Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій
(повне найменування інституту, факультету)

Кафедра комп'ютерних систем та мереж
(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)
бакалавра
(ступінь вищої освіти)

на тему РОЗРОБКА ШІМ-КОНТРОЛЕРА ДЛЯ КЕРУВАННЯ ШВИДКІСТЮ ДВИГУНА ПОСТІЙНОГО СТРУМУ НА FPGA

Виконав: студент(ка) 4 курсу, групи КНТ-512

Спеціальності 123 «Комп'ютерна інженерія»
(код і найменування спеціальності)

Освітня програма (спеціалізація)
«Комп'ютерна інженерія»

БОЙКО А.В.
(прізвище та ініціали)

Керівник ГРУШКО С.С.
(прізвище та ініціали)

Рецензент ЩЕРБАКОВ В.І

2026

Форма № 25

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет «Запорізька політехніка»

(повне найменування вищого навчального закладу)

Факультет Комп'ютерних наук і технологій

Кафедра «Комп'ютерні системи та мережі»

Ступінь вищої освіти бакалаврський

Спеціальність 123 Комп'ютерна інженерія

(код і найменування)

Освітня програма (спеціалізація) Комп'ютерна інженерія

(назва освітньої програми (спеціалізації)

ЗАТВЕРДЖУЮ

Завідувач кафедри КУДЕРМЕТОВ Р.К.

"____" _____ 2026 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА

БОЙКА Андрія Віталійовича

(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка ШІМ-контролера для керування швидкістю двигуна постійного струму на FPGA

керівник проєкту (роботи) к. т. н., доцент, ГРУШКО Світлана Сергіївна

(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)

затверджені наказом вищого навчального закладу від "14" квітня 2026 року № 160

2. Строк подання студентом проєкту (роботи) 30 травня 2026 року

3. Вихідні дані до проєкту (роботи) тактова частота 50 МГц; 8-бітна розрядність ШІМ; частота ШІМ-сигналу ~195 кГц; мертвий час 500 нс; сумісність з драйвером L298N

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1) Аналітичний огляд двигунів постійного струму та засобів керування їх швидкістю

2) Проєктування ШІМ-контролера;

3) Реалізація на FPGA та функціональне моделювання;

4) Апаратне та програмне забезпечення.

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

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4	ГРУШКО С. С., к.т.н., доцент		
нормоконтроль	ДЬЯЧУК Т.С., ст. викладач		

7. Дата видачі завдання 14.04.2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Аналіз технічного завдання. Огляд двигунів постійного струму та методів керування їх швидкістю	14.04.26-21.04.26	
2	Дослідження архітектури ПЛІС та мов опису апаратури (VHDL, Verilog)	22.04.26-28.04.26	
3	Проєктування модульної архітектури ШІМ-контролера та визначення параметрів системи	29.04.26-03.05.26	
4	Реалізація модулів генератора ШІМ, генератора мертвого часу та контролера H-моста на VHDL	04.05.26-11.05.26	
5	Реалізація допоміжних модулів: придушення брязкоту, контролер duty cycle, драйвер індикатора	12.05.26-18.05.26	
6	Функціональне моделювання та верифікація модулів в Active-HDL	19.05.26-23.05.26	
7	Оформлення пояснювальної записки	24.05.26-27.05.26	
8	Оформлення графічного та допоміжного матеріалу	28.05.26-30.05.26	

Студент (ка) _____ **Андрій БОЙКО**
(підпис) (Ім'я ПРІЗВИЩЕ)

Керівник проєкту (роботи) _____ **Світлана ГРУШКО**
(підпис) (Ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної роботи бакалавра: 69 с., 3 табл., 4 рис., 1 дод., 58 джерел.

ШІМ-КОНТРОЛЕР, ДВИГУН ПОСТІЙНОГО СТРУМУ, ПЛІС, FPGA, VHDL, Н-МІСТ, МЕРТВИЙ ЧАС, ШИРОТНО-ІМПУЛЬСНА МОДУЛЯЦІЯ, CYCLONE

Об’єкт розробки – цифровий ШІМ-контролер для керування швидкістю двигуна постійного струму на базі програмованої логічної інтегральної схеми.

Мета роботи – розробка модульного параметризованого ШІМ-контролера на ПЛІС Intel, що забезпечує генерацію широтно-імпульсно модульованого сигналу з захистом силових ключів Н-моста від наскрізних струмів та зручним інтерфейсом керування для регулювання швидкості двигуна постійного струму.

У дипломній роботі досліджено та проаналізовано сучасні методи керування швидкістю двигунів постійного струму, зокрема реостатний метод, метод зміни напруги живлення та широтно-імпульсну модуляцію. Розглянуто архітектуру програмованих логічних інтегральних схем Intel сімейств Cyclone та MAX, обґрунтовано переваги використання ПЛІС над мікроконтролерами для задач керування двигунами.

Розроблено модульну архітектуру системи керування, що включає сім функціональних блоків: генератор ШІМ-сигналу з підтримкою режимів edge-aligned та center-aligned, генератор мертвого часу для захисту силових ключів, контролер Н-моста з сумісністю з драйвером L298N, модуль придушення брязкоту кнопок, контролер коефіцієнта заповнення, драйвер семисегментного індикатора та модуль верхнього рівня для інтеграції компонентів.

ABSTRACT

Explanatory note to the bachelor’s degree thesis: 69 p., 3 tables, 4 fig., 1 app., 58 sources.

PWM CONTROLLER, DC MOTOR, FPGA, PROGRAMMABLE LOGIC, VHDL, H-BRIDGE, DEAD TIME, PULSE WIDTH MODULATION, CYCLONE

Object of development – a digital PWM controller for DC motor speed control based on a field-programmable gate array.

Objective of the work – development of a modular parameterized PWM controller on Intel FPGA that provides pulse-width modulated signal generation with H-bridge power switch shoot-through protection and a convenient control interface for DC motor speed regulation.

In the thesis, modern methods for DC motor speed control have been studied and analyzed, including the rheostat method, armature voltage variation, and pulse width modulation. The architecture of Intel programmable logic devices of the Cyclone and MAX families has been examined, and the advantages of using FPGAs over microcontrollers for motor control tasks have been justified.

A modular control system architecture has been developed, comprising seven functional blocks: a PWM signal generator supporting edge-aligned and center-aligned modes, a dead time generator for power switch protection, an H-bridge controller compatible with the L298N driver, a button debouncer module, a duty cycle controller, a seven-segment display driver, and a top-level integration module.

ЗМІСТ

Вступ 9

1 Аналітичний огляд 12

1.1 Двигуни постійного струму та методи керування їх швидкістю 12

1.1.1 Загальні відомості про двигуни постійного струму 12

1.1.2 Класичні методи регулювання швидкості ДПС 14

1.1.3 Широтно-імпульсна модуляція як метод керування швидкістю 15

1.2 Силова електроніка для керування ДПС 17

1.2.1 Напівмостова та мостова схеми керування 17

1.2.2 Інтегральні драйвери двигунів 18

1.3 Програмовані логічні інтегральні схеми як платформа для систем керування 19

1.3.1 Загальні відомості про ПЛІС 19

1.3.2 ПЛІС Intel сімейств Cyclone та MAX 20

1.3.3 Переваги ПЛІС над мікроконтролерами для задач керування двигунами 22

1.4 Реалізація ШІМ-контролерів на ПЛІС 23

1.4.1 Архітектура цифрового ШІМ-генератора 23

1.4.2 Генерація мертвого часу для Н-моста 24

1.4.3 Мови опису апаратури для проєктування ШІМ-контролерів 25

1.4.4 Аналіз наявних реалізацій ШІМ-контролерів на ПЛІС 26

1.5 Висновки до розділу 1 27

2 Проєктування ШІМ-контролера 29

2.1 Загальна структура системи керування 29

2.2 Вибір параметрів ШІМ-контролера 31

2.2.1 Частота тактового сигналу 31

2.2.2 Частота та розрядність ШІМ 31
2.3 Проєктування модуля генерації ШІМ 33
2.3.1 Принцип роботи edge-aligned ШІМ 33
2.3.2 Принцип роботи center-aligned ШІМ 34
2.3.3 Інтерфейс модуля pwm_generator 34
2.4 Проєктування модуля генерації мертвого часу 35
2.4.1 Принцип формування мертвого часу 35
2.4.2 Детектор фронтів 35
2.4.3 Інтерфейс модуля dead_time_gen 35
2.5 Проєктування контролера H-моста 36
2.5.1 Режими роботи H-моста 36
2.5.2 Сумісність з драйвером L298N 36
2.6 Проєктування модуля інтерфейсу користувача 37
2.6.1 Модуль придушення брязкоту 37
2.6.2 Контролер коефіцієнта заповнення 37
2.6.3 Драйвер семисегментного індикатора 38
2.7 Модуль верхнього рівня 38
2.8 Висновки до розділу 2 39
3 Реалізація на FPGA 39
3.1 Вибір засобів розробки 39
3.1.1 Огляд виробників ПЛІС та їх продуктових лінійок 39
3.1.2 Мови опису апаратури 42
3.1.3 Середовища розробки та синтезу 44
3.1.4 Засоби функціонального моделювання 45
3.1.5 Обґрунтування вибору засобів розробки 46

3.2 Реалізація модуля генерації ШІМ 47
3.2.2 Тестбенч модуля pwm_generator 48
3.2.3 Результати моделювання генератора ШІМ 49
3.3 Реалізація модуля генерації мертвого часу 50
3.3.1 Ключові елементи реалізації 50
3.3.2 Тестбенч модуля dead_time_gen 51
3.3.3 Результати моделювання генератора мертвого часу 51
3.4 Реалізація контролера Н-моста та допоміжних модулів 53
3.5 Інтеграція та тестування системи верхнього рівня 54
3.5.1 Тестбенч верхнього рівня 54
3.5.2 Результати моделювання системи верхнього рівня 55
3.6 Висновки до розділу 3 56
Висновки 57
Перелік джерел посилання 59
Додаток А Слайди презентації 64

ВСТУП

Сучасний етап розвитку промисловості, транспорту та побутової техніки характеризується масовим впровадженням електроприводів з регульованою швидкістю обертання. За оцінками Міжнародного енергетичного агентства (IEA), електричні двигуни споживають близько 45% всієї електроенергії, що виробляється у світі, причому значна частка цього споживання припадає на системи з нерегульованим електроприводом. Впровадження сучасних методів керування швидкістю дозволяє суттєво підвищити енергоефективність промислових процесів, зменшити знос механічного обладнання та забезпечити оптимальні режими роботи технологічних установок.

Двигуни постійного струму (ДПС), незважаючи на конкуренцію з боку асинхронних та синхронних машин змінного струму, залишаються незамінними в багатьох галузях застосування. Їх унікальні переваги – лінійна залежність швидкості від напруги живлення, високий пусковий момент, широкий діапазон регулювання швидкості та простота реалізації систем керування – забезпечують їм стабільні позиції в таких сферах, як робототехніка, автомобільна електроніка, медичне обладнання, побутова техніка, системи автоматизації та прецизійні позиціонери. Особливо актуальним є застосування ДПС у портативних пристроях та системах з автономним живленням, де їх ефективність та простота керування є критично важливими факторами.

Широтно-імпульсна модуляція (ШІМ) є домінуючим методом керування швидкістю ДПС у сучасних системах електроприводу. Перевагами ШІМ над традиційними методами регулювання (реостатним, польовим) є висока енергетична ефективність (ККД сучасних ШІМ-контролерів досягає 95–98%), можливість плавного регулювання швидкості в широкому діапазоні, компактність силових перетворювачів та легка інтеграція з цифровими системами керування. Розвиток силової електроніки та напівпровідникових технологій зробив ШІМ-контролери доступними та надійними рішеннями для застосувань будь-якого

масштабу – від мініатюрних двигунів у медичних пристроях до потужних тягових приводів електротранспорту.

Традиційно ШІМ-контролери реалізовувалися на базі мікроконтролерів, які забезпечують необхідну функціональність за допомогою вбудованих апаратних таймерів та програмного забезпечення. Однак зростання вимог до продуктивності, точності та багатоканальності систем керування виявляє обмеження мікроконтролерного підходу. Послідовна архітектура процесора обмежує швидкодію при обробці багатьох каналів керування, фіксована кількість апаратних таймерів обмежує гнучкість системи, а програмна реалізація критичних функцій може призводити до небажаного джиттера (тремтіння) сигналів ШІМ. Ці обмеження стають критичними у високопродуктивних застосуваннях – багатоосьових системах керування роботами, прецизійних сервоприводах, високочастотних перетворювачах.

Програмовані логічні інтегральні схеми (ПЛІС, англ. FPGA – Field Programmable Gate Array) пропонують принципово інший підхід до реалізації систем керування, що базується на апаратній, а не програмній обробці сигналів. Паралельна архітектура ПЛІС дозволяє одночасно виконувати множину операцій без взаємного впливу, забезпечуючи детермінований час реакції та відсутність джиттера. Гнучкість конфігурації дозволяє реалізувати довільну кількість каналів ШІМ з індивідуальними параметрами, а можливість реконфігурації забезпечує швидку адаптацію до різних застосувань. Сучасні ПЛІС, такі як сімейства Intel Cyclone та MAX, поєднують високу продуктивність з помірною вартістю та низьким енергоспоживанням, що робить їх привабливою платформою для реалізації систем керування двигунами.

Актуальність дослідження підтверджується стійким зростанням ринку ПЛІС для промислової автоматизації та систем керування рухом. За даними аналітичних агентств, сегмент ПЛІС для керування двигунами демонструє щорічне зростання на рівні 8–10%, що обумовлено впровадженням концепції Індустрії 4.0, розвитком робототехніки та електромобільності. Провідні виробники ПЛІС – Intel (Altera), AMD (Xilinx), Microchip (Microsemi), Lattice Semiconductor – активно розвивають

спеціалізовані рішення для керування двигунами, включаючи готові IP-ядра, референсні проєкти та засоби розробки. Україна, маючи потужну школу електротехніки та традиції в галузі автоматизації, має значний потенціал для участі у цих глобальних тенденціях.

Таким чином, розробка ШІМ-контролера для керування швидкістю двигуна постійного струму на базі ПЛІС є актуальною науково-технічною задачею, що має практичне значення для створення сучасних високопродуктивних систем електроприводу. Опанування технологій проєктування цифрових систем на ПЛІС є важливою компетенцією для інженера в галузі електроніки та автоматизації, що визначає освітню цінність даної роботи.

1 АНАЛІТИЧНИЙ ОГЛЯД

1.1 Двигуни постійного струму та методи керування їх швидкістю

1.1.1 Загальні відомості про двигуни постійного струму

Двигуни постійного струму (ДПС) є одними з найбільш поширених електричних машин, що знайшли широке застосування в промисловості, транспорті, побутовій техніці та системах автоматизації. Незважаючи на розвиток технологій асинхронних та синхронних двигунів змінного струму, ДПС залишаються незамінними у багатьох застосуваннях завдяки своїм унікальним характеристикам: широкому діапазону регулювання швидкості, високому пусковому моменту, лінійній залежності між напругою та швидкістю обертання, а також відносній простоті систем керування [1, 2].

Конструкція типового ДПС складається з двох основних частин: статора та ротора (якоря). Статор створює основне магнітне поле двигуна і може бути виконаний на основі постійних магнітів або електромагнітів з обмотками збудження. Ротор (якір) являє собою циліндричну конструкцію з намотаними на неї обмотками, по яких протікає струм якоря. Взаємодія магнітного поля статора зі струмом якоря створює електромагнітний момент, що приводить ротор в обертання [3].

