Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп’ютерної інженерії та управління
Кафедра Комп’ютерних інтелектуальних технологій та систем

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Адаптивна інтелектуальна система автоматичного догляду за рослинами

Виконав:
студент 2 курсу, групи КІТм-21-1
Лукашин О.В.
Спеціальність 123 Комп’ютерна інженерія
Тип програми освітньо-професійна
Освітня програма Комп’ютерні інтелектуальні технології
Керівник доц. каф.КІТС Сердюк Н.М.

Допускається до захисту _____
(підпис)

Зав. кафедри _____ О.Г. Руденко
(підпис)

2022 р.

Харківський національний університет радіоелектроніки

Факультет	Комп'ютерної інженерії та управління
Кафедра	Комп'ютерних інтелектуальних технологій та систем
Рівень вищої освіти	другий (магістерський)
Спеціальність	123 Комп'ютерна інженерія
Тип програми	освітньо-професійна
Освітня програма	Комп'ютерні інтелектуальні технології

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)
«_____»_____202_ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові _____Лукашину Олександру Віталійовичу_____

1. Тема роботи (проекту)___Адаптивна інтелектуальна система автоматичного догляду за рослинами затверджена наказом університету від 7 листопада 2022р. № 1455Ст
2. Термін подання студентом роботи до екзаменаційної комісії _10 _____грудня____2022р.
3. Вихідні дані до роботи (проекту)____Теоретичні відомості про існуючі методи _____ __ програмування на ПЛІС. _____ ___Теоретичні відомості щодо розробки нейропроцесорних мереж_____ _____ _____ _____ _____ _____ _____
4. Перелік питань, що потрібно опрацювати в роботі _____ _____ вибір засобів для реалізації нейропроцесорної системи _____ ____програмна реалізація автоматичної системи догляду за рослинами_____ ____тестування системи_____ ____висновки по роботі_____
5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням кафедри)
_____17 слайдів_____ _____52 рисунки_____ _____7 таблиць_____ _____ _____

6. Консультанти розділів роботи (п.6 включається до завдання за наявністю консультантів згідно до наказу, зазначеному у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Видача та узгодження теми	08.11.2022	виконано
2	Аналіз предметної області, постановка задачі, вибір інструменатльних засобів	08.11.2022 – 10.11.2022	Виконано
3	Розробка алгоритмів і структури роботи	11.11.2022 – 17.11.2022	Виконано
4	Розробка структурної схеми нейропроцесора	18.11.2022 – 20.11.2022	Виконано
5	Реалізація архітектури логічного процесора	21.11.2022 – 24.11.2022	Виконано
6	Реалізація інтелектуальної системи	25.11.2022 – 29.11.2022	Виконано
7	Розрахунок надійності розробленої системи	30.11.2022 – 01.12.2022	Виконано
8	Оформлення матеріалів кваліфікаційної роботи	02.12.2022 – 17.12.2022	Виконано
9	Подання кваліфікаційної роботи до ЕК	19.12.2022	Виконано
10	Захист	20.12.2022	виконано

Дата видачі завдання 08 листопада 2021 р.

Студент _____
(підпис)

Керівник роботи _____ ___доц. каф. Н.М. Сердюк_____
(підпис) (посада, ініціали,прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 111 с., 52 рис., 7 табл., 9 дод., 65 джерел.

У магістерській кваліфікаційній роботі виконано проєктування нейропроцесора для виконання логічних операцій по керуванню електроприводом системи інтелектуального догляду за рослинами.

Метою роботи є проектування нейропроцесора для виконання логічних операцій по керуванню електроприводом системи інтелектуального догляду за рослинами, виконаного на сучасній елементній базі і перевірка ефективності його використання.

Об’єктом дослідження є нейропроцесор для управління електроприводами виконавчого механізму.

Предметом дослідження є модель нейропроцесора для виконання логічних операцій по керуванню електроприводами виконавчих механізмів системи автоматичного догляду за рослинами

За результатами проведеного проектування, розроблена система успішно впоралась зі своєю задачею. Розроблена автоматична система дає змогу безвідмовної роботи протягом 11 років, що дає змогу конкурувати серед інших продуктів даної області.

САПР, VHDL, ПЛІС, ПРОЄКТУВАННЯ, НЕЙРОМЕРЕЖА, ПРОЄКТУВАННЯ, НЕЙРОПРОЦЕСОР.

ABSTRACT

Explanatory note of the qualification work: 111 pages, 52 figures, 7 tables, 9 appendices, 65 sources.

In the master's qualification work, the design of a neuroprocessor for the execution of logical operations for controlling the electric drive of the intelligent plant care system was performed.

The purpose of the work is the design of a neuroprocessor to perform logical operations for controlling the electric drive of the system of intelligent plant care, made on a modern element base, and checking the effectiveness of its use.

The object of research is a neuroprocessor for controlling the electric drives of the executive mechanism.

The subject of the study is a model of a neuroprocessor for performing logical operations for controlling the electric drives of the executive mechanisms of the automatic plant care system

According to the results of the design, the developed system successfully coped with its task. The developed automatic system enables trouble-free operation for 11 years, which makes it possible to compete among other products in this area.

CAD, VHDL, FPGA, DESIGN, NEURAL NETWORK, DESIGN, NEURO-?ROCESSOR.

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп’ютерної інженерії та управління
Кафедра Комп’ютерних інтелектуальних технологій та систем

АНОТАЦІЯ

КВАЛІФІКАЦІЙНОЇ РОБОТИ

рівень вищої освіти другий (магістерський)

Адаптивна інтелектуальна система автоматичного догляду за рослинами

Виконав:
студент 2 курсу, групи КІТм-21-1
_____Лукашин О.В._____
Спеціальність 123 Комп’ютерна інженерія
Тип програми освітньо-професійна
Освітня програма Комп’ютерні інтелектуальні технології
Керівник к.т.н, доц. каф. КІТС Сердюк Н.М

2022 р.

АНОТАЦІЯ

Лукашин О. В. Адаптивна інтелектуальна система автоматичного догляду за рослинами. – Магістерська кваліфікаційна робота.

У магістерській кваліфікаційній роботі виконано проєктування нейропроцесора для виконання логічних операцій по керуванню електроприводом системи інтелектуального догляду за рослинами.

Метою роботи є проектування нейропроцесора для виконання логічних операцій по керуванню електроприводом системи інтелектуального догляду за рослинами, виконаного на сучасній елементній базі і перевірка ефективності його використання.

Об’єктом дослідження є нейропроцесор для управління електроприводами виконавчого механізму.

Предметом дослідження є модель нейропроцесора для виконання логічних операцій по керуванню електроприводами виконавчих механізмів системи автоматичного догляду за рослинами.

В ході дослідження нами проаналізовано теоретичний матеріал щодо аналізу об’єкта проєктування. Відповідно, мова опису віртуальних моделей HDL спроєктована для всього спектру потреб, які виникають в процесі проєктування.

Вона дозволяє описати структуру проєкту, тобто його поділ на складові частини та їх взаємозв’язок; вона дозволяє описати функцію проєкту використовуючи подібні до мови програмування форми; вона дозволяє змоделювати проєкт перед початком виготовлення, так що проєктувальники можуть швидко порівняти альтернативи та перевірити правильність функціонування без затримки та витрат на апаратне макетування.

В даній дипломній роботі розроблено приклад VHDL моделі логічного нейропроцесора для виконання логічних операцій. В ході проєктування проведений аналіз роботи логічного нейропроцесора та виконано розробку алгоритму функціонування типового фізичного нейропроцесора з подальшим перенесенням на VHDL-опис команд.

Виконано розробку структурної схеми логічного нейропроцесора на мові VHDL та виконано вибір апаратної частини проєктованого пристрою шляхом порівняльного аналізу декількох ПЛІС, які підходять для реалізації поставленого технічного завдання.

Алгоритм роботи пристрою є системою послідовних функціональних перетворень з метою одержання заданого вихідного сигналу. Оскільки ядро розроблюваної системи складає ПЛІС, то основне навантаження виконує програма, що здійснює необхідні функції перетворення та обчислення. Використовуючи потужний нейропроцесор, можна звести апаратні затрати до мінімуму. Незважаючи на це, позбутися їх повністю не можна, оскільки всі сигнали, потрібно перетворювати в норму, доступну для програмної обробки.

Для реалізації силової часини нами спочатку розроблено алгоритмічну та структурну схему мікропроцесорного ШІМ-перетворювача виконавчого механізму керування електроприводом системи поливу. Алгоритм роботи мікропроцесорного ШІМ-перетворювача є системою послідовних функціональних перетворень з метою одержання заданого вихідного сигналу.

Оскільки ядро розроблюваної системи складає мікропроцесор, то основне навантаження виконує програма, що здійснює необхідні функції перетворення та обчислення. Використовуючи потужний мікропроцесор, можна звести апаратні затрати до мінімуму. Незважаючи на це, позбутися їх повністю не можна, оскільки всі сигнали, потрібно перетворювати в норму, доступну для програмної обробки.

В результаті проведеної роботи сконструйовано систему мікропроцесорного ШІМ-перетворювача на базі мікроконтролера серії Atmega32. Вибраний мікроконтролер має досить високу розрядність шини даних ($n \geq 16$), оскільки використання мікроконтролера з шиною меншої розрядності приводить значних апаратних витрат – використання проміжних запам’ятовуючих прист регістрів для побайтової розбивки.

Мікроконтролер повинен мати широкий набір математичних функцій, оскільки для виконання поставленої задачі передбачається використання поставленої задачі передбачається використання чисельних методів інтегрування, та операція ділення. Мікроконтролер повинен мати досить високу швидкодію щоб не відбувалося, втрачання інформації, що поступає на вхід пристрою.

Всім необхідним вимогами для проєктування мікропроцесорного ШІМ-перетворювача відповідає мікропроцесорний комплект на великих інтегральних схемах підвищеної степені інтеграції. Центральний процесор Atmega32 має розрядність даних 16 біт, розрядність адреси – 20 біт і тактову частоту до 5 МГц. Його продуктивність на порядок вище ніж, ніж процесора К580УК – 80. Мікросхема Atmega32 являє собою одно кристальний 16-ти бітовий МП, виконаний по технології п-МОП. Кристал мікросхеми містить – 29000 транзисторів, та має розміри 5,5х5,5мм. Atmega32 містить 14 32-бітових внутрішніх регістрів і утворює 32-бітову шину даних для зв'язку з зовнішньою пам'яттю та портами вводу-виводу.

Шина адреси має 20 розрядів, що дозволяє безпосередньо адресуватись до пам'яті до ЗП ємністю до 1МВ = 1048576 байт. Отже, вибраний мікропроцесор задовольняє вимогами, які ставить до нього розроблюваний прилад і використаний для його розробки мікропроцесорного ШІМ-перетворювача.

В ході проєктування виконана розробка тестової програми та на її основі спроєктовано програмні мікрокоди для модулів з яких складається VHDL модель логічного нейропроцесора для виконання логічних операцій. В ході розробки я ознайомився з програмою САПР Xilinx WebPack, та основними її принципами роботи.

Виконано розрахунок надійності проєктованого пристрою, в ході чого установлено, що даний пристрій повинен пропрацювати 11 років безвідмовно, а подальша експлуатація приладу може призвести до його ремонту, який може перевищити вартість самого пристрою.

САПР, VHDL, ПЛІС, ПРОЄКТУВАННЯ, НЕЙРОМЕРЕЖА, ПРОЄКТУВАННЯ, НЕЙРОПРОЦЕСОР.

ЗМІСТ

ВСТУП

Інтелектуальні системи, що заснована на штучних нейронних мережах, дають можливість успішно вирішувати такі проблеми, як розпізнавання образів, прогнозування, оптимізація і керування. Підходи для усунення вищезазначених проблем, які вважаються традиційними, не завжди забезпечують бажану гнучкість і більшість додатків стають кращими завдяки використанню нейромереж. Штучні нейромережі – це електронні моделі нейронної структури мозку, який розвивається за допомогою досвіду. Приведення такої аналогії допомагає зрозуміти, що більшість проблем, які неможливо усунути звичайними комп'ютерами, можуть бути вирішені застосуванням нейромереж.

Еволюція, яка тривала досить довго, привнесла до людського мозку чимало чинників, яких не містять нові комп'ютери з архітектурою фон-Неймана. До таких чинників належать: обробка інформації у розподіленому вигляді, обрахунки, що виконуються паралельно, схильність до навчання новому та узагальнення отриманих знань, усвідомлення скоєних помилок, невеликий рівень енергозатрат.

Вид з'єднання між нейронами має великий вплив на роботу мережі. Більшість пакетів програмних реалізацій нейронних мереж дозволяють користувачеві додавати, віднімати і керувати з'єднаннями як завгодно. Постійно корегуємі параметри зв'язку можна зробити як збудливими так і гальмуючими.

Сьогодні для реалізації пристроїв виконання логічних операцій все частіше використовують ПЛІС (FPGA). Поєднання логічних приладів на основі поведінкового опису, що робиться виконавцем за допомогою такої мови програмування, як VHDL, вважається досить перспективним напряом проектування. Мова VHDL використовується для опису апаратних засобів нейропроцесорних систем. Для виконання зазначеного опису, у VHDL містяться особливі методи. Описування апаратних складників, у

найпростішому варіанті, має у своєму складі специфікації – специфікацію інтерфейсу та архітектури.

VHDL була розроблена для величезного діапазону необхідностей, що можуть з'явитися під час процесу проектування. Вона дає можливість для опису проектної структури, його розділення на частини та логічного зв'язку між ними. Також вона дає можливість охарактеризувати функціонал проекту завдяки використанню форм, які є подібними на мову програмування. Як наслідок, у користувача є змога створити приблизну модель проекту перед тим, як почати його виконання. Завдяки цьому, винахідники зможуть швидко проаналізувати будь-які інші рішення, проконтролювати функціонування та грошові витрати, що були спрямовані на створення макету.

Відповідно, актуальним напрямком дослідження є реалізація нейронних мереж мовою описання апаратних засобів VHDL.

Метою роботи є проектування нейропроцесора для виконання логічних операцій по керуванню електроприводом системи інтелектуального догляду за рослинами, виконаного на сучасній елементній базі і перевірка ефективності його використання.

Для досягнення поставленої мети дослідження, нами були поставлені наступні задачі:

– виконати аналіз характеристик проектованої системи;

– провести вибір засобів для реалізації нейропроцесорної системи;

– виконати апаратну та програмну реалізацію нейропроцесорної системи мовою VHDL для виконання логічних операцій.

Предметом дослідження є модель нейропроцесора для виконання логічних операцій по керуванню електроприводами виконавчих механізмів системи автоматичного догляду за рослинами.

Для виконання роботи використовували такі методи дослідження, як: пошуковий та аналітичний, для знаходження, структуризації та аналізу наявного матеріалу у методичках та науковій літературі, порівняння, проектування, теоретичне моделювання, систематизація, спостереження,

тестування, аналіз документації та результатів діяльності дослідників з проблеми проведеного дослідження та експертна оцінка.

Робота ґрунтується на аналізі науково-методичної літератури, посібників, статей, періодичних видань та напрацювань сучасних та попередніх програмістів та розробників HDL-моделей.

Теоретична та практична цінність роботи полягає в наявності теоретичного матеріалу по дослідженню, відсіяного з-поміж іншого в процесі пошуку інформації по темі, та в систематизації матеріалу напрямку дослідження. Проведене дослідження має більш глибокий ступінь розробки напрямку дослідження, відносно попередніх досліджень вчених, дисертантів та дослідників напрямку дослідження. В дипломній роботі розроблено приклад VHDL моделі логічного нейропроцесора для виконання логічних операцій по керуванню електроприводами системи автоматичного догляду за рослинами.

Робота складається з 111 листів друкованого тексту, 52 рисунків, 7 таблиць та 9 додатків і налічує 65 джерел використаної літератури.

1. КЛАСИФІКАЦІЯ ЗАГАЛЬНИХ МЕТОДІВ СТВОРЕННЯ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ ЗА ДОПОМОГОЮ НЕЙРОННИХ МЕРЕЖ ТА ГЕНЕТИЧНИХ АЛГОРИТМІВ

Системи, які розроблені на використанні біологічних нейронів, мають перераховані характеристики, які признані величезним досягненням в сфері обробки даних. Досягнення в області нейрофізіології дають початкове розуміння механізму природного мислення, де збереження інформації відбувається у виді складних образів. Процес збереження інформації як образів, використання образів і розв'язання поставленої проблеми визначають нову область в обробці даних, яка без використання звичайного програмування, дає змогу створити паралельні мережі та налагодити їх навчання. У лексиконі розроблювачів і користувачів нейромереж присутні слова, відмінні від традиційної обробки даних, зокрема, «поводитися», «реагувати», «самоорганізовувати», «навчати», «узагальнювати» і «забувати» [9, с. 36–45].

Кожен нейрон має тільки один вихідний відросток, по якому він може передавати імпульс декільком іншим нейронам. Одиничний нейрон приймає збудження від величезної кількості нейронів (їхнє число може досягати тисячі). Кожен нейрон передає збудження іншим нейронам через нервові стики, називані синапсами, при цьому процес передачі сигналів має складну електрохімічну природу. Синапс – це функціональний вузлом між двома нейронами. При досягненні синаптичного кінця імпульсом відбувається виробка хімчних речовин, називані нейротрансмиттерами. Нейротрансмиттери проходять через синаптичну щілину, і в залежності від типу синапса, впливають на здатність нейрона-приймача генерувати імпульси у системі через збудження або гальмування цієї здотності. Результативність синапса настроюється сигналами, які через нього проходять, саме через це навчання синапсу залежить від дій процесів, які використовують їх. Взаємодія нейронів відбувається завдяки черзі електричних імпульсів, а повідомлення мають змогу відправлятись завдяки частотоно-імпульсній модуляції [9, с. 36–45].

Синапси мають взаємодію на потужність імпульсу. Таким чином можна відзначити, що сила імпульсу змінюється в декілька разів через проходження синапсу, цю множину можна вважати вагою синапсу. При надходженні пари імпульсів до нейрону – вони підсумовуються. У випадку коли підсумований імпульс досягає визначеного порогу, нейрон збуджується та генерує особистий імпульс і відправляє його по аксіоні. Важливо відзначити, що ваги синапсів можуть змінюватися згодом, а звідси виходить, що змінюється і поводження відповідного нейрона. Синапси відіграють роль репитерів інформації, у результаті функціонування яких збудження може підсилюватися або послаблятися. Як наслідок, до нейрона приходять сигнали, одна частина з яких робить збудливий, а друга – гальмуючий вплив. Нейрон підсумовує збудливі і гальмуючі імпульси. Якщо їхня алгебраїчна сума перевищує деяке граничне значення, то сигнал з виходу нейрона пересилається за допомогою аксона до інших нейронів.

Побудуємо математичну модель описаного процесу (рис. 1.1).

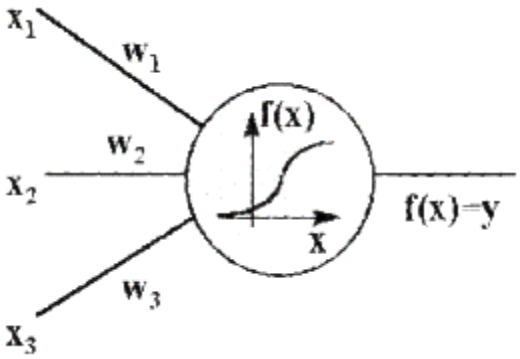


Рис. 1.1 – Спрощена модель нейрона

На рисунку 1.1 ілюстрована модель нейрона яка містить три входи (дендрити), і синапси цих входів мають такі ваги w_1, w_2, w_3. Таким чином при надходженні електричних імпульсів з силами x_1, x_2, x_3 відповідно, то після синапсів і дендритів будуть сформовані такі імпульси $w_1 x_1$, $w_2 x_2$, $w_3 x_3$, які відправляться до нейрону. Нейрон перетворить отриманий сумарний імпульс

відповідно до деякої передатної функції $f(x)$. Отже вихідний імпульс матиме таку силу $y=f(x)=f(w_1x_2+ w_2 x_2+ w_3 x_3)$.

Таким чином, нейрон цілком формується своїми вагами w_k і передатною функцією $f(x)$. Отримавши вектор чисел w_k як входи, нейрон видає деяке число y на виході.

Розглянемо узагальнену модель нейрона (рис. 1.2), зв'язану з першими спробами формалізувати опис функціонування нервової клітки [9, с. 36–45].

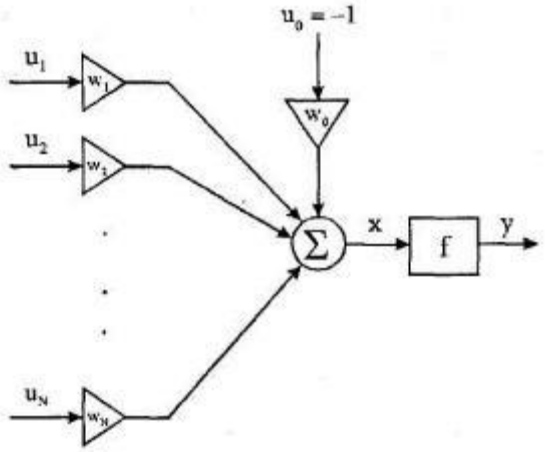


Рис. 1.2 – Узагальнена модель нейрона

Уведемо наступні позначення:

$u_1,..., u_N$ – вхідні сигнали даного нейрона, що приходять від інших нейронів;

$w_1.....w_N$ – синаптичні ваги;

y – вихідний сигнал нейрона;

v- граничне значення.

Формула, що описує функціонування нейрона, має вигляд

$$y=\begin{cases}1 \text{ при } \sum_{i=1}^{N} w_1u_1 \geq v, \\ 0 \text{ при } \sum_{i=1}^{N} w_1u_1 < v.\end{cases} \qquad (1.1)$$

Модель (1.1) може бути представлена у виді

$$y=f\left(\sum_{i=0}^{N} w_i u_i\right), \tag{1.2}$$

де

$$f(x)=\begin{cases}1 \text{ при } x \geq 0,\\ 0 \text{ при } x < 0,\end{cases} \tag{1.3}$$

а також $w_0 = v$, $u_0 = 1$.

Формула (1.2) описує модель нейрона, представлену на рис. 1.6. Ця модель була запропонована в 1943 р. Маккаллоком і Питтсом [9, с. 36–45]. В якості функції f може прийматися не тільки одинична функція (1.3), але й інші граничні функції виду

$$f(x)=\begin{cases}1 \text{ при } x\geq 0,\\ -1 \text{ при } x<0.\end{cases} \tag{1.4}$$

Або

$$f(x)=\begin{cases}1 \text{ при } x>1,\\ -1 \text{ при } x<-1,\\ x \text{ при } \vee x \vee \leq 1.\end{cases} \tag{1.5}$$

На початковій фазі моделювання біологічних нейроних мереж застосовувалися граничні функції (1.3), (1.4) і (1.5). В даний час найчастіше використовується сігмоїдальна функція, яка обумовлена виразом

$$f(x)=\frac{1}{1+e^{-\beta x}}>0. \tag{1.6}$$

Відзначимо, що при $\beta\to\infty$ характеристика (1.6) прагне до граничної уніполярної функції (1.3). Як альтернативу застосовується функція гіперболічного тангенса

$$f(x)=\text{th}\left(\frac{\alpha x}{2}\right)=\frac{1-e^{-\alpha x}}{1+e^{-\alpha x}}>0. \tag{1.7}$$

У цьому випадку характеристика (1.7) прагне до граничної біполярної функції (1.4) при α→∞. Приклади функції f у моделі (1.2) показані на рис. 1.3 [9, с. 36–45].

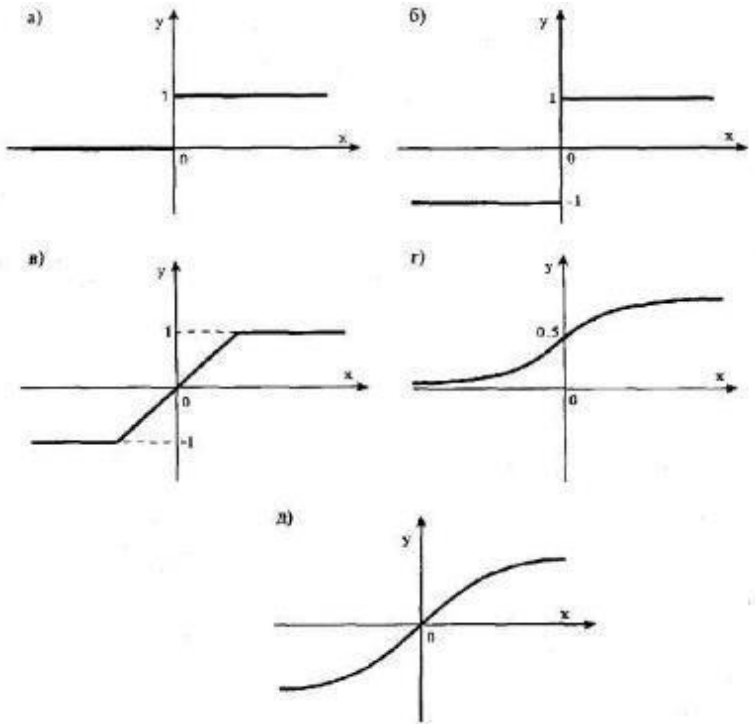


Рис. 1.3 – Приклади функції f

У наявних зараз програмних додатках штучні нерйони мають назву «елементи обробки» та у них є більше можливостей, чим простий штучний нейрон, описаний вище. На рис. 1.3 зображена детальна схема спрощеного штучного нейрона [9, с. 36–45].

Модифіковані входи передаються на функцію підсумовування, що переважно тільки підсумовує добутки. Можуть бути обрані будь-які операції: середнє арифметичне, найбільше, найменше, OR, AND, і т. п., котрі генерують відмінні один від одного значення. Багато додатків дають змогу формувати свої функції суматора з використанням різних підпрограм, котрі розробляються мовою високого рівня програмування. В деяких випадках підсумовування поскладнюється використанням функції активації, що дозволяє функції підсумовування діяти в часі. У кожному випадку проходження виходу функції підсумовування відбувається через передатну функцію на вихід за

використанням визначеного алгоритму. В існуючих нейромережах як передатні функції, як ми вже говорили, можуть бути використані сігмоїда, синус, гіперболічний тангенс і ін. Приклад того, як працює передатна функція показано на рис. 1.4.

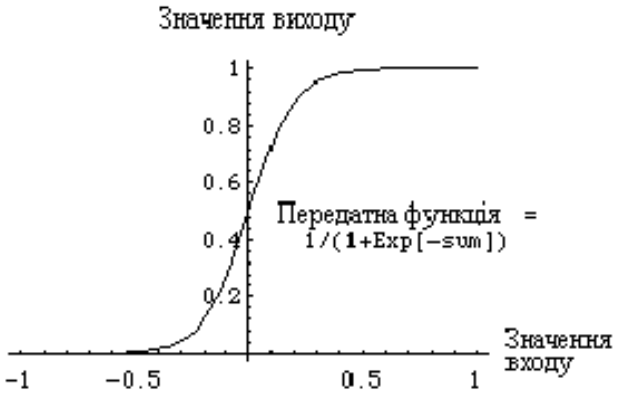


Рис. 1.4 – Сігмоїдна передатна функція

Усі штучні нейромережі конструюються з базового блоку – штучного нейрона. Існуючі розмаїтості і відмінності, є підставою для мистецтва талановитих розроблювачів при реалізації ефективних нейромереж [9, с. 36–45].

Штучна нейрона мережа (ШНМ) – це математична модель, та ще і пристрій з паралельних обчислювань, який має вигляд простих процесорів, називаємих штучними нейронами, котрі взаємодіють один з одним. Дана модель ілюструє собою сукупність методів, які відповідають за розпізнання образів чи дискриминатний аналіз. Здебільшого ці процесори дуже прості в порівнянні з процесорами у комп’ютерах звичайних користувачів. Процесори у мережах даного типу взаємодіють лише із сигналами, котрі поступають до нього та вихідними сигналами призначеними іншим процесорам. Однак незважаючи на прості процесори, можна побудувати керовану мережу для вирішення складних задач. Спроектовані мережі даного типу називають штучними нейронними мережами, усередині яких усе пов’язано на нейронах та зв’язках, що їх поєднують.

Нейромережі існуючі у даний момент представляють собою деяку кількість груп нейронів, у виді з’єднаних між собою шарів.

