на безпеку та надійність, може витримувати більше навантаження на ноги, діапазон потужності - 2,95-5,5 В і працює при 145 градусах [15]. STM8 має 8-бітове ядро CISC і містить 6 регістрів: A - Акумулятор - однобайтовий акумулятор, що містить результат арифметико-логічної команди; X, Y - індекс - двобайтові індексні регістри, які можуть містити відносну адресу комірок пам'яті, з якими можна виконувати операції; Лічильник програм на ПК - трибайтовий лічильник програм, що містить абсолютну адресу комірки пам'яті, що містить команду, яка тепер буде виконана; SP - покажчик стека - двобайтовий

покажчик поверх стека, що має власну пам’ять; CC - коди умов - однобайтовий реєстр прапорів (прапорів), що містить бітові властивості для арифметично-логічних результатів команд, які можуть бути використані для гілок програми [8].

Мікроконтролер STM8 має 32 вектори переривань, кожен з яких становить 4 байти (великі байти містять захищене значення 82 години) перед запуском відповідної інструкції щодо події, що спричиняє затримку. Закрийте менеджер вимкнення командою IRET. Тактову частоту STM8 для мікроконтролерів можна змінювати під час роботи, на відміну від AVR, коли годинник жорсткий. У додатках з низьким енергоспоживанням це іноді використовується, і коли генератор виходить з ладу, контролер перемикається на RC і переходить у режим очікування, залишаючи систему поза контролем. Пам'ять мікроконтролера умовно розділена на 64K частин, а кожен розділ - 256 сторінок з 256 байтами. Мікроконтролер STM8 дозволяє використовувати наступні методи: прямий, індексний, прямий, стековий, непрямий, бітовий та відносний. Залежно від довжини операнда режим адреси поділяється на короткий - 1 байт, довгий - 2 байти та розширений - 3 байти. З практичної точки зору спочатку слід розглянути єдине місце з 24-бітовою адресою, яке містить флеш-пам’ять, оперативну пам’ять, Eeprom та периферійні регістри. Це спрощує написання коду, наприклад, функції ОЗП та масиву Flash не вимагають кількох копій. Різні вузли стека та бібліотеки легко перетворюються, оскільки вони в першу чергу розроблені для архітектури фон Неймана (маючи на увазі адресний простір). Автобусам присвоюються різні типи пам'яті, що вказує на існування архітектури Гарварда [16].

Розробка в реальному часі високоефективних апаратно-нейронних мереж із використанням обладнання буде базуватися на компонентно-ієрархічному підході, який передбачає розподіл процесу розробки на ієрархічні рівні та типи програмного забезпечення (алгоритм, апаратне та програмне забезпечення).

Алгоритмічна підтримка нейронних мереж включає опис моделей нейронних мереж та розробку методів та алгоритмів навчання та роботи з нейронними мережами та їх компонентами [11].

Апаратне забезпечення - це розробка архітектури нейронної мережі та компонентів, орієнтованих на реалізацію VLSI та режим реального часу.

Програмне забезпечення - це програма для реалізації моделей нейронних мереж та їх компонентів.

Для реалізації цього підходу використовується метод декомпозиції, який передбачає розподіл фондів на окремі компоненти, які при необхідності припиняють нейронні мережі в конкретному реальному часі. На кожному рівні ієрархії проблеми вирішуються з відповідною складністю, які характеризуються як інформаційною одиницею, так і алгоритмами обробки.

Таблиця 2.1 - Рівні та види розробок нейромереж реального часу

Ієрархічний рівень	Види забезпечення та виконувані розробки		
	Алгоритмічне	Апаратне	Програмне
1-й	Методи, алгоритми навчання та функціонування нейромережі	Архітектура нейромережі – кількість шарів нейромережі, число нейронів у шарах і способи зв’язків між нейронами та шарами	Структура програмних засобів для навчання та функціонування нейромережі
2-й	Алгоритми функціонування шарів ШНМ, нейроелементів, паралельної пам'яті та перетворювачів даних	Структури шарів ШНМ, нейроелементів, паралельної пам'яті та перетворювачів даних	Програми реалізації шарів ШНМ, нейроелементів, паралельної пам'яті та перетворювачів даних
3-й	Алгоритми реалізації багатоопераднихнейроопераційпопередньої та процесорної обробки даних	Структури пристроїв для реалізації багатоопераднихнейроопераційпопередньої та процесорної обробки даних	Програми реалізацій багатоопераднихнейроопераційпопередньої та процесорної обробки даних
4-й	Алгоритми обчислення дво- і одноопераднихнейроопераційй	Структури операційних пристроїв для обчислення дво- і одноопераднихнейроопераційй	Програми реалізацій дво- і одноопераднихнейроопераційй

Залежно від складності виконуваної роботи процес розвитку нейронної мережі можна розділити на чотири ієрархічні рівні. Збільшення кількості рівнів ієрархії дорівнює збільшенню деталей розробки алгоритмів, обладнання та

програмного забезпечення. На найвищому рівні ієрархії інформаційні одиниці, алгоритми, програми та обладнання розташовані за одиницями інформації, а композиції алгоритмів, програмного та апаратного забезпечення знаходяться на нижчих рівнях ієрархії (табл. 2.1). Метод послідовного розподілу, що використовується при розробці нейронних мереж, відображає процес розробки зверху вниз.

Перший ієрархічний рівень розробляє архітектуру (визначає тип нейронної мережі, кількість шарів у нейронних мережах, кількість нейронів у кожному шарі, методи зв'язку між нейронами), методи навчання та функціонування, алгоритми та структуру програмного забезпечення нейронної мережі.

На другому рівні ієрархії розробляються методи та алгоритми для нейронних мереж, нейронних елементів, функцій паралельної пам'яті та передачі даних, структури пристрою та програм.

Третій ієрархічний рівень включає відділення для проведення кількох оперативних передопераційних та нейрохірургічних процесів обробки даних. Паралельні алгоритми, структури VLSI та додатки призначені для реалізації елементів третього рівня.

Четвертий рівень ієрархії включає одиниці обчислення двох операцій (додавання, множення, ділення) та одного операнда (вилучення квадратного кореня, функції передачі). Елементи четвертого рівня базуються на елементарних арифметичних операціях.

Ієрархічну структуру компонента систем нейродоната можна описати такими термінами:

$$C_{HC}^{1} = \bigcup_{i=1}^{n} C_{HC}^{2i} \bigcup_{j=1}^{m} C_{HC}^{3j} \bigcup_{p=1}^{h} C_{HC}^{4p}, \qquad (2.1)$$

де C_{HC}^{2i}, C_{HC}^{3j}, C_{HC}^{4p} – засоби відповідно другого, третього і четвертого ієрархічних рівнів;

n – кількість типів компонентів другого рівня;

m - кількість типів компонентів третього рівня;

h - кількість типів компонентів четвертого рівня.

2.2 Компонентно-орієнтована технологія

На сучасному етапі розвитку інформаційних технологій розвиток BSUTP був скорочений шляхом вирішення проблеми інтеграції існуючого апаратного та програмного забезпечення, оскільки розробка та виробництво нових вимагає значних ресурсів і часу. При вирішенні проблеми інтеграції потрібно враховувати багато факторів, а саме: інформацію про готові апаратні та програмні компоненти, їх технічні характеристики, відповідність інтерфейсів стандартам, доступність тощо.

Дизайн сучасних BSUTP базується на підході системної інтеграції, який охоплює всі рівні інтеграції процесів, споруд, обладнання та інфраструктури з урахуванням ефективності їх використання та вимог конкретних програм.

Пропонується системна інтеграція на основі компонентно-орієнтованої технології, яка передбачає розподіл процесу розробки на ієрархічні рівні та типи програм (алгоритм, обладнання та програмне забезпечення).

Для реалізації цієї технології використовується метод деградації, який передбачає поділ багаторівневих систем управління та апаратних та програмних компонентів. На кожному рівні ієрархії апаратні та програмні компоненти використовуються для вирішення проблем відповідної складності, що характеризуються як інформаційними одиницями, так і алгоритмами обробки.

Збільшення кількості рівнів ієрархії прирівнюється до збільшення деталізації алгоритмічних, апаратних та програмних компонентів. На вищому рівні ієрархії інформаційні одиниці, алгоритми, програмне та апаратне забезпечення розташовані за одиницями інформації, а на нижчому рівні ієрархії алгоритмів, програмних та апаратних збірок. Метод послідовного розбиття, що використовується при розробці BSUTP, описує процес розробки "зверху вниз". Процес синтезу BSUTP із використанням компонентно-орієнтованої технології

описується з використанням таких термінів:

$$C^{1}_{\text{БСУТП}} = \bigcup_{i=1}^{n} C^{2i}_{\text{БСУТП}} \bigcup_{j=1}^{m} C^{3j}_{\text{БСУТП}}, \quad (2.2)$$

де,

$C^{2i}_{\text{БСУТП}}$, $C^{3j}_{\text{БСУТП}}$ – відповідно апаратно-програмні компоненти другого та третього рівні управління;

n – кількість апаратно-програмних компонентів другого рівня;

m – кількість апаратно-програмних компонентів третього рівня.