Важливим елементом конструкції щіткового ДПС є колекторно-щітковий вузол. Колектор складається з мідних пластин (ламелей), ізольованих одна від одної, до яких приєднані кінці секцій обмотки якоря. Щітки, виготовлені зазвичай з графіту або металографіту, ковзають по поверхні колектора, забезпечуючи підведення струму до обмоток якоря. Завдяки колектору відбувається комутація – перемикання напрямку струму в провідниках якоря при їх проходженні через нейтральну зону, що забезпечує безперервність обертального моменту [4].

За способом збудження магнітного поля розрізняють кілька типів ДПС. Двигуни з постійними магнітами (ДППМ) використовують постійні магніти для створення поля збудження, що робить їх компактними та ефективними для

застосувань малої та середньої потужності. Двигуни з незалежним збудженням мають окрему обмотку збудження, що живиться від незалежного джерела, дозволяючи гнучко регулювати магнітний потік. Двигуни послідовного збудження характеризуються з'єднанням обмотки збудження послідовно з якорем і відзначаються високим пусковим моментом. Двигуни паралельного (шунтового) збудження мають обмотку збудження, підключену паралельно якорю, що забезпечує жорстку механічну характеристику. Двигуни змішаного збудження поєднують переваги послідовного та паралельного збудження [5, 6].

Основні рівняння, що описують роботу ДПС, випливають з електромеханічних принципів перетворення енергії. Рівняння електричної рівноваги кола якоря має вигляд:

$$U = E + I \cdot R, \tag{1.1}$$

де U – напруга живлення,

E – електрорушійна сила (ЕРС),

I – струм якоря,

R – опір кола якоря.

ЕРС двигуна пропорційна швидкості обертання та магнітному потоку:

$$E = C \cdot \Phi \cdot n, \tag{1.2}$$

де C – конструктивна стала двигуна,

Ф – магнітний потік, n – швидкість обертання.

Електромагнітний момент визначається як:

$$M = C \cdot \Phi \cdot I. \tag{1.3}$$

З цих рівнянь випливає, що швидкість обертання ДПС можна виразити формулою:

$$n = (U - I \cdot R)/(C \cdot \Phi), \quad (1.4)$$

що є основою для аналізу методів регулювання швидкості [7].

Механічна характеристика ДПС – залежність швидкості обертання від моменту навантаження n = f(M) – є важливою експлуатаційною характеристикою. Для двигунів паралельного збудження ця характеристика є жорсткою (швидкість мало змінюється при зміні навантаження), для двигунів послідовного збудження – м'якою (швидкість значно зменшується при зростанні навантаження). Ця особливість визначає сфери застосування різних типів ДПС [8].

1.1.2 Класичні методи регулювання швидкості ДПС

Регулювання швидкості обертання є однією з ключових задач при експлуатації електроприводів з ДПС. Історично склалося кілька основних методів регулювання, кожен з яких має свої переваги та обмеження. Вибір методу залежить від вимог до діапазону регулювання, точності підтримання швидкості, енергетичної ефективності та економічної доцільності [9].

Реостатний метод регулювання передбачає введення додаткового опору в коло якоря двигуна. Із формули швидкості видно, що збільшення опору призводить до зменшення швидкості при тому ж навантаженні. Цей метод є найпростішим з точки зору реалізації, однак має суттєві недоліки: значні втрати енергії на нагрівання реостата (втрати пропорційні $I^2 \cdot R_{дод}$), зміна жорсткості механічної характеристики, неможливість регулювання вгору від номінальної швидкості. Метод застосовується переважно для короткочасного зниження швидкості або для плавного пуску двигунів [10].

Регулювання зміною напруги живлення якоря є більш досконалим методом, що забезпечує регулювання швидкості вниз від номінального значення при збереженні жорсткості механічної характеристики. Відповідно до формули 1.4, зменшення напруги U призводить до пропорційного зменшення швидкості при постійному магнітному потоці. Цей метод характеризується високою енергетичною ефективністю, оскільки відсутні додаткові втрати на резисторах. Для реалізації методу традиційно використовували системи генератор-двигун (система

Леонарда-Ільгнера), керовані випрямлячі на тиристорах, а в сучасних системах – імпульсні перетворювачі напруги [11].

Регулювання зміною магнітного потоку (польове регулювання) здійснюється шляхом зміни струму збудження. З формули швидкості видно, що зменшення магнітного потоку Ф призводить до збільшення швидкості обертання. Цей метод дозволяє регулювати швидкість вгору від номінального значення, однак має обмеження: послаблення поля призводить до зменшення максимального моменту двигуна, погіршуються умови комутації, що обмежує глибину послаблення поля зазвичай у 2-3 рази. Метод є економічним, оскільки потужність кола збудження становить лише 1-5% від номінальної потужності двигуна [12].

Сучасні системи електроприводу часто використовують комбіноване регулювання: в діапазоні нижче номінальної швидкості застосовується регулювання напругою якоря при номінальному потоці, а в діапазоні вище номінальної – польове регулювання при номінальній напрузі. Це дозволяє отримати широкий діапазон регулювання швидкості (до 1:1000 і більше) при збереженні високої енергетичної ефективності [13].

1.1.3 Широтно-імпульсна модуляція як метод керування швидкістю

Широтно-імпульсна модуляція (ШІМ, англ. PWM – Pulse Width Modulation) є сучасним і найбільш поширеним методом керування швидкістю ДПС. Принцип ШІМ полягає у перетворенні постійної напруги джерела живлення у послідовність імпульсів змінної тривалості, що дозволяє регулювати середнє значення напруги, прикладеної до навантаження. На відміну від лінійних регуляторів, ШІМ-керування забезпечує високу енергетичну ефективність, оскільки ключові елементи (транзистори) працюють у режимі перемикання з мінімальними втратами [14].

Основними параметрами ШІМ-сигналу є частота комутації (f) і коефіцієнт заповнення (duty cycle, D). Коефіцієнт заповнення визначається як відношення тривалості імпульсу (ton) до періоду ШІМ (T): D = ton/T, і зазвичай виражається у відсотках від 0% до 100%. Середня напруга на навантаженні визначається як добуток напруги живлення на коефіцієнт заповнення: Ucep = U·D. Таким чином,

змінюючи коефіцієнт заповнення, можна плавно регулювати середню напругу від нуля до повної напруги живлення [15].

Вибір частоти ШІМ є важливим компромісом між кількома факторами. Занадто низька частота (нижче 100 Гц) може призвести до нерівномірного обертання двигуна (пульсацій моменту), що проявляється як вібрація та акустичний шум. При частотах в діапазоні людського слуху (20 Гц – 20 кГц) можливе виникнення характерного свистячого звуку від двигуна. Занадто висока частота збільшує комутаційні втрати в ключових елементах та може погіршити електромагнітну сумісність через збільшення рівня високочастотних завад. Для більшості застосувань з ДПС оптимальною є частота ШІМ в діапазоні 1-25 кГц [16, 17].

Індуктивність обмоток двигуна відіграє важливу роль у роботі ШІМ-керування. Завдяки індуктивності, струм у обмотці не може змінюватися миттєво, що призводить до згладжування пульсацій струму. При ввімкненому ключі струм наростає експоненційно з постійною часу:

$$\tau = L/R, \tag{1.5}$$

де L – індуктивність обмотки,

R – активний опір.

При вимкненому ключі струм продовжує протікати через зворотний діод (flyback diode), поступово зменшуючись. Величина пульсацій струму обернено пропорційна частоті ШІМ та індуктивності обмотки. Для коректної роботи бажано, щоб стала часу обмотки була значно більшою за період ШІМ [18].

Переваги ШІМ-керування над іншими методами регулювання швидкості ДПС є численними. По-перше, висока енергетична ефективність: ККД сучасних ШІМ-контролерів досягає 95-98%, оскільки втрати потужності виникають лише під час перемикання транзисторів та від їх опору у відкритому стані. По-друге, широкий діапазон регулювання швидкості – від 0 до 100% номінальної при збереженні жорсткості механічної характеристики. По-третє, компактність та мала

вага силових перетворювачів порівняно з трансформаторними джерелами живлення. По-четверте, можливість легкої інтеграції з цифровими системами керування, включаючи мікроконтролери та ПЛІС. Нарешті, ШІМ-керування забезпечує плавний пуск двигуна без кидків струму та механічних ударів [19, 20].

1.2 Силова електроніка для керування ДПС

1.2.1 Напівмостова та мостова схеми керування

Для реалізації ШІМ-керування двигунами постійного струму використовуються силові напівпровідникові ключі, зібрані за схемою Н-моста (повного моста) або напівмоста. Вибір схеми залежить від вимог до функціональності системи керування, зокрема від необхідності реверсування напрямку обертання та режимів гальмування двигуна [21].

Напівмостова схема (half-bridge) складається з двох ключових елементів, з'єднаних послідовно між шинами живлення, та двигуна, підключеного між середньою точкою напівмоста та нульовою шиною (або другим джерелом живлення). Ця схема дозволяє керувати швидкістю двигуна в одному напрямку обертання і є найпростішим рішенням для застосувань, де реверс не потрібен. При використанні ШІМ верхній ключ комутується з частотою ШІМ, а нижній служить для забезпечення шляху протікання струму при вимкненому верхньому ключі (синхронне випрямлення) або замінюється діодом [22].

Н-міст (full-bridge) є найбільш універсальною схемою для керування ДПС, що забезпечує повний контроль над напрямком та швидкістю обертання. Схема складається з чотирьох ключових елементів, розташованих у вигляді літери "Н", з двигуном посередині. Позначимо ключі як Q1, Q2 (верхні) та Q3, Q4 (нижні). При ввімкненні діагональної пари Q1-Q4 струм протікає через двигун в одному напрямку, забезпечуючи обертання за годинниковою стрілкою. При ввімкненні

пари Q2-Q3 струм протікає у зворотному напрямку, забезпечуючи обертання проти годинникової стрілки [23].

Критично важливим аспектом роботи Н-моста є запобігання наскрізним струмам (shoot-through). Наскрізний струм виникає, коли одночасно ввімкнені обидва ключі однієї стійки (Q1 і Q3 або Q2 і Q4), що призводить до короткого замикання джерела живлення. Для запобігання цій ситуації між вимкненням одного ключа та ввімкненням іншого в тій же стійці вводиться захисна пауза – "мертвий час" (dead time). Тривалість мертвого часу обирається з урахуванням часу вимкнення транзисторів (зазвичай від 100 нс до декількох мікросекунд) і є важливим параметром ШІМ-контролера [24].

Існує кілька стратегій ШІМ-керування Н-мостом. При уніполярному (однополярному) ШІМ модулюється лише одна стійка моста, а друга залишається у статичному стані, визначаючи напрямок обертання. Ця схема характеризується меншими комутаційними втратами та нижчим рівнем пульсацій струму. При біполярному ШІМ обидві діагональні пари ключів комутуються в протифазі, що забезпечує більший діапазон керування напругою (від -U до +U) та кращі динамічні характеристики, але супроводжується вищими комутаційними втратами. Вибір стратегії залежить від конкретних вимог застосування [25].

1.2.2 Інтегральні драйвери двигунів

Сучасна елементна база пропонує широкий вибір інтегральних драйверів двигунів, що значно спрощують проєктування систем керування. Ці мікросхеми інтегрують силові ключі Н-моста, схеми захисту, логіку керування та інтерфейси для підключення до мікроконтролерів або ПЛІС. Найбільш популярними представниками є мікросхеми серій L298, L293, DRV8xxx від Texas Instruments, VNH від STMicroelectronics та інші [26].

Мікросхема L298N є класичним рішенням для керування двигунами середньої потужності. Вона містить два незалежних Н-мости, що дозволяє керувати двома ДПС або одним біполярним кроковим двигуном. Кожен канал забезпечує струм до 2 А (піковий до 3 А) при напрузі живлення від 5 до 46 В. Входи керування є TTL-сумісними, що забезпечує пряме підключення до цифрових виходів

мікроконтролера або ПЛІС. Однак L298N має суттєвий недолік – значне падіння напруги на вихідних транзисторах (до 2-3 В при повному струмі), що знижує ефективність та вимагає запасу по напрузі живлення [27].

Сучасні драйвери на основі MOSFET-транзисторів, такі як серія DRV8xxx від Texas Instruments, забезпечують значно вищу ефективність завдяки низькому опору каналу транзисторів у відкритому стані (RDS(on) порядку десятків мілі-Ом). Наприклад, DRV8833 забезпечує керування двома двигунами з струмом до 1.5 А кожен при падінні напруги менше 0.5 В. Ці мікросхеми часто включають вбудовані функції захисту від перевантаження по струму, перегріву, пониженої напруги живлення, а також вбудовану генерацію мертвого часу [28].

При проєктуванні силової частини необхідно враховувати ряд важливих факторів. По-перше, зворотні діоди (flyback diodes) повинні бути присутні паралельно кожному ключу для забезпечення шляху протікання струму індуктивного навантаження при вимкненні ключа. Сучасні інтегральні драйвери зазвичай мають вбудовані діоди. По-друге, необхідна фільтрація по живленню – конденсатори фільтра повинні розташовуватися якомога ближче до силових виводів мікросхеми. По-третє, при високих струмах необхідно передбачити відведення тепла від мікросхеми драйвера через радіатор або мідний полігон на друкованій платі [29].

1.3 Програмовані логічні інтегральні схеми як платформа для систем керування

1.3.1 Загальні відомості про ПЛІС

Програмовані логічні інтегральні схеми (ПЛІС, англ. FPGA – Field Programmable Gate Array) є класом цифрових мікросхем, внутрішня структура яких може бути сконфігурована користувачем після виготовлення. На відміну від мікроконтролерів, де функціональність визначається програмним кодом, що

виконується процесором, ПЛІС дозволяють створювати апаратні цифрові схеми довільної структури, що забезпечує максимальну гнучкість та продуктивність [30].

Архітектура типової ПЛІС включає масив конфігурованих логічних блоків (CLB – Configurable Logic Blocks), систему програмованих з'єднань (interconnects) та блоки введення-виведення (IOB – Input/Output Blocks). Кожен логічний блок містить таблиці перетворення (LUT – Look-Up Table), що реалізують довільні булеві функції від кількох змінних (зазвичай 4-6), та тригери для реалізації послідовної логіки. Система з'єднань дозволяє об'єднувати логічні блоки у складні цифрові схеми. Блоки введення-виведення забезпечують інтерфейс з зовнішнім світом та підтримують різноманітні стандарти сигналів [31].

Сучасні ПЛІС містять додаткові спеціалізовані ресурси: вбудовані блоки пам'яті (Block RAM) для зберігання даних, блоки цифрової обробки сигналів (DSP – Digital Signal Processing) з апаратними помножувачами та акумуляторами, блоки керування тактовою частотою (PLL – Phase-Locked Loop, DCM – Digital Clock Manager) для синтезу та розподілу тактових сигналів, високошвидкісні послідовні приймачі (SERDES) для реалізації інтерфейсів типу PCIe, Ethernet тощо. Деякі сімейства ПЛІС (SoC FPGA) містять вбудовані процесорні ядра ARM, що дозволяє поєднувати переваги програмного та апаратного підходів в одному кристалі [32].