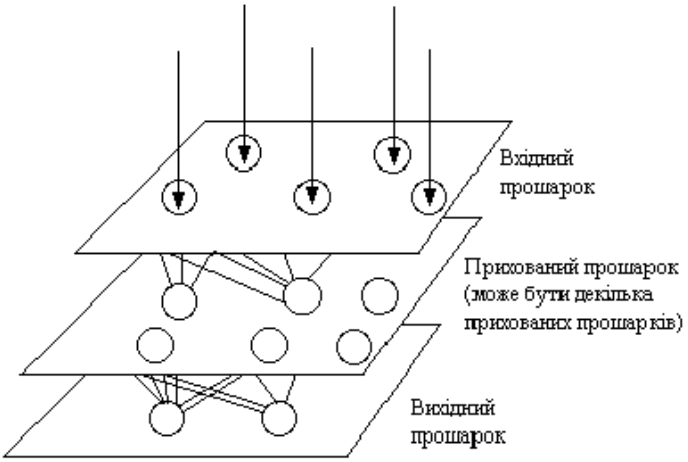


Рис. 1.5 – Діаграма простої нейронної мережі

На рис. 1.5 показана типова структура штучних нейромереж. Хоча існують мережі, що містять лише один шар, або навіть один елемент, більшість реалізацій використовують мережі, що містять як мінімум три типи шарів – вхідний, схований і вихідний. Шар вхідних нейронів одержує деякі значення з файлів або датчиків які формують вхідні дані. Перший шар пересилає інформацію безпосередньо в зовнішнє середовище, до вторинного комп’ютерного процесу, або до іншого пристрою. Між цими двома шарами може бути кілька схованих шарів, що містять багато різноманітно зв’язаних нейронів. Входи і виходи кожного зі схованих нейронів з’єднані з іншими нейронами. Напрямок зв’язку від одного нейрона до наступного вважається найважливішим моментом нейромереж. У великій кількості мереж будь який нейрон схованого шару одержує сигнали від усіх нейронів попереднього шару і звичайно від нейронів вхідного шару. Після виконання операцій над сигналами, нейрон передає свій вихід усім нейронам наступних шарів, забезпечуючи передачу сигналу вперед (feedforward) на вихід. При зворотному зв’язку, вихід нейронів шару направляється до нейронів попереднього шару (рис. 1.6).

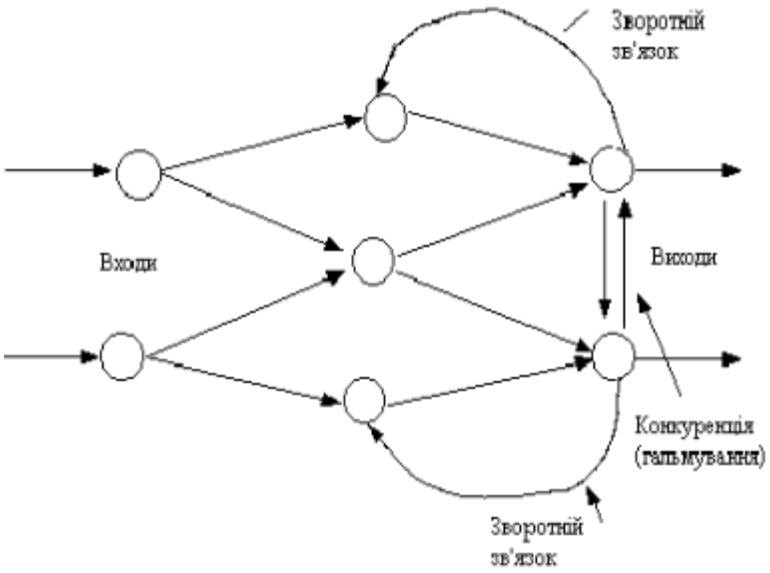


Рис. 1.6 – З’єднання між нейронами

Вид з’єднання між нейронами має великий вплив на роботу мережі. Більшість пакетів програмних реалізацій нейроних мереж дозволяють користувачеві додавати, віднімати і керувати з’єднаннями як завгодно. Постійно корегуємі параметри зв’язку можна зробити як збудливими так і гальмуючими. Відповідно, актуальним напрямком дослідження є реалізація нейронних мереж мовою описання апаратних засобів VHDL.

Мова програмування VHDL була розроблена для опису цифрових систем, також є підмножина мови – VHDL AMS (аналогових та змішаних сигналів), котра дає змогу описувати чисто аналогові, змішані та цифро-аналогові схеми. VHDL (англ. VHSIC (Very high speed integrated circuits) Hardware Description Language) – мова опису апаратури інтегральних схем. Мова проєктування VHDL є базовою мовою при розробці апаратури сучасних обчислювальних систем. Стандарти VHDL [42]:

– IEEE Std 1076–2002 IEEE Standard VHDL Language Reference Manual

– Sponsor: Design Automation Standards Committee of the IEEE Computer Society,

– Approved: 26 July 2002, American National Standards Institute,

– Approved: 21 March 2002, IEEE-SA Standards Board

– IEEE Std 1076–2008 IEEE Standard VHDL Language Reference Manual

– Approved: 26 September 2008 IEEE SA-Standards Board

– ГОСТ Р 50754–95 Мова опису апаратури цифрових систем VHDL. Опис мови.

Під час програмування на мові VHDL програміст має змогу створити паралельні оператори, які будуть працювати на віртуальній паралельній обчислювальній системі. На нижньому рівні даної моделі буде створена архітектура нейропроцессорного елемента, який буде реалізований лише одним паралельним оператором. Для верхнього рівня сформована система на міжпроцесорних зв’язках між великою кількість нейропроцесорних елементів, між якими будуть передаватись електричні сигшнали.

2. ОГЛЯД ІНСТРУМЕНТАРІЮ ДЛЯ ПРОЄКТУВАННЯ АДАПТИВНОЇ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ АВТОМАТИЧНОГО ДОГЛЯДУ ЗА РОСЛИНАМИ

2.1 Застосування мови програмування VHDL в якості основи проєктування системи автоматичного догляду за рослинами

Актуальним завданням є проєктування системи виконання операцій керування виконавчими механізмами по догляду за рослинами мовою описання апаратних засобів VHDL.

Беручи за основу мову VHDL'2008 була спроектована система верифікації цифрових функціональних блоків Open Source VHDL Verification Methodology (OC – BBM), котра дає змогу формувати функціональне покриття та керовану генерацію псевдовипадкових тестів. На сьогоднішній день на мові VHDL були спрограмовані описи мікропроцесорів ERC32 (SPARC V7) і LEON (SPARC V8). Вихідний код доступний під ліцензіями LGPL і GPL відповідно [42].

Модель обчислювача не може бути представлена програмістом як реальна багатопроцесорна система через високі апаратні затрати. Однак мова програмування VHDL була розроблена для реалізації у симуляторах дискретних систем. Стандарт IEEE Standard VHDL Language Reference Manual представляє у собі опис синтаксису мови програмування та порядок її використання у різних симуляторах. Вони найбільш часто бувають реалізовані на однопроцесорній електронній обчислювальній машині. Можемо розглянути структуру обчислювача з точки зору проектування симулятора. На рис. 2.1 умовно показана архітектура симулятора VHDL.

У даній моделі кожний віртуальний процесорний елемент реалізований як окремий обчислювальний процес. Кожен процес поділяє у часі один і той же ресурс – центральний процесорний елемент мікропроцесора ПЕОМ.

Обчислювальні процеси можна поділити на три підгрупи: процеси, які неактивні, черга процесів, що виконались та група процесів, які на даний

момент виконуються системою. Більшість обчислювальних процесів відноситься до числа тих, що сплять, тобто до обчислювальних процесів, для виконання яких немає готових вхідних даних. Наприклад, такий обчислювальний процес очікує який-небудь сигнал, до якого у нього підвищена чутливість свого оператора wait. При отриманні сигналу такого типу даний обчислювальний процес переноситься в чергу готових до виконання процесу. Один з готових обчислювальних процесів, який був виділений серед інших процесів, надсилається в кеш-ОЗП мікропроцесора, щоб почати своє виконання. Обчислювальний процес, що виконується, виконує свої оператори відповідно до семантики функціонування віртуальних процесорних елементів на ресурсах центрального процесорного елемента і використанням алгоритмів бібліотек для необхідних процедур. Обчислювальний процес зупиняється коли він досягає оператора wait. Вироблені ним сигнали запам'ятовуються та пересилаються іншим обчислювальним процесам, які є приймачами цих сигналів. Потім даний обчислювальний процес пересилається в чергу процесів, що сплять [9, с. 45–48].

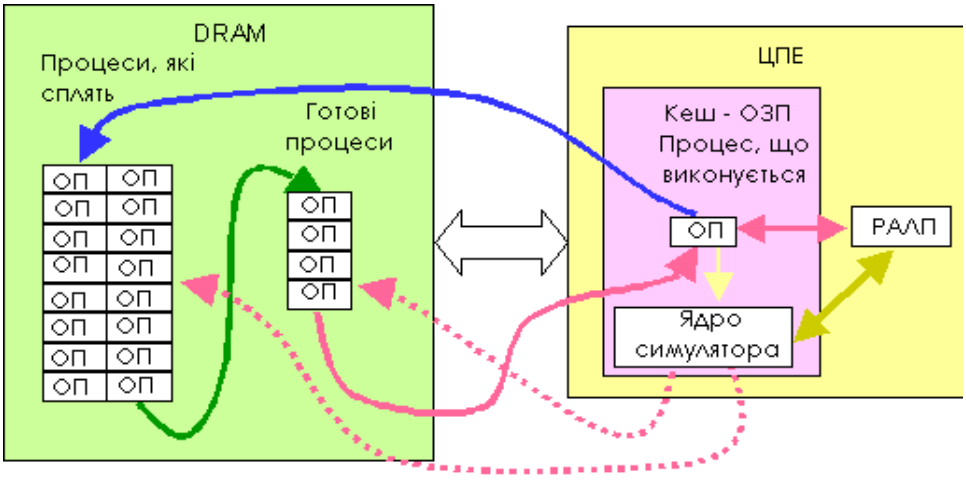


Рис. 2.1 – Архітектура симулятора VHDL [9, с. 45–48]

Один особливий обчислювальний процес, який знаходиться на етапі виконання виступає у ролі інформатора для наступних процесів. Його прийнято називати ядром симулятора, а також він формує черги обчислювальних процесів, які сплять, і готових до виконання, а також вибирає обчислювальний

процес для виконання. Він також розсилає сигнали від процесу, що виконується, до інших процесів. Крім того, він веде лічильник часу моделювання системи, виступає комутатором з консоллю, а також обробляє запити до пам’яті, як дискової, так і глобальних змінних.

2.2 Огляд можливостей логічних процесорів для реалізації інтелектуальної системи

Для моделювання програміст створює проект – каталог з файлами VHDL, що має назву проекту. Після компіляції в проекті створюється бібліотека проекту, що має назву проекту й містить всі скомпільовані об’єкти проекту. Після запуску програми на моделювання спершу виконується зв’язування об’єктів проекту і присвоєння початкових значень змінним та сигналам (elaboration) [30, с. 112–115].

Наступним етапом являється запуск особистого моделювання. Під час цього етапу окремі сигнали, які вибере програміст на власний розсуд, будуть записані в область пам’яті та при необхідності мають змогу на відображення у вигляді таблиць або графіків.

Для збереження семантики під час формування багатої кількості паралельних процесів, для кожного етапу моделювання, окрім першого, симулятором будуть використовуватись такі кроки:

Для кожного зупиненого обчислювального процесу, який отримав відповідну подію у попередньому циклі та яка очікує оператор запинки wait, відправляється на виконання до мікропроцесора. Інші процеси, які знаходились у етапі очікування затримки перевіряється час цієї затримки, у разі закінчення – процес відправляється на виконання.

Будь який виконавчий процес працює до зупинки при своєму операторі wait. Під час цього розраховуються значення для сигналів у відповідних операторах. Розраховується кожний момент нове значення для поточного часу, котрий представляє собою момент часу наближений до запуску будь-якого ОП. Нове значення часу буде записано у лічильник Tc = Tb.

Якщо будь якому сигналу було прораховане нове значення, то сигнал прийме саме його. Усі зміни сигналів верифікуються на приналежність до подій для процесів.

Якщо при циклі виконання не формувався новий процес по закінченню затримки або не було ніяких активацій від сигналів, то лічильник часу зберігається без змін представлений у фемта- або пікосекундах. У такому циклі прийнято вважати, що до часу моделювання додається дельта-затримка.

Процес моделювання володіє такою особливістю, що серія, котра містить в собі від 0 до деякої кількості циклів з дельта-затримками автоматично замінюється на один цикл з певною затримкою. Зазвичай серія містить у собі 0 – 10 таких циклів. При моделюванні комбінаційних схем з N рівнів, які представлені завдяки процесам, котрі не містять у собі затримок, то у результаті серія міситимете у собі N циклів. Але бувають такі ситуації коли у програмі допущені помилки і вона сама себе зациклює і циклів у серії становиться набагато більше, однак симулятор Active HDL призупиняє виконання, при досягненні циклів більше 1000.

Якщо у системі присутні декілька процесів, котрі використовують одну глобальну змінну. У такий ситуації черговість обміну доступу до змінної не може контролюватись програмістом, адже невідомо який з процесів перехопить першим доступ до цієї змінної. Для уникнення даної ситуації використовується підхід з позначенням процесів вкладеними, з використанням в програмі ключового слова postponed перед процесом.

Наприклад, підчас синтезу будь-яких схем точного множення двійкових чисел, першою чергою аналізується матриця, яка виступає нульовим шаром для схеми множення. Найпопулярнішими пристроями синтезування для виконання логічних операцій множення, які були реалізовані таким методом, є матричні пристрої множення з горизонтальним та діагональним розповсюдженням переносу [36].

Проведення параметричної оптимізації матричних пристроїв множення методом аналізу характеристик складності H-моделі алгоритму дає змогу

отримати оптимізовані пристрої множення з горизонтальним та діагональним розповсюдженням переносу.

Матричний пристрій множення з використанням горизонтального переносу можна реалізувати у вигляді матриці, яка в свою чергу може бути представлена у вигляді з'єднаних між собою елементів. Управління даною схемою обмежується завдяки подачі до входу двох n-розрядних співмножників [32].

Фрагмент матриці оптимізованого пристрою множення з горизонтальним переносом зображено на рисунку (рис. 2.2). Схема відтворена з відсутністю вхідних значень у нульовому шарі, та матриця кон'юнкцій відтворена у комірках Гілда (Г) та комірках напівсуматора (Н). Складність такої схеми можна представити за допомогою таких характеристик: структурна S=14, часова L=3n – 6, апаратна $A=(n-1)^2$ та програмна складність P=0 [35].

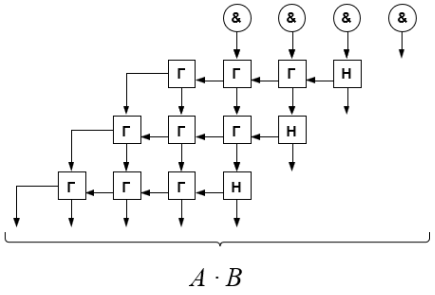


Рис. 2.2 – Пристрій множення з горизонтальним переносом

Матричному пристрою множення з горизонтальним переносом притаманне зменшення апаратної та часової характеристики складності, значення структурної складності зросло з появою неоднорідності елементів матриці.

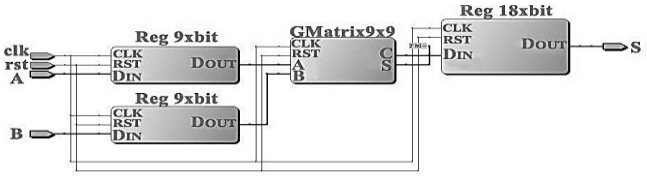


Рис. 2.3 – Структура VHDL моделі матричного пристрою множення з горизонтальним переносом [32]

Наприклад, результати моделювання в системі Quartus II VHDL-моделі (рис. 2.3) матричного пристрою множення з горизонтальним переносом відобразили, потреби у 154 логічних елементах для реалізації. При реалізації такого пристрою на ПЛІС сімейства Cyclone IV GX, то один такий пристрій множення займатиме менше одного відсотка кристалу. Час відклику може приблизитись до 11.9 нс (~84MHz). На рисунку (рис. 2.4) наведено часову діаграму роботи оптимізованого матричного пристрою множення з горизонтальним розповсюдженням переносу з випадковим набором вхідних даних.

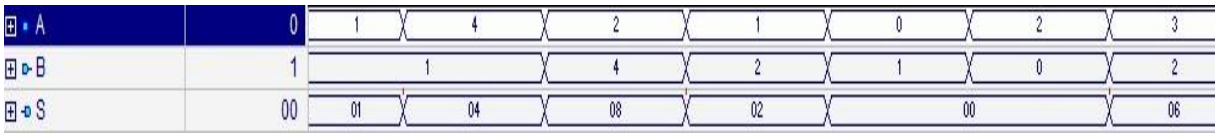


Рис. 2.4 – Часова діаграма роботи матричного пристрою множення з горизонтальним переносом

Матричному пристрою множення з діагональним розповсюдженням переносу характерна неоднорідну структуру, котра може складись з матриці комірок Гілда та елементів кон’юнкції, до якої приєднаний n розрядний суматор. Складність на апаратні витрати прагне наблизитись до 0. Апаратна складність є більшою $A=n^2-n$, це пояснюється появою додаткового nрозрядного суматора. Часова - L=2n – 2, структурна - S=21.1 [33]. На рисунку (рис. 2.5) зображена структурна схема пристрою множенняз діагональним розповсюдженням переносу [32].

Перевагою даного пристрою множенняє зменшена часова складність, приблизно в 1,5 рази у порівнянні з горизонтальним розповсюдженням переносу. Таке значення часової складності отримано завдяки методу по зростанню структурної та апаратної характеристик складності розробляємого пристрою, що дозволяє применшити максимальний критичний шлях розповсюдження сигналу [33]. Даний варіант пристрою множення (рис. 2.4) варто удосконалити зменшенням значення часової характеристики складності

методом конвеєрної організації схеми. Наступним кроком є реалізація VHDL-моделі конвеєрного матричного пристрою множення та проведення емуляції розробляємої моделі на ПЛІС, щоб отримати чисельні характеристики для пристрою.

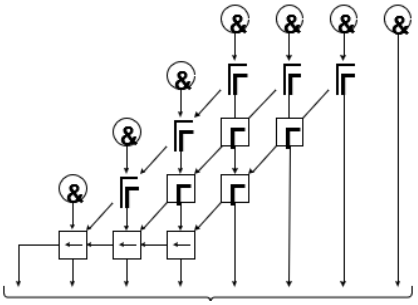


Рис. 2.5 – Фрагмент матриці пристрою множення з діагональним розповсюдженням переносу

Двоступеневий конвеєрний варіант матричного пристрою множення з діагональним переносом зображений на рисунку (рис. 2.6). Розподіл схеми проводиться за допомогою двох регістрів по горизонталі між матрицею комірок Гілда і елементів кон’юнкції, та додатковим nрозрядним суматором. Для кожної частини розділеної схеми часова складність однакова L=n – 1. Апаратна та структурна характеристики складності такого пристрою множення зросли, за рахунок доданих конвеєрних регістрів A=n2+4n – 2, та S=25.3.

Використовуючи метод реалізації структури за допомогою мови програмування VHDL, був проведений аналіз моделі конвеєрного пристрою множення з діагональним переносом. Такий пристрій займатиме приблизно однин відсоток від усього кристалу, якщо точніше, то 197 простих логічних елементів. А відгук сходинки такого пристрою множення на кристалі буде рівним 7.06 нс (~141,64 MHz).

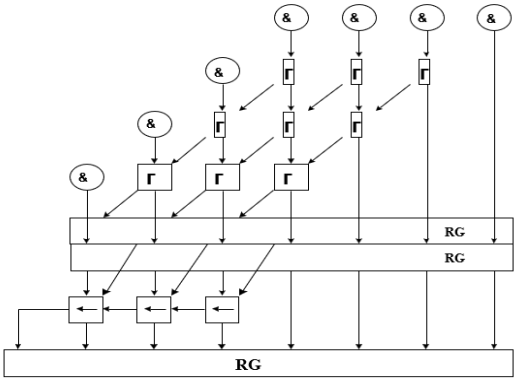


Рис. 2.6 – Конвеєрний матричний ПМ з діагональним розповсюдженням переносу

Провівши аналіз характеристик складності конвеєрного варіанту матричного пристрою множення з діагональним розповсюдженням переносу був зроблений висновок, що завдяки зростанню апаратної та структурної ознак системи можна досягти поліпшення часової характеристики складності.

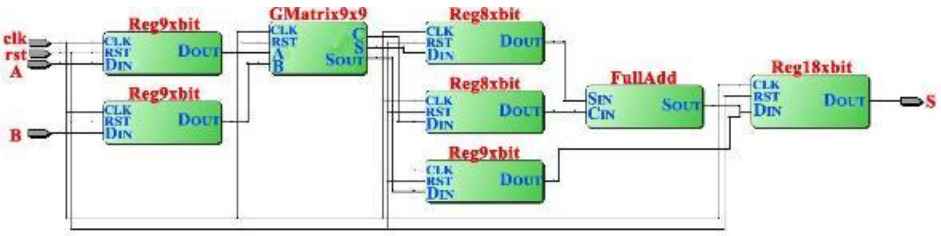


Рис. 2.7 – Структура VHDL моделі конвеєрного пристрою множення з горизонтальним переносом

На рисунку (рис. 2.8) наведено часову діаграму роботи конвеєрного варіанту пристрою множення з набором випадкових вхідних даних.

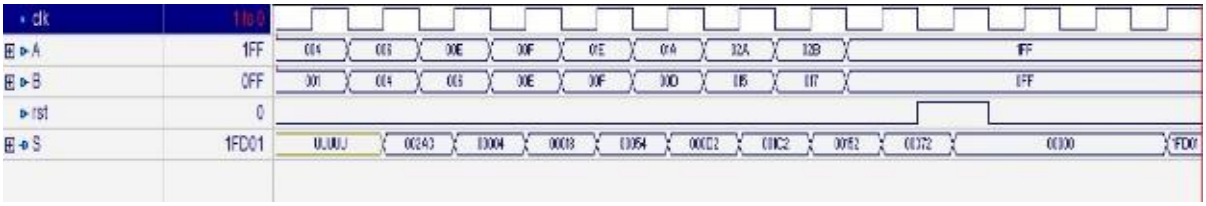


Рис. 2.8 – Часова діаграма роботи конвеєрного матричного пристрою множення з діагональним переносом

Проаналізувавши діаграму (рис. 2.8) можна зробити висновок, що результат від операції множення формується вже через 2 такти після подання даних на вхід, що є свідченням роботи двоступеневого конвеєра.

Реалізація VHDL моделі кожного з розглянутих пристроїв множення дає можливість промоделювати їх на ПЛІС використовуючи ПЗ, в результаті чого можна отримати числові характеристики їх розміру на кристалі, і час формування результатів від операцій множення на вихідні порти системи. Після моделювання можна зробити висновок, що пристрої множення мають відмінності за ступенями складності та часом праці на кристалі. Обирати той, чи інший пристрій залежить виключно від мети застосування у СКС.

Отже, при подальшому програмуванні VHDL-програми добре брати до уваги наступні моменти:

– і малий процес завжди буде потребувати декілька кілобайт пам’яті, а об’єм динамічної пам’яті обмежує по кількості процеси у вихідній програмі;

– зміна одного процесу іншим являє собою зміну контексту в роботі центрального процесорного елемента, що може тривати кілька тисяч його тактів. Отже, чим менше буде використано паралельних операторів і процесів, тим більш швидше буде відбуватись процес моделювання;

– найбільшу частину часу займає виконання обчислювального процесу ядра симулятора, що зростає не лінійно зі збільшенням числа обчислювальних процесів та кількість сигналів у чергах чутливості операторів wait. Звідси можна зробити висновок, якщо у скомпільованій програмі буде менше процесів, через це і списків чутливості – то і виконання програми буде швидшим.

Алгоритм роботи пристрою зображує його роботу на рівні функцій роботи схеми. Фактичної кінцевої точки роботи схеми немає, оскільки робота процесу не припиняється, тобто схема постійно аналізує датчики та блоки та виводить інформацію у вигляді світлової та звукової індикації. Кінцевою точкою роботи пристрою можна назвати ту точку, коли пристрій перестає функціонувати взагалі, тобто виходить з ладу. Система аналізує сигнали та

виконує функції відносно запрограмованих реакцій на стан датчиків, враховуючи дозвіл на виконання результуючої операції. Відповідно, кінцевим результатом проєктування повинен бути виконаний логічний нейропроцесор для виконання логічних операцій по керуванню електроприводом виконавчого механізму системи догляду за рослинами.

Процесор – являє собою центральний пристрій електронної машини, задачею котрого є виконання будь яких операцій використаних у програмній реалізації, також процесор відповідає на управління обчисленнями і координацію пристроїв у системі. В розроблювальних системах існує варіант використання паралельно працюючих нейропроцесорів, такі системи називають багатопроцесорними [15, с. 138–147].

Серед основних параметрів процесора виділяють його розрядність та швидкодію. Під поняттям швидкодії мають на увазі кількість операцій, котру здатен виконати процесор за секунду часу, а розрядність – це кількісна характеристика об’єму інформації, яку процесор може обробити за операцію, наприклад 32 розрядний процесор здатний обробити 32 біти за одну операцію.

У моделях нейропроцесорів визначають деякі пристрої:

Пристрій управління (ПУ). Даний прилад формує сигнали, які необхідні іншим пристроям для коректної праці у нейропроцессорі, а також відповідає за управління та взаємодію процесів з шиною пам’яті.

Арифметико-логічний пристрій (АЛП, ALU). Таку назву дають пристрою, який використовується для виконання цілочислених операцій. А саме він відповідає за логічні, арифметичні та операції по зсуву. Саме ці операції найчастіше використовуються в програмних реалізаціях у великій кількості програм. Деякі нейропроцесори можуть містити в собі декілька АЛП.

AGU (Address Generation Unit) – пристрій для генерації адрес. Основними задачами даного пристрою полягають у тому, що завдяки йому відбувається коректне завантаження і збереження даних, а також їх адресація. При використанні масивів даних у програмі, до роботи вступає AGU з його непрямою адресацією, однак існує ще і абсолютна адресація, але її використання можна зустріти вкрай рідко.

Математичний співпроцесор (FPU). У нейропроцесорі може використовуватись декілька FPU і можливості кожного з них укладаються в тому, що кожен зданий обробляти, якнайменше, по одній операції з плаваючою точкою, не беручи до уваги чим зараз зайняти АЛП. Завдяки методу по конвертації обробки даних кожний співпроцесор може опрацьовувати кілька одночасних операцій. Немалим плюсом математичного співпроцесора є те, що він містить у собі певні константи, які прискорюють його швидкодію на операціями. Ще більша продуктивність системи досягається тим, що співпроцесор працює незалежно і паралельно до центрального нейропроцесору. Черговість команд які будуть виконані напряму залежить від того, як вони поступають до співпроцесора. Через усі можливості математичного співпроцесора, він дає змогу обробляти з високою точністю логарифмічні, арифметичні та тригонометричні операції за вкрай малий час. [15, с. 138–147].

Дешифратор інструкцій (команд). Даний пристрій проаналізовує інструкції на предмет адрес і операндів, для розміщення в них результатів операцій. Після чого надсилає повідомлення до інших пристроїв системи про необхідні дії, щоб виконати поставлену інструкцію. Дешифратор інструкцій або команд, завдяки своїй структурі, дозволяє обробку декількох інструкцій в один і той же час, щоб дозволити використання інших пристроїв у системі.

Кеш-пам’ять. Важлива високошвидкісна пам’ять для нейропроцесора. Кеш призначений як буфер досягнення найшвидшого обміну даними для оперативної пам’яті та нейропроцесора, а також для підтримки дешифратора шляхом збереження копії інструкції та даних, які нещодавно виконав процесор. Отримання даних з кеш пам’яті отримується без будь-якої взаємодії з основною пам’яті.

Шина – це канал для пересилки даних, призначення якого у тому, щоб об’єднувати інші пристрої до купи. Уся інформація яка передається завдяки шині представляється у вигляді групи бітів. До шини можуть приєднуватись одразу багато інших пристроїв, котрі потребують у даних для роботи. Для кожного біта може бути виділена ціла шина і в такому випадку дані будуть передаватись паралельно, або дані можуть передаватись по одні шині

послідовно і таким чином шина даних може бути або паралельною, або послідовною. [15, с. 138–147].

Типи шин:

– шина даних. Служить для пересилки даних між регістрами нейропроцесора та АЛП;

– шина адрес. Даний тип шини спроектований таким чином, щоб для отримання блоку пам’яті або пристрою – на шині повинна бути встановлена конкретна адреса з цільовим пристроєм.

– шина управління. Ця шина використовує сигнали управління усередині себе, завдяки котрим указується напрям для передачі даних або до нейропроцесора або від нього.

Регістри – це внутрішня пам’ять нейропроцесора. Це спеціально спроектовані додаткові елементи пам’яті, а також можуть бути представлені, як внутрішні елементи мікропроцесора. Дані у регістрах зберігаються тимчасово, ці дані можуть бути числами або командами, основною метою яких є полегшення опрацювання операцій. Регістр спроектований таким чином, що він складається з електронних мікросхем (тригерів), які можуть зберігати всередині себе один двійковий розряд. А увесь регістр може бути представлений, як сукупність цих мікросхем під одним елементом управління.

Існують спеціальні регістри з певними назвами, наприклад:

– лічильник команд – регістр, який містить у собі певні адреси наступної команди; основною ціллю лічильника команд є вибірка команд в автоматичному режимі з послідовних комірок пам’яті;

– регістр команд – регістр ПУ основним принципом якого є збереження команди на певний період часу, котрий буде необхідний для її завершення. Дані середини нього – це код операцій або дані з адресами операндів;

– показник на вершину стека – використовується для роботи стеку [16, с. 136–149].

Процесор слугує центром усієї системи і керує усіма операціями, що відбуваються у системі. Основна мета полягає у послідовній реалізації операцій вибірки-дешифрації-виконання.