Як горизонтальна, так і вертикальна інтеграція використовуються при проектуванні багаторівневих рівнів управління. Горизонтальна інтеграція передбачає інтеграцію багаторівневих систем управління в один рівень в ієрархії, тоді як вертикальна інтеграція передбачає управління сусідньою ієрархією ієрархії.

Використання ANN у галузях, де розташоване обладнання, тобто таке, що транспортується, перевозиться, літає та плаває, накладає серйозні обмеження щодо їх масових та габаритних властивостей. У той же час SNM має суворі вимоги до енергоспоживання, що впливає на розмір джерел енергії та тепловіддачу. Для завдань даного класу необхідність задоволення вимог щодо масогабаритних властивостей, енергоспоживання, витратних сил повинна бути дуже суворою, коли мова йде про вибір параметрів, що визначають витрати обладнання на їх створення. Це проявляється зменшенням довжини бітової мережі для представлення фіксованих командних операндів, зменшенням списку використовуваних команд та кількості рядків шини адреси, які визначають функції пам'яті.

Крім того, ANN має високі вимоги як до виживання, надійності, так і до тестування ефективності, швидкої локалізації та усунення несправностей. Проблема високої виживаності ПІК виникає, коли його застосовують особливо в системах управління відповідальних об'єктів, які знаходяться на великій відстані від людей, або з об'єктами, що мають великий зовнішній вплив. Для забезпечення достатньо високого виживання АНН його структурні частини повинні бути

замінені. Цю проблему можна вирішити лише за рахунок однорідності компонентів ANN та одноманітності архітектури.

Зменшення характеристик маси та розміру, енергоспоживання, підвищення надійності ANN та забезпечення режиму реального часу можна досягти, запровадивши їх VLSI. Впроваджуючи VLSI ANN, вони повинні забезпечити високу ефективність обладнання, враховуючи кількість інтерфейсних королів, однорідність конструкції, кількість з'єднань та розташування, а також пов'язувати експлуатацію із витратами на обладнання та оцінювати елементи блоку для роботи [2]. Кількісне значення ККД обладнання визначається наступним чином:

$$E = \frac{R}{t_o(k_1 \sum_{i=1}^{s} W_{\Phi У_i} d_i + k_2 Q + k_3 Y)}, \qquad (2.3)$$

де R - складність алгоритму, яка визначається кількістю елементарних арифметичних операцій, необхідних для реалізації;

From - час реалізації алгоритмів підготовки та роботи нейронної мережі; WFUi - витрати на обладнання для реалізації функціонального блоку I;

di - кількість функціональних вузлів типу i;

k1 - коефіцієнт однорідності k1 = f (s);

s - кількість типів функціональних вузлів;

Q - загальна кількість з'єднань;

k2 - коефіцієнт k2 = f () для врахування регулярності сполук; - Відстань просторового спілкування; Y - кількість виводів інтерфейсу;

k3 - коефіцієнт, який повинен враховувати кількість контактів у комунікаційному інтерфейсі k3 = f (Y).

Завданням проектування SNM, орієнтованих на реалізацію VLSI в режимі реального часу, з високою ефективністю використання обладнання, є зменшення апаратних витрат, збільшення кількості інтерфейсних штифтів, підвищення однорідності структури та підвищення регулярності з'єднань.

Висока ефективність досягається при використанні обладнання для узгодження інтенсивності даних Pd = knFd, де k - кількість каналів даних; n -

бітовий розмір каналів даних; Fd - частота даних, з інтенсивністю обчислень (обчислювальною потужністю), яка визначається наступним чином [1]:

$$D_к = \frac{hn_m}{T_к}, \tag{2.4}$$

Де h - кількість шляхів обробки, nm - кількість шляхів передачі даних на етапах конвеєра; Tk - конвеєрна стрічка. Кількість процесних шляхів дорівнює кількості нейронів у кожному шарі нейронних мереж.

Незважаючи на досягнення в автоматизації ANN із використанням сучасної бази даних елементів, моделювання та обслуговування є одним із найбільш трудомістких етапів розробки ANN. Для корекції розроблена ANN повинна бути оснащена функціями контролю, спостережливості та передбачуваності. Керований - функція ANN, де поведінка контролюється, тобто можна запустити, зупинити, продовжити роботу за будь-якою адресою. Спостереження - функція ANN, яка дозволяє контролювати поведінку та зміни внутрішніх умов. Передбачуваною є властивість ANN, яка дозволяє йому визначати стан, з якого можна визначити всі наступні стани.

2.3 Рівень контролю та управління технологічним процесом

На цьому рівні їх можна використовувати як окремі комп’ютери, системи на базі SoC та програмовані логічні контролери Mitsubishi Melsec FX3U.

Програмовані контролери Mitsubishi FX3U є найпотужнішими та найпотужнішими в контролерах серії MELSECFX (FX1S, FX3S, FX1N, FX3G). Архітектура контролера - це дві шини, що збільшує його можливості. Програмований контролер Mitsubishi з'єднує як модулі розширення попереднього покоління, так і модулі нового покоління FX3U. Коли модулі FX3U підключені, контролер автоматично перемикається на комутаційну шину у високошвидкісному режимі, і обмін даними відбувається на більш високій швидкості. Модулі FX0N,

FX2N працюють з контролером на звичайній швидкості. Таким чином, використання модулів розширення збільшує кількість входів / виходів до 256 (пряма адреса) та за допомогою децентралізованих станцій вводу-виводу до 384.

Крім того, програмований контролер Mitsubishi підключає високошвидкісні адаптерні модулі FX3U-XXX-ADP, що покращує функції контролера при роботі з аналоговими сигналами та збільшує кількість додаткових комунікаційних інтерфейсів (RS232 / 422/425).

За останнє десятиліття з’явилося кілька мов програмування PLC, для яких був розроблений стандарт IEC 61131. Цей стандарт регулює зміни мови програмування для систем управління процесами. Цей стандарт охоплює вимоги до обладнання, встановлення, тестування, документації, зв'язку та програмування ПЛК. Стандарт визначає 5 мов програмування, які поділяються на тексти та графіки. Мови тексту PLC включають:

Список інструкцій (IL) - мова викладання;

1. Структурований текст (ST) - це мова структурованого тексту.

Мова IL - це типова установка з елементом та позначками переходу. Набір інструкцій стандартизований, він не залежить від конкретної цільової платформи. Ця мова дозволяє працювати з усіма іншими типами даних, функціями викликів та функціональними блоками, реалізованими іншими мовами. Алгоритми будь-якої складності реалізовані за допомогою ІЛ.

ST - мова високого рівня, яка синтаксично нагадує Паскаль. Замість мовних процедур Pascal ST використовує компоненти програми IEC.

Мова графіки в PLC включає:

F Діаграма послідовних функцій (SFC) - мова послідовних діаграм функцій;

2. Функціональні блок-схеми (FBD) - мова функціональних блок-схем;

3. Східні схеми (LD) - мова для ланцюгів контактів реле;

SFC - це висококласний графічний інструмент, де графічна діаграма складається з кроків і переходів між ними. Дозвіл на перехід визначається умовно, а крок пов'язаний з певними діями.

Мова FBD зосереджена на описі принципових схем електронного пристрою на мікросхемах. Лідери FBD використовуються для передачі сигналів будь-якого

типу (логічного, аналогового, часового та ін.). Сигнали з виходу блоків можуть надходити на виходи інших блоків або безпосередньо на виходи ПЛК. "Блоки", представлені на схемі як "чорні ящики", можуть виконувати будь-яку функцію. Мова FBD забезпечує опис взаємозв'язку між входом і виходом на діаграмі. Якщо алгоритм добре описаний з точки зору сигналу, його подання FBD завжди видніше, ніж у текстовій мові.

Мова LD зосереджена на реалізації структури електричних ланцюгів. Графічно діаграма LD представлена у вигляді двох вертикальних шин живлення. Серед них є підключення ланцюгів від роз’ємів. Навантаження на кожну схему - реле. Кожне реле має роз'єми, які можна використовувати в інших ланцюгах. Послідовне (І), паралельне (або) контактне з'єднання та інверсія (ні) складають логічну основу. Як результат, мова LD ідеально підходить не тільки для побудови релейних блоків, а й для реалізації програмного забезпечення логічної схеми. Завдяки можливості використовувати LD-функції та функціональні блоки, створені іншими мовами, обсяг збільшується.

IEC 61131 Мови базується на найпопулярніших мовах програмування для сучасних контролерів. Програмне забезпечення, написане для сучасних контролерів, можна перевести на середовища IEC 61131-3 за мінімальних витрат. IEC 61131 характеризується можливістю створювати бібліотеки незалежно від обладнання, контролерів, фільтрів, управління дисками, модулів із розмитою логікою тощо.