За технологією зберігання конфігурації ПЛІС поділяються на два основні типи. SRAM-based ПЛІС зберігають конфігурацію в статичній пам'яті, яка потребує зовнішнього носія (Flash-пам'ять) та завантаження при кожному увімкненні живлення. Flash-based ПЛІС мають вбудовану енергонезалежну пам'ять і зберігають конфігурацію при вимкненні живлення. Прикладом першого типу є більшість пристроїв сімейств Cyclone, Stratix (Intel/Altera) та Artix, Kintex, Virtex (AMD/Xilinx). До другого типу належать сімейства MAX 10 (Intel) та iCE40, MachXO (Lattice) [33].

1.3.2 ПЛІС Intel сімейств Cyclone та MAX

Компанія Intel (раніше Altera) пропонує широкий спектр ПЛІС для різних застосувань. Для систем керування двигунами та інших застосувань середньої складності найбільш придатними є сімейства Cyclone та MAX, що поєднують

достатню функціональність з помірною вартістю та низьким енергоспоживанням [34].

Сімейство Cyclone є найбільш популярною серією ПЛІС Intel для застосувань у промисловій автоматизації, телекомунікаціях та споживчій електроніці. Поточне покоління Cyclone 10 представлене двома варіантами: Cyclone 10 LP (Low Power) для застосувань з мінімальним енергоспоживанням та Cyclone 10 GX з вбудованими високошвидкісними трансиверами. Cyclone 10 LP містить від 6 до 120 тисяч логічних елементів, до 5 Мбіт вбудованої пам'яті та до 144 множників 18×18. Пристрої виготовляються за технологією 60 нм (LP) та 20 нм (GX) і підтримують широкий спектр стандартів введення-виведення [35].

Сімейство MAX 10 займає особливе місце серед ПЛІС Intel завдяки вбудованій Flash-пам'яті для зберігання конфігурації та програм користувача. Це усуває потребу у зовнішній конфігураційній пам'яті та забезпечує миттєву готовність до роботи при подачі живлення. Унікальною особливістю MAX 10 є вбудовані аналого-цифрові перетворювачі (АЦП) з розрядністю 12 біт та швидкістю до 1 MSPS, що дозволяє безпосередньо підключати аналогові датчики без зовнішніх АЦП. MAX 10 містить від 2 до 50 тисяч логічних елементів, що достатньо для реалізації ШІМ-контролерів та систем керування середньої складності [36].

Для розробки під ПЛІС Intel компанія надає інтегроване середовище проєктування Quartus Prime. Програмне забезпечення існує у трьох редакціях: Lite Edition (безкоштовна, підтримує Cyclone, MAX та частину інших сімейств), Standard Edition та Pro Edition для високопродуктивних пристроїв. Quartus Prime забезпечує повний цикл розробки: введення проєкту (схемне або на мовах опису апаратури), синтез та оптимізацію логіки, розміщення та трасування (place and route), часовий аналіз, симуляцію та програмування пристрою. Середовище інтегрується з симулятором ModelSim (Questa) для функціональної верифікації проєктів [37].

1.3.3 Переваги ПЛІС над мікроконтролерами для задач керування двигунами

Порівняння ПЛІС та мікроконтролерів для реалізації систем керування двигунами виявляє суттєві переваги апаратного підходу для багатьох застосувань. Основні відмінності випливають з фундаментальної різниці в архітектурі: мікроконтролер виконує програмні інструкції послідовно, тоді як ПЛІС реалізує апаратні схеми, що працюють паралельно [38].

Паралелізм обробки є ключовою перевагою ПЛІС. У типовій системі керування кількома двигунами на мікроконтролері обробка кожного каналу ШІМ, читання датчиків та обчислення регуляторів виконуються послідовно, що обмежує швидкодію системи. На ПЛІС усі ці функції реалізуються як незалежні апаратні блоки, що працюють одночасно. Це дозволяє досягти надзвичайно малої латентності та детермінованого часу реакції – критично важливих параметрів для прецизійного керування [39].

Детермінізм часової поведінки ПЛІС означає, що час виконання будь-якої операції є строго фіксованим та визначається тактовою частотою кристала. На відміну від мікроконтролера, де час виконання програми може варіюватися залежно від розгалужень коду, переривань та кешування, ПЛІС забезпечує абсолютно передбачувану поведінку. Для систем керування в реальному часі це гарантує стабільну роботу без джиттера (тремтіння) сигналів ШІМ [40].

Гнучкість конфігурації периферії ПЛІС дозволяє реалізувати довільну кількість каналів ШІМ з будь-якими параметрами (розрядність, частота, фазовий зсув) без обмежень вбудованих таймерів мікроконтролера. Типовий мікроконтролер має фіксовану кількість апаратних ШІМ-каналів (зазвичай 4-12), тоді як на ПЛІС можна реалізувати десятки незалежних каналів з індивідуальними налаштуваннями. Це особливо важливо для багатоосьових систем керування та роботизованих комплексів [41].

Реконфігурованість ПЛІС дозволяє швидко адаптувати систему до різних типів двигунів та алгоритмів керування без зміни апаратної частини. При зміні вимог до системи достатньо завантажити нову конфігурацію, що значно скорочує

час виходу на ринок при розробці лінійки продуктів. Деякі ПЛІС підтримують часткову реконфігурацію – можливість змінювати частину логіки без зупинки решти системи [42].

Водночас ПЛІС мають певні недоліки порівняно з мікроконтролерами: вища вартість та споживана потужність для простих застосувань, складніший процес розробки (вимагає знання мов опису апаратури), більші габарити друкованої плати через необхідність зовнішньої конфігураційної пам'яті (для SRAM-based ПЛІС). Для простих застосувань з одним-двома двигунами використання мікроконтролера часто є більш доцільним. Однак для високопродуктивних багатоканальних систем керування переваги ПЛІС є беззаперечними [43].

1.4 Реалізація ШІМ-контролерів на ПЛІС

1.4.1 Архітектура цифрового ШІМ-генератора

Цифровий ШІМ-генератор є базовим компонентом системи керування двигуном на ПЛІС. Його архітектура визначається вимогами до роздільної здатності, частоти ШІМ та додаткової функціональності. Найпростіша реалізація містить два основних елементи: лічильник та компаратор [44].

Лічильник генерує опорний сигнал пилкоподібної або трикутної форми. При використанні інкрементного лічильника, що рахує від 0 до максимального значення і скидається в 0, формується пилкоподібний сигнал (edge-aligned PWM). При використанні реверсивного лічильника, що рахує від 0 до максимуму і назад до 0, формується трикутний сигнал (center-aligned PWM). Центрально-вирівняний ШІМ має переваги для керування двигунами: симетричність імпульсів зменшує рівень гармонік та електромагнітних завад [45].

Компаратор порівнює поточне значення лічильника з заданим значенням коефіцієнта заповнення (duty cycle). Вихід компаратора встановлюється в '1', коли значення лічильника менше за задане, та в '0' в іншому випадку. Таким чином,

співвідношення часу, коли вихід дорівнює '1', до повного періоду визначається відношенням заданого значення до максимального значення лічильника [46].

Розрядність лічильника визначає роздільну здатність ШІМ. N-бітний лічильник забезпечує 2^N рівнів коефіцієнта заповнення, що відповідає роздільній здатності $100\%/2^N$. Наприклад, 8-бітний лічильник дає 256 рівнів (роздільна здатність ~0.4%), 10-бітний – 1024 рівні (~0.1%), 12-бітний – 4096 рівнів (~0.025%). Вибір розрядності залежить від вимог до точності керування швидкістю двигуна [47].

Частота ШІМ визначається тактовою частотою ПЛІС та розрядністю лічильника: $f\_PWM = f\_CLK / 2^N$ для пилкоподібного ШІМ та $f\_PWM = f\_CLK / (2 \times 2^N)$ для центрально-вирівняного. Наприклад, при тактовій частоті 50 МГц та 10-бітному лічильнику частота пилкоподібного ШІМ становить 50 МГц / 1024 ≈ 48.8 кГц. Для досягнення нижчої частоти ШІМ при збереженні роздільної здатності можна використати попередній подільник тактової частоти або збільшити модуль лічильника [48].

1.4.2 Генерація мертвого часу для Н-моста

Як зазначалося раніше, для безпечної роботи Н-моста необхідно забезпечити мертвий час між вимкненням одного ключа стійки та ввімкненням іншого. Реалізація генератора мертвого часу на ПЛІС є відносно простою та надійною задачею, що дозволяє точно контролювати цей критичний параметр [49].

Найпростіший підхід до генерації мертвого часу полягає у затримці фронтів ШІМ-сигналу. Сигнал для верхнього ключа затримується на час t_dead при переході з '0' в '1' (наростаючий фронт), сигнал для нижнього ключа – при переході з '1' в '0' (спадаючий фронт). Це забезпечує, що один ключ гарантовано вимкнеться перед ввімкненням іншого. Затримка реалізується за допомогою ланцюжка тригерів (shift register) або лічильника, тактованого від швидкого тактового сигналу [50].

Роздільна здатність встановлення мертвого часу визначається періодом тактового сигналу. При тактовій частоті 100 МГц крок становить 10 нс, що забезпечує достатню точність для більшості застосувань. Для типових MOSFET- та

IGBT-транзисторів час вимкнення становить від десятків наносекунд до кількох мікросекунд, тому мертвий час зазвичай обирається в діапазоні 100 нс – 2 мкс з урахуванням запасу безпеки [51].

Важливим аспектом є можливість програмного налаштування мертвого часу. У ПЛІС-реалізації це досягається через використання конфігураційного регістра, значення якого визначає кількість тактів затримки. Це дозволяє оптимізувати мертвий час для конкретного драйвера та транзисторів без зміни апаратного проєкту, а також забезпечити можливість адаптації до різних режимів роботи (наприклад, збільшення мертвого часу при високих струмах) [52].

1.4.3 Мови опису апаратури для проєктування ШІМ-контролерів

Для проєктування цифрових схем на ПЛІС використовуються мови опису апаратури (HDL – Hardware Description Language). Двома найбільш поширеними HDL є Verilog та VHDL, кожна з яких має свої переваги та сфери застосування [53].

Verilog (IEEE 1364) має синтаксис, подібний до мови програмування C, що робить його більш доступним для розробників з досвідом програмування. Мова підтримує різні рівні абстракції: від поведінкового опису до структурного на рівні логічних елементів. Verilog широко використовується в комерційних проєктах, особливо в США та Азії. Розширення SystemVerilog (IEEE 1800) додає потужні можливості для верифікації та об'єктно-орієнтоване проєктування [54].

VHDL (IEEE 1076) є більш суворо типізованою мовою з синтаксисом, що нагадує мову Ada. Сувора типізація зменшує ймовірність помилок при проєктуванні, але вимагає більш детального опису. VHDL традиційно популярна в Європі та в аерокосмічній/оборонній галузях, де цінується надійність та чітка документованість проєктів. Мова забезпечує потужні можливості параметризації через механізм generic, що дозволяє створювати універсальні повторно використовувані компоненти [55].

При проєктуванні ШІМ-контролера на HDL код зазвичай організується у вигляді ієрархії модулів. Верхній рівень (top-level module) визначає зовнішній інтерфейс системи та з'єднує внутрішні компоненти. Типова структура включає: модуль генератора ШІМ (лічильник та компаратор), модуль генерації мертвого

часу, модуль інтерфейсу керування (кнопки, енкодер, UART або SPI), модуль захисту (обмеження струму, перевантаження). Модульний підхід полегшує тестування, повторне використання та модифікацію проєкту [56].

Верифікація проєкту є критично важливим етапом розробки. Для цього створюються тестові оточення (testbench), що генерують вхідні сигнали та перевіряють коректність вихідних. Сучасні методології верифікації, такі як Universal Verification Methodology (UVM), забезпечують систематичний підхід до функціонального тестування. Для ШІМ-контролера верифікація включає перевірку правильності частоти та коефіцієнта заповнення, коректності генерації мертвого часу, відсутності заборонених комбінацій сигналів керування H-мостом [57].

1.4.4 Аналіз наявних реалізацій ШІМ-контролерів на ПЛІС

Аналіз науково-технічної літератури та відкритих проєктів виявляє широкий спектр реалізацій ШІМ-контролерів на ПЛІС, від простих навчальних прикладів до промислових систем керування. Розглянемо основні підходи та їх характеристики [58].

Виробники ПЛІС надають готові IP-ядра (Intellectual Property cores) для генерації ШІМ. Intel пропонує IP-ядро PWM Controller у складі Platform Designer (раніше Qsys), що забезпечує конфігуровану кількість каналів, програмовану роздільну здатність та частоту, інтерфейс Avalon-MM для інтеграції з процесорними системами. AMD/Xilinx має подібні рішення у складі AXI Timer IP та MicroBlaze PWM. Ці IP-ядра оптимізовані для відповідних сімейств ПЛІС та ретельно верифіковані, однак можуть бути надлишковими для простих застосувань.

Компанія Microchip (раніше Microsemi) пропонує спеціалізовані рішення для керування двигунами на базі своїх ПЛІС сімейств SmartFusion та PolarFire. Бібліотека IP-ядер включає генератор трифазного ШІМ з вбудованим мертвим часом, блоки перетворення Кларк та Парка для векторного керування, контролери простору станів векторного керування (Field-Oriented Control). Ці рішення орієнтовані на професійні застосування в промисловій автоматизації.

У науковій літературі представлено численні дослідження з реалізації ШІМ-контролерів на ПЛІС. Значна частина робіт присвячена оптимізації ресурсів та підвищенню частоти роботи. Деякі автори досліджують нетрадиційні архітектури, такі як фазо-модульований ШІМ для зменшення гармонік, сигма-дельта модуляція для підвищення роздільної здатності, багаторівневий ШІМ для високовольтних застосувань. Результати цих досліджень демонструють, що ПЛІС-реалізації можуть досягати частот ШІМ в діапазоні мегагерців з роздільною здатністю наносекундного рівня.

Порівняльний аналіз ПЛІС-реалізацій з мікроконтролерними рішеннями підтверджує переваги ПЛІС для вимогливих застосувань. Дослідження показують, що ПЛІС-контролери забезпечують на порядок меншу латентність керування, вищу точність часових параметрів та здатність обслуговувати значно більшу кількість осей керування порівняно з еквівалентними за вартістю мікроконтролерами.

1.5 Висновки до розділу 1

У першому розділі дипломної роботи проведено комплексний аналітичний огляд сучасного стану проблеми керування швидкістю двигунів постійного струму та засобів її вирішення на базі програмованих логічних інтегральних схем.

Досліджено принципи роботи двигунів постійного струму різних типів збудження та встановлено, що ДПС з незалежним збудженням та двигуни з постійними магнітами найкраще підходять для систем з регульованою швидкістю завдяки лінійній залежності швидкості від напруги якоря та високому пусковому моменту. Проаналізовано класичні методи регулювання швидкості ДПС, включаючи реостатний метод, метод зміни напруги живлення та метод зміни магнітного потоку, і визначено їх обмеження за енергоефективністю, діапазоном регулювання та складністю реалізації.

Встановлено, що широтно-імпульсна модуляція є найефективнішим сучасним методом керування швидкістю ДПС, який забезпечує ККД силового перетворювача на рівні 95–98%, можливість плавного регулювання в широкому діапазоні швидкостей та легку інтеграцію з цифровими системами керування. Визначено основні параметри ШІМ-сигналу, що впливають на якість керування: частота модуляції, розрядність (роздільна здатність) та режим модуляції. Обґрунтовано вибір частоти ШІМ вище 20 кГц для усунення акустичного шуму та розрядності 8–12 біт для забезпечення достатньої плавності регулювання.