З цього можна виділити основні функції нейропроцесора:

– обирає команди для програми із основної пам’яті;

– дешифрацію команд;

– опрацювання логічних, арифметичних і інших операцій, закладених всередині команда;

– управління пересилкою інформації між регістрами і основною пам’яттю, між пристроями введення / виведення;

– обробка сигналів, котрі поступають від пристроїв введення/виведення, а саме обробку і виконання переривань на цих пристроях;

– керування роботи важливих вузлів логічного нейропроцесора.

Структура створеного нейропроцесорного елемента зображена в додатку А.

Створюючи власну структуру нейропроцесорного елемента вирішено, що кращим рішенням щодо розрядності процесора є 8 біт, розрядність адрес також 8 біт. Відповідно до процесора і адрес варто обрати розрядність регістрів також 8 біт.

Команди – одноопернадні. А розмір коду самої операції буде фіксований, і буде 8 біт. Звідси структуру команди можна представити таким чином:

15 8	7 2	1 0
Операнд	Код операції	Rm

Після дешифрації команди у разі потреби буде обиратись операнд. Щоб спростити сприйняття, вважається, що операнд завжди є у команді, однак він може просто не використовуватись. В нашій проектованій системі операндом буде константа, ціль якою буде лише порівняння. Rm (біти 0 та 1) – це номер регістра, саме значення цього регістра буде порівнюватись із нашою константою у операнді. Припустимо що 2-7 біти нашої команди приймають

значення: 000001b. В нашій системі 4 регістри і з цього можна зробити висновок, що команди можуть мати вигляд:

```
00000100  SBR R0, B
00000101  SBR R1, B
00000110  SBR R2, B
00000111  SBR R3, B
```

Через малу кількість портів введення/виведення обрана так мало регістрів загального призначення (R0..R3).

Для нашою системи додається пристрій приросту у структуру нейропроцесора. Основна мета якого у тому, щоб збільшити значення регістру на 1 за один такт, і щоб не використовувати шину даних.

На шині даних розташовуються усі структурні елементи системи нейропроцесора, окрім пристроїв приросту та управління, бо пристрій приросту має можливість отримувати дані використовуючи тільки регістр IR, а виведення інформації відбувається тільки через IRoff. Через нестачу портів введення/виведення в пристрої управління у нашій системі, немає змоги підключити шину даних.

Щоб скоротити час на пересилках адрес для наступних команд – було вирішено з’єднати регістри PC та MAR між собою. Завдяки цій маніпуляції можна зберегти 1 такт головного процесору. Також, можна пересилати адреси команд під час їх виконання, через те, що така схема підключення не потребує у цій схемі шини даних, через це економиться ще один такт нейропроцесора.

У структурній схемі лініями синхронізації та керування, завдяки яким пристрій управління відправляє електричні сигнали до інших елементів системи, були проігноровані.

В Додатку А показана розроблена архітектура нейропроцесора. У цій архітектурі АЛП та регістр з’єднуються по одній загальній шині, однак це не зовнішня шина, а внутрішня шина самого нейропроцесора.

Внутрішня шина нейропроцесора використовуючи регістр даних пам’яті, адреси пам’яті, MAR та MDR дозволяє з’єднати лінію даних та адреси шини. Регістр MDR реалізований таким чином, що у ньому присутні два входи та стільки ж виходів. Використання даних може бути з шини пам’яті чи

внутрішньої шини процесора. Дані із цього регістра можна помістити до будь якою під'єднаної шини. Регістр MAR з'єднує собою через входи внутрішню шину та регістр PC із зовнішньою шиною даних. Керуючі лінії поєднані з керуючим блоком та дешифратором команд.

Кожна операція у нейропроцесорі виконуються протягом періодів часу, що обумовлені тактовим сигналом нейропроцесора, або часом, який виділяє на це пристрій управління, який у нас реалізований у вигляді мікроконтролера. Операції пересилання, що керуються сигналами, можуть активізуватись лише на початку такту.

Пристрій управління реалізований у вигляді мікроконтролера. Мікроконтролер можна взяти будь-який, єдина вимога до нього – наявність п'яти портів введення / виведення. Чотири порти мають з'єднувати усі лінії елементів нейропроцесора. Останній порт призначений для введення/виведення даних у 8-бітному вигляді. Також цей порт має змогу пересилки даних із байтами регістру IR. Порти та їх застосування зображено на рис 2.9.

Розшифровка позначень:

– MFC – лінія для отримання сигналу готовності пам'яті.

– Xin – лінія, що надає дозвіл для входу даних до регістру X з шини.

– Xout – лінія, що надає дозвіл на видачу даних з регістру X на шину.

– Xin_mc – лінія, яка надає дозвіл на вхід даних в регістр X з порта нейропроцесора.

– Xout_mc – лінія дозволу для видачі даних з регістру X на порт процесору.

– CLK – лінія синхронізації сигналів

– PCinc – лінія зарезервована для активації пристрою приросту

– MRead – лінія читання з пам'яті

– PC_to_MAR – лінія, що надає дозвіл по пересилці даних з регістра PC в регістр MAR.

– ALUe0.. ALUe0 – лінії видачі АЛП коду операції

– ALUm – лінія вибору режиму роботи АЛП (арифметичний / логічний)

– IR – порт, який має можливість обмінюватись даними регістром IR.

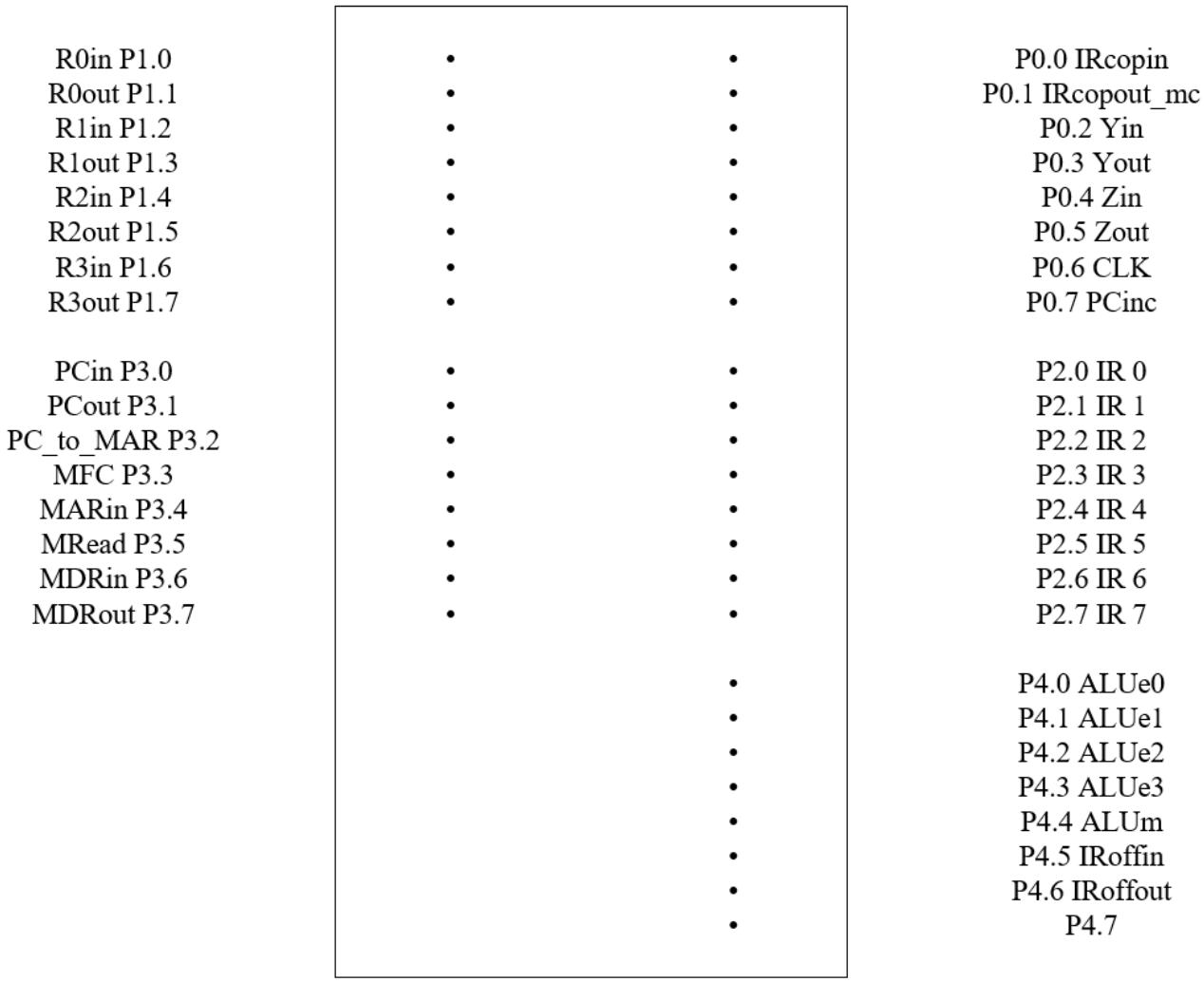


Рис. 2.9 – Призначення портів пристрою управління

Для виконання програми логічний нейропроцесор обирає команди, які знаходяться у пам’яті, по черзі та виконує їх інструкції. Вибір команд відбувається до припинення через команди переходу або розгалуження які можуть трапитись у пам’яті. Лічильник команд використовується для відстеження адреси команди, яка підлягає опрацюванню на цей момент часу. Після прийняття команди до опрацювання регістр PC зміню своє значення на команду, яка йде наступною у пам’яті. Команда розгалуження має змогу втрутитись у чергу та завантажити у регістр PC іншу адресу. Загальний алгоритм роботи логічного нейропроцесора зображено в Додатку Б.

2.3. Математичний аналіз операцій нейропроцесора та розробка алгоритму його функціонування

Алгебраїчні операції в аналізі роботи процесора приводять до низки важливих конструкцій в означених галузях, наприклад, скінчених проєктивних площин, кодів Ріда-Мюлера і кодів Гоппа. Засновані на теорії скінчених полів алгоритми перевірки на простоту і факторизації цілих чисел відіграють важливу роль у сучасній прикладній теорії чисел. Проаналізуємо основу математичної функціональності нейропроцесора.

Для будь-якого простого числа p, кільце залишків (mod p) – це скінчене поле з p елементів, яке позначається GF (p)=Fp=Z / pZ. Елементи цього поля можуть бути представлені цілими числами 0, 1 , … , p–1, які додаються і множаться «за модулем p. Будь-яке скінчене поле містить p^n елементів і однозначно задається своєю характеристикою p і степенем n [43].

Таким полем називають множину елементів, на якій визначено дві операції. Перша множина цих елементів позначається, як додавання і відзначається, як $a+b$, а друга – це множення і має позначку $a \cdot b$, незважаючи на те що ці операції не завжди можуть бути операціями додавання і множення чисел. Щоб множина елементів, на якій задаються операції, представляла з себе поле, потрібно, щоб кожна операція виконувала усі групові аксіоми, а саме комутативність ($a+b = b+a$ і $ab = ba$), асоціативність ($a+ (b+c) = (a+b) +c$ і $a (bc) = (ab) c$), а також виконувала дистрибутивний закон, тобто для трьох будь-яких елементів поля a, b, c була справедлива рівність $a (b+c) = ab+ac$ і $(b+c) a = ba+ca$.

Зазначмо, що групові властивості для операції множення справедливі для всіх ненульових елементів поля. Поля з скінченним числом елементів q називають полями Галуа на ім’я їх першого дослідника Евариста Галуа і позначають $GF (q)$ [9]. Число елементів поля q прийнято називати порядком поля.

Найменша кількість елементів що будуть відтворені полем сягає двох. Це поле повинне мати 2 одиничні елементи: 0 призначений для операцій додавань, а 1 необхідне для операцій множення. Це поле позначається, як *GF (2)*, або називається двійковим.

У залежності від q поля поділяють на прості або розширені. Поле є простим, якщо q – просте число. Для позначення простих чисел використовуватимемо символ p. Просте поле утворюють числа за модулем p: $0, 1, 2., p - 1$, а операції додавання і множення виконуються за модулем p. Якщо ж поле утворене з q^m елементів, то таке поле прийнято вважати розширеним або поля ступеня m над $GF(p)$. Кожне таке поле містить у собі p^m елементів і позначається GF (p^m). Тепер буде розглядатись поле $GF(2^m)$. Будь-яке скінченне поле $GF(2^m)$ є m-вимірним вектором над полем $GF(2)$. Многочлен $f(t)$ ступеня m над полем $GF(2)$ є многочлен вигляду $f(t) = t_m + f^m - {}^1t_{m-1} + \ldots + f_0$, де коефіцієнти многочлена f_i належать $GF(2)$. Операції з цими многочленами опрацьовуються як і над звичайними многочленами, однак коефіцієнти розраховуються у полі $GF(2)$ [43].

Многочлен $f(t)$ ненульового ступеня називається незвідним над полем $GF(2^m)$, якщо він може поділитись сам на себе, а також многочлени нульового ступеня без залишку [34]. Елемент x скінченного поля $GF(2^m)$ називається коренем многочлена $f(t)$, якщо $f(x)=0$. Якщо x – корінь незвідного многочлена $p(x)$ ступеня m, то елементи $(x^m - {}^1, \ldots, 1)$ утворюють базис скінченного поля $GF(2^m)$ як векторного простору над полем $GF(2)$. Зазвичай, такий базис прийнято називати поліноміальним. Многочлен, коренем якого є примітивний елемент, вважається примітивним многочленом. Таким чином, якщо $P(x)$ вибрати примітивний многочлен ступеня m, незвідний над полем $GF(2)$, то у результаті буде отримане поле $GF(2^m)$ зі всіх 2^m двійкових послідовностей довжини m. Поліноміальний базис задається примітивним многочленом.

У поліноміальному базисі прийнято відображати елементи скінченного поля у вигляді многочленів ступінь яких не переважає $m - 1$ або, те, що має як свій еквівалент двійкові рядки довжиною m, та котрі складаються з коефіцієнтів для таких многочленів. Для операцій множення і додавання, які виконуються у

скінченому полі є операціями над многочленами ступеня не більше $m - 1$ зі зведенням результату за потреби за модулем примітивного многочлена.

Операція по додаванню 2 елементів опрацьовується як додавання за модулем 2 до многочленів двійкових рядків. Множення відбувається аналогічно і відбувається множення відповідних многочленів зі зведенням до примітивного многочлену.

Нормальний базис для поля вигляду $GF(2^m)$ можна представити як:

$$B=\{\theta,\theta^2,\theta^{2^2},\ldots,\theta^{2^{m-1}}\} \quad (2.1)$$

Дані елементи також можна записати як двійкові рядки $(a_0\ a_1\ a_2\ \ldots\ a_{m-1})$

$$a0\theta+a1\theta2+a2\theta22+\ldots+am-1\theta2m-1 \quad (2.2)$$

У залежності від того, який параметр T, можна розрізнити типи нормального базису і нормальні поліноми:

При T=1: $p(x) = t^m + t^{m-1} + \ldots + t^2 + t + 1$;

при T=2: $p_0(t) = 1$, $p_1(t) = t + 1$, $p_{i+1}(t) = tp_i(t) + p_{i-1}(t)$, $i = 1,\ldots, m$.

Суматор за модулем обчислює суму $Z=(X+Y)_q$, де число Z дорівнює залишку від ділення суми X+Y на число q. Числа Z, X, Y, і q зображені у вигляді двійкових рядків та мають n - розрядність. Потрібно сформувати суматор за модулем q для усіх значень n.

Наприклад роздивимось суму, яка має вигляд:

$$S=(X+Y) + (2n - q), \quad (2.3)$$

де S має n+1 двійкових розрядів. З цього отримуємо, що сума S може приймати значення $S<2^n$ і $S>=2^n$ залежно від значень X і Y. Якщо сума $S<2^n$, то (n+1) розряд числа S дорівнює 0 і з співвідношення (2.1) звідси можемо отримати, що X+Y<q, а отже, Z=X+Y.

Якщо ж сума $S>=2^n$, то (n+1) розряд числа S дорівнює 1 і з співвідношення (2.3) випливає, що X+Y>=q, а отже, Z=X+Y-q. Таким чином може існувати таке співвідношення:

$$Z=(X+Y)_q=\begin{cases} X+Y, \text{ якщо } X+Y<q \\ X+Y-q, \text{ якщо } X+Y\geq q \end{cases} \quad (2.4)$$

На основі співвідношення (2.4) можна побудувати суматора за модулем по схемі, де q – будь-яке просте число. На рис. 2.10 показана схема суматора для n-розрядного числа q. Суматор D1 виконує обчислення для суми чисел X та Y, суматор D2 віднімає від суми X+Y значення q, оскільки 2^n-q – доповнення числа q до 2^n. Мультиплексор подає на вихід суму $Z=(X+Y)_q$ залежно від розряду s_{n+1}.

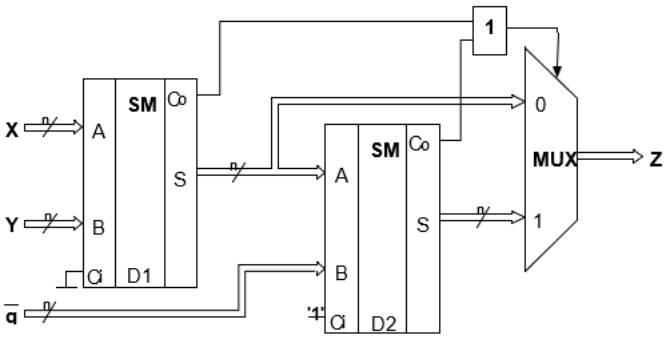
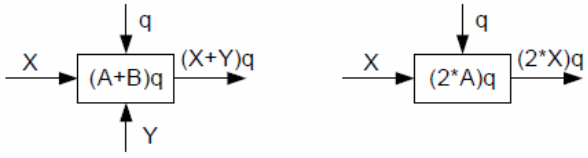


Рис. 2.10 – n-розрядний суматор за модулем

Даний розроблений суматор можливо спрограмувати завдяки мові VHDL для наступного синтезу на ПЛІС.

Умовні позначення, які варто використовувати для структурних схем суматорів зображені на рис. 2.11.



а *б*

Рис. 2.11 – Умовне графічне позначення суматора за модулем q і перемножувача за модулем [34]

Для добутку чисел X · Y існує співвідношення

$$X*Y=\sum_{p=1}^{n} y_p*X*2^{p-1} \qquad (2.5)$$

де y_p=0 або 1. Можна зробити висновок, що побудова перемножувача за модулем q потребує синтезу типової схеми, яка виконує операцію $(2 \cdot X)_q$ – перемноження на 2 за модулем q. Отже, оскільки $(X \cdot 2^{j+1})_q=(2*(X \cdot 2^j)_q)_q$, значення $(X \cdot 2^{p-1})_q$ можуть бути отримані послідовним використанням перемножувачів на 2 за модулем. Правило побудови схеми такого перемножувача отримуємо з співвідношень (2.3) і (2.4), якщо візьмемо X=Y і S=2 · X+ (2^n-q). На рис. 2.11 показано умовне позначення перемножувача на 2 за модулем. На рис. 2.13 показано схема перемножувача на 2 за модулем для n-розрядного q. Перемноження числа X на 2 досягається зсувом розрядів числа X на один розряд відносно входів суматора, з цього можна зробити висновок що потрібен лише один суматор. А на виході мультиплексора формується величина $Z=(2 \cdot X)_q$. [34]

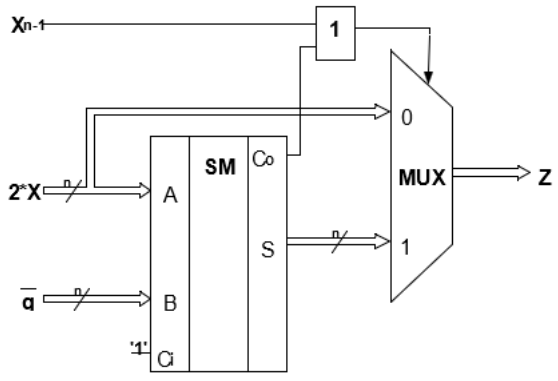


Рис. 2.12 – n-розрядний перемножувач на 2 за модулем

На рис. 2.12 відображена схема перемножувача по модулю, котра може обчислити

$$Z=(X*Y)_q=(\sum_{p=1}^{n} y_p*X*2^{p-1})q \qquad (2.6)$$

тут q і Z – n-розрядні двійкові числа. Тут числа q і X являють собою n-розрядні дані, а вузол & – це сукупність логічних елементів І для порозрядного логічного перемноження числа X на розряди y_p, де p=1, 2, …, n.

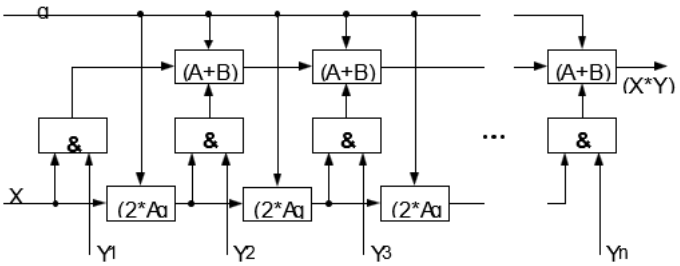


Рис. 2.13 – n-розрядний перемножувач за модулем

Процес вирахування операції по додаванню і множенню в поліноміальному базисі скінченного поля *GF* (2^n) відбувається наступним чином: додавання двох елементів виконується як додавання відповідних многочленів чи як додавання за модулем 2 до цих многочленів двійкових рядків. Іншими словами, додавання виконується як порозрядне додавання за модулем 2.

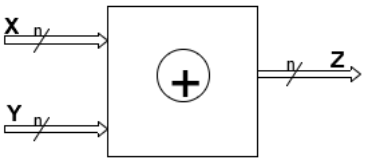


Рис. 2.14 – n-розрядний додавач в поліноміальному базисі скінченного поля

Процес множення двох елементів відбувається за принципом множення відповідних многочленів з подальшим зведенням результатів за модулем примітивного многочлена.

Для випадку, коли є потреба з обчисленням Z=C mod q, де C – *(2n – 1)* – розрядний, q – (*n+1)* – розрядний двійковий рядок, а Z – *n*-розрядний результат можна використати такий алгоритм:

```
T:=C; YY:={qn, qn – 1,…,q0,0,…,0};
для і від (2n – 2) до n виконати
початок
якщо xi=1 тоді T:=T⊕YY;
YY=YY логічний зсув вправо на 1 позицію;
i:=i – 1;
кінець.
Z:={tn – 1,…,t1, t0}
```

Процес по додаванню і множенню в нормальному базисі скінченного поля $GF\ (2^m)$ можна представити у вигляді: додавання двох елементів виконується як додавання многочленів або як додавання за модулем 2 відповідних до цих многочленів двійкових рядків. Таким чином розуміємо, що операція додавання виконується таким чином, як і додавання в поліноміальному базисі.

Однак операція множення елементів в нормальному базисі відбувається наступним чином:

За вхідні дані візьмемо: двійкові рядки $a = (a_0\ a_1\ \dots\ a_{m-1})$ і $b = (b_0\ b_1\ \dots\ b_{m-1})$ та мультиплікативнц матрицю М. Результатом буде такий вираз $c = (c_0\ c_1\ \dots\ c_{m-1})$. А алгоритм множення виглядає наступним чином:

```
Set x ← a.
Set y ← b.
For k from 0 to m – 1 do
Compute via matrix multiplication
ck := x M ytr
where ytr denotes the matrix transpose of the vector y.
Set x ← LeftShift (x) and y ← LeftShift (y), where LeftShift
denotes the circular left shift operation.
Output c = (c0 c1 … cm–1).
```

Для обрахунку мультиплікативної матриці М будуть використовуватись наступні розрахунки:

t=a0,0+a0,1t+a0,2t2+…+a0, m – 1tm – 1 (mod (p (t)))

t2=a1,0+a1,1t+a1,2t2+…+a1, m – 1tm – 1 (mod (p (t)))

t4=a2,0+a2,1t+a2,2t2+…+a2, m – 1tm – 1 (mod (p (t)))

…

$2m–1\ 2 \qquad m-1$

$T = am-1,0 + am-1,1t + am-1,2t + \ldots + am-1, m-1t \pmod{p(t)}$

Отримаємо матрицю А:

$$A := \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,m-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,m-1} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m-1,0} & a_{m-1,1} & \cdots & a_{m-1,m-1} \end{bmatrix}$$

Обчислимо матрицю В: $B=A$. Маючи нормальний поліном

$p(t) = t^m + c_{m-1} t^{m-1} + \ldots + c_1 t + c_0$ і матриці А і В, спочатку визначимо матрицю С

$$C := \begin{bmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & 1 \\ c_0 & c_1 & c_2 & \ldots & c_{m-1} \end{bmatrix}$$

Тепер знайдемо $D := ACB$ в полі GF (2); $d_{i,j}$ є (i, j) елементом матриці D, для $0 \le i, j < m$, і обрахуємо мультиплікативну матрицю М так, щоб $\mu_{i,j} := d_{j-i,-i}$:

$$M := \begin{bmatrix} \mu_{0,0} & \mu_{0,1} & \cdots & \mu_{0,m-1} \\ \mu_{1,0} & \mu_{1,1} & \cdots & \mu_{1,m-1} \\ \cdots & \cdots & \cdots & \cdots \\ \mu_{m-1,0} & \mu_{m-1,1} & \cdots & \mu_{m-1,m-1} \end{bmatrix}$$

Матриці М має у собі $2 \cdot m - 1$ ненульових елементів. Тому замість матриці обраховуються явні формули, котрі формують один розряд множення через розряди співмножників.

Схема елементу перемножувача, який здатний обчислити розряд добутку, відображений на рис. 2.7.

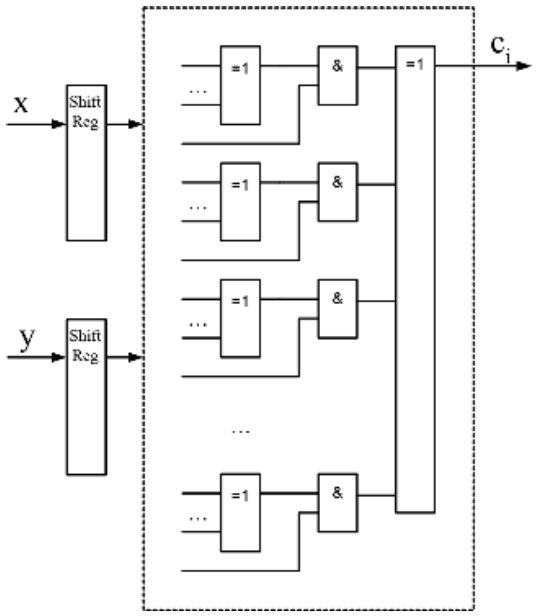


Рис. 2.15 – Схема елемент перемножувача в нормальному базисі скінченного поля

Відповідно, актуальним є реалізація логічних операцій мовою описання апаратних засобів VHDL. Беручи до уваги те, що на проектування моделі обчислювача у обчислювальній системі будуть надмірно великі апаратні затрати, можна зробити висновок, що дана модель не може бути реалізована. Однак з іншого погляду є мова VHDL, котра біла розроблена спеціально для відтворення у симуляторах різних дискретних систем. Саме стандарт IEEE Standard VHDL Language Reference Manual зберігає у собі той синтаксис який потрібен для використання у симуляторі, найчастіше такі симулятори розробляють на однопроцесорних ЕОМ.

Висновки до розділу 2

Отже, актуальним завданням є проєктування системи виконання операцій керування виконавчими механізмами по догляду за рослинами мовою описання апаратних засобів VHDL. Для моделювання програміст створює проект – каталог з файлами VHDL, що має назву проекту. Після компіляції в проекті створюється бібліотека проекту, що має назву проекту й містить всі скомпільовані об’єкти проекту.

Після запуску програми на моделювання спершу виконується зв’язування об’єктів проекту та видача перших значені для усіх змінних та сигналів. Варто завжди пам’ятати, що навіть невеликий процес завжди буде потребувати незначної але все ж таки кількості пам’яті. А границя по кількості процесів завжди буде опиратись на об’єм динамічної пам’яті ПК. Заміна процесу іншим являє собою зміну контексту в роботі центрального процесорного елемента, що може тривати кілька тисяч його тактів; найбільшу частину часу займає виконання обчислювального процесу ядра симулятора, який збільшується із збільшенням кількості обчислювальних процесів і числа сигналів у списках чутливості їхніх операторів wait.

3. МОДЕЛЮВАННЯ ПРОЄКТОВАНОЇ ДАПТИВНОЇ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ АВТОМАТИЧНОГО ДОГЛЯДУ ЗА РОСЛИНАМИ

3.1. Розробка алгоритму функціонування виконавчого механізму керування поливом

Живлення двигуна керування електроприводом системи поливу здійснюється за допомогою ШІМ-перетворювача. Для стабільного живлення необхідно відбирати максимально можливу потужність від джерела живлення. Роль регулятора відбору максимальної потужності виконує – блок перемноження (обчислює потужність, є результатом виміряного струму і напруги) і мікроконтролер, який змінює напругу навантаження після чого іде спостереження на реакцію цієї системи, тим самим проявляючи максимум на кривій потужності ШІМ-перетворювача.

Все ж мікроконтролер, є основою всього пристрою і виступає основою інтелектуальної системи управління процесом живлення двигуна постійного струму і відбору максимальної потужності від джерела живлення.