Середовище програмування CoDeSys (система розробки контролерів) було розроблено для програмування контролерів на мові IEC 61131-3, редактори та засоби пошуку несправностей яких базуються на принципах популярного професійного середовища програмування (VisualC ++ та ін.). Особливістю середовища програмування CoDeSys є те, що воно не підключене до певної технічної платформи. Середовище програмування змінилося для різних ПЛК і адаптує програму до низькорівневих ресурсів - розподілу пам'яті, інтерфейсу зв'язку та інтерфейсу вводу-виводу. CoDeSys пропонує: безпосереднє виготовлення машинного коду; Повна реалізація мов IEC 61131-3; Інтелектуальні мовні редактори, які виправляють помилки початківців; Вбудований емулятор

управління, який забезпечує роботу проекту без додаткового обладнання; Вбудовані елементи візуалізації дозволяють створювати об'єктну модель управління та корекцію проекту без виробництва інструментів імітації; Використання готових бібліотек та сервісних функцій.

Рівень оператора та формування управлінських рішень. На цьому рівні контроль оператора представлений автоматизованим робочим місцем оператора. Методики, що застосовуватимуться на цьому рівні, можуть визначатися інтенсивністю даних, складністю алгоритмів обробки та вимогами до надійності системи управління. Операційні станції на платформах RISC або Intel та промислові комп'ютери, такі як Mitsubishi Electric, BECKHOFF, Eaton, AXIOMTEK, можуть використовуватися з обладнанням, яке відповідає умовам експлуатації, з підвищеними вимогами до міцності та надійності. Сховища даних; Обробка даних; Обробка відеопотоків, розпізнавання зображень і сцен в системах технічного зору; Синхронізація розподілених підсистем; Візуалізація та відображення технологічного процесу; Формування управлінських рішень.

Системи SCADA використовуються для вирішення проблем, основною функцією яких є створення інтерфейсу оператора та збір даних про технологічний процес. Інструменти, що являють собою поєднання апаратних засобів, каналів зв'язку та алгоритмічних програм, використовуються для управління та управління технологічним процесом.

Функціями завдань, що вирішуються на рівні контролю оператора та прийняття управлінських рішень, є:

Велика кількість та різноманітність даних;

1. Невідповідність та неповні дані;

Послідовність вхідних даних та висока інтенсивність;

2. Велика кількість обчислень, з перевагою обчислювальних операцій логічно, в системах відеозору при обробці відеопотоку, розпізнавання зображень і сцен;

Постійне ускладнення алгоритмів обробки Al та підвищення вимог до точності результатів;

Можливість паралельної обробки даних у часі та просторі.

Враховуючи широкий розвиток та впровадження бездротового зв'язку в останні роки та зростаючу інтеграцію різних систем в Інтернеті, завдання полягає у створенні універсальних платформ, що підтримують функції гарантованого обміну даними з периферійними пристроями, та у створенні універсального безпечного сховища даних. Сили в будь-якій точці світу. Таким чином, реалізація BSUTP ставить ще одне завдання - формування інтегрованої системи для обміну та накопичення інформації з дисперсійних систем.

Водночас метою є забезпечення з’єднання дротових та бездротових каналів, взаємодія з Інтернет-датчиками, мікроконтролерними системами та активаторами в BSUTP. Однією з можливих технологій є пристрій відкритого обміну даними Hive Platform. Ця технологія є гнучкою, масштабною та простою у використанні та забезпечує обмін даними між технічними компонентами на основі M2M. Використання технології Device Hive забезпечує створення комунікаційного середовища, програмного управління та використання багатоплатформенних бібліотек для розвитку дистанційного управління та моніторингу, телеметрії, дистанційного управління та моніторингу. Технологію Device Hive можна використовувати з використанням широкого кола технологій, таких як вбудовані Linux, Python, бібліотека C ++, протокол JSON або мікроконтролери AVR, Microchip. Однією з функцій пристрою є робота з Hive, спочатку організація доступу до мережі, а потім програмування конкретних програм, що скорочує час на розробку методів передачі даних.

2.4 Висновки до розділу

Створення сучасних багаторівневих систем управління процесами базується на підході до системної інтеграції, що включає всі рівні інтеграції процесів, об'єктів, об'єктів та інфраструктури, з метою ефективності їх використання та вимог конкретних додатків.

Пропонується системна інтеграція на основі компонентно-орієнтованої технології, яка забезпечує розподіл процесу розробки на ієрархічні рівні та типи програмного забезпечення: алгоритмічне, технічне та програмне.

Розробка високошвидкісних, малопотужних, високошвидкісних апаратних компонентів для багаторівневих рівнів управління базується на мікроконтролерах із фіксованою точкою, невеликих мережах, усічених інтерфейсах та скорочених системах команд.

Для програмування мікроконтролерів рекомендується використовувати середовище програмування CoDeSys (система розробки контролерів), редактори та програми налагодження на основі принципів професійного середовища програмування (VisualC ++ та ін.).

У сучасних багаторівневих системах управління процесами рекомендується використовувати пристрій технології Hive для поєднання датчиків, систем мікроконтролера та дротових та бездротових каналів для драйверів та підключення до Інтернету.

3 СИНТЕЗ СТРУКТУРИ БСУТП НА ОСНОВІ КОМПОНЕНТНО-ОРІЄНТОВАНОЇ ТЕХНОЛОГІЇ

3.1 Апаратні засоби БСУТП

Більшість апаратних засобів, що використовуються в BSUTP, вбудовані в комп'ютерні системи, і концепція полягає в тому, що такі системи працюють безпосередньо шляхом вбудовування пристроїв, якими вони керують. Такі системи характеризуються оптимізацією архітектури для виконання певних функцій у режимі реального часу. Основними елементами реалізації базових систем є програмовані логічні контролери (PLC). Архітектура сучасних ПЛК зосереджена на обробці сигналів, керуванні пристроями та керуванні виконавчим механізмом. Основним завданням ПЛК є збір даних про генераторні вимикачі, датчики та, залежно від їх стану, генерувати сигнали керування рухом. Сучасний ПЛК має великі можливості для зміни програмування та існуючого програмного забезпечення. PLC, використовуючи апаратно-програмний компонент як модуль, розроблений для BSUTP, який інтегрує реле, таймери, лічильники та адаптери. Можливість перепрограмувати такі апаратні та програмні компоненти збільшила гнучкість та забезпечила швидку адаптацію BSUTP до змін. Апаратні та програмні компоненти на основі ПЛК мають можливість віддаленого налаштування та управління, що значно розширило сферу їх використання [9].

На рівні рішень управління та управління операторами промислові комп'ютери, такі як Mitsubishi Electric, BECKHOFF, Eaton, AXIOMTEK, використовуються для побудови апаратних та програмних компонентів, які відповідають промисловим умовам експлуатації з підвищеними вимогами до міцності та надійності. Апаратні та програмні компоненти, що використовуються для збору даних, управління активатором та управління, а також технологічні процеси реалізуються за допомогою PLC сімейства STM, Modicon, LOGIC, MAXIM. Крім того, програмовані реле ZelioLogic, ZEN та Easy широко використовуються як апаратні компоненти в BSUTP, повністю замінюючи старі реле з точки зору функціональності та розмірів.

Мова програмування для систем мікроконтролера. За останнє десятиліття з’явилося кілька мов програмування PLC, для яких був розроблений стандарт IEC 61131. Цей стандарт регулює зміни мови програмування для систем управління процесами. Цей стандарт охоплює вимоги до обладнання, встановлення, тестування, документації, зв'язку та програмування ПЛК. Стандарт визначає 5 мов програмування, які поділяються на тексти та графіки. Мови тексту PLC включають:

1. Instruction List (IL) - мова інструкцій;
2. Structured Text (ST) - мова структурованого тексту.

Мова IL - це типова колекція з батареєю, яка рухається до позначок. Набір інструкцій стандартизований, він не залежить від конкретної цільової платформи. Ця мова дозволяє працювати з усіма типами даних, функціями викликів та функціональними блоками, реалізованими іншими мовами. ІЛ будь-якої складності реалізується за допомогою алгоритмів.

ST - це мова високого рівня, яка синтаксично схожа на Pascal. Замість мовних процедур Паскаля ST використовує компоненти програми IEC.До графічних мов ПЛК відносяться:

1. Sequential Function Chart (SFC) - мова послідовних функціональних діаграм;
2. Function Block Diagram (FBD) - мова функціонально-блокових діаграм;
3. Ladder Diagrams (LD) - мова релейно-контактних схем;

SFC - це висококласний графічний інструмент, де графічна діаграма складається з кроків і переходів між ними. Дозвіл на перехід визначається умовно, а крок пов'язаний з певними діями.

Мова FBD зосереджена на описі принципових схем електронного пристрою на мікросхемах. Лідери FBD використовуються для передачі сигналів будь-якого типу (логічного, аналогового, часового та ін.). Сигнали з виходу блоків можуть надходити на виходи інших блоків або безпосередньо на виходи ПЛК. "Блоки", представлені на схемі як "чорні ящики", можуть виконувати будь-яку функцію. Мова FBD забезпечує опис взаємозв'язку між входом і виходом на діаграмі. Якщо алгоритм добре описаний з точки зору сигналу, його подання FBD завжди видніше, ніж у текстовій мові.