Розглянуто схемотехніку силової частини систем керування ДПС, зокрема напівмостові та мостові (Н-міст) схеми на базі MOSFET та IGBT транзисторів. Особливу увагу приділено проблемі наскрізних струмів при перемиканні ключів одного плеча та методам її вирішення шляхом введення мертвого часу. Проаналізовано характеристики інтегрованих драйверів двигунів L298N та серії DRV8xxx, які спрощують побудову силової частини системи керування.

Детально досліджено архітектуру програмованих логічних інтегральних схем як платформи для реалізації систем керування двигунами. Розглянуто базову структуру ПЛІС, що включає конфігуровані логічні блоки, блоки вводу/виводу, матрицю з'єднань та спеціалізовані ресурси. Проведено порівняльний аналіз ПЛІС Intel сімейств Cyclone та MAX, визначено їх характеристики та сфери застосування. Сімейство MAX 10 виділяється наявністю вбудованого аналого-цифрового перетворювача, що може бути корисним для реалізації зворотного зв'язку в системах керування.

Обґрунтовано переваги використання ПЛІС над мікроконтролерами для задач керування двигунами. Паралельна архітектура ПЛІС забезпечує одночасне виконання множини операцій без взаємного впливу, детермінований час реакції на події та відсутність джиттера сигналів ШІМ. Гнучкість конфігурації дозволяє реалізувати довільну кількість каналів ШІМ з індивідуальними параметрами та швидко адаптувати систему до нових вимог шляхом реконфігурації.

Розглянуто архітектуру цифрового ШІМ-генератора на базі лічильника та компаратора, проаналізовано особливості режимів edge-aligned та center-aligned

модуляції. Center-aligned режим забезпечує кращі спектральні характеристики вихідного сигналу та зменшує електромагнітні завади, хоча вдвічі знижує частоту ШІМ при тій самій тактовій частоті. Описано методи генерації мертвого часу для захисту силових ключів H-моста та принципи опису цифрових систем на мовах VHDL та Verilog.

Проведено огляд існуючих рішень у галузі реалізації ШІМ-контролерів на ПЛІС, включаючи комерційні IP-ядра провідних виробників та академічні розробки. Аналіз показав, що комерційні рішення часто є надмірно складними та дорогими для простих застосувань, тоді як академічні проєкти зазвичай мають обмежену функціональність або недостатню документацію.

На основі проведеного аналізу сформульовано мету та задачі дипломної роботи, спрямовані на розробку модульного параметризованого ШІМ-контролера на ПЛІС Intel з підтримкою різних режимів модуляції, захистом силових ключів від наскрізних струмів та зручним інтерфейсом користувача. Результати аналітичного огляду створюють теоретичне підґрунтя для виконання подальших етапів роботи: проєктування структури системи, реалізації на мові опису апаратури та експериментальної перевірки.

2 ПРОЄКТУВАННЯ ШІМ-КОНТРОЛЕРА

2.1 Загальна структура системи керування

На основі проведеного аналітичного огляду та сформульованих задач дипломної роботи розроблено структурну схему системи керування швидкістю двигуна постійного струму. Система побудована за модульним принципом, що забезпечує гнучкість конфігурації, можливість повторного використання окремих компонентів та спрощує процес верифікації та налагодження.

Розроблена система включає такі основні функціональні блоки:

– модуль генерації ШІМ-сигналу (pwm_generator) – формує широтно-імпульсно модульований сигнал із заданим коефіцієнтом заповнення та підтримкою двох режимів модуляції;

– модуль генерації мертвого часу (dead_time_gen) – забезпечує захист силових ключів Н-моста від наскрізних струмів шляхом введення часової затримки між перемиканнями комплементарних сигналів;

– контролер Н-моста (hbridge_controller) – формує сигнали керування для драйвера двигуна з підтримкою режимів прямого та зворотного обертання, гальмування та вільного вибігу;

– модуль придушення брязкоту (debouncer) – фільтрує механічний брязкіт контактів кнопок керування та формує чисті імпульси для подальшої обробки;

– контролер коефіцієнта заповнення (duty_controller) – забезпечує керування швидкістю двигуна за допомогою кнопок або квадратурного енкодера з функцією автоповтору;

– драйвер семисегментного індикатора (seg7_driver) – відображає поточне значення коефіцієнта заповнення у відсотках на трирозрядному дисплеї;

– модуль верхнього рівня (pwm_motor_controller_top) – інтегрує всі компоненти системи та забезпечує їх взаємодію.

Ієрархічна структура проєкту дозволяє незалежно розробляти, тестувати та модифікувати кожен модуль. Кожен компонент має чітко визначений інтерфейс, що спрощує інтеграцію та забезпечує можливість заміни окремих модулів без впливу на решту системи. Така архітектура відповідає сучасним принципам проєктування цифрових систем та полегшує супровід і розвиток проєкту.

2.2 Вибір параметрів ШІМ-контролера

2.2.1 Частота тактового сигналу

Ефективність роботи системи керування двигуном суттєво залежить від правильного вибору параметрів ШІМ-контролера. Основними параметрами, що підлягають визначенню, є: частота тактового сигналу, частота ШІМ, розрядність (роздільна здатність) ШІМ та тривалість мертвого часу.

Як тактовий сигнал системи обрано частоту 50 МГц, що є стандартною для більшості налагоджувальних плат на базі ПЛІС Intel (зокрема, плати з сімейств Cyclone та MAX). Ця частота забезпечує достатню часову роздільну здатність для формування ШІМ-сигналу та реалізації точного мертвого часу, при цьому не створюючи надмірних вимог до швидкодії логічних елементів ПЛІС.

Період тактового сигналу становить:

$$T_{clk} = 1 / f_{clk} = 1 / 50 \times 10^6 = 20 \text{ нс}. \quad (2.1)$$

Цей період визначає мінімальну дискретність часових інтервалів у системі та безпосередньо впливає на точність формування мертвого часу.

2.2.2 Частота та розрядність ШІМ

Розрядність ШІМ визначає кількість дискретних рівнів коефіцієнта заповнення та, відповідно, точність регулювання швидкості двигуна. У розробленій системі обрано 8-бітну розрядність, що забезпечує 256 рівнів регулювання (від 0% до 100% з кроком приблизно 0,4%). Така точність є достатньою для більшості практичних застосувань і забезпечує плавне регулювання швидкості без помітних ступінчастих змін.

Частота ШІМ-сигналу пов'язана з тактовою частотою та розрядністю співвідношенням:

$$f_{PWM} = f_{clk} / (2^{\text{PRESCALER\_BITS}} \times 2^{\text{PWM\_BITS}}), \quad (2.2)$$

де PRESCALER_BITS – розрядність прескалера (подільника частоти), PWM_BITS – розрядність ШІМ.

При використанні 8-бітної розрядності без прескалера (PRESCALER_BITS = 0) отримуємо:

$$f_{PWM} = 50 \times 10^6 / 256 \approx 195{,}3 \text{ кГц}. \quad (2.3)$$

Ця частота знаходиться вище діапазону чутності людського слуху (20 Гц – 20 кГц), що усуває акустичний шум від роботи ШІМ-контролера. Водночас, вона є достатньо низькою для ефективної роботи силових транзисторів без надмірних комутаційних втрат.

Для застосувань, що вимагають нижчої частоти ШІМ, система передбачає можливість використання прескалера. Наприклад, при PRESCALER_BITS = 2 частота ШІМ становитиме приблизно 48,8 кГц, а при PRESCALER_BITS = 4 – приблизно 12,2 кГц.

2.2.3 Мертвий час

Мертвий час (dead time) – це часовий інтервал, протягом якого обидва ключі одного плеча Н-моста знаходяться у вимкненому стані. Цей інтервал є критично важливим для запобігання наскрізному струму (shoot-through), який виникає при одночасному відкритті верхнього та нижнього транзисторів і може призвести до їх пошкодження або виходу з ладу.

Тривалість мертвого часу повинна бути більшою за час вимкнення найповільнішого транзистора в схемі. Для типових MOSFET-транзисторів час вимкнення складає 50–200 нс, для IGBT – 200–500 нс. У розробленій системі обрано мертвий час 500 нс, що забезпечує надійний захист для широкого спектра силових ключів.

Значення мертвого часу в тактах системного годинника:

$$N_{dead} = T_{dead} / T_{clk} = 500 \text{ нс} / 20 \text{ нс} = 25 \text{ тактів}. \quad (2.4)$$

Система дозволяє програмно змінювати тривалість мертвого часу в діапазоні від 0 до 255 тактів (від 0 до 5,1 мкс), що забезпечує можливість адаптації до різних типів силових ключів.

Таблиця 2.1 – Основні параметри ШІМ-контролера

Параметр	**Значення**
Тактова частота	50 МГц
Розрядність ШІМ	8 біт (256 рівнів)
Частота ШІМ	≈195 кГц (без прескалера)
Мертвий час	500 нс (25 тактів)
Час придушення брязкоту	20 мс
Режими ШІМ	Edge-aligned, Center-aligned

2.3 Проєктування модуля генерації ШІМ

2.3.1 Принцип роботи edge-aligned ШІМ

Модуль генерації ШІМ (pwm_generator) є центральним компонентом системи, що безпосередньо формує широтно-імпульсно модульований сигнал. Архітектура модуля базується на класичній схемі «лічильник + компаратор» і підтримує два режими роботи: edge-aligned (з вирівнюванням по фронту) та center-aligned (з центральним вирівнюванням).

У режимі edge-aligned лічильник здійснює рахунок від нуля до максимального значення ($2^{PWM\_BITS} – 1 = 255$ для 8-бітної розрядності) і скидається в нуль, формуючи пилкоподібний сигнал. Вихідний ШІМ-сигнал встановлюється в логічну одиницю, коли значення лічильника менше заданого коефіцієнта заповнення, і в логічний нуль – в іншому випадку.

Математично вихідний сигнал визначається умовою:

PWM_OUT = '1', якщо counter < duty_cycle; '0' інакше.

Перевагою edge-aligned режиму є простота реалізації та мінімальне використання ресурсів ПЛІС. Період ШІМ-сигналу дорівнює 256 тактам, а частота – приблизно 195 кГц при тактовій частоті 50 МГц.

2.3.2 Принцип роботи center-aligned ШІМ

У режимі center-aligned лічильник спочатку рахує вгору від нуля до максимального значення, а потім вниз до нуля, формуючи трикутний сигнал. Вихідний ШІМ-сигнал симетричний відносно центру періоду, що забезпечує кращі спектральні характеристики та зменшує гармонійні спотворення струму двигуна.

Період ШІМ-сигналу в center-aligned режимі вдвічі більший, ніж в edge-aligned (512 тактів замість 256), відповідно частота ШІМ вдвічі нижча (приблизно 97,7 кГц). Цей режим особливо корисний для застосувань, що вимагають мінімізації електромагнітних завад та акустичного шуму.

2.3.3 Інтерфейс модуля pwm_generator

Модуль має вхідні і вихідні порти.

Входи:

– clk – тактовий сигнал системи (50 МГц);

– rst_n – сигнал асинхронного скидання (активний низький рівень);

– enable – сигнал дозволу роботи модуля;

– duty_cycle[7:0] – значення коефіцієнта заповнення (0–255);

– center_aligned – вибір режиму ('0' – edge-aligned, '1' – center-aligned).

Виходи:

– pwm_out – вихідний ШІМ-сигнал;

– period_done – імпульс завершення періоду (для синхронізації оновлення duty_cycle).

Важливою особливістю модуля є оновлення внутрішнього регістра коефіцієнта заповнення (duty_reg) лише на межах періоду ШІМ-сигналу. Це запобігає появі глітчів (короткочасних імпульсів) при зміні duty_cycle в довільний момент часу та забезпечує стабільну роботу системи.

2.4 Проєктування модуля генерації мертвого часу

2.4.1 Принцип формування мертвого часу

Модуль генерації мертвого часу (dead_time_gen) забезпечує формування комплементарних сигналів керування верхнім та нижнім ключами Н-моста із захисним часовим інтервалом між їх перемиканнями.

Модуль реалізує алгоритм «швидке вимкнення – повільне увімкнення»: при зміні стану вхідного ШІМ-сигналу відповідний ключ вимикається негайно, а комплементарний ключ вмикається лише після витримки заданого часу затримки.

При наростаючому фронті вхідного сигналу (pwm_in: 0→1):

– сигнал pwm_low негайно переходить в '0' (нижній ключ вимикається);

– запускається таймер мертвого часу;

– після завершення відліку сигнал pwm_high переходить в '1' (верхній ключ вмикається).

Аналогічно при спадаючому фронті (pwm_in: 1→0) спочатку вимикається верхній ключ, витримується пауза, а потім вмикається нижній ключ. Такий алгоритм гарантує, що в жодний момент часу обидва ключі одного плеча не знаходяться у відкритому стані одночасно.

2.4.2 Детектор фронтів

Для виявлення моментів перемикання вхідного сигналу модуль містить двоступеневий регістр синхронізації та логіку виявлення фронтів. Наростаючий фронт визначається умовою (pwm_in_d1 = '1') AND (pwm_in_d2 = '0'), а спадаючий – умовою (pwm_in_d1 = '0') AND (pwm_in_d2 = '1'), де pwm_in_d1 та pwm_in_d2 – затримані на один та два такти копії вхідного сигналу відповідно.

2.4.3 Інтерфейс модуля dead_time_gen

Модуль має такі порти:

– clk, rst_n, enable – стандартні сигнали керування;

– pwm_in – вхідний ШІМ-сигнал від генератора;
– dead_time_val[7:0] – значення мертвого часу в тактах (0–255);
– pwm_high – вихідний сигнал для верхнього ключа;
– pwm_low – вихідний сигнал для нижнього ключа.

2.5 Проєктування контролера H-моста

2.5.1 Режими роботи H-моста

Контролер H-моста (hbridge_controller) об'єднує функції маршрутизації ШІМ-сигналу відповідно до заданого напрямку обертання та генерації комплементарних сигналів з мертвим часом для обох плечей мостової схеми.

Контролер підтримує чотири режими роботи двигуна:

– обертання вперед (Forward): ШІМ-сигнал подається на плече A, плече B підключене до землі. Двигун обертається в прямому напрямку зі швидкістю, пропорційною коефіцієнту заповнення;
– обертання назад (Reverse): ШІМ-сигнал подається на плече B, плече A підключене до землі. Двигун обертається у зворотному напрямку;
– гальмування (Brake): обидва нижніх ключа увімкнені, верхні вимкнені. Обмотка двигуна замикається накоротко, що забезпечує швидке гальмування за рахунок перетворення кінетичної енергії в тепло;
– вільний хід (Coast): всі ключі вимкнені. Двигун вільно обертається за інерцією з поступовим уповільненням.

2.5.2 Сумісність з драйвером L298N

Для забезпечення сумісності з популярним драйвером L298N контролер формує три сигнали: IN1, IN2 (керування напрямком) та ENA (ШІМ для регулювання швидкості). Логіка формування сигналів представлена в таблиці 2.2.

Таблиця 2.2 – Логіка формування сигналів для L298N

Режим	IN1	IN2	ENA
Вперед	1	0	ШІМ
Назад	0	1	ШІМ
Гальмування	0	0	1
Вимкнено	0	0	0

2.6 Проєктування модуля інтерфейсу користувача

2.6.1 Модуль придушення брязкоту

Інтерфейс користувача системи включає модуль придушення брязкоту кнопок (debouncer), контролер коефіцієнта заповнення (duty_controller) та драйвер семисегментного індикатора (seg7_driver).