Мікроконтролер – це спеціалізована мікросхема, призначення якої – керування електроприладами, можна сказати що це комп’ютер виконаний у вигляді невеликої мікросхеми, в якій на одному «кристалі» містяться всі основні компоненти: процесор, периферія, пристрої введення-виведення, а також, найчастіше, оперативна пам’ять (ОЗУ) і енергонезалежна пам’ять (ПЗУ). Звичайно, потужність такого комп’ютера зовсім невелика і не зрівняється з потужністю настільного або портативного комп’ютера.

Мікроконтролер вирізняється значною кількістю показників, адже він водночас є складним приладом, який програмно управляється, і електропристроєм. Префікс «мікро» у назві мікроконтролера означає, що виконується він по мікроелектронної технології.

Алгоритм роботи мікропроцесорного ШІМ-перетворювача є системою послідовних функціональних перетворень з метою одержання заданого вихідного сигналу для живлення двигуна постійного струму. Оскільки мікропроцесор є основою системи, що розробляється, то навантаження виконує саме програма, яка призначена для здійснення функцій перетворення та обчислень, реалізовує суттєве навантаження. Завдяки застосуванню мікропроцесора, можна значно мінімізувати апаратні затрати. Проте, уникнути їх зовсім неможливо – треба всі сигнали переводити до норми, необхідної для програмного оброблення. Алгоритм роботи проектованого мікропроцесорного ШІМ-перетворювача на Рис. 3.1.

Після включення системи проектованого мікропроцесорного ШІМ-перетворювача, починається встановлення системи, загрузка програми в оперативну пам’ять, для чого потрібний певний час. Після чого система починає свою роботу. Після цього відбувається опитування портів АЦП1 і АЦП2. Незважаючи на те, яке значення містять інформаційні виходи, воно є одиничною вибіркою (n) і записується до пам’яті. Цей процес продовжується, доки кількість вибірок не досягне 1024. Потім опитування АЦП1 і АЦП2 зупиняється і починається процес оброблення накопичених вибірок, із яких у результаті виводиться середньоквадратичне значення (СЗ), що також заноситься до пам’яті і також до вихідних регістрів (N). Якщо у результаті таких регістрів виявилося менше 200, то система продовжує процес опитування АЦП, щоб отримати нові значення СЗ. Якщо їх кількість досягла 200, за допомогою сервісних функцій виокремлюється максимальне значення, також обраховується середнє значення, після чого, результати обрахунків заносяться до вихідних регістрів. Цифро-аналогові перетворювачі, які під’єднані до вихідних регістрів, трансформують цифровий код в аналоговий сигнал. Це є завершенням циклу одного вимірювання, після чого система розпочинає наступний. Цей процес буде тривати до того часу, доки система проектованого мікропроцесорного ШІМ-перетворювача працює. Підводячи підсумки роботи системи, протягом одного циклу вона проводить обрахунки 200 значень СЗ, 1 максимальне та 1 середнє значення відповідно.

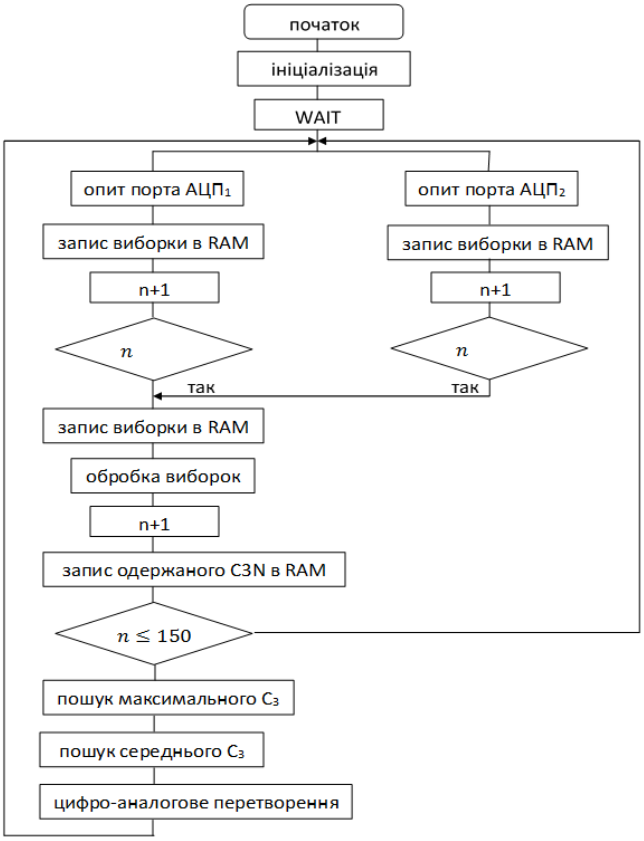


Рис. 3.1 – Алгоритм роботи проектованого мікропроцесорного ШІМ-перетворювача

Алгоритмічна програма роботи мікропрограми мікропроцесорного контролера складається з чотирьох функцій, які ув’язуються між собою наступним чином:

– choose – load – функція опитування портів $АЦП_1$ і $АЦП_2$ і запису в пам’ять одержаних виборок.

– choose processing – функція обробки виборок з метою отримання середньо квадратичного значення.

– S-max – пошук максимального значення.

– S-medium – пошук середнього значення.

Загальна алгоритмічна програма наведена на рис. 3.2.

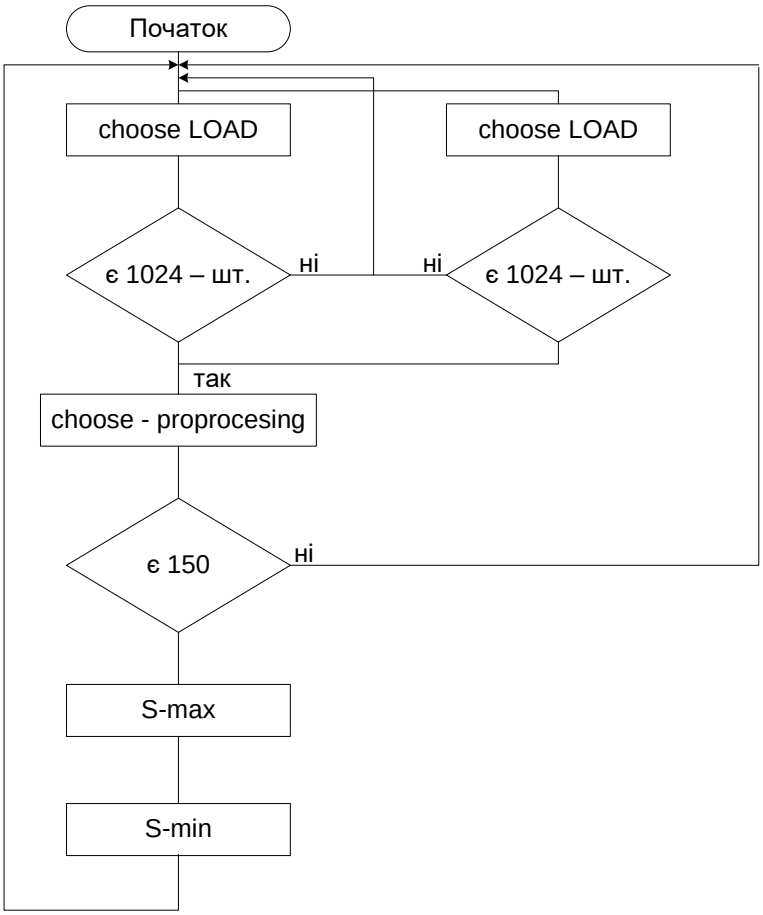


Рис. 3.2 – Загальний алгоритм роботи програми

В свою чергу функції представляють собою окремі підпрограми і алгоритмічно представлені на рис. 3.3.

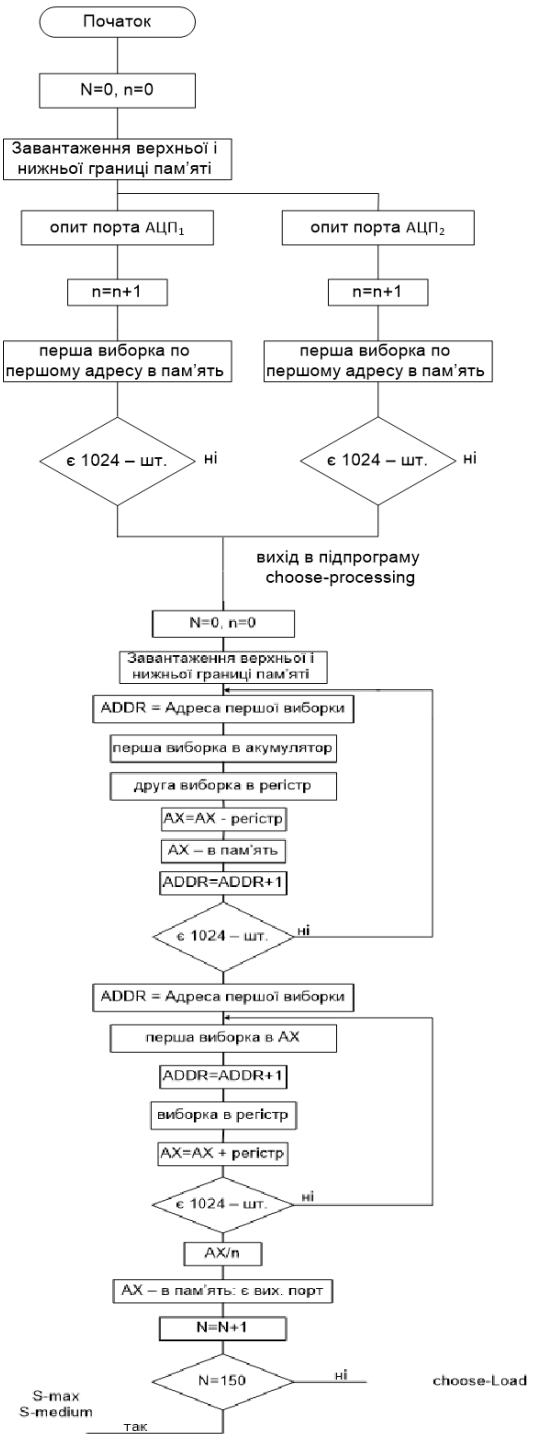


Рис. 3.3 – Алгоритм підпрограми функцій

Спочатку проведемо розрахунок пам’яті для створення мікропрограми мікроконтролера. Системна пам’ять складається із 2х видів: оперативної та постійної. У постійній пам’яті міститься код програми і певні константи системи. В оперативній – розміщуються результати проміжних обрахунків. Необхідно орієнтовно 6К оперативної пам’яті для коректної роботи системи.

Постійна пам’ять:

Після обчислення кількості байтів усіх програмних команд, можемо узагальнити дані у таблицюі 3.1.

Таблиця 3.1 - Розподіл пам’яті мікроконтролера

Команда	К-сть в програмі	n тактів	n байт
MOVE	14	10+E	3–6
IN	1	10	2
OUT	3	10	2
LES	12	16+E	3–4
DES	4	16+E	2–4
ADD	14	3	2
CMP	7	4	2–3
MVC	1	133	2–4
DIV	2	162	2–4
JMP	7	24+E	2–4
CALL	5	37+E	2–4
RET	5	18	3
WAIT	1	3	1
HLT	1	2	1
LOOP	1	17	2

Лістинг програм наведено в Додатку И. Програма керування контролера, розроблена в середовищі TwidoSoft на мові List Instruction відповідно до запропонованої структури мікропроцесорного контролера. Під код програми виділяється 300 байт, тому достатнім об’ємом ПЗУ буде 1К. Час виконання програми складає орієнтовно 9000 тактів, – 600 мкс із частою 5 мГц (один такт триває 200нс).

Для реалізації проекту мікропроцесорного контролера потрібен 12-розрядний АЦП. Вибираємо ATMEGA32, з такими характеристиками: 32-розрядний, нелінійність Si=2MP, похибка перетворення 2 мкс, струм споживання від джерела Iu1=80mA, Iu2=150mA.

Наступним етапом проводим дослідження на похибку АЦП. Вона складається в свою чергу з наступних похибок: дві похибки квантування по часу, одного відліку, динамічну і статичну.

Динамічною похибкою вважають інерційність елементів, які мають вплив на час, який витрачається на виконання процесів та на швидкість змін на вхідних сигналах до системи.

Похибку квантування, котра формується пуд час кінцевого циклу квантування та інструментальна – це похибки, які відносяться до похибок одного відліку. Під час квантування за рівнем вхідна напруга може розбитися на 2^n дозволених дискретних кванти, де квант матиме таке значення:

$$q=\frac{(U_{вхmax}-U_{вхmin})}{2^n}=\frac{Umn}{2^n}, \quad (3.1)$$

де n – представляє собою кількість розрядів, які визначають кількість рівнів, на які розбивається вхідний сигнал.

Uмп – це діапазон, яких характеризує зміни у вхідному сигналі (шкала квантування), який обмежується верхнім та нижнім показником аналогового сигналу.

Кожний крок квантування має здатність перетворювати за рівнем, чи найменший приріст, або по іншому називають – зміною, від його вхідного сигналу. Саме цю зміну сигналу зазвичай визначають, як одиницю молодшого розряду. Вона представляє з себе деяку функцію числа розрядів, котра при зростанні кількості розрядів на виході – зменшується, числове значення якої представляють у процентах від установленої шкали, або у мілівольтах. При чому, найбільша допустима похибка вважається 9 / 2.

У нашому розроблювальному АЦП з 12 розрядами сигнал може квантуватись по 4096 рівням, іншими словами перетворювач поділить усю вхідну напругу на 2^{11} частин, де роздільна здатність буде дорівнювати 1 / 1024, а похибка відповідно буде 1 / 4096.

$$n=\lg\frac{2*100}{\delta kT6} \quad (3.2)$$

при n>12 δ≈0,01%

Найменша зміна на вході при якій виробляється цифровий код, буде рівна

$$g=\frac{Umn}{21^2}=\frac{10}{4096}=0{,}0024В=2{,}4МВабо0{,}024\ м/с \quad (3.3)$$

Абсолютна похибка включає в себе похибку квантування і інструментальні похибки, диференціальну не лінійність, похибку підсилення та зміщення нуля. Похибка схеми вибірки дорівнює 0,1% Отже загальна похибка системи: $\delta = \delta_k + g = 0{,}01 + 0{,}024 + 0{,}1 = 0{,}135\%$

3.2. Розробка структурної схеми нейропроцесора

Структурна схема – це схема, у якій окреслюються головні функціональні елементи створюваного виробу, їх зв’язки між собою та призначення. Функціональні елементи – це складові деякої схеми: елемент, прилад, функціональна група чи ланку. Звичайні функціональні ланки відображаються простими прямокутниками, а зв’язки між ними – стрілками, які демонструють напрям її дії. Деколи у прямокутник також вносять формулу закону перетворення сигналу, у такому випадку схема має назву «алгоритмічна».

Основою структурної схеми спеціалізованого нейропроцесора для виконання логічних операцій в галузі керування електроприводами виконавчих механізмів системи догляду за рослинами є арифметико-логічний пристрій. Арифметико-логічний пристрій (АЛП; arithmetic and logic unit, ALU) є комбінаційної схемою, здатної виконувати цілий ряд різних арифметичних і логічних операцій з парою n-розрядних операндів. Виконувана операція визначається комбінацією сигналів на входах вибору функції [15, с. 139].

Арифметико-логічний пристрій можна поділити на дві категорії за способом дії: паралельні та послідовні. Послідовні АЛП розташовують операнди у послідовному вигляді, та операції над ними можуть проводитись лише в окремий час, так як виконання йде у послідовному порядку. У

паралельних АЛП все навпаки, усе відбувається паралельно, від представлення операндів до операцій над ними. Програмний АЛП можна представити у вигляді операційного блока, функціями якого є прийом та передача операндів з одного пристроя до іншого. Опис функцій у АЛП приведений у таблиці 3.2.

Хоча мова VHDL має вбудовані оператори складання (+) і віднімання (-), вони працюють тільки з цілими і дійсними числами і фізичними типами. Зокрема, вони не працюють з типами BIT_VECTOR і типом STD_LOGIC_VECTOR стандарту IEEE. Для цих типів в стандартних пакетах визначені спеціальні оператори

Таблиця 3.2 - Перелік стандартних функцій АЛП

Функція	Опис функції
R=X+Y	Додавання X і Y
R=X+Y+Cl	Додавання X і Y з перенесенням
R=X – Y	Відняти Y з X
R=X – Y – Cl	Відняти Y з X з позичкою
R=Y – X	Відняти X з Y
К=Y – X – Cl	Відняти X з Y з позичкою
R= – X	Арифметичне заперечення X
R= – Y	Арифметичне заперечення Y
R=Y+1	Інкремент Y
R=Y – 1	Декремент Y
R=PASS X	Результат дорівнює операнду X
R=PASS Y	Результат дорівнює операнду Y
R=0 (PASS 0)	Очистити результат
R=ABS X	Результат дорівнює абсолютному значенню X
R=X AND Y	Логічне і (AND) X і Y
R=X OR Y	Логічне або (OR) X і Y
R=X XOR Y	Виключає логічне або (XOR) X і Y
R=NOT X	Логічне заперечення X

R=NOT Y	Логічне заперечення Y

У пакеті IEEE_std_logic_arith визначені два нових типи масивів – SIGNED і UNSIGNED – і набір функцій порівняння для операндів типу INTEGER, SIGNED і UNSIGNED. В даному пакеті визначені операції додавання і віднімання для операндів тих же типів, а також для 1-розрядних операндів типу STD_LOGIC і STDJJLOGIC.

При великому числі перекриваються функцій додавання і віднімання не настільки очевидно, яким виявиться тип результату додавання або віднімання. Якщо хоча б один з операндів належить типу SIGNED, то зазвичай результат буде типу SIGNED, в іншому випадку результат буде типу UNSIGNED. Але якщо результуюче значення присвоюється сигналу або змінної типу STD_LOGIC_VECTOR, то результат типу SIGNED або UNSIGNED перетворюється до цього типу. Розрядність будь-якого результату зазвичай дорівнює розрядності найдовшого операнда. Однак, коли операнд типу UNSIGNED бере участь в одній операції з операндом типу SIGNED або INTEGER, його розрядність збільшується на 1 для розміщення в ньому знакового біта, рівного 0, і тільки після цього встановлюється розрядність результату [28, с. 110–117].

Наведемо VHDL-програму складання 8-розрядних операндів, що ілюструє ці правила. Перший результат S оголошений як 9-розрядне двійкове число, яке може виникнути при додаванні 8-розрядних операндів A і B типу UNSIGNED. За допомогою оператора конкатенації & операнди A і B розширюються так, щоб функція складання поміщала біт перенесення в старший розряд результату.

Лістинг програмного коду:

```
library IEEE;
use IEEE. std_logic_1164. all;
use IEEE. std_logic_arith. all;
entity lab4 is
port (
A, B: in UNSIGNED (7 downto 0);
C: in SIGNED (7 downto 0);
D: in STD_LOGIC_VECTOR (7 downto 0);
G: in STD_LOGIC_VECTOR (7 downto 0);
S: out UNSIGNED (8 downto 0);
```

```
T: out SIGNED (8 downto 0);
U: out SIGNED (7 downto 0);
V: out STD_LOGIC_VECTOR (8 downto 0);
Log_i: out STD_LOGIC_VECTOR (7 downto 0);
Log_ili: out STD_LOGIC_VECTOR (7 downto 0);
Log_iili_iskl: out STD_LOGIC_VECTOR (7 downto 0);
Log_not_x: out STD_LOGIC_VECTOR (7 downto 0);
Log_not_y: out STD_LOGIC_VECTOR (7 downto 0)
);
end lab4;
architecture lab4_arch of lab4 is
begin
S <= ( '0' & A) + ( '0' & B);
T <= A + C;
U <= C + SIGNED (D);
V <= C - UNSIGNED (D);
Log_i <= (D AND G);
Log_ili <= (D OR G);
Log_iili_iskl <= (D XOR G);
Log_not_x <= (NOT D);
Log_not_y <= (NOT G);
end lab4_arch;
```

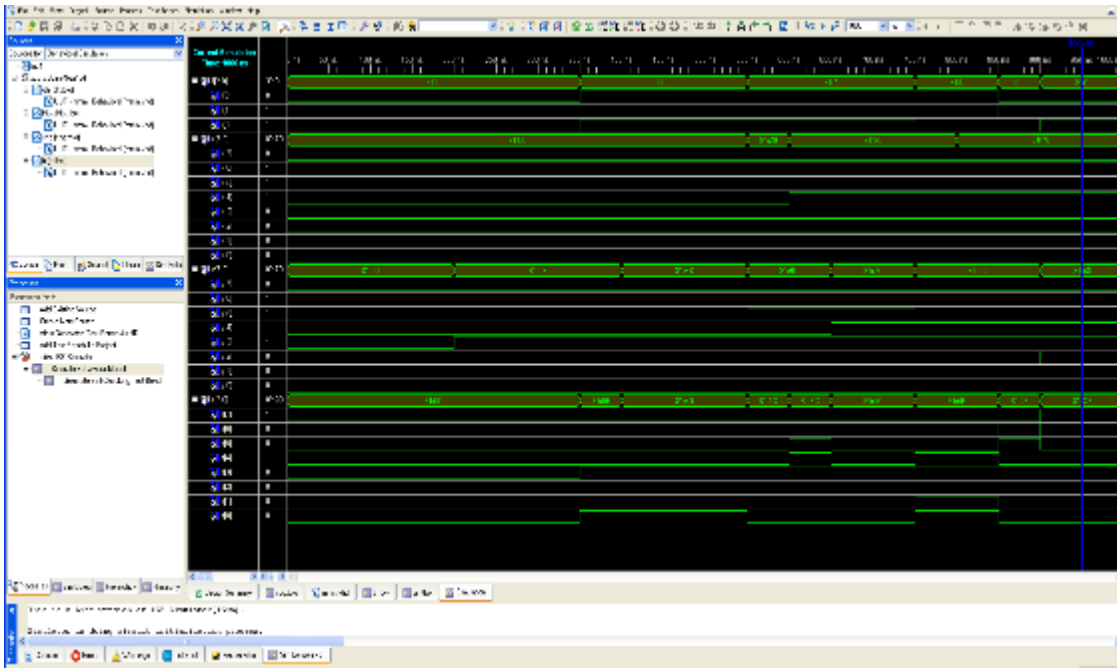


Рис. 3.4 – Часова діаграма логічних операцій нейропроцесора

Складемо структуру логічного нейропроцесора для виконання логічних операцій.

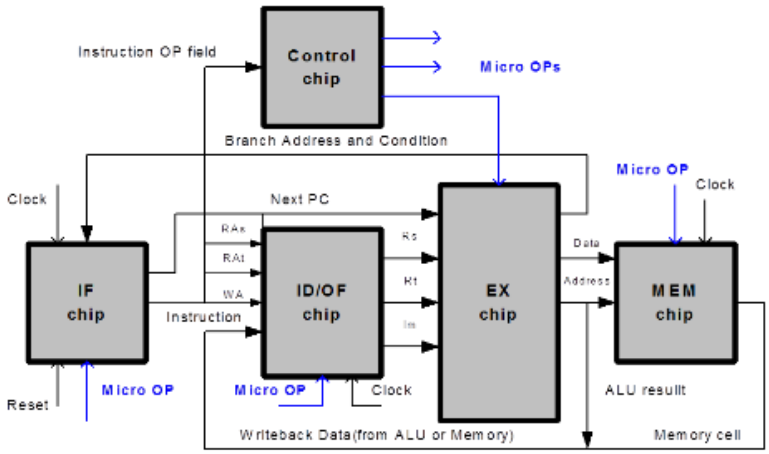


Рис. 3.5 – Спрощене подання синтезованої структури логічного нейропроцесора

Далі представимо детальну структуру логічного нейропроцесора для виконання логічних операцій.

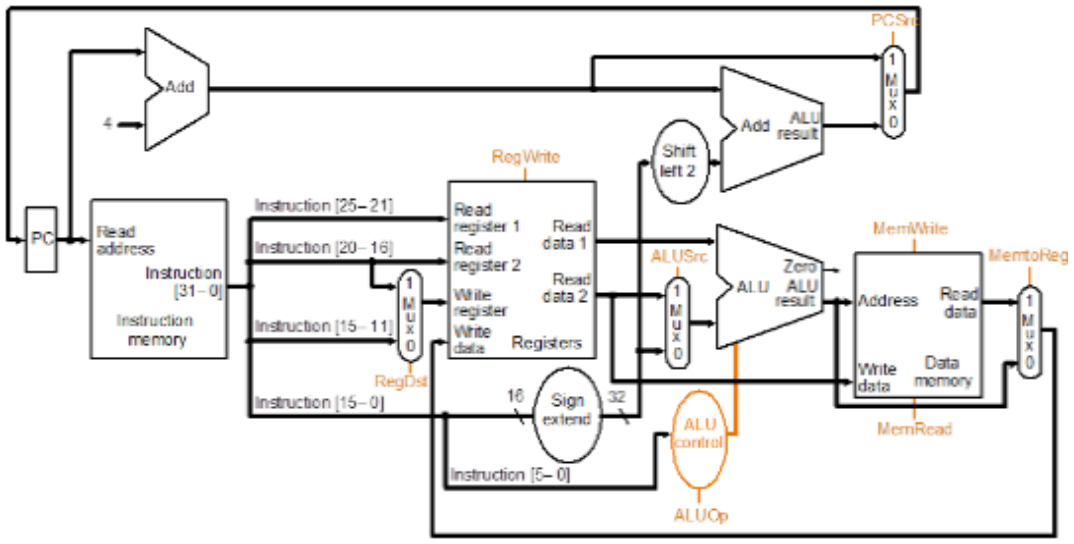


Рис. 3.6 – Структурна схема нейропроцесора для виконання логічних операцій

Сучасна компанія Xilinx з розробки ПЛІС дає змогу проектувати моделі для синтезу в ПЛІС з незмінною розрядністю, а саме 32 регістрів для загального призначення та об’ємом пам’яті програми і даних по кілька кілобайтів.

Через полегшення MIPS архітектури, яку реалізовано у прототипі, можна зрозуміти що не важко буде виправити. Ці спрощення допускаються, щоб збільшити варіативність вихідних даних для розробки проекту.

3.3. Розробка структурної схеми виконавчого механізму керування поливом

Опишемо загальну структуру проектованої системи ШІМ-перетворювача керування електроприводом системи поливу. На рис. 3.7 показана структурна схема ШІМ-перетворювача.

Імпульсний перетворювач, служить основним елементом і виступає виконавчим елементом у системі контролю за роботою двигуна.

Принцип дії ШІМ-перетворювачів постійної напруги закладається в тому, що відбувається комутація навантаження до джерела живлення, а саме час від часу підключається, та відключається від. Досягається це за допомогою електронно-силових ключів – транзисторів, діодів, тиристорів. Середня напруга та струм навантаження залежать від проміжків між підключенням джерела напруги до накопичувальних елементів схеми ШІМ-перетворювача.

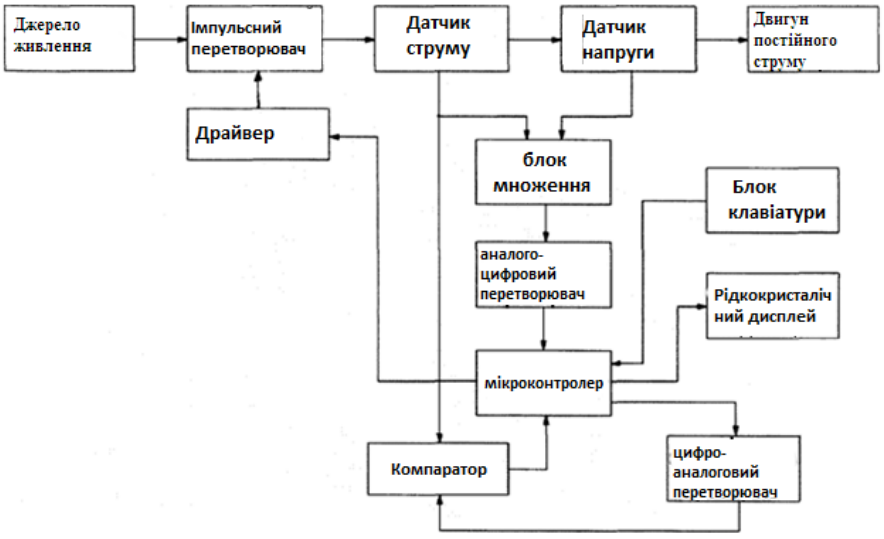


Рис. 3.7 – Структурна схема ШІМ-перетворювача

Датчики струму і датчики напруги здійснюють перетворення поточних значень струму і, відповідно, напруги (в контрольованому колі) в електричний сигнал, у якого носієм інформації зазвичай є напруга. У датчиках може передбачатися гальванічна розв’язка вихідний (слабкострумової) ланцюга від

вхідних (силовий) ланцюга, а також нормування сигналу (приведення його значень до певної області, наприклад, до напруги з діапазону 0... 220 В). До складу такого датчика входять наступні функціональні частини: чутливий елемент (первинний вимірювальний перетворювач), пристрій гальванічної розв'язки (потенційний роздільник), підсилювальні пристрої.