Мова LD зосереджена на реалізації структури електричних ланцюгів. Графічно діаграма LD представлена у вигляді двох вертикальних шин живлення. Серед них схеми створюються із з'єднувальними роз'ємами. Навантаження на кожну схему - реле. Кожне реле має роз'єми, які можна використовувати в інших ланцюгах. Послідовне (І), паралельне (або) контактне з'єднання та інверсія (ні) складають логічну основу. Як результат, мова LD ідеально підходить не тільки для створення пристроїв ретрансляції, але і для реалізації програмного забезпечення комбінованої логічної схеми. Завдяки можливості використовувати LD-функції та функціональні блоки, створені іншими мовами, сфера застосування збільшується.

IEC 61131 Мови базується на найпопулярніших мовах програмування для сучасних контролерів. Програмне забезпечення, написане для сучасних контролерів, можна перевести на середовища IEC 61131-3 за мінімальних витрат. IEC 61131 має можливість створювати незалежні від апаратного забезпечення бібліотеки для контролерів, фільтрів, управління дисками, модулів з незрозумілою логікою тощо.

Середовище програмування CoDeSys (система розробки контролерів) призначене для програмування контролерів в IEC 61131-3, редактори та засоби пошуку несправностей яких базуються на принципах популярних професійних середовищ програмування (VisualC ++ та ін.). Особливістю середовища програмування CoDeSys є те, що воно не підключене до певної технічної платформи. Середовище програмування змінилося для різних ПЛК, пристосувавши програму до низькорівневих ресурсів - розподілу пам'яті, інтерфейсу зв'язку та інтерфейсу вводу-виводу. CoDeSys забезпечує: генерацію кодів безпосередньо в машині; Повна реалізація мов IEC 61131-3; Інтелектуальні мовні редактори, які виправляють помилки початківців; Вбудований емулятор управління, який вирішує проектні проблеми без додаткового обладнання; Вбудовані елементи візуалізації забезпечують створення об'єктної моделі управління та корекцію проекту без виробництва інструментів імітації; Використання священних бібліотек та службових функцій.

Технологія вулика пристрою. Технологія Device Hive використовується для підключення дротових та бездротових каналів, а також для підключення датчиків,

систем мікроконтролера та виконавчих механізмів до Інтернету. Ця технологія є гнучкою, масштабною та простою, що забезпечує обмін даними між технічними компонентами BSUTP за принципом M2M. Використання технології Device Hive забезпечує створення комунікаційного середовища, програмного управління та використання багатоплатформенних бібліотек для розвитку дистанційного управління та моніторингу, телеметрії, дистанційного управління та моніторингу. Технологію Device Hive можна використовувати з використанням широкого кола технологій, таких як вбудовані Linux, Python, бібліотека C ++, протокол JSON або мікроконтролери AVR, Microchip. Однією з функцій пристрою є робота з Hive, спочатку організація доступу до мережі, а потім програмування конкретних програм, що скорочує час на розробку методів передачі даних.

Мікроконтролерні системи. Програмовані контролери Mitsubishi FX3U - це найпотужніші контролери MELSECFX у серії (FX1S, FX3S, FX1N, FX3G). Архітектура контролера - це дві шини, що збільшує його можливості. Програмований контролер Mitsubishi з'єднує як модулі розширення попереднього покоління, так і модулі нового покоління FX3U. Коли модулі FX3U підключені, контролер автоматично переводить комутаційну шину у високошвидкісний режим, і обмін даними відбувається на більш високій швидкості. Модулі FX0N, FX2N працюють з контролером на звичайній швидкості. Таким чином, використання модулів розширення збільшує кількість входів / виходів до 256 (пряма адреса), а через децентралізовані станції вводу-виводу до 384.

Крім того, програмований контролер Mitsubishi забезпечує з'єднання FX3U-XXX-ADP з високошвидкісними адаптерними модулями, що розширює можливості контролера при роботі з аналоговими сигналами та збільшує кількість додаткових інтерфейсів зв'язку (RS232 / 422/485) [11].

Одноразовий мікрокомп’ютер Raspberry Pi - це перспективна платформа для систем автоматизації, що характеризується низькою вартістю, операційною системою та відкритою архітектурою. Мікрокомп'ютер Raspberry Pi побудований на системі Broadcom BCM2835 (SoC), яка включає в себе процесор ARM на 700 МГц, графічний процесор VideoCore IV і 512 або 256 МБ пам'яті, використовуючи SD-карту як додаткову пам'ять. [12] Цей мікрокомп’ютер доступний у двох версіях:

молодша (A) (256 Мб оперативної пам’яті, один порт USB) та старіша (B) (з Ethernet, 512 Мб оперативної пам'яті, два порти USB)

Raspberry Pi має дистрибутив для Linux, ОС Raspbian (на основі дистрибутиву Debian), а також ряд програм Pi-Store, як платних, так і безкоштовних. Raspberry Pi має власні порти GPIO, які можна використовувати для різноманітних функцій.

Сімейство мікроконтролерів STM8 - це відносно нове сімейство мікроконтролерів від ST Microelectronics, яке містить майже все, що можна очікувати від 8-бітного програмованого контролера. Переваги контролерів STM8 включають, перш за все, низьку вартість не тільки самого контролера, але й карток налагодження, програмістів та виправлень помилок. Вбудований завантажувач дозволяє завантажувати програми з UART, SPI, CAN або I2C. Надається безкоштовна бібліотека зовнішніх драйверів. STM розробила бібліотеку мікропрограм STM8x, яка є основою для мікроконтролерів. Загалом, вимоги до електромагнітної сумісності та надійності зростають у всій серії STM8, що також є привабливим для їх використання [14].

До недоліків можна віднести відсутність контролерів з USB і у випадках низької продуктивності (тип SO8).

Наразі існує три сімейства STM8:

1. STM8S - "standard" контролери загального застосування, зазвичай 10 бітна аналогова периферія, середнє за сучасними стандартами енергоспоживання та діапазон живлення - 2.95 - 5.5В.

2. STM8L - "low-power" контролери з низьким споживанням, 12 бітова аналогова периферія, покращена електромагнітна сумісність і діапазон живлення - 1.8-3.6В,. У порівнянні зі стандартними контролерами, в цьому сімействі додається деяка кількість периферії, зокрема, DMA.

3. STM8A - "automobile" – сімейство зосереджено на безпеці і надійності, витримують більше навантажень на ніжки, діапазон живлення - 2.95 - 5.5В і працюють при 145 градусах [15].

STM8 має 8-бітове ядро CISC і містить 6 регістрів: A - Акумулятор - однобайтовий акумулятор, що містить результат арифметико-логічної команди; X, Y - індекс - двобайтові індексні регістри, які можуть містити відносну адресу

комірок пам'яті, з якими можна виконувати операції; Лічильник програм на ПК - трибайтовий лічильник програм, що містить абсолютну адресу комірки пам'яті, що містить команду, яка тепер буде виконана; SP - покажчик стека - двобайтовий покажчик поверх стека, що має власну пам’ять; CC - коди умов - однобайтовий реєстр прапорів (прапорів), що містить бітові властивості арифметично-логічного результату команди, який можна використовувати для гілок програми [8].

Мікроконтролер STM8 має 32 вектори переривань, кожен з яких становить 4 байти (великі байти містять зарезервоване значення 82 години) перед запуском відповідної підпрограми події, що спричиняє затримку. Закрийте диспетчер припинення за допомогою команди IRET.

Тактова частота мікроконтролера STM8 може змінюватися під час роботи, на відміну від AVR, коли годинник жорсткий. У додатках з низьким енергоспоживанням це іноді використовується, і коли генератор виходить з ладу, контролер перемикається на RC і перемикається в режим зупинки, в результаті чого система не залишається поза контролем.

Пам'ять мікроконтролера умовно розділена на 64K частин, а кожен розділ - 256 сторінок з 256 байтами. Мікроконтролер STM8 дозволяє застосовувати наступні методи: прямий, індексний, прямий, стековий, непрямий, біти та відносний. Залежно від довжини операнда режим адреси поділяється на короткий - 1 байт, довгий - 2 байти та розширений - 3 байти.

На практиці, перш за все, слід звернути увагу на розташування єдиної 24-бітової адреси, що містить флеш-пам’ять, оперативну пам’ять, Eeprom та периферію. Це значно спрощує написання коду, наприклад, функції оперативної пам’яті та флеш-пам’яті не вимагають написання кількох копій. Різні коди стеків та бібліотек легко перетворюються, оскільки вони в першу чергу розроблені для архітектури фон Неймана (маючи на увазі адресний простір). Автобуси є окремими для доступу до різних типів пам'яті, що вказує на існування архітектури Гарварда.

Методи навчання в нейронних мережах поділяються на три основні типи:

– освіта вчителів (контрольоване навчання);

– навчання без вчителя (неконтрольоване навчання);

– покращена підготовка.

Ці три типи навчання класифікуються на основі наступного:

– присутність або відсутність викладача;

– інформація, що надається навчальній мережі.

Відповідно до застосовуваних правил навчання цей вид навчання поділяється на такі категорії:

– навчання Хебб;

– тренування на основі градієнтного спуску;

– змагальні тренування;

– стохастичне навчання.

На рисунку 3.1 вказані методи показані у ієрархічному вигляді.