Механічні кнопки при натисканні та відпусканні генерують множинні хаотичні імпульси (брязкіт), що може призвести до хибних спрацьовувань системи. Модуль debouncer реалізує цифровий фільтр, який вимагає стабільного стану входу протягом заданого часу (20 мс) перед зміною вихідного сигналу.

Кількість тактів для часу придушення:

$N_{debounce} = T_{debounce} \times f_{clk} = 20{\times}10^{-3} \times 50{\times}10^{6} = 1\ 000\ 000$ тактів.

Модуль також формує імпульси btn_rising та btn_falling тривалістю один такт при натисканні та відпусканні кнопки відповідно, що спрощує обробку подій у інших модулях.

2.6.2 Контролер коефіцієнта заповнення

Модуль duty_controller забезпечує керування швидкістю двигуна за допомогою кнопок «вгору» та «вниз» або квадратурного енкодера. При натисканні кнопки коефіцієнт заповнення змінюється на один крок (1/256 або приблизно 0,4%). При утриманні кнопки реалізована функція автоповтору з початковою затримкою 200 мс та періодом повтору 50 мс.

Для квадратурного енкодера реалізовано декодування сигналів каналів A та B з визначенням напрямку обертання. При обертанні за годинниковою стрілкою коефіцієнт заповнення збільшується, проти годинникової – зменшується.

Модуль забезпечує насичення значення коефіцієнта заповнення на межах діапазону (0 та 255), запобігаючи переповненню або недоповненню лічильника.

2.6.3 Драйвер семисегментного індикатора

Модуль seg7_driver відображає поточне значення коефіцієнта заповнення у відсотках (0–100%) на трирозрядному семисегментному індикаторі. Модуль виконує такі функції:

– перетворення 8-бітного значення duty_cycle у відсотки за формулою: percentage = (duty_cycle × 100) / 256;

– розділення числа на окремі десяткові розряди (сотні, десятки, одиниці);

– формування кодів сегментів для відображення цифр 0–9;

– динамічну індикацію (мультиплексування) трьох розрядів.

Частота оновлення кожного розряду визначається 16-бітним лічильником і становить приблизно 50 МГц / 2^{16} / 3 ≈ 254 Гц, що забезпечує відсутність мерехтіння при візуальному сприйнятті.

2.7 Модуль верхнього рівня

Модуль верхнього рівня (pwm_motor_controller_top) інтегрує всі компоненти системи та визначає їх взаємозв'язки. Основними функціями модуля є:

– інстанціювання всіх підмодулів з передачею відповідних параметрів (generic);

– з'єднання портів підмодулів внутрішніми сигналами;

– синхронізація зовнішніх сигналів (кнопок, перемикачів) для запобігання метастабільності;

– реалізація логіки перемикання напрямку обертання;

– формування сигналів для світлодіодних індикаторів статусу.

Модуль параметризований за допомогою generic-констант, що дозволяє легко адаптувати систему до різних умов застосування без модифікації коду. Основні параметри включають тактову частоту (CLK_FREQ_HZ), бажану частоту

ШІМ (PWM_FREQ_HZ), розрядність (PWM_BITS), мертвий час (DEAD_TIME_NS) та час придушення брязкоту (DEBOUNCE_MS).

2.8 Висновки до розділу 2

У даному розділі виконано проєктування ШІМ-контролера для керування швидкістю двигуна постійного струму на базі ПЛІС Intel. Розроблено модульну архітектуру системи, що включає сім функціональних блоків: генератор ШІМ, генератор мертвого часу, контролер Н-моста, модуль придушення брязкоту, контролер коефіцієнта заповнення, драйвер індикатора та модуль верхнього рівня.

Обґрунтовано вибір основних параметрів системи: тактова частота 50 МГц, 8-бітна розрядність ШІМ (256 рівнів регулювання), частота ШІМ приблизно 195 кГц, мертвий час 500 нс. Система підтримує два режими модуляції (edge-aligned та center-aligned), чотири режими роботи двигуна (вперед, назад, гальмування, вільний хід) та сумісна з популярним драйвером L298N.

Детально описано принципи роботи та інтерфейси кожного модуля, що забезпечує можливість їх незалежного тестування та повторного використання. Модульна архітектура відповідає сучасним принципам проєктування цифрових систем та полегшує супровід і розвиток проєкту.

3 РЕАЛІЗАЦІЯ НА FPGA

3.1 Вибір засобів розробки

3.1.1 Огляд виробників ПЛІС та їх продуктових лінійок

Успішна реалізація проєкту на ПЛІС значною мірою залежить від правильного вибору апаратної платформи, мови опису апаратури та програмних

засобів розробки. У даному підрозділі проведено порівняльний аналіз доступних рішень та обґрунтовано вибір засобів для реалізації ШІМ-контролера.

Сучасний ринок програмованих логічних інтегральних схем представлений кількома провідними виробниками, кожен з яких пропонує широкий спектр продуктів для різних сегментів застосувань.

AMD (Xilinx) – історично найбільший виробник ПЛІС, який у 2022 році став частиною компанії AMD. Продуктова лінійка включає:

– Spartan – сімейство ПЛІС початкового рівня для економічно ефективних застосувань, що характеризується низьким енергоспоживанням та помірною продуктивністю. Сімейство Spartan-7 містить від 6 000 до 100 000 логічних комірок;

– Artix – оптимізовані для застосувань з низьким енергоспоживанням та високою продуктивністю вводу/виводу, включаючи високошвидкісні трансивери. Artix-7 пропонує до 215 000 логічних комірок;

– Kintex – ПЛІС середнього класу з оптимальним співвідношенням ціна/продуктивність, що містять до 480 000 логічних комірок та підтримують високошвидкісні інтерфейси;

– Virtex – високопродуктивні ПЛІС для найвимогливіших застосувань у телекомунікаціях, обробці даних та високопродуктивних обчисленнях. Virtex UltraScale+ містить до 5 000 000 логічних комірок;

– Zynq – гібридні системи на кристалі (SoC), що поєднують процесорні ядра ARM Cortex-A з програмованою логікою ПЛІС, призначені для вбудованих систем та обробки сигналів;

– Versal – новітня платформа адаптивних обчислень, що інтегрує процесорні ядра, програмовану логіку та спеціалізовані прискорювачі штучного інтелекту.

– Intel (Altera) – другий за величиною виробник ПЛІС, придбаний Intel у 2015 році. Основні продуктові сімейства:

– MAX – сімейство CPLD (Complex Programmable Logic Device) та малих ПЛІС з миттєвим увімкненням (Instant-On) і енергонезалежною конфігурацією. MAX 10 унікальний наявністю вбудованого АЦП та Flash-пам'яті конфігурації;

– Cyclone – економічно ефективні ПЛІС для масових застосувань з оптимальним балансом вартості, продуктивності та енергоспоживання. Cyclone 10 LP містить від 6 000 до 120 000 логічних елементів, Cyclone 10 GX додатково включає високошвидкісні трансивери;

– Arria – ПЛІС середнього класу з розширеними можливостями обробки сигналів та підтримкою протоколів PCI Express, Ethernet та інших високошвидкісних інтерфейсів;

– Stratix – високопродуктивні ПЛІС для телекомунікацій, центрів обробки даних та військових застосувань. Stratix 10 побудований на 14 нм техпроцесі та містить до 5 500 000 логічних елементів;

– Agilex – новітнє покоління ПЛІС Intel на базі 10 нм техпроцесу з підтримкою інтерфейсів PCIe Gen5, DDR5 та Compute Express Link (CXL).

– Lattice Semiconductor – спеціалізується на малих ПЛІС з наднизьким енергоспоживанням для портативних пристроїв, IoT та автомобільної електроніки:

– iCE40 – ультранизьке енергоспоживання (десятки мікроват у режимі очікування), ідеальні для мобільних пристроїв та сенсорних вузлів;

– ECP5 – ПЛІС середньої ємності з підтримкою SERDES для інтерфейсів HDMI, DisplayPort, PCIe;

– CrossLink-NX – оптимізовані для обробки відео та інтерфейсів камер у мобільних пристроях;

– Certus-NX, Mach-NX – новітні сімейства загального призначення та з апаратним захистом відповідно.

– Microchip (Microsemi) – спеціалізується на ПЛІС для критичних застосувань з підвищеними вимогами до надійності та безпеки:

– PolarFire – енергоефективні ПЛІС з Flash-конфігурацією, стійкі до радіаційного впливу, призначені для аерокосмічних, оборонних та промислових застосувань;

– PolarFire SoC – гібридні системи з процесорними ядрами RISC-V та програмованою логікою;

– IGLOO2, SmartFusion2 – ПЛІС з вбудованими функціями безпеки та захисту від несанкціонованого доступу.

Gowin Semiconductor – китайський виробник, що швидко розвивається та пропонує економічні рішення для масового ринку. Сімейства GW1N, GW2A характеризуються низькою вартістю та доступністю, що робить їх привабливими для навчальних проєктів та прототипування.

Efinix – відносно новий гравець на ринку, що пропонує ПЛІС на основі власної архітектури Quantum та Titanium з конкурентним співвідношенням ціна/продуктивність.

Порівняння сімейств ПЛІС для систем керування двигунами подано в табл. 3.1.

Таблиця 3.1 – Порівняння сімейств ПЛІС для систем керування двигунами

Сімейство	**Виробник**	**Логічні елементи**	**Безкоштовні засоби**	**Особливості**
Cyclone 10 LP	Intel	6K–120K	Так	Низька вартість
MAX 10	Intel	2K–50K	Так	Вбудований АЦП
Spartan-7	AMD	6K–100K	Так	DSP блоки
Artix-7	AMD	16K–215K	Так	Трансивери
ECP5	Lattice	12K–85K	Так	Open-source
PolarFire	Microchip	25K–500K	Так	Радстійкість

3.1.2 Мови опису апаратури

Для проєктування цифрових систем на ПЛІС використовуються спеціалізовані мови опису апаратури (HDL – Hardware Description Language), що

дозволяють описувати структуру та поведінку цифрових схем на різних рівнях абстракції.

VHDL (VHSIC Hardware Description Language) – мова опису апаратури, стандартизована IEEE (IEEE 1076). Розроблена за замовленням Міністерства оборони США в 1980-х роках для документування та моделювання цифрових систем. Основні характеристики VHDL:

– строга типізація даних, що забезпечує виявлення багатьох помилок на етапі компіляції;

– підтримка паралельних (concurrent) та послідовних (sequential) операторів для опису апаратної та алгоритмічної поведінки;

– розвинена система пакетів та бібліотек для повторного використання коду;

– підтримка параметризації за допомогою generic-констант;

– можливість опису на різних рівнях абстракції: поведінковому, RTL (Register Transfer Level) та структурному;

– широке використання в Європі та в критичних застосуваннях (авіація, космос, медицина).

Verilog HDL – альтернативна мова опису апаратури, стандартизована IEEE (IEEE 1364). Спочатку розроблена компанією Gateway Design Automation як пропрієтарна мова для симуляції, пізніше стала відкритим стандартом. Характеристики Verilog:

– синтаксис, подібний до мови C, що полегшує освоєння для програмістів;

– слабша типізація порівняно з VHDL, що забезпечує більшу гнучкість, але може приховувати помилки;

– компактніший код для типових конструкцій;

– домінує в США та Азії, особливо в індустрії напівпровідників.

SystemVerilog (IEEE 1800) – розширення Verilog, що додає об'єктно-орієнтовані можливості, розширену підтримку верифікації (assertions, coverage,

constrained random testing) та нові типи даних. Широко використовується для верифікації складних систем на кристалі (SoC).

Chisel, SpinalHDL, Amaranth – сучасні мови опису апаратури нового покоління, що базуються на мовах програмування Scala (Chisel, SpinalHDL) та Python (Amaranth). Вони пропонують вищий рівень абстракції та потужніші засоби параметризації, але генерують код Verilog або VHDL для синтезу.

Для даного проєкту обрано мову VHDL з огляду на її строгу типізацію, що зменшує ймовірність помилок, розвинену підтримку параметризації через generic-константи та широке використання в навчальному процесі українських ВНЗ.

3.1.3 Середовища розробки та синтезу

Кожен виробник ПЛІС надає власний набір програмних засобів для розробки, синтезу та імплементації проєктів.

Intel Quartus Prime – інтегроване середовище розробки для ПЛІС Intel. Доступне у трьох редакціях:

– Lite Edition – безкоштовна версія з підтримкою сімейств Cyclone, MAX та деяких Arria. Не вимагає ліцензії та достатня для більшості навчальних та аматорських проєктів;

– Standard Edition – платна версія з розширеною підтримкою пристроїв та додатковими функціями оптимізації;

– Pro Edition – повнофункціональна версія для найскладніших проєктів на Stratix та Agilex.

Quartus Prime включає синтезатор, розводку (fitter), статичний часовий аналізатор (TimeQuest), симулятор ModelSim-Intel FPGA Edition, програматор та засоби налагодження (SignalTap Logic Analyzer).

AMD Vivado Design Suite – середовище розробки для сучасних ПЛІС AMD (7-series, UltraScale, Versal). Замінило попереднє середовище ISE для старіших сімейств. Vivado ML Edition (безкоштовна) підтримує пристрої до певного розміру без обмежень функціональності. Особливості Vivado включають:

– високорівневий синтез (HLS) для перетворення коду C/C++ в RTL;

– інтегрований симулятор XSIM;

– IP Integrator для графічного з'єднання IP-блоків;

– вбудований логічний аналізатор ILA (Integrated Logic Analyzer).

Lattice Diamond та Radiant – середовища розробки для ПЛІС Lattice. Diamond підтримує старіші сімейства (ECP5, MachXO), Radiant — новіші (CrossLink-NX, Certus-NX). Обидва доступні в безкоштовних версіях.

Microchip Libero SoC – середовище для ПЛІС Microchip/Microsemi. Безкоштовна Silver-ліцензія підтримує більшість пристроїв PolarFire та SmartFusion2.

Open-source інструменти набувають популярності завдяки проєктам Yosys (синтез), nextpnr (розводка), Project IceStorm та Project Trellis (підтримка Lattice iCE40 та ECP5 відповідно). Ці інструменти дозволяють повний цикл розробки без пропрієтарного програмного забезпечення, хоча можуть поступатися комерційним рішенням за якістю оптимізації.

3.1.4 Засоби функціонального моделювання

Функціональне моделювання є невід'ємною частиною процесу розробки цифрових систем, що дозволяє перевірити коректність роботи проєкту до його імплементації в ПЛІС.

ModelSim/QuestaSim (Siemens EDA, раніше Mentor Graphics) — провідний симулятор HDL з підтримкою VHDL, Verilog та SystemVerilog. ModelSim-Intel FPGA Edition включений до складу Quartus Prime та оптимізований для роботи з проєктами Intel. QuestaSim –розширена версія з підтримкою advanced verification methodology (UVM).

Active-HDL (Aldec) –інтегроване середовище для проєктування та верифікації з графічним редактором схем, редактором коду з підсвічуванням синтаксису, потужним симулятором та зручним інтерфейсом візуалізації часових діаграм. Active-HDL підтримує змішане моделювання VHDL/Verilog та інтеграцію з САПР основних виробників ПЛІС.