Датчик струму і датчик напруги, необхідні для визначення величини струму і напруги. Далі відбувається оцифровка сигналу з датчиків і використання цих значень для розрахунку потужності на виході ШІМ-перетворювача.

У вигляді комбінаційній схемі, котра опрацьовує множення чисел з 8 розрядами і водночас має змогу округлити результат – називають блоком живлення. Це комбінаційна матриця, в якій формуються часткові похідні від порозрядного множення.

Блок множення запускається імпульсом малої тривалості. Зчитування програми відбувається на різних швидкостях руху, що впливає на різкість фронту, амплітуду і тривалість зчитуваного імпульсу.

Блок множення служить для обчислення потужності, яка визначається шляхом перемноження струму і напруги.

Всяка система, котра хоч якось обробляє інформацію у цифровому вигляді, мусить містити у собі прилад, який займається перетворенням цифрових та аналогових сигналів. Ці прилади називають аналого-цифрові (цифро-аналогові) перетворювачі.

Цифро-аналоговий перетворювач (ЦАП) виконує важливу роль, яка полягає у тому, щоб перетворити число з вигляду коду у напругу, яка буде відповідати значенню вхідного числа. Даний пристрій здатний перетворювати не усі числа, а лише числа у певному двійковому коді, який має назву вагового.

Аналого-цифровий перетворювач (АЦП) – представляє з себе пристрій, якій отримує на вхід аналоговий сигнал і перетворює його до цифрового коду, який може бути використаний для операцій у мікропроцесорі або інших будь яких пристроях.

Існують певні статичні характеристики, котрі характеризуються характером перетворення, встановлюючи певну залежність між аналоговими і цифровими величинами, а також беруть на себе відповідальність за точність перетворень, які відбувались.

До них поставляють такі параметри:

– розрядність n - це число розрядів цифрового коду, який утворюється на виході АЦП або подається на вхід ЦАП.

– максимальна кількість кодових комбінацій на виході АЦП або вході ЦАП, визначається завдяки числу розрядів цифрового коду і дорівнює: 2n – для двійкових ЦАП (АЦП); 3n – для трійкових ЦАП (АЦП);

– максимальна напруга Umax (діапазон зміни вихідної напруги) - це найбільша вхідна напруга для АЦП і вихідна для ЦАП;

– роздільна здатність (абсолютна роздільна здатність) - мінімальне значення для вхідної величини, яке пристрій здатний розпізнати і зафіксувати при вихідних значеннях. Роздільна здатність для ЦАП – це найменше значення зміни вихідного сигналу, яке створюється різницею вхідного значення на самий менший розряд. Для АЦП – це мінімальна величина аналогових даних, котрі викликають різницю вихідного коду на одиницю у молодшому розряді;

– нелінійність δL - це найбільше відхилення точки реальної характеристики перетворення від абсолютної;

– диференціальна нелінійність - це відхилення дійсного кроку квантування від його середнього значення; абсолютна похибка перетворення в кінцевій точці шкали – відхилення реальних максимальних значень вхідного для АЦП і вихідного для ЦАП аналогових сигналів від значень, що відповідають кінцевій точці ідеальної характеристики перетворення;

– напруга зсуву для нуля U0. Для АЦП це напруга, котра необхідна бути на його вхідному сигналі, для отримання так само нульового сигналу на виході. Для ЦАП – це напруга, котра формується на виході пристрою при подачі на вхід пристрою нульового коду.

До динамічних ознак ЦАП та АЦП можна віднести такі параметри як:

– максимальна частота перетворення - це максимальна частота дискретизації, формуючі сигнали будуть відповідати нормативним вимогам;

– час перетворення - це проміжок часу, за який необхідний для появи напруги на виході ЦАП при подачі на нього вхідного значення або час затрачений на зміну сигналу на виході після зміни його на вході АЦП.

Драйвер – електронний компонент для управління перетворювачем. Використовується для посилення, формування фронту і спаду керуючих імпульсів.

Драйвер – буферний драйвер, апаратна компонента (мікросхема чи дискретна схема), за допомогою котрої відбувається комутація між різними компонентами системи при різних несумісних сигналах. Прикладом є адаптер, який погоджує рівні сигналів на різних приладах.

Драйвер складається з декількох функцій, які обробляють певні події системи. Зазвичай це 5 основних подій:

– завантаження драйвера.

– відкриття драйвера. Початок основної роботи.

– запис;

– закриття: операція, зворотна відкриттю, звільняє зайняті при відкритті ресурси й знищує дескриптор файлу;

– керування вводом - виводом. Найчастіше драйвер підтримує інтерфейс вводу-виводу специфічний для даного пристрою. За допомогою цього інтерфейсу програма може послати спеціальну команду, що підтримує даний пристрій.

Під час виконання роботи мікроконтролер отримує команди з постійної пам’яті або портів введення та обробляє їх. Суть та мету команди формує система команд. Саме ця система вбудована до мікроконтролеру та обробка команд полягає у виконанні вбудованими елементами мікросхеми деякий мікрооперацій.

Завдяки головному мікроконтролеру управління усіма приладами системи відбувається дуже гнучко та чітко. А потужність окремих

мікроконтролерів настільки велика, що їм під силу, навіть, перемикання силових реле.

Зазвичай, мікроконтролери не працюють поодиноко, вони розкривають увесь свій потенціал коли їх додають у певну систему, в якій можуть використовуватись різноманітні екрани, датчики, клавіатури, тощо.

Прошити мікроконтролер можливо виконуючою програмою, яку можна написати на асемблері або на Сі.

Мікроконтролер так само бере на себе функції управління перетворювачем. Формує імпульси встановленої тривалості для збільшення або зменшення напруги навантаження з метою визначення максимуму відібраної потужності, або не формує (відключення навантаження) у разі припинення обертання двигуна.

Клавіатура – це стандартний клавішний пристрій введення, призначений для введення даних та інструкцій щодо виконання певних команд. Коли об’єднується клавіатура та монітор, то цією комбінацією можна забезпечити найпростіше сприйняття інтерфейсу користувача та взаємодію з ним. Клавіатура надає змогу керувати розробленою системою, а монітор відслідковувати дії, що відбуваються завдяки взаємодії з клавіатурою.

Багато хто з виробників різних систем вважають клавіатуру стандартним засобом, який є необхідним для взаємодії з системою, тому найчастіше усі драйвери вже знаходяться у постійній пам’яті мікросхеми, до якої під’єднується клавіатура, через це від розробника не вимагається ніяких системних програм для коректної роботи.

Блок клавіатури для вибору тієї чи іншої акумуляторної батареї з переліку (бази даних) занесеного в пам’ять мікроконтролера. Рідкокристалічна індікація-LSD дисплей необхідний для зручності візуального «діалогу» між користувачем і пристроєм.

Рідкі кристали – це речовини, що проявляються в певному температурному інтервалі, мають властивості як рідини, так і кристалів. Вони здатні в рідкому стані зберігати впорядкованість молекул (подібно кристалам). Для створення індикаторів на рідких кристалах використовуються так звані

нематичні рідкі кристали, які є структурною різновидом даного класу речовин. Матеріалом для них служать суміші органічних сполук, молекули яких формуються в впорядковані решітки.

Існують два принципи роботи індикаторів на рідких кристалах. Перший з них полягає в тому, що при взаємодії електричного імпульсу від поля на тонкий шар РК речовини, який знаходиться проміж пластинками зі скла, структура кристалів дещо змінюється, а саме руйнується її впорядкування і через це відбувається дифузне розсіювання, котре ще називають ефектом динамічного розсіювання. У результаті прозорий рідкокристалічний шар стає непрозорим і при взаємодії з навколишнім світлом – стаю дуже просто побачити різницю у тому, яка ділянка збуджена, а яка не є і не збудженою. При знятті зовнішнього електричного поля первісна структура рідких кристалів відновлюється і вказаний контраст зникає.

Рідкокристалічні індикатори загалом можна представити, як дві скляні пластинки, всередині яких розташовуються рідкі кристали, як правило товщиною 12-20 мкм. На одній із двох пластин за допомогою струмопровідного шару розташовують ілюстрацію цифри, яка, як правило складається із сегментів, з використанням яких можна буде відтворити будь яку цифру від 0 до 9. Однак до другої пластинки тим же покриттям під’єднують електрод, який буде загальним для усіх сегментів цифри.

Існує другий спосіб з використанням іншої структури для табло, яке основане на рідких кристалах. Існує такий ефект, як обертання площини поляризації поляризованого світла за допомогою шару рідких кристалів, котрий зникає при дії на нього електричної взаємодії. Сегменти, які використовують даний тип структури, розміщають малу краплю рідкий кристалів між схрещеними поляроїдними пластинами, де вони займають місце, як дуже тонка плівка. Завдяки саме цьому у даній структурі розташовуються площини поляризації світла перпендикулярно, що дає змогу робити сегменти повністю непрозорими. Але якщо між цими пластинами є шар неметалічних рідких кристалів, які в результаті технологічної обробки отримали властивість

обертання площини поляризації минаючого світла на 90 °, то вся ця оптична система виходить прозорою.

У структурній схеми, розробляємий мікропроцесорний контролер, можна представити як деяку послідовність функціональних блоків, загальний вигляд яких можна зобразити у вигляді незалежного пристрою, який здатний виконувати свої окремі задачі і функції. Для досягнення технічного звдання, до розробляємого пристрою відводяться вимоги за такими функціональними блоками:

Блок вхідного сигналу – цей пристрій буде виконувати функцію по зменшенню напруги для її подальшої обробки.

Блок аналогового-цифрового перетворення – необхідний для переведення значення напруги в двійковий код.

Блок цифрової обробки.

Блок цифрової обробки повинен реалізувати функцію:

$$p=\frac{1}{n}\sum_{K=1}^{n-1} u(tk)\, s(tk) \qquad (3.5)$$

Для обчислення потужності значення необхідно здійснити числові перетворення величин, що зображено на рис. 3.8.

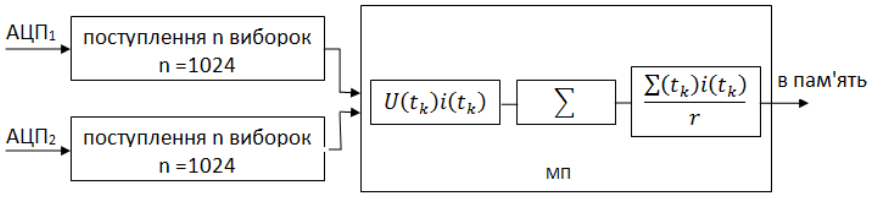


Рис. 3.8 – Структурна схема числових перетворень величин

Операції множення та ділення будуть у стандартному вигляді для нашого обраного мікропроцесора. Вони реалізуються в мікроконтролері програмними засобами.

Відповідно, функціональна схема перетворювача зображена на рис. 3.9

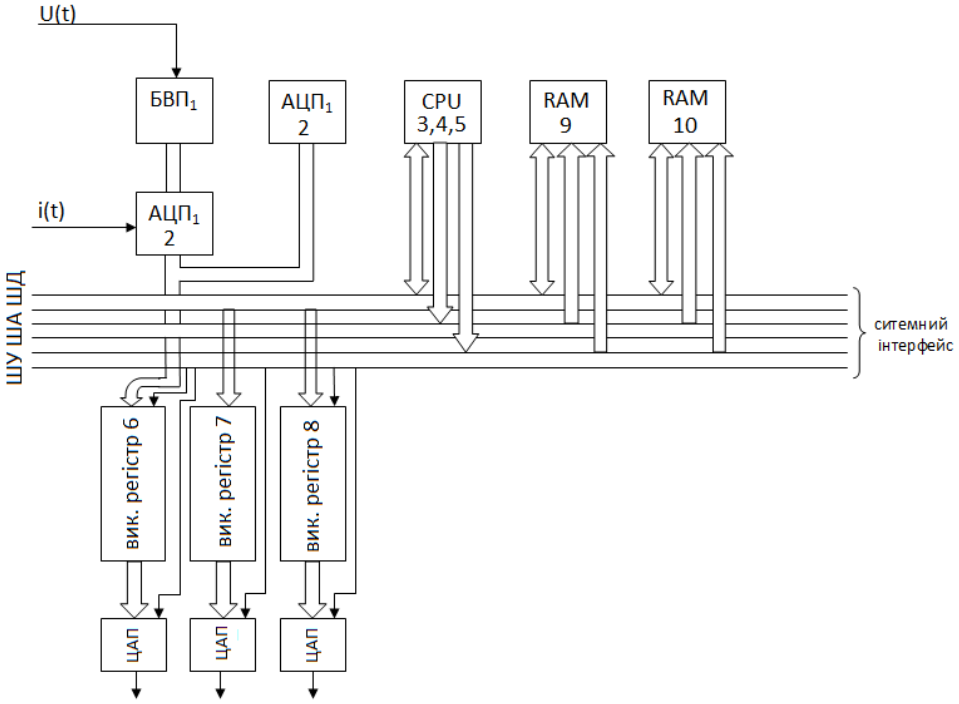


Рис. 3.9 – Функціональна схема ШІМ-перетворювача

Будь яка система потребує у запам’ятовуючих пристроях, тому і в нашій мікропроцесорній системі існують блоки під номерами 9 та 10 – це оперативна та постійна пам’яті. У постійній пам’яті буде розташовуватись програмна реалізація, котра у вигляді коду під час роботи системи буде перенаправлятись до оперативної пам’яті, також постійна пам’ять буде слугувати і для зберігання даних, таких як вхідні дані або проміжні результаті під час обчислень.

Висновки до розділу 3

В даному розділі дипломної роботи виконано розробку алгоритму функціонування нейропроцесора та його виконавчих механізмів. Алгоритм роботи пристрою зображує його роботу на рівні функцій роботи схеми. Фактичної кінцевої точки роботи схеми немає, оскільки робота процесу не припиняється, тобто схема постійно аналізує датчики та блоки та виводить інформацію у вигляді світлової та звукової індикації. Кінцевою точкою роботи пристрою можна назвати ту точку, коли пристрій перестає функціонувати

взагалі, тобто виходить з ладу. Система аналізує сигнали та виконує функції відносно запрограмованих реакцій на стан датчиків, враховуючи дозвіл на виконання результуючої операції.

Створюючи власну структуру нейропроцесорного елемента вирішено, що для гарного результату було б ефективно обрати розрядність – 8 біт, через це розрядність для адрес також варто зробити 8 біт. Відповідно розрядність абсолютно всіх регістрів теж 8 біт.

Реалізація VHDL моделі пристрою дає можливість моделювати їх працю на ПЛІС використовуючи мову опису апаратних засобів, в результаті чого можна отримати числові характеристики їх розміру на кристалі, а також вирахувати час, який потрібен для формування результату після операції множення. Досліджені схеми для матричних пристроїв множення мають відмінності за характеристиками складності та часом формування сигналів на кристалі. Вибір необхідної схеми полягає у виборі її від того, у якому курсі СКС буде виконуватись реалізація.

В якості силової частини виконано розробку алгоритму функціонування виконавчого механізму керування електроприводом. Живлення двигуна керування електроприводом системи поливу здійснюється за допомогою ШІМ-перетворювача. Роль регулятора відбору максимальної потужності виконує – блок перемноження (обчислює потужність, є результатом виміряного струму і напруги) і мікроконтролер, який змінює напругу навантаження після чого іде спостереження на реакцію цієї системи, тим самим проявляючи максимум на кривій потужності ШІМ-перетворювача. Все ж мікроконтролер, є основою всього пристрою і виступає основою інтелектуальної системи управління процесом живлення двигуна постійного струму і відбору максимальної потужності від джерела живлення.

Наведена структурна схема нейропроцесора. Основою структурної схеми спеціалізованого нейропроцесора для виконання логічних операцій є арифметико-логічний пристрій. Арифметико-логічний є комбінаційної схемою, здатної виконувати цілий ряд різних арифметичних і логічних операцій з парою

n-розрядних операндів. Виконувана операція визначається комбінацією сигналів на входах вибору функції.

4. РЕАЛІЗАЦІЯ АРХІТЕКТУРИ ЛОГІЧНОГО ПРОЦЕСОРА ДЛЯ ВИКОНАННЯ ОПЕРАЦІЙ ІНТЕЛЕКТУАЛЬНОГО ДОГЛЯДУ ЗА РОСЛИНАМИ ЗАСОБАМИ НЕЙРОПРОЦЕСОРНОЇ ТЕХНОЛОГІЇ НА БАЗІ VHDL

4.1. Вибір апаратної частини нейроконтролера

Програмовані логічні інтегральні схеми (ПЛІС) є одним із найцікавіших і перспективних напрямків сучасної цифрової мікроелектроніки. За останнє десятиліття спостерігалося бурхливе зростання ринку цих пристроїв і істотне поліпшення їх характеристик. Прогнози в цій галузі на найближчий час видаються найбільш оптимістичними [28, с. 15–28].

З появою ПЛІС проектування цифрових мікросхем перестало бути завданням для великих компаній, які створюють велику кількість кристалів. Проектування і випуск невеликої партії унікальних цифрових пристроїв стало можливим в умовах проектно-конструкторських підрозділів невеликих підприємств, з метою дослідити дану технологію та навіть в умовах домашніх електронних робочих місць. Промислово випускаються «заготовки» програмованих мікросхем з електричним програмуванням і автоматизованим процесом перекладу схеми користувача в послідовність імпульсів програмування робить проектування нових цифрових пристроїв аналогічним з розробкою програмного забезпечення.

Привабливість даної технології полягає в наданою кінцевому користувачу можливістю швидкого створення цифрових пристроїв з довільною внутрішньою структурою. У порівнянні зі спеціалізованими цифровими мікросхемами НВІС (ASIC – Application Specific Integral Circuit), цикл розробки пристроїв на ПЛІС займає значно менший час і є незмірно дешевшим (завдяки тому, що зміна принципової електричної схеми виконується шляхом перепрограмування одного і того ж примірника мікросхеми.

В даний час гігантами у сфері виробництва ПЛІС є дві окремі фірми Xilinx та Altera. Серед продуктів компаній є багато різноманітних ПЛІС, котрі відрізняються різною архітектурою, різних засобів програмування та налагодження. Основним моментом є те, що САПР від цих фірм надається цілком безкоштовно для вивчення технології та власної розробки систем.

На сьогоднішній день у компанії Xilinx є дві серії ПЛІС [9]:

– FPGA – Field Programmable Gate Array – пристрої даної серії використовують для зберігання конфігурації незалежну пам’ять;

– CPLD – Complex Programmable Logic Device – пристрої, які використовують для зберігання конфігурації енергозалежну пам’ять, яка вимагає ініціалізації після включення живлення.

ПЛІС типу FPGA були розроблені наприкінці 1985 року за ініціативою компанії Xilinx, яка запропонувала «мікросхему-конструктор» – напівпровідниковий кристал з цифровими компонентами без жорстко зафіксованих металевих з’єднань, які формують конкретну схему. Сполуками в FPGA можна управляти завдяки перемиканню транзисторних ключів, яких існує купа на самому ж кристалі. Увесь існуючий час характеристики FPGA покращуються, кількість створених ПЛІС все збільшується та сфера використання розширюється завдяки впровадженим поліпшенням. На сьогодні варто зазначити такі підродини випущених серій ПЛІС на прикладі сімейства Virtex – 5:

– LX – логічна обробка даних;

– LXT – логічна обробка даних з високошвидкісними послідовними інтерфейсами;

– SXT – цифрова обробка сигналів (DSP) з високошвидкісними послідовними інтерфейсами;

– TXT – цифрова обробка сигналів (DSP) з високошвидкісними послідовними інтерфейсами подвоєною продуктивності;

– FXT – нейропроцесорна система і супершвидкісний обмін даними.

На рис. 4.1 представлені сучасні серії ПЛІС Xilinx [28].

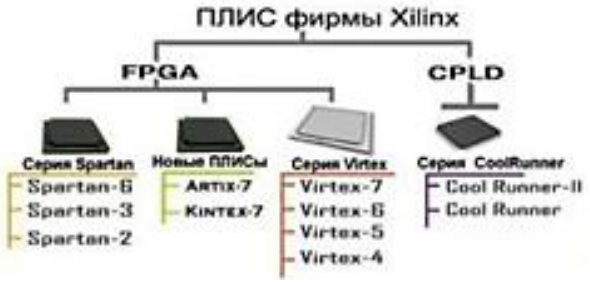


Рис. 4.1 –ПЛІС виробництва фірми Xilinx

Ми будемо роздивлятись архітектурну ПЛІС із сімейства Spartan – 3 (рис. 4.2). Сімейство Spartan™-3 було спроектовано для того, щоб їх використовувати в електронних пристроях, при чому дане сімейство призначене для великих партій з відносно недорогими компонентами.

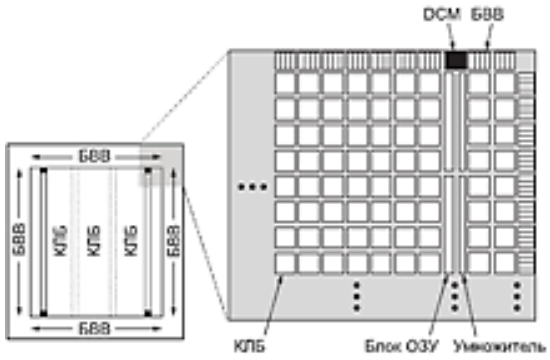


Рис. 4.2 – Структурна схема кристала ПЛІС Spartan – 3

На рис. 4.3 зображені п’ять основних частин архітектури ПЛІС із сімейства Spartan – 3:

– конфігурований логічний блок – КЛБ. Завдяки КЛБ виконується логіка, як комбінаторна, так і синхронна, яка містить у собі основні елементи необхідні для запам’ятовування;

– блок введення-виведення – БВВ. БВВ здійснює комутацію виходів корпусу мікросхеми з внутрішньої конфігурованою логікою. БВВ підтримують більшість сигнальних стандартів введення-виведення, що існують в даний час;

– блок пам’яті. Кожен блок може бути сконфігурованим як двопортовий ОЗП ємністю 18 кбіт;

– блок помножувача. Вбудований помножувач 18х18 біт;

– цифровий блок управління синхронізацією – DCM (Digital Clock Manager).

Варто детально розглянути в подальшому проектуванні характеристики ПЛІС із сімейства Spartan – 6, Virtex, а також Artix – 7 і Kintex – 7.

У зв'язку підвищення складності ПЛІС відбувається така ситуація, що інколи навіть недостатньо ознайомлення з документацією для вибору ПЛІС для конкретного проєкту з урахуванням особливостей архітектури та характеристик встановлених на кристалі елементів.

Виходячи з цього, перед самим процесом проєктування моделі ПЛІС логічного нейропроцесора буде проведена робота по визначенню найбільш значущих властивостей ПЛІС і тих якостей, які важливі для побудови спеціалізованого нейропроцесора для виконання логічних операцій.

Напротязі довго часу, до 2005 року, основним рішенням для створення логічних комірок ПЛІС була конфігурація «4-входової таблиці істинності + тригер» (4-LUT + FF). Очевидною реалізацією такої системи був 16-розрядний зсувний регістр або однопортовий синхронний ОЗП ємністю 16x1. Дві сусідніх комірки можуть бути налаштовані як двопортовий ОЗП 16x1 із функцією запису та читання по одній адресі, а також у цей самий часможе відбуватись читання за іншою адресою. При необхідності великої кількості ОЗП, він будується на базі декількох комірок. Такий ОЗП розподілено по площі ПЛІС і тому називається розподіленим (Distributed RAM) [9].

Після появи нової структури ПЛІС Virtex – 5 у 2005 році, яки розробляються за технологією 65 нм, архітектура комірок почала еволюціонувати більш швидкими темпами. У Virtex – 5 замість стандартної конфігурації комірок 4-LUT + FF була введена конфігурація 6-LUT + FF, а починаючи з сімейств Virtex – 6, Spartan – 6 і до нині ще більш прогресивна конфігурація 6-LUT + 2 · FF (рис. 4.3–4.5).

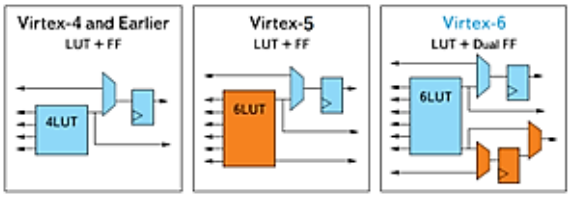


Рис. 4.3. Еволюція архітектури логічної комірки сімейства Virtex

Відповідно 6-входова таблиця істинності (6-LUT) може конфігуруватись у вигляді 32-розрядного зсувного регістру або як блок однопортової пам'яті з організацією 64x1 (рис. 4.3) [1, 2]. Тобто можливість створити нові входи з'явилась через збільшення конфігураційної пам'яті, без використання нових мультиплексорів. Також комбінація «генератор + мультиплексори» дає змогу створити не будь яку логічну функцію, у протилежність від фізично зростаючої місткості пам'яті призначеної для зберігання таблиць істинності.

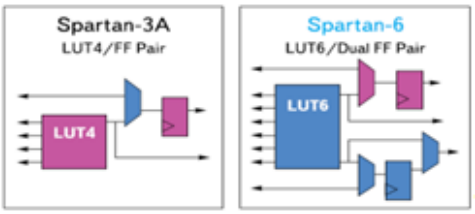


Рис. 4.4 – Еволюція архітектури логічної комірки сімейства Spartan

Значення переходу від 4-LUT до 6-LUT важко переоцінити. Перш за все, для складних проєктів істотно зменшується число послідовних комірок, що реалізують функцію для великого числа входів. Економія комірок при цьому не так важлива, як важливо, що в ланцюгах комбінаторної логіки виявляється в середньому в півтора рази менше комірок, що в свою чергу зменшує затримку сигналу у стільки ж разів.

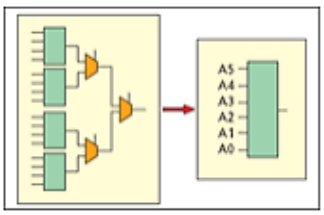


Рис. 4.5 – Зміни в реалізації розподіленої пам'яті на 6-LUT

Під час перевірки можливостей ПЛІС Virtex – 5 у навантажених проектах логічними функціями показало зниження кількості логічних рівнів на 25–30% [29]. На рис. 4.6 проілюстрована проблема, пов’язана напряму з кількістю рівнів логіки. Путь сигналу з першого тригера до другого пролягає через дві LUT, а також по трасуванню ланцюга між двома LUT (додається затримка tTP). Звідси можна зробити висновок, що при зростанні логічних рівнів, час який буде потрібен на проходження схема значно зросте і через це знизить частоту ПЛІС в проекті.

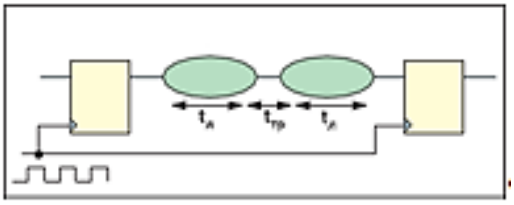


Рис. 4.6 – Проблема збільшення числа рівнів логіки.

Сімейства Virtex – 6, Spartan – 6, як уже було зазначено вище, мають ще більш прогресивну архітектуру комірок 6-LUT + 2 · FF. Варто підмітити, що змінились не тільки Virtex, а ще і Spartan, звідси ясно, що дешевші ПЛІС Spartan нового покоління володіють повнофункціональною логічною коміркою.

Другий тригер має застосування у глибокій конвертації та може використовуватись для підтримки режиму, коли генератор може використовуватись як пара генераторів із 5-ма входами.

В цілому, нова комірка являє собою вид ресурсу ПЛІС, який покращує характеристики проектів, котра виконує свою роботу без оперативного втручання програміста. Через прописані алгоритми у засобах синтезу конструкцій HDL, дозволяється використовування логічних генераторів із 6-ма входами та застосування другого тригеру. Розробник може реалізувати «найбільш щільне прилягання до апаратури» за допомогою реалізації вузлів у складі апаратних ресурсів [29].

Одним з важливих ресурсів ПЛІС для побудови логічного нейропроцесора є пам’ять. Стартуючи з лінійки Virtex – 5 місткість блоку

пам’яті збільшилась у два рази та дорівнює тепер 36 Кбіт. Проте, кожен блок може бути використаний як два незалежних блоки по 18 Кбіт. Додатково в складі блоку присутні ланцюги, що полегшують побудову модулів FIFO (черги) і пристрої корекції помилок. Логіка FIFO забезпечує прапори «FIFO повний», «FIFO порожній», а також прапори «FIFO майже повний», «FIFO майже порожній» з програмованими порогами спрацьовування (рис. 2.10). Пам’ять може використовуватися з шириною даних від 1 до 72 розрядів. У 72-розрядному варіанті доступний тільки так званий «простий двопортовий режим», в якому один порт використовується для читання, а інший – для запису. В інших випадках доступний «істинний двопортовий режим», при якому порти можуть використовуватися як для читання, так і для запису.

Рис. 4.7 – Архітектура блокової пам’яті Virtex – 5

Частота роботи блокової пам’яті в Virtex – 5 досягає 550 МГц, а її ємність – майже 12 Мбіт. Для порівняння: в Virtex – 6 максимальна ємність блокової пам’яті становить 38 Мбіт, а в Virtex – 7–68 Мбіт. Якісні зміни, що відбулися в способі організації блокової пам’яті, відповідаю загальної тенденції збільшення пропускної здатності інтерфейсів обміну між пам’яттю і арифметичними пристроями. За оцінкою виробника, пристрої Virtex – 5 в 4.4 рази перевершують своїх попередників Virtex – 4 по пропускній здатності пам’яті, що важливо для побудови систем цифрової обробки сигналів, а також високопродуктивних паралельних обчислень.