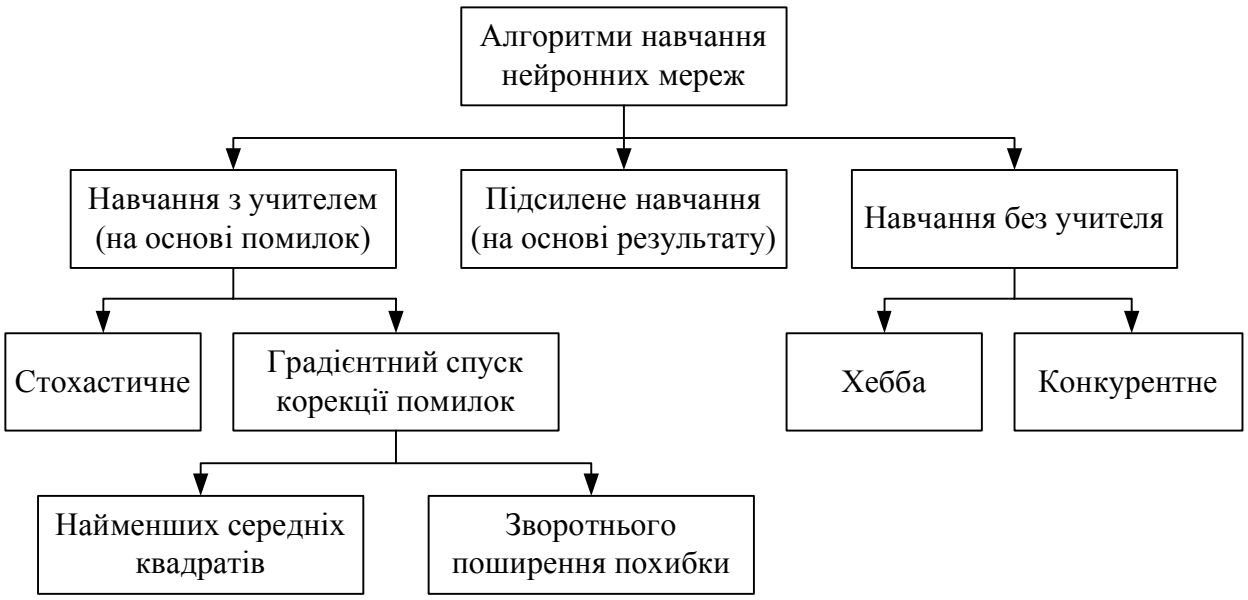

Рисунок 3.1 - Класифікація алгоритмів навчання нейронних мереж.

Навчання з викладачем:

– у процесі навчання є вчитель, який показує очікуваний результат в кінці мережі;

– усі вхідні дані використовуються для навчання мережі;

– процес навчання заснований на порівнянні результату, отриманого нейронними мережами при розрахунку, і правильного результату, одночасно з розрахунком помилки;

– розрахована похибка використовується для зміни налаштувань мережі, що покращить якість нейронних мереж.

Навчання без вчителя:

– у процесі навчання немає вчителя;

– очікуваний або бажаний результат для нейронних мереж невідомий;

– система вивчає себе, відокремлює структурні особливості даних і відповідно змінює структуру.

Покращене навчання:

– учитель присутній, але не дає очікуваних чи бажаних результатів, лише вказує правильно або неправильно розрахований результат;

– надана інформація допомагає нейромережам у процесі навчання;

– винагорода надається за правильно розрахований результат, а штраф за неправильний результат.

Слід зазначити, що інтенсивне тренування не дуже популярне. Найчастіше використовується навчання з викладачем або тренування без вчителя.

Навчіться. У 1949 р. Фізіолог Д. Хеб написав книгу «Організація свідомості». У цій книзі він намагався пояснити, як можна вивчати нервові клітини в мозку людини. Пізніше його теорію назвали "Вченням Євреїв". Правила вивчення мережі Хопфілда базувалися на Д. Хеббі.

Оригінальний дизайн Розенблатта складався з фотоелементів, які, залежно від використовуваного на них сигналу, виробляли сигнал від логічної одиниці або логічного нуля. Сигнали з фотоелементів направляли на зважене додавання (елементний процесор, штучний нейрон) з функцією активації бар’єру. Нейрон також подавав сигнал про логічний нуль, або логічну одиницю. Можна замість {0.1. Сигнал використати сигнал {-1,1 дюйма .

Метою навчання персептрону було відображення блоку кодування при відображенні закодованого зображення на фотографії, якщо асоційоване зображення належить до заздалегідь визначеного класу і в іншому випадку дорівнює нулю. Логіка тренінгу така: якщо сигнал персептрону є дещо правильним, коригувати нічого не потрібно, якщо ні - регулюється вага чоппера. Правила регулювання ваги, запропоновані Хеббом, мають такі результати:

Перше правило Хебба - якщо сигнал персептрона неправильний і дорівнює нулю, необхідно збільшити вагу входів, на яких використовувався пристрій.

Друге правило Хебба - якщо сигнал персептрона неправильний і дорівнює одиниці, потрібно зменшити вхідну вагу, на якій використовувався пристрій.

Правила послідовно застосовуються до всіх зображень, на яких проводиться тренінг. На питання про те, чи стане персептрон в сталому стані, коли він правильно класифікує всі вхідні зображення, відповідає теорема про збіжність персептрона.

Математично ці правила можна описати наступним чином. Припустимо, що класи позначаються цифрами 0 і 1:

$$a(x) = \left[\langle w, x\rangle > 0\right] \tag{3.1}$$

де ω – вектор синаптичних ваг,

$x_i = (x_i,\ldots,x_i^n)$ об’єкт з навчальної вибірки прецентдентів $X^L = \{x_1,\ldots, x_n\}$, для якого відомий правильну відповідь y_i.

Перцептрон викладається згідно з правилом Гебсів. Відправляємо один об’єкт до входу. Якщо вихідний сигнал персептрона відповідає правильній реакції, ніяких дій робити не потрібно. У разі помилки необхідно навчити персептрон правильно розв'язувати цей приклад.

Помилки можуть бути двох типів. Подумайте про кожного з них.

Перший тип помилки: персептрон а (xi) = 0 Правильна відповідь, коли вихідний результат уі = 1.

Щоб перцептрон реагував правильно, скалярний продукт повинен бути більшим. Оскільки змінні мають значення 0 або 1, збільшення суми може бути досягнуто за рахунок збільшення ваг. Однак немає сенсу збільшувати вагу всіх змінних на нуль. Ми збільшуємо вагу лише на вагу, рівну 1. Щоб зафіксувати один ом сигналу, потрібно виконати ту саму процедуру на всіх інших шарах.

Це Євр. Перше правило.

Другий тип помилки: а (xi) = 1, уі = 0.

Для того, щоб звести скалярний добуток до правильної частини, необхідно зменшити вагу зв’язків у змінних, рівних 1. Також необхідно виконати цю процедуру для всіх активних нейронів у передніх шарах.

3.2 Вимоги до компонентів багаторівневої системи управління

Однією з найпоширеніших вимог до апаратних та програмних компонентів у багаторівневих системах управління є забезпечення високої продуктивності.

Подібна проблема зазвичай виникає при використанні компонентів для вирішення проблем у режимі реального часу, що накладає певні обмеження на процес обробки інформації [8]. Для забезпечення управління базами даних у реальному часі продуктивність апаратних та програмних компонентів (тобто кількість операцій в секунду) повинна бути такою:

$$П \geq \frac{R \beta F_d}{n_d}, \tag{3.2}$$

де R – складність алгоритму розв’язання задачі (кількість операцій);

β >1 – коефіцієнт врахування особливостей засобів реалізації алгоритму;

F_d – частота надходження даних (кількість разів виконання алгоритму за секунду); n_d – кількість компонент розпаралелення.

Використання апаратних та програмних компонентів на нижчому рівні управління, де пристрій розташований із датчиками та двигунами, створює серйозні обмеження щодо їх масових та розмірних характеристик. У той же час до таких компонентів пред'являються жорсткі вимоги щодо енергоспоживання, що впливає на розмір джерел живлення та тепловіддачу. Таким вимогам можна задовольнити, зменшивши довжину бітової мережі, кому, зафіксовану для подання операндів, зменшивши список команд та кількість шинних рядків адрес, які визначають, скільки пам'яті доступне користувачеві.

Крім того, апаратні та програмні компоненти BSUTP повинні забезпечувати високу економію, надійність, тестування продуктивності та швидке виявлення несправностей. Проблема високої виживаності апаратних та програмних компонентів виникає, коли вони використовують особливо важливі елементи в системах управління, розташованих занадто далеко від людей або об'єктів із великим зовнішнім впливом. Заміна конструкційних деталей необхідна для забезпечення високої виживання компонентів.

Зменшення розміру, енергоспоживання, підвищення надійності компонентів і доставка в режимі реального часу можливі за допомогою найсучасніших інтегрованих технологій та надвеликих інтегральних схем (VLSI).