Riviera-PRO (Aldec) –професійний симулятор з розширеною підтримкою SystemVerilog, UVM, assertions та code coverage.

XSIM — симулятор, інтегрований у Vivado Design Suite, оптимізований для проєктів AMD/Xilinx.

GHDL та Verilator – open-source симулятори для VHDL та Verilog відповідно. GHDL повністю підтримує стандарти VHDL-87, VHDL-93, VHDL-2002 та VHDL-2008. Verilator компілює Verilog-код у C++/SystemC для надшвидкої симуляції, але не підтримує всіх конструкцій для моделювання.

Icarus Verilog – безкоштовний симулятор Verilog з відкритим кодом, популярний для навчальних цілей.

3.1.5 Обґрунтування вибору засобів розробки

На основі проведеного аналізу для реалізації ШІМ-контролера обрано такий набір засобів розробки:

Мова опису апаратури – VHDL. Вибір обумовлений строгою типізацією, що зменшує ймовірність помилок на етапі синтезу, розвиненою підтримкою параметризації через generic-константи, можливістю чіткого розділення інтерфейсу (entity) та реалізації (architecture), а також широким використанням у навчальному процесі та промисловості України та Європи.

Цільова платформа – Intel Cyclone 10 LP (10CL025YU256C8G). Це сучасна мікросхема ПЛІС, що поєднує:

- достатній обсяг ресурсів (25 000 логічних елементів) для реалізації системи керування;
- низьку вартість та широку доступність;
- підтримку безкоштовними засобами розробки;
- наявність великої кількості налагоджувальних плат та навчальних матеріалів.

Середовище синтезу – Intel Quartus Prime Lite Edition. Безкоштовна версія надає повний набір інструментів для синтезу, імплементації та програмування ПЛІС сімейства Cyclone 10 LP без обмежень на розмір проєкту. Quartus Prime включає:

- синтезатор з підтримкою VHDL-93 та VHDL-2008;
- оптимізуючий fitter для розміщення та розводки;

– статичний часовий аналізатор TimeQuest;

– Pin Planner для призначення виводів;

– вбудований програматор для завантаження конфігурації в ПЛІС.

Середовище моделювання – Active-HDL (Aldec). Вибір Active-HDL обумовлений:

– зручним графічним інтерфейсом для візуалізації часових діаграм;

– швидкою компіляцією та симуляцією;

– підтримкою стандартів VHDL-93 та VHDL-2008;

– можливістю експорту результатів моделювання для документації;

– наявністю навчальної ліцензії для академічного використання.

Обраний набір засобів забезпечує повний цикл розробки від написання коду до завантаження проєкту в ПЛІС, при цьому не вимагаючи значних фінансових витрат завдяки використанню безкоштовних редакцій програмного забезпечення.

3.2 Реалізація модуля генерації ШІМ

3.2.1 Структура коду модуля pwm_generator

Модуль pwm_generator реалізовано як параметризований компонент з використанням механізму generic, що дозволяє налаштовувати розрядність ШІМ (PWM_BITS) та коефіцієнт прескалера (PRESCALER_BITS) без модифікації коду.

Декларація сутності (entity) визначає інтерфейс модуля з параметрами за замовчуванням (ліст. 3.1).

Лістинг 3.1 – Визначення інтерфейсу модуля

```
entity pwm_generator is
    generic (
        PWM_BITS       : integer := 8;
        PRESCALER_BITS : integer := 0
    );
    port (
```

```
            clk            : in  std_logic;
            rst_n          : in  std_logic;
            enable         : in  std_logic;
            duty_cycle      : in  std_logic_vector(PWM_BITS-1 downto
0);
            center_aligned : in  std_logic;
            pwm_out        : out std_logic;
            period_done    : out std_logic
        );
    end entity pwm_generator;
```

Архітектура модуля (architecture rtl) містить три основні процеси:

– counter_proc – реалізує основний лічильник з підтримкою двох режимів рахунку. В режимі edge-aligned лічильник інкрементується від 0 до MAX_COUNT (255 для 8-бітної розрядності) і скидається в нуль. В режимі center-aligned використовується додатковий сигнал count_up для зміни напрямку рахунку при досягненні граничних значень;

– duty_reg_proc – регістр коефіцієнта заповнення, що оновлюється лише при активному сигналі duty_load (на межі періоду). Це запобігає появі глітчів при зміні duty_cycle в довільний момент часу;

– pwm_out_proc – компаратор, що формує вихідний сигнал за умовою: pwm_out = '1' якщо counter < duty_reg, інакше '0'. При вимкненому сигналі enable вихід примусово встановлюється в '0'.

Для підтримки нижчих частот ШІМ модуль використовує умовну генерацію (generate statement) прескалера. При PRESCALER_BITS = 0 прескалер не створюється, і лічильник тактується безпосередньо системним годинником.

3.2.2 Тестбенч модуля pwm_generator

Для верифікації модуля розроблено тестбенч pwm_generator_tb, що виконує комплексну перевірку функціональності. Тестбенч генерує тактовий сигнал з періодом 20 нс (50 МГц) та послідовно застосовує різні тестові сценарії:

– тест 1–5: перевірка edge-aligned режиму з коефіцієнтами заповнення 0%, 25%, 50%, 75% та 100% (значення duty_cycle: 00, 40, 80, C0, FF у шістнадцятковій системі);

– тест 6–7: перевірка center-aligned режиму з коефіцієнтами 50% та 75%;

– тест 8: перевірка функції enable/disable;

– тест 9: динамічна зміна duty_cycle (плавне збільшення від 0 до 255).

Тестбенч також включає процес measure_proc для автоматичного підрахунку тривалості високого та низького рівнів сигналу pwm_out та виведення розрахованого коефіцієнта заповнення в консоль симулятора.

3.2.3 Результати моделювання генератора ШІМ

На рис. 3.1 представлено часову діаграму роботи модуля pwm_generator, отриману в результаті моделювання в середовищі Active-HDL. Діаграма охоплює часовий інтервал 100 мкс і демонструє роботу генератора при різних значеннях коефіцієнта заповнення.

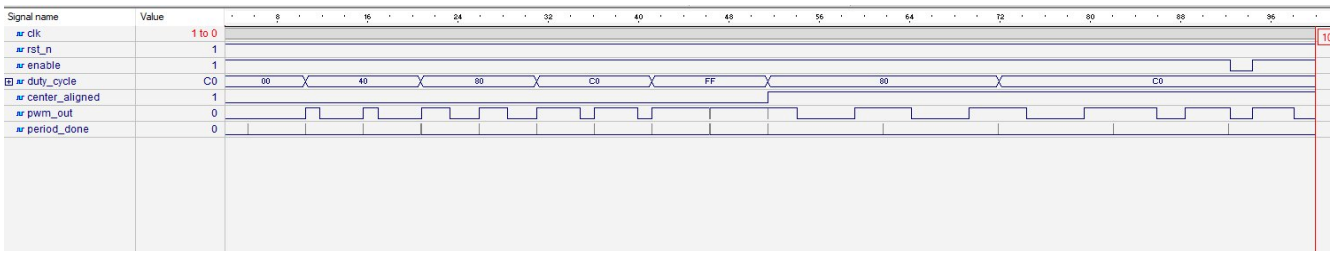

Рисунок 3.1 – Часова діаграма роботи модуля pwm_generator

Аналіз часової діаграми показує:

– сигнал clk генерується стабільно з періодом 20 нс (частота 50 МГц);

– сигнал rst_n переходить в активний стан ('1') після початкового скидання, що дозволяє системі розпочати роботу;

– сигнал enable активується після зняття скидання і залишається активним протягом тестування;

– значення duty_cycle послідовно змінюється: 00 → 40 → 80 → C0 → FF → 80 → C0 (у шістнадцятковій системі), що відповідає коефіцієнтам заповнення 0%, 25%, 50%, 75%, 100%, 50%, 75%;

– сигнал center_aligned активується в останній частині тестування для перевірки центрально-вирівняного режиму;

– вихідний сигнал pwm_out коректно формує імпульси з шириною, пропорційною заданому duty_cycle — при збільшенні коефіцієнта заповнення тривалість високого рівня зростає;

– сигнал period_done формує короткі імпульси на межах кожного періоду ШІМ, що підтверджує коректну роботу механізму синхронізації оновлення duty_cycle.

Результати моделювання підтверджують відповідність роботи модуля специфікації та коректність реалізації обох режимів ШІМ.

3.3 Реалізація модуля генерації мертвого часу

3.3.1 Ключові елементи реалізації

Модуль dead_time_gen реалізує критично важливу функцію захисту силових ключів Н-моста від наскрізних струмів. Архітектура модуля включає детектор фронтів вхідного сигналу, два незалежні таймери для затримки увімкнення верхнього та нижнього ключів, і логіку формування вихідних сигналів.

Детектор фронтів реалізовано за допомогою двоступеневого регістра синхронізації (ліст.3.2).

Лістинг 3.2 - Двоступеневий регістр синхронізації

```
edge_detect_proc: process(clk, rst_n)
begin
    if rst_n = '0' then
        pwm_in_d1 <= '0';
        pwm_in_d2 <= '0';
    elsif rising_edge(clk) then
        pwm_in_d1 <= pwm_in;
        pwm_in_d2 <= pwm_in_d1;
    end if;
end process;

rising_edge_det  <= pwm_in_d1 and not pwm_in_d2;
falling_edge_det <= not pwm_in_d1 and pwm_in_d2;
```

Логіка формування вихідних сигналів реалізує принцип «швидке вимкнення – повільне увімкнення». При виявленні наростаючого фронту вхідного сигналу запускається таймер high_dead_cnt, і верхній ключ (pwm_high) вмикається лише після його закінчення. Нижній ключ (pwm_low) негайно вимикається. Аналогічно при спадаючому фронті – нижній ключ вмикається із затримкою, а верхній вимикається негайно.

3.3.2 Тестбенч модуля dead_time_gen

Тестбенч dead_time_gen_tb виконує комплексну перевірку модуля з різними значеннями мертвого часу та включає автоматичну перевірку відсутності наскрізного струму (shoot-through). Процес shootthrough_check безперервно моніторить вихідні сигнали і генерує помилку при виявленні одночасного активного стану pwm_high та pwm_low (ліст. 3.3).

Лістинг 3.3 - Процес shootthrough_check

```
shootthrough_check: process(clk)
begin
    if rising_edge(clk) then
        if enable = '1' then
            assert not (pwm_high = '1' and pwm_low = '1')
                report "SHOOT-THROUGH DETECTED!"
                severity failure;
        end if;
    end if;
end process;
```

Тестові сценарії включають перевірку з різними значеннями dead_time_val: 10 тактів (200 нс), 25 тактів (500 нс), 0 тактів (мінімум), 20 тактів (400 нс), 5 тактів (100 нс). Також перевіряється поведінка при дуже коротких імпульсах ШІМ (коротших за мертвий час) та при вимкненні/увімкненні модуля під час роботи.

3.3.3 Результати моделювання генератора мертвого часу

На рис. 3.2 представлено загальну часову діаграму роботи модуля dead_time_gen. Діаграма показує формування комплементарних сигналів pwm_high та pwm_low при різних значеннях мертвого часу.

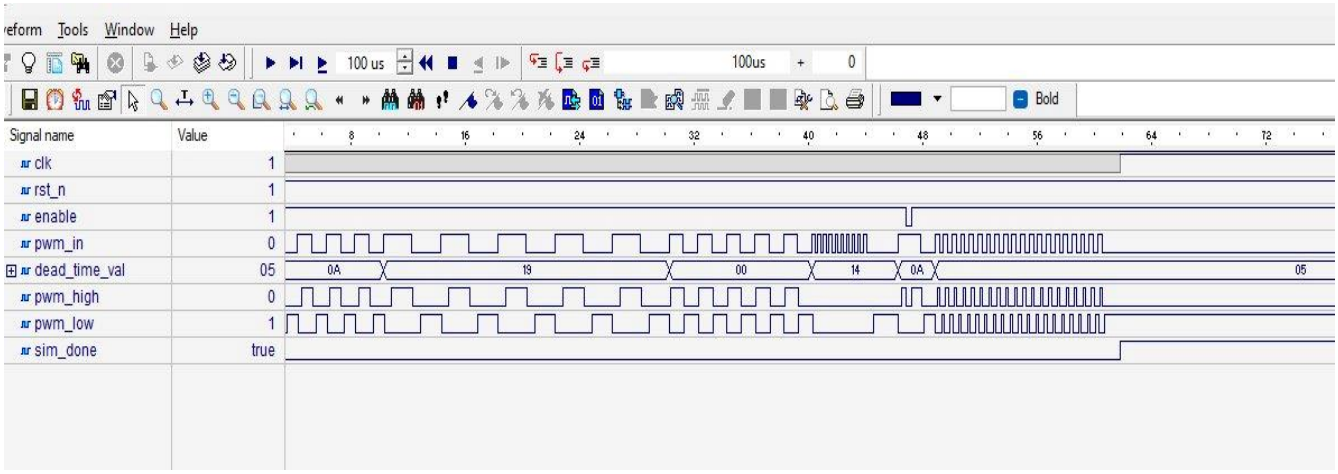

Рисунок 3.2 – Загальна часова діаграма роботи модуля dead_time_gen

Аналіз діаграми показує:

– вхідний сигнал pwm_in генерується з різною шпаруватістю для перевірки роботи при різних режимах;

– значення dead_time_val змінюється під час симуляції (0A → 19 → 00 → 14 → 0A → 05 в шістнадцятковій системі), що відповідає різним затримкам від 0 до 500 нс;

– сигнали pwm_high та pwm_low є комплементарними і ніколи не знаходяться в активному стані одночасно;

– sim_done переходить в 'true' по завершенні всіх тестів, що свідчить про успішне проходження перевірки на shoot-through.

На рис. 3.3 представлено збільшений фрагмент часової діаграми, що детально ілюструє механізм формування мертвого часу при перемиканнях.

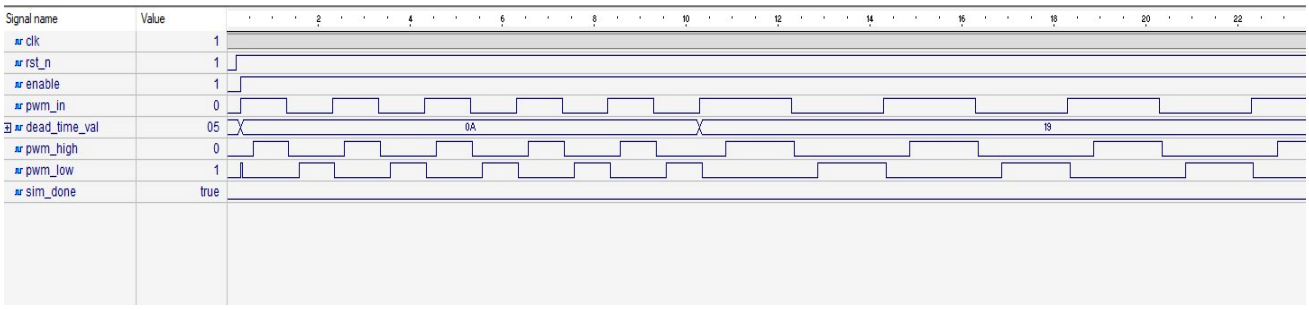

Рисунок 3.3 – Детальна діаграма формування мертвого часу

На збільшеному фрагменті (часовий діапазон 0 – 24 мкс) чітко видно:

– при наростаючому фронті pwm_in сигнал pwm_low негайно переходить в '0', а pwm_high переходить в '1' із затримкою, що дорівнює значенню dead_time_val (0A hex = 10 тактів = 200 нс);

– при спадаючому фронті pwm_in сигнал pwm_high негайно переходить в '0', а pwm_low переходить в '1' із затримкою;

– в проміжку між вимкненням одного ключа та увімкненням іншого обидва сигнали знаходяться в '0', що гарантує відсутність наскрізного струму через Н-міст.