З блоковою пам’яттю зазвичай асоційовані апаратні помножувачі. Як правило, блокова пам’ять зберігає коефіцієнти цифрових фільтрів. Однак зі

збільшенням рівня інтеграції і поширенням софт-процесорів з'явилася можливість реалізувати на тому ж кристалі не тільки блоки логічного нейропроцесора, але і керуючий нейропроцесор (наприклад MicroBlaze). Тому кількість блоків DSP зростає трохи повільніше, ніж кількість блоків пам'яті.

Основною операцією в задачах виконання логічних операцій є додавання та множення з накопиченням (Multiply-and-Accumulate – MAC).

Структурна схема блоку DSP48E сімейства Virtex – 5 показана на рис. 4.8.

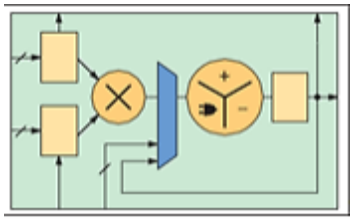


Рис. 4.8 – Блок DSP48E сімейства Virtex – 5

У цьому блоці замість множення двох 18-розрядних операндів стало доступно множення 25-розрядного числа на 18-розрядне. Акумулятор замінений 48-розрядним суматором. Крім того, можливо каскадування блоків DSP із застосуванням виділених трасувальних ресурсів в межах однієї колонки. Нарешті, доданий незалежний 48-розрядний вхід і можливість підсумовування трьох операндів в межах одного блоку. Відзначимо також підвищення тактової частоти до 550 МГц (500 МГц у Virtex – 4) і приблизно 40% – ве зниження споживаної потужності в порівнянні з Virtex – 4 [9, с. 95–109].

Сімейства Virtex – 6 і Spartan – 6 мають більшу кількість блоків DSP (в молодшій ПЛІС Virtex – 6 SXT їх більше ніж в будь-якій з мікросхем Virtex – 5 SXT). Слід зазначити якісні зміни цих блоків, які стали називатися XtremeDSP (DSP48E1S). Нововведенням в порівнянні з сімейством Virtex – 5 є попередній суматор (pre-adder). Цей модуль є хорошим прикладом технічного рішення, яке при невеликих апаратних витратах дозволяє отримати суттєвий виграш при реалізації певного класу алгоритмів, а саме цифрових фільтрах з симетричними коефіцієнтами. У таких проєктах маємо двократне зменшення кількості необхідних для реалізації блоків XtremeDSP [27, с. 169–180].

У табл. 4.1 показані зведені характеристики продуктивності в задачах логічного нейропроцесора для FPGA сімейств Virtex різних поколінь.

Таблиця 4.1 - Продуктивність FPGA Xilinx в задачах виконання логічних операцій

Показник	Virtex – 4	Virtex – 5	Virtex – 6
Тактова частота, МГц	500	550	600
Кількість блоків	512	1056	2016
Пікова продуктивність, GMAC / s	256	580	1200

Всі частоти вказані для виконання ПЛІС з найбільш швидким класом швидкості (speed grade). З таблиці видно, що пікова продуктивність зростає в 2 рази при переході до кожного нового сімейства, що досягається насамперед відповідним дворазовим зростанням кількості блоків цифрової обробки сигналів. Тактова частота при цьому зростає приблизно на 10% щодо попереднього сімейства. Це означає, що отримання максимальної віддачі від FPGA серії SX можливо при глибокому розпаралелюванні процесів обробки, наприклад при реалізації багатоканальних процесів високих порядків.

Основні нововведення торкнулись блоків введення/виведення, а саме їх пропускна здатність виросла із 1.2 Гбіт/с у попередньому поколінні ПЛІС Virtex – 5, до 1.4 ГБіт/с у серії Virtex – 6. З першого погляду можна сказати, що це не значне поліпшення, однак завдяки цьому була реалізована одна функція – динамічне керування вхідним імпедансом, що у свою чергу допомагає у реалізації інтерфейсу HSLVDCI (High-Speed Low Voltage Dynamically Controlled Impedance). Завдяки йому досягається зниження використаної енергії через динамічне відключення термінаторів (Рис. 4.9).

На зображеному рисунку відображена реалізація двобічної шини. Для процесу читання, необхідне використання резисторів pull-up і pull-down, реалізація яких відбувається у параметрах блоку введення/виведення. Однак, для операції запису необхідність у термінаторах відпадає, хоча у минулих поколіннях FPGAвони все ще приєднувались до буфера виходу, що і

підвищувало енергозатрати. Саме динамічне управління має змогу відключити термінатори, коли деякий вхід перемикається на режим виходу.

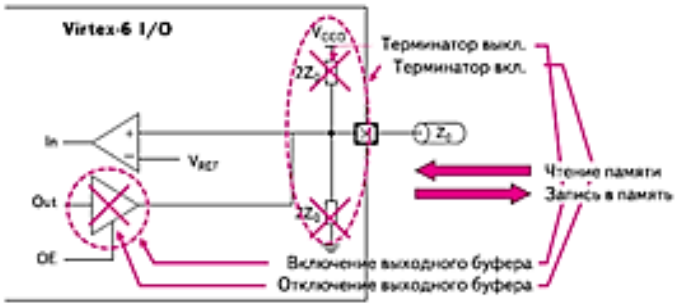


Рис. 4.9 – Динамічне вимикання термінаторів під час роботи із двонаправленими шинами

Відносно обсягу доступних контактів для користувача, ситуація наступна: ПЛІС від фірми Xilinx, з 4-ї версії видають приблизно від 320 до 960 контактів, у 6-ій версії ця кількість збільшилась до значень 360-1200, та 7-ма серія зробила стрибок у архітектурі та може виділяти від 840 до 3600 контактів.

Відносно систем виконання логічних операцій кількість призначених для користувача контактів не є визначальними показниками при виборі моделі ПЛІС.

Проведемо порівняльний аналіз реалізації логічного нейропроцесора для виконання логічних операцій на ПЛІС серії Spartan – 3 і Kintex – 7. VHDL опис логічного нейропроцесора наведено в додатку Б.

Як об’єкти для порівняння біли обрані дві XC3S200–4-FT256 (3-ї серії Spartan) і XC7K160T – 3-FBG484 (7-й серії Kintex). Основними критеріями вибору цих мікросхем були:

– здатність відслідкувати зміни у тех-процесі даних ПЛІС, а самі від мікросхем у 90 нм до сучасних 28 нм;

– можливість порівняти й оцінити сучасні ПЛІС за можливістю реалізації на них систем з використанням логічних операцій.

Не менш важливим фактором є опис даних моделей у бібліотеці ISE WebPack версії 14.2 від фірми Xilinx. У більшій степені це відстежується у 7-й

серії ПЛІС з того, що у даній бібліотеці описані переважно найновіші мікросхеми.

Порівняння буде опиратись на звіт від синтезатора XST, який вбудований до ISE WebPack. В Додатку Ж наведені звіти синтезу проекту логічного нейропроцесора для виконання логічних операцій з використання ПЛІС XC3S200–4-FT256 та XC7K160T – 3-FBG484.

Проаналізовавши звіт синтезатора, можна зробити висновок, що неважлива модель ПЛІС, бо необхідні схемові елементи (2х-портовий ОЗП, ПЗП, помножувач, акумулятор) були чітко визначені. Звіт синтезу проекту на ПЛІС XC7K160T – 3-FBG484 відображає дані по використанню мультиплексорів, що знаходяться в логічних комірках поряд з таблицею істинності (LUT) і тригерами (FF).

Бачимо, що конфігурація логічної комірки 6-LUT + 2 · FF дає змогу знизити кількість використаних LUT-ів, також дозволяє зменшити використання комірок із 139 одиниць до 64. Беручи до уваги те, що присутні 2 тригери у кожній логічній комірці, то це дає змогу додати тригер на рівні HDL опису, задіюючи при цьому комірки з невикористовуваними або на половину використовуваними тригерами в своєму складі. Таких комірок в даному проекті вийшло 75 штук, що разом з 6-входовою таблицею істинності (6-LUT) дає можливість обрати мікросхему XC7K160T – 3-FBG484 в малих проектах, пов’язаних з логічною обробкою даних, як для проектування спеціалізованого нейропроцесора керування електроприводом.

Що стосується звітів синтезатора стосовно систем виконання логічних операцій, відзначимо різко зменшену кількість блоків BRAM, котрі здатні підтримувати логіку FIFO. Реалізація множення з накопиченням реалізується тепер на спеціальному блоці (XtremeDSP) і не потребує використання інших суматорів чи регістрів, на відміну від XC3S200–4-FT256. Також важливим аспектом сучасної ПЛІС XC7K160T – 3-FBG484 є те що в ній збільшена тактова частота праці логічного нейропроцесора для виконання логічних операцій майже на 20 МГц, що є хорошим показником навіть для такого відносно невеликого пристрою.

Отже, мікросхема ПЛІС XC7K160T – 3-FBG484 7-ї серії Kintex, котра не представляється як основна у лінійці, має конструктивно такі характеристики: 325 блоків пам’яті BRAM із загальною ємністю 11.7 Мбіт; 600 спеціалізованих блоків логічного нейропроцесора DSP48E1S; трансивери з пропускною спроможністю до 12.5 Гбіт / с. Усі ці ключові характеристики з досить невисокою вартістю мікросхеми, в порівнянні з сімейством 7-й серії Virtex, робить обрану мікросхему найкращою для реалізації систем на ній з виконанням логічних операцій. Інформація, отримана в результаті проведеного порівняльного аналізу, допоможе в подальшому для постановки завдання проектування моделі ПЛІС для побудови на ній системи логічного нейропроцесора для виконання логічних операцій, та генерації вхідної черги показників та вдалих/невдалих фактів.

4.2. Розробка програмної моделі прототипу нейропроцесора

Рішення завдання проектування моделі ПЛІС для реалізації проекту керування електроприводом є непростим процесом. Пов’язано це насамперед зі зростанням складності ПЛІС, а також диференціюванням підсімейств, спрямованих на реалізацію конкретних завдань. Тому в даному проекті буде розглядатись значно вузьке завдання, якщо точніше, то це відбір ПЛІС для реалізації на ній системи логічного нейропроцесора для виконання логічних операцій під час керування електроприводом.

Підсумуємо важливі характеристики ПЛІС, які можуть знадобитись при реалізації поставленої задачі:

– технологія виробництва;
– пристрій секції;
– кількість логічних комірок;
– напруга живлення ядра;
– ємність блокової пам’яті;
– кількість блоків цифрової обробки сигналів;
– максимальна кількість доступних варіантів;

– тип приймачів;

– вартість мікросхеми.

За зовнішні показники якості були прийняті:

– площа кристала;

– тактова частота роботи ПЛІС;

– продуктивність в задачах цифрової обробки сигналів;

– пропускна здатність послідовних інтерфейсів.

Безпосередньо пов’язані з зовнішніми показниками якості та основні вимоги при виборі моделі ПЛІС:

– технологія виробництва;

– ємність блокової пам’яті;

– кількість блоків цифрової обробки сигналів;

– тип приймачів.

Зробимо деякі пояснення стосовно зовнішніх показників, які напряму пов’язані з вибором ПЛІС. Коли впроваджуються нові технології під час випуску нових версій ПЛІС з тими же розмірами кристалу, це дозволяє впровадити більші ресурси та збільшити тактову частоту в порівнянні із минулими поколіннями.

Через ці зміни надається можливість збирати високопродуктивні системи для виконання логічних операцій, що складаються з сотень (а іноді і тисяч) спеціалізованих ПЛІС у відносно невеликих за розміром корпусах.

Ємність блокової пам’яті BRAM грає не останню роль в роботі систем виконання логічних операцій. У сучасних системах блокова пам’ять зберігає коефіцієнти цифрових фільтрів, а також є буфером для вхідних даних – цьому сприяє підтримка логіки FIFO в блоках BRAM. Тому достатня ємність блокової пам’яті для роботи системи виконання логічних операцій середнього рівня в даний час складає близько 20–30 Мбіт [12, с. 94–102].

Ще одним важливим параметром є наявність високошвидкісних приймачів (трансиверів). Це дозволяє реалізовувати на їх базі такі інтерфейси як Gigabit Ethernet, SATA і PCI Express. Наприклад, для реалізації PCI Express 3.0, що працює на швидкості до 8 Гбіт / с потрібні трансивери GTH, що

забезпечують пропускну здатність даних до 13.1 Гбіт / с в Virtex – 7, а для останньої 4-ї версії (PCI-E 4.0 до 16 Гбіт / с) необхідні трансивери GTZ 28 Гбіт / с, які є тільки в самих старших ПЛІС Virtex – 7 [12].

Найбільш значущим показником при побудові систем виконання логічних операцій є продуктивність в задачах. Щоб забезпечити високе значення даного параметра, ПЛІС повинна мати достатню кількість спеціалізованих блоків DSP48E1S. В даний час цей показник знаходиться на рівні 800 і більше блоків.

У таблиці 4.2 наведено порівняння мікросхем ПЛІС серій Virtex – 5, Virtex – 6 і Virtex – 7 з однаковою площею кристала (розмір корпусу) за основними характеристиками [12, с. 94–102]. Всі ПЛІС можна використовувати в проєктуванні логічного нейропроцесора для виконання логічних операцій.

Таблиця 4.2 – Зведена таблиця характеристик ПЛІС різних технологій виробництва

Параметр / Модель ПЛІС	XC5VSX95T (Virtex – 5)	XC6VSX475T (Virtex – 6)	XC7VX690T (Virtex – 7)
Технологія виробництва	65 нм	40 нм	28 нм
Кількість комірок (4-LUT + FF)	94 тис.	476 тис.	693 тис.
Загальна ємність BRAM	8.7 Мбіт	38 Мбіт	52.9 Мбіт
Кількість блоків DSP48E1	640	2016	3600
Розмір корпусу, мм	35 · 35 мм		
Макс. Тактова частота	до 550 МГц	до 600 МГц	до 1 ГГц
Пікова продуктивність, GMAC / s	380	1200	2000
Тип і пропускна здатність приймачів GTP	GTP до 3.75 Гбит / с	GTX до 6.5 Гбит / с	GTH до 13.1 Гбит / с
Максимальна кількість доступних варіантів	640	840	1000

Спочатку наведемо зразок тестової програми для перевірки результатів реалізації VHDL моделі під певну ПЛІС. В таблиці 4.3 подаємо варіант тестової програми.

Лістинг програми для тестування:

```
lab1:
lw $a0,0 ($zero)
lw $a1,1 ($zero)
add $a0, $a0, $a1
sw $a0, 0 ($zero)
beq $a0, $zero, lab1
lab2:
beq $a0, $a0, lab2
or $a0, $a0, $a0
or $a0, $a0, $a0
```

Таблиця 4.3 – Набір інструкцій для тестової програми

Адреса	Директива	Мітка	Інструкція	Код
	. text			
		start:		
00			lw $a0,0 ($zero)	0x8c040000
04			lw $a1,1 ($zero)	0x8c050001
08			add $a0, $a0, $a1	0x00852020
0c			sw $a0, 0 ($zero)	0xac040000
10			beq $a0, $zero, lab1	0x1080fffb
		label:		
14			beq $a0, $a0, lab2	0x1084ffff

Інструкція beq $a0, $a0, lab2 виключає подальші зміни програмного лічильника (pc). Тобто дана інструкція програми завершує її виконання.

Коли комірка пам’яті з абсолютною адресою 0 містить у собі код числа +2, а сусідня комірка 1 – код числа (-2), тоді можна зробити висновки щодо послідовності змін у програмному лічильнику під час опрацювання тестової програми: 00, 04, 08, 0c, 10, 00, 04, 08, 0c, 10, 14, 14, 14, 14.

За допомогою часової симуляції ПЛІС у віртуальній VHDL моделі комп’ютера ми повинні отримати задану послідовність інструкцій у лічильнику.

Щоб одержати машинні коди програми, нам потрібно додати її до нашого симулятора машинних інструкцій Pcspim. Вміст даних кодів мусять містити у собі VHDL модель нейропроцесора, яку ми розробляємо.

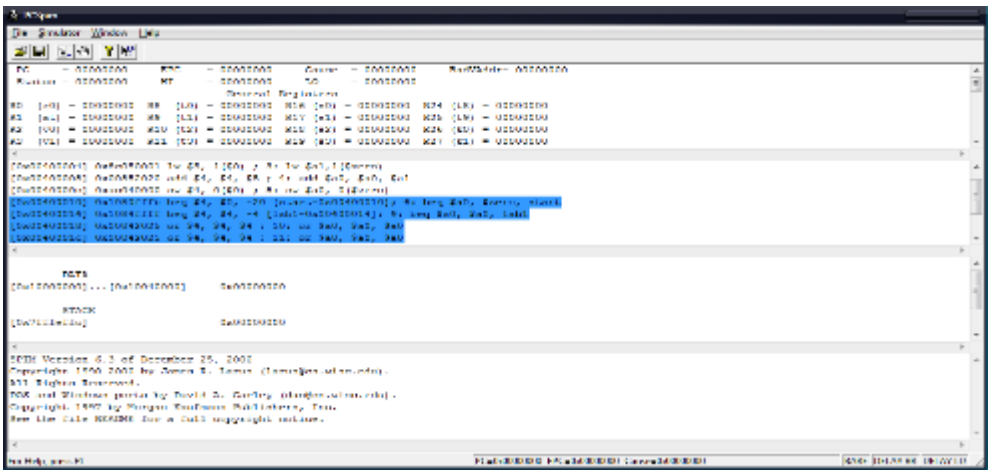


Рис. 4.10 – Вікно симулятора із завантаженою тестовою програмою

Через реалізацію архітектури відмінної від оригінальної архітектури SPIM, симуляція нашої тестової програми не зможе виконатись. Звідси виходить те, що треба використовувати стандартний симулятор RISC MIPS архітектури для формування нашої тестовою програми, адже це буде можливо коли наш проект буде містить у собі дані і інструкції, котрі відповідають оригінальним.

Для використання симулятором правильних мишиних кодів, треба провести конфігураційні дії над ним, насамперед перед завантаженням до нього тестової програми. Для цього потрібно налаштувати режим bare (що позначає чисту систему, з відсутністю програмної підтримки), конфігураційне вікно Settings має прийняти наступний вигляд:

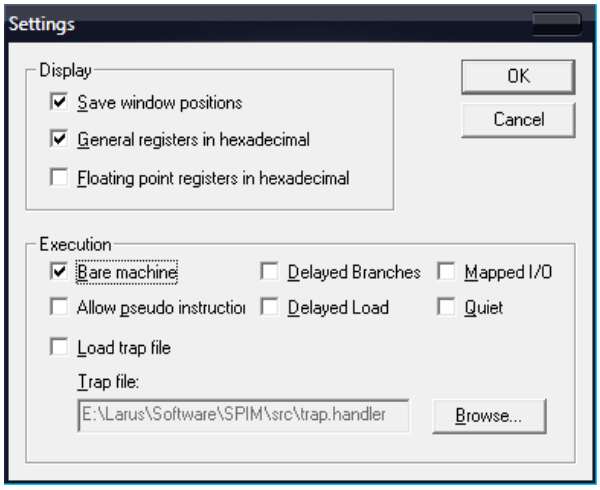


Рис. 4.11 – Конфігурування програмного симулятора Pcspim

Загалом, синтезована структура прототипу наближена до відомої структури логічного нейропроцесора. Але в оригінальну архітектуру логічного нейропроцесора ми спеціально додали деякі зміни, для того щоб полегшити імплементування у ПЛІС Spartan, самі зміни наведені нижче:

– формат даних має довжину є 8 бітів, а не 32 біти;

– трактування інструкцій було змінено навпаки до оригіналу, щоб можна біло використати наявні симулятори та асемблери, однак формат самих інструкцій змінам не підлягав;

– були реалізовані лише інструкції, котрі необхідні для виконання тестової програми;

– пам’ять даних була змінена до формату двох комірок замість 4, бо двох цих комірок вистачить для коректної роботи тестової програми;

– кількість регістрів зменшена з 32 до 8, а їх розрядність змінилась із 32 бітів до 8.

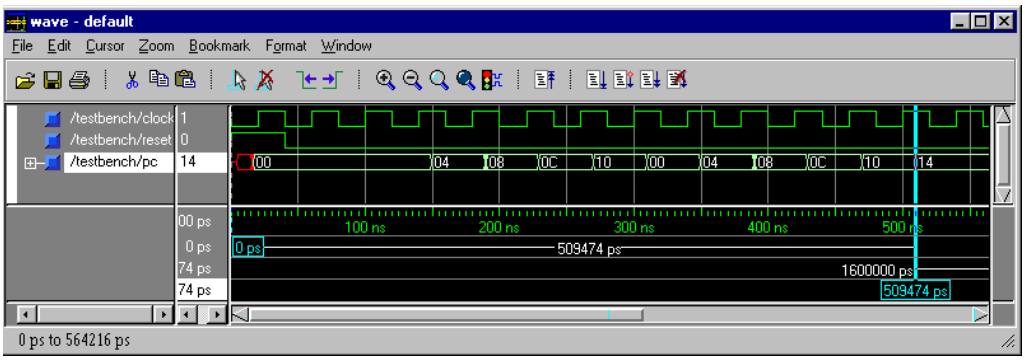


Рис. 4.12 – Часове симулювання моделі на рівні вентилів

Після введення тестової програми VHDL проімплементована. Наступним етапом потрібно провести симулювання за часовою лінією на рівні вентелів, щоб переконатись у правильності роботи тестової програми.

Розробимо програмні мікрокоди для модулів з яких складається VHDL модель логічного нейропроцесора для виконання логічних операцій під час керування електроприводом (Додаток Г).

Топ-файл – структурна архітектура усього логічного нейропроцесора, котрий має структуру, яка складається з наступних модулів: IF, ID, EXE, MEM та CTL (керування).

Модуль керування (CTL). Для проектованого логічного нейропроцесора може бути достатньо містити у собі комбінаційний пристрій керування. Бо через те, що інструкції обираються одноразово, послідовно та не змінюються у період виконання, то цього елементу буде достатньо.

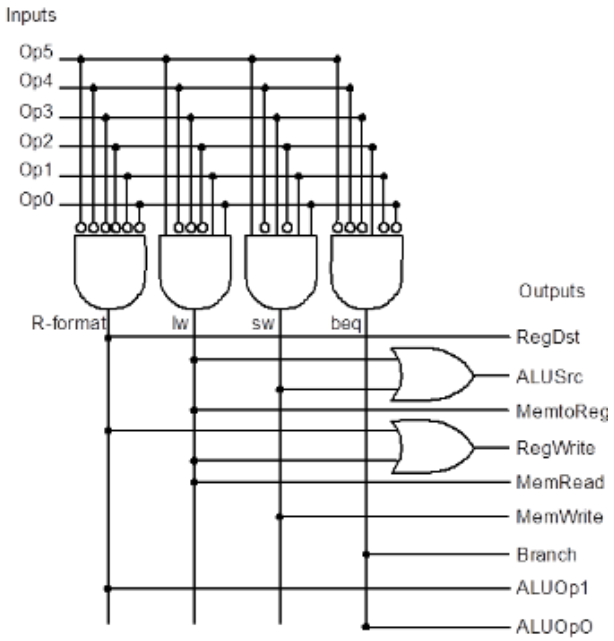


Рис. 4.13 – Структура комбінаційного модуля керування: на вході – біти інструкції, на виході – біти керування

Як R-format позначені машинні інструкції, що працюють з регістровими даними, наприклад: add r1, r2, r3; lw – це машинна інструкція завантаження слова з пам’яті напряму до регістра; sw – машинна інструкція для виконання збереження вмістимого регістра у комірці пам’яті; beq – машинна інструкція призначена для умовного переходу за ознакою рівності.

Можна відобразити усі команди, які можуть бути сформовані на виході, які генеруються вузлом керування: RegDest, ALUsrc, MemToReg, RegWrite, MtmRead, MemWrite, Branch, ALUOp1 та ALUOp2.

Модуль декодування інструкцій (ID) містить регістровий файл, інтерфейс якого до логічного нейропроцесора подає рис. 4.14.

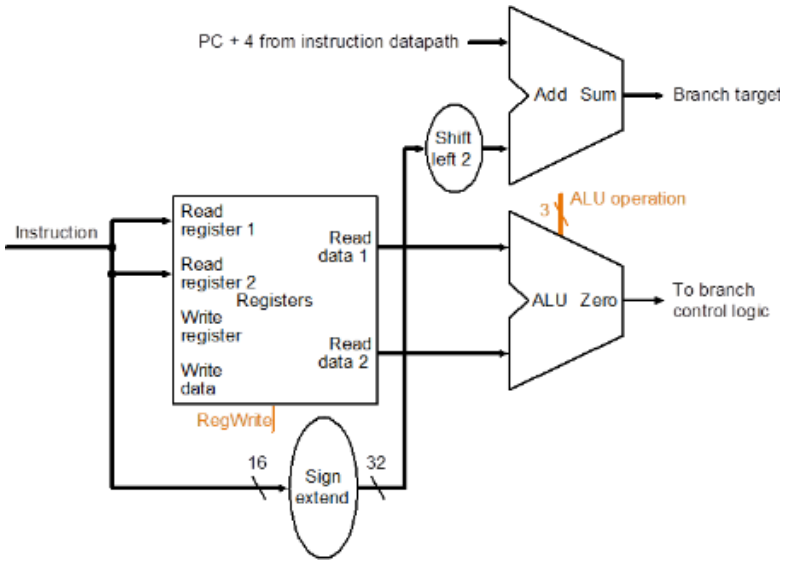


Рис. 4.14 – Інтерфейс регістрового файлу до логічного нейропроцесора

Лістинг програми декодування інструкцій наведений в Додатку З.

Модуль пам’яті (MEM). Даний модуль розроблений для збереження певних результатів чи операндів після виконання операцій. Цей модуль і виступає у ролі пам’яті даних. Лістинг програми MEM наведений в Додатку З.

Автоматично згенерований файл Test Bench задає нульові стартові значення всіх вхідних сигналів. Лістинг програми наведений в Додатку З.

4.3. Проєктування силової частини виконавчого механізму керування поливом

Основна його функція – управління потоком струму, тобто правильний перерозподіл між накопичувальними елементами і найголовніше запобігання перенапруги двигуна.

На практиці, всі функції контролера можна описати в наступних пунктах:

– управління підключенням ШІМ-перетворювача до накопичувача;

– різні варіанти живлення двигуна;

Всі ці функції ШІМ-перетворювача служать одному – збереженню робочих характеристик двигуна.

На першому місці по використанню і популярністю стоять контролери з використанням ШІМ контролера. Використання такого допоміжного механізму дозволяє більш ефективно використовувати накопичувач за рахунок більш якісного живлення двигуна.

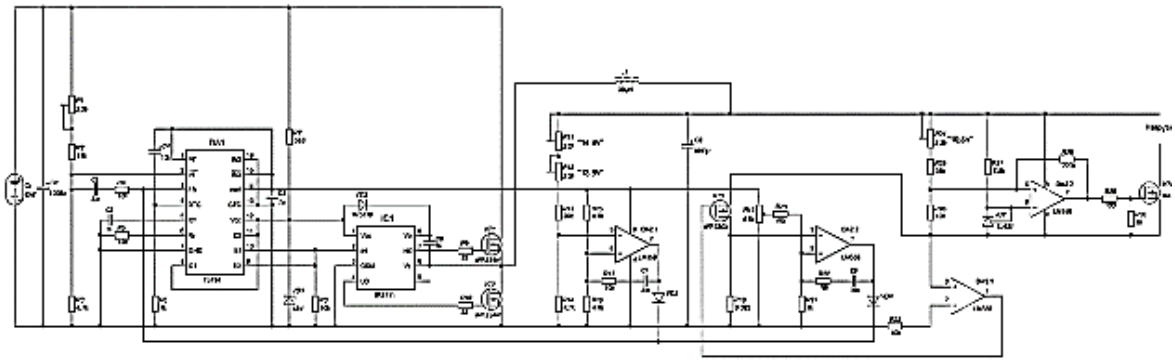


Рис. 4.15 – Принципова електрична схема ШІМ-перетворювача

Контролер для ШІМ-перетворювача випускається в різних варіантах. Складність даного пристрою залежить від наявності вбудованих функцій. Спектр подібної продукції досить широкий, починаючи від найпростіших різновидів, здатних працювати виключно в автоматичному режимі, і до складних промислових варіантів.

Існують спеціально створені для побудови джерела живлення, мікросхеми TL494 забезпечують розробнику розширені можливості при конструюванні схем управління ДЖ. Прилади TL494 включають в себе підсилювач помилок, вбудований регульований генератор, компаратор регулювання мертвого часу, тригер управління, прецизійне джерело опорної напруги (ИОН) на 5В і схему управління вихідним каскадом. Підсилювач помилки здатен формувати синфазний струм у діапазоні від 0,3... (Vcc – 2) В. Однак компаратор регулювання мертвого часу завжди використовує постійний

зсув, який обмежує мінімальну тривалість мертвого часу величиною приблизно 5%.