Функціональне моделювання. Функціональне моделювання - це визначення цифрового видавництва, залежно від набору цифрових носіїв інформації. Нейронні мережі утворюють модель, яка знаходить зв’язок між входами та бажаним виходом. Така модель може вирішити широке коло проблем. Прикладом цього є оцінка (виробництво) цін на житло, яка залежить від ряду показників - під’їздів (розмір будинку, розмір землі, відстань до найближчої школи тощо).

Інший приклад - дослідження нейронної мережі, що включає зображення дороги, і вихід - сигнали рульового управління на кермо, педалі газу, гальма тощо.

Функціональне моделювання передбачає надзвичайно широкий спектр застосувань, і нейронні мережі часто використовуються для цієї мети. Це можна розглядати як альтернативу традиційному програмуванню. У традиційному комп'ютерному програмуванні безпосередньо даються вказівки щодо того, як виконати обчислення або завдання. У функціональному моделюванні замість того, щоб розповідати комп’ютеру, як це робити, наводяться приклади того, що потрібно робити. Це дає нам дані про вхідні значення і показує, якими повинні бути вихідні значення, тоді як нейронні мережі вже самостійно визначають функцію залежності від виходу на входах.

Класифікація. Класифікація - це окремий випадок функціонального моделювання, коли виконується розпізнавання зразків. Нейронна мережа вчить, як категорії класифікуються, беручи вхідні значення в набір класів. Нейронні мережі

показали перевагу над традиційними методами у всіх аспектах класифікації і дуже популярні у розпізнаванні образів.

Типовим прикладом класифікації є розпізнавання почерку. Рукописний символ подається на вхід такої системи, а на виході відображається, який це символ (конкретна буква або цифра).

Інший приклад - тестування якості пива (яке насправді використовується на деяких пивоварнях), де нейронні мережі розподіляють різні купи пива в різних класах якості (вихід), залежно від різних хімічних параметрів (вхід).

Класифікація також є результатом прикладу парадигми навчання. Нейронні мережі вивчаються на основі прикладів вхідних даних та їх належності до певного класу. Нейронні мережі будують загальну класифікацію, яку можна застосувати до нових вхідних даних.

Динамічна фільтрація. Описані вище проблеми стосувались статичних даних, тобто коли певний ввід ініціював певне видання. У системах, що мають пам'ять (де вихідні значення залежать від попередніх вхідних значень) та внутрішню динаміку, існують різні типи проблем. Такі проблеми пов'язані з динамікою часу і їх дуже важко вирішити традиційними алгоритмічними методами. У той же час нейронні мережі легко справляються із завданням.

Прикладом завдання динамічної фільтрації є придушення шуму нейронної мережі, що дозволяє ефективно видаляти шум від різних (наприклад, звукових) сигналів.

Іншим прикладом проблеми динамічної фільтрації є класифікація динамічних сигналів. На відміну від звичайної класифікації, яка показує динамічну фільтрацію вхідних даних до класу, вона класифікує зміни сигналу з часом. Наприклад, є посилання на тіло, де людину однозначно ідентифікують відповідно до того, як вона рухається.

Динамічна фільтрація, така як класифікація, також використовує принцип навчання на прикладах. Різниця між ними полягає в тому, що тут введена ціла послідовність введених значень.

Прогнозування. Прогнози - це особливий тип динамічної фільтрації, де минулі значення використовуються для прогнозування майбутніх значень. Він широко використовується для прогнозування у випадках, коли це часовий ряд, який

поширюється на сучасність, і вам потрібно передбачити, яким він буде в майбутньому.

Типовим прикладом прогнозу є прогноз фондового ринку або валютного ринку. Іншим прикладом прогнозування є прогнозування пікового навантаження в мережі.

Прогнозування майбутніх умов дуже складних систем є складним завданням. Але нейронні мережі, які навчились на правильних прикладах навчання, показали, що вони можуть робити хороші прогнози. І вони показують набагато кращі результати, ніж традиційні методи прогнозування.

Ідентифікація системи. Ідентифікація системи - це ще один особливий тип динамічної фільтрації, який має на меті побудувати математичну модель, що відображає функції та динаміку реальної системи. Такі моделі дуже важливі в процесі аналізу, моделювання, прогнозування, моніторингу та розробки системи управління. Основна відмінність між нейронними мережами та традиційними методами ідентифікації систем полягає в тому, що нейронні мережі є нелінійними, що дає їм набагато більше можливостей моделювання.

Ще однією великою перевагою є те, що нейронні мережі можуть використовувати пасивні дослідження для побудови моделі. Традиційні методи зазвичай вимагають чіткого визначення спектральних властивостей системи, посилаючи через неї білий шум. Ви можете помітити, що ніхто не хоче надсилати білий шум через атомну електростанцію. Нейронні мережі можуть "слухати" лише звичайні сигнали, без небажаних досліджень, і при цьому створюють хороші загальні моделі.

Типовим прикладом можливості використання систем ідентифікації нейронних мереж є обробна промисловість. Для виконання відхилень, оптимізації тощо.

Результат ідентифікації системи може бути введений, формування інвертованої системи ідентифікації або системи управління. У цьому режимі нейронна мережа вказує, який результат повинна мати система, і показує, які дані потрібно вказати.

Групування. Кластеризація - це поділ набору вхідних даних на групи (кластери) відповідно до рівня подібності між ними. Нейронні мережі можуть

реалізовувати автономний кластер даних, тобто їм не потрібно подавати будь-який сигнал зворотного зв'язку або систему, щоб робити те, що їм потрібно робити.

Зокрема, кластеризація використовується для визначення сегментації ринку. Використовуючи основні дані досліджень, нейронна мережа може групувати людей за подібністю (багатовимірна). Кластерні системи є дуже корисним інструментом для видобутку даних.

Вибір брендів. Ідентифікація символів - це пошук характерних ознак даних. Нейронні мережі можуть здійснювати автономний вибір властивостей із вхідних даних. Ця функція дуже цінна для стиснення даних. Оскільки нейронні мережі адаптуються до конкретних даних, вони можуть стискатись набагато краще, ніж універсальний метод стиснення.

Виявлення розбіжностей. Нейронні мережі можуть дізнатися про стандартну поведінку системи (або людини) і можуть реагувати певною мірою, коли ця поведінка характеризується. Цей процес називається виявленням аномалій або фільтром новин.

Типовим прикладом є система мережевої безпеки, де відстежується мережевий трафік даних, а дані про діяльність передаються в адаптивну систему. Через деякий час адаптивна система створює модель для нормальної роботи мережі. Коли трапляється щось незвичне (наприклад, спроба хакера атакувати), адаптивна система повідомляє про це.

Виявлення відхилень важливо на промислових підприємствах та в обладнанні медичного контролю. Його також можна використовувати як показник ситуації на фінансовому ринку.

3.3 Структура БСУТП

Основними завданнями сучасного BSUTP є дистанційне управління, управління, управління активацією та процеси в цілому. Виконання таких завдань включає збір даних, створення загального інформаційного простору з надійною,

повною та оперативною інформацією, обробку даних, генерацію керуючих сигналів керуючого сигналу та рішення оператора щодо управління. У такій системі кожен рівень виконує свої завдання, які пов'язані з підвищенням ефективності системи в цілому. Характеристиками BSUTP повинні бути: висока гнучкість конфігурації, функція дистанційного керування, а також використання компонентно-орієнтованих протоколів для взаємодії між системними компонентами.

Апаратні та програмні компоненти BSUTP повинні включати: збір даних; Створити єдиний інформаційний простір із надійною, всебічною та оперативною інформацією; Обробка даних; Формування керуючих сигналів для операторів та правильних управлінських рішень; Контроль за виконавчими механізмами. Для забезпечення перерахованих вимог розроблена базова структура БСУТП, яка наведена на рисунку 3.2.

Рисунок 3.2. Базова структура БСУТП

Розроблена базова структура БСУТП складається з трьох рівнів:

1. збору даних та управління виконавчими механізмами;
2. контролю та управління технологічними процесами;
3. операторського контролю та формування управлінських рішень.

Специфіка кожного рівня системи управління процесами визначається апаратними та програмними компонентами [8]. На кожному ієрархічному рівні управління вирішуються завдання відповідного рівня складності.

Рівень збору даних та механізмів управління. На цьому рівні формується первинна інформація, яка попередньо обробляється, накопичується і надходить за допомогою контролю. Використовуючи цю інформацію, генеруються сигнали для управління активаторами та процесами. Мікроконтролери сімейства STM8 використовуються для попередньої обробки даних датчиків на наявність шуму та неповної інформації.

Рівень контролю та управління технологічним процесом. Цей рівень управління вважається досить автономним, який за відсутності зв'язку верхнього рівня може працювати автономно протягом тривалого часу, не втрачаючи інформації. На цьому рівні використовуються системи мікроконтролерів RaspberryPi та Arduino, програмовані логічні контролери Mitsubishi Melsec FX3U, засоби візуального контролю та управління процесами. Мікроконтролерні системи використовуються як для оперативних обчислень, для прогнозування поведінки процесу та об'єктів управління, так і для інтелектуалізації компонентів системи управління. Обробка даних на цьому рівні зменшує обсяг даних, які потрібно передати верхньому шару, і зменшує вимоги до пропускної здатності каналів зв'язку.