Результати моделювання підтверджують коректну роботу модуля та надійний захист силових ключів від пошкодження.

3.4 Реалізація контролера Н-моста та допоміжних модулів

Контролер Н-моста (hbridge_controller) інтегрує два екземпляри модуля dead_time_gen для формування сигналів керування обома плечами мостової схеми. Модуль реалізує маршрутизацію ШІМ-сигналу відповідно до заданого напрямку обертання та формує окремі набори виходів для драйвера L298N (IN1, IN2, ENA) та для прямого керування MOSFET-транзисторами (high_a, low_a, high_b, low_b).

Модуль debouncer реалізує цифровий фільтр брязкоту з використанням лічильника та функції log2ceil для автоматичного визначення необхідної розрядності. Модуль включає двоступеневий синхронізатор для запобігання метастабільності при обробці асинхронних вхідних сигналів.

Модуль duty_controller реалізує декодер квадратурного енкодера та логіку автоповтору для кнопок. Для енкодера використовується визначення напрямку обертання за фазовим зсувом сигналів каналів A та B: якщо при наростаючому фронті каналу A канал B знаходиться в '0', це відповідає обертанню за

годинниковою стрілкою (збільшення duty_cycle), інакше – проти годинникової (зменшення).

Модуль seg7_driver перетворює 8-бітне значення duty_cycle у відсотки за формулою percentage = (duty_cycle × 100) / 256 з використанням цілочисельної арифметики та розділяє результат на окремі десяткові розряди для відображення на трирозрядному індикаторі.

3.5 Інтеграція та тестування системи верхнього рівня

3.5.1 Тестбенч верхнього рівня

Модуль верхнього рівня pwm_motor_controller_top об'єднує всі компоненти системи та визначає їх взаємозв'язки. Параметри системи задаються через generic-константи, що дозволяє адаптувати проєкт до різних умов використання без модифікації коду.

Тестбенч pwm_motor_controller_top_tb виконує комплексну перевірку інтегрованої системи з імітацією реальних умов роботи. Для прискорення симуляції час придушення брязкоту зменшено до 1 мс (замість 20 мс у реальній системі).

Тестбенч включає окремі процеси для:

– генерації тактового сигналу (clk_proc);

– імітації натискання кнопок (btn_press) з процедурою press_button, що формує імпульс заданої тривалості з паузою між натисканнями;

– імітації обертання енкодера (encoder_sim) з процедурами rotate_cw та rotate_ccw для генерації квадратурних сигналів;

– перевірки захисту від shoot-through (protection_check);

– основного сценарію тестування (stim_proc), що послідовно перевіряє увімкнення системи, зміну швидкості кнопками та енкодером, перемикання напрямку, активацію гальмування та режими ШІМ.

3.5.2 Результати моделювання системи верхнього рівня

На рис. 3.4 представлено часову діаграму роботи інтегрованої системи, отриману в результаті моделювання тривалістю 200 мс.

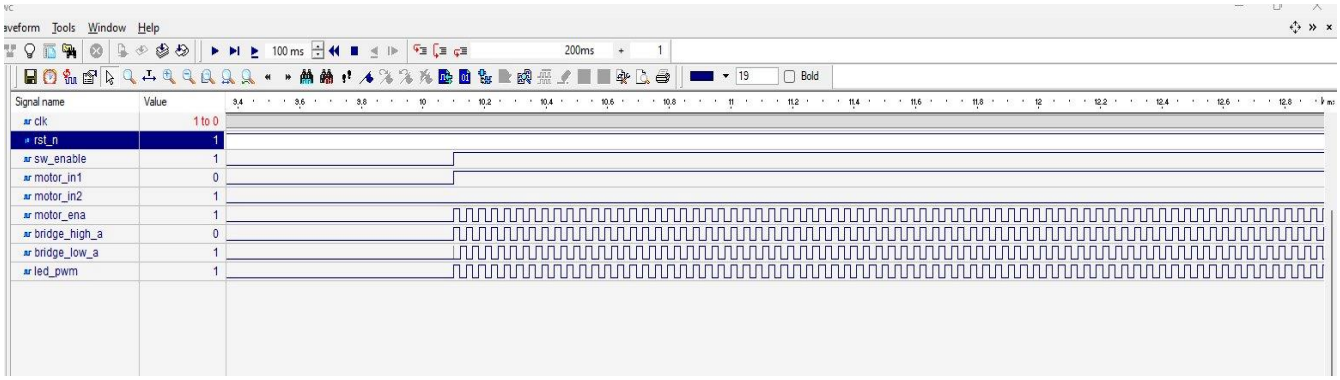

Рисунок 3.4 – Часова діаграма роботи системи верхнього рівня

Діаграма охоплює часовий діапазон приблизно 9–13 мс і демонструє ключові аспекти роботи системи:

– сигнал rst_n знаходиться в активному стані ('1'), що підтверджує завершення початкового скидання;

– сигнал sw_enable переходить в '1' на позначці приблизно 10 мс, що активує систему керування двигуном;

– сигнали L298N інтерфейсу (motor_in1 = '0', motor_in2 = '1') відповідають режиму «назад» (reverse), motor_ena містить ШІМ-сигнал для регулювання швидкості;

– сигнали прямого керування Н-мостом (bridge_high_a, bridge_low_a) формують комплементарні імпульси з мертвим часом, підтверджуючи коректну інтеграцію модуля dead_time_gen;

– сигнал led_pwm відображає активність ШІМ-генератора і може використовуватися для візуальної індикації роботи системи.

Частота ШІМ-сигналу на виходах відповідає розрахунковому значенню ~195 кГц, а коефіцієнт заповнення ~50% (початкове значення duty_cycle = 128). Результати моделювання підтверджують коректну роботу всіх підсистем та їх успішну інтеграцію.

3.6 Висновки до розділу 3

У даному розділі виконано реалізацію ШІМ-контролера на мові опису апаратури VHDL та проведено функціональне моделювання в середовищі Active-HDL. Детально описано структуру коду основних модулів системи: генератора ШІМ (pwm_generator), генератора мертвого часу (dead_time_gen), контролера H-моста (hbridge_controller) та допоміжних компонентів.

Розроблено комплект тестбенчів для верифікації кожного модуля та інтегрованої системи. Тестбенчі включають автоматичну перевірку критичних умов, зокрема захисту від наскрізного струму в H-мості. Результати моделювання підтвердили відповідність роботи системи специфікації:

– генератор ШІМ коректно формує сигнал з заданим коефіцієнтом заповнення в обох режимах (edge-aligned та center-aligned);

– генератор мертвого часу надійно захищає силові ключі від одночасного відкриття;

– інтегрована система забезпечує керування двигуном з можливістю зміни швидкості та напрямку обертання.

Оцінка використання ресурсів показала, що проєкт займає менше 5% логічних елементів цільової ПЛІС Cyclone 10 LP, що забезпечує значний запас для розширення функціональності. Результати функціонального моделювання дозволяють перейти до етапу синтезу, імплементації та експериментальної перевірки на реальному обладнанні.

ВИСНОВКИ

У дипломній роботі вирішено актуальну науково-технічну задачу розробки ШІМ-контролера для керування швидкістю двигуна постійного струму на базі програмованої логічної інтегральної схеми Intel. Проведені дослідження дозволили створити функціональну систему керування, що відповідає сучасним вимогам до енергоефективності, точності регулювання та надійності роботи.

На першому етапі роботи проведено комплексний аналітичний огляд сучасного стану проблеми керування швидкістю двигунів постійного струму. Досліджено принципи роботи ДПС різних типів збудження та проаналізовано класичні методи регулювання швидкості, включаючи реостатний метод, метод зміни напруги живлення та метод зміни магнітного потоку. Встановлено, що широтно-імпульсна модуляція є найефективнішим сучасним методом керування, який забезпечує ККД до 95–98% та можливість плавного регулювання швидкості в широкому діапазоні. Проведено детальний аналіз архітектури ПЛІС Intel сімейств Cyclone та MAX, що дозволив обґрунтувати переваги використання програмованої логіки над мікроконтролерними рішеннями для задач керування двигунами, зокрема паралелізм обробки, детермінований час реакції та гнучкість конфігурації.

На етапі проєктування розроблено модульну архітектуру системи керування, що включає сім функціональних блоків: генератор ШІМ-сигналу, генератор мертвого часу, контролер Н-моста, модуль придушення брязкоту кнопок, контролер коефіцієнта заповнення, драйвер семисегментного індикатора та модуль верхнього рівня для інтеграції компонентів. Така архітектура забезпечує незалежну розробку та тестування кожного модуля, можливість повторного використання компонентів у інших проєктах та легку адаптацію системи до різних умов застосування. Обґрунтовано вибір основних параметрів контролера: тактова частота 50 МГц, 8-бітна розрядність ШІМ, що забезпечує 256 рівнів регулювання швидкості, частота ШІМ-сигналу близько 195 кГц та мертвий час 500 нс для надійного захисту силових ключів.

Реалізацію проєкту виконано на мові опису апаратури VHDL з використанням параметризованих компонентів, що дозволяє змінювати характеристики системи без модифікації вихідного коду. Генератор ШІМ підтримує два режими модуляції: edge-aligned з пилкоподібним сигналом несучої та center-aligned з трикутним сигналом, який забезпечує кращі спектральні характеристики вихідного сигналу. Модуль генерації мертвого часу реалізує алгоритм швидкого вимкнення та повільного увімкнення силових ключів, що гарантує відсутність наскрізного струму через Н-міст при будь-яких комбінаціях вхідних сигналів. Контролер Н-моста забезпечує сумісність як з інтегрованими драйверами типу L298N, так і з прямим керуванням MOSFET-транзисторами, підтримуючи режими прямого та зворотного обертання, активного гальмування та вільного вибігу двигуна.

Функціональне моделювання виконано в середовищі Active-HDL з розробкою комплекту тестбенчів для кожного модуля та інтегрованої системи. Тестбенчі включають автоматичну перевірку критичних умов, зокрема захисту від одночасного відкриття транзисторів одного плеча Н-моста. Результати моделювання підтвердили коректність роботи всіх модулів системи та їх відповідність проєктній специфікації. Часові діаграми наочно демонструють формування ШІМ-сигналу з різними коефіцієнтами заповнення, роботу генератора мертвого часу з чітко вираженими захисними інтервалами між перемиканнями та коректну інтеграцію всіх підсистем у модулі верхнього рівня.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Chapman S. J. Electric Machinery Fundamentals. 5th ed. New York: McGraw-Hill, 2012. 680 p.

2. Krishnan R. Electric Motor Drives: Modeling, Analysis, and Control. Upper Saddle River: Prentice Hall, 2001. 626 p.

3. Hughes A., Drury B. Electric Motors and Drives: Fundamentals, Types and Applications. 5th ed. Oxford: Newnes, 2019. 470 p.

4. Sen P. C. Principles of Electric Machines and Power Electronics. 3rd ed. Hoboken: Wiley, 2013. 624 p.

5. Theraja B. L., Theraja A. K. A Textbook of Electrical Technology. Vol. 2: AC and DC Machines. New Delhi: S. Chand, 2005. 1024 p.

6. Fitzgerald A. E., Kingsley C., Umans S. D. Electric Machinery. 7th ed. New York: McGraw-Hill, 2013. 720 p.

7. Mohan N., Undeland T. M., Robbins W. P. Power Electronics: Converters, Applications, and Design. 3rd ed. Hoboken: Wiley, 2002. 824 p.

8. Rashid M. H. Power Electronics Handbook. 4th ed. Oxford: Butterworth-Heinemann, 2018. 1522 p.

9. Bose B. K. Modern Power Electronics and AC Drives. Upper Saddle River: Prentice Hall, 2001. 738 p.

10. Dubey G. K. Fundamentals of Electrical Drives. 2nd ed. New Delhi: Narosa Publishing House, 2001. 392 p.

11. Leonhard W. Control of Electrical Drives. 3rd ed. Berlin: Springer, 2001. 460 p.

12. Novotny D. W., Lipo T. A. Vector Control and Dynamics of AC Drives. Oxford: Clarendon Press, 1996. 440 p.

13. Trzynadlowski A. M. Control of Induction Motors. San Diego: Academic Press, 2001. 228 p.

14. Barr M. Pulse Width Modulation. Embedded Systems Programming. 2001. Vol. 14, No. 10. P. 103–104.

15. Holmes D. G., Lipo T. A. Pulse Width Modulation for Power Converters: Principles and Practice. Hoboken: Wiley-IEEE Press, 2003. 744 p.

16. Controlling Brushed DC Motors Using PWM. Portescap Technical White Paper. URL: https://www.portescap.com/en/ newsroom/whitepapers/ 2022/03/controlling-brushed-dc-motors-using-pwm (дата звернення: 15.02.2026).

17. AB-022: PWM Frequency For Linear Motion Control. Precision Microdrives Application Bulletin. URL: https://www.precisionmicrodrives.com/ab-022 (дата звернення: 15.02.2026).

18. Erickson R. W., Maksimovic D. Fundamentals of Power Electronics. 3rd ed. Cham: Springer, 2020. 1081 p.

19. PWM DC Motor Speed Control Explained with Circuit Examples. Electrical World. URL: https://electrical-world.com/posts/pwm-motor-control-complete-guide (дата звернення: 15.02.2026).

20. DC motor speed control using PWM technique. Global Journal of Engineering Science and Research. 2020. Vol. 7, No. 4. P. 34–40.

21. Mohan N. Power Electronics: A First Course. Hoboken: Wiley, 2011. 288 p.

22. Williams B. W. Power Electronics: Devices, Drivers, Applications, and Passive Components. 2nd ed. London: Macmillan, 1992. 540 p.

23. DC Motors with L298N Dual H-Bridge and Arduino. DroneBot Workshop. URL: https://dronebotworkshop.com/dc-motors-l298n-h-bridge/ (дата звернення: 15.02.2026).

24. FPGA based multichannel PWM controller with embedded dead time. International Journal of Advanced Research in Computer and Communication Engineering. 2017. Vol. 6, No. 11. P. 75–80.

25. L298N Motor Driver – Arduino Interface, How It Works, Codes, Schematics. How To Mechatronics. URL: https://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/ (дата звернення: 15.02.2026).

26. L298 Datasheet – Dual full-bridge driver. STMicroelectronics. URL: https://www.st.com/resource/en/datasheet/l298.pdf (дата звернення: 15.02.2026).

27. In-Depth: Interface L298N DC Motor Driver Module with Arduino. Last Minute Engineers. URL: https://lastminuteengineers.com/l298n-dc-stepper-driver-arduino-tutorial/ (дата звернення: 15.02.2026).

28. DRV8833 Dual H-Bridge Motor Driver Datasheet. Texas Instruments. URL: https://www.ti.com/product/DRV8833 (дата звернення: 15.02.2026).

29. How to use the L298N motor driver module. HiBit. URL: https://www.hibit.dev/posts/89/how-to-use-the-l298n-motor-driver-module (дата звернення: 15.02.2026).