Допускається синхронізація вбудованого генератора, з допомогою підключення виводу R до виходу опорної напруги і подачі вхідної пилкоподібної напруги на вивід C, що використовується при синхронній роботі декількох схем ДВЖ.

З використанням схеми вихідного каскаду із загальним емітером чи емітерного перетворювача можна використовувати незалежні формувачі виведення на транзисторах. Через це даний каскад мікросхем працює в однотактному або двотактному режимі з можливістю вибору режиму з допомогою спеціального входу. Вбудована схема контролює кожен вихід і забороняє видачу здвоєного імпульсу в двотактному режимі.

Особливості:

– Повний набір функцій ШІМ-керування
– Вихідний струм: 200мА
– Можлива робота у одно- або два-тактному режимах
– Вбудована схема придушення здвоєних імпульсів
– Широкий діапазон регулювання
– Вихідна опорна напруга: 5В + -05%

Характеристики:

– Напруга живлення: 42 В.
– Напруга на колекторі вихідного транзистора: 42 В.
– Струм колектора вихідного транзистора: 500 мА.
– Діапазон вхідної напруги підсилювача: від – 0,3 В до +42 В.
– Потужність, що розсіюється (при t <45 °C): 1000 мВт.
– Діапазон температур зберігання: від – 55 до +125 °C.
– Діапазон робочих температур навколишнього середовища: від – 25 до +85 °C.

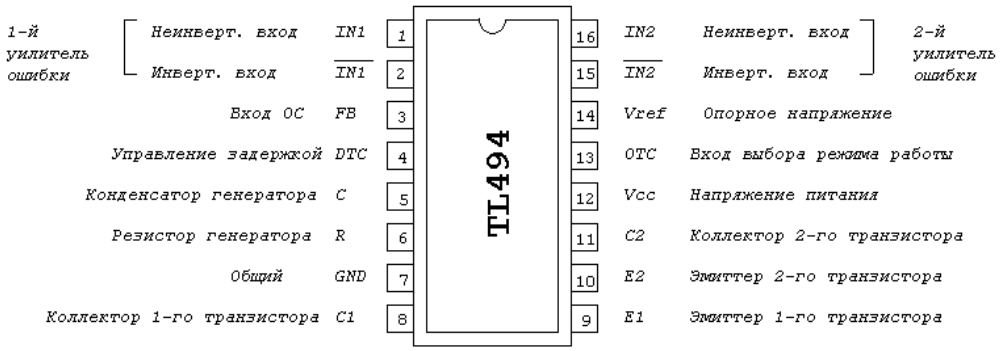


Рис. 4.16 – Призначення виводів мікросхеми TL494

Мікросхема TL494 представляє собою деякий ШІМ-контролер з використанням імпульсного живлення, який може працювати лише на певній частоті, і включає в себе всі необхідні для цього блоки. Вбудований генератор пилоподібної напруги вимагає для встановленої частоти тільки двох зовнішніх компонентів R і C. Частота генератора визначається за формулою:

$$f_{OSC}=\frac{1.1}{R*C}$$

Модуляція ширини вихідних імпульсів досягається порівнянням позитивної пилкоподібної напруги, одержаної на конденсаторі C, з двома керуючими сигналами (див часову діаграму). Логічний елементи АБО-НЕ збуджує вихідні транзистори Q1 і Q2 тільки тоді, коли лінія тактованого вбудованого тригера знаходиться в низькому логічному стані. Це відбувається тільки тоді, коли амплітуда пилоподібної напруги вище амплітуди керуючих сигналів. Отже підвищення амплітуди керуючих сигналів викликає відповідне лінійне зменшення ширини вихідних імпульсів. Під керуючими сигналами розуміють напругу, яка була вироблена схемою регулювання мертвого часу (вивід 4), підсилювачами помилки (вивід 1, 2, 15, 16) і ланцюгом зворотного зв’язку (вивід 3).

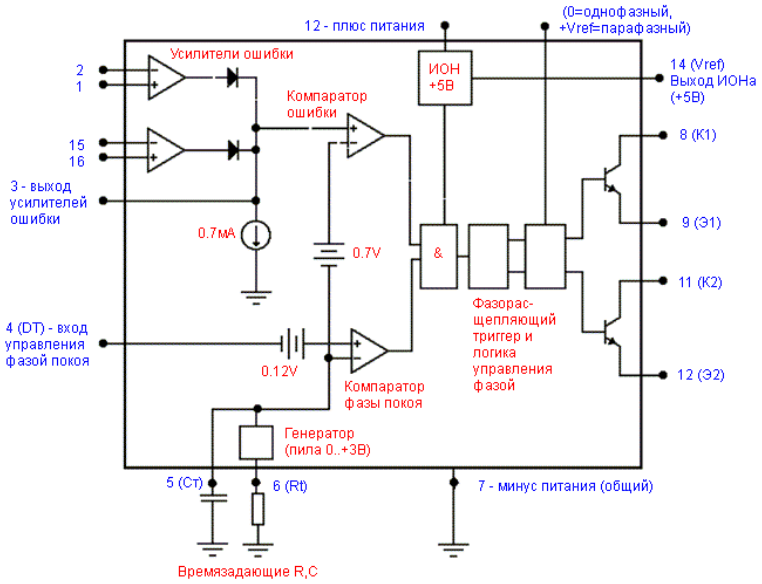


Рис. 4.17 – Блок схема TL494

Зсув для порту введення у компаратора дорівнює 120мВ, який дає змогу обмежити найменший мертвий час на порті виведення не більше ніж 4% від часу усього циклу пилообразної напруги. В результаті максимальна тривалість робочого циклу складає 96% у тому випадку, якщо вивід 13 заземлений, і 48% у тому випадку, якщо на вивід 13 подається опорна напруга.

Збільшити тривалість мертвого часу на виході можливо якщо подавати на вивід 4 (регулювання мертвого часу) постійну напругу в діапазоні 0..3,3В. ШІМ-компаратор займається регулюванням шириною вхідних імпульсів. Обидва підсилювача помилки мають вхідний діапазон синфазного сигналу від – 0,3 до (Vcc – 2,0) і можуть бути використані для отримання напруги з виходу джерела живлення. Порти виведення підсилювачів помилки можуть мати активний великий рівень напруги та можуть бути об’єднані функцією АБО на неінвертуючому вході ШІМ-компаратора. При використані даної конфігурації підсилювач помилки, який буде вимагати мінімального часу для включення виходу, буде признаний домінуючим. При розрядці конденсатора С на виході компаратора буде формуватися позитивний сигнал, котрий буде збуджувати тригер та блокувати транзистори Q1 і Q2, які працюють на введення. При створенні на вході виводу 13 опорної напруги (вибір режиму роботи), тригер безпосередньо буде управляти двома вихідними транзисторами в протифазі

(двотактний режим), при чому вихідна частота прийме значення половині частоти генератора. При відкритті обох транзисторів, вихідний формувач перейде до однотактного режиму, та ці трансформатори закриваються одночасно, коли потрібно щоб максимальний робочий цикл не перевищував 50%. Це потрібно, коли у конструкції трансформатора використовується обмотка з обмежувальним діодом, задачами якого є упокоєння перехідних процесів.

При необхідності у великих струмах можна перевести вихідні транзистори у паралельний режим, для переведення потрібно перемкнути вивід 13 на землю, який буде блокувати вихідний сигнал від тригеру. При застосуванні даної схеми частота буде залежить лише від генератора і чисельно буде дорівнюватись його частоті.

Джерело напруги у мікросхемі TL494 вбудоване та має напргу 5,0В, що забезпечує витікаючий струм до 10мА для зміщення зовнішніх компонентів схеми. Напруга має похибку 5% в діапазоні робочих температур від 0 до 70 °С.

Мостовий реверсивний чотирьохключовий драйвер IR2111

Відмінні особливості:

– управляючі канали розроблені для навантаженого функціонування;
– повністю працездатні до + 600В;
– нечутливий до негативних напруженням при перехідних процесах;
– стійкість до швидкості наростання напруги (dV / dt);
– діапазон напруги живлення драйверів 10… 20В;
– блокування при зниженні напруги;
– вхідна логіка з тригерами Шмідта з резисторами до мінуса;
– узгоджена затримка поширення для обох каналів;
– внутрішньо встановлена пауза при перемиканні каналів;
– вихід драйвера верхнього рівня у фазі зі входом;
– напруга зсуву VOFFSET не більше 600В;
– імп. Вих. Струм к. з Io ± 200 мА / 420 мА;
– вихідна напруга драйверів VOUT 10–20В;
– час вкл. / вимик. 850 / 150 нс;

– пауза 700 нс.

IR2111 – це драйвер призначений для МОП-транзисторів або IGBT-транзисторів, у яких залежні вихідні канали верхнього та нижнього рівнів, драйвер розроблений для мостових додатків. Власна HVIC-технологія і стійка до виключення КМОП-технологія можуть забезпечити створення монолітної структури. Логічні порти введення будуть сумісні із звичайними КМОП портами виведення.

Виходи драйверів відрізняються високим імпульсним струмом буферного каскаду, що зроблено для мінімізації зустрічної провідності драйвера. Внутрішня пауза при перемиканні передбачена, щоб уникнути наскрізних струмів.

Для керування N-канальним силовим МОП-транзистором чи IGBT-транзистором, у яких верхній рівень напруги може досягати 600В, використовується вихідний канал.

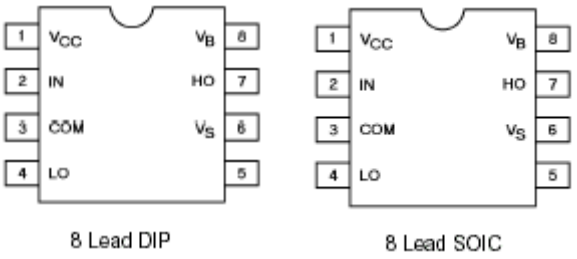


Рис. 4.18 – Розташування виводів

Опис виводів:

– IN- Логічний вхід управління виходами драйверів верхнього і нижнього рівнів, у фазі з HO;

– VB – Напруга живлення ключів верхнього рівня;

– HO- Вихід драйвера верхнього рівня;

– VS- Повернення живлення верхнього рівня;

– VCC- Живлення драйверів нижнього рівня і логіки;

– LO- Вихід драйвера нижнього рівня;

– COM- Повернення живлення нижнього рівня;

– транзистор IRFZ34N.

Характеристики:

– корпус – TO – 220AB;
– напруга пробою стік-витік 55 В;
– максимальна напруга затвора 20 В;
– опір у відкритому стані 40.0 мОм;
– струм стоку 26 А;
– заряд затвора 22.7 нКл.

Мікросхема LM358 має всередині свого корпусу 2 малопотужних оперативних підсилювачів з високим коефіцієнтом посилення і частотної компенсацією. Відрізняється низьким споживанням струму. Особливість даного підсилювача – полягає у тому, що завдяки ньому мікросхема може використовуватись у системах з напругою 3-32 вольти. Вихід має захист від короткого замикання.

Щодо логічної частини мікропроцесорного контролера, то всім необхідним вимогами для проєктування мікропроцесорного ШІМ-перетворювача відповідає мікропроцесорний комплект на великих інтегральних схемах підвищеної степені інтеграції Atmega32. Структурна схема мікроконтролера наведена на рисунку 4.19.

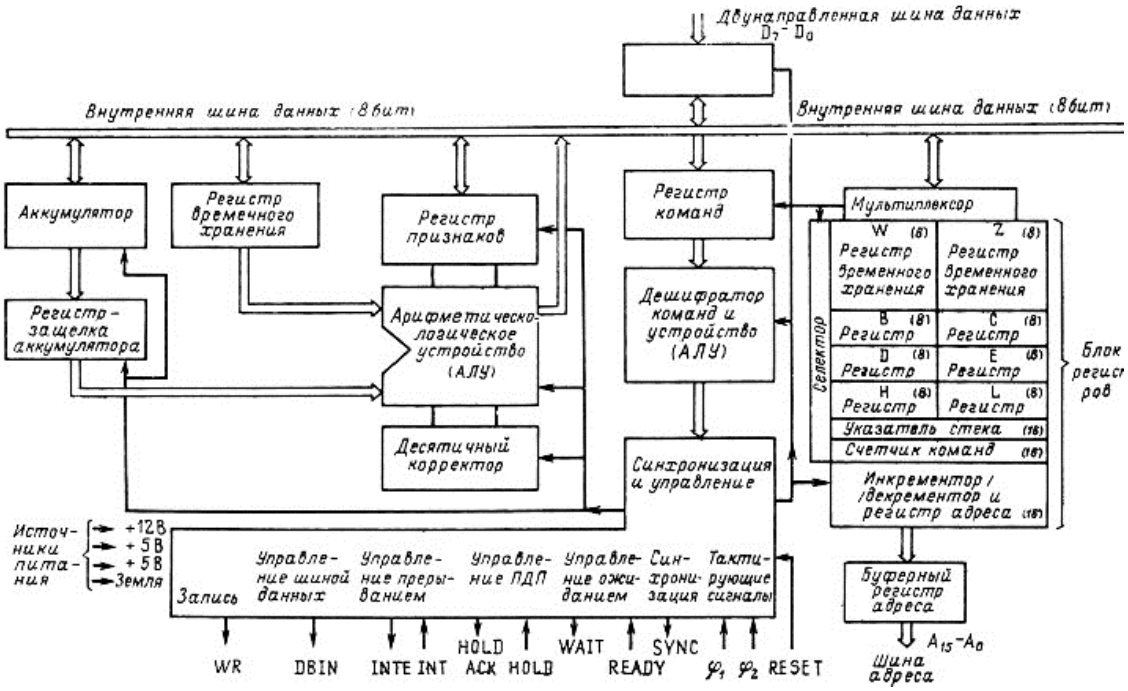


Рис. 4.19 – Схема структурна мікропроцесора

Електрична принципова схема мікропроцесорного ШІМ-перетворювача наведена на рисунку 4.20.

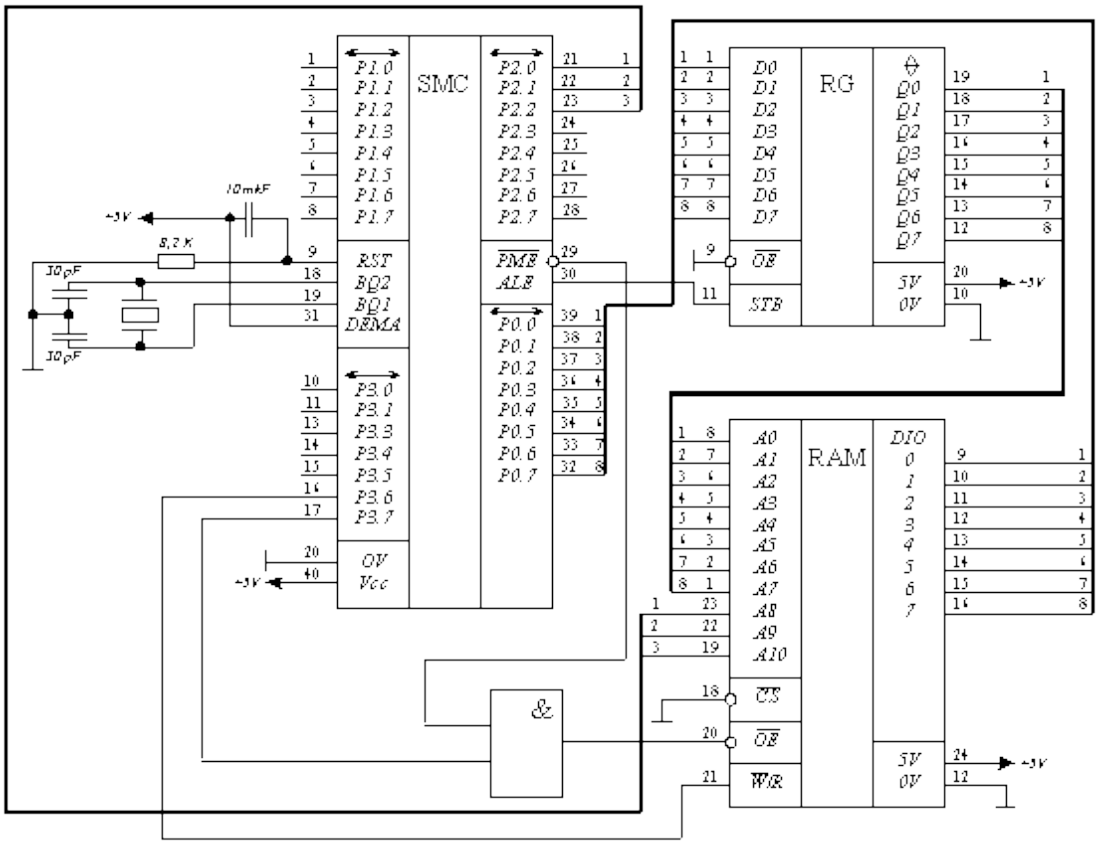


Рис. 4.20 – Схема електрична принципова мікропроцесорного ШІМ-перетворювача

Дільник напруги, що входить до складу мікропроцесорного контролера – це пристрій, в якому вхідна і вихідна напруга пов’язані коефіцієнтом передачі. У якості подільника напруги зазвичай застосовують регульовані опори (потенціометри). Плече між нульовим потенціалом і середньою точкою мають назву нижнього, а інше – верхнього. Розрізняють подільники напруги на лінійні та нелінійні. Подільники лінійного типу змінюють напругу опираючись на лінійний закон у залежності від вхідного. Ці дільники використовуються для завдання потенціалів і робочих напружень в різних аспектах системи. Напруга у нелінійних дільниках формується за коефіцієнтом нелінійності. Використовуються нелінійні подільники у функціональних потенціометрах. Опір буває активний і реактивним. Типова схема дільника напруги для мікропроцесорного ШІМ-перетворювача представлена на рисунку 4.21.

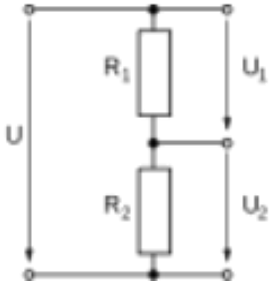


Рис. 4.21 – Схема резистивного дільника напруги

Найпростіший резистивний дільник напруги являє собою два послідовно включених резистора R1 і R2, підключених до джерела напруги U. Через те що резисторі під'єднані послідовно, то це вирівняє струм і він буде однаковим за першим правилом Кірхгофа. Падіння напруги на кожному резисторі відповідно до закону Ома буде пропорційно опору.

Центральний процесор Atmega32 володіє розрядністю даних 16 біт, розрядність адреси – 20 біт та тактову частоту до 5 МГц. Продуктивність даного процесора зросла на порядок у порівнянні із процесором K580UK – 80. Мікросхема Atmega32 являє собою одно кристальний 32-ти бітовий МП, виконаний по технології п-МОП. Кристал даної мікросхеми містить у собі –

29000 транзисторів, та характеризується розмірами 5,5х5,5мм. Atmega32 містить 14 16-бітових внутрішніх регістрів і утворює 32-бітову шину даних, яка слугує шляхом зв'язку муж портами введення/виведення та постійною пам'яттю. В свою чергу шина адрес містить у собі 20 розрядів, що дає змогу адресуватись до пам'яті ємністю до 1 Мб.

Комплект з мікропроцесором використовує дуже багато різноманітних мікросхем у різних сферах. В нашому проекті використовується декілька мікросхем, а саме:

– Atmega32 – центральний процесор
– К1810ГФ84 – генератор тактових сигналів
– К1810ВГ88 – системний контролер
– К1810ИР82 – регістр
– К1810ВА86 – шинний формувач

Простір пам'яті розділяється на сегменти по 64К, та у всякий час мікроконтролер може звертатись до ділянок 4 сегментів, які програмно вибрали в якості поточних. Сегментація пам'яті дає змогу розробити простий механізм обчислення фізичних адресів та допомагає модульному проектуванню програмного забезпечення, що спрощує проектуванню та наладку.

Для досягнення зменшеної кількості виводів у мікросхеми, були згруповані лініями даних та сформовані у шину даних, 16 молодших розрядів. Останні чотири адресні лінії були мультиплексовані з лініями стану. Для використання даних ліній у системі використовуються зовнішні схеми.

Для операцій введення/виведення беруться до опрацювання 8-ми або 32-х бітні адреси, так що при доступу до основної пам'яті мікроконтролера можуть комутувати з портами, ємність яких у сумі дає 64К. Пам'ять мікроконтролера Atmega32 фізично організована у вигляді двох банків – молодшого і старшого.

Отже, вибраний мікропроцесорний комплект виконує усі вимоги, які формуються розроблювальним пристроєм та його використання буде доцільним для розробки мікропроцесорного ШІМ-перетворювача.

Модуль клавіатури призначений для вводу команд з клавіатури. В системі може бути декілька таких модулів. Електрична принципова схема модуля

наведена на рисунку 4.22. Живлення модуля здійснюється від мережі системи «+5В». Основою модуля кодового доступу є мікроконтролера DD8 типу PIC16F84–041 / p. Клавіатура контролера – це набір кнопок 0–9 та * і #. Частота роботи процесора – 4 МГц, задана кварцовим резонатором ZQ3. Конденсатори С5 – С6 потрібні для коректної роботи задаючого генератора на кварцовому резонаторі. При натисканні на кнопки клавіатури, звуковий резонатор ZQ4 видає 1 короткий імпульс. Лістинг програми мікроконтролера PIC16F84–041 / p наведено в додатку К.

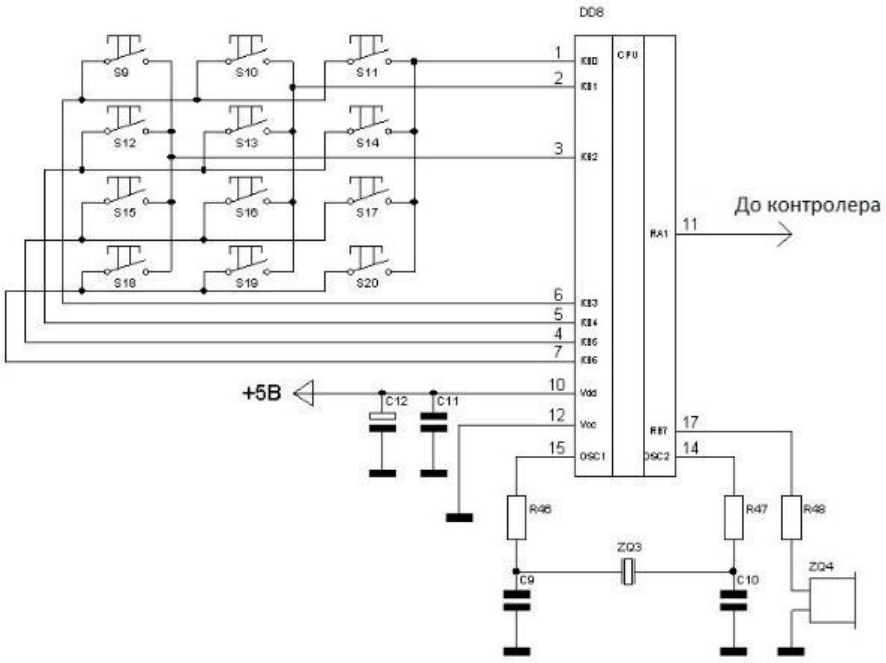


Рис. 4.22 – Принципова електрична схема модуля клавіатури

У даній проєктній роботі обрано алфавітно-цифровий рідкокристалічний індикатор (LCD) на основі контролера з HD44780. Він відображає символи ASCII з кодами від 32 до 122 та деякі інші, залежно від виробника та моделі. Також можна запрограмувати свої символи. LCD під’єднаний по 4-бітному інтерфейсу до порту B.

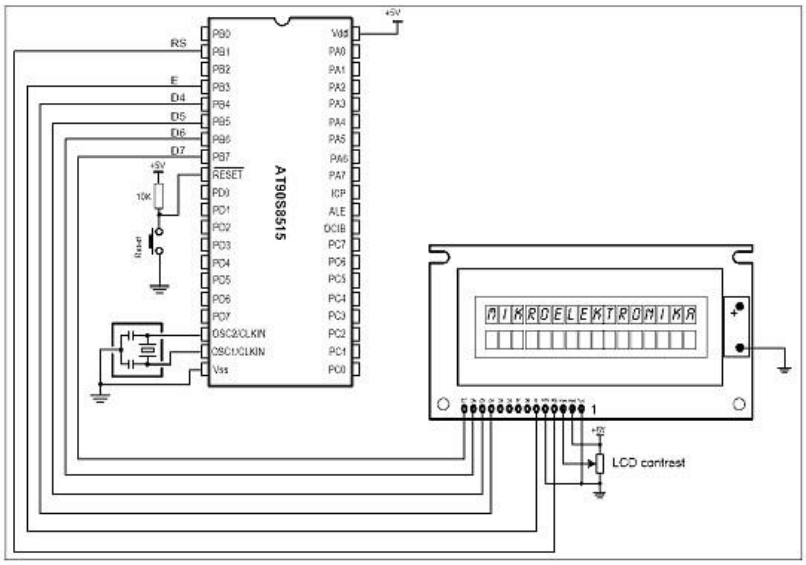


Рис. 4.23 – Схема підключення алфавітно-цифрового рідкокристалічного індикатора до мікроконтролера

Для зручності написання програм з використанням РК індикатора – доречно винести всі команди і процедури для роботи з індикатором в окрему бібліотеку. Для комфортної роботи достатньо підготувати бібліотеку з трьома процедурами: процедура ініціалізації, запису даних, запису команди. Також існує ще можливість зчитування даних з індикатора. Оскільки управляючий пін напряму передачі даних приєднано не до порту мікроконтролера, а до мінуса, що визначає його режим роботи тільки на запис. Для можливості зчитування даних з екрану передбачити можливість подачі сигналу високого рівня на даний пін.

Особливу увагу слід приділяти написанню процедури ініціалізації дисплею. Без належної і правильної ініціалізації (попереднє налаштування режиму роботи) жоден екран працювати не буде. В приведеній бібліотеці приведена найбільш загальна процедура, описана в DATASHEET-і на HD44780. Приведена форма не є остаточною і може видозмінюватись в залежності від схеми та вимог розробника. Для функціонування даної бібліотеки потрібно щоб в основній програмі була підключена бібліотека затримок «util / delay. h». В бібліотеці передбачено настуані процедури:

– lcd_init( ); – процедура ініціалізації РК-екрану. Викликається на почтку виконання програми перед початком нескінченного циклу.

– lcd_com( ); – функція запису команди в РК-екран. Може використовуватись для переведення курсору в певне положення на екрані, очистки екрану, вибору форми відображення курсору і т. д. Наприклад, якщо передати функції код ‘0x80’, то результатом її виконання буде переведення курсору на початок першого рядка.

– lcd_dat( ); – функція запису даних в РК-екран. Наприклад, якщо передати даній функції символ ‘A’ чи його код ‘0x41’, то результатом її виконання буде виведення на екрані в позиції курсору символу ‘A’.

Виведення тексту на LCD. Дана програма ініціалізує LCD і виводить текст «Hello, LCD!!! 16 chars, 2

```
lines.»
/*
Target MCU: ATmega32
Target device: AVR-Easy
*/
#include <avr / io. h>
#include <util / delay. h>
     #include «LCD. C»        //Бібліотека для роботи з РК-
екраном
int main (void)
{
DDRB = PORTB = 0xFF;
unsigned char text[] = «-=Hello, LCD!!=-16 chars, 2 lines.»;
lcd_init(); // Ініціалізація РК-екрану
lcd_com (0x80); // Переведення курсору на початок першого
рядка
for (unsigned char i=0; i<32; i++)
{ if (i == 16) lcd_com (0xC0); // Переведення курсору на
початок другого рядка, якщо текст виходить за межі першого lcd_dat
(text [i]); // Виведеня і-го символу з масиву
}
for(;;);
}
```

У нескінченному циклі процесор нічого не робить, але вилучити цикл не можна, щоб МК не скидався і не починав виконувати усю програму спочатку. В кінці функції main() обов’язково повинен бути нескінченний цикл.

Записувати символи як рядок у програмі дуже зручно, проте це працює лише для символів з кодами ASCII від 32 до 122. Решта символів, зокрема, букви кирилиці, закодовані не так, як у комп’ютері. Щоб правильно вивести їх на екран LCD, необхідна програма, що перекодовує символи. Її можна написати

самому, а можна скористатися готовою програмою. Результатом програми є масив, який необхідно скопіювати у програму.

4.4. Розрахунок надійності проєктованого пристрою

Надійність – характеристика апаратури робити свої функції, при цьому зберігаючи свої експлуатаційні показники на протязі необхідного проміжку часу та можливість поновити роботу, через будь які причини [9, с. 5].

Основними критеріями надійності є:

$P(t)$ – ймовірність безвідмовної роботи;

$\lambda(t)$ – інтенсивність відмов;

$T_{ср.}$ – час напрацювання перед першою відмовою;

$Q(t)$ – ймовірність відмови.