На цьому рівні існують суворі вимоги до надійності апаратного та програмного забезпечення, часу відгуку на дані датчика та формування керуючих сигналів для виконавчих механізмів. Взагалі, апаратне та програмне забезпечення має працювати в режимі реального часу, тобто воно гарантовано реагує на зовнішні події в час, зазначений для кожної з цих подій.

Рівень контролю оператора та формування управлінських рішень. На цьому рівні контроль оператора представлений автоматизованим робочим місцем оператора. Методики, що застосовуватимуться на цьому рівні, можуть визначатися інтенсивністю даних, складністю алгоритмів обробки та вимогами до надійності системи управління. Робочі станції операторів на платформах RISC або Intel можна використовувати як апаратне забезпечення. Завданнями цього рівня є: збір даних із

програмованих логічних контролерів та систем мікроконтролера; Інформаційне сховище; Обробка даних; Потокове передавання відео, розпізнавання зображень та сцен у системах технічного зору; Формування управлінських рішень; Одноразова синхронізація в розподіленій системі; Синхронізація розподілених підсистем; Візуалізація та демонстрація прогресу технологічного процесу.

Для вирішення цих проблем використовуються системи SCADA, основною функцією яких є створення інтерфейсу оператора та збір даних процесу. Інструменти, що являють собою поєднання апаратних засобів, каналів зв'язку та алгоритмічних програм, використовуються для управління та управління технологічним процесом.

Особливостями задач, які розв’язуються на рівні операторського контролю та формування управлінських рішень є:

1. великий обсяг і різнорідність даних;

2. суперечливість та неповнота даних;

3. постійність і висока інтенсивність надходження вхідних даних;

4. великий обсяг обчислень з переважанням обчислювальних операцій над логічними при опрацюванні відео-потоків, розпізнавання зображень і сцен в системах технічного зору; постійне ускладнення алгоритмів опрацювання та підвищення вимог до точності результатів;

5. можливість розпаралелення опрацювання даних як у часі, так і у просторі.

На відміну від звичайних VLSI, ПЛІС доступні за доступною ціною і скоротили цикл розробки апаратного забезпечення. Крім того, системи FPGA можна налаштувати під конкретні конфігурації ANN. Наприклад, відкладені та інші. [91] Введено введення систолічної матриці для багатошарового персептрону на основі алгоритму конвеєра Xilinx Virtex XCV400 FPGA для вивчення зворотного множення. Гуціл і Гірау [24] відображають нейронну мережу для сегментації зображень на пристрої Xilinx Virtex XC2V1500FF896-4. Однак множення є дорогим при використанні FPGA, оскільки для кожного синаптичного хреста потрібен множник, і це число зазвичай збільшується на квадрат числа нейронів. Однак сучасні ПЛІС, такі як Xilinx Virtex II Pro [166] з інтегрованими ядрами IBM PowerPC та Altera Stratix III [16], можуть мати сотні спеціалізованих множників.

У відносно недавньому дослідженні Himavachi et al. [18] Xilinx FPGA XCV400hq240 Метод мультиплексування шарів для багатошарових мереж прямого множення. Запропонований спосіб мультиплексування шарів включає лише шар з найбільшою кількістю нейронів. Окремо розроблений блок управління правильно відбирає нейрони з цього шару, щоб імітувати поведінку будь-якого іншого шару, і паралельно призначає відповідні входи, ваги, компенсації та вихідні функції кожному нейрону в поточному модельованому шарі. Кожен окремий нейрон несеться у вигляді таблиці (таблиці пошуку). Для оцінки ефективності проектного рішення був використаний датчик оцінки потоку двигуна, який показав 50% зменшення кількості нейронів, хоча це збільшило час розрахунку на 17,7% завдяки наявності блоку управління.

В інших недавніх дослідженнях Райс та співавт. [17], що впровадження когнітивної моделі мозку на основі FPGA дало в 75 разів більше результатів, ніж впровадження суперкомп'ютерного програмного забезпечення Cray XD1. Вони використали ієрархічну байєсівську модель мережі, засновану на неокортексі, розроблену Джорджем та Хокінсом. Їх апаратне прискорене впровадження на Cray XD1 використовує Xilinx Virtex II Pro FPGA із зовнішнім SRAM та програмною реалізацією з використанням 5-ядерного 2 Гц процесора Opteron.

Важливою проблемою, з якою стикаються розробники ASM на базі FPGA, є вибір відповідної моделі ANN для певного завдання, забезпечуючи оптимальне використання апаратних ресурсів. З цією метою Саймон Джотсон та співавт. [17] пропонують цікаве рішення. Вони провели порівняльний аналіз вимог до обладнання на основі FPGA для чотирьох моделей ANP. Вибрані моделі включають класичний багатошаровий персептрон із зворотним множенням та мережу RBF та дві моделі стохастичних нейронних мереж - мережу LIF (інтеграція витоків та пожежа) та модель мережі управління голкою (мережа відповіді спайків). Витрати на методи впровадження ПЛІС аналізували в класифікаційному тесті. Результати дослідження показують, що мережа LIF може бути найбільш підходящим вибором для реалізації нелінійної проблеми класифікації.

FPGA Впровадження стохастичних моделей ANN: Впровадження великих практичних методів ANN критично вимагає значного скорочення схем множення.

Одним із способів їх зменшення є використання бітових стохастичних обчислень [173]. Вони використовують відносно довгі, приблизні потоки, чисельне значення яких пропорційне їх "1" -т-щільності. Наприклад, дійсне число r [-1; 1] представлена у такій двійковій послідовності, що ймовірність отримання значення 1 дорівнює (r + 1) / 2. Помноження потоку двох ймовірних потоків може бути здійснено за допомогою логічного затвора з двох входів. Це дозволяє реалізувати великі, щільні, повністю паралельні мережі з відмовостійкістю. Однак, хоча стохастичні розрахунки є простими, вони не завжди ефективні (для порівняння див. [59]).

3.4 Висновки до розділу

Створення сучасних багаторівневих систем управління процесами базується на системній інтеграції, яка базується на системному підході, що включає всі рівні інтеграції процесів, споруд, споруд та інфраструктури з урахуванням ефективності їх використання та вимог конкретних додатків.

Пропонується системна інтеграція на основі компонентно-орієнтованої технології, яка забезпечує розподіл процесу розробки на ієрархічні рівні та типи програмного забезпечення: алгоритм, апаратне та програмне забезпечення.

Розробка апаратних компонентів з високою швидкістю, малою потужністю та низькою потужністю в рівнях управління на декількох рівнях здійснюється на базі мікроконтролерів із фіксованими комами, невеликих мереж, скороченого інтерфейсу та скороченої системи команд.

Для програмування мікроконтролерів рекомендується використовувати середовище програмування CoDeSys (система розробки контролерів), редактори та програми для усунення несправностей, засновані на принципах професійного середовища програмування (VisualC ++ та ін.).

У сучасних багаторівневих системах управління процесами бажано використовувати пристрій технології Hive для підключення дротових та бездротових каналів датчиків, систем мікроконтролера та виконавчих механізмів для підключення до Інтернету.

ВИСНОВКИ

При реалізації БСУТП постає задача – формування інтегрованої системи обміну та накопичення інформації з розосереджених систем, метою якої є забезпечення об’єднання між собою провідними та безпровідними каналами, взаємодія з Інтернет давачів, мікроконтролерних систем і виконавчих механізмів у БСУТП.

Створення сучасних багаторівневих систем управління процесами базується на підході до системної інтеграції, що включає всі рівні інтеграції процесів, об'єктів, об'єктів та інфраструктури, з метою ефективності їх використання та вимог конкретних додатків.

В роботі пропонується системна інтеграція на основі компонентно-орієнтованої технології, яка забезпечує розподіл процесу розробки на ієрархічні рівні та типи програмного забезпечення: алгоритмічне, технічне та програмне.

Розробка високошвидкісних, малопотужних, високошвидкісних апаратних компонентів для багаторівневих рівнів управління базується на мікроконтролерах із фіксованою точкою, невеликих мережах, усічених інтерфейсах та скорочених системах команд.

В магістерській роботі виконано наступні завдання:

– здійснено аналіз предметної області

– досліджено та проаналізувано базову концепцію БСУТП

– проведено синтез структури БСУТП на основі компонентно-орієнтованої технології

Для програмування мікроконтролерів рекомендується використовувати середовище програмування CoDeSys (система розробки контролерів), редактори та програми налагодження на основі принципів професійного середовища програмування (VisualC ++ та ін.).