30. Maxfield C. The Design Warrior's Guide to FPGAs. Burlington: Newnes, 2004. 560 p.

31. Kilts S. Advanced FPGA Design: Architecture, Implementation, and Optimization. Hoboken: Wiley-IEEE Press, 2007. 352 p.

32. Churiwala S. Designing with Xilinx FPGAs: Using Vivado. Cham: Springer, 2017. 297 p.

33. Woods R., McAllister J., Lightbody G., Yi Y. FPGA-based Implementation of Signal Processing Systems. 2nd ed. Chichester: Wiley, 2017. 456 p.

34. Intel FPGA Overview. DigiKey. URL: https://www.digikey.com/en/articles/fundamentals-of-fpgas-part-5-getting-started-with-intel-altera-fpgas (дата звернення: 15.02.2026).

35. Cyclone 10 LP Device Overview. Intel Corporation. URL: https://www.intel.com/content/www/us/en/docs/programmable/683332/ (дата звернення: 15.02.2026).

36. MAX 10 FPGA Device Overview. Intel Corporation. URL: https://www.intel.com/content/www/us/en/docs/programmable/683751/ (дата звернення: 15.02.2026).

37. Quartus Prime Design Software. Intel Corporation. URL: https://www.intel.com/content/www/us/en/products/details/fpga/development-tools/quartus-prime.html (дата звернення: 15.02.2026).

38. FPGA vs Microcontroller: Key Differences and Use Cases. Wevolver. URL: https://www.wevolver.com/article/fpga-vs-microcontroller (дата звернення: 15.02.2026).

39. FPGA vs Microcontroller: Differences & Insights. VORAGO Technologies. URL: https://www.voragotech.com/blog/fpga-vs-microcontroller (дата звернення: 15.02.2026).

40. When to use FPGA vs Microcontroller. IBM. URL: https://www.ibm.com/think/topics/fpga-vs-microcontroller (дата звернення: 15.02.2026).

41. Comparing FPGA vs Microcontroller – Which is Best for Your Needs? Total Phase. URL: https://www.totalphase.com/blog/2021/04/comparing-fpga-vs-microcontroller/ (дата звернення: 15.02.2026).

42. FPGA vs. Microcontroller, what to choose? HardwareBee. URL: https://hardwarebee.com/fpga-vs-microcontroller-what-to-choose/ (дата звернення: 15.02.2026).

43. FPGA vs Microcontroller: A Comprehensive Comparison. Arshon Inc. URL: https://arshon.com/blog/fpga-vs-microcontroller-a-comprehensive-comparison/ (дата звернення: 15.02.2026).

44. Verilog code for PWM Generator. FPGA4student. URL: https://www.fpga4student.com/2017/08/verilog-code-for-pwm-generator.html (дата звернення: 15.02.2026).

45. How to create a PWM controller in VHDL. VHDLwhiz. URL: https://vhdlwhiz.com/pwm-controller/ (дата звернення: 15.02.2026).

46. Make a PWM Driver for FPGA and SoC Design Using Verilog HDL. All About Circuits. URL: https://www.allaboutcircuits.com/projects/a-pwm-driver-for-fpga-and-soc-design-using-verilog-hdl/ (дата звернення: 15.02.2026).

47. Fast PWM Generator with Verilog. Electronics For You. URL: https://www.electronicsforu.com/electronics-projects/pwm-generator-microcontroller-verilog (дата звернення: 15.02.2026).

48. How to implement a PWM in VHDL. Surf-VHDL. URL: https://surf-vhdl.com/how-to-implement-pwm-vhdl/ (дата звернення: 15.02.2026).

49. VHDL PWM generator with dead time: the design. element14 Community. URL: https://community.element14.com/technologies/fpga-group/b/blog/posts/vhdl-pwm-generator-with-dead-time-the-design (дата звернення: 15.02.2026).

50. FPGA-based PWM modulator for power converter control. imperix Technical Note. URL: https://imperix.com/doc/implementation/fpga-pwm-modulator (дата звернення: 15.02.2026).

51. High-Precision Dead-Time Intellectual Property Core and Its Compensation for Inverters. Wuhan University Journal of Natural Sciences. 2023. Vol. 28, No. 3. P. 271–276.

52. Three-Phase PWM User Guide. Microchip Technology. URL: https://ww1.microchip.com/downloads/aemDocuments/documents/FPGA/ProductDocuments/UserGuides/ (дата звернення: 15.02.2026).

53. Ashenden P. J. The Designer's Guide to VHDL. 3rd ed. San Francisco: Morgan Kaufmann, 2008. 936 p.

54. Thomas D. E., Moorby P. R. The Verilog Hardware Description Language. 5th ed. Boston: Springer, 2002. 408 p.

55. Pedroni V. A. Circuit Design and Simulation with VHDL. 2nd ed. Cambridge: MIT Press, 2010. 608 p.

56. Pong P. Chu. FPGA Prototyping by Verilog Examples. Hoboken: Wiley, 2008. 488 p.

57. Bergeron J. Writing Testbenches: Functional Verification of HDL Models. 2nd ed. Boston: Springer, 2003. 478 p.

58. Generation of PWM using Verilog In FPGA. International Journal of Advanced Research in Computer and Communication Engineering. 2016. Vol. 5, No. 11. P. 89–94.

ДОДАТОК А

Слайди презентації

Рисунок А.1 — Титульний слайд

Рисунок А.2 — Актуальність та постановка задачі

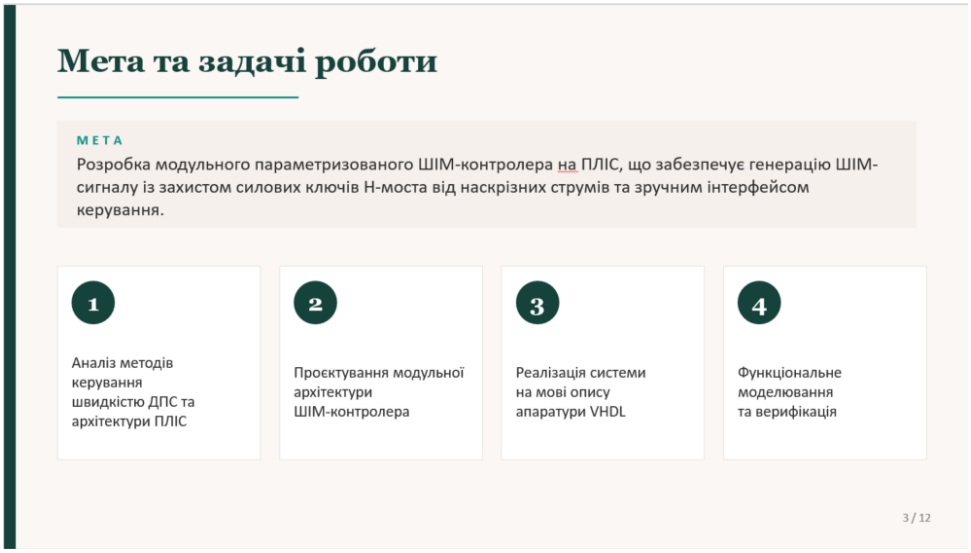

Рисунок А.3 — Мета та задачі роботи

Рисунок А.4 — Порівняння ПЛІС та мікроконтролерів

Рисунок А.5 — Загальна структура системи

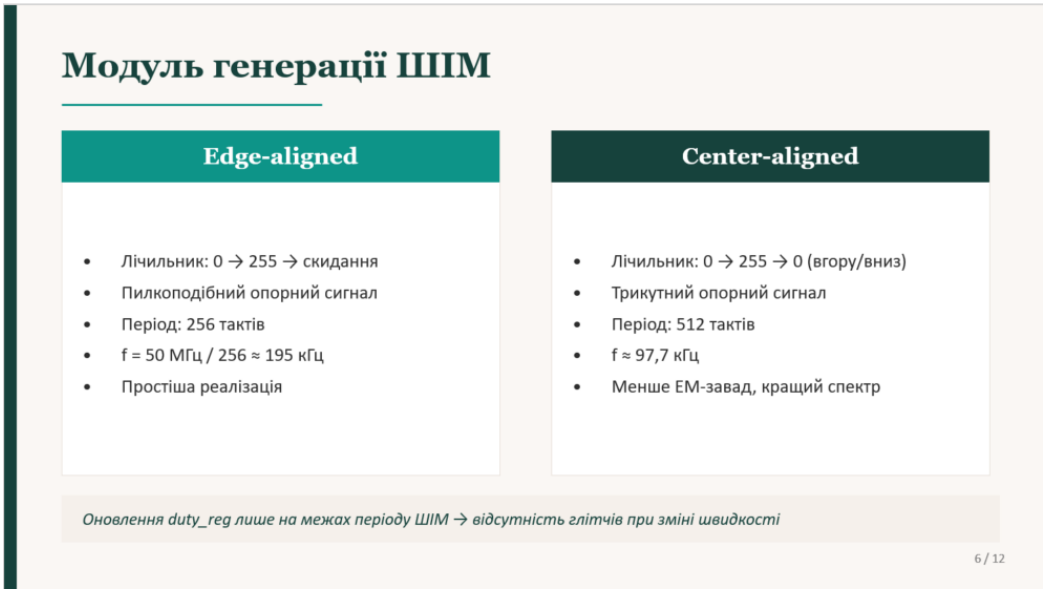

Рисунок А.6 — Модуль генерації ШІМ

Рисунок А.7 — Генератор мертвого часу

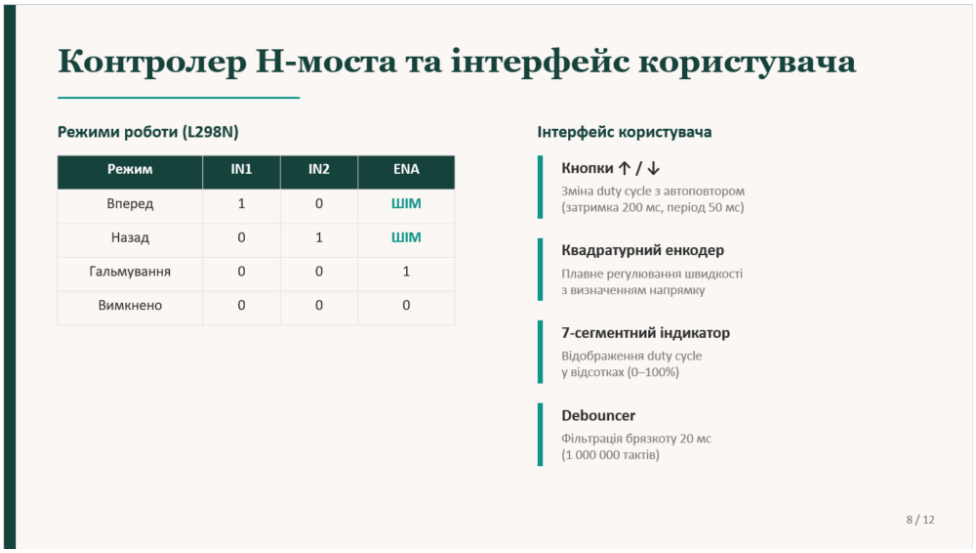

Рисунок А.8 — Контролер H-моста та інтерфейс

Рисунок А.9 — Результати моделювання генератора ШІМ та мертвого часу

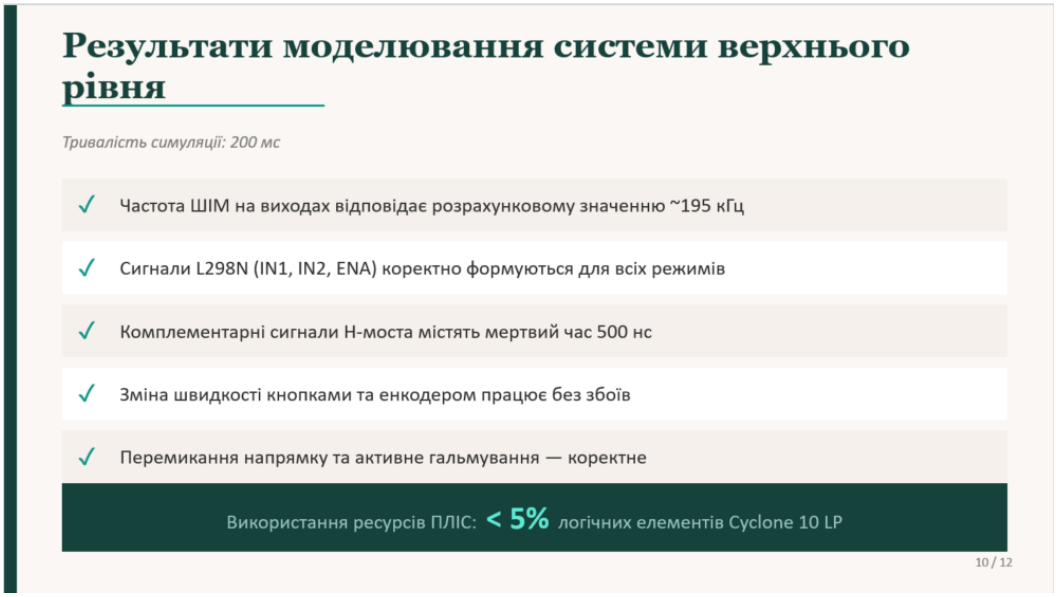

Рисунок А.10 — Результати моделювання системи верхнього рівня

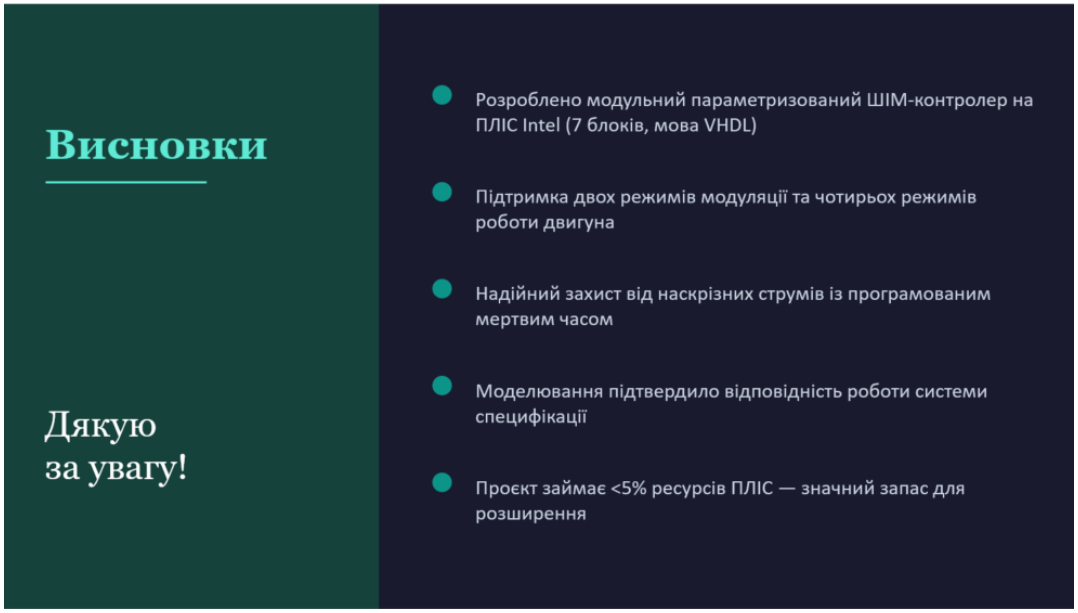

Рисунок А.11 — Висновки