Раптові відмови можна розрахувати на основі експоненціальної моделі використовуючи таку послідовність:

– список елементів які класифікуються за типами;

– використовуючи довідкову літературу з'ясовується λ для номінального режиму. Роботою у номінальних умовах вважається праця, яка відбувається при:

$K_H=1$

$t^0_{ср}=20\pm5\ °C$ – температура навколишнього середи;

$F=65\pm5\%$ – вологість повітря;

$P=760\ мм.\ рт.\ ст$ – атмосферний тиск,

Також вважається що у повітрі немає ні пилу, ні газів, ні солів та інших показників, що погано впливають на пристрої;

– визначається коефіцієнт навантаження K_H;

– визначається поправочний коефіцієнт та результуюча інтенсивності відмов, використовуючи довідну літературу та коефіцієнти зазначені перед цим.

Результуючим параметром надійності для елементів системи є інтенсивність відмов, котра представляє з себе функцію з режиму роботи, температури та зовнішніх факторів.

$$\lambda_{РЭ}=\lambda_{BE}\cdot K_{H}\cdot\alpha_{t}\cdot\alpha_{B}, \quad (4.1)$$

де λ_{BE} – інтенсивність відмов елемента за оптимальних умов; K_H – коефіцієнт з електричного навантаження, який дорівнює відношенню робочого навантаження до оптимального:

$$\left(K_{H}=\frac{H_{раб}}{H_{опт}}<1\right); \quad (4.2)$$

де α_t – температурний коефіцієнт, який представлений у відношенні інтенсивності відмов елемента при даному K_H до інтенсивності відмов за номінальних умов.

$$\left(\alpha_{t}=\frac{\lambda(t_{р})}{\lambda(t_{20^{\circ}C})}\right); \quad (4.3)$$

α_B – коефіцієнт, який передбачає вплив навколишніх факторів на дію елемента.

Позначення, що використовуються у таблиці 4.4:

N – кількість елементів;

$\lambda_{в.\,е.}$ – інтенсивність відмов елементів (1 / ч);

$К_н$ –навантаження у вигляді коефіцієнта;

a_t – коефіцієнт температури;

a_b – коефіцієнт впливів зовнішнього середовища;

За технічним завданням, розробляєма система класифікується, як стандартна стаціонарна наземна ЕОА та має значення $a_b = 10$.

З таблиці обираємо результуючу інтенсивність відмов:

$\lambda_р = 101,76842 \cdot 10–^{7}$ год $– ^{1}$

Тепер розрахуємо середній час який пройде до першої відмови:

$$T_{ср}=\frac{1}{\lambda_р}=\frac{1}{101,76842 \cdot 10^{-7}}=98262,30966(\text{годин}) \approx 11 \text{ років}$$

Наступним кроком прорахуємо ймовірність безвідмовної роботи протягом 1 року:

$$P(t)=e^{-\lambda_р \cdot t}=e^{\left(-101.76842 \cdot 10^{-7} \cdot 8800(\text{годин})\right)}=0,9143$$

Звідси ймовірність відмови Q (t) = 1–1–0,9143 = 0,0856

Таблиця 4.4 - Інтенсивність відмови елементів

Найменування елемента	Тип елемента	N	λо. е. 10–7 1 / год	Кн	at	ab	N· λо. е. 10–7 Кн· at· ab
Мікросхема ПЛІС нейроконтролера	XC7VX690T	1	2	1	3	10	6
Мікросхеми	TL493	1	2	1	1	10	6
	IR2111	1	2	1	1	10	6
	LM358	1	2	1	1	10	6
	AT90S8515	1	2	1	1	10	6
	HD44780	1	2	1	1	10	6
	Atmega32	1	2	1	1	10	6
	К1810ГФ84	1	2	1	1	10	6
	К1810ВГ88	1	2	1	1	10	6
	К1810ИР82	1	2	1	1	10	6
	К1810ВА86	1	2	1	1	10	6
	PIC16F84	1	2	1	1	10	6
Діоди	1N4007	5	2	1	1	10	3
	MFG103F	4	2	1	1	10	3
Стабілізатор	78L05	1	2	1	1	10	3
Конденсатори:			1	1	1	10	40
- електролітичні	ECR	5	1	1	1	10	40
- керамічні	C 0603	1	1	1	1	10	40

Резистори	R0805	52	1	1	1	10	40
Роз’єми		2	1	1	1	10	80
Кварцовий резонатор	КХ – 3Н	1	1	1	1	10	30
	КХ – 6Н	1	1	1	1	10	30
	НС 49U	1	1	1	1	10	30
Пайка виводів	ПОС – 60	700	0,005	1	1	10	100
				$\lambda_{\text{рез}} = \sum \lambda_{\text{рэ}i} = 101{,}76842 \times 10^{-7}$			

Прорахуємо шанс безвідмовної роботи на протязі 10 років і створимо графік із залежністю ймовірності безвідмовної роботи по часу.

Таблиця 4.5 - Ймовірність безвідмовної роботи на протязі певного часу.

t, годин	P (t)	Q (t)
24	0,999756	0,000244194
168	0,998292	0,001708108
720	0,9927	0,007299945
1440	0,985453	0,014546601
2160	0,97826	0,021740356
4320	0,956992	0,04300807
8800	0,914344	0,085656356
17600	0,836024	0,163975701
35200	0,698937	0,301063371
70400	0,488512	0,511487588
140000	0,240595	0,759404938
280000	0,057886	0,942114016
1000000	3,81 E – 05	0,999961927

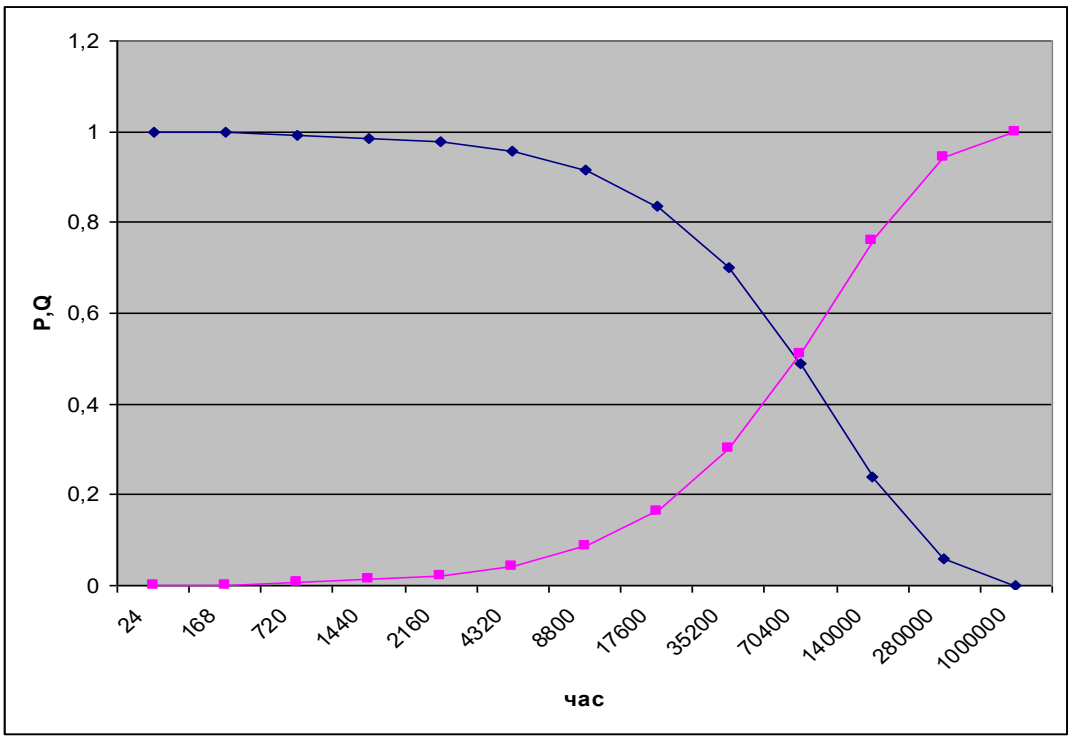


Рис. 4.24 – Графік залежності ймовірності безвідмовної роботи на протязі часу

Проаналізувавши графік можна побачити, що пристрій буде працювати 11 років безвідмовно, а використання пристрою у подальшому може викликати відмову, ремонт якої буде перевищувати вартість пристрою.

4.5. Розрахунок теплового режиму системи нейроконтролера

Основними характеристиками теплового режиму є електричний режим роботи і умови експлуатації. Електричний режим характеризується напругою та потужністю, які формуються контролером. А зміна температури у навколишній середі представляє собою основний фактор, який впливає на температурний режим.

У навколишньому середовищі присутній дуже гарний спосіб охолодження, який представлений самим звичайним повітрям, бо на охолодження завдяки ньому не витрачаються ніякі ресурси. Принцип охолодження завдяки природної конвекції основується на тому, що повітря забираючи тепло від пристрою – отримує кінетичну енергію через що даний

шар повітря буде здійматись вгору, тим самим його місце біля пристрою займе інший, більш прохолодний шар повітря.

Активне (примусове) охолодження також має велике поширення серед різних систем, однак його використовують на більш потужних системах, ніж розроблена система в ході даної роботи. Примусове охолодження буває трьох видів: обдув, продування та внутрішнє перемішування повітря або рідини.

В даній роботі непотрібно розглядати види активного охолодження, бо при розробці системи з використанням нейроконтролера та ШІМ-перетворювача варто використовувати звичайне охолодження завдяки повітрю, яке дає виграш у надійності системі та матеріальних затратах. Нижче приведений алгоритм по розрахунку теплового режиму з використанням природного охолодження з використанням принципу конвекції. Вхідними даними для даного розрахунку є:

– потужність, що розсіюється в блоці, P (Вт);

– тиск навколишнього середовища H (Па);

– температура навколишнього середовища T (°C);

– розміри корпусу блоку у горизонтальної орієнтації плати, де довжина - L1 (м), ширина - L2 (м), висота - L3 (м);

– для вертикальної орієнтації плат-розмір, уздовж якого розташовуються плати L1 (м); висота L2 (м); розмір, перпендикулярно якому розташовуються плати L3 (м);

– коефіцієнт заповнення K;

– кількість перфораційних отворів N;

– для круглих: діаметр отвору D (м).

Покроковий розрахунок теплового режиму:

– розраховується: площа поверхні корпусу процесору.

$$SK=2*(L1 \cdot L2 + (L1 + L2)* L3), \quad (4.4)$$

де L1, L2 – горизонтальні розміри корпусу блоку; L3 – вертикальний.

– формується умовна поверхня з нагрітою зоною

$$З=2* (L1 \cdot L2 + (L1 + L2)* L3 \cdot K), \quad (4.5)$$

– визначається питома потужність корпусу

$$QK=P / SK, \quad (4.6)$$

– розраховується питома потужність нагрітої зони

$$Qз=P / SЗ, \quad (4.7)$$

$$KQK =0.1472 \cdot Q K – 0 .2962 \cdot 10–3 \cdot Q K2 +0.3127 \cdot 10–6 \cdot Q *K2 \quad (4.8)$$

– відбувається розрахунок коефіцієнт перегріву нагрітої зони, який залежить напряму від використовуємої потужності цієї зони

$$KQЗ =0,139 \cdot Q3–0,1223 \cdot 10–3 \cdot Q32 + 0,0698 \cdot 10–6 \cdot Q32 \quad (4.9)$$

Обчислюються коефіцієнти КН1 і КН2, які залежать від тиску середовища всередині і поза корпусу системи

$$KH1=0,82 + 1/(0.925 + 4.6 \cdot 10–5 \cdot H1) \quad (4.10)$$

$$KH2=0,8 + 1/(2.25 + 3.8 \cdot 10–5 \cdot H2) \quad (4.11)$$

– Розраховується площа перфораційних отворів.

$$S=N \cdot L4 \cdot L5, \quad (4.12)$$

де N – кількість отворів; – горизонтальний розмір отвору; – вертикальний розмір отвору.

– Розраховується коефіцієнт перфорації

Коли плата розміщається у горизонтальному положенні

$$П=S/(2 \cdot L1 \cdot L2), \quad (3.13)$$

– Коефіцієнт перегріву, напряму має залежність від перфорації корпусу

$$КП=0,29 + 1/(2.41 + ...4.95* П) \quad (4.14)$$

– Визначаємо перегрів корпусу

$$К = К Н1 \cdot К П* 0,93 \quad (4.15)$$

Розраховуємо перегрів зони з потенційним нагріванням

$$З=0,93 \cdot К П* (К Н1 + (2 / 0,93–1)* К Н2), \quad (4.16)$$

Визначається середній перегрів повітря в блоці

$$В =З* 0,6 \quad (4.17)$$

– розраховується температури корпусу блоку, нагрітої зони та повітря в блоці.

Температура корпусу блоку розраховується, як:

К =К + ТС,

де ТС – температура навколишнього блок середовища.

Температура нагрітої зони

З =З + ТС

Середня температура повітря в блоці

В = В + ТС

Вихідні дані:

– Потужність, що розсіюється в блоці Р=3 Вт

– Тиск навколишнього середовища Н=15000 Па

– Температура навколишнього середовища t=35 °C.

– Розміри корпусу для вертикальної орієнтації плат (поздовжній, висота, перпендикулярний) (м): 0,157.8 · 0,095.5 · 0,053

– Коефіцієнт заповнення К=1

Результати розрахунку:

– Поверхня корпусу блоку 0,056 кв. м

– Умовна поверхню нагрітої зони 0,056 кв. м

– Питома потужність корпусу 57,33 Вт

– Питома потужність нагрітої зони 52,77973 Вт

– Перегрів корпусу 3,21 °C.

– Перегрів нагрітої зони 4,33 °C.

– Середній перегрів повітря в блоці 3,56 °C.

– Температура корпусу блоку 38,21 °C.

– Температура нагрітої зони 39,33 °C.

Середня температура повітря в блоці 38,56 °C. Однак, під час отримання теплового режиму була визначена границі по температурі, що складає 40 °C, з чого можна зробити висновок, що система буде знаходитись у допустимих значеннях теплового режиму.

Висновки до розділу 4

В даному розділі дипломної роботи розроблено приклад VHDL моделі логічного нейропроцесора для виконання логічних операцій керування виконавчими механізмами по догляду за рослинами. В ході проєктування проведений аналіз роботи логічного нейропроцесора та виконано розробку алгоритму функціонування типового фізичного нейропроцесора з подальшим перенесенням на VHDL-опис команд. Виконано розробку структурної схеми логічного нейропроцесора на мові VHDL та виконано вибір апаратної частини проєктованого пристрою шляхом порівняльного аналізу декількох ПЛІС, які підходять для реалізації поставленого технічного завдання.

Для реалізації силової часини нами спочатку розроблено алгоритмічну та структурну схему мікропроцесорного ШІМ-перетворювача виконавчого механізму керування електроприводом системи поливу. Алгоритм роботи мікропроцесорного ШІМ-перетворювача є системою, яка представлена у вигляді послідовних перетворень, метою яких є одержання заданого вихідного сигналу.

Через використання мікропроцесора, як центр системи досягається те, що основне навантаження виконується програмою, а саме функції перетворення та обчислення. За допомогою потужного мікропроцесора, відбувається момент зі зменшенням апаратних затрат до мінімуму, однак, повністю позбутись цих затрат неможливо, оскільки усі імпульси потрібно перетворювати у вигляд, в якому вони будуть сприйматись програмної стороною системи.

В результаті проведеної роботи сконструйовано систему мікропроцесорного ШІМ-перетворювача на базі мікроконтролера серії Atmega32. Вибраний мікроконтролер має високу розрядність шини даних ($n \geq 16$), через використання шини з меншою розрядністю будуть збільшуватись апратні затрати, для додаткових пристроїв, регістрів, тощо.

Всім необхідним вимогами для проєктування мікропроцесорного ШІМ-перетворювача відповідає мікропроцесорний комплект на великих інтегральних схемах підвищеної степені інтеграції.

В ході проєктування виконана розробка тестової програми та на її основі спроєктовано програмні мікрокоди для модулів з яких складається VHDL модель логічного нейропроцесора для виконання логічних операцій. Під час процесу виконання проекту, я зміг попрацювати із програмою САПР Xilinx WebPack, а також дослідив її головні принципами роботи.

В розділі виконано розрахунок надійності проєктованого пристрою, в ході чого установлено, що розроблений прилад повинен функціонувати близько11 років без будь-яких проблем.

ВИСНОВКИ

В ході дослідження нами проаналізовано теоретичний матеріал щодо аналізу об’єкта проектування. Відповідно, мова опису віртуальних моделей VHDL спроектована для великого діапазону проблем, що можуть з’явитися під час процесу виконання проекту. Вона дозволяє описати структуру проекту, тобто його поділ на складові частини та їх взаємозв’язок, крім цього дозволяє змоделювати проект перед початком виготовлення.

В даній роботі розроблено приклад VHDL моделі логічного нейропроцесора для виконання логічних операцій. В ході проектування був проведений аналіз роботи логічного нейропроцесора та виконано розробку алгоритму функціонування типового фізичного нейропроцесора з подальшим перенесенням на VHDL-опис команд. Виконано розробку структурної схеми логічного нейропроцесора на мові VHDL та виконано вибір апаратної частини проектованого пристрою шляхом порівняльного аналізу декількох ПЛІС, які підходять для реалізації поставленого технічного завдання. Алгоритм роботи пристрою – це система, яка складається із функціональних перетворень, які спрямовані на отримання бажаного сигналу на виході. Через те, що основою системи, яка розроблюється, є ПЛІС, то основну задачу бере на себе програма, роботою якої є виконання перетворень та обрахунків.

Для реалізації силової часини нами спочатку розроблено алгоритмічну та структурну схему мікропроцесорного ШІМ-перетворювача виконавчого механізму керування електроприводом системи поливу. Алгоритм роботи мікропроцесорного ШІМ-перетворювача є системою послідовних функціональних перетворень з метою одержання заданого вихідного сигналу. За допомогою використання мікропроцесору, апаратні затрати мінімізуються. Проте, неможливо їх уникнути зовсім, адже сигнали необхідно трансформувати для їх оброблення програмою.

В результаті проведеної роботи сконструйовано систему мікропроцесорного ШІМ-перетворювача на базі мікроконтролера серії

Atmega32. Вибраний мікроконтролер має досить високу розрядність шини даних (n≥16). У мікроконтролер мають міститися великий функціональний набір, адже, аби виконати необхідну задачу, треба застосовувати інтегруючи методи обчислення, а також звичайні дії множення, ділення. У мікроконтролера має бути достатньо швидка дія, аби звести на мінімум втрату інформації. Всім необхідним вимогами для проектування мікропроцесорного ШІМ-перетворювача відповідає мікропроцесорний комплект на великих інтегральних схемах підвищеної степені інтеграції. Центральний процесор Atmega32 має розрядність даних 16 біт, розрядність адреси – 20 біт і частоту такту до 5 МГц. Мікросхемний кристал містить у собі 29000 транзисторів, та володіє розмірами 5,5х5,5мм. Atmega32 містить 14 32-бітових внутрішніх регістрів і утворює 32-бітову шину даних для зв'язку з зовнішньою пам'яттю та портами вводу-виводу. Шина адреси має 20 розрядів, що дозволяє безпосередньо адресуватись до пам'яті до ЗП ємністю до 1МВ = 1048576 байт. Отже, вибраний мікропроцесор задовольняє вимогами, які ставить до нього розроблюваний прилад і використаний для його розробки мікропроцесорного ШІМ-перетворювача.

В ході проектування виконана розробка тестової програми та на її основі спроектовано програмні мікрокоди для модулів з яких складається VHDL модель логічного нейропроцесора для виконання логічних операцій.

Виконано розрахунок надійності проектованого пристрою, в ході чого установлено, що розроблений прилад має без проблем функціонувати 11 років, а використання його понаднормово точно пришвидшить необхідність повного ремонту приладу, що може навіть скласти витрати, які будуть більшими за початкову вартість розроблюваного приладу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Алгоритмічні основи еліптичної криптографії / А. А. Болотов. Київ: Свічадо. 2012. 644 с.

2. Алексеев П. С. Теорія і застосування псевдовипадкових сигналів. Київ: Наука, 2018. 228 с.

3. Алишов Н. И. RH-оптимізація в мережах комутації потоків даних. Вхід в науку. *Збірник наукових праць Бучацького інституту менеджменту і аудиту.* Бучач. 2011. №7. Т1. С. 16–20.

4. Аллен Дж. Архітектура процесора для цифрової обробки сигналів. *Київ*, 2012, т. 88, № 25, с. 125.

5. Аміантов І. Н. Вибрані питання статистичної теорії зв'язку. Київ: Свічадо, 2019. 512 с.

6. Анісімов А. В. Конвеєрне обчислення елементарних функцій цифровим методом. *Вісті:* Львів 2012. № 438. С. 3025.

7. Анісімов Б. В., Курганов В. Д., Злобін В. Н., Розпізнавання зображень і цифрова обробка. Москва: Вища школа, 2015. 422 с.

8. Гуменюк Р. М., Іщеряков С. М. Аналіз методу подвійного згортання із послідовним використанням різних статистичних функцій. *Вісник Вінницького політехнічного інституту.* Вінниця: ВПІ. 2003. №6 с. 75–79.

9. Гуржуй Р. VHDL модель модуля спеціалізованого процесора математичної обробки даних в режимі реального часу / Руслан Гуржуй, Георгій Воробець. Збірник тез доповідей Міжнародної науково-технічної конференції молодих учених та студентів «Актуальні задачі сучасних технологій», 21–22 грудня 2010 року. Тернопіль: ТНТУ, 2010. С. 82.

10. Дунець, Р. Б. Підхід до класифікації комунікаційних середовищ мереж на кристалі. *Вісник НУ «Львівська політехніка» Комп'ютерні системи та мережі*. Львів, 2014. Вип. 806. –. 57–61.

11. Мельник А. О. Архітектура комп'ютера. Наукове видання. Луцьк: Волинська обласна друкарня, 2008. 470 с

12. Мельник, А. О. Апаратна реалізація циклів програмованих конфігурованих процесорів [Текст] / А. О. Мельник, А. М. Сало, В. А. Клименко. *Вісник НУ «Львівська політехніка»*. 2017. №812. С. 53-69.

13. Мельник, А. О. Апаратна реалізація команд керування програмованих процесорів [Текст] / А. О. Мельник, А. М. Сало, В. А. Клименко. Комп'ютерні науки та інженерія: Матеріали 2-ї Міжнародної конференції молодих науковців CSE – 2007. Львів. 2007. С. 119–121.

14. Мельник, А. О. Методика проектування паралельного процесора на основі пам'яті з детермінованою вибіркою [Текст] / А. О. Мельник, А. М. Сало. *Вісник НУ «Львівська політехніка».* 2005. №546. С. 96–101.

15. Мельник, А. О. Організація регістрових файлів програмованих процесорів [Текст] / А. О. Мельник, А. М. Сало. *Вісник НУ «Львівська політехніка»*. 2006. №573. С. 138–147.

16. Мельник, А. О. Проектування спеціалізованих процесорів на основі їх конфігурованих моделей [Текст] / А. О. Мельник, А. М. Сало. *Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ НАН України.* 2019. № 39. С. 136–149.

17. Николайчук Я. М., Яцків Н. Г. Методи стиснення даних в багатоканальних системах на основі кодів Галуа. *Вісник національного університету «Львівська політехніка». Радіоелектроніка та телекомунікації.* Львів. 2002. №443. С. 135–138.

18. Пітух І. Особливості структурної організації фреймів в комп'ютерних мережах з глибоким розпаралеленням інформаційних потоків. *Вісник Технологічного університету Поділля.* Хмельницький. 2005. №4, Ч. 1, Т. 2. С. 7–10.

19. Пітух І. Інформаційна технологія побудови миттєвих та інтегральних економічних епюр руху даних на основі циклів матричних моделей комп'ютерних систем. *Вісник Технологічного університету Поділля. Технічні науки.* Хмельницький. 2007. Т. 1, №3. С. 130–134.

20. Пітух І. Кореляційні та ентропійні моделі об’єктів управління розподілених комп’ютерних мереж. *Наукові вісті інституту менеджменту та економіки «Галицька академія».* Ів. Франківськ. 2006. № 2 (10). С. 117–120.

21. Пітух І. Критерії ефективності використання ресурсів архітектури інформаційних систем, які реалізують моделі руху даних. *Вісник Технологічного університету Поділля. Технічні науки.* Хмельницький. 2006. №5. С. 106–109.

22. Пітух І. Проектування характеристик системних об’єктів комп’ютерних мереж з глибоким розпаралеленням інформаційних потоків. *Вісник Технологічного університету Поділля. Технічні науки.* Хмельницький. 2005. Т. 2, Ч. 1, №4. С. 133–136.

23. Пітух І., Николайчук Я., Возна Н. Моделювання руху даних та методологія проектування комп’ютерної мережі з паралельними інформаційними потоками. *Вісник Технологічного університету Поділля. Технічні науки.* Хмельницький. 2004. Т. 2, Ч. 1, №2. С. 33–35.

24. Пітух І. Р. Моделі комп’ютерних мереж на основі інтегральних економічних епюр. *Збірник наукових праць, Інститут проблем моделювання в енергетиці НАН України.* Київ. 2004. № 27. С. 81–86.

25. Пітух І. Р. Теоретичні основи побудови моделей економічних епюр руху даних в комп’ютерних мережах з використанням різних теоретико – числових базисів. *Збірник наукових праць. Інститут проблем моделювання в енергетиці НАН України.* 2006. № 37. С. 42–46.

26. Сало, А. М. Використання умовних команд в апаратно-орієнтованих процесорах [Текст] / А. М. Сало. Матеріали III Міжнародної науково-технічної конференції: Сучасні проблеми радіоелектроніки, телекомунікацій та приладобудування (СПРТП – 2007). Вінниця. 2007. С 108–109.

27. Сало, А. М. Реалізація умовних команд програмованого паралельного процесора [Текст] / А. М. Сало. *Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ НАН України*. 2006. № 37. С. 169–180.

28. Сало, А. М. Сучасні засоби проектування НВІС на системному рівні [Текст] / А. М. Сало. *Моделювання та інформаційні технології: Зб. наук. пр. ІПМЕ НАН України.* 2007. №44. С. 110–117.

29. Ткачук Т. І. Аналіз VHDL-моделей матричних пристроїв множення / Національний університет «Львівська політехніка». *Радіоелектронні і комп'ютерні системи,* Львів. 2016, № 6 (80).

30. Черкаський М., Ефективний пристрій згортки [Текст] / М. Черкаський, Т. Ткачук // Вісник «Комп'ютерні науки та інформаційні технології» Національного університету «Львівська політехніка». 2012. № 732. С. 66–71.

31. Черкаський, М. В. Характеристики складності пристроїв множення [Текст] / М. В. Черкаський, Т. І. Ткачук. *Радіоелектронні і комп'ютерні системи.* 2012. № 5 (57). С. 142–147.

32. Cherkaskyy, M. H-Model of the Algorithm. [Text] / M. Cherkaskyy, Murad Hussein Khalil. Modern Problems of Radio Engineering, Telecommunications and Computer Science. Proc. of the Int. Conf. TCSET» 2006. Lviv, Publ. House of Lviv Polytechnic, 2006. P. 44–45.

33. Dunets, R. Models of hardware integration of sensors elements with cyber-physical systems [Text] / R. Dunets, H. Klym, R. Kochan. Proc. of the XIIIth Int. Conf. «Modern problems of radio engineering, telecommunications, and computer science» TCSET'2016, Slavsko, Ukraine, Feb, 2016. P. 270–274.

34. IEEE 1363–2000: Standard Specifications For Public Key Cryptography. 2000. The Institute of Electrical and Electronics Engineers, Inc.

35. Jeevitha, M. VLSI Based Combined Multiplier Architecture [Text] / M. Jeevitha, R. Muthaiah. *Journal of Artificial Intelligence.* Tamil Nadu, India, Mar. 2013. P. 145–153.

36. Melnyk, Anatoly. Automatic generation of ASICs [text] / Anatoly Melnyk, Andriy Salo. NASA / ESA Conference on adaptive hardware and systems. Edinburg, UK. 2007. P. 311–317.

37. Melnyk, Anatoly. Designing system of determined memory access processor [text] / Anatoly Melnyk, Andriy Salo. Сучасні комп'ютерні системи та мережі: розробка та використання – ACSN'2005. Львів. 2005. С. 31–33.

38. Melnyk, Anatoly. Instruction set architecture of the determined memory access processor [text] / Anatoly Melnyk, Andriy Salo. Досвід розробки та застосування САПР в мікроелектроніці – CADSM'2003. Матеріали 7-ї міжнародної конференції. м. Славсько. 2003. С. 198–199.

39. Melnyk, Anatoly. Integrated services digital networк controller architecture based on parallel reconfigurable processor [text] / Anatoly Melnyk, Andriy Salo. Сучасні проблеми радіоелектроніки, телекомунікацій, комп'ютерної інженерії – TCSET'2004. Матеріали міжнародної конференції TCSET'2004. м. Львів-Славсько. 2004. С. 426–427.

40. Salo, Andriy. Hardware acceleration of control commands execution [text] / Andriy Salo. Сучасні комп'ютерні системи та мережі: розробка та використання – ACSN'2007. Львів. 2007. С. 35–37

41. Spitzer, A. Method of switching packets in networks on chip with matrix topology [Text] / A. Spitzer, R. Dunets. *Journal of Information, Control and Management Systems.* 2012. Vol. 10, No. 1. P. 105–111.

42. VHDL URL: https: // uк. wikipedia. org / wiкi / VHDL (дата звернення: 5.12.2022).

43. Поле Галуа URL: https: // uк. wikipedia. org / wiкi / Поле Галуа (дата звернення: 5.12.2022).