У сучасних багаторівневих системах управління процесами рекомендується використовувати пристрій технології Hive для поєднання датчиків, систем мікроконтролера та дротових та бездротових каналів для драйверів та підключення до Інтернету.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ахо А., Хопкрофт Дж., Ульман Дж. Структуры данных и алгоритмы М.: Изд. Дом «Вильямс», 2001. 384с.
2. Аряшев С.И., Параллельный перепрограммируемый вычислитель для систем обработки информационных сигналов / Бобков С.Г, Сидоров Е.А.//Нейроинформатика Москва, МИФИ. Часть 2. С.-2015.
3. Березький О.М., Методичні рекомендації до виконання кваліфікаційної роботи з освітнього ступеня “Магістр”. Спеціальність: 123 - Комп’ютерна інженерія. Магістерська програма - Комп’ютерна інженерія"/ Дубчак Л.О., Мельник Г.М. // Під ред. О.М. Березького. Тернопіль:ЗУНУ,2020.32 с.
4. Гураль І.В., Дубчак Л.О. Методичні вказівки до оформлення курсових проектів, звітів про проходження практики, випускних кваліфікаційних робіт для студентів спеціальності «Комп’ютерна інженерія»/Під ред. О.М. Березького. Тернопіль: ТНЕУ, 2019. 33 с.
5. Браунси К. Основные концепции структур данных и реализация в С++. – М.: Изд. Дом «Вильямс», 2002. 320с
6. Гончар Л.І., Методи та засоби забезпечення оперативного відновлення та доступності даних в автоматизованих інформаційних системах / Шурдак Р.В., Ваврух Р.В., Пастернак Ю.І., Олійник О.О // Комп’ютерні інформаційні технології: Матеріали школи-семінару молодих вчених і студентів СІТ’2020. Тернопіль: ЗУНУ, 2020 с -.21
7. Олійник О.О., Багаторівнева система управління технологічними процесами/ Ігнатєв І.В.// ІІІ Науково-практична конференція молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі». 26 листопада 2020 р. Тернопіль. Україна с.67
8. Кирсанов Є.Ю. Цифровые нейрокомпьютеры: Архитектура и схемотехника / Под ред. А.И. Галушкина. - Казань: Казанский Гос. У-т. 1995.
9. Минаев И.Г. Программируемые логические контроллеры: практическое руководство для начинающего инженера / И.Г. Минаев, В.В. Самойленко.

Ставрополь : АГРУС, 2009. 100 с.

10. Шишов О.В. Современные технологи промышленной автоматизации: учебник / О. В. Шишов. Саранск : Изд-во Мордов. ун-та, 2007. 273 с.

11. Рассел С., Норвиг П. Искусственный интеллект: современный подход / Пер. с английского М.: Вильямс, 2007. 1408 с.

12. Рассел С., Норвиг П. Искусственный интеллект: современный подход / Пер. с английского М.: Вильямс, 2007. 1408 с.

13. Яковлев О.Г. Программирование ПЛК / Яковлев О.Г. Москва, 2001.

14. Інтелектуальні компоненти інтегрованих автоматизованих систем управління : монографія / Медиковський М.О., Ткаченко Р.О., Цмоць І.Г., Цимбал Ю.В., Дорошенко А.В., Скорохода О.В. Львів : Видавництво Львівської політехніки, 2015.–280 с.

15. Цмоць І.Г. Визначення задач і формування вимог до інтелектуальних компонентів інтегрованих АСУ / І.Г. Цмоць, О.В. Скорохода // Технічні вісті : науково-технічний журнал. 2014. 2(40), 2(34). С. 53–54.

16. Цмоць І.Г. Багатоканальний пристрій з буферизацією даних для автоматизованих систем управління технологічним процесом з промисловою мережею / І.Г. Цмоць, Б.А. Демида, Х.Г. Гульовата // Науковий вісник національного лісотехнічного університету України : збірник науково-технічних праць. Львів : РВВ НЛТУ України. –2009. Вип. 19.5. С. 275–284.

17. Цмоць І.Г. Проектування багатоканального пристрою обміну даними для АСУ ТП з промисловою мережею / І.Г. Цмоць, М.Р. Подольський // Системні технології : регіональний міжвузівський збірник наукових праць. Дніпропетровськ, 2009. Випуск 6 (65). С. 131–140.

18. Харазов В.Г. Интегрированные системы управления технологическими процессами / Харазов В.Г. СПб : Профессия, 2009. 592 с.

19. Пупена О.М. Промислові мережі та інтеграційні технології в автоматизованих системах : навчальний посібник / Пупена О.М., Ельперін І.В., Луцька Н.М., Ладанюк А.П. К. : Вид-во «Ліра-к», 2011. 552 с.

20. Пьявченко Т.А. Проектирование АСУТП в SCADA-системе : учебное пособие / Пьявченко Т.А. Таганрог : Изд-во Технологического института ЮФУ,

2007. 84 с.

21. Bjorn Solvang, Gabor Sziebig, and Peter Korondi. Multilevel Control of Flexible Manufacturing Systems / Int. Conf. on Flexible Manufacturing Systems, Krakow, Poland, May 25–27, 2008, pp. 785–790.

22. S.H. Suh, B.E. Lee, D.H. Chung, S.U. Cheon. Architecture and implementation of a shop–floor programming system for STEP-compliant CNC / Computer–Aided Design, vol. 35, Elsevier, 2002, pp. 1069–1083.

23. X.W. Wu. Realisation of STEP-NC enabled machining / Robotics and Computer-Integrated Manufacturing, Volume 22, Elsevier, 2006. 144–153.

24. Tancheng Xie, Yuanzhou Xu, Xiang Nan, Hongru Wang, Yuanwei Xu. R&D of General MachineOriented Manufacturing Unit / International Conference on Mechatronics and Automation, 2006. 457–462.

25. Nuno Lopes, Pedro Lima. OpenSDK An Open-source Implementation of OPEN-R / Proc. of 7th International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2008), Estoril, Portugal, 2008.

26. Noriaki Ando, Geoffrey Biggs, Tetsuo Kotoku. Robotic Component Specification / Document 12-09- 01.pdf of AIST-National Institute of Advanced Industrial Science and Technology, 2012.22 p.

27. D. Y. Chao. Enumeration of lost states of a suboptimal control model of a well-known S3PR / IET Control Theory & Applications, Volume 5, Issue 11, (2011). 1277–1286.

28. Paulo Leitao. Agent-based distributed manufacturing control: A state-of-the-art survey / Engineering Applications of Artificial Intelligence. Volume 22, Issue 7 (2009). 979–991.

29. Herve Panetto. Enterprise integration and interoperability in manufacturing systems: Trends and issues / Herve Panetto, Arturo Molina. Computers in industry Volume 59, Issue 7 (2008). 641–646.

30. Artificial neural networks for intelligent manufacturing / Edited by Cihan H. Dagli. Intelligent Manufacturing Series, Springer Science & Business Media, 2012. ISBN 978-94-010-4307-6.

31. Mikell P. Groover. Automation, production systems, and computer-integrated

manufacturing / 4th Edition by G. Jayaprakash. Pearson Education Limited, 2016. 798 p.

32. James A. Rehg. Computer-Integrated Manufacturing / James A. Rehg, Henry W. Kraebber. 3rd Edition. Prentice Hall, 2012. 592 p.

33. Программируемые логические контроллеры Mitsubishi Electric [Электронный ресурс]. Режим доступа : http://www.sovras.com/fx.php

34. http://www.raspberrypi.org/archives/2180

35. http://raspberrypi.ru/page/doc/

36. http://eu.mouser.com/new/stmicroelectronics/stm8s-mcus/

37. http://www2.st.com/content/st_com/en.html

38. http://www.compel.ru/lib/ne/2013/8/8-stm8-dlya-chaynikov-pervyie-shagi-v-srede-iar/

39. Нейроприскорювачі на базі нейрочіпів - http://citforum.ru/hardware/neurocomp/neyrocomp_07.shtml

40. BPMN 2.0 ИЗ ЧЕГО СОСТОИТ МОДЕЛЬ БИЗНЕС ПРОЦЕССА [Електронний ресурс] Режим доступу до ресурсу: https://rzbpm.ru/knowledge/bpmn-2-0-iz-chego-sostoit-model-biznes-processa.html.

41. Camunda Architecture Overview [Електронний ресурс] Режим доступу до ресурсу: https://docs.camunda.org/manual/7.12/introduction/architecture/.

42. Camunda modeler [Електронний ресурс] Режим доступу до ресурсу: https://camunda.com/download/modeler/.

43. Camunda tutorial [Електронний ресурс] Режим доступу до ресурсу: https://camunda.com/learn/videos/.

44. Краткое описание BPMN с примером [Електронний ресурс] Режим доступу до ресурсу: https://habr.com/ru/company/trinion/blog/331254/.

45. Линейная регрессия и методы её восстановления [Електронний ресурс] Режим доступу до ресурсу: https://habr.com/ru/post/465743/.

46. Линейная регрессия [Електронний ресурс] Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/%D0%9B%D0%B8%D0%BD%D0%B5%D0%B9%D0%BD%D0%

47. Нотация BPMN 2.0: ключевые элементы и описание [Електронний ресурс] Режим доступу до ресурсу: https://www.comindware.com/ru/blog

48. Применение API Gateway [Електронний ресурс] Режим доступу до ресурсу: http://agilemindset.ru/

49. Клієнт-серверна архітектура [Електронний ресурс] Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/

50. Методи екстраполяції [Електронний ресурс] Режим доступу до ресурсу: https://pidruchniki.com1/metodi_ekstrapolyatsiyi.